УНИВЕРЗИТЕТ У КРАГУЈЕВЦУ
Факултет техничких наука у Чачку

Небојша Љ. Станковић

ФАКТОРИ УЧЕЊА И ПРЕДВИЂАЊЕ УСПЕШНОСТИ У ПРОГРАМИРАЊУ ПРИМЕНОМ ВЕШТАЧКИХ НЕУРОНСКИХ МРЕЖА

Докторска дисертација

Чачак, 2021. године

UNIVERSITY OF KRAGUJEVAC
Faculty of Technical Sciences Čačak

Nebojša Lj. Stanković

FACTORS OF LEARNING AND PREDICTING SUCCESS IN PROGRAMMING USING ARTIFICIAL NEURAL NETWORKS

Doctoral Dissertation

Čačak, 2021

Аутор
Име и презиме: Небојша Љ. Станковић
Датум и место рођења: 08. 08. 1966. године, Гњилане
Садашње запослење: Асистент, Факултет техничких наука у Чачку, Универзитет у Крагујевцу
Докторска дисертација
Наслов: Фактори учења и предвиђање успешности у програмирању применом вештачких неуронских мрежа
Број страница: 133
Број слика: 93
Број табела: 124
Број библиографских података: 153
Установа и место где је рад израђен: Универзитет у Крагујевцу, Факултет техничких наука у Чачку
Научна област (УДК): (004.43+004.032.36):37] (043.3)
Ментор: др Марија Благојевић, ванредни професор, Универзитет у Крагујевцу, Факултет техничких наука у Чачку
Оцена и одбрана
Датум пријаве теме: 03. 07. 2019. године
Број одлуке и датум прихватања теме докторске/уметничке дисертације: Број одлуке: IV-04-987/12 Датум: 11. 12. 2019. године
Комисија за оцену научне заснованости теме и испуњености услова кандидата: 1. др Дијана Каруовић, ванредни професор, Универзитет у Новом Саду, Технички факултет „Михајло Пупин“ у Зрењанину, ужа научна област: Информационе технологије, председник 2. др Вања Луковић, ванредни професор, Универзитет у Крагујевцу, Факултет техничких наука у Чачку, ужа научна област: Рачунарска техника, члан 3. др Марија Благојевић, ванредни професор, Универзитет у Крагујевцу, Факултет техничких наука у Чачку, ужа научна област: Информационе технологије и системи, ментор
Комисија за оцену и одбрану докторске/уметничке дисертације: 1. др Дијана Каруовић, ванредни професор, Универзитет у Новом Саду, Технички факултет „Михајло Пупин“ у Зрењанину, ужа научна област: Информационе технологије, председник 2. др Вања Луковић, ванредни професор, Универзитет у Крагујевцу, Факултет техничких наука у Чачку, ужа научна област: Рачунарска техника, члан 3. др Милош Папић, ванредни професор, Универзитет у Крагујевцу, Факултет техничких наука у Чачку, ужа научна област: Менаџмент информациони системи, члан
Датум одбране дисертације:

Фактори учења и предвиђање успешности у програмирању применом вештачких неуронских мрежа

Резиме

Академско образовање је једна од кључних области у процесу модернизације земље. Способност предвиђања успеха помаже наставницима да идентификују студенте који имају потенцијал да похађају напредне курсеве, као и студенте којима је потребно допунско образовање. У модерном друштву вештине програмирања постају све важније. Многа истраживања показују да је програмирање једна од критичних вештина технолошке писмености студената. Због тога постоји потреба за анализама велике количине података на основу којих се могу предвидети фактори који утичу на успешност студената у области програмирања. Последњих година, примена вештачке интелигенције у образовању значајно је порасла у целом свету. Вештачке неуронске мреже (ВНМ), као један од њених алата, доживљавају бројне успешне имплементације.

У докторској дисертацији *Фактори учења и предвиђање успешности у програмирању применом вештачких неуронских мрежа* представљен је модел ВНМ развијен у сврху предвиђања успеха студената у стицању програмерских знања и вештина.

Анализирано је 180 студената студијског програма Информационе технологије са Факултета техничких наука у Чачку. За сваког студента су прикупљени подаци о претходном образовању. Успех студената у учењу програмирања мерен је кроз постигнућа на тесту знања, и сврстан је у три категорије: *неуспешан*, *средње успешан* и *веома успешан*. За предикцију успеха студената коришћен је модел *трослојне ВНМ базиране на алгоритму учења са простирањем грешке уназад (backpropagation)*.

Креирано је 19 модела. Модел са најбољом предиктивном тачношћу (*90,7%*) искоришћен је као коначан модел за имплементацију. За тај модел је креирана веб апликација помоћу које наставник има могућност прилагођавања наставе, те ефикасније организације исте, што доводи до успешно савладаног градива.

Кључне речи: Предвиђање успешности, Програмирање, Тест знања, Вештачка интелигенција, Вештачке неуронске мреже, веб апликација

Factors of learning and predicting success in programming using artificial neural networks

Abstract

Academic education is one of the key areas in the process of modernization of a country. The ability to predict success helps teachers identify students who have the potential to attend advanced courses, as well as students who need additional education. In modern society programming skills are becoming increasingly important. Many studies show that programming is one of the critical skills of students' technological literacy. Therefore, there is a need to analyze a large amount of data on the basis of which factors that affect student performance in the field of programming can be predicted. In recent years, the application of artificial intelligence in education has increased significantly worldwide. Artificial neural networks (ANN), as one of its tools, are experiencing numerous successful implementations.

In the doctoral dissertation *Factors of learning and predicting success in programming using artificial neural networks*, the ANN model developed for the purpose of predicting the success of students in acquiring programming knowledge and skills is presented.

180 students of the study program Information Technology from the Faculty of Technical Sciences in Čačak were analyzed. Data on previous education were collected for each student. Students' success in learning programming is measured through achievements on the knowledge test and is classified into three categories: *unsuccessful*, *moderately successful* and *very successful*. A *three-layer ANN model based on a backpropagation learning algorithm* was used to predict student success.

19 models were created. The model with the best predictive accuracy (*90,7%*) was used as the final model for implementation. A web application was created for that model, with the help of which the teacher has the possibility of adapting the teaching, and more efficient organization of the same, which leads to successfully mastered material.

Key words: Success prediction, Programming, Knowledge test, Artificial intelligence, Artificial neural networks, Web application

САДРЖАЈ

1. УВОД

Убрзани развој технологије и даље утиче на корените промене економске и социјалне структуре. Неопходне су сложене промене које треба да резултују повећањем ефикасности, продуктивности и економског раста. Многобројна занимања нестају, већина занимања се аутоматизује, јавља се потреба за мултидисциплинарним кадром. Модерно друштво захтева специјалисте који су креативни и спремни на сарадњу са другим људима и рад у новим системима (посебно системима вештачке интелигенције).

Научно предвиђање је кључна карика у ланцу оптималне контроле било којег сложеног система. Проблеми прогнозирања, дугорочног планирања и оптималног управљања сада су важнији него икад. Предвиђање је неопходно да би се остварило оптимално управљање, при чему комбинација предвиђања и управљања треба да буде стално у жижи менаџерских одлука у било којој сфери активности [1].

Образовни систем је постао један од најважнијих фактора који доприносе развоју људских ресурса и повећању људског капитала. За повећање ефикасности образовног система неопходно је разумети потребе и потражњу тржишта за стручњацима, не само у садашњем тренутку, већ и дугорочно [2].

Учинак студената је важан део високошколских установа и предвиђање успешности студената, односно предвиђање њихових перформанси, постала је једна од активности бројних високошколских образовних субјеката. За боље разумевање, побољшане образовне перформансе и процену успешности студирања, потребно је из образовне базе извући корисне податке и идентификовати факторе који утичу на успешност студирања, па самим тим и развити алат за дијагнозу потенцијалног постигнућа студената [3].

Тренутно се предлаже много техника за анализу и процењивање успешности студената. Вештачка неуронска мрежа, која опонаша људски мозак, једна је од најбољих техника која може да реши ову врсту проблема.

Појавом првих личних (персоналних) рачунара (*PC*) програмирање је постало веома популарно. Први програми били су уско повезани са аритметиком и математиком, програмирање се предавало на универзитетима и у специјализованим средњим школама. Данас је најједноставнији паметни телефон моћнији од првих персоналних рачунара. Од малена деца су укључена у рачунарско окружење и комуницирају са рачунаром. Програмирање не треба да буде вештина коју је потребно савладати, већ средство којим развијамо способности ученика, што је основа за успех у даљем учењу.

Занимање програмера је постао један од кључних ресурса данашњег модерног друштва. Све већа потражња друштва за већом продуктивношћу може се задовољавати образовањем и обучавањем програмера да постану високо квалификовани. Интересовање средњошколаца и студената за програмске језике и програмирање је веома велико. Томе у прилог иду бројне студије, као и све већи броја матураната који се пријављују на студијске програме који студентима омогућавају стицање вештина програмирања. Држава Србија препознала је важност ових вештина и омогућила акредитацију наставног плана и програма у 2017. години са повећањем броја уписаних студената на студијске програме који изучавају информационе технологије.

Учење програмирања је једна од критичних вештина технолошке писмености студената. Претходно знање студената и учење различитих програмских језика током студија утиче на њихову перцепцију решавања „софтверских“ проблема. Потребно је анализирати знање стечено током различитих година студирања како би се утврдио утицај учења

различитих програмских језика на решавање проблема. Како би се унапредила настава програмирања и повећала успешност студената у програмирању, неопходно је применити различите ИТ алате и технологије за предвиђање понашања студената (са посебним освртом на примену техника вештачких неуронских мрежа за предвиђање).

Истраживања ове докторске дисертације проистекло је из анализе приступа који проучавају факторе успешног/неуспешног учења и специфичности развоја рачунарског размишљања и програмирања, с једне стране, и трагања за методологијом предвиђања успешности студената у програмирању која је заснована на савременом напретку у развоју техника дата мајнинга, са друге стране.

Постојећа решења и модели предвиђања постигнућа студената нису довољно прилагођени специфичностима области информационих технологија, нити универзално за ову област, нити узимајући у обзир специфичности посебних области информационих технологија као што су програмирање, рачунарске мреже, заштита података, заштита мрежа, базе података... Поред тога, крајњи корисници који немају посебне ИТ вештине везане за вештачке неуронске мреже морају бити у могућности да одреде улазне параметре и да добију резултат за предвиђени параметар (у овом истраживању то је успешност у програмирању).

Да би се креирао адаптивни систем (вештачка неуронска мрежа) за предвиђање успешности студента у програмирању, потребно је [4]:

- идентификовати факторе који утичу на перформансе студената;
- претворити ове факторе у форме погодне за кодирање адаптивног система;
- моделирати вештачку неуронску мрежу која се може користити за предвиђање успешности студената у програмирању на основу унапред одређених података за датог студента.

Главни циљеви истраживања докторске дисертације су:

- утврђивање фактора учења за предмете у области програмирања;
- усвајања знања и вештина из програмирања;
- креирање и евалуација модела неуронских мрежа како би се предвидела успешност у учењу програмирања;
- израда веб апликације за рад са креираним моделом вештачких неуронских мрежа;
- систематизација знања о учењу програмирања и предвиђању успешности студената помоћу вештачких неуронских мрежа.

Докторска дисертација такође има практичну сврху (циљ): Прилагођавање наставе (учења) у складу с предвиђеним вредностима параметра „успешност у програмирању“.

2. ТЕОРИЈСКИ ОКВИР

2.1. Трендови, вештине и знање

2.1.1. Кључни трендови будућности

Свет је све комплекснији и нестабилнији, наша будућност је неизвесна. Неопходно је креирати такве системе/моделе који ће минимизирати ризике и који ће омогућити будућим генерацијама одређену извесност. Потребно је претворити наш тренутни систем у систем који је трајно иновативан и прилагодљив – омогућити прихватање нових начина размишљања, нових идеја, нових технологија, развијати нове вештине.

Идентификовано је седам кључних трендова који чине економију будућности, шест распоређених у 3 категорије (технолошки, технолошко-друштвени и друштвени) и један општи метатренд (слика 2.1.1 [5]).

Слика 2.1.1: Кључни трендови који одређују избор радног места у XXI веку (Извор: [5])

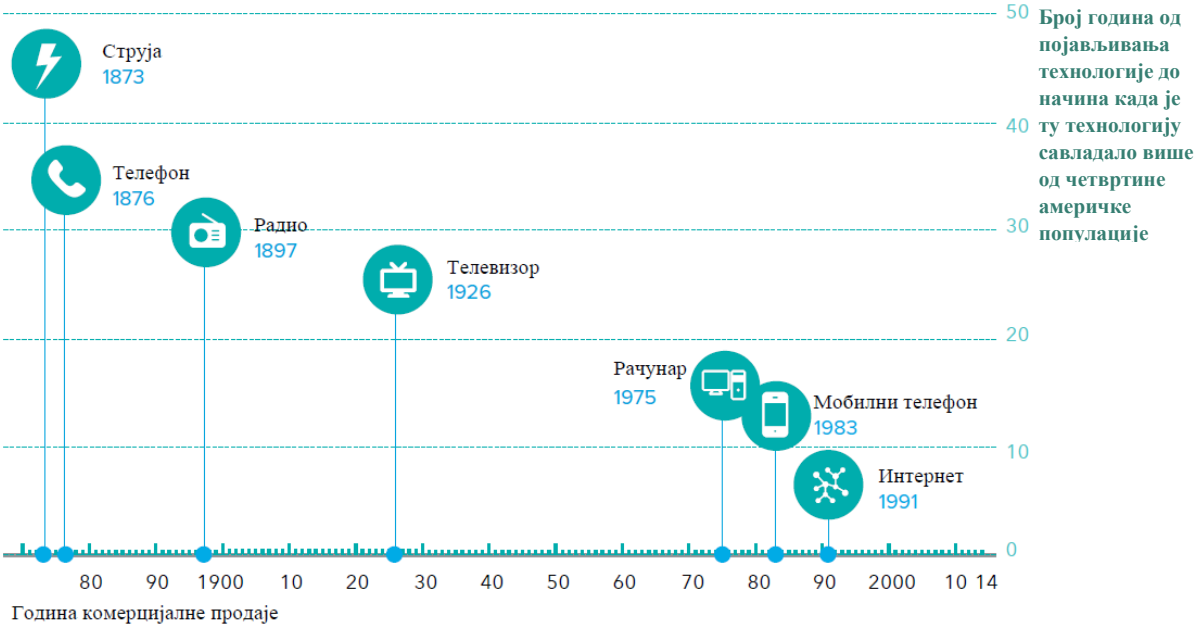
- **Технолошки трендови**
  1. *Дигитализација свих сфера живота*. Дигитализација података је све заступљенија, интернет све приступачнији, развијају се нове области људске активности. *„Још увек не разумемо у потпуности шта значи живети у дигиталном свету. То ће тек разумети генерације „дигиталних урођеника" (digital natives) – они који су рођени и одрасли у „свету бројева““* [6].

2. *Аутоматизација и роботизација*. Аутономни системи постали су уобичајена појава у свим секторима економије, роботика је створила револуцију у производњи.

- **Технолошко-друштвени трендови**
  3. *Глобализација* (економска, технолошка и културна). Један од најизраженијих трендова данашње економије је глобализација (производни ланци, роба широке потрошње, научна сазнања...); транснационална сарадња је све заступљенија.
  4. *Екологизација* (озелењавање). Озелењавање економије, озелењавање образовања, стварање и очување „зелених радних места“ су кључни фактори одрживог развоја.

- **Друштвени трендови**
  5. *Демографске промене*. Ново друштвено окружење дефинисано је повећањем очекиваног трајања живота, сталне урбанизације, растућом улогом жена у економији...; ове промене у великој мери утичу на мењање пословног окружења и пословања у савременом свету.
  6. *Формирање мрежног друштва*. Развој нових мрежних технологија и ширење решења базираних на *blockchain* технологији омогућавају ефикаснији начин управљања фирмама *(blockchain технологија, нов начин документовања на интернету, омогућила је да се дигитална информација дистрибуира, уместо копира, чиме је створена основа за нову верзију интернета „Интернет 3.0“)*.

- **Метатренд**
  7. *Убрзавање технолошких и друштвених промена*. Нова технолошка решења и друштвене праксе се појављују брже, темпо обнове околног света (окружења) је све јачи. Ове промене се дешавају под утицајем све веће стопе промена (убрзање). Када се упореди брзина ширења нових технологија у XX и XXI веку постаје јасно видљиво убрзање стопе технолошког раста (слика 2.1.2 [5]) Ако је за дистрибуцију класичног телефона од његовог оснивања било потребно преко 30 година, код мобилних телефона, у развијеним земљама, тај период је мало више од 10 година.



Слика 2.1.2: Масовна употреба изума (Извор: [5])

2.1.2. Вештине и образовање

Рачунар и технологија умрежавања рачунара, једни су од најважнијих технолошких изума човечанства. Тренд дигитализације је у сразмери са експоненцијалним растом интернета. Интернет је замишљен као мрежа рачунара, данас је то мрежа различитих уређаја, од рачунара, паметних телефона и сатова до аутомобила, семафора, робота, дрона и аутоматизоване индустријске машине. Нова технологија *Интернет интелигентних уређаја*, развијањем „дигиталне интелигенције“, омогућила је ефикасно повезивање дигиталног и физичког света.

Општа дигитализација довела је до ере велике количине података, отварају се нове могућности за развој уређаја који самостално решавају сложене проблеме (вештачка интелигеница – ВИ). Како би се брзо анализирала огромна количина података и извршило њихово структуирање неопходан је рад у хибридним системима (суперсистеми), који укључује тимове људи и системе засноване на вештачкој интелигенцији.

Некада су научници, да би добили нове информације, морали да уложе огромне напоре. Данас се дошло у доба вишка и преоптерећености информацијама (што може произвести и „затрпавање“ информацијама). Захваљујући дигитализацији и глобализацији, обим доступних информација се брзо увећава, човек се све теже носи са филтрирањем и анализом свих тих информација [7].

Да би стручњаци постали/остали експерти у својој области неопходно је да буду у току са објављеним студијама у тој области. У данашње време, они би требали да прочитају 30-40 чланака недељно, што је једноставно немогуће [8].

Због убрзаног темпа технолошких и друштвених промена, човек ће морати, више пута у току свог живота, да учи и да мења нове области активности. Биће приморан да стекне:

- вештине и знања која омогућавају рад са новим технологијама;
- вештине и општа знања која могу да се примене у широком спектру професионалних, друштвених и личних контекста.

Према наводима Федорове [9] „*Вештина је активност која се формира понављањем и довођењем у аутоматизам. Развијање вештине је процес који се постиже извођењем*“.

Образовање је постало кључни елемент свих стратегија, оно треба да омогући развој способности за прилагођавање захтевима новог модерног друштва. Неопходно је постојећи приступ учењу ревидирати (иновирати) и усмерити на функционална/примењена знања и вештине.

Основни задатак који се постављао пред образовањем у двадесетом веку (век описмењавања) био је масовна едукација људи основним вештинама (да читају, броје и пишу). Радници су, кроз специјализовано образовање, стицали вештине потребне за одређену професију. Стечене вештине су биле довољне за обављање задатака који се нису много мењали током времена, већина радника је, у току свог радног века, користила своје вештине на истом уређају и нису имали потребу за додатним образовањем/усавршавањем.

Али, за рад у новом, комплексном, свету те вештине више нису довољне. У 21. веку (век дигитализације) образовање добија и задатак – стицање нових вештина (**„4К“ вештине**): *комуникација* (*Communication*), *креативност* (*Creativity and Innovation*), *критичко размишљање* (*Critical Thinking and Problem Solving*) и *тимски рад* (*Collaboration*) [10]. Са циљем стварања целовитог образовања за 21. век, у току су корените реформе образовања. У Европи је водећа земља Финска [11], док је у Азији Сингапур [12].

У Србији се реформа основног и средњег образовања спроводи независно од реформе високог образовања. Реформа основног и средњег образовања спроводи се централистички, док се реформа високог образовања спроводи од стране самих високошколских установа. Године 2012. влада Републике Србије усвојила је „Стратегију развоја образовања у Србији до 2020. године“ [13]. Један од кључних делова тог плана је стратегија развоја основног и средњошколског образовања и васпитања. Нова стратегија образовног система требала је да осигура развој сваке особе, друштва и државе засноване на знању.

Кључни проблем, тренутног образовног система у Србији, је тај што се још увек инсистира на пасивном усвајању теоријских знања, а не на њиховој практичној примени. Од тога, колико су школе спремне да напусте традиционално образовање и примене нове мере образовне политике (мере које воде ка вишем степену знања и вештина ученика и студената потребних за учешће на дигиталном тржишту рада) зависи и одговор на питање „*Да ли образовање у Србији испуњава своју сврху?*“. Резултати *PISA* тестова (*Programme for International Student Assessment* – Међународни програм процене образовних постигнућа ученика) потврђују низак ниво писмености ученика у Србији [14].

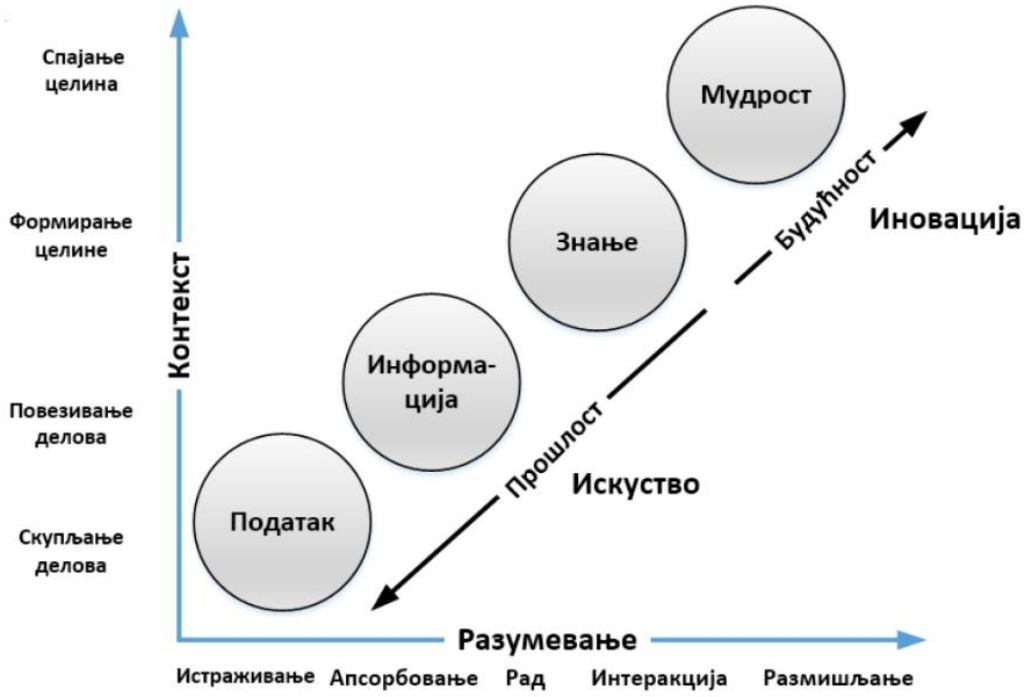
При планирању, реализацији и евалуацији система образовања пажњу треба усмерити на оно што је суштинско за ученике/студенте – шта они могу да ураде на крају процеса образовања, тј. обратити пажњу на компетенције и исходе (жељене резултате), који су изражени као индивидуална постигнућа [15]. Висока стопа незапослености младих у Европи, која је често проузрокована неусклађеношћу вештина, још више је нагласила потребе за побољшањем квалитета и важности вештина и компетенција које млади стичу након школовања.

Образовање треба да буде фокусирано не само на пренос знања и развој вештина, већ и на пуну подршку развоју особе као пуноправног аутора свог живота. У захтевима послодаваца, поред поседовања „**меких**“ и „**тврдих**“ вештина, неопходно је поседовање и „**дигиталних**“ вештина, као и комбинацију истих [9]. „*Меке вештине*“ су особине личности које омогућавају човеку да постигне успех без обзира на специфичности активности и правац у коме ради (лидерство, комуникација или управљање временом). „*Тврде вештине*“ су повезане се специфичним техничким знањем и обуком, тј. техничке вештине везане за активности које се обављају у подручју формализованих технологија: канцеларијски рад, логистика, вожња, програмирање, итд. „*Дигиталне вештине*“ су широк спектар вештина које су повезане са потпуном компјутеризацијом и дигитализацијом.

У Србији се 2019. године почело са израдом „Стратегије развоја дигиталних вештина у Србији за период 2019-2023“ [16]. Циљ ове стратегије је развој дигиталних вештина свих грађана кроз повећање рачунарске писмености, повећање употребе рачунара и интернета, примена дигиталних вештина и компетенција на свим нивоима образовања, подизање компетенција наставника.

2.1.3. Знање и ИТ

Знање није информација. Знање се састоји од интуиције, искуства, вештина, учења и скупа идеја. Знање омогућава стварање новог знања. Да би се донеле исправне одлуке неопходно је знање. Ток знања се може приказати кроз 4 аспеката знања – **ПИЗМ** (*податак, информација, знање, мудрост*), његовог разумевања и контекста употребе (слика 2.1.3 [17]). „*Знање информише, мудрост трансформише*“ [18].



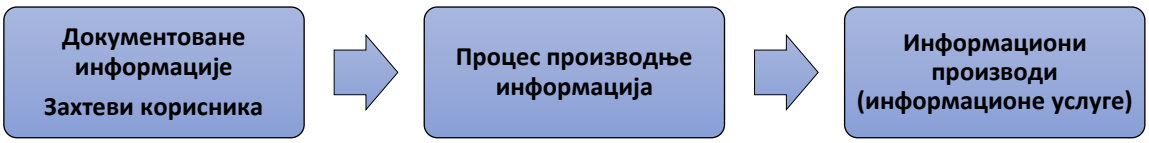
Слика 2.1.3: Први поглед на ПИЗМ хијерархију – линеарни ланац (Извор: [17])

Повезаност ова 4 аспекта следи из њихове хијерархије. Сваки наредни аспект представља претходни аспект који је проширен додатним факторима (*контекст, разумевање* и *примена*). *Податак* (који настаје путем истраживања, стварања, прикупљања и открића) је чињеница без контекста, и ако му се придружи контекст, он постаје *информација*, тј. претвара се у информације. Ако се информација, која је статична, усмери ка разумевању (расуђивању), она постаје *знање*, које је динамично. Када се знање, које се гради кроз искуство, често примењује, оно постаје *мудрост*, мудрост је крајњи ниво разумевања.

Како би савремени човек успешно примењивао стечена знања у решавању задатака свакодневног, „дигиталног“, живота, неопходно је да добро познаје информационе технологије (ИТ) и да располаже способностима (професионалним компетенцијама) [19]:

- пројектовање основних и примењених ИТ;
- развој средстава за имплементацију ИТ;
- обрада добијених резултата рада.

Коришћење рачунара, савремених средстава обраде и преноса информација, представља почетак нове фазе, назване информатизација (компјутеризација). Прикупљање, обрада, организовање, чување, претраживање, дистрибуција, коришћење информација представљају доминантне активности нових „дигиталних“ генерација. Производња информација, као специфичне врсте робе, данас у савременој економији представља најразвијенију и најпрофитабилнију производну грану.



Слика 2.1.4: Општа шема производње информација (Извор: аутор)

Циљеви ИТ су:

- *Основни*: формирање висококвалитетног ресурса (нове информације, знања) неопходног за побољшање ефикасности система у којем делује;
- *Специфични циљ*: ефикасна производња информационих производа и рационално коришћење информационих ресурса у процесу задовољавања информационих потреба корисника.

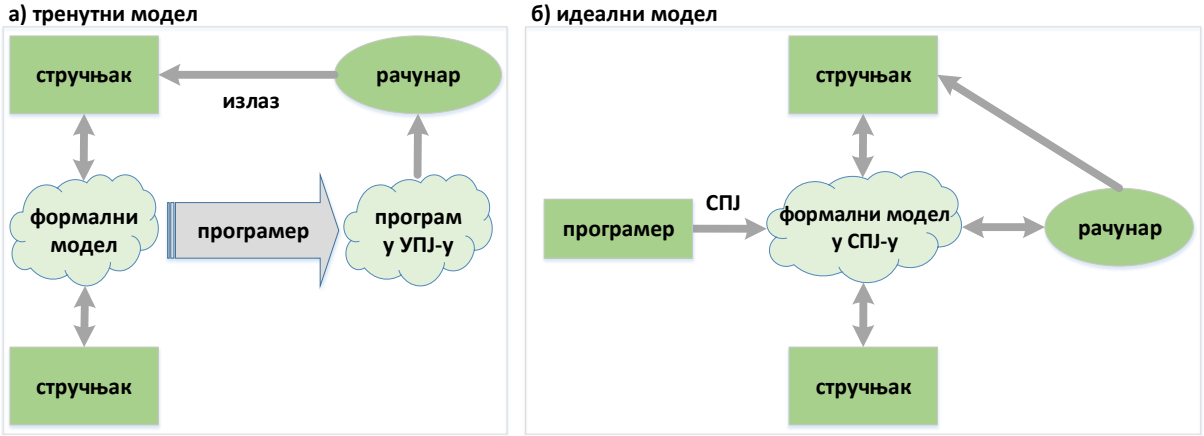
2.2. Програмирање и програмски језици

2.2.1. Значај програмирања/програмских језика

Софтвер је програм који прецизира радње које рачунар мора да изврши како би решио одређени задатак или групу задатака. Представљен је у програмском језику (*рачунарски програм*) или у машинском коду (*машински програм*). Најопштије, софтвер је подељен на *основни софтвер* и апликације (*примењени софтвер*). Програмски језици, оперативни системи (ОС), сервисни алати и услужни програми представљају основни софтвер, док посебни софтвер укључује пакете апликација који су креирани за решавање специфичних проблема, као што су управљање, превођење, дизајнирање, итд.

Бити успешан научник, инжењер, економиста, бизнисмен,... у данашњем дигиталном свету, захтева коришћење рачунарског моделовања, што подразумева истовремено познавање одређених научних модела и програмских језика. Програмерске вештине представљају данас кључну компетенцију у скоро свим сферама људског деловања. Програмски језици егзистирају преко седамдесет година и основни механизми њихових примена остали су исти (нису се мењали).

Да би се проблем у некој области решио неопходна је идеја коју треба трансформисати у одговарајући формални модел. Особа, која поседује способност трансформације идеје у формални модел је *стручњак* (научник, истраживач, привредник,...) у тој области, и она жели да овај модел примени као ИТ решење или да провери његову валидност помоћу рачунарске симулације. Због тога, стручњак контактира другу особу/тим која има способност да комуницира са рачунарима помоћу програмских језика (*програмер*). Програмер трансформише модел у изворни код неког *универзалног програмског језика (УПЈ)* и компајлер (преводилац) трансформише изворни код у извршни рачунарски програм (слика 2.2.1 а [20])).



Слика 2.2.1: Тренутни и идеалан модел сарадње са рачунарским системом (Извор: [20])

Главни недостак овог модела је тај што стручњак не разуме програмске језике, а програмер се не сналази са моделима баш најбоље. Због тога је потребан посредник који тумачи (преводи) модел на одређени програмски језик програмера. Решења овог проблема је да се од свих ових особа „направи“ *врхунски програмер*. Овај циљ се може остварити на два начина:

- изучавањем универзалних програмских језика опште намене у средњим школама и на универзитетима. Ови језици пружају одличну подршку за све врсте рачунарских програма – системско програмирање, нумеричко рачунање, обрада података, апликације крајњег корисника, пословна логика, рад у релном времену, итд. Ово решење је популарно, али нажалост само мали део стручњака је потпуно упознат са универзалним језицима за програмирање.
- прилагођавањем програмских језика формалним моделима, односно изучавање *специфичних програмских језика (СПЈ)* који су уско специјализовани за конкретан домен проблема (слика 2.2.1 б [20])

Стицање и развијање знања о програмирању је веома сложен процес. Пред програмерима је изазов превазилажења широког спектра потешкоћа. Приликом избора уводног програмског језика у обзир треба узети много битних фактора – доступност компајлера, квалитет уџбеника, трошкове успостављања и одржавања, популарност језика, итд. [21].

Већина програмера користи свега неколико програмских језика (најчешће су ограничени на коришћење једног или два) јер обично раде на пројектима који користе неки специфични језик (*Java, C, Python...*). Не треба се ограничавати на површно испитивање могућности програмског језика, већ покушати да се разумеју његови концепти развоја и како они утичу на његову имплементацију (примену).

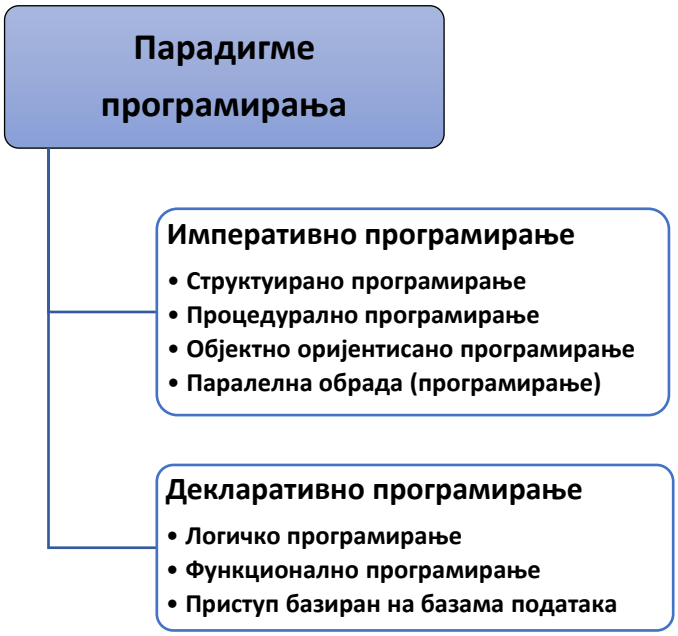
Разлози због којих треба приступити учењу различитих језика [22]:

- *Развијање ефикаснијих алгоритама*. Креирање метода програмирања које најбоље задовољавају нове услове захтева прилично дубоко разумевање природе програмских језика.
- *Ефикасније коришћење програмских језика*. Програмер пише ефикасније програме јер упознаје имплементацију тих нових услова на језику који најчешће користи.
- *Допуњавање корисничког дизајна софтвера*. Проучавајући дизајне који су доступни на различитим језицима, програмер проширује свој речник.
- *Избор програмског језика који је најпогоднији за имплементацију* одређеног пројекта. Познавање главних карактеристика језика, његових предности и недостатака даје програмеру шири избор и смањује количину посла.
- *Лакше учење нових језика*. Темељно познавање конструкција и метода имплементације бројних програмских језика омогућава програмеру да брже научи нови језик ако је то потребно.
- *Лакше развијање нових језика*. Програмери корисничког интерфејса за велики програм суочавају се са многим проблемима који се сусрећу у развоју програмских језика опште намене. Овај аспект развоја софтвера обично се поједностављује ако је програмер упознат са низом различитих конструкција и метода за увођење обичних програмских језика.

Учење програмских језика је више од брзог упознавања са њиховим способностима/могућностима.

2.2.2. Парадигме програмирања

Парадигма програмирања је начин класификације програмских језика у складу са њиховим стилом програмирања и карактеристикама које они пружају. Иако већина програмских језика припада једном типу парадигме (метод решавања проблема), неки језици показују елементе који се односе на различите парадигме (слика 2.2.2 [23]). Постоји неколико карактеристика које одређују програмску парадигму као што су: модуларност, објекти, прекиди или догађаји, ток контроле, итд.



Слика 2.2.2: Парадигме програмирања (Извор: [23])

Императивно програмирање је најстарија парадигма и још увек се користи. Има блиску везу са архитектуром машине. Заснива се на *Von Neumann*-овој архитектури. Она се углавном фокусира на кораке које треба урадити и ради на логици „*Прво уради ово, а затим уради то*“. Она дефинише редослед изјава којима се операције морају одвијати. Предности императивног програмирања су: једноставност имплементације; садржи петље, променљиве,... Недостаци овог програмирања су: сложени проблеми се тешко решавају; мања ефикасност и продуктивност.

Императивно програмирање је подељено у неколико широких категорија парадигми: структуирано програмирање, процедурално програмирање, објектно оријентисано програмирање, паралелно програмирање (слика 2.2.2 [23]).

- **Структуирано програмирање**, познато као и „класично“ императивно програмирање. Код овог програмирања ток контроле се дефинише угњежденим петљама, условима и потпрограмима, а не преко *GOTO* (што доводи до сложеног и замршеног кода). Променљиве су углавном локалне за блокове. Језици који наглашавају структуирано програмирање су: *Algol, PL/I, Pascal, C, C++, C#, Ada, Modula, PHP, Java, Perl, Ruby*.
- **Процедурално програмирање** је изведено из структуираног програмирања. Процедурално програмирање се заснива на процедуралним позивима. Свака изјава процедуралног језика је или позив на процедуру или додела података. Неки популарни процедурални програмски језици су: *C, Pascal, BASIC, Fortran*.

- Парадигма **објектно оријентисаног програмирања** је широко примењена парадигма програмирања – заснива се на концепту објеката – објекти су стварни свет. Све што је присутно око нас је објекат. Сваки објекат има два важна атрибута: својства (*data*) и понашање (*function*). Популарни објектно оријентисани програмски језици су: *Simula-67, Java, C++, C#, Objective C, Smalltalk, Python, Ruby*.
- **Паралелна обрада** је обрада програмских инструкција са више процесора. Систем паралелне обраде поседује велики број процесора са циљем да се програм подели на мања времена. Примери језика који имају могућност паралелне обраде су: *NESL* (један од најстаријих), као и *C/C++*.

Декларативно програмирање развија структуру и елементе рачунарског програма указујући на рачунање и/или логику без тока контроле. Фокус је на ономе *шта треба да се уради, а не на томе како то треба да се уради*. То је основна разлика између императивних (како то учинити) и декларативних (шта да раде) парадигми програмирања. Парадигма декларативног програмирања је подељена на: логичко програмирање, функционално програмирање и програмирање базирано на базама података.

- **Логичко програмирање** се користи у развоју вештачке интелигенције. У логичком програмирању главни нагласак је на бази знања и проблему. Рачунару се не дефинише начин на који ће решити проблем, већ се сам проблем детаљно описује. На основу базе знања коју корисник познаје, и уз базу података која се прослеђује рачунару, рачунар даје резултате. Пример оваквог језика је *Prolog*.
- **Функционално програмирање** представља мало другачији приступ програмирању, код кога се избегава коришћење променљивих, а цео програм се дефинише као комплексна функција. Парадигма функционалног програмирања има своје корене у математици и она је независна од програмског језика. Ово програмирање користи комбинацију позива функције за покретање тока програма при чему резултат функције постаје улаз у другу функцију. Примери програмских језика који користе ову парадигму су: *Python, JavaScript, Scala, Lisp, Clojure, Haskell*.
- **Програмирање засновано на базама података** обрађују датотеке података чији садржај узрокује да програм уради нешто другачије. Ово програмирање је срце пословног информационог система и омогућава креирање датотека, унос података, ажурирање, креирање упита и извештавања. Пример језика који користи ово програмирање је *SQL*.

Улога програмских парадигми у развоју програмских језика и било којег софтверског дизајна је очигледна. И поред тога што су се многе парадигме програмирања развиле, софтверска индустрија активно користи само неколико парадигми програмирања. Софтвер има велику прилику за иновације и повећану ефикасност. Пред дизајнерима софтвера је да изаберу одговарајуће концепте парадигме програмирања и њихове програмске језике, да истраже и пронађу сваки могући аспект, слој и архитектуру како би добили што ефикаснији софтвер у складу са хардвером [24].

2.2.3. Историја програмских језика

Године 1823. *Ada Lovelace*, пионирка рачунарске науке, креирала је алгоритам за аналитички мотор (машина *Charles Babbage*-а) који је користио физички покрет (промена брзине за извршавање калкулација). Призната је као први програмер на свету, иако никада није написала нити једну линију кода какву данас разумемо. Њена шема за машински језик постала је основни оквир за програмирање.

Америичка влада 1942. године уместо коришћења механичких зупчаника (као што је *Babbage*-ов уређај) развија електронски нумерички интегратор и рачунар (*ENIAC*) где је физички покрет замењен електричним сигналима. Први чисто електрични универзални рачунар *ENIAC* користио је форму програмирања у процесу који је могао трајати и до неколико недеља [25].

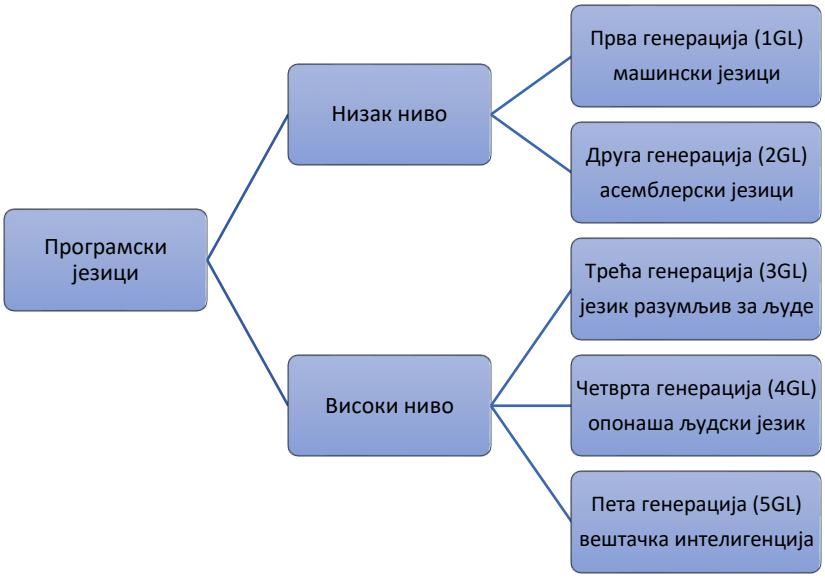
Године 1945. *John Von Neumann* је развио **два важна концепта** који су директно утицали на даљи развој рачунарских програмских језика. **Први концепт** је познат као *shared program technique* (*техника дељеног програма*). Овај концепт је био револуционаран у томе што је програмерима омогућио да дизајнирају разнолику палету програма, а да не морају ни на који начин да мењају хардверске компоненте. **Други концепт** који је предложио је *conditional control transfer* (*условни управљачки пренос*), који је прецизирао потребна средства помоћу којих може да се развије флексибилност програмског језика. Ова идеја је довела до појма потпрограма, или малих блокова кôда који се могу прескочити у било ком редоследу, уместо једног сета хронолошки одређених корака које рачунар треба да предузме [25].

До данас је написано више стотина програмских језика, сваки за неку специфичну сврху. У табели 2.2.1 [26] дат је хронолошки приказ програмских језика који су се развили у 20. веку и почетком 21. века.

Табела 2.2.1: Значајни програмски језици који су се развили у 20. веку и почетком 21. века (Извор: [26])

• 1951 – Assembly language (Асемблер) • 1952 – Autocode (асемблерски језик за IBM 1440) • 1954 – IPL (претходник LISP-а) • 1955 – FLOW-MATIC (довео је до COBOL-а) • 1957 – FORTRAN (први компајлер) • 1957 COMTRAN (претходник COBOL-а) • 1958 – LISP • 1958 – ALGOL 58 • 1959 – FACT (претходник COBOL-а) • 1959 – COBOL • 1959 – RPG • 1962 – APL • 1962 – Simula • 1962 – SNOBOL • 1963 – CPL (претходник C-а) • 1964 – Speakeasy (рачунарска околина) • 1964 – BASIC • 1964 – PL/I • 1966 – JOSS • 1967 – BCPL (претходник C-а) • 1967 – BCPL (претходник B-а)	• 1968 – Logo • 1969 – B (претходник C-а) • 1970 – Pascal • 1970 – Forth • 1972 – C • 1972 – Smalltalk • 1972 – Prolog • 1973 – ML • 1975 – Scheme • 1978 – SQL (језик упита, касније проширен) • 1980 – C++ (као C са класама, преименован 1983. године) • 1983 – Ada • 1984 – Common Lisp • 1984 – MATLAB • 1984 – dBase III, dBase III Plus (Clipper и FoxPro као FoxBASE, касније развијен у Visual FoxPro) • 1985 – Eiffel • 1986 – Objective-C • 1986 – LabVIEW (Visual Programming Language) • 1986 – Erlang • 1987 – Perl • 1988 – Tcl • 1988 – Wolfram Language (као део Mathematica, име је добила у јуну 2013. године)	• 1989 – FL (Backus) • 1990 – Haskell • 1991 – Python • 1991 – Visual Basic • 1993 – Lua • 1993 – R • 1994 – CLOS (део ANSI Common Lisp) • 1995 – Ruby • 1995 – Ada 95 • 1995 – Java • 1995 – Delphi (Object Pascal) • 1995 – JavaScript • 1995 – PHP • 1996 – VBScript • 1997 – Rebol • 1998 – Standard C++ • 2000 – ActionScript • 2000 – C# • 2001 – D • 2001 – Visual Basic .NET • 2002 – Scratch • 2003 – Groovy • 2003 – C++03 • 2003 – Scala • 2004 – FreeBASIC • 2005 - Fantom • 2005 – F#	• 2006 - Cobra • 2006 – PowerShell • 2007 – Ada 2005 • 2007 – QB64 • 2007 – Clojure • 2008 - Genie • 2009 – Go • 2010 – Rust • 2011 – Dart • 2011 – C++11 • 2011 – Kotlin • 2011 – Red • 2011 – Elixir • 2012 – TypeScript • 2012 – Julia • 2012 – Ada 2012 • 2014 – Hack • 2014 – Swift • 2014 – C++14 • 2015 – Raku • 2016 – Ring • 2016 – Raku • 2017 – C++17 • 2017 – Balerina • 2018 – Fortran 2018 • 2019 – Bosque • 2020 – Citrine

У основи, програмски језици су класификовани у две главне категорије – **језици ниског нивоа** и **језици високог нивоа** (слика 2.2.3). Први програмски језици који су настали (*језици прве генерације – 1GL*) су језици на машинском нивоу (развијани су помоћу машинског кода). Потом су се развијали програмски језици који су за представљање инструкција машинског језика користили асемблерске језике (*језици друге генерације – 2GL*). Ова два језика (машински и асемблерски) представљају језике ниског нивоа *(low-level languages*). Након језика ниског нивоа, развијају се програмских језици на високом нивоу (*high-level languages*) који су били лакши за употребу од асемблерског језика и машинског кода. Они су класификовани у *језике треће (3GL), четврте (4GL)* и *пете генерације (5GL)* [27].

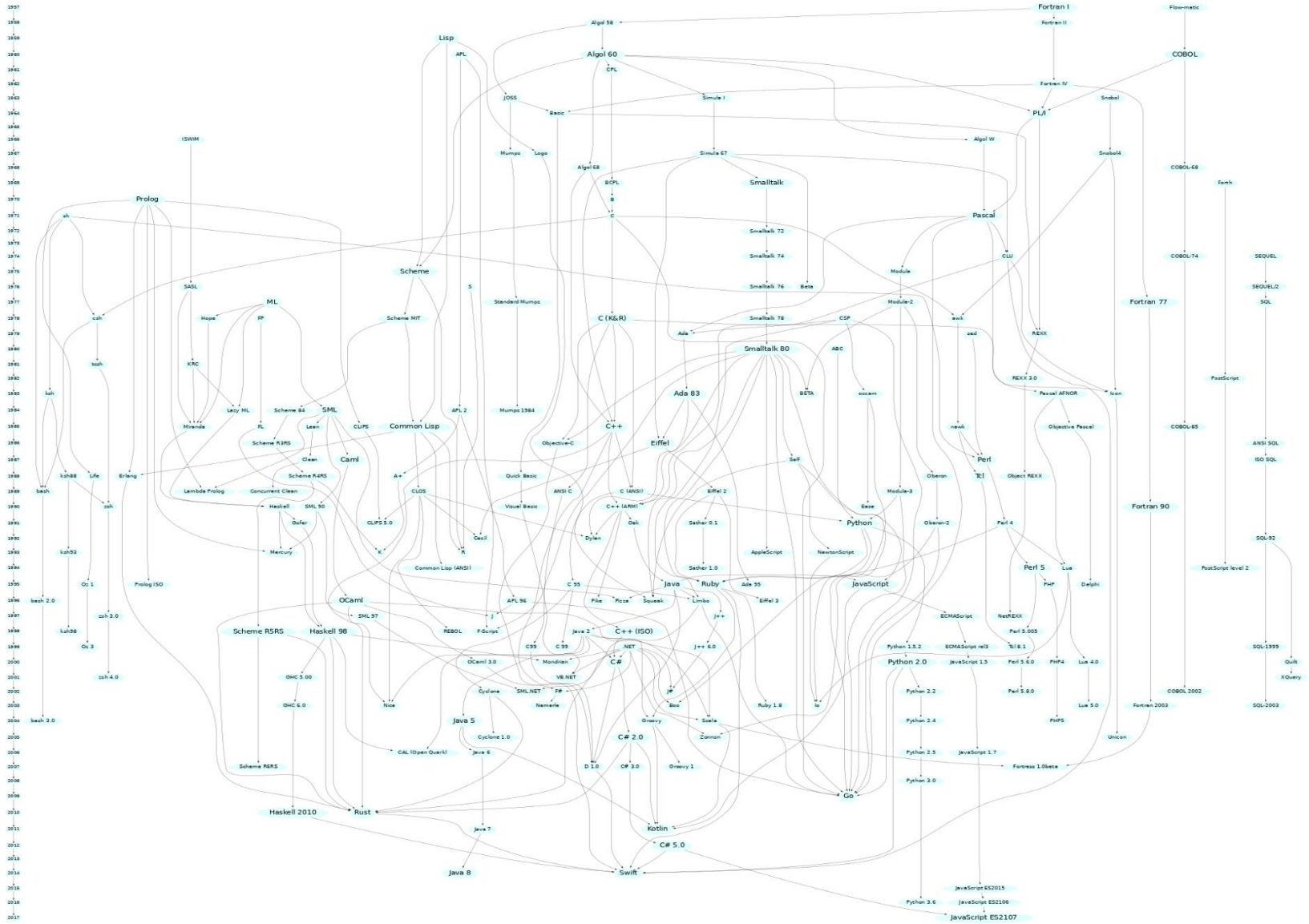


Слика 2.2.3: Генерације програмских језика (Извор: аутор)

На слици 2.2.4 [28] дато је развојно стабло програмских језика, од појаве првог компајлерског комерцијалног програмског језика (*Fortran*), па до најновијих програмских језика.

2.2.3.1. Језици ниског нивоа

Језици ниског нивоа су језици близу сета инструкција на машинском нивоу (нивоу машине). Програмски језик ниског нивоа директно ступа у интеракцију с регистрима и меморијом. Програми развијени на језицима ниског нивоа зависе од машине и нису преносиви. Ови језици не захтевају компајлер или интерпретер за превођење изворног кода у машински кôд. Углавном се користе за развој оперативних система, управљачких програма уређаја, база података и апликација које захтевају директан приступ хардверу. Језици ниског нивоа су класификовани у две категорије – *машински језици* и *асемблерски језици*.



Слика 2.2.4: Развојно стабло програмских језика (Извор: [28])

- **Машински језик (1GL).** Машински језик (језик машине) је најближи језик хардверу. Састоји се од скупа инструкција (низ бинарних битова) које директно извршава рачунар. Свака инструкција обавља веома специфичан и мали задатак. Упутства написана на машинском језику зависе од машине и варирају од рачунара до рачунара. Почеци програмирања, какво је данас познато, почиње касних 40-их и раних 50-их година 20. века и у то време програм је писан само на машинском језику. Развијање програма помоћу машинског језика је напоран посао и програмер, пре програмирања на машинском језику, мора да има додатна знања о архитектури одређеног уређаја.
- **Асемблерски језик (2GL).** Следећа генерација програмских језика су асемблерски језици и они представљају побољшање у односу на машински језик. И ови програмски језици су базирани на машинском коду и стриктно су везани за архитектуру машине за коју су дизајнирани, али су прилагођени човеку. Асемблерски или машински кôд писан за једну архитектуру не може да се извршава на другачијој машини. Ови језици су још у употреби.

У табели 2.2.2 [29] могу се видети неке од предности, односно недостатака језика ниског нивоа.

Табела 2.2.2: Предности и недостаци програмских језика ниског нивоа (Извор: [29])

Предности језика ниског нивоа	**Недостаци језика ниског нивоа**
1. Програми развијени на језицима ниског нивоа су брзи и меморијски ефикасни.	1. Програми развијени на језицима ниског нивоа зависе од машине и нису преносиви.
2. Програмери могу да користе процесор и меморију на бољи начин користећи језик ниског нивоа.	2. Тешко их је развијати, исправљати и одржавати.
3. Нема потребе да било који преводилац преводи изворни код у машински код. Тако се смањује време компилације и интерпретације.	3. Више су склони грешкама.
4. Језици ниског нивоа омогућавају директну манипулацију рачунарским регистрима и складиштењем.	4. Програмирање ниског нивоа обично резултује лошом продуктивношћу програмирања.
5. Могућност директне комуникације са хардверским уређајима.	5. За програмирање у језику ниског нивоа програмер мора да има додатна знања о архитектури одређеног уређаја.

2.2.3.2. Програмски језици високог нивоа

Програмски језици високог нивоа почињу да се појављују касних педесетих година 20. века. Ови програмски језици су независни у односу на хардвер и нису везани ни за једну архитектуру процесора и система. Ови језици су слични људском, њихова синтакса је базирана на енглеском језику и прилагођени су програмерима (лако се кодирају, исправљају, уклањају). Зависе од помоћних алата, компајлера или интерпретера којим се преводе у машински кôд. Овај приступ омогућава да програм писан у њима буде преведен и ради на било којој архитектури за коју постоји преводилац.

Језици високог нивоа могу се класификовати на основу: *парадигме програмирања* и модела извршавања/*преводиоца* (компајлер и интерпретер).

У табели 2.2.3 [29] могу се видети неке од предности, односно недостатака језика високог нивоа.

Табела 2.2.3: Предности и недостаци програмских језика високог нивоа (Извор: [29])

Предности језика високог нивоа	Недостаци језика високог нивоа
1. Погодни су за програмирање. Лако их је писати, исправљати и одржавати.	1. Потребно је додатно време за превођење да би се изворни кôд превео на машински.
2. Они обезбеђују виши ниво апстракције од машинских језика.	2. Програми на високом нивоу су релативно спорији од програма ниског нивоа.
3. То су језици који су независни о архитектури рачунара.	3. У поређењу са програмима ниског нивоа, они су обично мање меморијски ефикасни.
4. Лако се уче.	4. Није могућа директна комуникација са хардвером.
5. Мање склоности грешкама, грешке се лако проналазе и исправљају.	
6. Програмирање високог нивоа резултира бољом продуктивношћу програмирања.	

Разлика између програмских језика ниског и високог нивоа

Основна разлика између програмских језика ниског и високог нивоа је у томе да језик високог нивоа лако интерпретирају програмери, али не и рачунари (уређаји), док језик ниског нивоа могу лако разумети рачунари, али не и људи. Језици ниског нивоа могу ефикасно да рукују хардвером. Језици високог нивоа су популарнији међу програмерима јер се лако уче, читају, исправљају и тестирају.

У табели 2.2.4 [30] сумарно су дате основне разлике између програмских језика ниског и високог нивоа.

Табела 2.2.4: Разлике између програмских језика ниског и високог нивоа (Извор: [30])

	Језици високог нивоа	Језици ниског нивоа
Брзина извршења	Спорији од језика ниског нивоа.	Бржи од језика високог нивоа.
Ефикасност меморије	Нису меморијски ефикасни.	Меморијски ефикасни.
Превођење	Неопходан компајлер или преводилац да би се превео програм у машински код.	Асемблерски језик захтева од асемблера да конвертује програм у машински код док се машински језик извршава директно од стране рачунара.
Разумљивост и учење	Лако разумљиви програмеру, лако се уче.	Лако разумљиви за рачунар, тешко их је научити.
Зависност и преносивост (портабилност)	Независни су од уређаја, могу да раде на више платформи (портабилни).	Зависни су од уређаја, нису портабилни.
Грешке и одржавање	Мање су склони грешкама, отклањање грешака и одржавање је релативно лако.	Више су склони грешкама, отклањање грешака и одржавање је тешко.
Апстракција хардвера	Погодан је за програмирање који обезбеђује висок ниво апстракције хардвера.	Прилагођен је машини и не пружа никакву или мању апстракцију хардвера.
Додатна знања	Не захтевају додатно знање архитектуре рачунара.	Захтевају додатно познавање архитектуре рачунара.
Апликације	Десктоп апликације, веб-локације, мобилни софтвер, итд.	Развој системских софтвера (оперативних система) и уграђених апликација.
Подршка	Веома велика подршка.	Немају велику подршку.

- **Брзина програма.** Програми на језицима ниског нивоа се пишу у бинарном или у асемблерском језику, не захтевају никакво превођење, директно су у интеракцији са регистрима и меморијом. Језици високог нивоа користе енглеске исказе за писање програма, захтевају преводиоце да преведу изворни код на машински језик, не комуницирају директно са хардвером.
- **Ефикасност меморије.** Језици ниског нивоа су меморијски ефикасни, углавном троше мање меморије. Језици високог нивоа нису ефикасни у меморији, они се обично покрећу унутар специфичног окружења за извршавање.
- **Лакоћа учења.** Језици ниског нивоа су језици прилагођени за машину. Да би се написао програм на језику ниског нивоа, морају да се знају бинарне инструкција ниског нивоа. Програмирање ниског нивоа је тешко научити, учење језика ниског нивоа захтева додатно знање и искуство о специфичној архитектури машине. Језици високог нивоа су „пријатељски језици“ програмера. Програми на језику високог нивоа пишу се на енглеском језику, што је много лакше запамтити него бинарне инструкције ниског нивоа.
- **Преносивост (портабилност)**. Језик ниског нивоа садржи низ рачунарских инструкција ниског нивоа које зависе од машине и разликују се за различите архитектуре, па су и развијени програми (машински или асемблерски) такође зависни од машине и нису портабилни. Језици високог нивоа користе енглеске изразе за писање програма који се даље преводе на машински језик помоћу преводиоца (за различите архитектуре машина постоје посебни преводиоци) и независни су од уређаја (машине), и самим тим су и портабилни.
- **Отклањање грешака и одржавање.** Језици ниског нивоа су више склони грешкама, откривање и одржавање грешака је напоран и дуготрајан процес. Језици високог нивоа су мање склони грешкама, готово све синтаксне грешке се идентификују помоћу компајлера или интерпретера и обично их је лако исправити и одржавати.
- **Ниво апстракције.** Језик ниског нивоа обезбеђује малу или никакву апстракцију хардвера, он је најближи језик хардверу, директно комуницира са регистром рачунара и меморијом. Језик високог нивоа обезбеђује висок ниво апстракције хардвера, не комуницира директно са регистром рачунара и меморијом, интеракција са хардвером се одвија преко оперативног система и других софтвера.
- **Додатна знања и искуства.** Језици ниског нивоа зависе од машине, захтевају претходно знање о одређеној архитектури рачунара пре него што се заиста може написати програм за тај рачунар. Језици високог нивоа су независни од машина, не захтевају претходно знање о архитектури рачунара.
- **Апликације.** Језици ниског нивоа директно комуницирају са хардвером, обезбеђују веома мало или нимало апстракције од хардвера, брзи су када се упореде са језицима високог нивоа и зато се они генерално користе за развој оперативних система и уграђених апликација. Језици високог нивоа обезбеђују виши ниво апстракције од хардвера, данас су готово сви софтверски програми развијени на језику високог нивоа, користе се за развој разних апликација као што су – десктоп апликације, веб-локације, услужни програми, мобилне апликације, итд.
- **Подршка.** За језике ниског нивоа постоји далеко мања подршка него за језике високог нивоа. Број професионалаца који користе језике на ниском нивоу је много мањи од броја професионалаца који су корисници језика високог нивоа.

Програмски језици треће генерације

Програмски језици треће генерације (3GL) су језици опште намене и користе се за пословне, научне и опште примене. Укључују програмске језике високог нивоа као што су *Pascal, C, Fortran, C++, Java*. Програм који је написан на програмском језику високог нивоа компајлер преводи у машински језик ради извршења. Они су дизајнирани тако да их људи лакше разумеју и укључују карактеристике као што су именоване промењиве, условне структуре, итеративни искази, додела и структура података. Програмски језици на високом нивоу омогућују програмерима да се фокусирају на решавање проблема, а не на детаље ниског нивоа који су повезани са асемблерским језицима. Лакше их је исправити и одржавати него асемблерске језике.

Предности језика треће генерације су [26]:

- једноставност читљивости;
- јасно дефинисана синтакса (и семантика);
- погодни су за пословне и научне апликације;
- независност од уређаја;
- преносивост на друге платформе;
- једноставност отклањања грешака;
- брзина извршења.

Ови језици су независни од уређаја и могу се компајлирати за различите платформе. Први језици треће генерације су били процедурални, јер су се фокусирали на то како се нешто ради, а не на оно шта треба да се уради. Касније, језици постају објектно оријентисани и програмски задаци су подељени на објекте.

Рани примери програмских језика треће генерације

- **Flow-matic.** У периоду између 1952. и 1955. године, у *Remington Rand*-у од стране научнице *Grace Hopper*, развијен је један од првих компајлерских језика за *UNIVAC*. Циљ је био да се програмирање приближи пословним корисницима којима није одговарало математичко означавање које се до тада користило у рачунарству. *Flow-matic* је имао велики утицај на даљи развој програмских језика, пре свега *Cobol*-а [31].
- **Fortran**. Године 1957. појавио се први од језика на високом нивоу који је имао придружени преводилац (компајлер) – *FORTRAN*-а (*FORmula TRANslating*). Језик је дизајниран у *IBM*-у за научно рачунарство (*John Backus*). Компоненте су биле веома једноставне и омогућавале су програмеру приступ ниским нивоима до рачунара. Када је први пут представљен, третиран је са сумњом због уверења да ће програми компајлирани из језика високог нивоа бити мање ефикасни од оних који су написани директно у машинском коду. *Fortran* је постао популаран јер је омогућио преношење постојећег кода на нове рачунаре, на тржишту хардвера који се брзо развијао; језик је на крају постао познат по својој ефикасности. Данас би се овај језик сматрао рестриктивним јер је укључивао само изјаве *IF*, *DO* и *GOTO*, али у то време, ове наредбе су биле велики корак напред [32]. Његова најновија верзија изашла је крајем 2016.
- **LISP.** *Lisp* је први интерпретерски језик, после *Fortran*-а најстарији програмски језик који је и данас у употреби. *LISP* је изумео *John McCarthy* 1958. године док је радио на *Massachusetts Institute of Technology (MIT)*. Дизајниран је за истраживање вештачке интелигенције [33].

- **COBOL** (*Common Business-Oriented Language*). *Cobol* је био први пословни програмски језик и уведен је 1959. године, први рачунарски језик којег је користило Министарство одбране САД-а. Језик се и данас користи, постоји и објектно оријентисана верзија овог језика [26].
- **ALGOL** (*ALGOrithmic Language*). *Algol* је из породице императивних програмских језика који је првобитно развијен 1958. године од стране међународног комитета *Association of Computing Machinery (ACM)* под руководством *Alan J. Perlis* са *Carnegie Mellon University*. Његов главни допринос је корен стабла које је довело до језика као што су *Pascal, C, C ++* и *Java* [34].
- **BASIC** (*Beginner's All-purpose Symbolic Instruction Code*). *Basic* је настао 1964. године и дизајниран да буде једноставан за почетнике. Развили су га математичари *John George Kemeny* и *Tom Kurtzas* [35].
- **Pascal.** *Pascal* је развио 1968. године *Niklaus Wirth*. Овај језик је врло брзо нашао широку употребу, посебно се развио као добар наставни алат, комбиновао је најбоље особине језика који су се тада користили, *Cobol*, *Fortran* и *Algol* [32].
- **C.** Програмски језик *C* развио је *Dennis Ritchie* 1972. године, то је процедурален структуирани програмски језик компајлерског типа који комбинује ефикасност сличну асемблеру са лакоћом рада својственом језицима високог нивоа. *Ritchie* је развио *C* за нови *Unix* систем који је креиран у исто време. Због тога се *C* често користи за програмирање оперативних система као што су *Unix, Windows, MacOS* и *Linux* [26].

Објектно оријентисани језици

Крајем седамдесетих и раних осамдесетих година прошлог века развија се нови програмски метод. Познат је као *Object Oriented Programming* или *OOP*. Формални концепт објеката је први пут представљен 1960. године у ревизији програмског језика *Simula* у норвешком Центру за истраживање рачунарства. Већина модерних програмских језика подржава објектно оријентисано програмирање и објектно оријентисане функције додате су многим постојећим језицима, као што су *Basic*, *Fortran, Ada*, итд.

Главне карактеристике објектно оријентисаних језика дате су у табели 2.2.5 [26].

Табела 2.2.5: Објектно оријентисана парадигма (Извор: [26])

Својство	**Опис**
Класа (Class)	Класа дефинише апстрактне карактеристике неке ствари, укључујући њене атрибуте (или својства) и њено понашање (или методе). Чланови класе се називају објекти.
Објекат (Object)	Објекат је посебна инстанца класе са својим властитим скупом атрибута. Скуп вредности атрибута објекта назива се његово стање.
Метод (Method)	Методе повезане са класом описују понашање објеката у класи.
Пренос поруке (Message passing)	Пренос поруке је процес којим објекат шаље податке другом објекту или тражи од другог објекта да позове метод. Порука се састоји од адресе (објект примаоца поруке) и обавештења (говори шта треба да се уради). Порука се састоји од адресе (објект примаоца поруке) и поруке (говори шта треба учинити).
Наслеђивање (Inheritance)	Класа може да има подкласе (или класе деце) које су више специјализоване верзије класе. Механизам за креирање нових класа из постојећих назива се наслеђивање. Подкласа наслеђује атрибуте и методе родитељске класе. Наслеђивањем се формирају релације између класа.

Својство	Опис
Енкаспулација (Encapsulation)	Један од основних принципа објектно оријентисаног света је енкапсулација (или скривање информација). Унутрашњост објекта се чува као приватна за објекат и не може јој се приступити изван објекта. То значи да енкапсулација скрива детаље о томе како је одређена класа имплементирана, и захтева јасно дефинисани интерфејс око пружених услуга.
Апстракција (Abstraction)	Апстракција поједностављује сложеност моделирањем класа и уклањањем свих непотребних детаља. Сви битни детаљи су представљени, а небитне информације се игноришу.
Полиморфизам (Polymorphism)	Полиморфизам је понашање које варира у зависности од класе у којој се понашање позива. Две или више класа могу другачије реаговати на исту поруку. Исто име се даје методама у различитим подкласама, тј. једном интерфејсу и више метода.

Објектно оријентисано програмирање постало је популарно у развоју софтвера великих размера и постало је доминантна парадигма у програмирању од раних 1990-их. Његовом расту популарности помогао је пораст популарности графичких корисничких интерфејса (*GUI*), који је веома погодан за објектно оријентисано програмирање.

Од 80-их година 20. века појавио се велики број објектно оријентисаних програмских језика:

- Програмски језик **C++**. *C++* настаје 1983. године као објектно оријентисано проширење програмског језика *C* (*C++* је развио *Bjarne Stroustrup*). *C++* је дизајниран тако да користи моћ објектно оријентисаног програмирања и да одржава брзину и преносивост *C*-а. Најчешће се користи у симулацијама, као што су игре [26]. То је језик који се данас често изучава на курсевима информатике.
- Програмски језик **Python**. *Python* се појавио на тржишту 1991. године, развио га је *Guido van Rossum* и то је слободан софтвер. *Python* дозвољава коришћење променљивих без њиховог декларисања (имплицитно одређује типове). Програмер није приморан да дефинише класе (за разлику од *Java*), али је слободан да то уради када је потребно. *Python* је добар избор за математичке прорачуне. Због његове флексибилности он је изузетно популаран за многе употребе (главни алат веб програмера). Данас је један од најкоришћенијих језика на свету [36].
- Програмски језик **Java**. *Java* је развијена 1991. године у *Sun Microsystems*-у, за потребе интерактивне телевизије. Језици *C* и *C++* су утицали на синтаксу језика, а *Java* је дизајнирана на основу преносивости. Године 1994. *Java* пројектни тим је променио фокус на веб. Следеће године *Netscape* је лиценцирао *Java*-у за употребу у свом интернет претраживачу *Navigator*. Програмски језик *Java* је веома популаран и имао је велики утицај на касније језике укључујући и *C#* и може се рећи да је програмски језик садашњости и будућности, мада овај језик има озбиљне проблеме у оптимизацији (програми написани у њему раде споро) [32].
- Програмски језик **Ruby**. *Ruby* је интерпретирани, објектно оријентисани језик који је 1995. године развио *Yukihiro „Matz“ Matsumoto*. *Ruby* има сличну синтаксу у односу на многе програмске језике као што су *C* и *Java*, тако да га *Java* и *C* програмери лако уче. Подржава углавном све платформе као што су *Windows, Mac, Linux*. *Ruby* је базиран на многим другим језицима као што су *Perl, Lisp, Smalltalk, Eiffel* и *Ada*. То је интерпретирани скриптни језик, што значи да већина његових имплементација извршава инструкције директно и слободно, без претходног компајлирања програма у упутства на машинском језику [37].

Језици четврте генерације

Програмски језици четврте генерације (4GL) спадају у групу програмских језика који покушавају да се приближе људском језику. Дизајнирани су како би смањили напор у програмирању и укључују генераторе извештаја и генераторе образаца. Развој језика четврте генерације почео је 70-их година двадесетог века паралелно са језицима треће генерације. Језици четврте генерације су генерално непроцедурални, дизајнирани су за обраду велике количине података без фокусирања на индивидуалне бајтове. Ови језици су повезани са базама података и обрадом података и користе графички кориснички интерфејс. Представници језика четврте генерације су језици за обраду података као *SAS* і *SPSS*. Недостатак 4GL-а је што су они спори у односу на компајлиране језике [29].

Језици пете генерације

Пета генерација програмских језика (5GL) је група програмских језика који су били у развоју 80-их година двадесетог века. То су језици који су засновани на решавању помоћу ограничења која су дата у програму, а не на основу коришћења програма који је написао програмер, тј. програмер поставља параметре за програм на основу којих тражи решење. Ови језици су тако дизајнирани да рачунар решава задате проблеме без програмера. Ови језици се углавном користе у истраживачке сврхе, посебно у области вештачке интелигенције. Данас су језици ове генерације синоними за декларативне језике. Када су настајали многи су их сматрали будућношћу програмирања, предвиђали су да ће ови језици заменити све језике високог нивоа за развој система. Али, показало се да је стварање овако ефикасног алгоритма веома компликован задатак, да се развој не може лако аутоматизовати и да још увек захтева присуство „људског програмера“. Један од најпознатијих језика пете генерације је *Prolog* и то је језик логичког програмирања [29].

2.2.4. Модерни програмски језици[1]

Последњих година све више се испоручују рачунари са више процесора који имају више од једног језгра, корисници наручују све захтевније апликације. Од програмера се тражи висока продуктивност. Програмери морају да владају програмским језицима који у потпуности користе предности нове процесорске архитектуре, који омогућавају брзу и ефикасну израду апликација различитих намена. Није довољно да знају само „стандардне“ вишенаменске језике као што су: *C#, Java, Python, C++, Objective C, PHP, Javascript.* Главни недостатак ових „старих“ језика је што они никада нису грађени од темеља до решавања насталих проблема у модерном свету програмирања.

Очигледно је да програмери морају да прихвате нове програмске језике како би се позабавили проблемима модерног софтверског инжењеринга. Следи преглед модерних језика [38, 39]:

- **Scala**. Ово је вероватно један од најстаријих модерних програмских језика. То је производ академске заједнице, који је дизајнирао *Martin Odersky* док је радио на *EPFL* (*École polytechnique fédérale de Lausanne*) у Швајцарској. Први пут издат 2003. године, *Scala* комбинује парадигме објектно оријентисаног и функционалног програмирања. Апликације писане у *Scala*-и раде на врху *JVM*-а (*Java virtual machine*), па су лако преносиве на више платформи.

[1] Ови језици се називају модерним, јер су сви они настали у овом веку (21. век).

- **Golang (Go).** Ово је језик који је *Google* креирао за сопствену интерну употребу. *Go* је посебно занимљив јер је један од његових дизајнера (*Ken Thompson*) творац *UNIX* оперативног система. *Go* је објављен 2009. године и од тада је објављен као *open-source* пројекат. *Go* се често наводи као „*C* за 21. век".
- **Rust.** *Rust* је системски програмски језик који је *Mozilla Research* створио као сигурнију и ефикаснију алтернативу *C/C++*. Званично је објављен као пројекат отвореног кода још 2010. године. Главни дизајнер тог језика био је *Graydon Hoare*.
- **Kotlin.** До недавно, *Kotlin* је једва био познат унутар шире програмске заједнице. Језик је креиран је од стране компаније *JetBrains*. Службено је објављен 2011. године, и ради на врху *JVM*-а (слично *Scala*-и). *Kotlin*-у је 2017. године пружена првокласна подршка на *Android* мобилној платформи (постао је трећи језик којем је додељен такав статус, након програмских језика *Java* и *C++*).
- **Dart.** *Dart* је скриптни и објектно оријентисани програмски језик који је оптимизован за клијенте на разним платформама. Развио га је *Google* 2011. године, а користи се за израду мобилних, десктоп и веб апликација. *Dart* подсећа на програмски језик *C*. Када се користи у веб апликацијама, преводи се у *JavaScript* и може се покретати на свим веб браузерима.
- **TypeScript.** *TypeScript* је бесплатан скриптни и објектно оријентисан програмски језик, дизајнирао га је 2012. године *Microsoft*, омогућава развој *JavaScript*-а за пословне апликације. *TypeScript* нуди програмерима да приликом писања кода користе строго типизирање (*strong typing*), класе, интерфејсе, наслеђивање, генеричке компоненте...
- **Julia**. *Julia* је динамичан, програмски језик високог нивоа који нуди првокласну подршку за конкурентно, паралелно и дистрибуирано програмирање. Овај језик је развио *Massachusetts Institute of Technology (MIT)* 2012. године. *Julia* се може користити у научном рачунарству, вештачкој интелигенцији…
- **Swift.** Створен у *Apple*-у, *Swift* је замишљен да буде сигурнија и концизнија алтернатива за *Objective-C* за развој апликација унутар *Apple*-овог екосистема. Језик је објављен 2014. године као пројекат отвореног кода. Њен водећи дизајнер је био *Chris Lattner* (који је такође један од изворних аутора *LLVM* пројекта). *Swift* код је сличан коду написаном у *Rust*-у.

2.2.5. Најпопуларнији програмски језици

Тешко је одредити који програмски језици су најпопуларнији. Један језик захтева већи број програмерских сати, други има више линија кода, трећи може да користи највише процесорског времена, и тако даље. Поједини језици су веома популарни за одређене врсте апликација. На пример, иако су поприлично „стари", неки програмски језици су и даље популарни: *Cobol* у пословној обради података; *Fortran* у рачунарској науци и инжењерству; *C* у уграђеним апликацијама и оперативним системима...

Најважнији параметри на основу којих се може мерити популарност програмског језика су [40]:

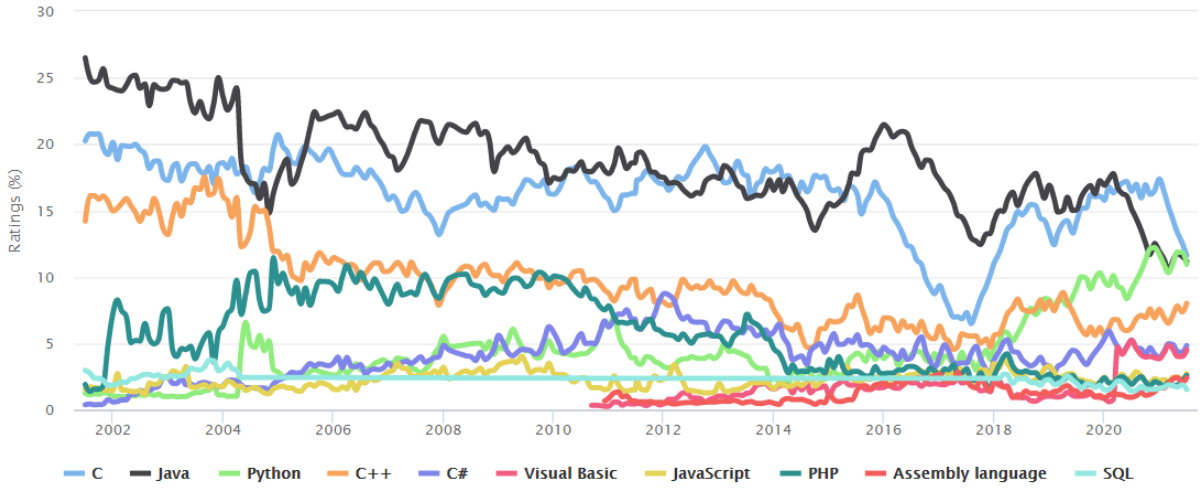
- Колико пута се име језика спомиње у претраживању веба;
- Број огласа за посао који спомињу језик;
- Број продатих књига за подучавање или опис језика;
- Процене броја постојећих линија кода написаних на језику;

- Број пројеката на том језику;
- Број постова у форумима и дискусионим групама о језику;
- Број студената уписаних на часове програмирања широм света;
- Број видео записа на том језику;
- Број постова;
- ...

Постоје бројни сајтови који објављују податке о популарности језика, сваки према различитим критеријумима. Следи кратак преглед неколико њих.

Tiobe Software

Компанија ***Tiobe Software*** [41] нуди сваког месеца, једну од најопсежнијих и најажурнијих истраживања употребе најпопуларнијих програмских језика и објављује *TIOBE Programming Community* индекс за тај месец. Постоји 25 претраживача који се користе за израчунавање *TIOBE* индекса, као што су *Google, Bing, Yahoo!, Wikipedia, Amazon, YouTube* и *Baidu*. На слици 2.2.5 [41] дат је приказ најпопуларнијих (најчешће коришћених) програмских језика у 21. веку (јул 2021. године).



Слика 2.2.5: TIOBE Programming Community Index (Извор: [41])

У табели 2.2.6 [41] дат је степен коришћења најпопуларнијих језика за јул 2021. и за јул 2020. године, као и позиције тих језика периодично на сваких 5 година од 1986. године.

Табела 2.2.6: Најпопуларнији програмски језици, јул 2021. (Извор: [41])

Програмски језик	2021	2020	2016	2011	2006	2001	1996	1991	1986
C	1	2	2	2	2	1	1	1	1
Java	2	1	1	1	1	3	22	-	-
Python	3	3	5	6	8	26	21	-	-
C++	4	4	3	3	3	2	2	2	8
C#	5	5	4	5	7	13	-	-	-
Visual Basic	6	11	13	-	-	-	-	-	-
JavaScript	7	6	7	10	9	9	24	-	-
PHP	8	7	6	4	4	11	-	-	-
SQL	9	8	-	-	-	38	-	-	-

Програмски језик	2021	2020	2016	2011	2006	2001	1996	1991	1986
R	10	13	17	28	-	-	-	-	-
Ada	33	33	27	17	16	20	8	4	2
Lisp	36	27	28	13	13	16	7	6	3
(Visual) Basic[2]	-	-	-	7	5	4	3	3	5
Swift	-	9	17	-	-	-	-	-	-
Ruby	-	10	11	10	24	30	-	-	-
Fortran	-	31	28	23	15	16	4	3	5
Pascal	-	243	16	14	35	12	3	10	6

Занимљиво је да три најпопуларнија програмска језика у 21. веку су у скоро истом редоследу: *Java, C* и *C++, Python* се тек 2020. године попео на постоље. *PHP* је 2011. године заузимао четврто место (2010. треће место), али је 2020. године пао на 7. место, односно 2021. на 8. место. *SQL* се први пут појављује на листи најпопуларнијих програмских језика 2020. године (8. место, 2021. године 9. место).

Преглед програмских језика са највећим порастом рејтинга у последњих 18 година (2003 – 2020) дат је у табели 2.2.7 [41].

Табела 2.2.7: Програмски језици са највећим порастом рејтинга, јул 2021. (Извор: [41])

Година	Највећи пораст рејтинга	Година	Највећи пораст рејтинга	Година	Највећи пораст рејтинга
2020	Python	2014	JavaScript	2008	C
2019	C	2013	Transact-SQL	2007	Python
2018	Python	2012	Objective-C	2006	Ruby
2017	C	2011	Objective-C	2005	Java
2016	Go	2010	Python	2004	PHP
2015	Java	2009	Go	2003	C++

IEEE Spectrum

Компанија ***IEEE Spectrum*** [42] нуди могућност интерактивног рангирања популарности програмских језика бирајући факторе рангирања језика: *тренд* (*Trending*), *посао* (*Jobs*) или *отворени код* (*Open*), за различите типове апликација *Web, Mobile, Enterprise* и *Embedded*.

- *Web* – језици који се користе за развој веб страница и апликација;
- *Mobile* – језици који се користе за апликације на мобилним уређајима;
- *Enterprise* – језици који се користе за пословне, десктоп и научне апликације;
- *Embedded* – језици који се користе за програмирање контролера уређаја.

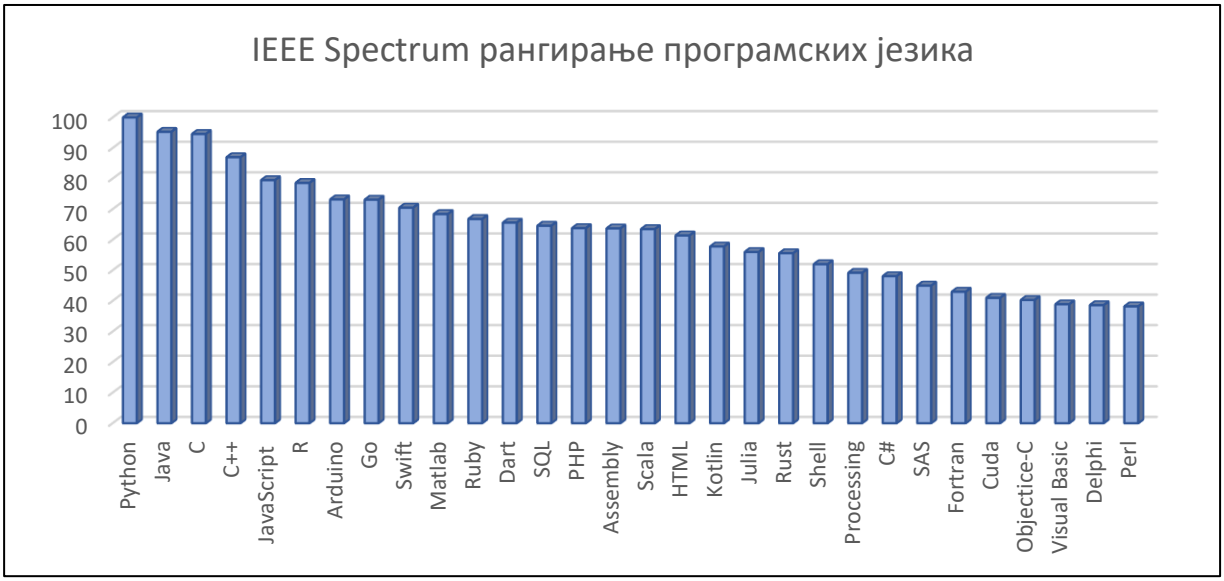
У табели 2.2.8 [42] дати су најпопуларнији програмски језици за различите типове апликација за сваки фактор рангирања, за 2020. годину.

[2] У овој табели постоји разлика између „Visual Basic“ и „(Visual) Basic“. До 2010. „(Visual) Basic“односио се на све могуће дијалекте Basic-a, укључујући Visual Basic. Након неке расправе, одлучено је да се „(Visual) Basic“ подели на све његове дијалекте, као што су Visual Basic .NET, Classic Visual Basic, PureBasic и Small Basic… Пошто је Visual Basic .NET постао главна имплементација Visual Basic-a, сада се назива „Visual Basic“.

Табела 2.2.8: Најпопуларнији програмски језици за различите типове апликација (Извор: [42])

Ранг	Trending (језици који брзо напредују)				Jobs (језици које траже послодавци)				Open (језици отвореног кода)			
	Web	Mobile	Enterprise	Embedded	Web	Mobile	Enterprise	Embedded	Web	Mobile	Enterprise	Embedded
1.	Python	Java	Python	Python	Python	C	Python	Python	Python	Java	Python	Python
2.	Java	C	Java	C	Java	Java	C	C	Java	C	Java	C
3.	Go	C++	C	C++	Go	C++	Java	C++	JavaScript	C++	C	C++
4.	JavaScript	Dart	C++	Arduino	JavaScript	Swift	Go	Assembly	HTML	Dart	C++	Arduino
5.	Dart	Swift	Go	Assembly	HTML	Scala	C++	Arduino	Dart	Swift	Go	C#
6.	HTML	Scala	Swift	C#	Ruby	Dart	R	C#	Go	C#	R	Rust
7.	Scala	Kotlin	Scala	Rust	Scala	C#	Swift	Rust	Ruby	Kotlin	Swift	Assembly
8.	Ruby	C#	Ruby	Elixir	Dart	Objective-C	SQL	Ada	PHP	Scala	Ruby	Elixir
9.	Kotlin	Objective-C	Julia	Ada	PHP	Kotlin	Ruby	D	C#	Objective-C	C#	Verilog
10.	C#	Scheme	R	Verilog	C#	D	Matlab	Forth	Kotlin	Delphi	Shell	VHDL
11.	Rust	Delphi	Matlab	D	Processing	Scheme	Scala	LabView	Scala	Scheme	Matlab	LadderLogic
12.	PHP	D	C#	VHDL	Kotlin	Delphi	Shell	Verilog	Rust	D	Julia	Ada

Сумарни приказ рејтинга програмских језика, укључујући све факторе рангирања и све типове апликација дат је на слици 2.2.6 [42]. Извршено је пондерисање према *Python*-у, као најбоље рангираном програмском језику.

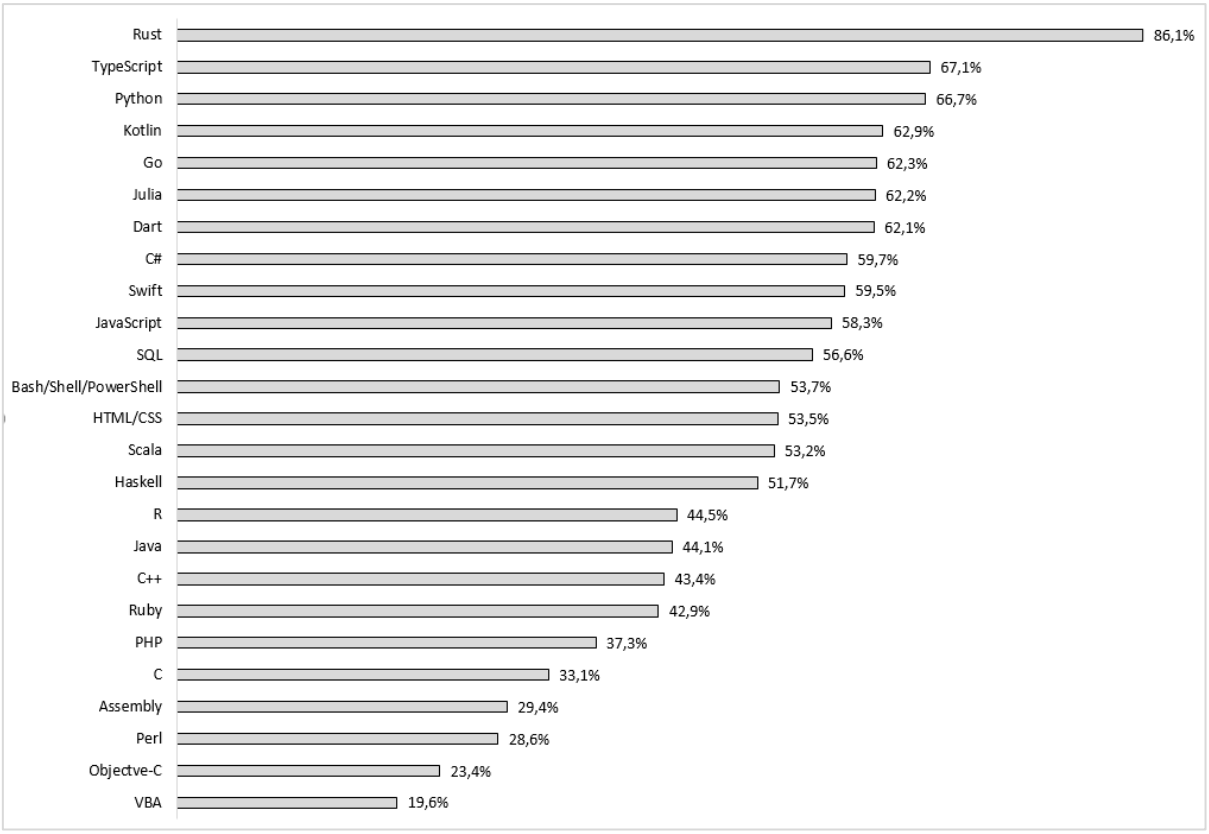


Слика 2.2.6: Рангирање популарности програмских језика, 2020. година (Извор: [42])

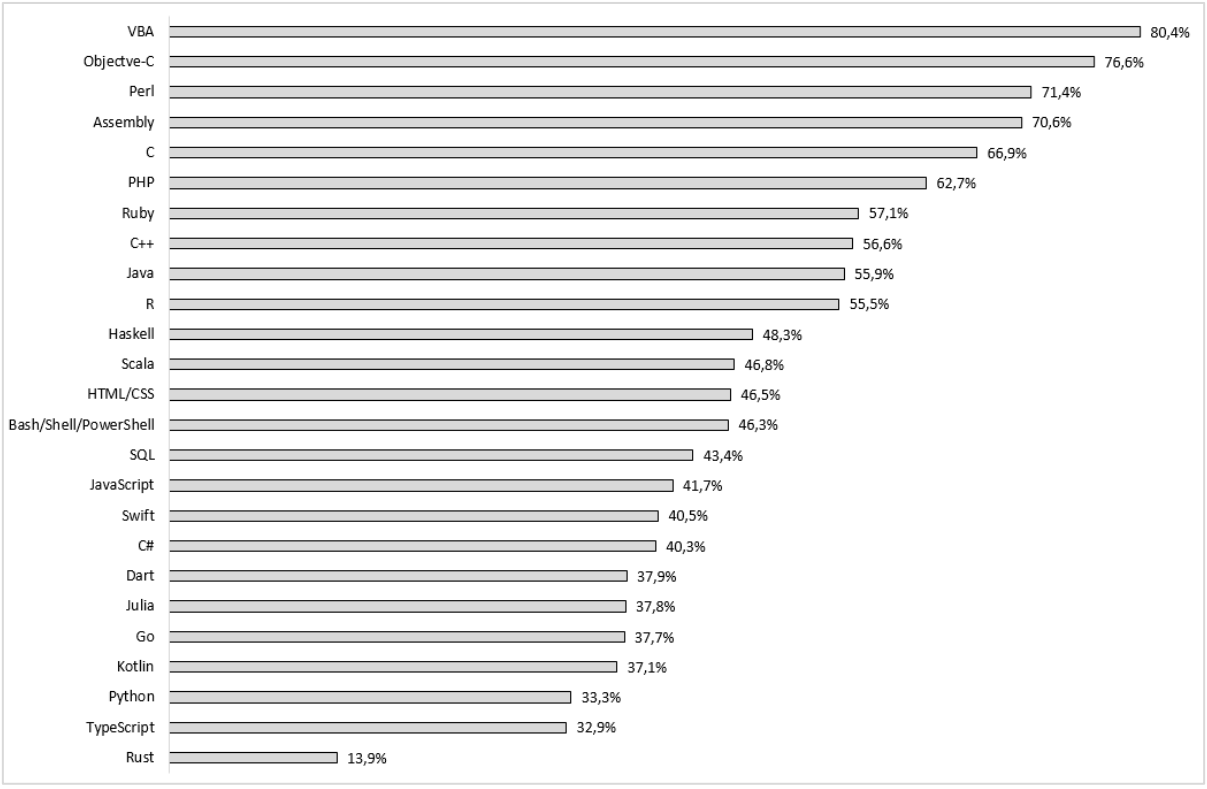
Stack Overflow

Stack Overflow [43] је веб сајт који служи као платформа професионалним програмерима и ентузијастима. То је једна од највећих и најпоузданијих онлајн заједница за програмере широм света. У фебруару 2020. години скоро 65000 програмера из 186 земаља је учествовало у истраживању (*2020 Developer Survey*).

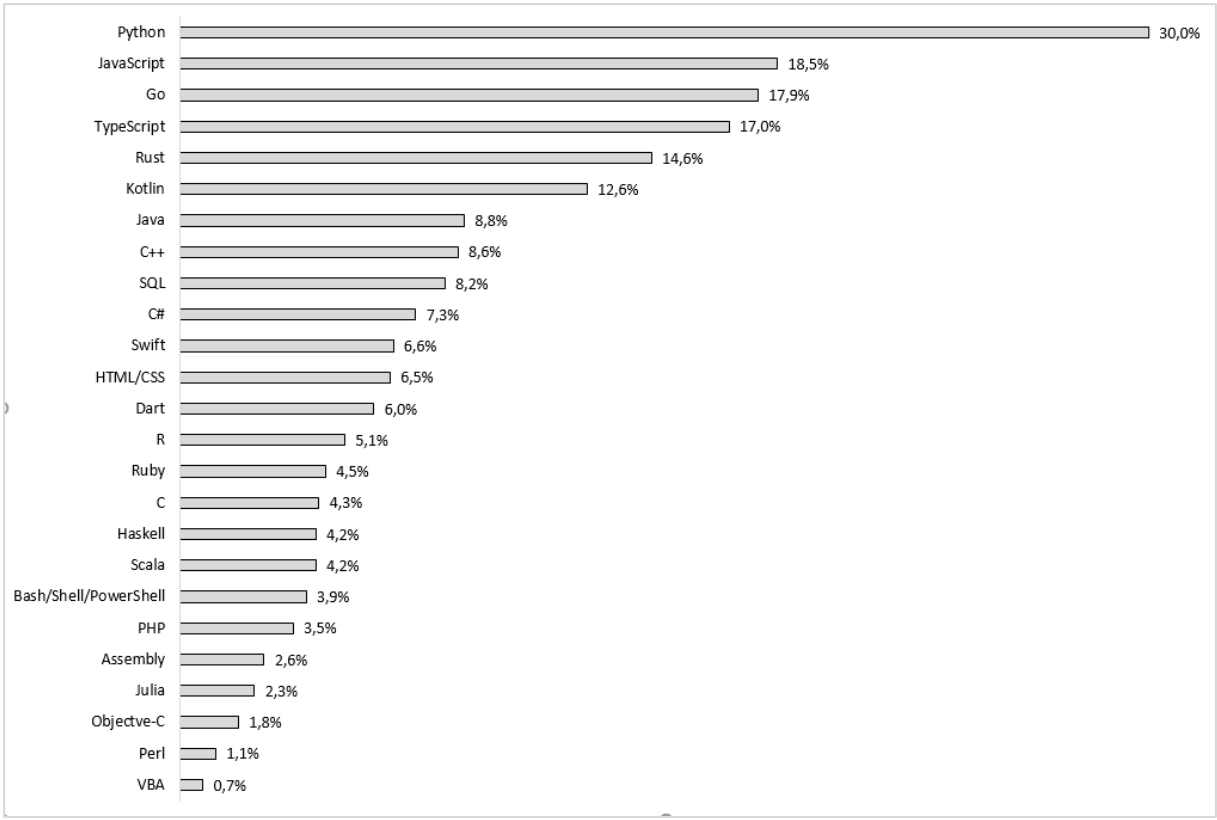
На сликама 2.2.7 – 2.2.9 [44-46] дати су резултати тог истраживања: приказ најомиљенијих програмских језика, програмских језика који се програмери плаше (не желе да наставе да их користе), програмских језика које желе да науче.



Слика 2.2.7: Листа програмских језика које програмери користе и даље желе да их користе (Извор: [44])



Слика 2.2.8: Листа програмских језика које програмери користе, али не желе да их и даље користе (Извор: [45])



Слика 2.2.9: Листа програмских језика које програмери не користе, али желе да их науче (Извор: [46])

Очигледно је да програмери који користе *Rust* воле *Rust*, јер 86,1% *Rust* програмера се изјаснило да жели да настави да га користи. Програмски језици *TypeScript* (67,1%), *Python* (66,7%), *Kotlin* (62,9%), *Go* (62,3%), *Julia* (62,2%), *Dart* (62,1%) су језици за које су програмери такође показали велико задовољство (изнад 60%). Занимљиво је да само 1/3 испитаника (33,1%) који користе програмски језик *C* и даље жели да га користи [44].

Поједине програмске језике програмери избегавају, односно не желе да наставе да их користе. На врху листе је језик *VBA* где чак 80,4% корисника овог програмског језика не жели да наставе да га користи, потом следе *Objective-C* (76,6%) и *Perl* (71,4%). *Rust* се поново показао као најомиљенији програмски језик, јер само 13,9% програмера не жели да наставе да га користи [45].

Од програмских језика који не користе, 30% програмера је изражило жељу да програмира у *Python*-у, док је следећи „жељени“ језик *JavaScript* (18,5%). Програмери су најмање заинтересовани за програмске језике *Perl* (1,1%) и *VBA* (0,7%) [46].

Приказани прегледи могу бити добри показатељи како би се проверило да ли су вештине програмирања и даље актуелне или да се донесе стратешка одлука о томе који програмски језик треба да се усвоји када се почиње са градњом новог софтверског система, па самим тим и који програмски језик треба учити.

2.2.6. Стандардизација програмских језика

У току кратке рачунарске историје направљено је и имплементирано на хиљаде програмских језика. Програмске језике креирају појединци, мањи тимови или су део пројекта неке софтверске фирме. Током употребе програмског језика откривају се грешке и јављају се проблеми (питања) које треба решити. Неопходно је да се формира тим/одбор који ће решити проблеме и преузети укупну одговорност за континуирану еволуцију програмског језика и за креирање стандардне језичке спецификације која није под контролом једног појединца или организације. Под покровитељством признатих организација за развој стандарда (*ISO, ANSI, Ecma International, JISC, IEEE*, итд.) развијају се стандарди за програмске језике од стране „*комитета за стандарде*" који се састоји од експерата за програмске језике. Појединци или организације које желе директно да утичу на еволуцију програмских језика морају да се придруже и ефикасно учествују као чланови комитета за стандарде језика [47].

Неопходна је стандардизација, како на глобалном нивоу (*међународни стандарди* [48]), тако и на локалном нивоу (*српски стандарди* [49]), која би упутила на повезивање знања које би довело до запажања потенцијалних разлика и утврђивање мера за унапређење програмских језика. *Међународна класификација стандарда (ICS)* развијена је почетком деведесетих година XX века као основ за структуру каталога међународних, регионалних и националних стандарда и других нормативних докумената. *ICS* на светском нивоу унапређује широку употребу међународних, регионалних и националних стандарда и других нормативних докумената [50].

Стандардима се олакшава и поспешује: системски приступ решавању проблема; распрострањеност и приступ новим технологијама; инклузија свих заинтересованих страна у одређивању правила за будућа истраживања и развој, трансфер знања и технологија; умрежавање са другим индустријама и носиоцима активности у будућим истраживањима и технологијама; компатибилност сопствених технологија са другим технологијама; компатибилност са другим произвођачима; обезбеђивање платформе за иновације; коришћење резултата истраживања [51-55].

ICS је хијерархијска класификација која се састоји од три нивоа. *Ниво 1 (ICS1)* обухвата 40 области активности у стандардизацији. Свака област има двоцифрену одредницу (*ICS1 од 01 до 99*), нпр.: 35 означава област *Информационе технологије (ICS1= 35)* [56]. Области су подељене у 392 подобласти које чине *ниво 2 (ICS2)*. Одредница сваке подобласти нивоа 2 састоји се од одреднице области и од троцифреног броја одвојеног тачком од броја области, нпр.: 35.240 означава стандардизовану подобласт *Примена информационих технологија (ICS2= 35.240)*. Од 392 подобласти, њих 127 се даље дели на 909 подобласти које чине *ниво 3 (ICS3)*. Одредница подобласти нивоа 3 састоји се од одреднице подобласти нивоа 2 и од двоцифреног броја одвојеног тачком од броја подобласти нивоа 2, нпр.: 35.240.20 представља подобласт *Примене ИТ у канцеларијским пословима (ICS3= 35.240.20)*. Све подобласти *ICS2* подељене у подобласти садрже, уз неколико изузетака, подобласти *ICS3* која обухвата предмет те подобласти *ICS2* у целини. Такве подобласти *ICS3* са општим предметом имају ознаке које се завршавају са „.01". На пример, подобласт *ICS2* 35.240 Примена ИТ садржи као прву подобласт *ICS3 35.240.01 Примена информационе технологије уопште* у коју морају да се укључе стандарди који обухватају подручје примене ИТ уопште (нпр. *ISO/IEC* 10027:1990 Information technology – *Information Resource Dictionary System (IRDS) framework*). Стандарди код нпр. Примене у пројектовању подржаном рачунарима, Идентификационе картице, Примене ИТ у канцеларијским пословима,… морају да се укључе тим редоследом у друге посебне подобласти *ICS3*: 35.240.01, 35.240.10, 35.240.15, 35.240.20, 35.240.30, 35.240.40, 35.240.50, 35.240.60, 35.240.63, 35.240.67, 35.240.68, 35.240.69,

35.240.70, 35.240.80, 35.240.90, 35.240.95 или 35.240.99. Највећи број подобласти *ICS2* које су подељене у подобласти *ICS3* садрже подобласт која се завршава одредницом „.99“. Таква подобласт *ICS3* обухвата стандарде који не одговарају ни предмету опште подобласти *ICS3* нити предметима посебних подобласти *ICS3* у оквиру те подобласти *ICS2*. На пример, предмет стандарда *IEC 948:1988 Numeric keyboard for home electronic systems (HES)* не одговара ниједном предмету подобласти *ICS 3* 35.240.01 – 35.240.80. У овом случају, овај стандард мора да се укључи у подобласт *35.240.99 – Примене ИТ у осталим областима рада* [50].

Област Информационих технологија (*IT, ICS1 = 35*) има *15 подобласти ICS2* [50]:

- 35.020 – Информационе технологије (ИТ) уопште;
- 35.030 – ИТ безбедност;
- 35.040 – Скупови знакова и кодирање информација;
- 35.060 – Језици који се користе у информационим технологијама;
- 35.080 – Софтвер;
- 35.100 – Међусобно повезивање отворених система (*OSI*);
- 35.110 – Умрежавање;
- 35.140 – Рачунарска графика;
- 35.160 – Микропроцесорски системи;
- 35.180 – ИТ терминалска и друга периферијска опрема;
- 35.200 – Опрема за међусобно повезивање и интерфејс;
- 35.210 – Клауд (*Cloud*) рачунарство;
- 35.220 – Јединице за складиштење података;
- 35.240 – Примена информационих технологија;
- 35.260 – Канцеларијске машине.

Међународна организација за стандардизацију (ISO) и *Међународна електротехничка комисије (IEC)* разрађују међународне стандарде у области информационих технологија у оквирима *Првог обједињеног техничког комитета (ISO/IEC JTC1)*, који тренутно обједињује 37 поткомитета (*SC*). Примене *ISO/IEC* стандарда захтевају одговарајући ниво Е-друштва, Е-институције образовања, као и националну стандардизацију одговарајућих и циљних нивоа квалитета, обезбеђења и менаџмента квалитетом.

У развоју нових пројеката за подобласт *Језици који се користе у информационој технологији – 35.060* (пресек 1.1.2020.) укључен је 1 комитет и 3 подкомитета. У табели 2.2.9 [57] дат је приказ техничких комитета и подкомитета према количини публикованих и развојних пројеката за стандарде у подобласти 35.060.

Табела 2.2.9: Технички комитети и подкомитети за стандарде у подобласти 35.060 (Програмски језици у ИТ) (Извор: [57])

Комитети/ подкомитети	Област	Број публикова-них пројеката	Број развојних пројеката
ISO/IEC JTC 1/SC 32	Управљање и размена података	32	16
ISO/IEC JTC 1/SC 34	Опис докумената и језици за обраду	12	1
ISO/IEC JTC 1/SC 35	Кориснички интерфејси	2	0
ISO/IEC JTC 1/SC 24	Рачунарска графика, обрада слика и представљање података о животној средини	22	0
ISO/IEC JTC 1/SC 22	Програмски језици, њихова окружења и системски софтверски интерфејси	101	7
ISO/IEC JTC 1/SC 6	Телекомуникације и размена информација између система	1	0
ISO/IEC JTC 1/SC 7	Системско и софтверско инжењерство	5	0
ISO/IEC JTC 1	Први технички комитет за развој стандарда у ИКТ за пословне и потрошачке апликације	5	0

2.3. Програмирање у образовању Србије

2.3.1. Услови за развој и имплементацију ИКТ-а у Србији

Као последица убрзаног развоја и употреба информационо-комуникационих технологија (ИКТ) савремено друштво се трансформисало у информационо друштво у којем ИКТ играју веома важну улогу у производњи и економији, као и у свим другим сферама живота људи и друштва у целини [58].

Институт *Portulans* [59] је 2020. године објавио *The Network Readiness Index* – индекс мрежне спремности – извештај о дигиталној трансформацији у глобалној економији (*дигитална трансформација је интеграција дигиталних технологија у сва подручја пословања како би се испунили пословни и тржишни захтеви*). Рангиране су економије укупно 134 држава на основу њихових перформанси у преко 50 варијабли/фактора. Србија је рангирана на 52. месту са индексом од 52,96; на врху су Шведска и Данска са индексима 82,75, односно 82,19. Словенија је на високом 27. месту (индекс 66,58), Хрватска је на 43. месту (индекс 55,94), Црна Гора на 58. месту (индекс 50,95), док је Босна и Херцеговина на 87. месту (индекс 41,73). Индекси процењују стање мрежне спремности и омогућавају идентификацију приоритетних области како би се потпуније искористиле ИКТ за дигитализацију и друштвено-економски развој.

У табели 2.3.1 [59] дата је анализа неких фактора, из овог извештаја, за имплементацију ИКТ у Србији.

Табела 2.3.1: ИКТ профил Србије (Извор: [59])

	Фактори	Ранг Србије		Најбољи ранг		Индика-тори
		ранг	вредност	држава	вредност	
1.	Индекс државних услуга на мрежи	42	0,79	Korea, Rep.	1	[0-1]
2.	Индекс е-учешћа	41	0,82	Estonia	1	[0-1]
3.	Усвајање нових технологија	80	3,26	Netherlands	5,70	[1-7]
4.	Улагање у нове технологије	93	3,25	United States	6,0	[1-7]
5.	У којој мери предузећа користе најновије дигиталне алате за продају своје робе и услуга	82	4,44	United States	6,21	[1-7]
6.	У којој мери активно становништво поседује довољно дигиталних вештина	74	4,09	Finland	5,83	[1-7]
7.	Издаци за истраживање и развој које врше влада и високошколске установе (% БДП-а)	32	56,10	Denmark	107,58	%
8.	Фирме са веб сајтом	19	79,10	Finland	95,64	%
9.	Професионалци	48	13,78	Luxembourg	40,9	%
10.	Стопа писмености одраслих	21	98,84	Ukraine	99,97	%
11.	Појединци који користе интернет	56	73,36	Qatar	99,65	%
12.	Домаћинства са приступом Интернету	58	72,8	Kuwait	100	%
13.	Број активних корисника друштвених медија (% становништва)	91	42	Kuwait	99	%
14.	Претплате на фиксни широкопојасни приступ ≥10 Mbit/s (% од укупног броја претплата)	28	92,91	Korea, Rep	100	%
15.	Укупна потрошња рачунарског софтвера (% БДП -а)	106	0,04	United States	1,17	%
16.	Људи који су користили Интернет да би купили нешто на мрежи у претходној години	52	19,64	Denmark	77,97	%
17.	Квалитет образовног система (ПИСА просечни резултати из математике)	45	448,3	China	591,4	
18.	Међународна пропусност Интернета по кориснику интернета (bit/s)	49	73692	Luxembourg	8328979	
19.	Број развијених активних мобилних апликација по особи	42	76,31	Singapore	97,24	
20.	Број робота у употреби на 10.000 запослених у прерађивачкој индустрији	53	0,99	Germany	100	
21.	Сигурни Интернет сервери (на милион становника)	42	8072	Denmark	277134	
22.	Удео основних школа са приступом Интернету	н/п		23 zemlje	100	%
23.	Извоз високе технологије (% укупног извоза индустријске робе)	н/п		Hong Kong	65,57	%
24.	Број пријава за патенте везане за ИКТ (на милион становника)	н/п		Sweden	145,2	

На основу овог извештаја Србија је земља која се по питању ИКТ сектора сврстава у мање развијене земље. Поражавајуће је да Србија за *Укупну потрошњу рачунарског софтвера* издваја само 0,04% од БДП-а (САД издваја 29,25 пута више од нас). *Стопа писмености одраслих је на високом нивоу* (98,84%). Нажалост, за фактор *Удео основних школа са приступом Интернету* не постоје подаци.

С друге стране, Републички завод за статистику Републике Србије издао је извештај за 2019. годину о тренутном стању у области ИКТ у домаћинствима/појединцима, односно у предузећима. У табели 2.3.2 [58] дат је статистички преглед главних налаза тог истраживања за године 2016 – 2019.

Табела 2.3.2: Употреба ИКТ у Србији (Извор: [58])

Питање	2019.	2018.	217.	2016.
Поседовање рачунара у домаћинству	73,1%	72,1%	68,1%	65,8%
Интернет у домаћинствима	80,1%	72,9%	68%	64,7%
Корисници рачунара (појединци)	78,3%	77,2%	73,9%	72,8%
Употреба интернета (појединци)	80,6%	75,8%	74,4%	70,8%
Куповина/наручивање робе или услуга преко интернета у приватне сврхе	55,2%	54,6%	50,1%	45,4%
Коришћење рачунара у пословању (предузеће)	100%	99,3%	100%	99,8%
Интернет у предузећима	99,8%	99,8%	99,7%	99,8%
Предузеће поседује веб сајт	83,6%	82,6%	80,4%	80,8%
Наручивање производа/услуга преко интернета (предузеће)	/	42,3%	41,9%	41,4%
Примање поруџбина путем интернета	/	27,5%	26,3%	23,8%

Постоји незнатан напредак, али је потребно уложити додатне ресурсе (материјалне и људске) како би напредовали на листи светског економског форума. Постоји добра основа за планирање даљег развоја ИКТ, а самим тим и области програмирања, односно програмских језика.

2.3.2. ИТ и програмирање у предуниверзитетском образовању у Србији

Развијање ИТ сектора један је од приоритета Владе Републике Србије. Србији у овом тренутку недостаје око 15000 ИТ стручњака. Нови програм наставе и учења за опште средњошколско образовање, односно гимназије, уводи нови модел образовања који поред досадашње традиционалне предметне организације наставе и развоја предметних знања, способности и компетенција, уводи интердисциплинарне везе кроз изборне програме, развија интердисциплинарне и трансверзалне компетенције [60].

Креирањем одељења ученика са посебним способностима у области рачунарских наука и информатике, образовни систем Србије настоји да одговори на потребе тржишта рада и да на тај начин побољша квалитет студената који студирају информатичке технологије на различитим факултетима (средњошколци ће кроз специјалистичко образовање стећи дигиталну писменост, програмерске вештине...). Матуранти тих одељења ће, осим проходности на факултете који школују ИТ стручњаке, моћи да студирају и у другим областима које ће се дигитализовати (пољопривреда, здравство, енергетика...) [61].

2.3.2.1. Основно образовање

Информатика и рачунарство се до школске 2019/20. године у основној школи изучавало само у вишим разредима. Од школске 2020/21. године ученици првог разреда су добили нови обавезан предмет „*Дигитални свет*“ (1 час седмично, све 4 године), помоћу којих ће ученици развијати дигиталне компетенције како би могли безбедно и исправно да користе уређаје за учење, комуникацију, сарадњу и развој алгоритамског начина мишљења (*Просветни гласник LXIX – Број 2, 9. април 2020*).

У вишим разредима предмет Информатика и рачунарство је од школске године 2018/19 постао обавезан. Настава рачунарства и информатике се реализује искључиво кроз вежбе у рачунарским кабинетима. Ученици кроз предмет Информатика и рачунарство упознају рачунарско размишљање/вештине које изучавају од 5. до 8. разреда са годишњим фондом од: 36 часова у петом, шестом и седмом разреду и 34 часа у осмом разреду. Кроз овај предмет ученици треба да се оспособе за: управљање информацијама, сигурну комуникацију у дигиталном окружењу, креирање дигиталних садржаја и рачунарских програма – за решавање различитих проблема у друштву које се брзо мења са развојем дигиталних технологија.

У табели 2.3.3 [62] дат је фонд и садржај програма за овај предмет од 5. до 8. разреда.

- У петом разреду, изучавајући област Рачунарство, ученици се упознају са основама програмирања и блоковским програмирањем (*Scrtach, Stencil, AppInventor, Alice*,...), у теми Пројектна настава један од пројектних задатака је из области Рачунарства.

- У шестом разреду, изучавајући област Рачунарство, ученици се упознају са основама рада у програмском језику *Python*.
- У седмом разреду, изучавајући област Рачунарство, ученици се упознају са основама рада у изабраном графичком програмском језику.
- У осмом разреду, област Рачунарство је посвећена анализи и визуелизацији података помоћу *Python*-а у окружењу Џупитера (*Jupyter*). У оквиру теме Дигитална писменост, обрађује се наставна јединица о Вештачкој интелигенцији.

Табела 2.3.3: Школски програм за Информатику и рачунарство од 5. до 8. разреда (Извор: [62])

Разред	НАСТАВНА ТЕМА/ОБЛАСТ	БРОЈ ЧАСОВА ПРЕМА ТИПУ ЧАСА			Укупно часова
		Обрада	Вежба	Утврђивање и систематизација	
ПЕТИ	ИКТ	7	2	0	9
	Дигитална писменост	5	0	0	5
	Рачунарство	10	6	0	16
	Пројектна настава	0	0	6	6
ШЕСТИ	ИКТ	7	3	0	10
	Дигитална писменост	4	0	0	4
	Рачунарство	10	5	0	15
	Пројектна настава	0	0	7	7
СЕДМИ	ИКТ	7	3	0	10
	Дигитална писменост	4	0	0	4
	Рачунарство	10	5	0	15
	Пројектна настава	0	0	7	7
ОСМИ	ИКТ	7	3	0	10
	Дигитална писменост	2	0	0	2
	Рачунарство	10	2	0	12
	Пројектни задатак	0	0	6	10

2.3.2.2. Средње образовање

У средњим школама чији је оснивач Република Србија, постоје два образовна профила на којима се рачунарство и информационе технологије (самим тим и програмирање, односно програмски језици) уче у току целог школовања. То су образовни профили:

- ученици са посебним способностима за рачунарство и информатику у специјализованим ИТ одељењима у гимназијама (од школске године 2018/2019);
- електротехничар информационих технологија у техничким школама (од школске године 2012/2013).

Преглед планираних, односно уписаних ученика у гимназијама са ИТ одељењима за школске године 2019/2020 и 2020/2021 дат је у табели 2.3.4 [63]:

- 2019/2020 – 53 државне гимназије, 60 одељења, од планираних 1197 уписано је 1002 ученика (83,7%);
- 2020/2021 – 49 државних гимназија, 57 одељења, од планираних 1140 уписано је 990 ученика (86,8%).

Преглед планираних, односно уписаних ученика у средњим школама на четворогодишњи образовни профил електротехничар ИТ, за школске године 2019/2020 и 2020/2021, дат је у табели 2.3.5 [63]:

- 2019/2020 – 42 државне техничке школе, 49 одељења, од планираних 1457 уписано је 1409 ученика (96,7%);
- 2020/2021 – 42 државне техничке школе, 49 одељења, од планираних 1443 уписано је 1397 ученика (96,8%).

Табела 2.3.4: Списак гимназија које школују ИТ стручњаке (Извор: [63])

Гимназије са ИТ одељењима		2020/2021		2019/2020	
Назив	**Град**	**Плани-рано**	**Уписа-но**	**Плани-рано**	**Уписа-но**
1. Ваљевска гимназија	Ваљево	20	20	20	19
2. Гимназија „Патријарх Павле“	Раковица-Београд	40	38	40	40
3. Гимназија „Светозар Марковић“	Суботица	20	15	20	20
4. Гимназија „ Јован Јовановић Змај“	Нови Сад	60	60	60	60
5. Гимназија „9. мај“	Ниш	20	20	20	16
6. Гимназија „Бољаи“,(мађарски)	Сента	20	20	20	20
7. Гимназија „Бора Станковић“	Врање	20	20	20	20
8. Гимназија „Бора Станковић“	Ниш	40	40	40	40
9. Гимназија „Вељко Петровић“	Сомбор	20	20	20	20
10. Гимназија „Вук Караџић“	Лозница	20	20	20	14
11. Гимназија „Милош Савковић“	Аранђеловац	20	18	20	20
12. Гимназија „Свети Сава“	Београд	20	20	20	20
13. Гимназија „Свети Сава“	Пожега	40	32	20	20
14. Гимназија „Светозар Марковић“	Јагодина	20	2	20	13
15. Гимназија „Урош Предић“	Панчево	20	20	20	20
16. Гимназија Пирот	Пирот	20	18	20	13
17. Гимназија	Зајечар	20	19	20	19
18. Гимназија	Ивањица	20	20	20	18
19. Гимназија	Косовска Митровица	20	15	20	11
20. Гимназија	Краљево	20	20	20	20
21. Гимназија	Крушевац	20	20	20	20
22. Гимназија	Лесковац	20	20	20	20
23. Гимназија	Младеновац-Београд	20	19	20	20
24. Гимназија	Нови Пазар	20	20	20	20
25. Гимназија	Параћин	20	11	20	9
26. Гимназија	Прокупље	20	20	20	20
27. Гимназија	Рашка	20	17	20	18
28. Гимназија	Смедерево	20	20	20	20
29. Гимназија	Ћуприја	20	13	20	19
30. Гимназија	Чачак	20	8	20	20
31. Девета гимназија „Михаило Петровић Алас“	Београд	20	20	20	20
32. Друга крагујевачка гимназија	Крагујевац	20	8	19	13
33. Земунска гимназија	Земун-Београд	20	20	20	20
34. Зрењанинска гимназија	Зрењанин	20	18	20	20
35. Митровачка гимназија	Сремска Митровица	20	9	20	20
36. Осма београдска гимназија	Београд	40	27	39	39
37. Паланачка гимназија	Смедеревска Паланка	20	17	20	16
38. Прва београдска гимназија	Београд	20	20	40	40
39. Прва крагујевачка гимназија	Крагујевац	60	53	39	39
40. Пријепољска гимназија	Пријепоље	20	13	20	20
41. Средња школа Младост	Петровац на Млави	20	11	20	2
42. Средња школа	Брус	20	13	20	10
43. Трећа београдска гимназија	Врачар-Београд	20	20	20	20
44. Ужичка гимназија	Ужице	20	16	20	20
45. Шабачка гимназија	Шабац	20	20	20	20
46. Шеста београдска гимназија	Звездара-Београд	20	20	20	20
47. Гимназија „Вук Караџић“	Трстеник	20	20	0	0
48. Гимназија	Пожаревац	20	20	0	0
49. Дванаеста београдска гимназија	Београд	20	20	0	0
50. Алексиначка гимназија	Алексинац	0	0	20	2
51. Гимназија „Душан Васиљев“	Кикинда	0	0	20	5
52. Гимназија „Јан Колар“ са домом ученика, (словачки)	Бачки Петровац	0	0	20	0
53. Гимназија „Стеван Пузић“	Рума	0	0	20	5
54. Гимназија и економска школа „Бранко Радичевић“	Ковин	0	0	20	6
55. Неготинска гимназија	Неготин	0	0	20	11
56. Средња школа „Лукијан Мушицки“	Темерин	0	0	20	5
		1140	**990**	**1197**	**1002**

Табела 2.3.5: Списак средњих стручних школа које школују ИТ стручњаке (Извор: [63])

Средње стручне школе са профилом Електротехничар ИТ		2020/2021		2019/2020	
Назив	**Град**	**Планирано**	**Уписано**	**Планирано**	**Уписано**
1. Гимназија „Михајло Пупин“	Ковачица	30	21	30	30
2. Електро-саобраћајна техничка школа „Никола Тесла“	Краљево	30	30	30	30
3. Електротехничка и грађевинска школа „Никола Тесла“	Зрењанин	30	30	30	30
4. Електротехничка и грађевинска школа „Никола Тесла“, (мађарски)	Зрењанин	30	19	30	19
5. Електротехничка и грађевинска школа „Никола Тесла“	Јагодина	30	30	30	30
6. Електротехничка школа „Михајло Пупин“	Нови Сад	59	59	57	57
7. Електротехничка школа „Земун“	Земун	56	56	60	60
8. Електротехничка школа „Мија Станимировић“	Ниш	54	54	57	57
9. Електротехничка школа „Никола Тесла“	Београд	59	59	57	57
10. Електротехничка школа „Никола Тесла“	Ниш	60	60	60	60
11. Електротехничка школа „Никола Тесла“	Панчево	30	30	30	30
12. Електротехничка школа „Раде Кончар“	Београд	59	59	57	57
13. Машинско-електротехничка школа „Гоша“	Смедеревска Паланка	30	30	30	30
14. Машинско-електротехничка школа	Прибој	30	30	30	30
15. Прва техничка школа	Крагујевац	29	29	30	30
16. Техничка школа „Раде Металац“	Лесковац	30	30	30	30
17. Техничка школа „9. Мај“	Бачка Паланка	30	30	30	30
18. Техничка школа „Милета Николић“	Аранђеловац	30	30	30	30
19. Техничка школа Иван Сариц	Суботица	30	30	30	30
20. Техничка школа Иван Сариц, (мађарски)	Суботица	28	28	30	30
21. Техничка школа „Михајло Пупин“	Инђија	30	30	30	30
22. Техничка школа „Прота Стеван Димитријевић“	Алексинац	30	30	30	30
23. Техничка школа	Ваљево	27	27	30	31
24. Техничка школа	Уб	30	30	30	30
25. Техничка школа	Бечеј	30	30	30	30
26. Техничка школа	Зајечар	30	30	30	27
27. Техничка школа	Кикинда	30	30	30	30
28. Техничка школа	Младеновац	29	29	30	30
29. Техничка школа	Неготин	30	30	30	25
30. Техничка школа	Нови Пазар	30	27	30	30
31. Техничка школа	Пирот	30	30	30	30
32. Техничка школа, (албански)	Прешево	30	14	30	30
33. Техничка школа	Смедерево	30	30	30	30
34. Техничка школа	Ужице	27	27	30	29
35. Техничка школа	Чачак	60	60	59	59
36. Средња техничка школа „Михајло Пупин“	Кула	30	30	30	30
37. Средња техничка школа	Сомбор	30	30	30	30
38. Средња школа „Ђура Јакшић“	Рача	30	30	30	30
39. Средња школа „Никета Ремезијански“ са домом ученика	Бела Паланка	30	23	30	18
40. Техничка школа	Мајданпек	30	29	30	13
41. Електротехничка школа „Стари Град“	Београд	26	27	0	0
42. Техничка-пољопривредна школа	Сјеница	30	30	0	0
43. Школски центар „Никола Тесла“	Вршац	0	0	30	30
44. Техничка школа	Шабац	0	0	30	30
		1443	**1397**	**1457**	**1409**

Осим државних школа у којима се школује средњошколски ИТ кадар, отворене су и приватне средње школе, као што су:

- Рачунарска гимназија у Београду, школује ученике по програму наставе и учења из 2018. године за надарене ученике у рачунарској гимназији;
- Рачунарска гимназија „SMART“ Нови Сад, школује ученике по програму наставе и учења из 2004. године за надарене ученике у рачунарској гимназији;
- Средња школа за информационе технологије (IHTS) у Београду, школује образовни профил електротехничар информационих технологија.

У табели 2.3.6 [64] дат је преглед информатичких предмета у одељењима гимназија за ученике са посебним способностима за рачунарство и информатику (ИТ одељења), са бројем часова вежби и блоковске наставе, као и кратким садржајем програма за те предмете (по годинама). У току свог школовања ученици ових специјализованих ИТ одељења имају тринаест (13) ИТ предмета (у првом разреду 3 предмета, другом – 3, трећем – 4, четвртом – 3).

Табела 2.3.6: ИТ предмети у ИТ одељењима гимназија (Извор: [64])

Р.Б.	Предмет	Разред	Број часова годишње		Садржај предмета (број часова)
			Вежбе	Блоковска настава	
1.	Примена рачунара	први	108		Основи информатике (12), Архитектура рачунарског система (4), Програмска подршка рачунара (14), Основе рада у оперативном систему са графичким интерфејсом (14), Обрада текста уз помоћ рачунара (18), Слајд-презентације (16), Рад са табелама (18), Интернет и електронска комуникација (12)
2.		други	72		Рачунарска графика (28), Мултимедија (18), Презентације на интернету (18), Рачунарство и друштво (18)
3.		трећи	70		Примена рачунара у математици (34), Примена рачунара у разним областима (12), Рачунарска графика – напреднији курс (24)
4.	Програми-рање	први	108	30	Појам и примери алгоритама (20), Основни концепти програмских језика и окружења за развој програма (10), Основни алгоритми линијске и разгранате структуре (20), Основни алгоритми цикличке структуре (30), Детаљни преглед основних типова података (променљивих, константи, оператора и израза) (8), Низови, ниске и основни алгоритми за рад са њима (20)
5.		други	108	30	Вишедимензиони низови, матрице и основни алгоритми за рад са њима (12), Кориснички дефинисани типови (6), Улаз и излаз програма (8), Анализа алгоритама (6), Опште технике конструкције алгоритама (38), Динамичке структуре података и апстрактни типови података (38)
6.		трећи	70	30	Графови и алгоритми за рад са графовима (20), Алгоритми текста (10), Геометријски алгоритми (10), Алгоритми теорије бројева (10), Алгоритми над битовима (4), Преглед одабраних структура података и алгоритама (16)
7.	Рачунарски системи	први	72		Увод у рачунарске системе (12), Дигитални запис података (16), Логичке основе обраде података (14), Основи архитектуре и организације рачунара (14), Асемблерско програмирање (14)
8.	Оперативни системи и рачунарске мреже	други	72		Увод у оперативне системе (8), Процеси (6), Конкурентност и синхронизација процеса (10), Заглављивање (6), Управљање меморијом (12), Систем датотека (8), Управљање улазно-излазним уређајима (10), Рачунарске мреже (12)
9.	Објектно оријентисано програмирање	трећи	105	30	Настанак и карактеристике објектно оријентисаног програмирања (9), Основни појмови објектно оријентисаног програмирања (24), Везе између класа и полиморфизам (30), Израда пројектног задатка (42)
10.	Базе података	трећи	70		Пројектовање база података, (24) Релационе базе података (14), Упитни језик SQL (32)
11.		четврти	62	30	Програмирање и базе података (46), Друге актуелне технологије (16)
12.	Програмске парадигме	четврти	62		Увод (4), Исказна логика (10), Предикатска логика (12), Логичко програмирање (22), Функционално програмирање (14)
13.	Веб програмирање	четврти	62		Рачунарске мреже (4), Интернет сервиси и протоколи (4), Описни језик HTML (8), Стилски листови - језик CSS (8), Скрипт језик JavaScript за клијентско програмирање (8), Серверско програмирање (30)
			1041	**150**	

У табели 2.3.7 [65] дат је преглед информатичких предмета за образовни профил електротехничар информационих технологија. Ученици ових профила у току 4 разреда школовања имају укупно од 19 до 21 ИТ предмета (19 обавезних + 2 изборна; у првом разреду 3, другом – 4, трећем и четвртом разреду имају 6 или 7 ИТ предмета (зависно од избора)).

Табела 2.3.7: ИТ предмети на образовном профилу електротехничар информационих технологија (Извор: [65])

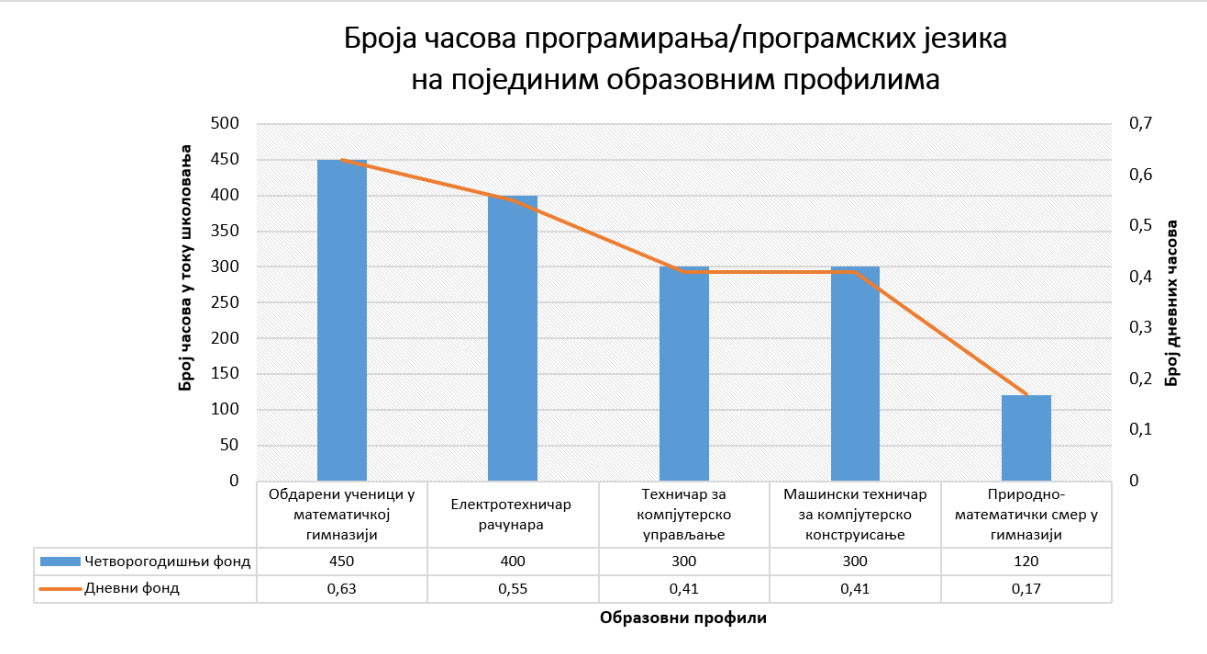
Р.Б.	Предмет	Разред	Број часова годишње		Садржај предмета-теме (број часова)
			Вежбе	Блоковска настава	
1.	Рачунарство и информатика	први	74		Основе информатике (8), Архитектура рачунарских система (6), Програмска подршка рачунара (2), Основе рада у оперативном систему са графичким интерфејсом (12), Текст процесор (12), Мрежне информационе технологије (10), Интернет (12), Слајд-презентације (12)
2.	Рачунарска графика и мултимедија	први	74		Увод у рачунарску графику и мултимедију (2), Рад са текстом (6), Обрада слике на рачунару (30), Обрада звука (10), Обрада видео записа (8), Анимације (10), Израда самосталног пројекта (8)
3.	Програмирање	први	74		Увод (4), Алгоритми (30), Структура језика и типови података (8), Изрази и наредбе (8), Разгранате програмске структуре (8), Цикличне програмске структуре (8), Једнодимензионални низови (8)
4.		други	70+70		Једнодимензионални низ или вектор (8+8), Показивачи (8), Функције (14+14), Вишедимензионални низови (14+14), Стрингови и текстуалне датотеке (20+20), Структурне и бинарне датотеке (10+10)
5.		трећи	70+70		Основни концепти објектно оријентисаног језика (4+4), Објектни језик и C језик (6+6), Класе (8+8), Изведене класе (10+10), Руковање изузецима (6+6), Библиотека компонената (36+36)
6.		четврти	93		Динамичке структуре података (15), Генеричке класе (15), Класа за рад са стринговима и класа за рад са датумом (9), Менији, дијалози и рад са више форми (9), Датотеке (9), Графика и нити (15), Базе података (21)
7.	Апликативни програми	други	70		Рад са табелама (30), Рад са базама података (20), Програм за презентације (10), Програми за прорачуне у науци (10)
8.	Рачунарски хардвер	други	70		Архитектура рачунара (28), Склапање рачунара (28), Тестирање хардвера (14)
9.	Веб дизајн	други	70		Увод у веб дизајн (4), Основе HTML језика (20), CSS (20), Визуелна израда интернет презентације (Dreamweaver) (26)
10.	Веб програмирање	трећи	105		Увод у веб програмирање (10), Објектно оријентисано програмирање (10), Серверски скрипт језици (40), Клијентски скрипт језици (30), Веб сервери (15)
11.		четврти	93		Програмирање база података на вебу (30), XML веб сервиси и серверске компоненте (30), MVC - Model View Control (33)
12.	Оперативни системи	трећи	70		Структура и функције оперативног система (18), Инсталирање оперативног система (16), Конфигурисање оперативног система (6), Инсталирање и уклањање додатног софтвера и хардвера (8), Одржавање оперативног система (22)
13.	Рачунарске мреже и комуникације	трећи	70+35		Принципи рачунарских комуникација (10+2), Мрежна комуникација и протоколи (18+8), Уређаји за повезивање (16+10), Мрежне технологије и умрежавање (26+15)
14.	Информациони системи и базе података	трећи	105+70		Увод у информационе системе (7), Пројектовање информационих система (20), Увод у базе података и СУБП (8), Модел објекти-везе и релациони модел базе података (40), Основни елементи упитног језика SQL (70), Основе ADO.NET-а (30)
15.	Интернет технологије и сервиси	четврти	62+31		Интернет технологије (34+17), Интернет сервиси (28+14)
16.	Заштита информационих система	четврти	62+31		Основи криптологије (16+8), Контрола приступа (8+4), Сигурност рачунарских мрежа (14+7), Сигурност оперативних система (8+4), Сигурност софтвера/апликација/информационих система (10+5), Детекција и превенција напада (6+3)
17.	Електронско пословање	четврти	62+31		Увод у електронско пословање (4+1), Електронски пословни системи (10+0), Електронско пословање у трговини (10+12), Електронско пословање у банкарству (10+9), Електронски маркетинг (6+0), Електронско пословање у јавној управи (6+0), Електронско образовање (6+6), Електронско здравство (4+0), Мобилно електронско пословање (6+3)
18.	Практична настава	трећи		60	Веб програмирање (12), Објектно оријентисано програмирање (18), Информациони системи и базе података (18), Рачунарске мреже и комуникације (12)
19.		четврти		90	Веб програмирање (30), Програмирање (30), Заштита информационих система (18), Електронско пословање (12)
20/21. Изборни предмет			**1356+337**	**150**	
	Софтверски мултимедијални алати	трећи	70		Креирање анимација (34), Обрада видео сигнала (36)
	Базе података	четврти	62		Администрација SQL server-а (30), Програмирање SQL server-а (32)
	Рачунари у системима управљања	четврти	31+31		Основе управљања и система управљања (8+6), Рачунари у системима непосредног управљања и надзора (3+0), Систем за надзорно управљање и аквизицију података (SCADA) (3+6), Технике комуникација у системима управљања (8+4)

У табели 2.3.8 [64, 65] дат је компаративни преглед броја часова из ИТ предмета и програмирања/програмских језика за два образовна профила: Гимназија – ученици са посебним способностима за рачунарство и информатику, и Средња стручна школа – електротехничар за информационе технологије.

Табела 2.3.8: Број часова из ИТ предмета и програмирања (Извор: [64, 65])

Образовни профил	Број часова			
	ИТ предмети		Програмирање	
	за 4 године	дневно	за 4 године	дневно
Електротехничар информационих технологија	1900	2,62	1000	1,38
Гимназија за ученике са посебним способностима за рачунарство и информатику	1200	1,67	800	1,11

Осим ова два образовна профила, постоји још неколико профила у средњим школама у Србији на којима се учи програмирање и програмски језици, али са мањим фондом часова, слика 2.3.1 [66].



Слика 2.3.1: Програмирања на средњошколским образовним профилима који не школују ИТ кадар (Извор: [66])

2.3.2.3. Истраживање „Употреба ИТ у средњим школама Србије“

У другом полугодишту школске године 2016/17 аутор ове дисертације, за потребе докторских студија, извршио је истраживање „*Употреба ИТ у средњим школама Србије*“ са циљем да се сагледа стање у средњим школама по питању употребе информационих технологија, са посебним акцентом на програмирање. Истраживање је извршено у 22 града Србије, у 9 гимназија и 22 средње стручне школе, које имају образовне профиле на којима се ИТ и програмирање изучавају у одређеном фонду. Укупно је анкетирано 1794 ученика трећег и четвртог разреда, као и 244 наставника који предају информатичке предмете у тим школама. У табели 2.3.9 дат је приказ броја анкетираних ученика и наставника по градовима и образовним профилима.

Табела 2.3.9: Истраживање „Употреба ИТ у средњим школама Србије“ (Извор: аутор)

Истраживање је извршено у периоду фебруар – јун 2017. године	*Образовни профили*: 1-Гимназија – природно математички смер; 2-Гимназија – надарени ученици у математичкој гимназији; 3-Гимназија – надарени ученици у рачунарској гимназији; 4- Гимназија – информатички смер; 5-Електротехника – електротехничар рачунара; 6-Електротехника – електротехничар мултимедија; 7-Електротехника – администратор рачунарских мрежа; 8-Електротехника – електротехничар информационих технологија; 9-Машинство и обрада метала – машински техничар за компјутерско конструисање; 10-Машинство и обрада метала – техничар за компјутерско управљање *НАС* – Наставници; *УЧ* – Ученици																					
	Образовни профили																				**Укупно**	
	1		**2**		**3**		**4**		**5**		**6**		**7**		**8**		**9**		**10**			
Град	**НАС**	**УЧ**	**НАС**	**УЧ**	**НАС**	**УЧ**	**НАС**	**УЧ**	**НАС**	**УЧ**	**НАС**	**УЧ**	**НАС**	**УЧ**	**НАС**	**УЧ**	**НАС**	**УЧ**	**НАС**	**УЧ**	**НАС**	**УЧ**
Београд									12	112	3	52	2	20	12	59					**29**	**243**
Ваљево	2	19	2	11																	**4**	**30**
Врање	7	35					2	19	2	20									4	22	**18**	**96**
Врњачка Бања							4	29													**4**	**29**
Зајечар	2	25					2	21	2	20					3	22	5	17			**14**	**105**
Земун									6	18			6	18	1	14					**13**	**50**
Јагодина									2	16	2	11			3	15					**7**	**42**
Кладово	1	8							3	27											**5**	**35**
Крагујевац	5	46	2	22	2	17			1	26		10	1	7	1	14					**12**	**142**
Краљево	6	23	3	13					4	25					6	21	3	37	3	47	**25**	**166**
Лесковац									3	25	4	20			4	22	2	21	2	21	**15**	**109**
Ниш									3	46					7	21					**10**	**67**
Нови Пазар	2	32					3	30													**5**	**62**
Нови Сад									2	19	5	59			4	23					**11**	**101**
Панчево									1	18	1	19	3	18	3	25					**8**	**80**
Пожега																	4	15	4	28	**8**	**43**
Прибој									5	30							2	20	2	22	**9**	**72**
Пријепоље								1		17							1	20			**1**	**38**
Рума									3	26							1	11			**4**	**37**
Смедерево									3	21			3	18			1	23			**7**	**62**
Ужице									2	14					7	25	3	16			**12**	**55**
Чачак	6	49							6	26					6	21	3	17	2	17	**23**	**130**
Укупно	**32**	**237**	**7**	**46**	**2**	**17**	**11**	**100**	**63**	**506**	**15**	**171**	**15**	**81**	**57**	**282**	**25**	**197**	**17**	**157**	**244**	**1794**

Упитник за ученике је био организован у три дела:

- *Први део*: Општи подаци о ученику/-ци – назив школе, место школе, тип школе и образовни профил (ОП);
- *Други део*: Употреба ИТ у образовању – ученици су од понуђених одговора заокруживали тврдњу (једну или више) која је највише одражавала њихово мишљење на постављено питање (одлучујући разлог за упис ОП, да ли би опет уписали ОП, утицај ОП на избор факултета, стечено знање као помоћ у даљем школовању, факултет који желе да упишу, највећи квалитет ОП, недостаци ОП);
- *Трећи део*: Процене – ученици су заокруживањем оцене процењивали понуђене тврдње које су се односиле на разне аспекте школовања посебно на наставу информатике и програмирање (квалитет наставе информатичких предмета, квалитет ОП, знање из одређених области ИТ, важност одређених области ИТ).

Упитник за наставнике је такође био организован у три дела:

- *При део*: Општи подаци – школа у којој раде (место, назив, образовни профил, којим разредима предају), године радног стажа у настави, завршен факултет, академско звање;
- *Други део*: Употреба ИТ у образовању – наставници су заокруживали тврдње које су највише одражавале њихово мишљење (информатичке вештине, професионално искуство у ИТ, највећи квалитет ОП на коме предају, недостаци ОП, приоритетне иновативности за унапређење ОП);

- *Трећи део*: Процене – процењивања тврдњи (квалитет ОП, допринос ИТ ефикаснијој реализацији часова, допринос ИТ повећању постигнућа ученика, њихово знање из одређених области ИТ, важност одређених области ИТ).

Ученици су процењивали важност одређених области ИТ оценама *1 (неважно), 2 (важно)* и *3 (неопходно)*. Њихове процене важности (тренутно и у будућности) ИТ и програмских језика, по образовним профилима, дате су на сликама 2.3.2 и 2.3.3.

Слика 2.3.2: Процена важности Информационих технологија (ученици) (Извор: аутор)

Слика 2.3.3: Процена важности Програмских језика (ученици) (Извор: аутор)

Анализа ученичких процена:

- Радује то што, ученици свих образовних профила сматрају да ће информационе технологије у будућности имати још важнију улогу у развоју дигиталног света (просечна процена: сада – 2,13; у будућности – 2,31), док ће знање програмских језика, као област ИТ, у будућности бити веома важно (просечна оцена: сада – 2,23; у будућности – 2,42);
- Надарени ученици у математичкој гимназији сматрају да ће у будућности познавање нових технологија бити веома важно (2,46), док је знање програмских језика скоро неопходно (2,84);
- Електротехничари мултимедија су важност ИТ и програмских језика у будућности најмање вредновали (ИТ – 2,19; програмски језици – 2,14). Такође, они су један од два образовна профила који је дао предност важности ИТ у односу на важност програмских језика (други ОП је природно математички смер у гимназији).

Процене важности ИТ и програмских језика од стране наставника који предају информатичке предмете дате су на слици 2.3.4.

Слика 2.3.4: Процена важности ИТ и програмских језика (наставници) (Извор: аутор)

Анализа наставничких процена:

- Наставници су важност ИТ и програмских језика оценили високо (ИТ – 2,32; програмски језици – 2,34);
- Наставници који предају на образовним профилима Машински техничар за компјутерско конструисање, Електротехничар мултимедија и Администратор рачунарских мрежа сматрају да је познавање ИТ важније од знања програмских језика, док наставници са осталих образовних профила процењују да је развој програмских језика важнији од ИТ;
- Наставници електротехничара мултимедије су, као и њихови ученици, најмање вредновали важност ИТ (2,28) и програмских језика (1,93);
- Наставници који раде у гимназијама са надареним ученицима (њих 9) сматрају да је знање програмских језика веома важно (просечна оцена 2,56).

Упоређујући анализе ученичких и наставничких процена важности информационих технологија и програмских језика у данашњем модерном дигиталном свету, видимо да се сви слажу да је неопходно усавршавати знања из информационих технологија, а посебно из области програмских језика. Такође, у корелацији су резултати процене наставника и ученика по образовним профилима. Тамо где наставници сматрају да је важно/неопходно улагати у додатна знања ученика по питању програмирања, то резултује и жељом ученика да у тој области напредују (рачунарска и математичка гимназија са надареним ученицима). Наставници електротехничар мултимедије сматрају да програмирање није толико важно за тај образовни профил, такве су процене и ученика тог образовног профила. Ученици и наставници образовног профила електротехничар информационих технологија имају сличне процене важности.

Ученици су свесни да је неопходно да своја знања иновирају и увећавају у свим областима, посебно у информационим технологијама и програмским језицима. Неопходно је да наставници буду на истом колосеку, како би припремили своје ученике за наставак њиховог образовног развоја и што лакши прикључак ИТ свету.

2.3.3. ИТ и програмирање у високошколском образовању

Очекује се да ће се до 2022. године број рачунарских програма повећати за 37%. Велики број универзитета у свету, на различитим студијским програмима, образује студенте из области рачунарства и информационих технологија. *Times Higher Education* је за 2020. годину, на основу 13 индикатора учинка, рангирао 750 универзитета у свету. У табели 2.3.10 [67] дат је преглед 100 најбољих универзитета који образују студенте рачунарства и ИТ, као и преглед универзитета у региону. У 2020. години београдски универзитет није на листи (у 2019. години био је рангиран у групи универзитета од 501. до 600. места). Бугарска, такође нема универзитете на листи (ни 2019. године), три универзитета из Румуније нашла су се ове године на листи, док 2019. године Румунија није имала рангиране универзитете. Италија има укупно 28 универзитета на листи (*Polytechnic University* из Милана је на 97 месту) [67].

Табела 2.3.10: Најбољи универзитети у свету за студирање рачунарства и ИТ (Извор: [67])

Ранг 2020.	Универзитет	Држава
1	University of Oxford	United Kingdom
2	Stanford University	USA
3	ETH Zurich	Switzerland
4	Massachusetts Institute of Technology	USA
5	University of Cambridge	United Kingdom
6	Carnegie Mellon University	United States
7	Imperial College London	United Kingdom
8	Harvard University	United States
9	Princeton University	United States
10	California Institute of Technology	United States
11	National University of Singapore	Singapore
12	University of California, Los Angeles	United States
13	Nanyang Technological University, Singapore	Singapore
14	Cornell University	United States
15	Tsinghua University	China
16	Georgia Institute of Technology	United States
17	The Hong Kong University of Science and Technology	Hong Kong
18	Technical University of Munich	Germany
19	UCL	United Kingdom
20	École Polytechnique Fédérale de Lausanne	Switzerland
21	Columbia University	United States
22	University of Michigan-Ann Arbor	United States
23	University of Toronto	Canada
24	University of Edinburgh	United Kingdom
25	University of Texas at Austin	United States
26	University of Washington	United States
27	Peking University	China
28	Yale University	United States
29	University of Illinois at Urbana-Champaign	United States
30	Johns Hopkins University	United States
31	University of Montreal	Canada
31	University of Pennsylvania	United States
33	New York University	United States
34	Paris Sciences et Lettres – PSL Research University Paris	France
35	California, San Diego	United States
36	Chinese University of Hong Kong	Hong Kong
37	University of Southern California	United States
38	University of Hong Kong	Hong Kong
39	University of Chicago	United States
40	University of Waterloo	Canada
41	Zhejiang University	China

Ранг 2020.	Универзитет	Држава
42	The University of Tokyo	Japan
43	University of British Columbia	Canada
44	Korea Advanced Institute of Science and Technology (KAIST)	South Korea
45	Shanghai Jiao Tong University	China
46	RWTH Aachen University	Germany
47	Delft University of Technology	Netherlands
47	Seoul National University	South Korea
49	University of Maryland, College Park	United States
50	Karlsruhe Institute of Technology	Germany
50	University of Wisconsin-Madison	United States
52	KU Leuven	Belgium
53	Duke University	United States
54	University of California, Santa Barbara	United States
55	Technical University of Berlin	Germany
56	Aalto University	Finland
56	University of Melbourne	Australia
58	LMU Munich	Germany
59	Eindhoven University of Technology	Netherlands
60	Rice University	United States
61	University of Science and Technology of China	China
62	University of California, Irvine	United States
63	École Polytechnique	France
64	Technion Israel Institute of Technology	Israel
65	University of Freiburg	Germany
66	McGill University	Canada
67	National Taiwan University	Taiwan
68	Technical University of Darmstadt	Germany
69	Australian National University	Australia
69	KTH Royal Institute of Technology	Sweden
71	Pohang University of Science and Technology (POSTECH)	South Korea
72	Nanjing University	China
72	Vrije Universiteit Amsterdam	Netherlands
74	Brown University	United States
74	ITMO University	Russian
76	TU Wien	Austria
77	University of Massachusetts	United States
78	University of Amsterdam	Netherlands
79	King's College London	United Kingdom
79	University of Technology Sydney	Australia
79	UNSW Sydney	Australia
82	Kyoto University	Japan

Ранг 2020.	Универзитет	Држава
83	Northwestern University	United States
83	University of Zurich	Switzerland
85	University of Manchester	United Kingdom
86	Queensland University of Technology	Australia
86	University of Sydney	Australia
88	Sorbonne University	France
89	Virginia Polytechnic Institute and State University	United States
90	University of Luxembourg	Luxembourg
91	Boston University	United States
92	Chalmers University of Technology	Sweden
92	Heidelberg University	Germany
92	Penn State (Main campus)	United States
95	Moscow Institute of Physics and Technology (MIPT)	Russian Federation
96	University of Adelaide	Australia
97	Polytechnic University of Milan	Italy
98	Hong Kong Polytechnic University	Hong Kong

Ранг 2020.	Универзитет	Држава
99	Michigan State University	United States
100	Aarhus University	Denmark
601+	University of Zagreb	Croatia
401-500	University of Ljubljana	Slovenia
501-600	University of Maribor	Slovenia
601+	Babeş-Bolyai University	Romania
601+	University of Bucharest	Romania
601+	Technical University of Cluj-Napoca	Romania
301-400	Athens University of Economics and Business	Greece
401-500	University of the Aegean	Greece
401-500	Aristotle University of Thessaloniki	Greece
401-500	National and Kapodistrian University of Athens	Greece
501-600	University of Crete	Greece
501-600	University of Thessaly	Greece
501-600	University of Patras	Greece
601+	University of Ioannina	Greece
601+	Eötvös Loránd University	Hungary
601+	University of Szeged	Hungary

Избор свог будућег позива је један од најважнијих избора младих људи. ИТ стручњаци релативно лако проналазе посао, добро су плаћени и уз константну едукацију и преданост професионални напредак је неминован, што резултује и већим примањима. Добри српски ИТ стручњаци су цењени и тражени у свету. ИТ сектор омогућује и рад за компаније (и стране) које су стотинама километара удаљене од ИТ стручњака. Због тога код средњошколаца постоји велико интересовање за студирање на факултетима и вишим школама на којима се студира информационе технологије и рачунарство. Захваљујући реформама школства (основно и средње образовање) све је већи број средњошколаца који располажу и владају основним програмерским технологијама, што им омогућава лакши избор при упису на факултете информационих технологија и рачунарства. Посебно су изазовни студијски програми за звања *software developer* (програмер) које омогућава развијање апликација за: мобилне телефоне, индустрију, банкарство, здравство, видео-игрице, забаву...

Студирање на појединим факултета ИТ и рачунарства изискује неопходно предзнање из програмирања, па се на те факултете уписују првенствено средњошколци ИТ образовних профила. Постоје и факултети на којима се креће од основних неопходних програмерских вештина, па се на те факултете уписују средњошколци образовних профила који у току средње школе нису имали довољан број часова или нису уопште имали часове из програмирања (табела 2.3.11 [68]).

Табела 2.3.11: Факултети у Србији са ИТ студијским програмима (Извор: [68])

Назив факултета	Град	Студијски програми
Електротехнички факултет	Београд	- Софтверско инжењерство - Рачунарска техника и информатика
Факултет организационих наука	Београд	- Информациони системи и технологије
Математички факултет	Београд	- Информатика - Астроинформатика
Електронски факултет	Ниш	- Рачунарство и информатика
Природно-математички факултет	Нови Сад	- Математика и информатика
Технички факултет	Зрењанин	- Информационе технологије - Информационе технологије – софтверско инжењерство
Факултет техничких наука	Чачак	- Информационе технологије (академске) - Информационе технологије (струковне) - Софтверско и рачунарско инжењерство

Назив факултета	Град	Студијски програми
Природно-математички факултет	Крагујевац	- Информатика
Факултет техничких наука	К. Митровица	- Електротехника и рачунарство
Природно-математички факултет	К. Митровица	- Информатика
Рачунарски факултет (Универзитет Унион)	Београд	- Рачунарске науке - Рачунарско инжењерство - Информационе технологије (струковне)
Факултет за компјутерске науке (Мегатренд универзитет)	Београд	- Информационе технологије
Факултет за математику и рачунарске науке (Алфа БК Универзитет)	Београд	- Рачунарске науке
Факултет информационих технологија (Алфа БК Универзитет)	Београд	- Информационе технологије
Факултет информационих технологија (Универзитет Метрополитан)	Београд	- Информационе технологије - Софтверско инжењерство
Факултет за информатику и рачунарство (Универзитет Сингидунум)	Београд	- Информационе технологије - Информатика и рачунарство

У Србији је крајем 2019. године успешно пословало преко 2400 ИТ фирми, на тржишту рада недостаје скоро 40000 програмера, велики број школованих људи преквалификацијама прелази из разних сфера пословања у ИТ. Програмирање не подразумева само пуко писање кодова за паметне телефоне, рачунаре, разне аутоматизоване машине. Професија програмера има много подручја и специјализација, од којих свака има своје карактеристике и потешкоће. Факултети су акредитовали ИТ програме које ће произвести ИТ стручњаке за разна подручја активности [69]:

- **Front-end програмер**. Ови програмери су се специјализовали за интерфејсе и веома су тражени у компанијама који се баве израдом веб страница и развојем апликација. Неопходно је да знају: *HTML, CSS, JavaScript*.
- **Back-end програмер**. *Back-end* програмер креира архитектуру, смишља логику, пише извршни кôд. Обавезно морају да знају *JavaScript* и *MySQL*, као и неки од програмских језика: *PHP, Python, Java, Go, Ruby*.
- **Full-stack програмер**. Ово су програмери који комбинују вештине и знања *Front-end* и *Back-end* програмера.
- **Game Developer**. Ови програмери се баве развојем игара од почетка, подршком, исправљањем грешака и другим ажурирањима. Морају да знају програмске језике: *C/C++, C#* (или неки други програмски језик из те класе), *Java*, *Open GL* (или *DirectX*).
- **Android програмер**. Креирају апликације за мобилне уређаје са *Android* оперативним системом. Неопходно је да знају: *Android Studio, Java, OpenGL, Android SDK*.
- **iOS програмер**. Креира различите мобилне апликације за *iOS* (оперативне системе за *Apple* уређаје). *iOS* програмер мора да зна: *Swift* или *Objective-C, CoreData, Xcode, OpenGL, Cocoa Touch, CoreGraphics*.
- **Софтверски инжењер**. Они се најчешће баве развојем софтвера за аутоматизацију производње: CNC машине и транспортне линије. Морају да знају програмске језике: *C/C++, C#* и друге, да познају програм ниског нивоа (*Assembler*) и да поседују техничка знања која су везана за индустрију у којој програмер ради.

2.3.3.1. Програмирање на Факултету техничких наука у Чачку

На Факултету техничких наука (ФТН) у Чачку постоји више студијских програма на којима се изучава програмирање. Информационе технологије (ИТ) на академским студијама је студијски програм на коме је програмирање највише заступљено.

Године 2020. на ФТН у Чачку иновиран је акредитовани студијски програм основних академских студија Информационе технологије. На овом студијском програму проблематика програмирања се изучава од првог семестра и до краја школовања студентима се пружа могућност да кроз двоцифрен број предмета стекну и усаврше своје програмерске компетенције (табела 2.3.12 [70]).

Табела 2.3.12: ФТН Чачак, Основне академске студије Информационе технологије – предмети који развијају програмерске компетенције (Извор: [70])

Р.Б.	Назив предмета	Сем.	О/И	Шта се изучава
1.	Увод у програмирање	први	О	C
2.	Практикум из програмирања	први	О	C
3.	Практикум из рачунарских апликација	други	О	Основе HTML-a, основе Matlab-a
4.	Програмски језици	други	О	C (показивачи), C++
5.	Базе података	трећи	О	MS Access, Oracle Application Express
6.	Објектно оријентисано програмирање	трећи	О	C#, Java
7.	Структуре података и алгоритми	трећи	О	C, Java
8.	Рачунарска графика	трећи	О	Autodesk Maya, Adobe Illustrator, C++, Python
9.	Софтверски алати	четврти	И	Matlab
10.	Рачунарско моделовање физичких појава	четврти	И	Mathematica, Origin
11.	Рачунарске симулације	пети	О	Autodesk Maya, 3D Studio Max
12.	Веб технологије	пети	О	HTML5, CSS3, JavaScript, AJAX, XML
13.	Електронско пословање	шести	О	ASP.NET, PHP, MySQL
14.	Напредно објектно оријентисано програмирање	шести	О	Java
15.	Програмирање база података	шести	О	PL/SQL, Oracle базе
16.	Практикум из база података	шести	О	Oracle Application Express
17.	Софтверско инжењерство	седми	О	UML, C++, C#, Java
18.	Тестирање софтвера	седми	О	Алати за тестирање софтвера
19.	Интернет интелигентних уређаја	седми	О	Софтверски алати за креирање IoT
20.	Програмирање мобилних апликација	седми	И	Развој Андроид апликација
21.	Визуелно програмирање	седми	И	Visual C#.NET
22.	Мултимедијални системи	седми	И	Camtasia Studio, Adobe Flash
23.	Интернет програмирање	осми	О	Java, PHP
24.	Савремене софтверске архитектуре	осми	О	Класичне софтверске архитектуре
25.	Рачунарство у облаку	осми	И	C#, Java, Python
26.	Развој дигиталних игара	осми	И	C#, Java, Python

Студијски програм ИТ пружа дипломираним студентима потребне вештине и знања за успешно обављање ИТ послова у привреди, студенти ће бити спремни за рад на процесу пројектовања, израде и имплементације софтвера. Студенти крећу од почетног нивоа програмирања у првој години студирања; преко објектно оријентисаног програмирања (друга година); програмирања база података (трећа година); у завршној години студирања развијају софтвере за дигиталне игре, тестирају и одржавају софтвере (слика 2.3.5).

Слика 2.3.5: ФТН Чачак, СП Информационе технологије - од Увода у програмирање до Развоја дигиталних игара (Извор: аутор)

2.4. Неуронске мреже

Неуронске мреже су нова и врло перспективна рачунарска технологија која пружа нове приступе проучавању динамичких проблема у различитим областима активности. Неуронске мреже се користе за предвиђање тржишта, оптимизацију робних и новчаних токова, анализу и сумирање испитивања јавног мњења, предвиђање динамике политичких рејтинга, оптимизацију производног процеса, свеобухватну дијагнозу квалитета производа...

Неуронска мрежа се састоји од три главне компоненте: **неурона**, **мрежне топологије** (архитектуре) и **алгоритма учења**. Поред главних компоненти мрежа има и додатне компоненте: величину мреже (број слојева, број неурона у слоју), функционалност неурона (улазни оператор неурона, функција преноса, активациона функција), обучавање/валидност и имплементацију/реализацију.

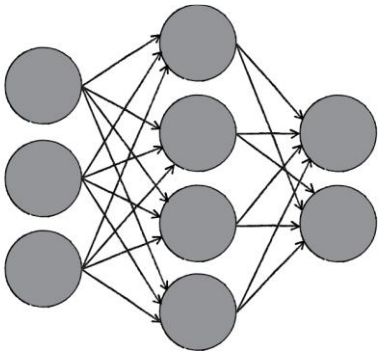
Редослед корака при раду са неуронским мрежама је следећи: дефинисање модела, прикупљање података, узорковање, дизајнирање мреже, одређивање жељеног броја итерација за учење мреже, тренирање мреже, тестирање/валидација, анализа резултата, експлоатација мреже.

2.4.1. Биолошке основе вештачких неуронских мрежа

Вештачка неуронска мрежа је парадигма обраде информација која је инспирисана биолошким нервним системима људског мозга који се састоји од великог броја међусобно повезаних неурона. Оснивачи неуронске мреже и неурокомпјутерских технологија имају за циљ да направе електронске уређаје који су структурно и функционално адекватни људском мозгу. Људски нервни систем изграђен је од око 10^{11} неурона, сваки неурон је повезан са 10^4 других неурона, што укупно чини око 10^{15} синаптичких веза. Јединствена способност сваког неурона је да прима, обрађује и преноси електрохемијске сигнале дуж нервних путева који формирају комуникацијски систем мозга.

Неуронска мрежа има способност да учи из окружења и прилагођена је специфичној апликацији, као што су препознавање узорка или класификација података кроз образовни процес. Неуронска мрежа је сложен адаптивни систем, односно систем може мењати своју унутрашњу структуру на основу информација које пролазе кроз њега [71].

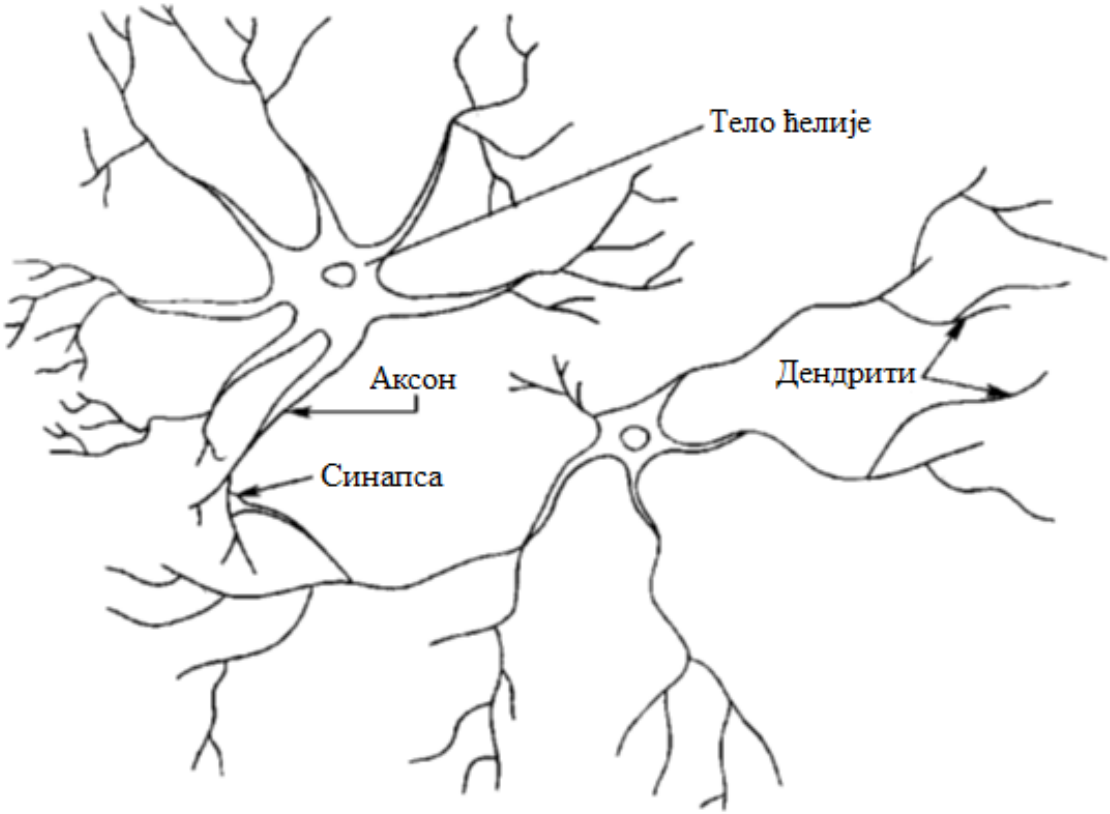
Ово се обично постиже прилагођавањем вредности везе (тзв. *тежине*). На слици 2.4.1 [71] свака стрелица представља везу између два неурона (чвора) и означава пут за проток информација.



Слика 2.4.1: Комуникација између неурона у вештачкој неуронској мрежи (Извор: [71])

Свака веза има тежину, односно број који покреће сигнал између два неурона. Када мрежа генерише „добар“ излаз, тада нема потребе за подешавањем тежине. Међутим, ако мрежа генерише „лош“ излаз, онда се систем прилагођава променом тежине како би побољшао накнадни резултат.

Неурон се састоји од три дела: *ћелијског тела*, *дендрита* и *аксона*. На слици 2.4.2 [72] дат је шематски приказ два типична биолошка неурона.



Слика 2.4.2: Неурони људског мозга (Извор: [72])

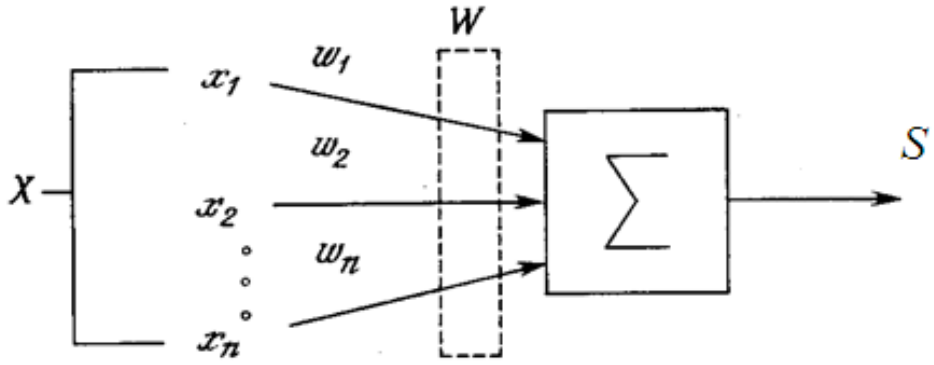
Дендрити се крећу од тела нервне ћелије до других неурона, где примају сигнале (улазне информације) у тачкама спајања које се зову *синапсе*. Улазни сигнали примљени од синапсе доводе се у тело неурона. Овде се сумирају, са неким улазима који настоје побуђивати неуроне, други – да спрече његово побуђивање. Када укупна побуда у телу неурона пређе одређени праг, неурон је побуђен и шаље сигнал дуж *аксона* (*perceptron*) другим неуронима. Сваки неурон (биолошки) може постојати у два стања – побуђено и непобуђено. Већина вештачких неуронских мрежа моделира само ова једноставна својства.

Закључак: неуронске мреже опонашају рад људског мозга при извршавању неког задатка или функције. Сличност између неуронских мрежа и људског мозга огледа се у следећем:

- Неуронска мрежа усваја знање кроз процес обучавања;
- Тежине између неуронских веза (јачина синапси) користе се за меморисање знања.

2.4.2. Вештачки неурон

Вештачки неурон опонаша својства биолошког неурона (слика 2.4.3 [72]). Скуп сигнала, означених x_1, x_2,…, x_n стиже на улаз вештачког неурона, од којих је сваки излаз другог неурона. Сваки унос множи се одговарајућом тежином, сличном синаптичкој јачини, а сви производи се додају заједно, одређујући ниво активације неурона. Ови улазни сигнали, који заједнички означавају вектор X, одговарају сигналима који улазе у синапсе биолошког неурона.



Слика 2.4.3: Вештачки неурон (Извор: [72])

Сваки сигнал се множи са одговарајућом „тежином“ w_1, w_2,…, w_n и доводи у блок за сумирање, означен са Σ. Свака тежина одговара „снази“ једне биолошке синаптичке везе (скуп тежина означен је вектором W) Блок сабирања који одговара телу биолошког елемента додаје пондерисане улазе, стварајући излаз, који се означава као S.

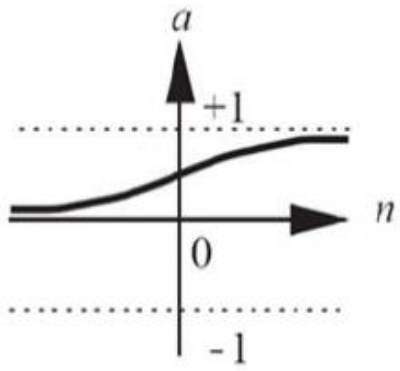
У векторској нотацији то се може написати на следећи начин:

$$S = XW$$

Примљени сигнал S трансформише се активационом функцијом (*Activation function*) неурона, која формира излазни сигнал Y (*output*). Скуп функционалних могућности неуронске мреже, као и методе њеног тренирања, зависе од врсте функције активације.

Математички неурон, попут његовог биолошког прототипа, постоји у два стања. Ако пондерисани збир улазних сигнала S не достигне одређену граничну вредност, тада математички неурон није побуђен и његов излазни сигнал је нула. Ако су улазни сигнали довољно интензивни и њихова сума достиже праг осетљивости, тада неурон прелази у побуђено стање и на његовом се излазу формира сигнал Y = 1.

Да би се тачније моделирао нелинеарни трансфер карактеристичан за биолошки неурон, често се као активациона функција (активациона функција одређује однос између улаза и излаза чвора и мреже) неурона користи *Log-Sigmoid* (сигмоидална логистичка) преносна функција приказана на слици 2.4.4 [72]. Ова функција свој унос (који може имати вредности између плус и минус бесконачно) претвара у износ са вредношћу у интервалу од 0 до 1.



Слика 2.4.4: Log-Sigmoid (сигмоидална логистичка) преносна функција (Извор: [72])

Сигмоидална функција математички је изражена

$$F(x) = \frac{1}{1 + e^{-x}} \tag{1}$$

Увођењем такве континуиране функције нелинеарне активације, спектар проблема које треба решити значајно се проширио.

2.4.3. Основне архитектуре неуронских мрежа

Неуронска мрежа се може сматрати мрежом „неурона“ који су организовани у слојеве. *Предиктори* (или улази) формирају доњи слој (**улазни слој**), а *прогнозе* (или резултати) горњи слој (**излазни слој**). Неуронске мреже могу да имају и средњи слој (**скривени слој**) који садржи „скривене неуроне“. Мреже могу да имају више од једног скривеног слоја, мада је за већину мрежа довољан један скривен слој.

Неуронска мрежа је потпуно повезана – сваки чвор у одређеном слоју повезан је са чвором у следећем слоју, иако не мора бити повезан са чвором у истом слоју. Везе између чворова имају тежине које се на почетку иницијализују (насумичним) вредностима између 0 и 1.

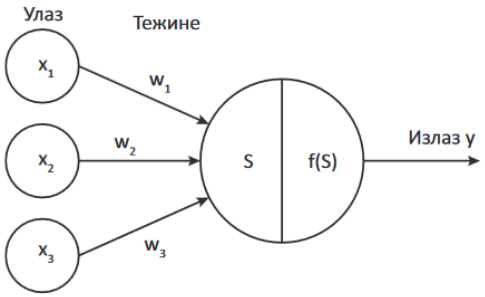
2.4.3.1. Једнослојни перцептрон (SLP – Single Layer Perceptron)

Једнослојни перцептрон је рачунарски прототип неурона и то је најједноставнији модел неуронске мреже, мрежа састављена од једног неурона. Перцептрон има један или више улаза и само један излаз.

Као поједностављени облик неуронске мреже, конкретно једнослојна неуронска мрежа, перцептрони играју важну улогу у бинарној класификацији. То значи да се перцептрон

користи за разврставање података у два дела, дакле бинарно. Због тога се понекад рецептрони називају и линеарним бинарним класификаторима.

Ова мреже не садржи скривене слојеве, и еквивалентне су линеарним регресијама. На слици 2.4.5 [73] приказана је верзија једнослојног перцептрона са три улаза. Излази се добијају линеарном комбинацијом улаза. Тежине се бирају у оквиру неуронске мреже користећи „алгоритам учења“ који минимизира „функцију трошкова“.

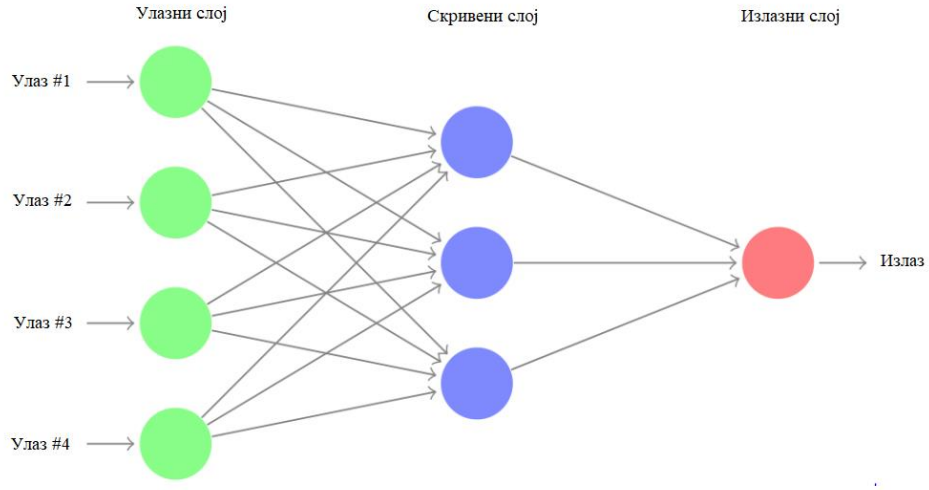


Слика 2.4.5: Илустрација модела перцептрона (Извор: [73])

2.4.3.2. Вишеслојни перцептрон (MLP – Multi Layer Perceptron) за прослеђивање унапред (Feedforward network)

Једном када се дода средњи слој са скривеним неуронима, неуронска мрежа постаје нелинеарна. Корисник сам дефинише број скривених слојева и број чворова у сваком скривеном слоју. Што је већи број чворова у скривеном слоју повећава се моћ неуронских мрежа, али превише скривених слојева доводи до превеликог подударања и меморисања скупа података за тренинг. Повећањем броја чворова у скривеном слоју повећава се тачност тренирања података.

На слици 2.4.6 [74] дат је приказ неуронске мреже са три слоја (улазни, скривени и излазни).



Слика 2.4.6: Неуронска мрежа са четири улаза и једним скривеним слојем са три скривена неурона (multilayer feedforward network) (Извор: [74])

Претходна мрежа (слика 2.4.6) је позната као **вишеслојна мрежа за прослеђивање унапред** (*multilayer feedforward network)*, где сваки слој чворова прима улазе из претходних слојева. Излази чворова у једном слоју су улази на следећи слој, у мрежи нема повратних веза или петљи. Садржи улазни слој, излазни слој и скривени слој.

Уопштено, може бити више скривених слојева. Сваки чвор у слоју је неурон, који се може сматрати основном процесном јединицом неуронске мреже.

Излаз неуронске мреже се израчунава у процесу пропагације (*forward propagation*) који се одвија у три корака [74]:

1. Копирање улаза мреже у активацију улазних јединица;
2. Израчунавање активација у скривеним слојевима;
3. Израчунавање активације излазног слоја и копирање у излаз мреже.

Ова архитектура мреже је најчешће коришћена. За дате улазе са жељеним излазом подешавају се тежине и мрежа производи жељени излаз. Сагласно својој активацијској функцији и тежинској суми свих својих улаза неурон рачуна свој излаз.

2.4.4. Карактеристике неуронских мрежа

Вештачке неуронске мреже се користе за решавање комплексних проблема и могу се реализовати хардверски и софтверски. Неуронске мреже које су хардверски реализоване су аналогне и код ових мрежа проблеми се решавају у паралелном процесирању. Реализација проблема помоћу рачунара представља софтверску реализацију (неурони и везе између неурона су виртуелни). Софтверска имплементација зависи од карактеристика рачунара (брзина процесора, количине радне меморије) и често се дешава да је време за обраду веома дуго, што је лоше решење. Зато се софтверска реализација користи у фази тестирања након чега се прелази на хардверску реализацију [75].

Најважније **карактеристике неуронских мрежа** су [76]:

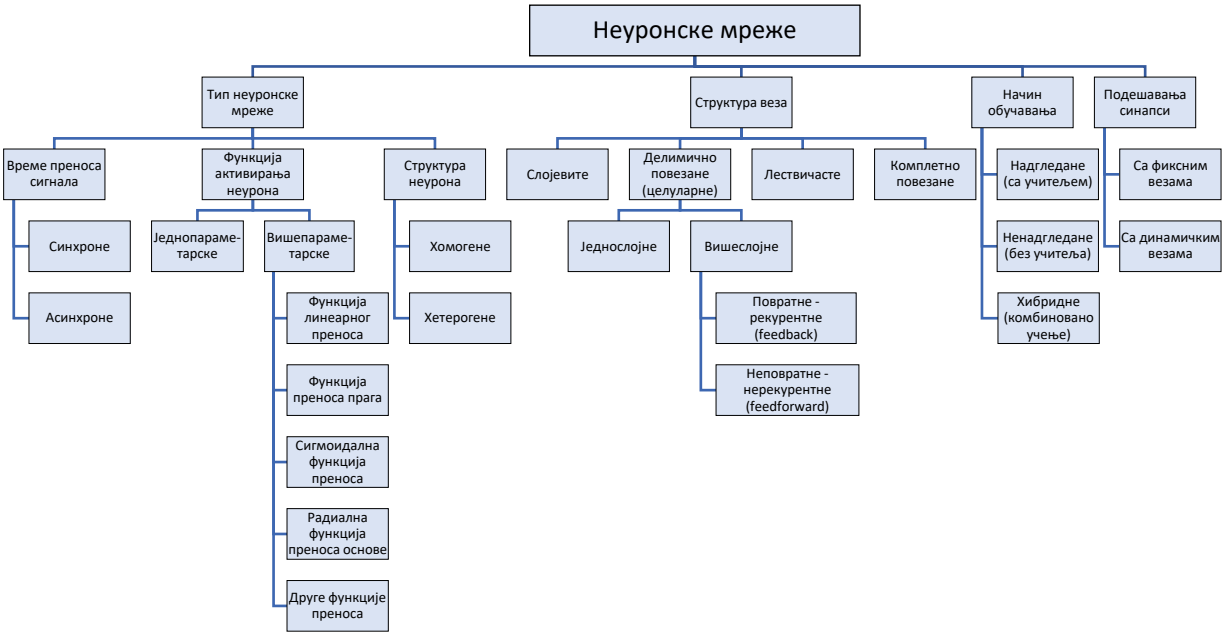
1. *Универзални регресијски системи* – могућност моделирања система са непознатим односом улаза и излаза;
2. *Способност обучавања на основу тренинга* – мрежа „без знања“ може да се обучи са скупом упарених улазно-излазних података како би се добили жељени излази за познате улазе. Кроз процес обучавања (софтверска реализација) на систематичан начин се мењају синаптичке тежине у циљу достизања жељених перформанси мреже;
3. *Генерализација* – својство добре класификације за непознате улазе (излази су задовољавајући и за улазе који нису били присутни у току обучавања);
4. *Адаптивност (прилагодљивост)* – способност мењања јачине синаптичких веза (прилагођавање реакција на промене у окружењу);
5. *Нелинеарност* – суочавање са нелинеарним подацима и окружењем;
6. *Масивна паралелна обрада* – многи неурони се активирају истовремено током обраде података:
7. *Толеранција грешке* – добар одзив чак и ако су улазни подаци мало нетачни;
8. *Робусност* – иако неки неурони „пођу по злу“, читав систем и даље може да функционише добро.

Због свих ових карактеристика вештачке неуронске мреже можемо срести и под следећим називима: паралелно-дистрибуциони процесни модели (*parallel distributed processing models*), самоорганизујући системи (*self-organizing systems*), неурорачунарски системи (*neurocomputing systems*), неуроморфни системи (*neuromorphic systems*) [77].

2.4.5. Класификација неуронских мрежа

Вештачке неуронске мреже, сходно параметру по коме се класификација врши, могу се класификовати у много категорија. Могуће је окарактерисати неуронске мреже према типовима неурона који се користе у мрежи, структури везе, методама вежбања мреже, задацима које мрежа решава.

На слици 2.4.7 [78] дата је класификација неуронских мрежа према одређеним параметрима.



Слика 2.4.7: Класификација неуронских мрежа (Извор: [78])

По структури веза између неурона, неуронске мреже се могу поделити на [75]:

- *Слојевите мреже* – мреже које су организоване у слојеве, излаз из једног слоја улази у следећи слој. Овај тип мрежа је примењен код *backpropagation* алгоритма.
- *Комплетно повезане* неуронске *мреже* – мреже у којима сваки неурон шаље свој излазни сигнал другим неуронима (сви улазни сигнали иду на све неуроне).
- *Лествичасте мреже* – садрже групе поља неурона код којих је сваки од улаза спојен на све улазе неурона у том пољу. Излаз из поља може бити излазни слој, или представљати улаз новом пољу неурона. Ова мрежа је без повратних спрега.
- *Целуларне* (делимично повезане) *мреже* – мреже код којих су повезани само суседни чворови. Целуларне мреже се, према броју слојева деле на: *једнослојне* и *вишеслојне*.

Према смеру простирања информације у мрежи, вишеслојне мреже, могу бити [75]:

- *Мреже са простирањем унапред – нерекурентне* (*feedforward networks*) – омогућавају пренос сигнала само у једном правцу: од улаза до излаза, тј. нема повратних информација (петље), односно излаз било којег слоја не утиче на исти ниво.
- *Мреже са простирањем унапред – рекурентне (feedback networks)* – могу имати сигнале који се крећу у оба смера (виши слојеви враћају информације назад у ниже слојеве) увођењем петље у мрежу.

Према типовима неуронских структура неуронске мреже се могу поделити на *хомогене* и *хетерогене*. Хомогене мреже чине неурони исте врсте са једном функцијом активирања, док хетерогена мрежа укључује неуроне са различитим активацијским функцијама [78].

На основу времена преноса података, неуронске мреже се деле на *синхроне* и *асинхроне*. Код синхроних мрежа само један неурон мења своје стање, док код асинхроних мрежа стање се одједном мења за читаву групу неурона, по правилу за цео слој.

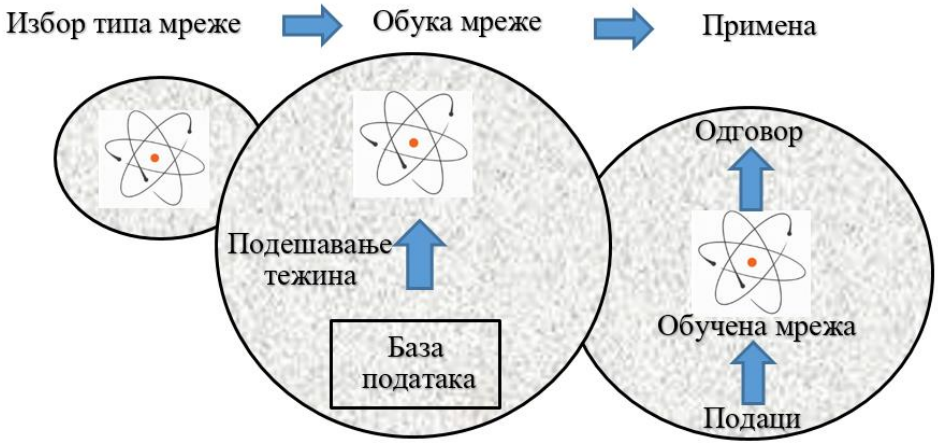
Према начину обучавања вештачке неуронске мреже се могу поделити на мреже за:

- *надгледано* обучавање *(Supervised training);*
- *ненадгледано* обучавање *(Unsupervised training);*
- *делимично надгледано* (комбиновано) обучавање.

Неуронске мреже се **у зависности од начина подешавања синапси** деле на мреже са *фиксним везама* (тежински коефицијенти неуронске мреже бирају се одмах на основу услова проблема) и мреже са *динамичким везама* (синаптичке везе се успостављају у току процеса учења).

2.4.6. Учење неуронске мреже

Након избора врсте неуронске мреже, следи фаза њене обуке. То је кључна карактеристика неуронских мрежа и *обука се односи на учење односа између улазних и излазних параметара кроз поступак тренирања мреже*. Сврха правила учења је обучавање мреже да изведе одређени задатак (слика 2.4.8 [72]).



Слика 2.4.8: Главне фазе пројекта неуронске мреже (Извор: [72])

Тренирање мреже користи се за учење образаца понашања а основни циљ тренирања је проналажење скупа тежина између неурона који одређују глобални минимум „функције грешке“. Овај поступак укључује одлуке у смислу броја итерација при тренирању, односно тачку у којој тренирање престаје.

Мрежа је оспособљена за давање жељеног скупа излаза за неки скуп улаза. Сваки такав улаз (или излаз) скупа се сматра вектором. *Обука* се врши секвенцијалним представљањем улазних вектора уз истовремено подешавање тегова у складу са одређеном процедуром. У процесу обуке, мрежне тежине постепено постају такве да сваки улазни вектор производи излазни вектор.

Постоји много правила учења која се сврставају у три основне категорије: учење под надзором (учење с учитељем), учење без надзора (учење без учитеља) и комбиновано учење (делимично надгледање).

Надзирано учење претпоставља да за сваки улазни вектор постоји циљни вектор који представља жељени излаз. Улазни и излазни вектори се називају паром за учење. Мрежа се обично обучава на већини таквих парова за тренинг. Дат је улазни вектор, излаз мреже се израчунава и упоређује са одговарајућим циљним вектором, разлика (грешка) се враћа у мрежу уз помоћ повратних информација, а тежине се мењају у складу са алгоритмом који настоји да минимизира грешку.

Значи, алгоритми учења под надзором могу се применити само ако је унапред познато жељено владање неуронске мреже, односно подаци на основу којих се мрежа учи морају да садрже парове вредности улазно-излазних сигнала. Ово учење се може изводити у оба начина рада, *off-line* или *on-line* режиму.

Код *учење без надзора* излази, односно резултати нису унапред познати. Овај тип учења се користи за препознавање узорака и кластерисање [79].

2.4.6.1. Backpropagation алгоритам за обучавање

Најкоришћенији **алгоритам за обучавање *MLP* мрежа** је алгоритам са пропагацијом уназад (*backpropagation*) [80-83].

Циљ *backpropagation* алгоритма је како смањити вредности грешке у насумично додељеним тежинама како би се добио тачан излаз. Систем се обучава методом учења под надзором, где је грешка између излаза система и познатог очекиваног излаза представљена систему и коришћена за модификовање његовог унутрашњег стања.

Током фазе тренирања подаци за тренирање налазе се у улазном слоју. У овом алгоритму сваки чвор у скривеном слоју добија улазне податке из сваког чвора у улазном слоју који се множе са одговарајућим тежинама и затим сабирају. Излаз из скривеног чвора је нелинеарна трансформација резултујуће суме. Слично, сваки чвор у излазном слоју добија улаз из свих чворова из скривеног слоја који се множе са одговарајућим тежинама и затим сабирају.

Потом се добијене вредности на излазу упоређују са циљаним излазним вредностима. Циљане излазне вредности су оне вредности помоћу којих се покушава да се неуронска мрежа подучи.

Након тога се рачуна грешка између добијених излазних вредности и циљаних вредности и враћа уназад преко скривеног слоја. Овај поступак се зове повратни поступак кроз скривен слој. Грешка се користи да би се кориговала снага конекције између чворова, односно на основу ње се ажурирају тежине између улазног и скривеног слоја, и скривеног и излазног слоја.

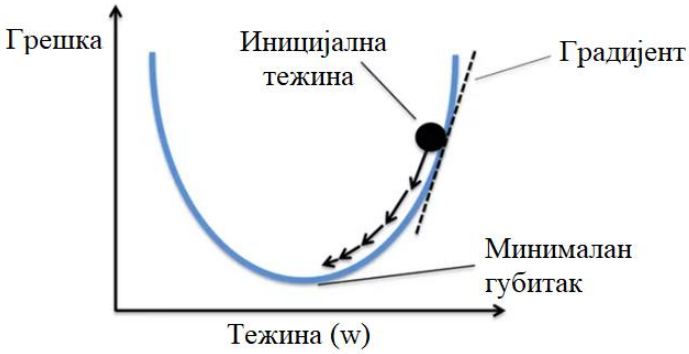
Принцип рада *Backpropagation* алгоритма за учење дат је на слици 2.4.9 [82]:

- *Израчунавање грешке* – колико је удаљен модел од стварног излаза.
- *Минимална грешка* – провера да ли је грешка сведена на минимум или не.
- *Ажурирање параметара* – ако је грешка велика, треба ажурирати параметре (тежине и нагиб). Након тога се поново проверава грешка. Поступак се понавља све док грешка не постане минимална.
- *Модел је спреман да предвиди* – када грешка постане минимална, тада могу у модел да се унесу улазне вредности и модел ће произвести излаз.

Слика 2.4.9: Принцип рада Backpropagation алгоритма за обучавање (Извор: [82])

Тежине се морају ажурирати (оптимизовати) помоћу спуштања градијента како би се добио минимум губитка (минимизирати разлику између очекиваног и добијеног).

Функција губитка своди сву сложеност неуронске мреже на један број који показује колико је далеко од неуронске мреже. Функција губитка има своју криву и градијенте који се могу користити као водич за подешавање тежина, нагиб кривуље функције губитка указује на минималну вредност. Циљ је лоцирати минимум на целој кривуљи, а то представља улазе тамо где је неуронска мрежа најтачнија. Овај поступак се назива *градиент спуштања (Gradient Descent – GD)* (слика 2.4.10 [82]).



Слика 2.4.10: Backpropagation – Графикон функције (Извор: [82])

Тежине се мењају у супротном смеру од израчунатог градијента. Циклус се понавља све док се не постигне минимум функције губитка.

2.4.7. Процес развоја неуронских мрежа

Модели неуронских мрежа се могу користити у сврхе:

- *Класификације* – уколико објекат треба да буде придружен некој од постојећих, предефинисаних група или класа, што је случај у овом раду;
- *Предвиђања* – уколико се предвиђа објекат који није у скупу постојећих класа.

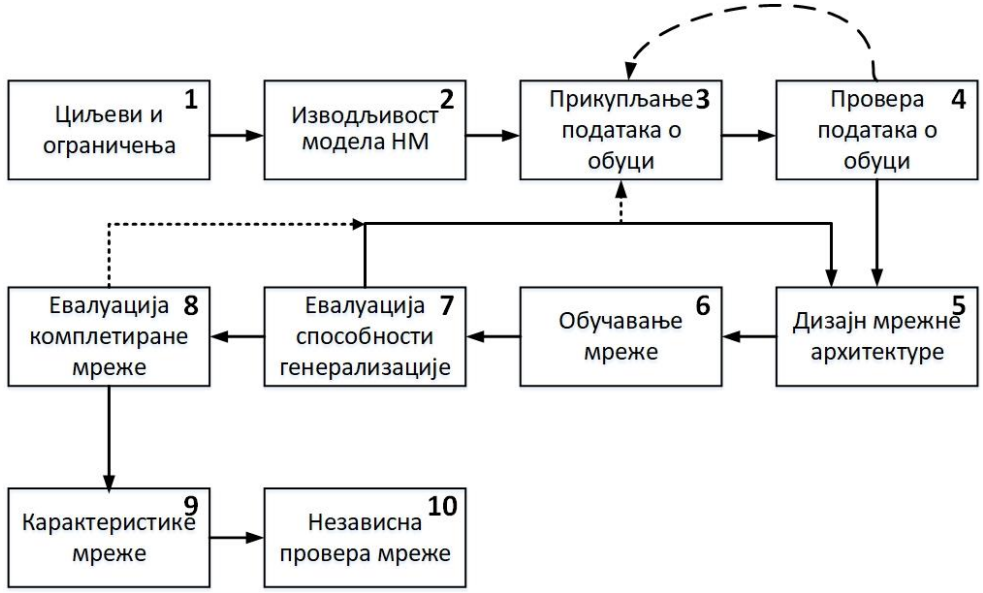
У процесу развоја неуронских мрежа потребно је извршити различите активности евалуација. *Евалуација* представља процес [84]: 1) развијања испитивања које може бити основа за исправне процене; 2) спровођење испитивања; 3) употреба резултата за доношење одлуке о томе шта даље.

Како би се добили објективни резултати евалуације, потребно је најпре извршити евалуцију креираног модела, а који се односи на неуронске мреже. У процесу евалуације се испитује да ли модел задовољава постављене циљеве; уколико то није случај, модел се креира поново, уз промену параметара, све док се не постигне жељени резултат; након завршеног процеса евалуације корисници доносе одлуку о начину коришћења добијених резултата.

Основне смернице за развој модела, које имплицирају да ће евалуација бити континуирана активност током развоја, су:

- грешке морају да се препознају, дијагностикују и морају да се отклоне што је раније у процесу развоја;
- треба предвидети врсте грешака које ће се вероватно појавити и уградити практичне контроле како би се грешке смањиле.

На слици 2.4.11 [84] дат је приказ основних фаза развоја неуронских мрежа са нагласком на евалуацијским активностима.



Слика 2.4.11: Фазе у развоју неуронске мреже (Извор: [84])

Кораци евалуације приказани су у фазама 2, 4, 7, 8 и 10 [84].

- *Развој неуронске мреже* почиње жељом да се постигне неки циљ. Овај циљ треба изричито навести како би се осигурала почетна основа за евалуациону активност. Поред тога, требало би навести и сва позната ограничења, попут формата улазних података или програмског окружења. Ове активности су приказане у **кораку 1**.
- Следећа активност, приказана у **кораку 2**, јесте *испитивање циљева и ограничења* и доношење одлуке о томе да ли је неуронска мрежа права врста технологије за дати задатак. Зато је потребно разумети карактеристике домена у коме је неуронска мрежа одржива.
- Ако се процени да је проблем могуће постићи методама неуронске мреже, тада се *сакупљају и верификују подаци о тренинзима* (**кораци 3 и 4**). *Верификација* се састоји од процене свеобухватности и тачности података и процене њихове примерености за употребу у обуци мреже. У овој фази је потребно да се одлучи о стратегији претходне обраде података о обуци. Подаци о неуронској мрежи се обично унапред обрађују како би жељене функције постале „видљивије“.

- Задатак *дизајнирања мрежне архитектуре* (**корак 5**) је одлука о врсти мреже која ће се користити, броју скривених чворова, броју слојева и како распоредити тежине. Након тога, мрежа се *тренира* (**корак 6**) аутоматским подешавањем тежине док стварни излаз за сваки од случајева тренинга не буде што је могуће ближи тачном одговору.
- У следећој фази *испитује се способност генерализације мреже*, односно њена способност да се приближи тачном одговору на претходно невидљиве податке (**корак 7**) и затим се доноси одлука о томе да ли мрежа треба да има измењену архитектуру или да ли треба реализовати другачији алгоритам. Ако је тако, кораци дизајна и обуке се понављају са побољшаном архитектуром или алгоритмом тренинга. Може да се утврдити да лоша генерализација произилази из неадекватности у подацима о обуци. У том случају прикупљање и верификација узорка тренинга се понавља.
- Када је оптимизирана способност мреже за оптимизацију, комплетирана *мрежа се евалуира* за прихватљивост корисника (**корак 8**), а тестови се изводе у сврху разграничења границе прихватљивог понашања и квантификације осталих *карактеристика мреже* (**корак 9**). Ове карактеристике су одређене и постају основа за коначну валидацију.
- У последњој фази *испитује се* довршена (комлетирана) *мрежа* (**корак 10**) и доноси се одлука о њеној подобности за употребу на терену. Објективност ће бити побољшана ако коначну валидацију изврши тим независан од развојног тима.

Технике евалуације података

За евалуација података користиће се иницијално тестирање питањима и испитивањем. Помоћу ових техника могуће је идентификовати: тачке података које су далеко од осталих података; тачке података или подскупови који би могли да дају више од наследника утицаја на исход процеса обуке; ситуације у којима су вредности једне карактеристике линеарне комбинације вредности других карактеристика. Ако се неки од ових потенцијалних проблема нађе, треба га доставити програмерима неуронске мреже.

Поузданост резултата

Пре употребе софтвера који укључује технике неуронских мрежа неопходно је утврдити поузданост резултата који се добијају. Поузданост се утврђује кроз поступке:

- **Валидације** (Овај поступак даје одговор на питање: „*Да ли је направљен одговарајући производ?*“)

  Процесом валидације проучава се систем са различитих аспеката. Почев од полазних потреба које систем треба да обезбеди и промена које су настале током процеса развоја система испитује се да ли је крајњи резултат у складу са полазним планом. Валидација захтева испуњење свих захтева кроз процес развоја система. Валидација представља процес у коме се одређује степен тачности репрезентације модела у односу на сврху коришћења модела [85].

- **Верификације** (Овај поступак даје одговор на питање: „*Да ли је производ направљен на прави начин?*“)

  Процес верификације. У изградњи неуронске мреже, подаци о обуци се користе за одређивање мреже. Веома је важно да ти подаци буду репрезентативни за стварне податке који ће се користити на терену. Да би модел био користан мора бити свеобухватан у смислу да постоје случајеви обуке за сваку карактеристику која треба да се препозна. Узорковање треба чешће обављати тамо где се улазни подаци брже мењају [84].

Верификација обезбеђује комплетну спецификацију и осигурава непостојање грешака у имплементацији модела. Међутим, верификација не обезбеђује да модел решава проблем, као и да тачно репрезентује реалне процесе који се догађају [86]. У овом раду **верификација модела је извршена путем анализе циљева**. То је урађено тестирањем случаја коришћења и анализе да ли су добијеним резултатима испуњени захтеви, односно циљеви.

Евалуација резултата тестирања

Процеси верификације и валидације укључују евалуацију модела коришћењем реалних података. Како би неуронска мрежа била успешно примењена у реалним ситуацијама неопходно је извршити и евалуацију резултата тестирања.

Верификација модела потврђује да је [86]:

- Дизајн модела коректан;
- Дизајн модела у складу са потребама;
- Модел креиран без грешака;
- Модел имплементирао адекватне улазне и излазне неуроне.

Постоји више приступа за добијање информација о квалитету информација које се добијају преко модела анализе података (валидација модела) [86]:

- Коришћење статистичких показатеља;
- Подела скупа података на два дела: део за тренирање и део за тестирање;
- Консултација са стручњацима и преглед добијених резултата како би се утврдило да ли откривени обрасци понашања имају смисао у конкретном случају коришћења.

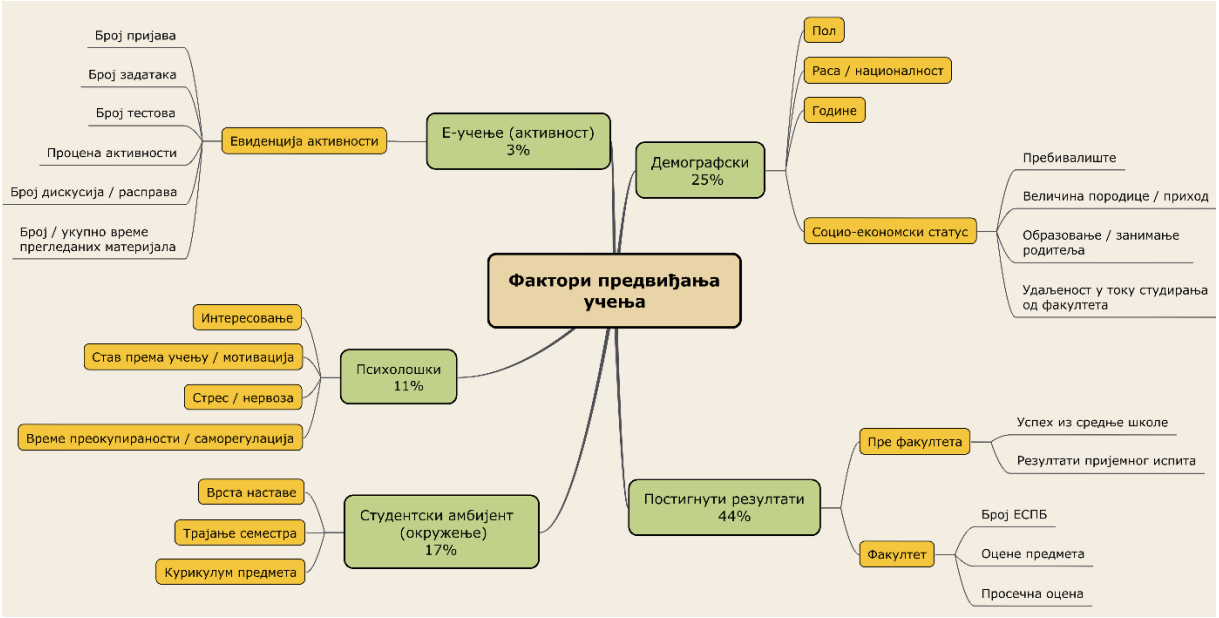
Овај поступак се спроводи како би се одредила испуњеност следећих карактеристика креираног модела: тачност, поузданост, корисност.

2.4.8. Вештачке неуронске мреже у предвиђању успеха у учењу

Многи факултети у Србији и окружењу имају проблем: одређени број кандидата (матураната) који су успешно положили пријемни испит, већ на крају првог семестра сусреће се са тешкоћама полагања испита. Ово се посебно односи на студијске програме које су усмерене на обуку ИТ стручњака. Успех студената игра кључну улогу у образовним високошколским институцијама, и представља кључни показатељ за оцењивање квалитета, односно успешности тих образовних установа.

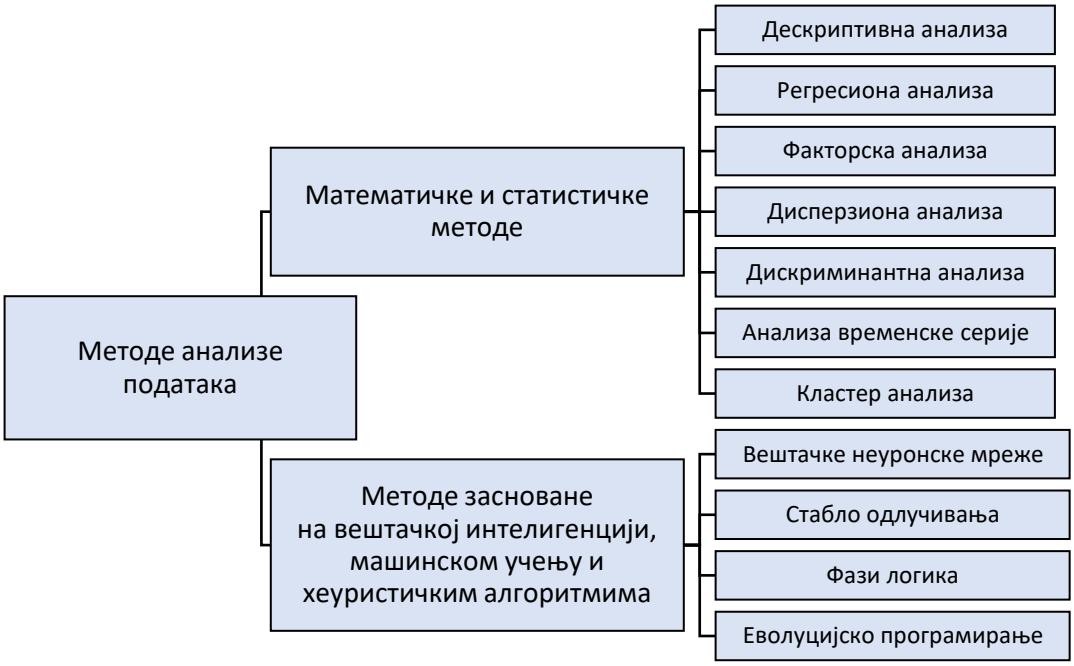
Шта представља успех студента, који све фактори утичу на успех студената на факултетима? На ове теме објављен је велики број радова [87-90]. Успех студената представља: ангажовање у образовним активностима, стицање знања, вештина и компетенција, упорност, постизање циљева и резултата образовања, постизање пост-факултетских резултата (успех у каријери).

У литератури је истражено (идентификовано) мноштво фактора који могу потенцијално да утичу на предвиђање успеха студирања и самим тим и дефинисати податке које треба „рударити“. Пет фактора којима су аутори радова највише посвећивали пажњу су: *постигнути резултати студената* (овај фактор је представљен у 44% истраживачких радова), *демографија студената* (25%), *окружење у којем студенти студирају* (17%), *психолошке особине студената* (11%) и *активност е-учења* (3%). На слици 2.4.12 [87] дат је преглед фактора који могу потенцијално да утичу на предвиђање успеха студената.



Слика 2.4.12: Широк распон фактора који потенцијално утичу на предвиђање успеха студената (Извор: [87])

Рударење података нема сопствене методе анализе података већ користи методологије и технике других сродних области науке. На слици 2.4.13 [91] дат је приказ метода који се користе у анализи података.



Слика 2.4.13: Кључне врсте метода и техника анализе података (Извор: [91])

Предиктивна аналитика широко је поље техника које имају заједнички циљ предвиђања будућег понашања, тј. усмерена је на интерпретацију постојећих података у циљу давања предвиђања о будућим догађајима. Она укључује технике статистике: Рударење података (*Data mining*), Анализу текста (*Text analytics*) и Предиктивно моделирање (*Predictive modeling*). У табели 2.4.1 [92] дат је приказ најпопуларнијих метода/техника за предвиђање.

Табела 2.4.1: Најпопуларније технике за предвиђање (Извор: [92])

Технике	Сврха	Предности/Недостаци	На која питања треба одговорити?
Стабло одлучивања	Предвиђање будуће класе података	*Предност*: Једноставан за имплементацију и разумевање. *Недостатак*: Може бити превише поједностављен за многе проблеме	„Који“ или одговор на Да/Не питања.
Неуронске мреже	Кластерисање и класификација препознавањем по обрасцима у подацима	*Предност*: Надмашује већину алгоритама за машинско учење. *Недостатак*: Тешко за имплементацију; захтева паралелни процесор.	Скоро свако питање (све док има довољно података и некакве корелације/узрочно-последичне везе).
Регресија (линеарна и логичка)	Процењивање односа између варијабли које користи за предвиђање будућих исхода	*Предност*: Резултате је лако разумети. *Недостатак*: Ограничено на линеарне/логистичке податке.	Колико су одређене циљне варијабле?
Временска серија	Прогнозирање континуиране варијабле током времена	*Предност*: Резултате је лако разумети. *Недостатак*: Ограничено на податке који зависе од времена.	Какав ће бити резултат код будућих података?
Кластеровање (К-вредности)	Организовање података у сличне групе	*Предност*: Једноставно за имплементацију. *Недостатак*: Тешко је предвидети „К“ број кластера.	Који су обрасци у подацима?
Факторска анализа	Проналажење објашњења за резултате / корелације	*Предност*: Смањивање података. *Недостатак*: Факторе је тешко протумачити; информације се могу изгубити.	Која су објашњења за теме у подацима?

2.5. Стилови учења

Људски мозак је веома сложен и још увек је мистерија. Учење је примарна функција мозга. Године 1969. Дејвид Осубел и Флојд Робинсон [93] предложили су значајну парадигму учења, која је заснована на следећа два полазишта:

- „*Најважнији фактор који утиче на учење је количина, јасноћа и организација садашњег знања ученика. Ово садашње знање, које се састоји од чињеница, концепата, предлога, теорија и сирових перцептивних података које је имао у сваком тренутку на располагању, временом се назива његова когнитивна структура* [93]“;
- Природа материјала који ће се научити.

Приметно је да свака особа преферира различите стилове и технике учења. **Стилови учења** омогућавају груписање уобичајених начина на које људи уче. Сви имају комбинацију стилова учења. Неки ће можда открити да имају доминантан стил учења са много мање употребе других стилова. Други ће можда открити да, у различитим околностима, користе различите стилове. Нема праве комбинације.

Стил учења је добро успостављен и доминантан начин пријема, обраде и коришћења подстицаја/информација у *процесу учења*, који је најпрепознатљивији током организованог учења у наставном процесу [94]. *Настава* је сложен интерактивни процес у којем ученици и наставници остварују различите односе и комуникацију ради постизања циљева и задатака образовања [95].

Према једној од највише цитираних класификација педагошких варијабли релевантних за успешност наставе Осубела и Робинсона [93], издвојене су [94]:

- *Когнитивне (сазнајне) варијабле*: когнитивна структура, развојна готовост, интелектуалне способности и вештине, наставни материјал;
- *Афективне и социјалне варијабле*: мотивациони и вредносни фактори (жеља за знањем, мотив постигнућа); фактори личности; групни и социјални фактори (разредна клима, сарадња и такмичење, културна депривација (ограничење)) и карактеристике наставника.

У образовању се најчешће спомињу следећи **модели стилова учења** [95]:

- *Myers-Briggs*-ов модел;
- Модел *Dunn* и *Dunn*;
- *Felder-Silverman*-ов модел;
- *Колбов модел*.

2.5.1. Колбов модел стилова учења

Колбов модел је, према мишљењу многих аутора, најутицајнији модел стилова учења. Модел стилова учења *Дејва Колба* [96] наглашено интегрише својства личности и процесе важне за школско учење.

Колб је учење дефинисано као „*процес у којем се знање ствара трансформацијом искуства. Знање је резултат комбинације схватања и преображавања искуства*“. Схватање искуства односи се на процес преузимања информација, а трансформација искуства је начин на који појединци тумаче и делују на те информације. Колбов модел учења укључује *четири* узастопне *фазе* [97]:

- *Прва фаза*: стицање конкретног искуства (*concrete experience, CE*);
- *Друга фаза*: рефлексивно посматрање током којег ученик размишља о ономе што је научио (*reflective observation, RO*);
- *Трећа фаза*: апстрактна концептуализација, разумевање нових знања, њихова теоријска генерализација (*abstract conceptualization, AC*);
- *Четврта фаза*: активна експериментална верификација нових знања и самостална примена у пракси (*active experimentation, AE*).

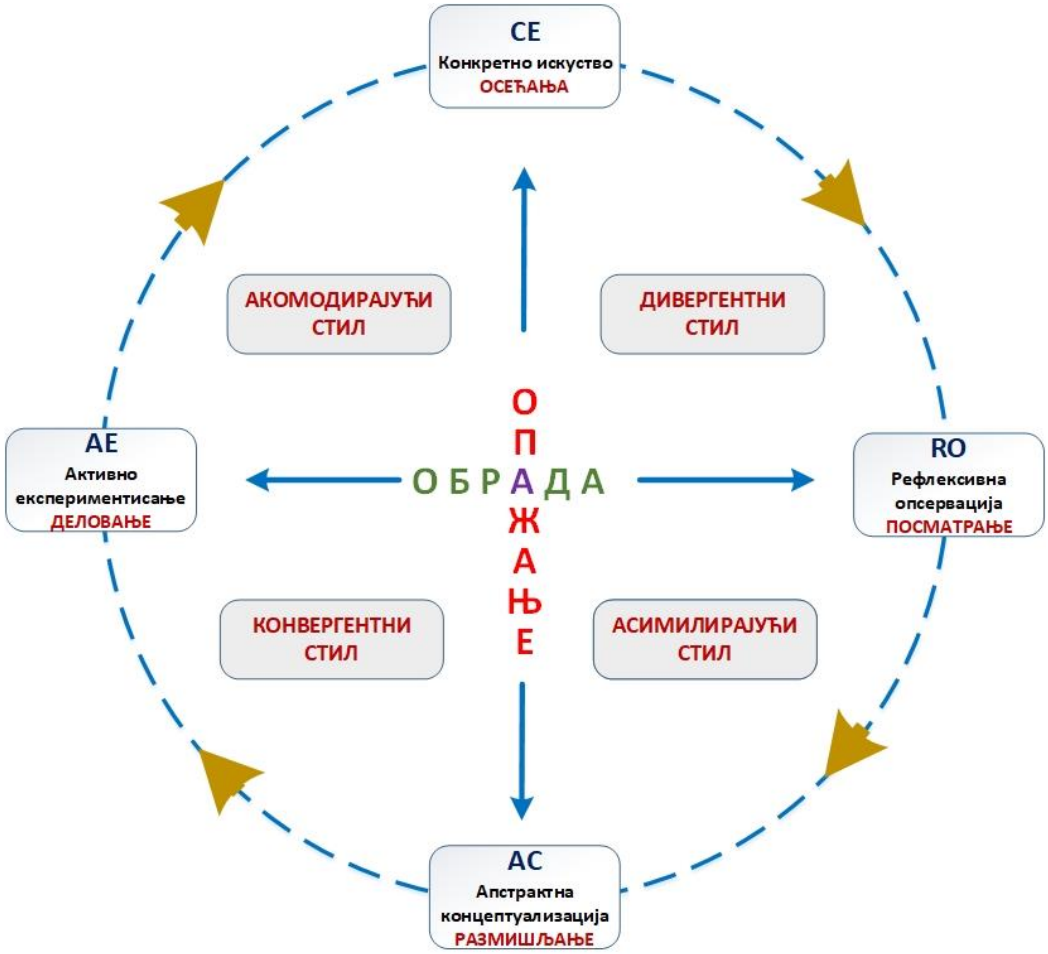
Учење проистиче из решења креативне напетости између ова четири начина учења. Овај процес је представљен као идеализовани циклус учења или спирале где ученик „додирује све 4 фазе“ – *осећање (CE), посматрање (RO), размишљање (AC)* и *деловање (AE)*. Стил учења сваке особе је јединствен и није фиксна психолошка особина, већ је то динамичко стање које произилази из трансакција између особе и околине. Нису све фазе циклуса учења подједнако важне и циклус учења ће зависити од оног шта ученик преферира и од самог садржаја учења [97].

Комбинацијом ова 4 начина учења Колб је засновао свој модел стилова учења на четири елемента распоређена у оквиру две димензије [94]:

- *хоризонтална димензија (AE-RO)* – димензија обраде или процесирања одређује да ли при обради информација доминира рефлексивна опсервација или активно експериментисање;
- *вертикална димензија (AK-CE)* – димензија перцепције (опажања) одређује одакле се примају информације – да ли је извор конкретно искуство или апстрактна концептуализација.

Да би се одредио жељени стил учења, развијен је инструмент (упитник) који мери стилове: Инвентар стила учења III (*LSI3 – The Learning Style Inventory 3*) [94]. *Ова верзија упитника је коришћена у дисертацији за самостално оцењивање студената и тумачење резултата.*

Студије спроведене у последњих 40 година показале су да се сви студенти, зависно о стилу учења, могу поделити у четири главне групе: са *дивергентним*, *акомодирајућим*, *конвергентним* и *асимилирајућим* стиловима учења (слика 2.5.1 [94, 95]).



Слика 2.5.1: Модели и стилови учења по Колбу (Извор: [94, 95])

- **Дивергентни стил учења:** *конкретно искуство (CE) трансформисано кроз рефлективно опажање.* **Рефлексивни мислилац или дивергер** осећа се самоуверено у ситуацијама које захтевају стварање нових идеја и развој алтернативних перспектива; ужива у креативним активностима повезаним са свеобухватним разматрањем проблема; обично користи индукцијске методе и има изузетну ширину интересовања; карактерише га развијена машта, емотивност, жудња за уметношћу и жеља за радом у групама, чији чланови могу имати широку палету мишљења; преферирано питање за ученика оваквог стила учења је „*Шта ће се догодити и зашто*?" [94]. Методе подучавања за дивергентни стил су: визуелна презентација (припрема презентација); стварање нових идеја и развој алтернатива; рад у групи (дискусија, играње пословних улога; активна комуникација) [98].

- **Асимилирајући стил учења:** *апстрактна концептуализација (AC), при чему се искуство рефлектује и повезује са општим и апстрактним идејама, и трансформише се кроз рефлективно опажање (RO)*. **Теоретичар или асимилатор** најприкладнији је за обраду великих количина информација и њихово презентовање на тачан, компактан и логичан начин; склон је добијању информација током интеракције са другим људима, радије ради са апстрактним идејама и концептима; широко користи методе индукције и настоји да пронађе смисао свих доступних информација; ставља логичку беспрекорност теорије изнад њене практичне, или примењене, вредности; преферирано питање је „*Шта знамо о томе?*“ [94]. Пожељне методе учења за асимилаторе су: обрада и анализа информација (аналитички извештај); предавања, читање, рад са апстрактним појмовима (извођење формула, изградња алгоритама); претраживање и обрада информација (интерпретација) [98].
- **Конвергентни стил учења:** *апстрактна концептуализација (AC) трансформисана активним експериментисањем (AE), при чему студент тестира идеје примењујући их на друге области.* **Практичар (прагматичар) или конвергер** вешто користи различите идеје и теорије у пракси; приликом решавања проблема и доношења одлука радије се бави техничким проблемима и формулисаним проблемима, него питањима социјалних и међуљудских односа; зна како да преведе идеје у праксу и како да решава проблеме које разуме; преферирано питање је „*Како то функционише и шта можемо урадити са тим?*" [94]. Методе подучавања које одговарају овом стилу укључују: практично тестирање идеја и теорија (експеримент); решавање теоријских и техничких проблема (задаци, случајеви); израду модела (истраживање, моделирање ситуација) [98].
- **Акомодирајући стил учења:** *конкретно искуство (CE) трансформисано активним експериментисањем (AE).* **Активиста или акомодатор** јасно планира своје активности и воли да експериментише са нечим новим и изазовним; више се ослања на интуицију него на логичку анализу и при решавању проблема прибегава интеракцији са другим људима, а не систематској критици; склон је активностима које захтевају ризик и прилагодљивост; преферирано питање је „*Шта најбоље можемо да учинимо у овој ситуацији?*" [94]. Методе учења за акомодаторе су: практично животно искуство (апропријација, експеримент); интеракција са другим људима (менторство, групни рад); руковођење, лидерство [98].

Тренутно у високошколским установама у Србији, које школују ИТ стручњаке, даје се велики акценат на теоријским методама поучавања које одговарају рефлективном посматрању (*RO*) и апстрактној концептуализацији (*AC*). У исто време, практичне активности и стицање специфичног искуства веома су важни за будуће стручњаке ИТ. Значи, курикулима недостају експерименталне вежбе које су осмишљене да помогну студентима да развију неопходне вештине. Најбољи програми обуке за будуће стручњаке ИТ треба да укључе методе активног учења и експериментисања, као и расправу у учионици и полагање испита.

2.6. Сродна истраживања

Постоји значајан број истраживања која се баве предикцијом успеха студената. Велики број истраживача је користио неуронске мреже за предвиђање резултата студената. Проведене су значајне студије засноване на неуронским мрежама на подацима из школа, факултета и курсева образовања на даљину са циљем предвиђања постигнућа студената, напредовања студената и њиховим потенцијалима (одређивања успешности).

Сродна истраживања [99-126] дата су по годинама истраживања.

Аутори неколико америчких универзитета [99] упоредили су неуронске мреже и статистичке моделе за груписање студената у две различите групе, користећи тест. Показано је да неуронска мрежа може да надмаши класичну дискриминантну анализу у исправном предвиђању препоручених резултата и да обучена неуронска мрежа може да буде ефикасна алтернатива писменом тесту. Аутори рада [100] применили су неуронске мреже да би предвидели број грешака које ће студент да направи при решавању наредног задатка, користећи четири улаза: време (у секундама) које је студенту потребно за генерисање решења, ниво помоћи који студент тражи, ниво сложености проблема и тренутни ниво студента. Мрежа прво предвиђа број грешака које ће студент да направи, а онда одлучује о прикладном задатку за студента. Постигнута тачност предвиђања је велика, али развијена мрежа не може да се користи онлајн, јер мрежа захтева вредности које нису лако доступне. Аутори рада [101] су на основу података корисника са Moodle система (фреквенција и врсте приступа разним ресурсима), користећи RBF неуронску мрежу, показали да је могуће предвидети студенте који ће имати проблема да успешно савладају предмет. Добили су тачност предвиђања око 80%. Неуронске мреже, за процену будућих резултата студената на основу породичних и социјалних карактеристика ученика, карактеристика наставног предмета и наставника, као и амбијента у учионици, коришћене су у раду [102]. Аутори су у раду [103] приказали предвиђање академског успеха студената које је пресудно за образовне институције које планирају стратешке програме за побољшање или одржавање успеха. У овој студији демографски профил студената и просечна оцена за први семестар студија користе се као предикторска променљива за академски учинак студената. Имајући у виду слабији квалитет дипломаца, нигеријски истраживачи [104] користе вештачке неуронске мреже и следеће улазе како би идентификовали степен успешности студената: резултати из предмета из претходног школовања, резултати на матурском испиту, године, порекло родитеља, врсте и локација похађане средње школе и пол. Модел мреже је заснован на топологији вишеслојног перцептрона за предвиђање могућих перформанси кандидата. Тачност предвиђања која су добили аутори рада била је око 74%. Да би предвидели стопу дипломирања студената, аутори рада [105] развили су модел трослојне *backpropagation* перцептронске неуронске мреже. У истраживању је укупно учествовало 1407 испитаника, 1100 за тренинг, а преосталих 307 за тестирање. Просечна стопа предвидљивости за обуку и тест су биле 72% и 68%, респективно. За предвиђане коначне оцене студената на курсевима за е-учење, аутори [106] су користили три feed-forward неуронске мреже. Улаз у сваку мрежу састојао се од резултата тестова вишеструког избора довршених током курса, док је мрежни циљни излаз био завршни тест вишеструког избора завршен на крају курса. Купер [107] је представио модел који треба, на крају прве године студирања, да предвиди (идентификује) студенте који ће највероватније успешно да савладају други семестар, као и студенте који су „изложени ризику“ да њихов резултат у другој години не буде успешан. У раду [108] представљене су методе засноване на неуронским мрежама и кластерима података и извршена је класификација (груписање) студената (на успешне и на мање успешне студенте), ради

даљих препорука и саветовања. Халачев [109] је предложио модел неуронске мреже која се користи за предвиђање показатеља исхода е-учења са малим узорцима података, заснованог на избалансираној листи резултата (*Balanced ScoreCard*). Аутор је добио грешку прогнозе од 3-4% која је прихватљива са практичне тачке гледишта. Аутори рада [110], користећи фази логику и предиктивне факторе као што су: резултати из средњих школа, начин уписа, писменост родитеља и други, предвидели су статус ризика новопримљених студената рачунарских наука Универзитета Поханг у Малезији. Истраживачи су моделирали трансформисане улазне варијабле користећи приступ фази логике. Рад [111] се бави употребом техника дата мајнинга како би се путем предикције решио проблем одустајања од студирања. Предвиђање резултата студената важан је проблем за менаџмент универзитета који желе да избегну феномен раног напуштања. Неуронском мрежом се предвиђају резултати студената измерени просеком оцена у првој години студија. Коришћен је вишеслојни перцептрон са једним улазним слојем, два скривена слоја и једним излазним слојем, мрежа је обучена користећи верзију *backpropagation* алгоритма. Улазни подаци укључивали су профил студената у време уписа на универзитет, информације о годинама студената, просек успеха при завршетку средње школе, јаз између матуре и уписа у високо образовање. У раду [112] се посебна пажња посвећује избору параметара за предикцију и индивидуалном приступу сваком студенту па се у том циљу користе упитници отвореног типа где студенти након сваке лекције уписују своје утиске а који се користе као један од параметара за предикцију успешности. Лесински и остали [113] су користили вештачку неуронску мрежу за класификовање статуса дипломирања студената на основу изабраних академских, демографских и других показатеља. Девет улазних променљивих састоје се од категоријских и нумеричких елемената података који укључују: ранг средње школе, квалитет средње школе, стандардизоване оцене на тесту, оцена ваннаставне активности, образовни статус родитеља и време од завршетка средње школе. Ови улази и модел вишеслојне неуронске мреже користе се за класификацију студената на оне који су дипломирали, оне који нису дипломирали и оне који су касно дипломирали. Рад [114] се фокусира на предвиђање перформанси програмирања студената прве године основних студија на предмету Рачунарске апликације помоћу предиктивног модела дата мајнинга користећи алгоритме засноване на класификацији. Прикупљени подаци садрже демографске податке студента, оцену из уводног програмирања на факултету и оцену из уводног програмирања на тесту који садржи 60 питања. Јанг и Ли [115] формулишу модел студената са факторима који су повезани са успехом и факторима који нису повезани са успехом студената. Помоћу *backpropagation* неуронске мреже, засноване на класификацији, предложени су алати за процену учинка студената и показатељи напретка студената. Ова неуронска мрежа може да процени успешност студената према претходном знању студената, као и успешност других студената који имају сличне карактеристике. Аутори [116] анализирају алгоритме предвиђања који се користе у истраживању података и нуде увид у бољи алгоритам предвиђања који се користи за предвиђање успеха студената. На основу успеха студената на крају другог семестра користећи модел вишеслојне перцепције неуронске мреже, Ахман и Шахзади [117] су предвидели студенте који су ризични, односно који нису ризични за наставак студирања (успешност модела преко 95%). Рад [118] представља модел предвиђања који се заснива на вештачкој неуронској мрежи применом селекције карактеристика (*Feature Selection*). У истраживању је учествовао 161 студент са универзитета рачунарских и информатичких наука у Басри у Ираку. Аутори радова [119, 120] су за своја истраживања развили алгоритме засноване на конволуцијским неуронским мрежама и аутоматској детекцији лица за предвиђање (идентификацију) студената који су потенцијално могли да запну на својим студијама. У раду [121] Лау и остали користе вештачку неуронску

мрежу за процену и предвиђање успеха студената користећи социјално-економске податке и резултате пријемних испита. Неуронска мрежа је моделирана са 11 улазних варијабли, два слоја скривених неурона и једним излазним слојем, модел је постигао тачност од 85%. У раду [122] коришћено је више алгоритама за предикцију успешности и упоређена је њихова тачност. Улазни параметри су класификовани на: демографске (пол, националност, место рођења, одговоран родитељ), карактеристике претходног школовања, карактеристике понашања и учешће родитеља у учењу. У студији [123] Ајдоџу предвиђа перформансе студената засноване на понашању ученика у окружењу за учење путем интернета. Идентификована су три кључна улаза вештачке неуронске мреже за допринос излазној варијабли: број похађања наставе уживо, број похађања архивираних курсева, време проведено за праћење одређеног садржаја. Креирана неуронска мрежа даје предвиђања са тачношћу од 80%. У [124] је доказано да је техника неуронских мрежа адекватна за проблеме у којима треба да се уради предикција вештине учесника, као што су аналитичко размишљање, решавање проблема као и склоност програмирању. Као неки од значајних параметара за предикцију наводе се пол, старост, ниво образовања родитеља, посао родитеља, оцена из математике, време које се користи за домаће задатке, поседовање електронског уређаја и време проведено са уређајем, као и знање о кодирању. Истраживање [125] приказује тестирање поступка за примену вештачких неуронских мрежа за предвиђање академских перформанси у високом образовању. Други циљ анализа важности неколико добро познатих предиктора академског учинка у високом образовању. За предикцију успешности студената који желе да постану пилоти, Џенкинс и остали [126], користе технику дата мајнинга. С обзиром на огромне ресурсе који се троше за обуку пилота предикција успешности усмерава правилан одабир кандидата.

Поређењем мотивације и предмета истраживања, предложено истраживање у докторској дисертацији је примењено на област програмирања, али се релативно једноставно може применити на друге области. Оригиналност истраживања огледа се у креирању специфичног инструмента за мерење успешности у програмирању, као и одабиру осталих релевантних параметара.

3. МЕТОДОЛОГИЈА ИСТРАЖИВАЊА

3.1. Проблем и предмет истраживања

Проблем истраживања

У данашњем дигиталном свету неопходно је креирање модела високошколских образовних програма који ће бити засновани на уважавању могућности образовних активности студената и откривању унутрашњих ресурса студента за постизање бољег успеха у тим активностима. Зато је задатак високошколског образовања да обезбеди оптималне услове за развој флексибилног и свеобухватног научног мишљења и да студенту, током академског школовања, створи унутрашњу потребу за саморазвојем и самообразовањем.

Нарочито је значајно да се, већ од прве године студија, када студенти формирају професионалне компетенције и траже своје место у струци, утврде фактори успешности образовних активности. Зато, откривање фактора успешног учења студената, који су повезани са развојем њихових способности за самосталним савладавањем нових знања, вештина и техника, добија све већу важност.

Проблем овог истраживања проистекао је из потребе за предвиђањем образаца понашања студената, као и њиховог будућег постигнућа у области информационих технологија, а посебно у области програмирања. Проблем предвиђања успешности студената је једна од кључних тема образовне психологије и науке о образовању. Овај проблем данас постаје све значајнији у контексту истраживања применљивости различитих ИТ алата и технологија за предвиђање понашања студената (са посебним освртом на примену техника дата мајнинга за предвиђање), што је подржано референтним истраживањима и радовима објављеним у референтним часописима.

Предмет истраживања

Потреба за предвиђањем образаца понашања студената, као и њиховог будућег постигнућа, постоји у скоро свим областима информационих технологија, а посебно у програмирању. Решења и модели предвиђања постигнућа студената, који су тренутно присутни, нису довољно прилагођени специфичностима информатичких области, нити специфичностима појединих подручја информационих технологија као што су: програмирање, рачунарске мреже, заштита података, заштита мрежа, базе података... Поред тога, крајњи корисници који немају посебне ИТ вештине везане за вештачке неуронске мреже имају потребу да одреде вредности улазних параметара и добију резултат за предвиђени параметар (у овом истраживању успешност у програмирању).

Предмет овог истраживања јесте дефинисање фактора и предиктора успешности у усвајању знања и вештина програмирања и примена вештачких неуронских мрежа за предвиђање нивоа постигнућа у области програмирања, као и развој веб базиране апликације за рад са дизајнираним моделом вештачких неуронских мрежа. Истраживање обухвата и академске перформансе и личне карактеристике студената уписаних на Факултет техничких наука у Чачку. Степен и ниво развијености проблема откривени су у многобројним истраживачким радовима: студенти као активни когнитивни субјекти у стицању професије, фактори успешне образовне активности, механизми прилагођавања студената.

3.2. Циљ и задаци истраживања

Општи циљ ове докторске дисертације је побољшање наставе програмирања као специфичног динамичког система, а посебно ради побољшања успешности студената у програмирању.

Циљеви истраживања:

1) Утврђивање фактора учења и стицања знања и вештина из програмирања. Овај истраживачки циљ биће развијен кроз следеће истраживачке задатке: теоријски проучити факторе учења; описати факторе учења студената и могућу везу између личних и интелектуалних квалитета и успеха у учењу; спровести емпиријско истраживање о факторима успеха у учењу; на основу емпиријског истраживања проучити однос између академског учинка и личних карактеристика студената; статистички одредити факторе успеха учења.
2) Израда и евалуација модела неуронске мреже ради предвиђања успешности у учењу програмирања која ће се вршити кроз креирање, евалуацију, тестирање и верификацију модела неуронске мреже.
3) Креирање веб базиране апликације за рад крајњег корисника (наставника) са креираним моделом вештачких неуронских мрежа за предвиђање успешности (избор улазних величина и добијање резултата – успешност у програмирању).
4) Систематизација знања о учењу програмирања и предвиђање успешности користећи вештачке неуронске мреже.

3.3. Хипотезе истраживања

А) Општа хипотеза

Х0: Могуће је развити модел вештачких неуронских мрежа тако да довољно прецизно предвиђа успешност студената у програмирању узимајући у обзир специфичности учења програмирања и чије су мере евалуације тачности у складу са резултатима валидације у реалним условима.

Б) Посебне хипотезе

Прва посебна хипотеза:

Х1: Студенти су успешнији у решавању репродуктивних задатака у односу на решавање креативних задатака, што омогућава развој прецизнијег модела предвиђања успешности.

Друга посебна хипотеза:

Х2: Постоје разлике у успешности у учењу програмирања између студената различитог претходног образовања у овој области: успешнији су студенти који су у току школовања учили више различитих програмских језика; успешнији су студенти који су поред формалног похађали и различите облике неформалног образовања из програмирања; успешнији су студенти који су завршили средње образовање из области информационих технологија. Наведена претпоставка омогућава развој прецизнијег модела предвиђања успешности.

Трећа посебна хипотеза:

Х3: Постоје разлике у успешности у учењу програмирања између студената који преферирају различите типове програмирања, као и између студената који преферирају различите стилове учења, а на основу којих је могућ развој прецизнијег модела за предвиђање успешности.

3.4. Методе истраживања

За реализацију емпиријског истраживања биће примењене: методе анализе садржаја; методе анкетирања и тестирања (за анализу ставова и знања студената); статистичке процедуре и методе: дескриптивна статистика, факторска анализа, корелациона анализа, регресиона анализа, Т-тестови (Т-тест упарених узорака (*repeated measures*), Т-тест независних узорака); једнофакторска анализа варијансе (*one-way ANOVA*); дискриминативна статистика.

За развој модела (базираног на резултатима теоријских и емпиријских анализа) биће примењене методе моделовања. За обраду података користиће се статистички софтвер *IBM SPSS 20* [127-129].

У истраживању ће бити коришћена вештачка неуронска мрежа, и то вишеслојна мрежа (*multilayer perceptron*) са три слоја (улазни, скривени и излазни слој). Креирање модела биће спроведено кроз фазе: прикупљање података, припрема података (претпроцесирање и трансформација) и моделовање; тренирање и тестирање неуронских мрежа (предложени алгоритам тренирања је *backpropagation*); интерпретација резултата неуронских мрежа. За креирање модела неуронских мрежа биће коришћен софтвер *WEKA 3.8.4* [130-132], за креирање веб базиране апликације за рад са креираним моделом вештачких неуронских мрежа користиће се *Microsoft Visual Studio 2017* [133, 134]. Улазни слој садржи податке о образовању и социодемографске податке (педагошке и социодемографске варијабле у истраживању). Излазни слој садржи један параметар – успешност у програмирању. Успешност у програмирању је композитна мера постигнућа добијена на основу резултата из предмета из области програмирања и резултата на тесту знања.

Осим наведених метода, идући од посебног, одабраног система односно високообразовне установе, користиће се индуктивна метода за добијање општих закључака о факторима учења програмирања и предвиђања успешности.

3.5. Променљиве величине (варијабле) у истраживању

У овом истраживању коришћене су три врсте варијабли: варијабле програмирања, педагошке варијабле и демографске варијабле.

- *Варијабле програмирања:* успешност у програмирању (резултат на тесту знања); редослед учења програмских језика; врсте задатака програмирања према степену креативности (репродуктивни, креативни); врсте програмских језика; преферирани тип програмирања (процедурално, објектно оријентисано, декларативно програмирање); самопроцена спремности за програмирање.
- *Педагошке варијабле:* образовни профил и успех у средњој школи, број бодова на пријемном испиту, година студија, просечна оцена, број остварених ЕСПБ и број положених предмета из програмирања до тренутка испитивања, облици формалног и неформалног образовања из области програмирања, стил учења.
- *Демографске варијабле:* пол; величина места у којем је завршена средња школа; мрежа образовних профила (заступљеност ИТ профила у месту школовања).

3.6. Технике истраживања

За прикупљање, обраду, анализу података и приказ резултата на којима се заснива развој модела, користиће се следеће технике истраживања:

- *За прикупљање података (већину инструмената је развио аутор)*:
    - за прикупљање *примарних података*: анкентни упитници, тестови знања из области програмирања, Колбов инвентар стилова учења, скале самопроцене;
    - за прикупљање *секундарних података*: подаци о успешности студената из интерне евиденције у високошколској установи и екстерни подаци о програмима средњошколских предмета из ИТ области (јавно доступни подаци МПНТР).
- *За обраду и анализу података*: статистичке процедуре (дескриптивна статистика, корелациона анализа, анализа варијансе...);
- *За приказ резултата*: табеле и графикони.

3.7. Карактеристике узорка истраживања

Узорак испитаника чинили су студенти Факултета техничких наука у Чачку који похађају студијски програм Информационе технологије. У летњем семестру школске године 2017/18 анкетирано је и тестирано 142 студента са све 4 године студија, док је из летњег семестра школске године 2018/19, у истраживање укључено и 38 студента прве године (1(2018)). У табели 3.7.1 приказана је структура узорка по годинама студија и по полу (32,78% студената женског пола и 67,22% мушког пола).

Табела 3.7.1: Структура узорка (Извор: аутор)

Година студирања	Женски	Мушки	Свега
1(2018)	18	20	38
1(2017)	6	37	43
2	18	18	36
3	10	23	33
4	7	23	30
Свега	**59**	**121**	**180**

Студенти који на ФТН у Чачку уписују Информационе технологије имали су могућност да за пријемни испит, који је вреднован са максималних 60 бодова, полажу математику или информатику, док преосталих 40 бодова носе из средње школе. Преглед просечног броја бодова са пријемног испита, односно из средње школе, као и број студената који је полагао одређени предмет на пријемним испитима дат је у табели 3.7.2. Што се тиче школског успеха студената обухваћених узорком, студенти четврте године (генерација 2014) су у средњој школи имали у просеку врлодобар успех (просечна оцена 4,14); студенти треће године (генерација 2015), друге године (генерација 2016) и прве године (генерација 2017) имали су „јако“ врлодобар успех (оцене 4,37, 4,31 и 4,39, респективно), док су студенти прве године који су испитивани у школској 2018/19 у средњој школи имали у просеку одличан успех (просечна оцена 4,50).

Табела 3.7.2: Подаци о пријемном испиту учесника истраживања (Извор: аутор)

Година студирања	Пријемни испит	Средња школа	Информатика	Математика
1 (2018)	40,41	36,02	26	12
1 (2017)	39,68	35,11	35	8
2	44,66	34,44	32	4
3	45,42	34,98	26	7
4	50,52	33,10	27	3
Свега	**43,69**	**34,81**	**146**	**34**

У табели 3.7.3 дата је структура узорка по завршеним образовним профилима у средњој школи. Скоро 2/3 (63,9%) учесника истраживања је завршило електротехничку (прва три образовна профила – 69 (38,3%)) или економску школу (образовни профили 10, 11 и 12 – 46 студента (25,6%)).

Табела 3.7.3: Статистика по образовним профилима (ОП[3]) (Извор: аутор)

Пол/година студирања	ОП 1	ОП 2	ОП 3	ОП 4	ОП 5	ОП 6	ОП 7	ОП 8	ОП 9	ОП 10	ОП 11	ОП 12	ОП 13	ОП 14	Свега
женски	**3**	**1**	**1**	**6**	**2**	**11**	**2**	**1**		**13**	**3**	**7**	**2**	**7**	**59**
1 (2018)	2		1	1		3				4		3		4	18
1 (2017)	1			3		1						1			6
2		1			1	2	1	1		3	2	2	2	3	18
3				2	1	3				3	1				10
4						2	1			3		1			7
мушки	**11**	**35**	**18**	**8**	**4**	**2**	**4**	**8**	**1**	**14**	**8**	**1**		**7**	**121**
1 (2018)	3	5	4	2				1		2	2			1	20
1 (2017)	8	10	4	2	1	1	1	2	1	4	2			1	37
2		4	2	1	2	1	3			2				3	18
3		10	2	2	1			2		3	2			1	23
4		6	6	1				3		3	2	1		1	23
Свега	**14**	**36**	**19**	**14**	**6**	**13**	**6**	**9**	**1**	**27**	**11**	**8**	**2**	**14**	**180**

За све учеснике истраживања урађена је статистика успешности (табела 3.7.4), закључно са априлским испитним роком у години када је истраживање обављено. За „програмерске“ предмете израчунат је проценат долазности на наставу/вежбе, проценат положених предмета, као и просечна оцена. Такође, дата је и статистика просечно остварених ЕСПБ као и укупна просечна оцена.

Табела 3.7.4: Статистика успешности учесника истраживања (Извор: аутор)

	Статистика „програмерских“ предмета				Статистика свих предмета	
Година студирања	**Број предмета**	**Похађано >50% наставе/вежби**	**% положених предмета**	**Просечна оцена**	**Просек ЕСПБ (32, 34, 87, 155, 210)**	**Просечна оцена**
1 (2018)	2	98,68%	97,37%	7,37	29,7	7,50
1 (2017)	2	90,70%	96,51%	8,36	28,8	7,67
2	3	98,15%	96,30%	8,07	83,2	7,84
3	4	88,64%	87,88%	8,24	140,3	7,92
4	7	84,29%	90,95%	7,66	198,0	7,67
Свега		**92,43%**	**94,14%**	**7,95**		**7,71**

Места из којих долазе учесници истраживања су класификована на: мала (мање од 20.000 становника), средња (20.000 – 100.000) и велика (више од 100.000 становника) (табела 3.7.5 [135]). Такође, анализирано је да ли у местима у којима су студенти завршили средњошколско образовање постоје специјализована ИТ одељења (електротехничар ИТ и гимназије са ИТ одељењима) [136]. Близу половине испитаника (45,6%) завршило је средњу школу у великим градовима, а скоро 70% испитаника (68,9%) је завршило школу у месту у којем постоје образовни профили са ИТ одељењима.

[3] **ОП1** – Електротехничар ИТ; **ОП 2** – Електротехничар рачунара; **ОП 3** – Електротехника остали (мреже, мултимедија, електроника, енергетика, телекомуникација); **ОП 4** – Гимназија општи смер; **ОП 5** – Гимназија природно-математички смер; **ОП 6** – Гимназија друштвено-језички смер; **ОП 7** – Гимназија информатички смер; **ОП 8** – Машински техничар за компјутерско конструисање; **ОП 9** – Техничар мехатронике; **ОП 10** – Економски техничар; **ОП 11** – Финансијски администратор; **ОП 12** – Економска струка остали (пословни администратор, службеник осигурања...); **ОП 13** – Саобраћајна струка (техничар друмског саобраћаја, безбедност саобраћаја...); **ОП 14** – Остали ОП (туристичка, медицинска, пољопривредна, прехрамбрена...)

Табела 3.7.5: Статистика по величини места и расположивости ИТ одељења (Извор: [135, 136])

Година студирања	Мало (<20.000)	Средње (20.000-100.000)	Велико >100.000)	„ИТ“ одељење НЕ	„ИТ“ одељење ДА
1 (2018)	7	18	13	14	24
1 (2017)	3	20	20	15	28
2	3	16	17	6	30
3	5	18	10	15	18
4	3	5	22	6	24
Свега	**21**	**77**	**82**	**56**	**124**

Средњошколци уписују факултете из различитих разлога. Студентима је понуђено пет разлога за упис студијског програма ИТ. У табелама 3.7.6 и 3.7.7 приказани су најважнији разлози уписа на студијски програм ИТ, по полу, односно по образовним профилима. Највећи проценат студената оба пола се за ИТ определио због знања које ће стећи (женски пол – 40,68%, мушки – 47,11%). Знање је и најважнији разлог уписа по образовним профилима – студенти који су завршили Гимназију природно-математички смер (66,67%). Образовни профил Мехатроника није узет у разматрање јер је у истраживању учествовао само један студент.

Табела 3.7.6: Разлози[4] уписа на смер ИТ (по полу) (Извор: аутор)

Пол	Знање	Посао	Веб	Базе	Мреже
Женски	40,68%	37,29%	18,64%	3,39%	0.00%
Мушки	47,11%	38,84%	10,74%	3,31%	0.00%
Свега	**45,00%**	**38,33%**	**13,33%**	**3,33%**	**0.00%**

Табела 3.7.7: Разлози уписа на студијски програм ИТ у % (по образовним профилима) (Извор: аутор)

Разлог уписа	ОП 1	ОП 2	ОП 3	ОП 4	ОП 5	ОП 6	ОП 7	ОП 8	ОП 9	ОП 10	ОП 11	ОП 12	ОП 13	ОП 14	Све-га
Знање	57,14	36,11	42,11	35,71	66,67	53,85	50,00	33,33	100,0	48,15	45,45	50,00	50,00	42,86	45,00
Посао	28,57	47,22	52,63	35,71	33,33	38,46	16,67	44,44	0,0	33,33	27,27	25,00	0,00	50,00	38,33
Веб	14,29	13,89	5,26	28,57	0,00	7,69	33,33	0,00	0,0	11,11	18,18	25,00	50,00	7,14	13,33
Базе	0,00	2,78	0,00	0,00	0,00	0,00	0,00	22,22	0,0	7,41	9,09	0,00	0,00	0,00	3,33
Мреже	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,0	0,00	0,00	0,00	0,00	0,00	0,00

3.8. Поступак истраживања

На почетку процеса је фаза **прикупљања** података и њихово складиштење у структуираном формату. На пример, ако се врши детекција непожељних порука, потребно је прво обрадити поруке електронске поште и додатне информације о њима. Ако се врши класификација тумора, на основу слика добијених путем ултразвука или магнетне резонанце, потребно је обрадити слике и из њих извући релевантне информације. Након што су подаци припремљени, прелази се у фазу **претпроцесирања**. Претпроцесирање подразумева одабир атрибута, подешавање тежина атрибута, решавање проблема недостајућих вредности, шума, итд. После претпроцесирања, улази се у фазу **формирања класификационог модела**. Разматра се и случај када постоји и повратна веза са фазом претпроцесирања, јер се одабир атрибута и подешавање њихових тежина извршава у фази тренирања модела. Битан аспект фазе тренинга је начин оцењивања квалитета класификације на тренинг подацима.

[4] **Знање** – стицање општег програмерског знања; **Посао** – сигурно запослење, добра зарада, могућност рада на даљину („од куће“); **Веб** – веб и графички дизајн/програмирање, мобилно рачунарство; **Базе** – администрирање рачунарских мрежа и база података; **Мреже** – одржавање серверских и клијентских мрежа.

3.8.1. Прикупљање података

Истраживање је спроведено на Факултету техничких наука у Чачку, са студентима студијског програма Информационе технологије. Истраживање је спроведено током 2018. године (студенти са све 4 године студија) и 2019. године (само студенти прве године). Прикупљање података је спроведено у две фазе:

- **Прва фаза**: у овој фази (март и април), на основу података о успешности студената из интерних евиденција за текућу и претходне школске године, извршен је одабир студената (по годинама студија) који ће учествовати у истраживању. За истраживање је укупно било одабрано 210 студената (табела 3.8.1). Изабрани студенти су обавештени (е-мејлом) о терминима и начину истраживања.
- **Друга фаза**: ова фаза је реализована у другој седмици маја, из разлога што се поменути временски период поклапа са периодом трајања летњег семестра и временом након одржаних испитних рокова, а пре завршетка семестра и испитних рокова у јуну и јулу, па се стога сматрало да изабрани студенти неће бити оптерећени овим истраживањем и да ће се масовно одазвати на њега. У табели 3.8.1 дат је преглед (по годинама студија): број студената који су позвани на истраживање, број студената који су учествовали у истраживању (узорак) и број студената који нису учествовали у истраживању (нису се одазвали на истраживање или су резултати њиховог истраживања били неупотребљиви). Због великог броја студената истраживање 2018. године, је обављено у два термина, у првом термину су учествовали студенти прве и четврте године, у другом термину (истог дана, одмах након завршетка првог термина) студенти друге и треће године. Обезбеђена је тајност и поверљивост свега што су студенти написали, тј. увид у њихове појединачне податке и резултате имао је само аутор овог рада. Студенти су попуњавали упитник који је био подељен у три дела (попуњавање сваког дела упитника је било временски ограничено):
    - *Први део упитника*: општи подаци о студенту, самопроцене студената у вези програмских језика и програмирања (ниво знања програмских језика, формално и неформално образовање програмирања, тренутна спремност за програмирањем, тип програмирања који преферирају, ниво програмерске способности за рад у ИТ индустрији...);
    - *Други део упитника* представља тест знања из области програмирања; студенти су давали одговарајуће одговоре заокруживањем тачног одговора од понуђених одговора, односно уписивањем одговора/кôда задатка у предвиђен простор;
    - *Трећи део упитника* (Колбов инвентар стилова учења) је намењен одређивању начина на који студенти лично уче (није намењен за одређивање капацитета студената за учење и способности, већ само њихов начин учења).

Табела 3.8.1: Укупан број студената позваних на истраживање и број студената у узорку према години студија (Извор: аутор)

Година студирања	Термин	Одабрано/ позвано N	Радили истраживање		Узорак (учествовали у истраживању)		Нису учествовали у истраживању	
			N1	N1/N (%)	N2	N2/N (%)	N3	N3/N (%)
1(2018)	мај 2019.	43	38	88,4%	38	88,4%	5	11,6%
1(2017)	мај 2018.	45	43	95,6%	43	95,6%	2	4,4%
2	мај 2018.	42	38	90,5%	36	85,7%	6	14,3%
3	мај 2018.	41	35	85,4%	33	80,5%	8	19,5%
4	мај 2018.	39	32	82,5%	30	76,9%	9	23,1%
Свега		**210**	**186**	**88,6%**	**180**	**85,7%**	**30**	**14,3%**

3.8.2. Селекција података

Селекција (одабир) података је уобичајени корак издвајања података из скупа података (селектује се само део узорка) на основу одабраног критеријума. Приликом селекције података потребно је поћи од истраживачких питања, опсега студије, претходних истраживања која су утврдила најприкладније податке за прикупљање, типа података (квантитативни, квалитативни).

У овом истраживању селекција није примењена јер сви подаци који су, у току овог истраживања прикупљени, су и искоришћени.

3.8.3. Претпроцесирање

Претпроцесирање података један је од најважнијих задатака анализе података и он укључује припрему података пре примене метода истраживања података (сирови подаци морају бити претпроцесирани како би се учинили погоднијим за рад). Подаци који су у изворном облику могу да буду некомплетни, могу да имају атрибуте којима недостају неке вредности, могу да постоје неконзистентне вредности (нпр. недоследност у означавању појединих категорија или група) и слично. Ако се подаци не претпроцесирају, најчешће, се добијају непоуздани и лоши резултати [137].

- Од 186 студента који су се одазвали на истраживање упитници 6-оро студената (3,2%) нису коришћени као узорак, односно искључени су из истраживања, јер су њихови одговори били незадовољавајући.
- Код 11 испитаника било је неопходно, код скала процена у првом делу упитника, попунити недостајуће вредности ручно. Недостајуће вредности су попуњене просечном вредношћу одговора за испитанике који припадају истој години студирања.
- Код 3 испитаника, у трећем делу упитника (Колбов инвентар стилова учења), било је дуплираних одговора. Ти подаци су замењени тако што су коришћене највероватније вредности за тај атрибут за све преостале испитанике.
- Код 5 испитаника искључени су подаци за недостајуће одговоре у трећем делу упитника (Колбов инвентар стилова учења), а остали подаци из упитника су коришћени за анализу.

3.8.4. Трансформација

Након прикупљања и претпроцесирања података, потребно је податке трансформирати, тј. довести их у облик који је погодан за разумевање и истраживање. Трансформација података је урађена у неколико корака:

- Кодирани почетни подаци приказани су у облику матрице, при чему се у редовима налазе испитаници, а у колонама одговори на различита питања/задатке из упитника;
- Извршено је шифровање (кодирање) података. На пример, за затворено структуирано питање са вишеструким избором одговора о спремности за програмирањем одговори су кодовани: Нисам сигуран/на – код 1, Уз нечију помоћ – код 2, Једноставан програм – код 3, Сложен програм један ПЈ – код 4, Сложен програм више ПЈ – код 5.
- Над појединим подацима извршена је агрегација (операције здруживања). На пример у другом делу упитника (тест) извршено је сабирање резултата теста за одговоре ДА/НЕ;

- Подаци ниског нивоа (сирови подаци) замењени су концептима вишег нивоа – поступак генерализације. На пример, атрибут место, пресликан је на концепт вишег нивоа – мало, средње и велико.
- Извршена је нормализација података, тј. скалирање података са почетно великим распонима тако да они падају у мали распон. На пример, укупан број бодова са пријемног испита који је у опсегу 0 до 60 нормализован је на опсег 0,0 до 1,0. Нормализација је потребна због неуронских мрежа.

3.8.5. Развијање и оптимизација модела неуронских мрежа

За креирање неуронске мреже целокупан узорак истраживања подељен је на два дела:

- *Подузорак за тренирање и тестирање* – студенти који су учествовали у истраживању у мају 2018. године – укупно 142 испитаника (78,9%);
- *Подузорак за валидацију* – студенти прве године генерација 2018 – укупно 38 испитаника (21,1%)

Све предикторске величине груписане су у 21 улазну независну променљиву (табела 3.8.2), док је излаз једна зависна променљива која представља успех у програмирању (*неуспешан*, *средње успешан* и *веома успешан*).

Табела 3.8.2: Преглед улазних променљивих (Извор: аутор)

Ред. бр.	Назив променљиве	Кодирање
1.	Бодови из школе	Број бодова из средње школе (од 16 до 40)
2.	Бодови са пријемног испита	Остварени бодови на пријемном испиту (макс. 60)
3.	Предмет на пријемном испиту	Предмет који је полаган на пријемном испиту (1-Информатика; 2-Математика)
4.	Пол	1-мушки; 2-женски
5.	Образовање	Завршен образовни профил у средњој школи (1-Електротехничар ИТ; 2-Електротехничар рачунара; 3-Електротехника остали (мреже, мултимедија, електроника, енергетика, телекомуникација); 4-Гимназија општи; 5-Гимназија природно-математички; 6-Гимназија друштвено-језички; 7-Гимназија информатички; 8-Машински техничар за компјутерско конструисање; 9-Техничар мехатронике; 10-Економски техничар; 11-Финансијски администратор; 12-Економска остали (пословни администратор, службеник осигурања); 13-Техничар друмског саобраћаја, безбедност саобраћаја; 14-Остали (туристичка, медицинска, пољопривредна, прехрамбрена...))
6.	Величина места	1-испод 20.000 становника; 2-од 20.000 до 100.000 становника; 3-преко 100.000 становника
7.	Постојање ИТ одељења у месту	Да ли постоји ИТ одељење у месту у коме је завршена средња школа (1-ДА; 2-НЕ)
8.	Година студија	1-прва (генерација 2018) – валидација; 2-прва (генерација 2017); 3-друга, 4-трећа; 5-четврта
9.	Просечна оцена „програмерских“ предмета	Од 6,0 до 10,0
10.	Проценат посећивања наставе	Од 0,0 до 100,0
11.	Разлог уписа	Најважнији разлог за упис студијског програма ИТ (1-стицање општег програмерског знања; 2-сигурно запослење, добра зарада, могућност рада на даљину („од куће“); 3-веб и графички дизајн/програмирање, мобилно рачунарство; 4-администрирање рачунарских мрежа и база података; 5-одржавање серверских и клијентских мрежа)

Ред. бр.	Назив променљиве	Кодирање
12.	Спремност за програмирање	Спремност за креирањем програма (1-нисам сигуран/на; 2-уз нечију помоћ; 3-једноставан програм; 4-сложен програм на једном програмском језику; 5-сложен програм на више програмских језика)
13.	Решавање секвенцијалних задатака	Колико би било тешко/лако да студенти реше ову програмску структуру (1-веома тешко, 2-тешко; 3-осредње; 4-лако; 5-веома лако)
14.	Решавање задатака са условним и цикличним структурама	1-веома тешко, 2-тешко; 3-осредње; 4-лако; 5-веома лако
15.	Решавање сложених задатака	1-веома тешко, 2-тешко; 3-осредње, ни тешко ни лако; 4-лако; 5-веома лако
16.	Тип програмирања	Који тип програмирања највише преферирају (1-објектно оријентисано програмирање; 2-процедурално програмирање; 3-функционално и декларативно програмирање
17.	Спремност за рад у ИТ индустрији	Ниво програмерске спремности за рад у ИТ индустрији од 1 до 5 (1-најнижи; 5-највиши)
18.	Ниво знања програмских језика	Просечна оцена нивоа знања за понуђене програмске језике (од 1,0 до 5,0)
19.	Формално образовање	Укупан број програмских језика за које су студенти имали формално образовање, од 0 до 11 (за 11 понуђена ПЈ)
20.	Неформално образовање	Од 0 до 11
21.	Колбов стил учења	1-дивергер; 2-акомодатор; 3-конвергер; 4-асимилатор; 5-недиференциран

За предвиђање успешности у програмирању коришћен је *модел трослојне неуронске мреже базиране на алгоритму учења са простирањем грешке уназад (backpropagation)*. За имплементацију мреже коришћена је софтверски пакет Weka 3.8.4.

3.8.5.1. Софтвер WEKA

WEKA (*Waikato Environment for Knowledge Analysis*) [130-132] је графички кориснички интерфејс (*GUI*), софтвер отвореног кода који је развијен на Универзитету *Waikato* на Новом Зеланду. Садржи четири апликације: истраживач (*explorer*), експериментална примена (*experimental*), проток знања (*knowledge flow*) и интерфејс командне линије (*CLI*).

WEKA садржи и алате за претпроцесирање података, класификацију, регресију, кластеризацију, правила придруживања и визуализацију. Заснован је на програмском језику *Java*, користи се у многим различитим областима примене, посебно за образовање и истраживање. Главни кориснички интерфејс софтвера *WEKA* је *Explorer* [131] који има много панела помоћу којих се приступа главним компонентама. Технике софтвера *WEKA* засноване су на претпоставци да су подаци доступни као једна датотека или релација где је сваки податак описан фиксним бројем атрибута. *WEKA* омогућава приступ *SQL* базама података користећи *Java Database Connectivity*.

Дизајн модела неуронске мреже помоћу софтвера *WEKA* одвија се у следећим корацима:

- **Избор скупа података и атрибута**. Датотека подака који се анализирају мора бити у *CSV* или *ARFF* формату.
- **Претпроцесирање**. Као што је већ наглашено, претпроцесирање је важан корак који се користи за издвајање и побољшање квалитета података. Након учитавања података у *Explorer* могуће је, одабиром различитих опција, извршити чишћење података (брисање записа – инстанци) или уклањање одређених атрибута.

- **Класификација.** Концепт класификације је у основи дистрибуција података између различитих класа дефинисаних на скупу података. Класификациони алгоритми (класификатори) за класификацију уче овај облик дистрибуције из датог скупа обуке, а затим покушавају да га правилно класификују када су у питању подаци о тестовима за које класа није наведена. *WEKA* укључује огромну количину класификатора. Постоји много опција за класификаторе, од којих се већина односи на процену.
- **Начини тестирања**. Постоје четири уобичајена начина тестирања:
  - *Use training set*. Класификује се модел на основу скупа података са којим је првобитно обучен модел.
  - *Supplied test set*. Класификатор се процењује на основу тога колико добро предвиђа класу скупа инстанци учитаних из датотеке (*Set...*).
  - *Cross-validation*. Опција унакрсне провере (валидације) посебно се користи ако је количина скупова података ограничена. Оригинални скуп података насумично се дели на n различитих подскупова (слојева) приближно исте величине (подразумевана вредност је 10). Након тога, *WEKA* користи први слој за тестирање и n-1 слојева за први тренинг. Поступак се понавља n пута (n итерација), тако да се сваки од слојева по једном користи за тестирање. На крају се добија оцена стварне перформансе предикционог модела (оцена квалитета модела) као просечна перформанса свих n претходних предвиђања.
  - *Percentage split*. Код евалуације помоћу скупа за тестирање, оригинални скуп података дели се на два подскупа: подскуп за обучавање и подскуп за тестирање класификатора. Подразумевана подела процента поделе је 66%, што значи да се 66% података користити као за обуку, а осталих 34% за тестирање. На основу података за обуку (тренирање) добија се класификациони модел, док се тачност класификације процењује на основу података за тестирање.
- **Филтери**. Улазни скупови података често поседују недостатке, последица тих недостатака је неупотребљивост таквих података за потребе класификације. Квалитет класификације се може побољшати одређеним техникама претпроцесирања и оптимизације. Секција претпроцесирање омогућава дефинисање филтера који трансформишу податке на разне начине. Углавном постоје две категорије филтера – *надгледани* (*Supervised*) и *ненадгледани* (*Unsupervised*).
- **Избор атрибута** (поскупови атрибута). Постојање већег броја улазних атрибута повећава вероватноћу да одређен број атрибута неће бити релевантни, односно ти атрибути могу имати негативан утицај на процес креирања предикционог модела. Значи, већи број улазних атрибута није предуслов за постизање веће тачности предиктивног модела. *WEKA* апликација нуди коришћење метода за аутоматски избор (селектовање) подскупова атрибута за креирање предиктивних модела. Избор атрибута укључује претрагу свих могућих комбинација атрибута у подацима да би се утврдило који подскуп атрибута најбоље одговара за предвиђање. Да би се то постигло, морају се поставити два објекта: *attribute evaluator* (*евалуатор атрибута*) и *search method* (*метод претраживања*). Евалуатор одређује која се метода користи за додељивање вредности сваком подскупу атрибута. Метод претраживања одређује који се стил претраживања изводи. Избор потенцијалних подскупова (*subsets*) постиже се одабиром једног од 11 евалуатора (*CfsSubsetEval, ClassifierAttributeEval, ClassifierSubsetEval, CorelationAttributeEval, GainRationAttributeEval, InfoGainAttributeEval, OneRAttributeEval, PrincipalComponents, ReliefFAttributeEval, SymmetricalUncertAttributeEval, WrapperSub-setEval)* у комбинацији са једним од три метода претраживања (*BestFirst, GreedyStepwise, Ranker).*

На све изабране подскупове креираће се предиктивни модели применом класификатора (алгоритма) ***MultilayerPerceptron***, за два начина тестирања (***Cross-validation*** и ***Percentage split***) и са применом ***Resample*** филтера над њима. Модел са најбољом предиктивном тачношћу биће искоришћен као коначан модел за имплементацију.

3.8.6. Евалуација и тестирање модела неуронских мрежа

3.8.6.1. Мерење грешака

За процену тачности метода предиктивне класификације користе се разне мере грешака. Оне се односе на способност модела да тачно предвиди ознаку класе нових или претходно некоришћених података. За одређивање тачности алгоритма користе се други критеријуми оцењивања као што су брзина, скалабилност, поузданост и једноставност.

Мере грешака које се користе за процену техника класификације су [138, 139]:

- **Средња апсолутна грешка** – *Mean Absolute Error (MAE)*. Ова грешка мери просек апсолутне вредности разлике између предвиђених и стварних вредности.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - x_i| \tag{2}$$

  где су: y_i предвиђене вредности, x_i стварне вредости, а n укупан број података доступних за анализу.

- **Корен средње квадратне грешке** – *Root Mean Sequared Error (RMSE)*. То је квадратни корен средње вредности квадрата свих грешака (разлика између предвиђених и стварних вредности).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - x_i)^2} \tag{3}$$

  где су: y_i предвиђене вредности, x_i стварне вредости, а n укупан број података доступних за анализу.

- **Релативна апсолутна грешка** – *Relative Absolute Error (RAE)*. *RAE* је укупна апсолутна грешка и мери разлику између стварне вредности и појединачно измерене вредности.

$$RAE = \frac{\sum_{i=1}^{n}|y_i - x_i|}{\sum_{i=1}^{n}|\bar{x} - x_i|} \tag{4}$$

  где су: y_i предвиђене вредности, x_i стварне вредости, n укупан број података доступних за анализу, а $\bar{x}$ је средња вредност стварне вредности.

- **Корен релативне квадратне грешке** – *Root Relative Squared Error (RRSE)*. *RRSE* узима укупну квадратну грешку и нормализује је дељењем са укупном квадратном грешком једноставног предиктора. Узимајући квадратни корен релативне квадратне грешке, грешка се смањује на исте димензије као и величина која се предвиђа.

$$RRSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - x_i)^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}} \tag{5}$$

  где су: y_i предвиђене вредности, x_i стварне вредости, n укупан број података доступних за анализу, а $\bar{x}$ је средња вредност стварне вредности.

3.8.6.2. Утврђивање тачности класификације

Тачност се дефинише као удео тачне класификације у односу на укупан број случајева и зависи од **матрице конфузије** (матрице грешака). То је врста табеле која помаже да се сазнају перформансе класификационог модела на скупу тест података за које су познате праве вредности. Матрица конфузије представља упоредни однос између броја предвиђених (прогнозираних) и стварних (циљних) класа за неки скуп вектора атрибута [138, 140, 141].

Најједноставнија матрица конфузије дефинише проблем бинарне класификације који има само две класе за класификацију (по могућности позитивну и негативну класу). Табела 3.8.3 [142] приказује матрицу конфузије за проблем бинарне класификације. Код ове матрице могућа су 4 различита предвиђања (исхода):

- **Стварно позитивни** (*TP - true positive*) исходи – број правилно класификованих позитивних случајева, тј. број предвиђања где класификатор тачно предвиђа позитивну класу као позитивну.
- **Стварно негативни** (*TN - true negative*) исходи – број правилно класификованих негативних случајева, тј. број предвиђања где класификатор тачно предвиђа негативну класу као негативну.
- **Лажно позитивни** (*FP - false positive*) исходи – број нетачно класификованих позитивних случајева, тј. број предвиђања где класификатор погрешно предвиђа негативну класу као позитивну.
- **Лажно негативни** (*FN - false negative*) исходи – број погрешно класификованих негативних случајева, тј. број предвиђања где класификатор погрешно предвиђа позитивну класу као негативну.

Исходи *TP* и *TN* представљају исправну класификацију, док *FP* и *FN* исходи представљају два могућа типа грешке. *FP* исход је негативан пример класе која је грешком класификована као позитивна, док је *FN* исход заправо позитиван пример класе која је погрешно класификована као негативна.

Табела 3.8.3: Матрица конфузије за бинарну класификацију (Извор: [142])

Confusion Matrix 2x2		**Предвиђена класа (МОДЕЛ)**	
		Позитивни	**Негативни**
Стварна класа	Позитивни	TP	FN
	Негативни	FP	TN

Увек је боље користити матрицу конфузије као критеријуме за процену модела машинског учења. Матрица конфузије даје врло једноставне, а ефикасне мере перформанси за модел.

У овом истраживању потребно је класификовати успешност студената у програмирању у три различите категорије (3 класе):

- Класа a – *средње успешан*
- Класа b – *веома успешан*
- Класа c – *неуспешан*

У том случају резултат класификације приказује се у облику дводимензионалне матрице димензије 3x3 (табела 3.8.4). Редови матрице одговарају стварним вредностима класа вектора атрибута, док колоне представљају класе додељене од стране модела.

Табела 3.8.4: Матрица конфузије за класификацију класе 3 (Извор: аутор)

Confusion Matrix 3x3		**Предвиђена класа (МОДЕЛ)**		
		a	**b**	**c**
Стварна класа	a = средње упешан	n_{aa} (TP_a), (TN_b), (TN_c)	n_{ab} (FN_a), (FP_b), (TN_c)	n_{ac} (FN_a), (TN_b), (FP_c)
	b = веома успешан	n_{ba} (FP_a), (FN_b), (TN_c)	n_{bb} (TN_a), (TP_b), (TN_c)	n_{bc} (TN_a), (FN_b), (FP_c)
	c = неуспешан	n_{ca} (FP_a), (TN_b), (FN_c)	n_{cb} (TN_a), (FP_b), (FN_c)	n_{cc} (TN_a), (TN_b), (TP_c)

За разлику од бинарне класификације, овде нема позитивних или негативних класа. Потребно је пронаћи *TP*, *TN*, *FP* и *FN* за сваку појединачну класу. Нпр. за класу означену са „b“ (веома успешан) вредности показатеља из матрице конфузије су следеће

$$TP_b = n_{bb};\ TN_b = n_{aa} + n_{ac} + n_{ca} + n_{cc};\ FP_b = n_{ab} + n_{cb};\ FN_b = n_{ba} + n_{bc}.$$

На основу вредности матрице број тачних предвиђања је $n_{aa} + n_{bb} + n_{cc}$, док је укупан број нетачних предвиђања једнак $n_{ab} + n_{ac} + n_{ba} + n_{bc} + n_{ca} + n_{cb}$.

Из добијених информација од матрице конфузије могу се добити релевантне вредности за процену успешности модела [142-151]:

- **Тачност** (*Accuracy* – A_{cc}) показује однос укупног броја тачних предвиђања према укупном броју свих предвиђања [142, 143].

$$A_{CC} = \frac{n_{aa} + n_{bb} + n_{cc}}{n_{aa} + n_{bb} + n_{cc} + n_{ab} + n_{ac} + n_{ba} + n_{bc} + n_{ca} + n_{cb}} \tag{6}$$

- **Одзив** (*Recall* – R_i) се користи за мерење процента стварних позитивних случајева који су тачно идентификовани, тј. представља однос тачних позитивних предвиђања према свим **позитивним класама** [142, 143].

$$R = \frac{TP}{TP + FN} \tag{7}$$

$$R_a = \frac{n_{aa}}{n_{aa} + n_{ab} + n_{ac}}, R_b = \frac{n_{bb}}{n_{ba} + n_{bb} + n_{bc}}, R_c = \frac{n_{cc}}{n_{ca} + n_{cb} + n_{cc}} \tag{8}$$

- **Прецизност** (*Precision* – P_i) представља однос тачних позитивних предвиђања и укупног броја свих **позитивних предвиђања**.

$$P = \frac{TP}{TP + FP} \tag{9}$$

$$P_a = \frac{n_{aa}}{n_{aa} + n_{ba} + n_{ca}}, P_b = \frac{n_{bb}}{n_{ab} + n_{bb} + n_{cb}}, P_c = \frac{n_{cc}}{n_{ac} + n_{bc} + n_{cc}} \tag{10}$$

  Фокус прецизности је на позитивним предвиђањима, тако да показује колико је позитивних предвиђања тачно. Фокус одзива је стварна позитивна класа, тако да показује колико је позитивних класа модел у стању да правилно предвиди. Повећавање прецизности смањује одзив и обрнуто [142, 143].

- **F-мера** (*F-measure* – F_i) представља однос између прецизности и одзива (хармонијска средина прецизности и одзива). За постизање равнотеже прецизности и одзива, циљ је што већа *F*-мера [142, 143].

$$F = 2\frac{PR}{P + R} \tag{11}$$

$$F_a = \frac{2P_aR_a}{P_a + R_a}, F_b = \frac{2P_bR_b}{P_b + R_b}, F_a = \frac{2P_cR_c}{P_c + R_c} \tag{12}$$

- **Осетљивост** (*Sensitivity*) или стварна позитивна стопа (*True Positive Rate – TPR*$_i$), иста је као и одзив – мери пропорцију позитивне класе која је тачно предвиђена као позитивна [142, 143].

$$\text{Sensitivity} = TPR = \frac{TP}{TP + FN} \tag{13}$$

- **Специфичност** (*Specificity*) или стварна негативна стопа (*True Negative Rate – TNR*) је слична осетљивости, али је усредсређена на негативну класу. Она мери пропорцију негативне класе која је тачно предвиђена као негативна [142, 143].

$$\text{Specificity} = TNR = \frac{TN}{TN + FP} \tag{14}$$

- **Лажно позитивна стопа** (*False Positive Rate – FPR*$_i$) је удео негативних случајева који су у подацима погрешно идентификовани као позитивни случајеви (тј. вероватноћа да ће се покренути лажна упозорења) [142, 143].

$$FPR = 1 - TNR = \frac{FP}{TN + FP} \tag{15}$$

$$FPR_a = \frac{n_{ba} + n_{ca}}{n_{bb} + n_{bc} + n_{cb} + n_{cc} + n_{ba} + n_{ca}}, FPR_b = \frac{n_{bb}}{n_{ba} + n_{bb} + n_{bc}}, FPR_c = \frac{n_{cc}}{n_{ca} + n_{cb} + n_{cc}} \tag{16}$$

- **Лажно негативна стопа** (*False negative rate – FNR*) је супротна стварној негативној стопи (TNR) [142, 143].

$$\text{FNR} = 1 - \text{Specificity} = \frac{FN}{FN + TP} \tag{17}$$

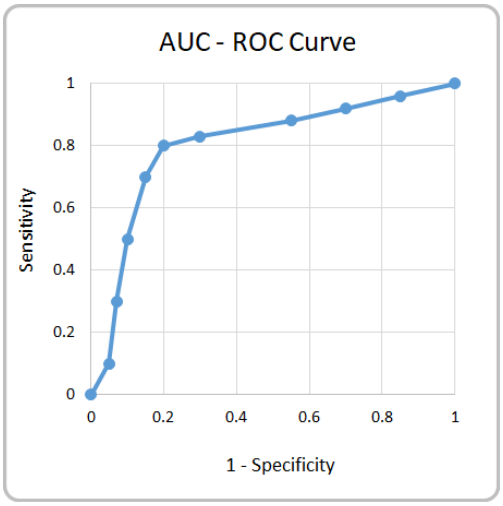
- **Капа статистика** (*Kapa statistic*) или капа вредност је метрика која упоређује уочену тачност са очекиваном тачношћу, тј. капа представља меру слагања између предвиђених и стварних класификација у скупу података.

$$\text{Kappa} = (\text{уочена тачност} - \text{очекивана тачност}) / (1 - \text{очекивана тачност}) \tag{18}$$

  Капа статистика се користи не само за процену појединачног класификатора, већ и за оцењивање класификатора између себе. Поред тога, узима у обзир случајну шансу (слагање са случајним класификатором), што генерално значи да је мање обмањујуће него једноставно коришћење тачности као метрике (тачност од 80% је много мање импресивна са очекиваном тачношћу од 75% наспрам очекиване тачности од 50%). При тумачењу капа статистике треба узети у обзир најмање два додатна разматрања. Прво, ако је могуће, капа статистику увек треба упоређивати са придруженом матрицом конфузије како би се добила најтачнија интерпретација. Друго, прихватљиве статистичке вредности капа варирају у зависности од контекста. На пример, у многим студијама поузданости међу оцењивачима са лако уочљивим понашањем, вредности капа статистике испод 0,70 могу се сматрати ниским. Међутим, у студијама које користе машинско учење за истраживање неприметних појава (појава које не могу да се посматрају) статистичке вредности капа изнад 0,40 могу се сматрати изузетним [144, 145].

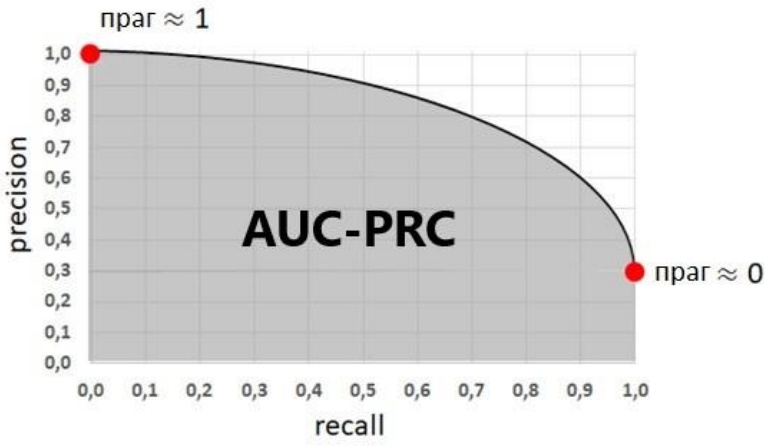
- **ROC (Receiver Operating Characteristic) Area** (подручје) је још једна корисна статистичка карактеристика за испитивање перформанси класификатора. Концепт *ROC* се надовезује на знање о матрици конфузије, специфичности и осетљивости. За обједињавање информација из матрица конфузија корисни су *AUC-ROC* графови. *AUC-ROC* графови су дводимензионални у којима се стварне позитивне стопе

(*Sensitivity=TPR*) цртају на Y оси, а лажне позитивне стопе под различитим праговима вероватноће (*1 – Specificity = FPR*) на X оси за низ различитих вредности прага између 0 и 1. Свака тачка у *ROC* криви произилази из вредности у матрици конфузије повезане са применом одређеног пресека на предвиђањима (резултатима) класификатора. Да би се конструисала *ROC* крива, једноставно се користи свака процена класификатора као пресек за разликовање позитивне од негативне класе. На слици 3.8.1 [147] приказана је *ROC* крива за 10 инстанци. Предиктивне перформансе класификатора могу се квантификовати у смислу вредности (површине) испод *ROC* криве – *Area Under Curve (AUC)* која лежи у опсегу [0,1]. *Напомена: AUC-ROC крива није прикладна за мерење перформанси модела класификације за високо неуравнотежене податке* [146, 147].



Слика 3.8.1: AUC-ROC крива креирана прагом скупа од 10 инстанци (Извор: [147])

- ***Precision-Recall Curve (PRC) Area*** (*PRC* подручје). Графици прецизног опозива (*PRC*) представљају графиконе одзива (*Recall (Sensitivity)*) наспрам прецизности (*Precision*), тј. показују компромис између прецизности и опозива под различитим праговима вероватноће. Будући да *PRC* криве не узимају у обзир стварне негативне стопе, требало би их користити само када специфичности нису у интересу класификатора. *AUC-PRC*, која се назива просечна прецизност, може се израчунати кроз интеграл и омогућава једнократно упоређивање вредности између модела или тестова. *Напомена: AUC-PRC крива је прикладна за мерење перформанси модела класификације за високо неуравнотежене податке* [148-150].



Слика 3.8.2: AUC-PRC крива (Извор: [150])

- **Matthews-ов коефицијент корелације *MCC*** (*Matthews correlation coefficient*) се може схватити као специфичан случај линеарног коефицијента корелације за поставку бинарне класификације, заправо мери корелацију стварних класа c са предвиђеним ознакама l.

$$MCC = \frac{Cov(c,l)}{\sigma_c \cdot \sigma_l} = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)\cdot(TP+FN)\cdot(TN+FP)\cdot(TN+FN)}} \tag{19}$$

(најгора вредност = -1; најбоља вредност = + 1) где је: $Cov(c,l)$ коваријанција истинских класа c и предвиђених ознака l, док су σ_c и σ_l стандардна одступања [151].

4. РЕЗУЛТАТИ И ДИСКУСИЈА

Резултати истраживања су презентовани по редоследу хипотеза истраживања.

4.1. Мерење успешности студената у програмирању

Како је општи циљ ове докторске дисертације побољшање наставе програмирања, ради побољшања успешности студената у програмирању, било је неопходно измерити успешност студената.

Успешност студената се мерила бодовима до максималних 100, кроз два фактора:

- **Успех до тренутка истраживања** (овај фактор је учествовао у мерењу са 14% – 14 бодова). Бодован је постигнут успех из 8 предмета из области програмирања („програмерски“ предмети): проценат положених предмета (са 4%) и просечна оцена положених предмета (са 10%). У табели 4.1.1 дат је списак предмета које су студенти могли да положе до тренутка истраживања.

Табела 4.1.1: Предмети из области програмирања из којих је мерен успех студената (Извор: аутор)

Р.Б	Назив предмета	Година студија			
		Прва	Друга	Трећа	Четврта
1.	Увод у програмирање	ДА	ДА	ДА	ДА
2.	Практикум из програмирања	ДА			
3.	Програмски језици		ДА	ДА	ДА
4.	Објектно оријентисано програмирање		ДА	ДА	ДА
5.	Веб технологије			ДА	ДА
6.	Интернет програмирање				ДА
7.	Програмирање базе података				ДА
8.	Софтверско инжењерство				ДА

Напомена: Практикум из програмирања је предмет који се изучава од школске 2017. године

- **Резултати постигнути на тесту знања из програмирања** (овај фактор је учествовао у мерењу са 86% – 86 бодова). Тест знања је био посебно конструисан за ово истраживање и садржи 14 задатака који су разврстани у 4 групе. У табели 4.1.2 дат је укупан број задатака по типовима, као и колико који задатак носи бодова.

Табела 4.1.2: Спецификација задатака за тест знања (Извор: аутор)

Тип задатка (групе)	Број задатака	Бодовање	
		Један задатак	Укупно
Двочлани избор (тачно/нетачно)	4	2,5	10
Вишеструки избор (понуђен један тачан одговор)	4	5	20
Допуњавање (уписивање тачног одговора)	4	7	28
Есејски (писање програмског кода)	2	14	28
	14		**86**

Напомена: Есејски задаци су могли бити вредновани и са 50% тачности, тј. 7 бодова („50% тачног задатка“).

4.2. Репродуктивни и креативни задаци (прва хипотеза)

X1: Студенти су успешнији у решавању репродуктивних задатака у односу на решавање креативних задатака, што омогућава развој прецизнијег модела предвиђања успешности.

Задаци, које су студенти решавали у тесту знања, груписани су у две категорије (табела 4.2.1):

- **Репродуктивни** (*репетитивни*) **задаци** – то су базични задаци који су неопходни у настави програмирања који помажу студентима да стекну основна знања из програмирања и усмерени су ка неговању памћења студената. Задаци из теста су слични задацима који су презентовани (обрађивани) на часовима наставе и вежби. Ови задаци су вредновани укупно са 31,5 бодова.
- **Креативни** (*продуктивни*) **задаци** – за решавање ових задатака студенти су морали да буду мисаоно активни, оригинални и флексибилни, морали су да повежу претходна усвојена знања из програмирања. Задаци из теста нису обрађивани на часовима („нови задаци“) и укупно су вредновани са 54,5 бодова.

Табела 4.2.1: Спецификација репродуктивних и креативних задатака за тест знања (Извор: аутор)

Тип задатка (групе)	**Репродуктивни задаци**		**Креативни задаци**	
	Број	**Бодови**	**Број**	**Бодови**
Двочлани избор (тачно/нетачно)	3	7,5	1	2,5
Вишеструки избор (понуђен један тачан одговор)	2	10	2	10
Допуњавање (уписивање тачног одговора)	2	14	2	14
Есејски (писање програмског кода)	/	/	2	28
	7	**31,5**	**7**	**54,5**

У провери прве посебне хипотезе претпостављено је да студенти показују боље резултате у решавању репродуктивних задатака у односу на решавање креативних задатака. За проверу ове хипотезе користићемо *Т-тест упарених узорака*. Резултати су дати у табелама 4.2.2 и 4.2.3.

Табела 4.2.2: Тест упарених узорака (репродуктивни и креативни задаци)(Извор: аутор)

Paired Samples Test								
	Paired Differences					**t**	**df**	**Sig. (2-tailed)**
	Mean	**Std. Deviation**	**Std. Error Mean**	**95% Confidence Interval of the Difference**				
				Lower	**Upper**			
Проценат бодова из репродуктивних задатака – Проценат бодова из креативних задатака	8,66379	26,7396	1,99307	4,73445	12,60033	4,349	179	,000

Paired Differences – упарене разлике; Mean – аритметичка средина (M); Std. Deviation – стандардна девијација (SD); Std. Error Mean – стандардна средња грешка; 95% Confidence Interval of the Difference – интервал 95% поузданости разлике, Lower – доња граница; Upper – горња граница; t – t вредност; df – степен слободе; Sig. (2-tailed) – ниво значајности (p)

Како је тражена вероватноћа доношења погрешног закључка (ниво значајности) p=0,000 (вероватноћа је заокружена на три децимале, а заправо није једнака нули) мања од 0,05 (p<0,05), закључујемо да *постоји значајна статистичка разлика у проценту успешности студената при решавању репродуктивних и креативних задатака*. Број степени слободе df=179, вредност t(179)=4,35, просечна разлика у процентуалној успешности репродуктивних и креативних задатака је 8,66, док се интервал 95-процентне поузданости разлике протеже од доњих 4,73 до горњих 12,60.

Табела 4.2.3: Статистика упарених узорака (репродуктивни и креативни задаци) (Извор: аутор)

Paired Samples Statistics				
	Mean	**N**	**Std. Deviation**	**Std. Error Mean**
Проценат бодова из репродуктивних задатака	57,1432	180	24,57310	1,83157
Проценат бодова из креативних задатака	48,4758	180	23,29303	1,73616

На основу података из табеле 4.2.3 закључујемо да су *студенти постигли боље резултате при решавању репродуктивних у односу на креативне резултате*. При решавању репродуктивних задатака студенти су од могућих 31,50 бодова остварили просечно 18,00 бодова (M=57,14%), док је, од максималних 54,50 бодова, код креативних задатака просечан број остварених бодова 26,42 (M=48,48%).

Показатељ величине утицаја у Т-тесту упарених узорака [129],

$$\text{Ета квадрат}_{\text{Т–тест}} = \frac{t^2}{t^2 + N - 1} = \frac{4{,}29^2}{4{,}29^2 + 180 - 1} = 0{,}096 \qquad (20)$$

указује на то да је утврђена умерено велика разлика према Коеновом критеријуму (табела 4.2.4 [129]).

Табела 4.2.4. Величина утицаја према Коену (Извор: [129])

Величина утицаја	**Ета квадрат (% објашњења варијансе)**	**Коенов d (јединица стандардног одступања)**
Мали	0,01 или 1%	0,2
Средњи	0,06 или 6%	0,5
Велики	0,138 или 13,8%	0,8

На основу претходних закључака:

- да постоји значајна статистичка разлика у проценту успешности студената при решавању репродуктивних и креативних задатака;
- да су студенти постигли боље резултате при решавању репродуктивних у односу на креативне резултате,

Хипотеза X1: *Студенти су успешнији у решавању репродуктивних задатака у односу на решавање креативних задатака, што омогућује развој прецизнијег модела предвиђања успешности* **је доказана.**

4.3. Претходно образовање (друга хипотеза)

Друга посебна хипотеза:

X2: Постоје разлике у успешности у учењу програмирања између студената различитог претходног образовања у овој области.

Да би доказали другу хипотезу потребно је доказати следеће подхипотезе:

X2а: успешнији су студенти који су у току школовања учили више различитих програмских језика;

X2б: успешнији су студенти који су поред формалног похађали и различите облике неформалног образовања из програмирања;

X2в: успешнији су студенти који су завршили средње образовање из области информационих технологија.

Анкетирани студенти, поред формалног образовања (образовање у основној, средњој школи, на факултету), своја знања и вештине из програмирања увећавали су и кроз неформално образовање – образовање ван школа (курсеви, семинари, радионице, тренинзи, онлајн образовање, волонтирање...). У табели 4.3.1 дати су статистички подаци о формалном и неформалном образовању за понуђена 11 програмска језика.

Табела 4.3.1: Формално и неформално образовање програмских језика (Извор: аутор)

Програмски језик	**Формално образовање**		**Неформално образовање**	
	N	**%**	**N**	**%**
Basic	17	9,4	8	4,4
Pascal	32	17,8	12	6,7
Delphi	16	8,9	7	3,9
C	174	96,7	67	37,2
C++	124	68,9	50	27,8
C#	57	31,7	39	21,7
Java	81	45,0	81	45,0
Python	2	1,1	19	10,6
Visual Basic for Applications	19	10,6	11	6,1
Matlab	49	27,2	8	4,4
LabVIEW	11	6,1	3	1,7

Програмски језик *C* је изучавало највише студената у току школовања – 96,7%. Програмски језик *Python* је изучавало у току школовања само 2 студента, али је зато њих 19-оро имало неформално образовање из овог програмског језика. Охрабрује да је, чак 45% студената имало неформално образовање из програмског језика *Java*.

У табели 4.3.2 дат је број студената, за оба начина образовања, који су изучавали: више од три програмска језика, два и три програмска језика и један програмски језик.

Табела 4.3.2: Групе по броју изучаваних програмских језика за формално и неформално образовање (Извор: аутор)

Група	**Формално образовање**		**Неформално образовање**	
	N	**%**	**N**	**%**
Ниједан програмски језик	/	/	56	31,1
Један програмски језик	43	23,9	52	28,9
Два и три програмска језика	58	32,2	44	24,4
Више од три програмска језика	79	43,9	28	15,6

Сви студенти су у току формалног образовања изучавали бар један програмски језик (на факултету – старије генерације *Pascal*, новије генерације програмски језик *C*), скоро 44% студената имало је формално образовање за више од три програмска језика. Ван школе, 31,1% студената није имало образовање из ниједног програмског језика, четвртина студената (24,4%) је имало неформално образовање из два и три програмска језика.

4.3.1. Подхипотеза X2a

У провери подхипотезе X2a претпостављено је да су *успешнији студенти који су у току школовања (формално образовање) учили више различитих програмских језика*. За проверу ове хипотезе користићемо *једнофакторску анализу варијансе (ANOVA)*. Резултати су дати у табелама 4.3.3 – 4.3.7.

Табела 4.3.3: Статистички подаци о групама по броју изучаваних програмских језика (формално образовање) (Извор: аутор)

Описи								
Укупно поена – Успех + Тест								
	N	**Mean**	**Std. Deviation**	**Std. Error**	**95% Confidence Interval for Mean**		**Mini-mum**	**Maxi-mum**
					Lower Bound	**Upper Bound**		
1-Један програмски језик	43	43,728	18,5901	2,8350	38,007	49,449	13,0	83,0
2-Два и три програмска језика	58	53,060	14,9467	1,9626	49,130	56,990	25,5	81,0
3-Више од три програмска језика	79	65,162	15,3654	1,7287	61,720	68,604	31,1	97,5
Свега	180	56,142	18,1965	1,3563	53,466	58,819	13,0	97,5

Студенти који су у току школовања изучавали два и три програмска језика имају бољи просечан број бодова M=53,06 у односу на студенте који су изучавали само један програмски језик (M=43,73), а имају лошији резултат од студената који су у току формалног образовања изучавали више од 3 програмска језика (M=65,16). Максималан број бодова 97,5 (максимум 100 бодова) је постигао студент прве године (генерација 2018) из групе „Више од три програмска језика“, док је минимум бодова (13,0) остварио студент прве године (генерација 2017) који је изучавао само један програмски језик.

Ниво значајности p је значајно већи од 0,05 (0,643>0,05), што значи да је *задовољена претпоставка о једнакости варијансе у групама*, тј. није прекршена претпоставка о хомогености варијансе. На то указује Левенов тест хомогености варијансе (табела 4.3.4), F(2,177)=0,443, p=0,643.

Табела 4.3.4: Левенов тест хомогености варијансе за групу програмских језика (формално образовање) (Извор: аутор)

Тест хомогености варијација			
Укупно поена – Успех + Тест			
Левенова статистика	**df1**	**df2**	**Sig.**
,443	2	177	,643

У случају да је p мање од 0,05 (прекршена је претпоставка о хомогености варијанте) резултате, уместо из *ANOVA*, потражили би у табели *Robust Tests of Equality of Means* (табела 4.3.5) у којој су приказани резултати два теста (*Welch* і *Brown-Forsythe*).

Табела 4.3.5: Robust Tests of Equality of Means за групу програмских језика (формално образовање) (Извор: аутор)

Robust Tests of Equality of Means				
Укупно поена – Успех + Тест				
Тестови	**Статистика***	**df1**	**df2**	**Sig.**
Welch	23,954	2	98,410	,000
Brown-Forsythe	24,875	2	131,513	,000

* Асимптотски распоређено F

Пошто је вредност p значајно већа од 0,05 (0,643), због лакше интерпретације резултата користимо *ANOVA* табелу (табела 4.3.6). У овој табели дати су суме квадрата одступања резултата од њихове средње вредности, број степени слободе, итд. за анализу различитих група и анализу истих субјеката.

Табела 4.3.6: ANOVA табела за групу програмских језика (формално образовање) (Извор: аутор)

ANOVA					
Укупно поена – Успех + Тест					
	Sum of Squares	**df**	**Mean Square**	**F**	**Sig.**
Различите групе	13605,028	2	6802,514	26,367	,000
Исти субјекти	45664,411	177	257,991		
Свега	59269,439	179			

Sum of Squares – сума квадрата

Резултат једнофакторске *ANOVA* анализе указује да *постоје статистички значајне разлике у успешности студената који су у току формалног образовања похађали различит број програмских језика: p=0,000<0,05, F(2,177)=23,367*. Величина утицаја једнофакторске анализе варијансе (*ANOVA*) рачуна се по формули [129]:

$$\text{Ета квадрат}_{ANOVA}=\frac{\text{Збир квадрата одступања различитих група}}{\text{Укупан збир квадрата}}=\frac{13605,028}{59269,439}=0,23 \quad (21)$$

Како је вредност ета квадрата = 0,23, то по Коеновом критерујуму (табела 4.2.4), следи да је *утицај разлике велики*. За одређивање која се група разликује од других група, неопходно је погледати резултате накнадних тестова у табели вишеструког поређења (табела 4.3.7).

Табела 4.3.7: Статистичке значајности разлика између сваког пара група програмских језика (формално образовање) (Извор: аутор)

Multiple Comparisons						
Зависна варијабла: Укупно поена – Успех + Тест **Tukey HSD**						
(I) 1-Један језик, 2-Два и три језика, 3-Више од 3 језика	**(J) 1-Један језик, 2-Два и три језика, 3-Више од 3 језика**	**Mean Difference (I-J)**	**Std. Error**	**Sig.**	**95% Confidence Interval**	
					Lower Bound	**Upper Bound**
Један програмски језик	Два и три програмска језика	-9,3324*	3,2323	,012	-16,972	-1,693
	Више од три програмска језика	-21,4341*	3,0439	,000	-28,629	-14,239
Два и три програмска језика	Један програмски језик	9,3324*	3,2323	,012	1,693	16,972
	Више од три програмска језика	-12,1017*	2,7774	,000	-18,666	-5,537
Више од три програмска језика	Један програмски језик	21,4341*	3,0439	,000	14,239	28,629
	Два и три програмска језика	12,1017*	2,7774	,000	5,537	18,666
*Две упоређене групе међусобно се значајно разлику на нивоу p<0,05						

Накнадна поређења, помоћу *Tukey*-евог *HSD* теста, указују на то да статистички значајне разлике постоје (звездице поред исписаних бројева за средњу разлику) између студената који су формално похађали:

- 1-Један програмски језик и 2-Два и три програмска језика – p=0,012<0,05. Средња вредност групе 1 (M=43,73, SD=18,59) разликује се од средње вредности групе 2 (M=53,06, SD=14,95);
- 1-Један програмски језик и 3-Више од три програмска језика – p=0,000<0,05. Средња вредност групе 1 (M=43,73, SD=18,59) значајно се разликује од средње вредности групе 3 (M=65,16, SD=15,37);
- 2-Два и три програмска језика и 3-Више од три програмска језика – p=0,000<0,05. Средња вредност групе 2 (M=53,06, SD=14,95) значајно се разликује од средње вредности групе 3 (M=65,16, SD=15,37).

Значи, студенти који су формално похађали више различитих програмских језика су статистички успешнији од студената који су учили мање програмских језика, тј:

успешност(3-Више од три програмска језика) > успешност(2-Два и три програмска језика) > успешност(1-Један програмски језик)

Тиме је **подхипотеза Х2а**: *успешнији су студенти који су у току школовања учили више различитих програмских језика* **доказана**.

4.3.2. Подхипотеза Х2б

У провери подхипотезе Х2б претпостављено је да су *успешнији студенти који су у току школовања поред формалног образовања похађали и различите облике неформалног образовања из програмирања*. За проверу ове хипотезе користићемо *једнофакторску анализу варијансе (ANOVA)*. Резултати су дати у табелама 4.3.8 – 4.3.12.

Табела 4.3.8: Статистички подаци о групама по броју изучаваних програмских језика (неформално образовање) (Извор: аутор)

Описи								
Укупно поена – Успех + Тест								
	N	**Mean**	**Std. Deviation**	**Std. Error**	**95% Confidence Interval for Mean**		**Mini-mum**	**Maxi-mum**
					Lower Bound	**Upper Bound**		
1-Ниједан програмски језик	56	46,655	13,8246	1,8474	42,953	50,358	13,0	71,0
2-Један програмски језик	52	62,598	19,2313	2,6669	57,244	67,952	19,5	97,5
3-Два и три програмска језика	44	58,636	17,8035	2,6840	53,224	64,049	18,0	88,0
4-Више од три програмска језика	28	59,207	17,7664	3,3575	52,318	66,096	26,7	84,5
Свега	180	56,142	18,1965	1,3563	53,466	58,819	13,0	97,5

Студенти који поред формалног образовања нису изучавали ниједан програмски језик имају лошији просечан број бодова (M=46,66) у односу на студенте који су неформално учили барем један програмски језик. На основу Левеновог теста хомогености варијансе (табела 4.3.9), F(3,176)=2,612, ниво значајности p је незнатно већи од 0,05 (0,053>0,05), што значи да је *задовољена претпоставка о једнакости варијансе у групама*, тј. није прекршена је претпоставка о хомогености варијансе

Табела 4.3.9: Левенов тест хомогености варијансе за групу програмских језика (неформално образовање) (Извор: аутор)

Тест хомогености варијација			
Укупно поена – Успех + Тест			
Левенова статистика	**df1**	**df2**	**Sig.**
2,612	3	176	,053

Значи, није прекршена ниједна од претпоставки *ANOVA*. Пошто је вредност p већа од 0,05 (0,053) користимо *ANOVA* табелу (табела 4.3.10).

Табела 4.3.10: ANOVA табела за групу програмских језика (неформално образовање) (Извор: аутор)

ANOVA					
Укупно поена – Успех + Тест					
	Sum of Squares	**df**	**Mean Square**	**F**	**Sig.**
Различите групе	7744,031	3	2581,344	8,817	,000
Исти субјекти	51525,409	176	292,758		
Свега	59269,439	179			

Резултат једнофакторске *ANOVA* анализе указује да *постоје статистички значајне разлике у успешности студената који су поред формалног образовања имали неформално образовање различитих програмских језика: p=0,000<0,05, F(3,176)=8,666*. Величина утицаја једнофакторске анализе варијансе (*ANOVA*), једнака је:

$$\text{Ета квадрат}_{ANOVA}=\frac{7744,031}{59269,439}=0,13 \quad (22)$$

Како је вредност ета квадрата = 0,13, то по Коеновом критеријуму (табела 4.2.4), следи да је *утицај разлике велики*. За одређивање која се група разликује од других група, неопходно је погледати резултате накнадних тестова у табели вишеструког поређења (табела 4.3.11).

Табела 4.3.11: Статистичке значајности разлика између сваког пара група (неформално образовање) (Извор: аутор)

Multiple Comparisons						
Зависна варијабла: Укупно поена – Успех + Тест **Tukey HSD**						
(I) 1-Ниједан језик 2-Један језик, 3-Два и три језика, 4-Више од 3 језика	**(J) 1-Ниједан језик 2-Један језик, 3-Два и три језика, 4-Више од 3 језика**	**Mean Difference (I-J)**	**Std. Error**	**Sig.**	**95% Confidence Interval**	
					Lower Bound	**Upper Bound**
Ниједан програмски језик	Један програмски језик	-15,9427*	3,2951	,000	-24,489	-7,396
	Два и три програмска језика	-11,9810*	3,4469	,004	-20,921	-3,041
	Више од три програмска језика	-12,5518*	3,9602	,010	-22,824	-2,280
Један програмски језик	Ниједан програмски језик	15,9427*	3,2951	,000	7,396	24,489
	Два и три програмска језика	3,9617	3,5048	,671	-5,129	13,052
	Више од три програмска језика	3,3909	4,0107	,833	-7,012	13,794
Два и три програмска језика	Ниједан програмски језик	11,9810*	3,4469	,004	3,041	20,921
	Један програмски језик	-3,9617	3,5048	,671	-13,052	5,129
	Више од три програмска језика	-,5708	4,1363	,999	-11,299	10,158
Више од три програмска језика	Ниједан програмски језик	12,5518*	3,9602	,010	2,280	22,824
	Један програмски језик	-3,3909	4,0107	,833	-13,794	7,012
	Два и три програмска језика	,5708	4,1363	,999	-10,158	11,299
* Две упоређене групе међусобно се значајно разлику на нивоу p<0,05						

Накнадна поређења, помоћу *Tukey*-евог *HSD* теста, указују на то да статистички значајне разлике постоје (звездице поред исписаних бројева за средњу разлику) између студената који ван школе (неформално образовање) нису похађали ни један програмски језик у односу на студенте који су неформално изучавали бар један програмски језик:

- 1-Ниједан програмски језик и 2-Један програмски језика – p=0,000<0,05. Средња вредност групе 1 (M=46,66, SD=13,82) значајно се разликује од средње вредности групе 2 (M=62,60, SD=19,23);
- 1-Ниједан програмски језик и 3-Два и три програмска језика – p=0,004<0,05. Средња вредност групе 1 (M=46,66, SD=13,82) значајно се разликује од средње вредности групе 3 (M=58,64, SD=17,80);
- 1-Ниједан програмски језик и 4-Више од три програмска језика – p=0,02<0,05. Средња вредност групе 1 (M=46,66, SD=13,82) разликује се од средње вредности групе 4 (M=56,142, SD=18,20).

Значи, студенти који су поред формалног образовања, ван школе, изучавали барем један програмски језик, статистички су успешнији од студената који ван школе нису изучавали ниједан програмски језик.

Тиме је **подхипотеза Х2б**: *успешнији су студенти који су поред формалног похађали и различите облике неформалног образовања из програмирања* **доказана**.

4.3.3. Подхипотеза Х2в

Студенти који студирају ИТ на ФТН у Чачку завршавају различите *образовне профиле* у средњој школи (табела 3.7.3). Број дневних/седмичних часова из предмета у којима се учи/примењује програмирања је, на тим образовним профилима, веома различит. Тако нпр., образовни профили „Електротехничар ИТ“ и „Гимназија за ученике са посебним способностима за рачунарство и информатику (Информатички смер)“ имају просечно респективно: 1,38 часова, односно 1,11 часова на дневном нивоу (табела 2.3.8); средњошколци који су завршили природно-математички смер у гимназијама имају само 0,17 часова на дневном нивоу (0,85 часова седмично); поједини образовни профили нису ни имали ниједан час програмирања у току четворогодишњег школовања (економска струка, туристичка, медицинска, пољопривредна, прехрамбена).

Четрнаест образовних профила груписано је, према седмичном просечном броју часова програмирања, у две групе: прва група – студенти који су завршили средње образовање из области информационих технологија (Електротехничар ИТ, Гимназија информатички смер), њихов седмични број часова из програмирања прелази 5 часова (у просеку више од 1 часа дневно), и друга група – студенти осталих образовних профила (табела 4.3.12).

Табела 4.3.12: Образовни профили из средње школе, груписани према седмичном просечном броју часова програмирања (Извор: аутор)

Група	Седмични фонд часова (h)	Образовни профили	Узорак	
			N	%
прва	h > 5	Електротехничар ИТ, Гимназија информатички смер	30	16,7
друга	h < 3	Електротехничар рачунара, Електротехника остали (мреже, мултимедија, електроника, енергетика, телекомуникација), Машински техничар за компјутерско конструисање, Техничар мехатронике, Гимназија општи смер, Гимназија природно-математички смер, Гимназија друштвено-језички смер, Економски техничар, Финансијски администратор, Економска струка остали (пословни администратор, службеник осигурања...), Саобраћајна струка (техничар друмског саобраћаја, безбедност саобраћаја...), Остали ОП (туристичка, медицинска, пољопривредна, прехрамбрена...)	150	83,3

У провери подхипотезе Х2в претпостављено је да су *успешнији студенти који су завршили средње образовање из области информационих технологија (прва група)*. За проверу ове хипотезе користићемо *Т-тест независних узорака*. Резултати су дати у табелама 4.3.13 – 4.3.17.

Табела 4.3.13: Статистички подаци о групама по броју седмичних часова из програмирања (h > 5 – ИТ одељења) (Извор: аутор)

Групна статистика					
	Образовни профил из средње школе груписани према седмичном броју часова из програмирања	N	Mean	Std. Deviation	Std. Error Mean
Укупно поена – Успех + Тест	1 h > 5	30	67,287	16,3953	2,9934
	2 h < 3	150	53,913	17,7611	1,4502

Видимо да су студенти који су у средњој школи били у тзв. „ИТ одељењима“ (образовни профили: Електротехничар ИТ и Гимназија информатички смер) постигли боље резултате (M=67,29, SD=16,40) него студенти који су завршили остале образовне профиле (M=53,913, SD=17,76).

На основу Левеновог теста једнакости варијансе (табела 4.3.14), F=0,454, ниво значајности p је знатно већи од 0,05 (0,501>0,05), што значи да *претпоставка о једнакости варијанси није нарушена*.

Табела 4.3.14: T-тест независних узорака за групу образовних профила (Извор: аутор)

Тест независних узорака										
		Левенеов тест за једнакост варијација		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Differe-nce	Std. Error Differe-nce	95% Confidence Interval of the Difference	
									Lower	Upper
Укупно поена – Успех + Тест	Претпоставка о једнакости варијанси је задовољена	,454	,501	3,811	178	,000	13,3733	3,5092	6,4484	20,2982
	Није задовољена претпоставка о једанкости варијанси			4,021	43,742	,000	13,3733	3,3261	6,6688	20,0779

У том случају треба узети податке из првог реда табеле 4.3.14 (*Претпоставка о једнакости варијанси је задовољена*), израчунате за случај једнаких варијанси. У случају да ниво значајности Левенеовог теста износи p=0,05 или мање од 0,05, то значи да варијансе ове две групе студената нису једнаке (подаци не задовољавају претпоставке о једнакости варијансе), тада треба користити податке из друге врсте резултата T-теста (*Није задовољена претпоставка о једанкости варијанси*) који су израчунати без претпостављања једнакости варијанси.

Како је, у нашем случају, ниво значајности p мањи од 0,05 (0,00<0,05), значи да *постоји значајна разлика између средњих вредности зависне променљиве (Укупно поена) у свакој од ове две групе (t(178)=3,811)*. Да би одредили величину утицаја, тј. величину разлике између група, потребно је да одредимо ета квадрат и Коенов d. Ета квадрат може да има вредност у опсегу од 0 до 1 и представља пропорцију варијансе у зависној променљивој објашњену независном променљивом (груписања). Ета квадрат за T-тест независних узорака налази се по формули [129]:

$$\text{Ета квадрат}_{\text{Т-тест}} = \frac{t^2}{t^2 + (N_{h>5} + N_{h<3} - 2)} = \frac{3{,}811^2}{3{,}811^2 + 30 + 150 - 2} = 0{,}075 \qquad (23)$$

Значи, разлика између средњих вредности обележја по групама (просечна разлика =13,37; 95% интервал поверења: од 6,45 до 20,30) је умерена (ета квадрат = 0,075).

Тиме је **подхипотеза Х2в**: *успешнији су студенти који су завршили средње образовање из области информационих технологија* **доказана.**

Како су доказане све три подхипотезе: Х2а: успешнији су студенти који су у току школовања учили више различитих програмских језика; **Х2б:** успешнији су студенти који су поред формалног похађали и различите облике неформалног образовања из програмирања; **Х2в:** успешнији су студенти који су завршили средње образовање из области информационих технологија; **тиме је доказана и друга посебна хипотеза:**

Х2*: Постоје разлике у успешности у учењу програмирања између студената различитог претходног образовања у овој области*.

4.4. Типови програмирања и стилови учења (трећа хипотеза)

Трећа посебна хипотеза:

Х3: Постоје разлике у успешности у учењу програмирања између студената који преферирају различите типове програмирања и различите стилове учења.

Да би доказали трећу хипотезу потребно је доказати следеће подхипотезе:

Х3а: постоје разлике у успешности у учењу програмирања између студената који преферирају различите типове програмирања;

Х3б: постоје разлике у успешности у учењу програмирања између студената који преферирају различите стилове учења.

4.4.1. Подхипотеза Х3а

Анкетирани студенти су процењивали, оценом од 1 (највише) до 3 (најмање), који тип програмирања највише преферирају: *Објектно оријентисано програмирање, Процедурално програмирање* или *Функционално и декларативно програмирање*. У табели 4.4.1 дати су статистички подаци о преферираном типу програмирања.

Табела 4.4.1: Преферирани тип програмирања (Извор: аутор)

Тип програмирања	**N**	**%**
Објектно оријентисано програмирање (ООП)	128	71,1
Процедурално програмирање (ПП)	14	7,8
Функционално и декларативно програмирање (ФП)	38	21,1

Најмањи број студената преферира процедурално програмирање (7,8%), а мало више од 70% (71,1%) преферира објектно оријентисано програмирање. То је добар показатељ, јер развој софтвера се креће од процедуралног програмирања (*C, Pascal, BASIC, Fortran, Ada,...*) ка објектно оријентисаним програмским језицима (*Java, C++, C#, PHP, Python,...*), који су дизајнирани да подржавају процедурално програмирање. Анкетирани студенти, након изучавања програмског језика *C* (процедурални програмски језик) у првом семестру, у другом семестру изучавају објектно оријентисани програмски језик *C++*, а у наредним годинама студија изучавају и остале ООП језике (друга година *Java* и *C#*; трећа година *PHP*, *Python*...). Функционално програмирање (*SQL, HTML, XML, LISP, Prolog,...*) преферира нешто више од петине анкетираних студената (21,1%), студенти који су највише окренути ка базама података и веб дизајну.

У провери подхипотезе Х3а претпостављено је да *постоје значајне разлике у успешности у учењу програмирања између студената који преферирају различите типове програмирања*. За проверу ове хипотезе користићемо *једнофакторску анализу варијансе (ANOVA)*. Резултати су дати у табелама 4.4.2 – 4.4.5.

На основу средњих вредности успешноси у програмирању (табела 4.4.2), закључује се да је група студената који преферирају процедурално програмирање најуспешнија (M=60,75), док је најмање успешна група студената која преферира функционални тип програмирања (M=45,81).

Ниво значајности p је значајно већи од 0,05 (0,421>0,05), што значи да је *задовољена претпоставка о једнакости варијансе у групама*, тј. није прекршена је претпоставка о хомогености варијансе. На то указује Левенов тест хомогености варијансе (табела 4.5.3), F(2,177)=0,869.

Табела 4.4.2: Статистички подаци о групама по преферираним типовима програмирања (Извор: аутор)

Описи								
Укупно поена – Успех + Тест								
	N	**Mean**	**Std. Deviation**	**Std. Error**	**95% Confidence Interval for Mean**		**Mini-mum**	**Maxi-mum**
					Lower Bound	**Upper Bound**		
1-Објектно оријентисано програмирање	128	58,705	17,7777	1,5713	55,595	61,814	13,0	97,5
2-Процедурално програмирање	14	60,750	18,8238	5,0309	49,881	71,619	20,0	83,0
3-Функционално програмирање	38	45,813	15,8825	2,5765	40,593	51,034	18,0	76,0
Свега	180	56,142	18,1965	1,3563	53,466	58,819	13,0	97,5

Табела 4.4.3: Левенов тест хомогености варијансе за групу преферирани типови програмирања (Извор: аутор)

Тест хомогености варијација			
Укупно поена – Успех + Тест			
Левенова статистика	**df1**	**df2**	**Sig.**
,869	2	177	,421

Очигледно је да није прекршена ниједна од претпоставки *ANOVA*, јер је p>0,05 (0,421). Због лакше интерпретације користимо *ANOVA* табелу (табела 4.1.4).

Табела 4.4.4: ANOVA табела за групу преферирани типови програмирања (Извор: аутор)

ANOVA					
Укупно поена – Успех + Тест					
	Sum of Squares	**df**	**Mean Square**	**F**	**Sig.**
Различите групе	5191,924	2	2595,962	8,497	,000
Исти субјекти	54077,516	177	305,523		
Свега	59269,439	179			

Резултат једнофакторске *ANOVA* анализе указује да *постоје статистички значајне разлике у успешности између студената који преферирају различите типове програмирања: p=0,000<0,05, F(2,177)=8,497*.

Величина утицаја ета квадрата је:

$$\text{Ета квадрат}_{ANOVA}=\frac{5191,924}{59269,439}=0,088 \quad (24)$$

Како је вредност ета квадрата = 0,088, то по Коеновом критеријуму (табела 4.2.4), следи да је *утицај разлике умерен* (мада се може рећи да је и близу великог утицаја, јер је вредност ета квадрата незнатно мања од 0,14). За добијање детаљнијег увида у разлике између група, неопходно је погледати резултате накнадних тестова у табели вишеструког поређења (табела 4.5.5).

Накнадна поређења, помоћу *Tukey*-евог *HSD* теста, указују:

- да постоје статистички значајне разлике (звездице поред исписаних бројева за средњу разлику) између студената који преферирају:
  - 1-Објектно оријентисано програмирање (ООП) и 3-Функционално програмирање; p=0,000<0,05 (*Mean Difference*=12,89). Средња вредност прве групе (M=58,71, SD=17,78) значајно се разликује од средње вредности групе 3 (M=45,81, SD=15,88). Студенти који преферирају ООП су успешнији у програмирању од студената који преферирају функционални тип програмирања.

- 2-Процедуларни тип програмирања и 3-Функционални тип програмирања; p=0,019<0,05 (*Mean Difference*=14,94). Средња вредност групе 2 (M=60,75, SD=18,82) значајно се разликује од средње вредности групе 3 (M=45,81, SD=15,88). Студенти који преферирају процедурално програмирање су успешнији у програмирању од студената који преферирају функционални тип програмирања. Ово је уједно и највећа разлика између поменутих група.

- не постоје статистички значајне разлике у успешности студената у програмирању између групе која преферира ООП и групе која преферира процедурално програмирање (p=0,909>0,05)

Табела 4.4.5: Статистичке значајности разлика између сваког пара група преферираних типова програмирања (Извор: аутор)

Multiple Comparisons						
Зависна варијабла: Укупно поена – Успех + Тест **Tukey HSD**						
(I) 1-ООП, 2-Процедурални, 3-Функционални	**(J) 1-ООП, 2-Процедурални, 3-Функционални**	**Mean Difference (I-J)**	**Std. Error**	**Sig.**	**95% Confidence Interval**	
					Lower Bound	**Upper Bound**
ООП	Процедурални	-2,0453	4,9204	,909	-13,675	9,584
	Функционални	12,8915*	3,2291	,000	5,259	20,524
Процедурални	ООП	2,0453	4,9204	,909	-9,584	13,675
	Функционални	14,9368*	5,4647	,019	2,020	27,853
Функционални	ООП	-12,8915*	3,2291	,000	-20,524	-5,259
	Процедурални	-14,9368*	5,4647	,019	-27,853	-2,020
* Две упоређене групе међусобно се значајно разлику на нивоу p<0,05						

Значи, студенти који преферирају објектно оријентисано и процедурално програмирање, статистички су успешнији од студената који преферирају функционални тип програмирања.

Тиме је **подхипотеза Х3а**: *постоје разлике у успешности у учењу програмирања између студената који преферирају различите типове програмирања* **доказана**.

4.4.2. Подхипотеза Х3б

Трећи део упитника (*Колбов инвентар стилова учења*) треба да покаже да ли постоје разлике у успешности у учењу програмирања између студената који преферирају различите стилове (начине) учења. У табели 4.4.6 дати су статистички подаци о преферираним стиловима учења.

Табела 4.4.6: Преферирани стил учења (Извор: аутор)

Стил (начин) учења	**N**	**%**
Дивергентни	31	17,2
Акомодирајући	34	18,9
Конвергентни	52	28,9
Асимилирајући	39	21,7
Неодређени	24	13,3

Код анкетираних студената ниједан стил учења није доминантан, студенти највише (28,9%), преферирају конвергентни стил учења (практичари – „како то функционише и шта можемо урадити са тим“), док најмање преферирају дивергентни стил учења (рефлексивни мислиоци – „шта ће се догодити и зашто“) – 17,2% и акомодирајући стил

(активисти – „шта најбоље можемо да учинимо у овој ситуацији“) – 18,9%. Скоро сваки пети студент (21,7%) преферира асимилирајући стил учења (теоретичари – „шта знамо о томе“), док 13,3% студената није сврстано у ниједну групу стила учења (неодређени).

У провери подхипотезе Х3б претпостављено је да *постоје значајне разлике у успешности у учењу програмирања између студената који преферирају различите стилове учења*. За проверу ове хипотезе користићемо *једнофакторску анализу варијансе (ANOVA)*. Резултати су дати у табелама 4.4.7 – 4.4.10.

Табела 4.4.7: Статистички подаци о групама по прифериранима стиловима учења (Извор: аутор)

Описи								
Укупно поена – Успех + Тест								
	N	**Mean**	**Std. Deviation**	**Std. Error**	**95% Confidence Interval for Mean**		**Mini-mum**	**Maxi-mum**
					Lower Bound	**Upper Bound**		
1-Дивергентни	31	53,374	20,7928	3,7345	45,747	61,001	13,0	97,5
2-Акомодирајући	34	55,909	15,8141	2,7121	50,391	61,427	20,0	80,5
3-Конвергентни	52	57,704	19,1312	2,6530	52,378	63,030	20,0	92,5
4-Асимилирајући	39	55,751	16,7604	2,6838	50,318	61,184	19,5	86,6
5-Неодређени	24	57,300	18,9982	3,8780	49,278	65,322	18,0	88,0
Свега	180	56,142	18,1965	1,3563	53,466	58,819	13,0	97,5

На основу средњих вредности успешности у програмирању (табела 4.4.7), закључује се да свих пет група студената имају слична постигнућа, од M=53,374 (дивергентни стил) до M=57,70 (конвергентни стил).

Ниво значајности p је значајно већи од 0,05 (0,300>0,05), што значи да је *задовољена претпоставка о једнакости варијансе у групама*, тј. није прекршена претпоставка о хомогености варијансе. На то указује Левенов тест хомогености варијансе (табела 4.4.8), F(4,175)=1,230.

Табела 4.4.8: Левенов тест хомогености варијансе за групу преферирани стилови учења (Извор: аутор)

Тест хомогености варијација			
Укупно поена – Успех + Тест			
Левенова статистика	**df1**	**df2**	**Sig.**
1,230	4	175	,300

Очигледно је да није прекршена ниједна од претпоставки *ANOVA*, јер је p>0,05 (0,300). Због лакше интерпретације користимо *ANOVA* табелу (табела 4.4.9).

Табела 4.4.9: ANOVA табела за групу преферирани стилови учења (Извор: аутор)

ANOVA					
Укупно поена – Успех + Тест					
	Sum of Squares	**df**	**Mean Square**	**F**	**Sig.**
Различите групе	404,316	4	101,079	,300	,877
Исти субјекти	58865,123	175	336,372		
Свега	59269,439	179			

Резултат једнофакторске *ANOVA* анализе указује да *не постоје статистички значајне разлике у успешности између студената који преферирају различите стилове учења:* F(4,175)=0,300; p>0,05 (p=0,877).

Тиме **подхипотеза Х3б**: *постоје разлике у успешности у учењу програмирања између студената који преферирају различите стилове учења* **није потврђена.**

На основу добијених резултата:

- *подхипотеза* **Х3а:** Постоје разлике у успешности у учењу програмирања између студената који преферирају различите типове програмирања је ***доказана***,
- *подхипотеза* **Х3б:** Постоје разлике у успешности у учењу програмирања између студената који преферирају различите стилове учења ***није потврђена***,

закључује се да је **посебна хипотеза Х3**: *Постоје разлике у успешности у учењу програмирања између студената који преферирају различите типове програмирања и различите стилове учења* **делимично доказана**.

4.5. Модел вештачких неуронских мрежа (нулта хипотеза)

Х0: Могуће је развити модел вештачких неуронских мрежа тако да довољно прецизно предвиђа успешност студената у програмирању узимајући у обзир специфичности учења програмирања и чије су мере евалуације тачности у складу са резултатима валидације у реалним условима.

Да би се прикупили подаци потребни за развој модела вештачке неуронске мреже (ВНМ) за предвиђање успеха ученика у стицању програмерских знања и вештина, анализирано је 180 студената студијског програма Информационе технологије на Факултету техничких наука у Чачку. Карактеристике узорка истраживања и поступак истраживања дати су у трећем поглављу (3.7 и 3.8).

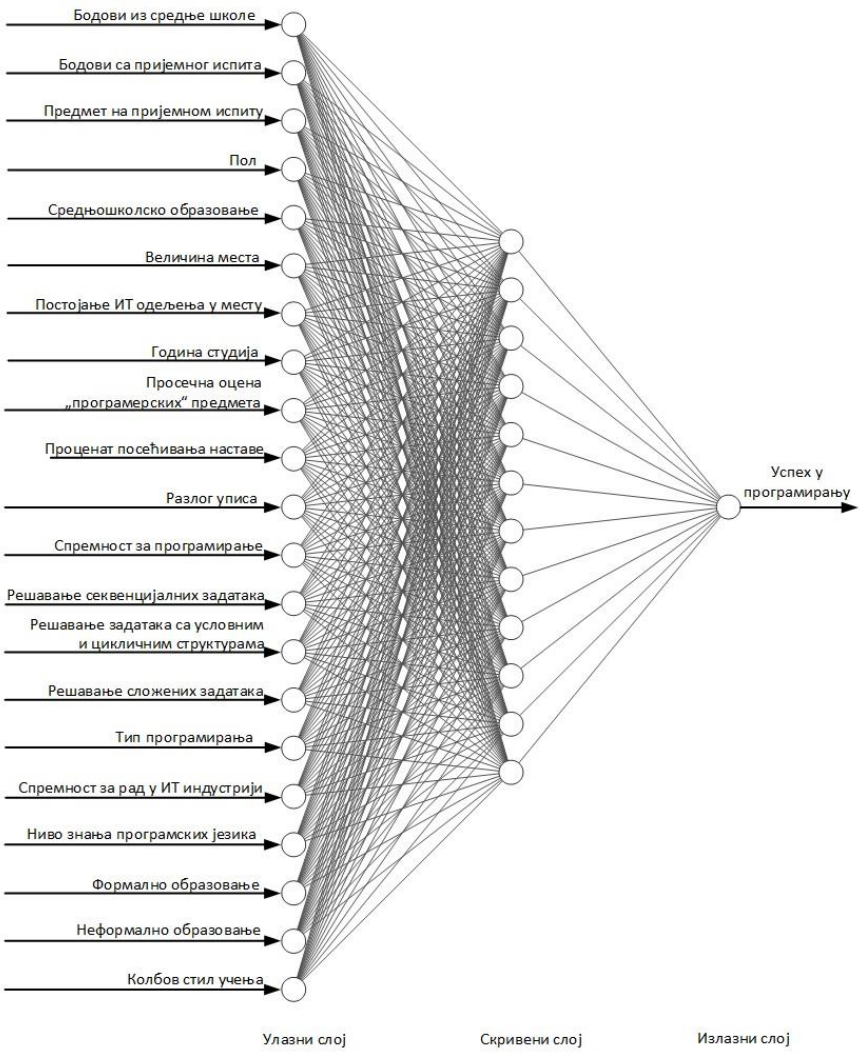
Поред резултата на тесту знања из програмирања, за сваког ученика прикупљен је 21 податак – улазне променљиве (табела 3.8.2). Поступак трансформације реализоваће се бодовима које су студенти постигли на тесту и на основу тих резултата студенти су груписани у три класе (категорије) везано за успех: *неуспешан*, *средње успешан* и *веома успешан*.

4.5.1. Моделовање перформанси студената

За предвиђање успешности у учењу програмирања коришћен је модел **трослојне неуронске мреже базиране на алгоритму учења са простирањем грешке уназад**. Модел мреже састављена је из три слоја: *улазног слоја* (21 улаз), *скривеног слоја* (12 неурона) и *једног излазног слоја* са једним излазом. На слици 4.5.1 [152, 153] дата је архитектура неуронске мреже која је коришћена у овом истраживању.

Процес креирање предиктивног модела студентске успешности реализован је помоћу софтвера за рударење података *WEKA* (поглавље 3.8.5.1).

На слици 4.5.2 дата је изглед дела *ARFF* датотеке података који су коришћени за тренирање и тестирање.



Слика 4.5.1: Архитектура неуронске мреже коришћене у истраживању (Извор: [152, 153])

Treniranje.arff - Notepad

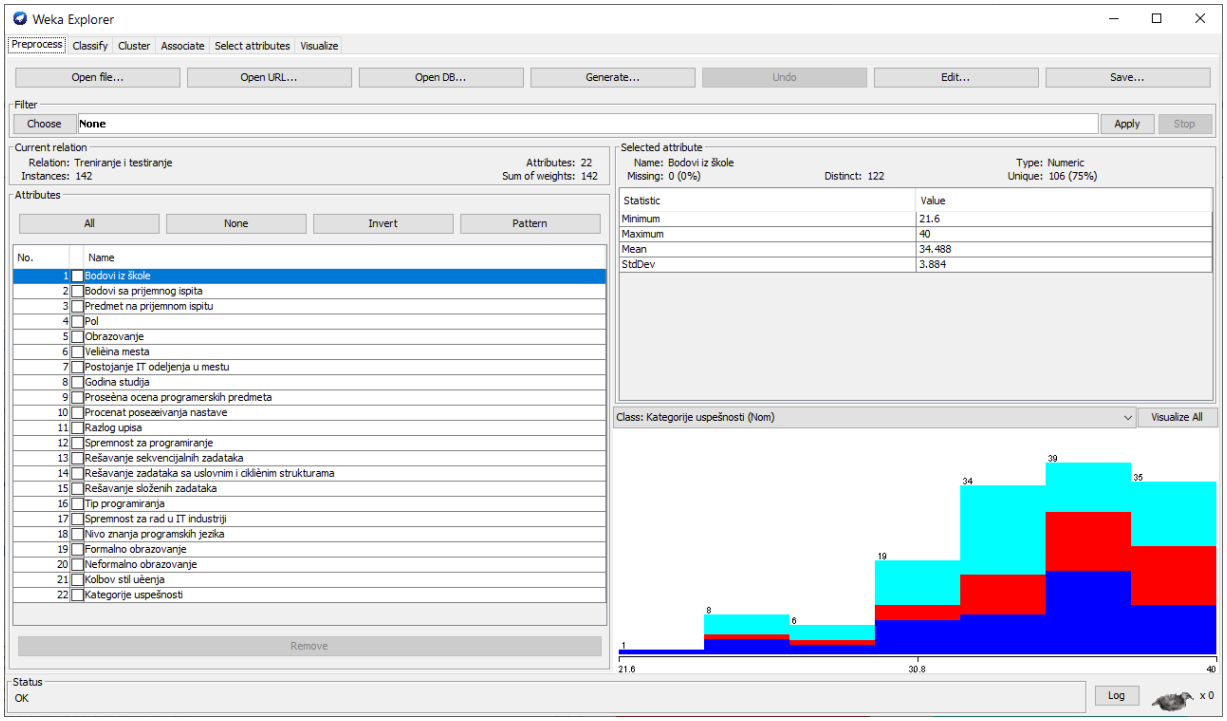
File Edit Format View Help

```
@relation 'Treniranje i testiranje'

@attribute 'Bodovi iz škole' numeric
@attribute 'Bodovi sa prijemnog ispita' numeric
@attribute 'Predmet na prijemnom ispitu' numeric
@attribute 'Pol ' numeric
@attribute 'Obrazovanje ' numeric
@attribute 'Veličina mesta' numeric
@attribute 'Postojanje IT odeljenja u mestu' numeric
@attribute 'Godina studija' numeric
@attribute 'Prosečna ocena programerskih predmeta' numeric
@attribute 'Procenat posećivanja nastave' numeric
@attribute 'Razlog upisa' numeric
@attribute 'Spremnost za programiranje' numeric
@attribute 'Rešavanje sekvencijalnih zadataka' numeric
@attribute 'Rešavanje zadataka sa uslovnim i cikličnim strukturama' numeric
@attribute 'Rešavanje složenih zadataka' numeric
@attribute 'Tip programiranja' numeric
@attribute 'Spremnost za rad u IT industriji' numeric
@attribute 'Nivo znanja programskih jezika' numeric
@attribute 'Formalno obrazovanje' numeric
@attribute 'Neformalno obrazovanje' numeric
@attribute 'Kolbov stil učenja' numeric
@attribute 'Kategorije uspešnosti' {'Srednje uspešan','Veoma uspešan','Neuspešan '}

@data
37.32,48,2,2,12,3,1,2,8,100,2,3,2,4,3,1,1,3.5,4,0,3,'Srednje uspešan'
33.1,40.8,1,1,1,3,1,2,7,100,1,5,4,5,4,1,2,3.5,3,1,3,'Veoma uspešan'
37.28,46,2,1,5,3,1,2,10,100,2,4,3,5,4,1,2,4.5,4,1,3,'Veoma uspešan'
36.52,34.2,1,1,10,3,1,2,7.5,100,3,4,5,5,4,1,3,4.5,2,1,4,'Srednje uspešan'
35.46,44,1,1,10,3,1,2,10,100,2,3,1,3,1,1,2,4.5,2,2,5,'Veoma uspešan'
33.46,44.7,1,1,2,3,1,2,10,100,1,5,5,5,4,1,3,4,5,4,2,2,'Veoma uspešan'
```

Ln 5, Col 49 | 100% | Unix (LF) | ANSI

Слика 4.5.2: ARFF датотека података за тренирање и тестирање (Извор: аутор)

На слици 4.5.3 дат је приказ почетног модела са свим улазним атрибутима у *Weka Explorer*-у.



Слика 4.5.3: Визуализација модела (на левој страни приказани су детаљи назива релација, броја атрибута и броја записа; десна страна даје детаље о вредностима атрибута, типу и броју различитих вредности; спецификација сваког атрибута приказана је у доњем десном углу слике) (Извор: аутор)

4.5.1.1. Први корак – основни модели

Коришћењем стандардног класификатора (алгоритма) ***MultilayerPerceptron*** **са** ***backpropagation*** **алгоритмом** за класификацију, креирана су два предикциона модела који користе различите начине за тестирање података:

- **CV модел** – користи се *10-слојна унакрсна валидација* (*Cross-validation*) података;
- **PP модел** – користи се *процентуална подела* (*Percentage split*) података, за обуку је коришћено 70% података од целокупног подузорка за обуку (99 инстанци) и 30% за тестирање (43 инстанци).

Резултати класификације за ова 2 модела дати су у табелама 4.5.1 – 4.5.6.

Модел CV (*Cross-validation*)

Резултати класификације CV модела за 10-струку унакрсну класификацију показали су да је тачност предвиђања модела 64,79% (табела 4.5.1.)

Табела 4.5.1: Тачност CV модела (10-струка унакрсна класификација) (Извор: аутор)

Тачно класификоване инстанце	92	64,7887%
Нетачно класификоване инстанце	50	35,2113
Капа статистика		0,4616
Средња апсолутна грешка (MAE)		0,2637
Корен средње квадратне грешке (RMSE)		0,4622
Релативна апсолутна грешка (RAE)		60,1863 %
Укупан број инстанци	142	

Мере тачности су представљене за сваку класу посебно (табела 4.5.2.)

Табела 4.5.2: Детаљна тачност CV модела по класама (10-струка унакрсна класификација) (Извор: аутор)

	Стварна позитивна стопа (TP Rate)	Лажна позитивна стопа (FP Rate)	Прецизност (Precision)	Одзив (Recall)	F–мера (F-Measure)	Матхевсов коефиц. корелације MCC	ROC подручје (ROC Area)	PRC подручје (PRC Area)	Класе
	0,500	0,213	0,545	0,500	0,522	0,294	0,626	0,497	Средње успешан
	0,622	0,124	0,639	0,622	0,630	0,502	0,845	0,683	Веома успешан
	0,789	0,200	0,726	0,789	0,756	0,583	0,847	0,746	Неуспешан
Пондерисана (Weighted Avg.)	0,648	0,184	0,642	0,648	0,644	0,464	0,772	0,645	

Матрица конфузије приказује тачно и нетачно класификоване случајеве по класама (табела 4.5.3.)

Табела 4.5.3: Матрица конфузије CV модела (10-струка унакрсна класификација) (Извор: аутор)

a	b	c	Просечан успех
24	11	13	a = Средње успешан
10	23	4	b = Веома успешан
10	2	45	c = Неуспешан

Модел PP (*Percentage split*)

Резултати класификације PP модела који користи 70% података за обуку и 30% за тестирање дати су у табелама 4.5.4 – 4.5.6.

Табела 4.5.4: Тачност PP модела (70% података за обуку, 30% за тестирање) (Извор: аутор)

Тачно класификоване инстанце	20	46,5116%
Нетачно класификоване инстанце	23	53,4884%
Капа статистика		0,1972
Средња апсолутна грешка (MAE)		0,3327
Корен средње квадратне грешке (RMSE)		0,5205
Релативна апсолутна грешка (RAE)		75,6109%
Укупан број инстанци	43	

Табела 4.5.5: Детаљна тачност PP модела по класама (70% података за обуку, 30% за тестирање) (Извор: аутор)

	Стварна позитивна стопа (TP Rate)	Лажна позитивна стопа (FP Rate)	Прецизност (Precision)	Одзив (Recall)	F–мера (F-Measure)	Матхевсов коефиц. корелације MCC	ROC подручје (ROC Area)	PRC подручје (PRC Area)	Класе
	0,222	0,240	0,400	0,222	0,286	-0,021	0,556	0,534	Средње успешан
	0,667	0,206	0,462	0,667	0,545	0,408	0,817	0,691	Веома успешан
	0,625	0,370	0,500	0,625	0,556	0,247	0,738	0,632	Неуспешан
Пондерисана (Weighted Avg.)	0,465	0,281	0,450	0,465	0,440	0,169	0,678	0,603	

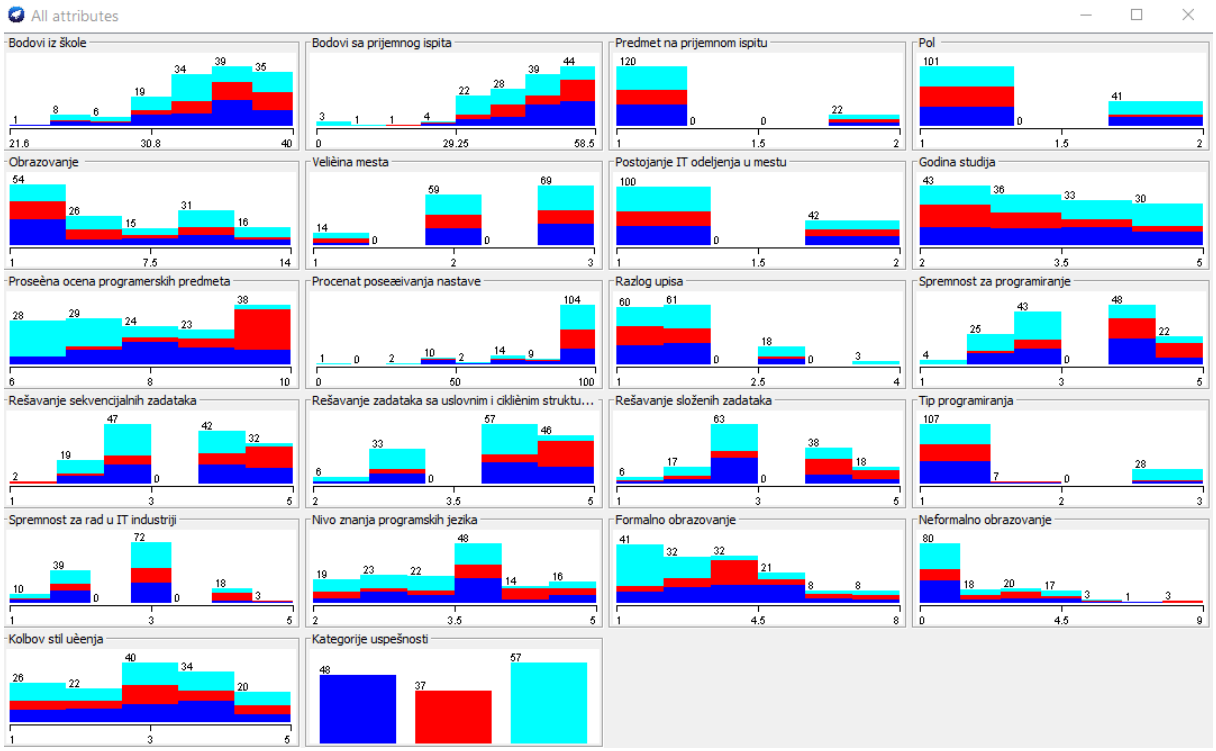
Табела 4.5.6: Матрица конфузије PP модела (70% података за обуку, 30% за тестирање) (Извор: аутор)

a	b	c	Просечан успех
4	5	9	a = Средње успешан
2	6	1	b = Веома успешан
4	2	10	c = Неуспешан

*Анализом приказаних резултата уочава се да су тачности предвиђања **незадовољавајуће**, боља је код CV модела са унакрсном валидацијом (64,79%) у односу на PP модел са % поделом података (46,51%).*

4.5.1.2. Други корак – примена филтера

Због небалансираности инстанци појединих класа улазног скупа, тачност предикција за класе са мањом дистрибуцијом инстанци је мања од тачности предикције бројније класе, што резултује да ће се предикциони модели понашати лоше (мање тачности предвиђања ова два модела: 64,79%, 46,51%). Такође, и визуализација свих података потврђује да подаци нису баш балансирани (слика 4.5.4).



Слика 4.5.4: Визуализација свих података (Извор: аутор)

Због тога је за добијање бољих предиктивних резултата, на постојеће моделе, примењен *Resample* филтер (поновно узорковање) за надгледано учење (*Supervised*). Применом овог филтера добијена су два нова модела CVF (модел унакрсне валидације са филтером) и PPF (модел процентуалне поделе са филтером).

Резултати класификације за ова 2 нова модела дати су у табелама 4.5.7 – 4.5.12.

Модел CVF (*Cross-validation* + филтер *Resample*)

Резултати класификације CVF модела, за 10-струку унакрсну класификацију који користи филтер *Resample*, дати су у табелама 4.5.7 – 4.5.9.

Табела 4.5.7: Тачност CVF модела (10-струка унакрсна класификација + филтер Resample) (Извор: аутор)

Тачно класификоване инстанце	118	83,0986%
Нетачно класификоване инстанце	24	16,9014
Капа статистика		0,742
Средња апсолутна грешка (MAE)		0,1154
Корен средње квадратне грешке (RMSE)		0,2982
Релативна апсолутна грешка (RAE)		26,3326 %
Укупан број инстанци	142	

Табела 4.5.8: Детаљна тачност CVF модела по класама (10-струка унакрсна класификација + филтер Resample) (Извор: аутор)

	Стварна позитивна стопа (TP Rate)	Лажна позитивна стопа (FP Rate)	Прецизност (Precision)	Одзив (Recall)	F–мера (F-Measure)	Матхевсов коефиц. корелације MCC	ROC подручје (ROC Area)	PRC подручје (PRC Area)	Класе
	0,708	0,085	0,810	0,708	0,756	0,646	0,865	0,780	Средње успешан
	0,865	0,057	0,842	0,865	0,853	0,801	0,936	0,903	Веома успешан
	0,912	0,118	0,839	0,912	0,874	0,785	0,935	0,868	Неуспешан
Пондерисана (Weighted Avg.)	0,831	0,091	0,830	0,831	0,829	0,742	0,911	0,874	

Табела 4.5.9: Матрица конфузије CVF модела (10-струка унакрсна класификација + филтер Resample) (Извор: аутор)

a	b	c	Просечан успех
34	5	9	a = Средње успешан
4	32	1	b = Веома успешан
4	1	52	c = Неуспешан

Модел PPF (*Percentage split* + филтер *Resample*)

Резултати класификације PPF модела, који користи 70% података за обуку и 30% за тестирање и филтер *Resample*, дати су у табелама 4.5.10 – 4.5.12.

Табела 4.5.10: Тачност PPF модела (70% података за обуку, 30% за тестирање + филтер Resample) (Извор: аутор)

Тачно класификоване инстанце	34	79,0698%
Нетачно класификоване инстанце	9	20,9302%
Капа статистика		0,6783
Средња апсолутна грешка (MAE)		0,153
Корен средње квадратне грешке (RMSE)		0,3673
Релативна апсолутна грешка (RAE)		34,8981%
Укупан број инстанци	43	

Табела 4.5.11: Детаљна тачност PPF модела по класама (70% података за обуку, 30% за тестирање + филтер Resample) (Извор: аутор)

	Стварна позитивна стопа (TP Rate)	Лажна позитивна стопа (FP Rate)	Прецизност (Precision)	Одзив (Recall)	F–мера (F-Measure)	Матхевсов коефиц. Корелације MCC	ROC подручје (ROC Area)	PRC подручје (PRC Area)	Класе
	0,583	0,129	0,636	0,583	0,609	0,467	0,637	0,485	Средње успешан
	1,000	0,094	0,786	1,000	0,880	0,884	0,938	0,730	Веома успешан
	0,800	0,087	0,889	0,800	0,842	0,721	0,861	0,786	Неуспешан
Пондерисана (Weighted Avg.)	0,791	0,100	0,792	0,791	0,787	0,682	0,818	0,688	

Табела 4.5.12: Матрица конфузије PPF модела (70% података за обуку, 30% за тестирање + филтер Resample) (Извор: аутор)

a	b	c	Просечан успех
7	3	2	a = Средње успешан
0	11	0	b = Веома успешан
4	0	16	c = Неуспешан

*Применом филтера Resample дошло је до **повећања тачности предикције** (са 64,79% на 83,10%, односно са 46,51% на 79,07%), тако да је **коришћење** овог **филтера препорука** за анализе у будућности.*

4.5.1.3. Трећи корак – селекција подскупова атрибута

У циљу оптимизације модела неуронске мреже и повећања његове тачности извршена је евалуација атрибута. Поред основног скупа који има све улазне атрибуте (21), анализа је извршена за још 9 подскупова. Нису разматрани подскупови који имају мање од 7 атрибута. У табели 4.5.13 приказани су подскупови атрибута анализираног улазног скупа који су добијени применом одређених комбинација евалуатора и метода претраживања. Значи, осим два основна модела CVF и PPF, креирано је још 18 модела CVF1-CVF9, односно PPF1-PPF9. Модел са најбољом предиктивном тачношћу биће искоришћен као коначан модел за имплементацију.

Табела 4.5.13: Подскупови битних атрибута (Извор: аутор)

Подскуп (Модел)	**Евалу-атор**	**Метод претражи-вања**	**Бр. ат.**	**Селектовани/избачени атрибути**
CVF, PPF	/	/	21	Користе се сви атрибути
SS1 (CVF1, PPF1)	Ranker	Corelation AttributeEval (>0.1)	17	Просечна оцена програмерских предмета, Формално образовање, Решавање задатака са услovним и цикличним структурама, Спремност за програмирање, Тип програмирања, Бодови са пријемног испита, Решавање секвенцијалних задатака, Решавање сложених задатака, Пол, Неформално образовање, Образовање, Разлог уписа, Година студија, Спремност за рад у IT индустрији, Ниво знања програмских језика, Просечан посећивања наставе, Бодови из школе *Избачени атрибути: Величина места, Постојање IT одељења у месту, Предмет на пријемном испиту, Колбов стил учења*
SS2 (CVF2, PPF2)	Ranker	Corelation AttributeEval (>0.2)	7	Просечна оцена програмерских предмета, Формално образовање, Решавање задатака са услovним и цикличним структурама, Спремност за програмирање, Тип програмирања, Бодови са пријемног испита, Решавање секвенцијалних задатака *Избачени атрибути: Решавање сложених задатака, Пол, Неформално образовање, Образовање, Разлог уписа, Година студија, Спремност за рад у IT индустрији, Ниво знања програмских језика, Просечан посећивања наставе, Бодови из школе, Величина места, Постојање IT одељења у месту, Предмет на пријемном испиту, Колбов стил учења*
SS3 (CVF3, PPF3)	Ranker	GainRation AttributeEval (>0)	9	Просечна оцена програмерских предмета, Решавање задатака са услovним и цикличним структурама, Решавање секвенцијалних задатака, Формално образовање, Неформално образовање, Спремност за програмирање, Решавање сложених задатака, Тип програмирања, Пол *Избачени атрибути: Колбов стил учења, Величина места, Предмет на пријемном испиту, Бодови са пријемног испита, Образовање, Разлог уписа, Постојање IT одељења у месту, Година студија, Ниво знања програмских језика, Спремност за рад у IT индустрији, Просечан посећивања наставе, Бодови из школе*
SS4 (CVF4, PPF4)	Ranker	InfoGain AttributeEval (>0.1)	7	Просечна оцена програмерских предмета, Решавање задатака са услovним и цикличним структурама, Решавање секвенцијалних задатака, Формално образовање, Неформално образовање, Спремност за програмирање, Решавање сложених задатака *Избачени атрибути: Тип програмирања, Пол, Колбов стил учења, Величина места, Предмет на пријемном испиту, Бодови са пријемног испита, Образовање, Разлог уписа, Постојање IT одељења у месту, Година студија, Ниво знања програмских језика, Спремност за рад у IT индустрији, Просечан посећивања наставе, Бодови из школе*
SS5 (CVF5, PPF5)	Ranker	OneRAttribute Eval (преко 40)	13	Просечна оцена програмерских предмета, Решавање задатака са услovним и цикличним структурама, Спремност за програмирање, Решавање секвенцијалних задатака, Формално образовање, Образовање, Просечан посећивања наставе, Бодови са пријемног испита, Тип програмирања, Решавање сложених задатака, Ниво знања програмских језика, Спремност за рад у IT индустрији, Величина места *Избачени атрибути: Бодови из школе, Предмет на пријемном испиту, Постојање IT одељења у месту, Неформално образовање, Колбов стил учења, Пол, Година студија, Разлог уписа*
SS6 (CVF6, PPF6)	Ranker	OneRAttribute Eval (преко 45)	8	Просечна оцена програмерских предмета, Решавање задатака са услovним и цикличним структурама, Спремност за програмирање, Решавање секвенцијалних задатака, Формално образовање, Образовање, Просечан посећивања наставе, Бодови са пријемног испита *Избачени атрибути: Решавање сложених задатака, Тип програмирања, Ниво знања програмских језика, Спремност за рад у IT индустрији, Величина места, Бодови из школе, Предмет на пријемном испиту, Постојање IT одељења у месту, Неформално образовање, Колбов стил учења, Пол, Година студија, Разлог уписа*
SS7 (CVF7, PPF7)	Ranker	RelefF AttributeEval (>0)	16	Просечна оцена програмерских предмета, Пол, Решавање задатака са услovним и цикличним структурама, Формално образовање, Година студија, Решавање секвенцијалних задатака, Тип програмирања, Спремност за програмирање, Ниво знања програмских језика, Бодови са пријемног испита, Просечан посећивања наставе, Решавање сложених задатака, Бодови из школе, Образовање, Спремност за рад у IT индустрији, Неформално образовање *Избачени атрибути: Разлог уписа, Величина места, Постојање IT одељења у месту, Предмет на пријемном испиту, Колбов стил учења*

Подскуп (Модел)	Евалуатор	Метод претраживања	Бр. ат.	Селектовани/избачени атрибути
SS8 (CVF8, PPF8)	Ranker	RelefF AttributeEval (>0.02)	8	Prosečna ocena programerskih predmeta, Pol, Rešavanje zadataka sa uslovnim i cikličnim strukturama, Formalno obrazovanje, Godina studija, Rešavanje sekvencijalnih zadataka, Tip programiranja, Spremnost za programiranje *Избачени атрибути: Nivo znanja programskih jezika, Bodovi sa prijemnog ispita, Procenat posećivanja nastave, Rešavanje složenih zadataka, Bodovi iz škole, Obrazovanje, Spremnost za rad u IT industriji, Neformalno obrazovanje, Razlog upisa, Veličina mesta, Postojanje IT odeljenja u mestu, Predmet na prijemnom ispitu, Kolbov stil učenja*
SS9 (CVF9, PPF9)	Ranker	Symmetrical UncertAttributeEval (>0)	9	Prosečna ocena programerskih predmeta, Rešavanje zadataka sa uslovnim i cikličnim strukturama, Formalno obrazovanje, Rešavanje sekvencijalnih zadataka, Neformalno obrazovanje, Spremnost za programiranje, Rešavanje složenih zadataka, Tip programiranja, Pol *Избачени атрибути: Kolbov stil učenja, Veličina mesta, Predmet na prijemnom ispitu, Bodovi sa prijemnog ispita, Obrazovanje, Razlog upisa, Postojanje IT odeljenja u mestu, Godina studija, Nivo znanja programskih jezika, Spremnost za rad u IT industriji, Procenat posećivanja nastave, Bodovi iz škole*

Резултати класификације и избор модела са најбољом предиктивном тачношћу приказан је у наредном поглављу.

4.5.1.4. Резултати класификације и избор оптималног модела

Након избора битних (релевантних) подскупова, применом модела трослојне неуронске мреже базиране на алгоритму учења са простирањем грешке уназад и филтера *Resample*, добијене су тачности креираних модела за селектоване подскупове (табела 4.5.14).

Табела 4.5.14: Тачности предиктивних модела креираних над селектованим подскуповима атрибута студената (Извор: аутор)

Подскуп	Модел	Предиктивна тачност	Верификација	
			Модел	Тачност
Основни	CVF	83,10%	M0_CV	71,05%
	PPF	79,07%	M0_PP	71,05%
SS1	CVF1	**85,92%**	M1_CV	**78,95%**
	PPF1	81,40%	M1_PP	**78,59%**
SS2	CVF2	83,10%	M2_CV	65,79%
	PPF2	**90,70%**	M2_PP	**65,79%**
SS3	CVF3	85,92%	M3_CV	60,53%
	PPF3	86,05%	M3_PP	60,53%
SS4	CVF4	80,99%	M4_CV	60,53%
	PPF4	76,74%	M4_PP	60,53%
SS5	CVF5	81,69%	M5_CV	65,79%
	PPF5	76,74%	M5_PP	65,79%
SS6	CVF6	85,21%	M6_CV	57,89%
	PPF6	76,74%	M6_PP	57,89%
SS7	CVF7	84,51%	M7_CV	68,42%
	PPF7	81,40%	M7_PP	68,42%
SS8	CVF8	81,69%	M8_CV	50,00%
	PPF8	81,40%	M8_PP	50,00%
SS9	CVF9	85,92%	M9_CV	60,53%
	PPF9	86,05%	M9_PP	60,53%

Из табеле 4.5.14 може се видети, да је најбоља предиктивна тачност од **90,70%** (тачност верификације 65,79%) постигнута за модел **PPF2**, док је најбоља тачност верификације од *78,95%* (предиктивна тачност 85,92%) постигнута за модел *CVF1*.

Модел PPF2 са најбољом предиктивном тачношћу је модел: *процентуалне поделе* (*Percentage split модел) са Resample филтером и подскупом са 7 улазних атрибута (података)*: Бодови са пријемног испита, Просечна оцена програмерских предмета, Спремност за програмирање, Решавање секвенцијалних задатака, Решавање задатака са условним и цикличним структурама, Тип програмирања, Формално образовање.

Детаљни резултати класификације модела PPF2, дати су у табелама 4.5.15 – 4.5.18.

Табела 4.5.15: Тачност оптимизованог PPF2 модела (Извор: аутор)

Тачно класификоване инстанце	34	90,6977%
Нетачно класификоване инстанце	9	9,3023%
Капа статистика		0,855
Средња апсолутна грешка (MAE)		0,1362
Корен средње квадратне грешке (RMSE)		0,2636
Релативна апсолутна грешка (RAE)		31,0511%
Укупан број инстанци	43	

Табела 4.5.16: Детаљна тачност оптимизованог PPF2 модела (Извор: аутор)

	Стварна позитивна стопа (TP Rate)	Лажна позитивна стопа (FP Rate)	Прецизност (Precision)	Одзив (Recall)	F–мера (F-Measure)	Матхевсов коефиц. корелације MCC	ROC подручје (ROC Area)	PRC подручје (PRC Area)	Класе
	0,750	0,032	0,900	0,750	0,818	0,762	0,898	0,873	Средње успешан
	1,000	0,063	0,846	1,000	0,917	0,891	0,974	0,889	Веома успешан
	0,950	0,043	0,950	0,950	0,950	0,907	0,963	0,946	Неуспешан
Пондерисана (Weighted Avg.)	0,907	0,045	0,909	0,907	0,905	0,862	0,948	0,911	

Табела 4.5.17: Матрица конфузије оптимизованог PPF2 модела (Извор: аутор)

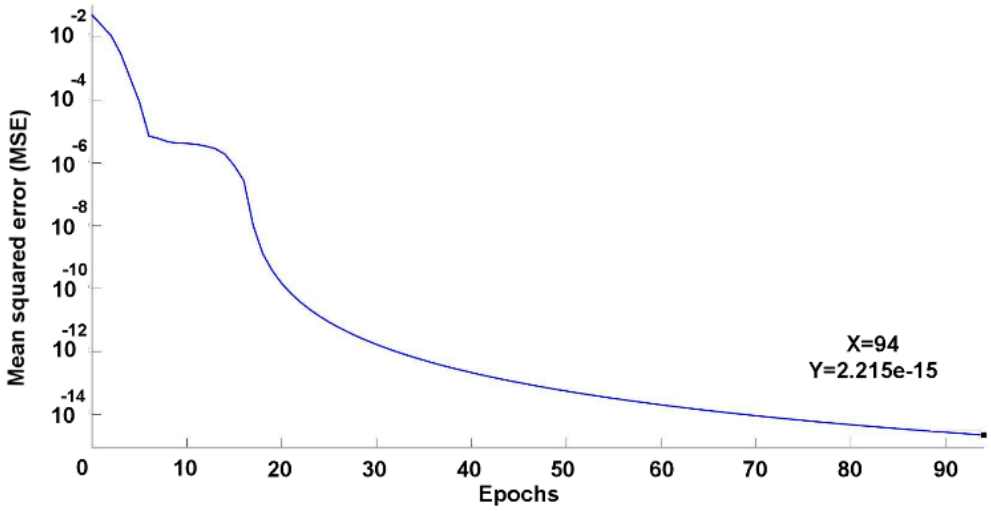
a	b	c	Просечан успех
9	2	1	a = Средње успешан
0	11	0	b = Веома успешан
1	0	19	c = Неуспешан

Табела 4.5.18: Предвиђања (исходи) за сваку појединачну класу оптимизованог PPF2 модела (Извор: аутор)

	Исходи			
Класа	TP	TN	FP	FN
a = Средње успешан	9	30	1	3
b = Веома успешан	11	22	2	0
c = Неуспешан	19	22	1	1

Детаљни резултати класификације модела, који су дати у табелама 4.5.15 – 4.5.18, потврђују да се ***креирани оптимизовани модел PPF2 са великом предиктивном тачношћу може искористити за предвиђање успешности у области програмирања****.*

За оптимизовани модел PPF2 на слици 4.5.5 [152] приказана је крива конвергенције, за средњу квадратну грешку (MSE). На овој слици вертикална оса одговара средњој квадратној грешци, а хоризонтална епохама потребним за завршетак тренинга. Насумично генерисани почетни пондери и прагови утичу на збирну квадратну грешку између померања из нумеричке симулације и оних из *backpropagation* мрежа.



Слика 4.5.5: Крива конвергенције неуронске мреже у процесу учења (Извор: [152])

Значи, као коначан модел за имплементацију биће искоришћен PPF2 модел.

Закључак – доказана је нулта хипотеза:
Могуће је развити модел вештачких неуронских мрежа тако да довољно прецизно предвиђа успешност студената у програмирању узимајући у обзир специфичности учења програмирања и чије су мере евалуације тачности у складу са резултатима валидације у реалним условима.

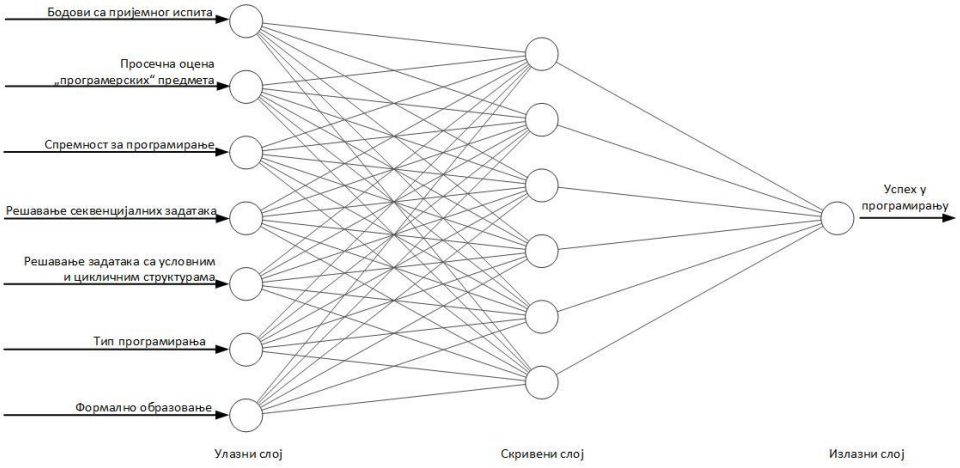
Као што се види из добијених резултата, тачност је 90,70%. Поређење резултата (потпоглавље 4.7) са сродним истраживањима показује да модели који се користе у овом пољу имају задовољавајућу тачност.

Додатак:
*Претходно креирани модели су развијени за 3 категорије успешности: Веома успешан, Средње успешан и Неуспешан. Када се модел PPF2 модификује и развије за две категорије успешности: постојећа категорија **Неуспешан** и нова категорија **Успешан** (укључује постојеће категорије Средње успешан и Веома успешан), добијају се следеће тачности: предиктивна тачност од **92,96%** и тачност верификације од 89,47%.*

4.6. Имплементација предиктивног модела – развој веб апликације

С обзиром да су циљна група корисника наставници који се баве програмирањем очекивано је да поседују напредна знања и вештине из информационих технологија. Међутим, с обзиром да апликација укључује вештачке неуронске мреже неопходно је обезбедити интерфејс за кориснике који не познају поменуту технику машинског учења. веб базирана апликација креирана је коришћењем *Microsoft SQL Server*-а *2017* и *Microsoft Visual Studio 2017*.

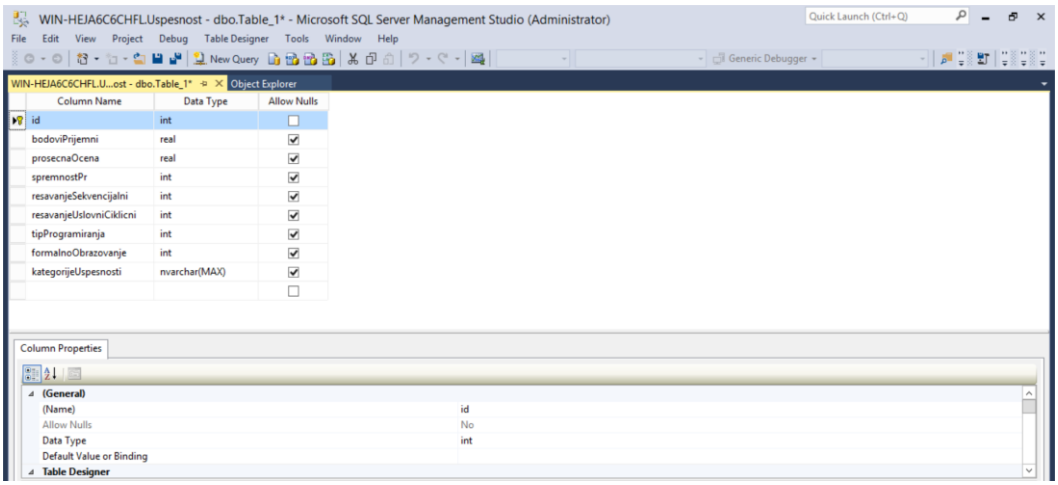
За креирање веб апликације коришћен је оптимизован модел вештачке неуронске мреже PPF2, приказан на слици 4.6.1.



Слика 4.6.1: Архитектура модела вештачке неуронске мреже PPF2 (Извор: аутор)

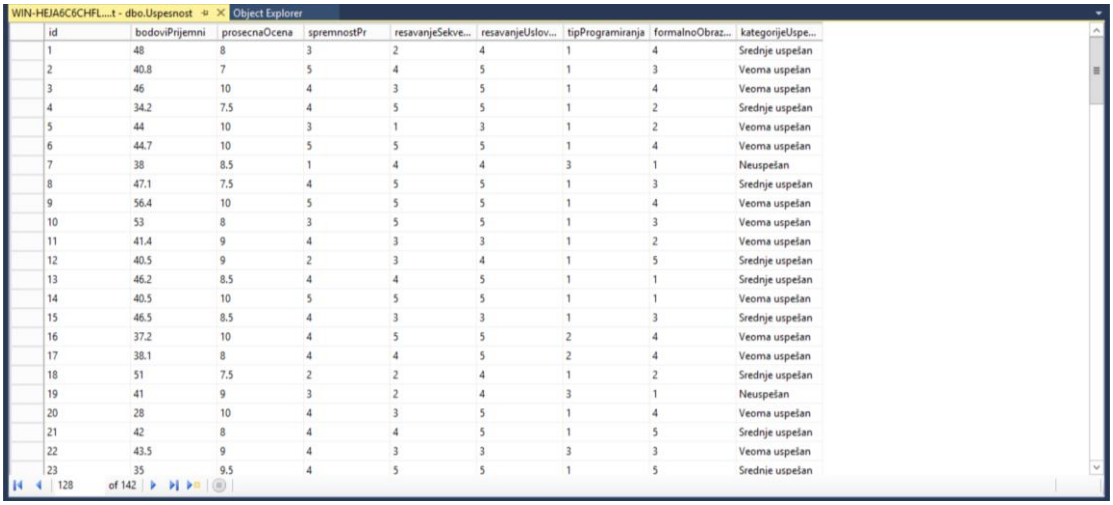
Први корак подразумева креирање базе података и увоз података који су прикупљени током почетне фазе истраживања.

На слици 4.6.2 приказани су називи и типови атрибута: *id* (тип int), *бодови постигнути на пријемном испиту* (bodoviPrijemni, тип real), *просечна оцена из предмета који се односе на програмирање* (prosecnaOcena, тип real), *спремност за програмирање* (spremnostPr, тип int), *решавање секвенцијалних задатака* (resavanjeSekvencijalni, тип int), *решавање задатака са условним и цикличним структурама* (resavanjeUslovniCiklicni, тип int), *тип програмирања* (tipProgramiranja, тип int), *формално образовање* (formalnoObrazovanje, тип int), *категорије успешности* (kategorijeUspesnosti, тип nvarchar (max)).



Слика 4.6.2: Приказ атрибута и њихових типова у бази података (Извор: аутор)

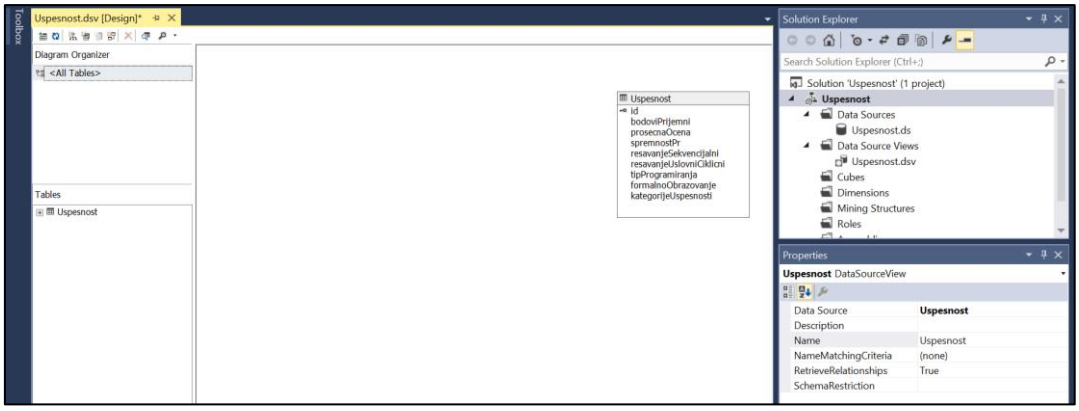
Након креирања базе података и увоза података у базу добија се приказ као на слици 4.6.3.



id	bodoviPrijemni	prosecnaOcena	spremnostPr	resavanjeSekve...	resavanjeUslov...	tipProgramiranja	formalnoObraz...	kategorijeUspe...
1	48	8	3	2	4	1	4	Srednje uspešan
2	40.8	7	5	4	5	1	3	Veoma uspešan
3	46	10	4	3	5	1	4	Veoma uspešan
4	34.2	7.5	4	5	5	1	2	Srednje uspešan
5	44	10	3	1	3	1	2	Veoma uspešan
6	44.7	10	5	5	5	1	4	Veoma uspešan
7	38	8.5	1	4	4	3	1	Neuspešan
8	47.1	7.5	4	5	5	1	3	Srednje uspešan
9	56.4	10	5	5	5	1	4	Veoma uspešan
10	53	8	3	5	5	1	3	Veoma uspešan
11	41.4	9	4	3	3	1	2	Veoma uspešan
12	40.5	9	2	3	4	1	5	Srednje uspešan
13	46.2	8.5	4	4	5	1	1	Srednje uspešan
14	40.5	10	5	5	5	1	1	Veoma uspešan
15	46.5	8.5	4	3	3	1	3	Srednje uspešan
16	37.2	10	4	5	5	2	4	Veoma uspešan
17	38.1	8	4	4	5	2	4	Veoma uspešan
18	51	7.5	2	2	4	1	2	Srednje uspešan
19	41	9	3	2	4	3	1	Neuspešan
20	28	10	4	3	5	1	4	Veoma uspešan
21	42	8	4	4	5	1	5	Srednje uspešan
22	43.5	9	4	3	3	3	3	Veoma uspešan
23	35	9.5	4	5	5	1	5	Srednje uspešan

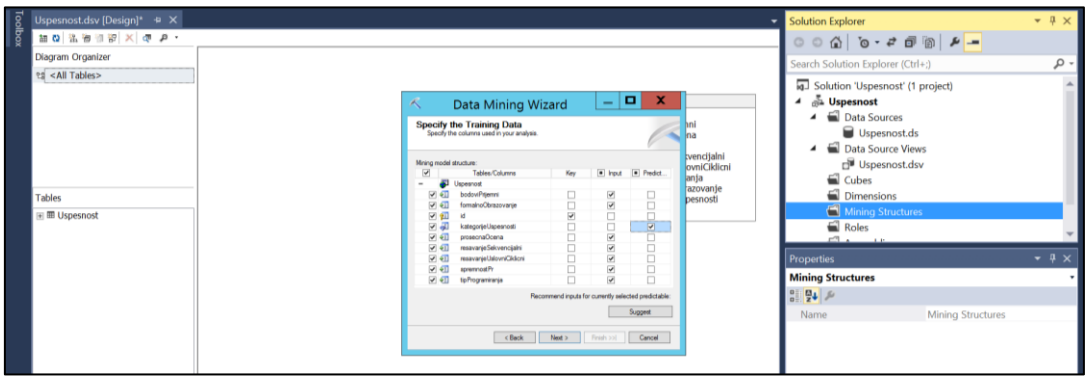
Слика 4.6.3: Приказ импортованих података у бази података (Извор: аутор)

Осим креирања базе података у окружењу *Microsoft Visual Studio 2017* прави се веза са базом података Успешност (*Data Source*), као што је приказано на слици 4.6.4, и креира се вештачка неуронска мрежа (слика 4.6.5).



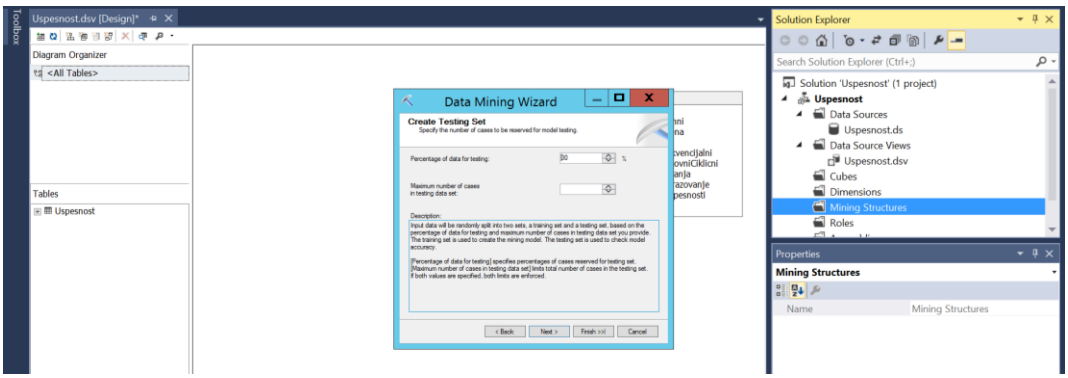
Слика 4.6.4: Приказ табеле Успешност из окружења Microsoft Visual Studio (Извор: аутор)

Креирањем модела неуронских мрежа бирају се улази, излази и кључни атрибут (слика 4.6.5).



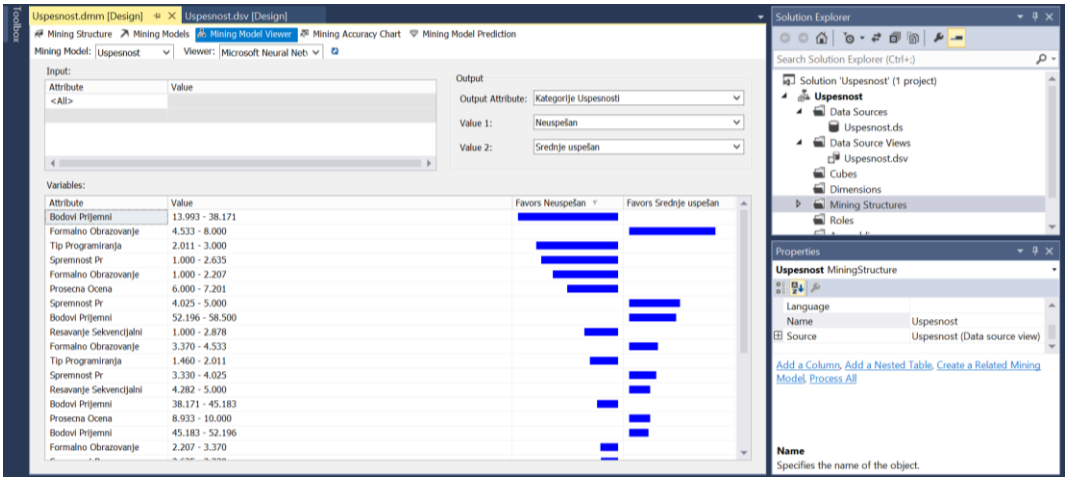
Слика 4.6.5: Избор улазних и излазних атрибута у вештачкој неуронској мрежи (Извор: аутор)

Након одабира улазних и излазних параметара може се подесити проценат података који се користи за тестирање. Тај проценат је иницијално подешен на 30 и та вредност није мењана (слика 4.6.6).



Слика 4.6.6: Избор процента података који ће се користити за тестирање (Извор: аутор)

Након развоја овог решења у *Mining Model Viewer* (слика 4.6.7) може се визуелно прегледати категорија успешности у односу на вредност улазних атрибута.



Слика 4.6.7: Преглед резултата у оквиру Microsoft Visual Studio (Извор: аутор)

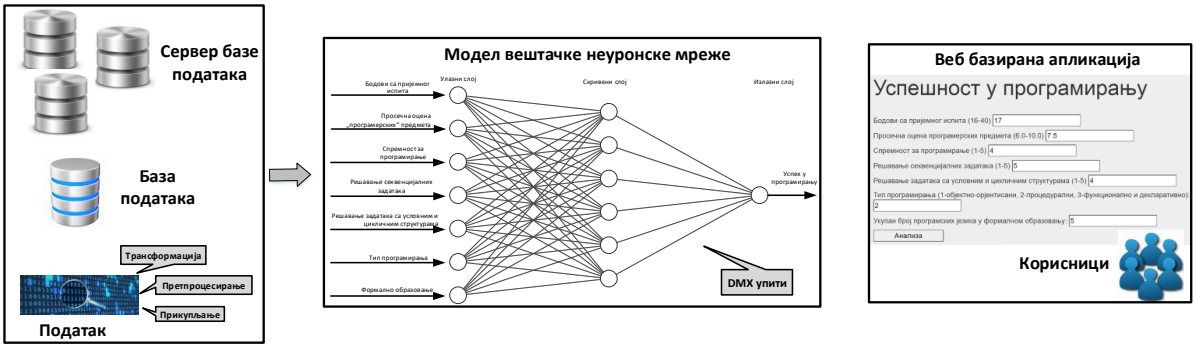
У циљу олакшања коришћења модела неуронских мрежа креирана је веб базирана апликација коришћењем програмских језика *C#* и *.NET* технологије. Оптимизовани модел неуронских мрежа имплементиран је тако што је у оквиру саме веб апликације уграђен одговарајући *DMX (Data Mining Extensions)* упит који се користи да би се за унете улазне вредности добио излаз.

DMX упит који је коришћен у овом случају дат је на слици 4.6.8.

```
"SELECT[Uspesnost].[kategorijeUspesnosti]," +
                    "Predict([Uspesnost].[kategorijeUspesnosti])" +
                    "From[Uspesnost] NATURAL PREDICTION JOIN" +
                    "(SELECT @bodovi AS[bodoviPrijemni]," +
                     "@ocena AS[prosecnaOcena]," +
                     "@spremnost AS[spremnostPr]," +
                     "@sekvencijalni AS[resavanjeSekvencijalni]," +
                     "@uslovni AS[resavanjeUslovniCiklicni]," +
                     "@tip AS[tipProgramiranja]," +
                      "@formalno AS[formalnoObrazovanje]) AS t";
```

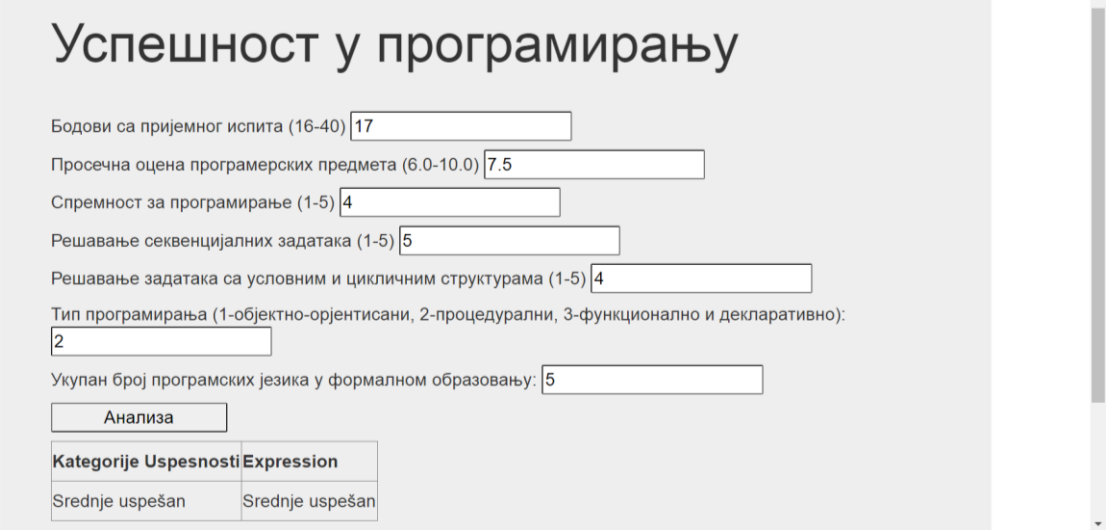
Слика 4.6.8: DMX упит – за унете улазне вредности добија се излаз (Извор: аутор)

Архитектура апликације приказана је на слици 4.6.9 [152].



Слика 4.6.9: Архитектура предложеног система (Извор: [152])

Приказ графичког корисничког интерфејса дат је на слици 4.6.10 [152]. Као случај коришћења одабрани су следећи параметри: *бодови са пријемног испита (17), просечна оцена програмерских предмета (7,5), спремност за програмирање (4), решавање секвенцијалних задатака (5), решавање задатака са условном и цикличном структуром (4), тип програмирања (процедурални – 2), укупан број програмских језика у формалном образовању (5)*.



Слика 4.6.10: Приказ графичког интерфејса апликације (Извор: [152])

4.7. Поређење резултата сродних истраживања са спроведеним истраживањем

Поређењем сродних истраживања са спроведеним истраживањем у оквиру докторске дисертације уочене су сличности и разлике, као и допринос предложеног приступа. Сличност се огледа у примењеној техници за процес предикције у области образовања. Аутор је користио технику вештачких неуронских мрежа како би имао ефикаснији образовни процес и како би оптималније планирао наставу.

У односу на приказана сродна истраживања предложено истраживање аутора дисертације се разликује у приступу и методологији. У раду [103] доказана је боља успешност вештачких неуронских мрежа у односу на друге технике машинског учења за предикцију у области образовања. Развијен је оригиналан инструмент за мерење постигнућа из области програмирања и тај резултат је коришћен као улаз у неуронску мрежу. Уз поменути улаз коришћени су и други релевантни параметри. Оптимизација модела је корак који је допринео већој прецизности модела и представља још један

прилог развоју предикција у области образовања, тачније у настави програмирања. Развијени модел је уз модификације инструмента за мерење постигнућа, могуће применити и на друге предмете, што моделу даје особину универзалног решења. Посебан значај се огледа у могућности коришћења модела од стране особа које не познају технике машинског учења, а уз помоћ развијене веб апликације могу адекватно применити модел неуронских мрежа за своје улазне податке.

У погледу тачности предложено истраживање даје задовољавајуће резултате. Најбољи избор оптимизованих параметара достиже тачност од 90,7% што је задовољавајуће за овакве проблеме предикције. Истраживање приказано у [125] даје тачност од 82% и 71%, за предвиђање група студената високог и ниског постигнућа, респективно. Тачност од 95,8% постигнута је у истраживању [113]. Оваква тачност истраживања постигнута је уз узорак од преко 5000 студената, што је неупоредиво више од узорка у предложеном истраживању. У истраживању приказаном у [111] такође су коришћене три групе успешности, са тачностима од 86,6%, 94,2% и 85,7%, за лошије, средње и добре резултате успешности студената. Такође, у раду [104] добијена је тачност за три групе успешности и то 82% за добре резултате, 53% за просечне резултате и 88% за лоше резултате. Вештачке неуронске мреже у раду [112] достижу тачност од 82,6% и 76,1% у предикцији студентских оцена. У раду [122] достигнута је тачност од 84% са дубоким учењем и неуронским мрежама, а уз то су упоредили и друге технике машинског учења уз закључак да се најбољи резултат постиже коришћењем неуронских мрежа.

Имајући у виду преглед тачности из приказаних сродних истраживања предложено истраживање даје задовољавајућу тачност и може послужити за предикцију категорија успешности студената из области програмирања.

5. ЗАКЉУЧАК

Резиме

Данас се све више пажње посвећује интелектуалним ресурсима. Људски капитал у развијеним земљама чини око 80% националног богатства и важан је фактор у развоју државе у целини. За управљање системом образовања неопходно је поседовати вештину предвиђања (првенствено успеха студената) и препознати доследност у предстојећим догађајима и процесима. Рано предвиђање успеха студената може да помогне универзитетима да предузму правовремене акције, попут планирања одговарајуће обуке за побољшање стопе њиховог успеха. У савременом друштву, ослањајући се на информационе технологије, образовање о програмирању је изузетно важно. Последњих година потражња за програмерима и интересовање студената за програмирање брзо су порасли, уводни курсеви програмирања су све популарнији. Али, уводни курсеви програмирања на многим универзитетима обично укључују студенте са различитим програмерским предзнањима. Неки студенти долазе са дугогодишњим искуством у програмирању са курсева које су похађали у средњој школи, док други могу да раде само основне рачунарске задатке као што је обрада текста, што често ствара диспаритет у раду са студентима. Због тога постоји потреба за ефикасним анализама велике количине података на основу којих се могу предвидети фактори који утичу на успешност студената у области програмирања.

Истраживање које је реализовано у докторској дисертацији „*Фактори учења и предвиђање успешности у програмирању применом вештачких неуронских мрежа*“ обухвата могућности примене вештачких неуронских мрежа у сврху анализа података који се о студентима генеришу у току средњошколског образовања, као и у образовању ван образовних институција.

У овој дисертацији дат је преглед истраживања у области примене техника машинског учења у области предвиђања успешности, приказан је развој инструмента за мерење успешности, као и методологија развоја оптимизованог модела неуронских мрежа за предвиђање успешности студената у области програмирања, и описан начин креирања веб базиране апликације за коришћење модела неуронских мрежа од стране наставника. Имајући у виду добијене резултате као и поређење са резултатима сродних истраживања, утврђено је да предложени модел даје задовољавајућу тачност и да успешно може предвидети успех студената на основу одабраних улазних параметара. На основу успешности коју модел предвиђа наставник добија могућност прилагођавања наставе према добијеним резултатима, ефикасније организације наставе као и бољег крајњег резултата – успешно савладано градиво у области програмирања, што је кључна вештина у области информационих технологија.

Допринос дисертације

Научни доприноси докторске дисертације огледају се у:

- **Креирању оригиналног модела неуронских мрежа** за предвиђање успешности у области програмирања. Систематизовани преглед и анализа репрезентативних референци у контексту предвиђања успешности учења потврђује оригиналност истраживања и модела, и дат је у поглављу *Теоријски оквир*. Опис метода и неуронске мреже дат је у делу *Методологија истраживања*, док је у поглављу *Резултати и дискусија* дат избор оптималног модела и развој веб базиране апликације за рад са моделом вештачких неуронских мрежа.

- **Развоју процедура за прикупљање података** о факторима учења програмирања. Већину инструмената за прикупљање података развио је кандидат: а) за прикупљање примарних података: анкентни упитници, тестови знања из области програмирања, Колбов инвентар стилова учења, скале самопроцене; б) за прикупљање секундарних података: подаци о успешности студената из интерне евиденције у високошколској установи и екстерни подаци о програмима средњошколских предмета из ИТ области (јавно доступни подаци МПНТР).
- **Применљивости креираног решења** на остале стандардизоване области информационих технологија. Предложени приступ је тестиран и верификован, што даје могућност примене и у осталим стандардизованим областима информационих технологија.

Правци даљег истраживања

Даља истраживања у овој области односе се на увођење тзв. дубоког учења у области предикције успешности. Дубинско учење и обрада података у образовању стекли су значајну пажњу у протеклим годинама.

Дубоко учење (*Deep learning*) је подскуп метода машинског учења засновано на вишеслојним (до 150 слојева) вештачким неуронским мрежама. Више аутоматски скривених слојева користили би се за аутоматско дизајнирање неуронске мреже.

Приказани модел неуронске мреже за предвиђање успешности студената у области програмирања могао би да се побољша развојем модела **дубоке неуронске мреже** (*Deep Neural Networks (DNN)*). Овај модел укључио би више варијабли у вези са ангажовањем студента, породицом и аспектима понашања у учењу као варијабли за истовремено повећање тачности и осетљивости модела.

Дубоке неуронске мреже са неколико слојева постале су веома популарна тема истраживања машинског учења због својих одличних перформанси у многим референтним проблемима и апликацијама. Ови алгоритми су тренутно један од најпопуларнијих типова модела машинског учења, који су изграђени на принципу организације и функционисања биолошких неуронских мрежа.

ЛИТЕРАТУРА

[1] А. И. Карманчиков, „Логистика педагогического прогнозирования“, Монография, LAP Lambert Academic Publishing, 2012.

[2] A. ElGamal, „An Educational Data Mining Model for Predicting Student Performance in Programming Course“, *International Journal of Computer Applications*, Volume 70, No.17, pp. 22-28, May 2013.2013.

[3] S. Susan Bergin and R. Reilly, „Programming: factors that influence success“, *SIGCSE'05*, February 23–27, 2005 St. Louis, Missouri, USA, Volume 37 Issue 1, Pages 411-415, 2005.

[4] S. A. Naser, I. Zaqout, M. A. Ghosh, R. Atallah and E. Alajrami, „Predicting Student Performance Using Artificial Neural Network: in the Faculty of Engineering and Information Technology”, *International Journal of Hybrid Information Technology*, Vol.8, No.2, pp.221-228, 2015.

[5] E. Loshkareva, P. Luksha, I. Ninenko, I. Smagin, and D. Sudakov, „*Skills of the future*”, Global Education Future, Future Skills, WorldSkills Russia, 2017.

[6] M. Prensky, „Digital Natives, Digital Immigrants Part 1”, *On the Horizon*, MCB University Press, Vol. 9 Issue: 5, pp.1-6, 2001.

[7] J. Ruff, „*Information Overload: Causes, Symptoms and Solutions"*, Learning Innovations Laboratories, Harvard Graduate School of Education, 2002.

[8] A. Fraser and F. Dunstan, „On the impossibility of being expert”, *British Medical Journal (BMJ)*, Volume 341:c6815, pp. 1314-1315, 2010.

[9] О. В. Федорова, „Формирование hard skills, soft skills и digital skills у студентов факультета информационных технологий”, *Образовательные технологии и общество*, 2018, Т. 21, № 2, С. 335-340, 2018.

[10] NEA - National Education Association, „Preparing 21st Century Students for a Global Society: An Educator's Guide to the „Four Cs“ Great Public Schools for Every Student“, [Online], Available: available: https://www.academia.edu/36311252. [Accessed: June, 2021].

[11] I. Halinen, „The new educational curriculum in Finland“, *Improving the Quality of Childhood in Europe*, Volume 7, Chapter 6, pp. 75-89, Alliance for Childhood European Network Foundation, Brussels, Belgium, 2018.

[12] C. Tan, K. Koh, and W. Choy, „The education system in Singapore“, S. Juszczyk (Ed.), *Asian Education Systems* (pp. 129-148), Toruñ: Adam Marszalek Publishing House, 2016.

[13] „*Стратегија развоја образовања у Србији до 2020. године*“, „Службени гласник РС“, бр. 107/2012, 2012.

[14] „PISA 2018 results“, [Online], Available: available: https://www.oecd.org/pisa/PISA-results_ENGLISH.png, [Accessed: June, 2021].

[15] „Proposal for a council recommendation on Key Competences for Lifelong Learning“, ANNEX, EU Publications, European Commission, Brussels, 17.1.2018 COM(2018) 24 final, 2018.

[16] „*Стратегија развоја дигиталних вештина у Републици Србији за период 2019-2023. године*“, „Службени гласник РС“, број 30/18, 2018.

[17] „The Continuum of Understanding“, [Online], Available: available: http://www.nwlink.com/~donclark/performance/understanding.html, [Accessed: May, 2021].

[18] T. Davenport and L. Prusak, „Working Knowledge: How Organizations Manage What They Know“, *Ubiquity: an ACM IT Magazine and forum*, August 1 - August 31, 2000.

[19] Ю. Ю. Громов, И. В. Дидрих, О. Г. Иванова, М. А. Ивановский и В. Г. Однолько, „*Информационные технологии*“, Тамбов: ФГБОУ ВПО „ТГТУ“, 260 стр., 2015.

[20] J. Fišer, „Will we think in programming languages?“ *Acta Informatica Pragensia*, Vol 1, No 1, page 1-21, 2012.

[21] M. Vujošević and D. Tošić, „The role of programming paradigms in the first programming courses“, *The Teaching of Mathematics*, 2008, Vol. XI, 2, pp. 63–83, 2008.

[22] T. M. Pratt, and M. V. Zelkowitz, „*Programming Languages: Design and Implementation*“, 4th Edition, Pearson, Prentice Hall, pages 649, 2001.

[23] B. Rani, „Introduction of Programming Paradigms“, GeeksforGeeks, [Online], Available: available: https://www.geeksforgeeks.org/introduction-of-programming-paradigms/, [Accessed: March, 2021].

[24] S. Selvakumar, „An Insight intoProgramming Paradigms and TheirProgramming Languages*“*, *Journal of Applied Technology and Innovation*, vol. 1, no. 1, pp. 37-57, 2017.

[25] A. Ferguson, „The History of Computer Programming Languages“, [Online], Available: available: http://artemisa.unicauca.edu.co/~lgarreta/elenguajes/classes/intro/historyproglangs.pdf, [Accessed: March, 2021].

[26] D. Sureau, „History and Evolution of Programming Languages“, [Online], Available: available: https://www.scriptol.com/programming/history.php, [Accessed: April, 2021].

[27] G. O'Regan, „World of Computing, Introduction to Programming Languages“, *Springer International Publishing AG*, part of Springer Nature 2018, pp 155 -178 (Chapter 8), 2018.

[28] „Diagram & history of programming languages“, [Online], Available: available: http://rigaux.org/language-study/diagram.html, [Accessed: April, 2021].

[29] Č. Pavlović, „Istorijski razvoj programskih jezika“, [Online], Available: available: https://raf.edu.rs/citaliste/programiranje/3673-istorijski-razvoj-programskih-jezika, [Accessed: May, 2021].

[30] M. Lithmee, „Difference Between High Level Language and Low Level Language“, [Online], Available: available: https://www.differencebetween.com/difference-between-high-level-language-and-vs-low-level-language/, [Accessed: April, 2021].

[31] M. Boyer, „Grace Hopper and the Flow-Matic“. Flatiron school, January 28, 2016, [Online], Available: available: https://flatironschool.com/blog/code-history-lesson-grace-hopper/, [Accessed: April, 2021].

[32] P. Grogono, „Evolution of Programming Languages“, Notes for students, *Department of computer Science Concordia University*, Montreal, Canada, 2002.

[33] Tutorijals point, „Lisp tutorial“, [Online], Available: available: https://www.tutorialspoint.com/lisp/, [Accessed: May, 2021]

[34] D. Hemmendinger, „Algol computer language“, Ecyclopedia Britannica, [Online], Available: available: https://www.britannica.com/technology/ALGOL-computer-language, [Accessed: April, 2021]

[35] M. Bellis, „The History of the BASIC Programming Language“, ThoughtCo, [Online], Available: available: https://www.thoughtco.com/history-basic-programming-language-1991662, [Accessed: May, 2021]

[36] A. Sweigart, „*Automate the Boring Stuff with Python*“, Kompjuter biblioteka, 504 pp, 2015.

[37] A. Anam, „Ruby Programming Language (Introduction)“, GeeksforGeeks, [Online], Available: available: https://www.geeksforgeeks.org/ruby-programming-language/, [Accessed: April, 2021]

[38] C. Chiedo, „The Rise of Modern Programming Languages“, The Andela Way, [Online], Available: available: https://medium.com/the-andela-way/the-rise-of-modern-programming-languages-c923a2b914fc, [Accessed: June, 2021]

[39] M. Kamaruzzaman, „Top 7 Modern programming languages to learn now“, Towards data science, [Online], Available: available: https://towardsdatascience.com/top-7-modern-programming-language-to-learn-now-156863bd1eec, [Accessed: July, 2021]

[40] D. Gewirtz, „Which programming languages are most popular?“ ZDNet, [Online], Available: available: https://www.zdnet.com/article/which-programming-languages-are-most-popular-and-what-does-that-even-mean/, [Accessed: July, 2021]

[41] TIOBE Software BV, „TIOBE Programming Community Index“, [Online], Available: available: http://www.tiobe.com/tiobe_index, [Accessed: August, 2021]

[42] IEEE Spectrum Magazine, „Interactive: The Top Programming Languages 2021“, [Online], Available: available: https://spectrum.ieee.org/static/interactive-the-top-programming-languages-2020, [Accessed: August, 2021]

[43] „2020 Developer Survey“, [Online], Available: available: https://insights.stackoverflow.com/survey/2020, [Accessed: July, 2021]

[44] „Most loved programming languages“ [Online], Available: available: https://insights.stackoverflow.com/survey/2020#technology-most-loved-dreaded-and-wanted-languages-loved, [Accessed: July, 2021]

[45] „Most dreaded programming languages“ [Online], Available: available: https://insights.stackoverflow.com/survey/2020#technology-most-loved-dreaded-and-wanted-languages-dreaded, [Accessed: July, 2021]

[46] „Most wanted programming languages“ [Online], Available: available: https://insights.stackoverflow.com/survey/2020#technology-most-loved-dreaded-and-wanted-languages-wanted, [Accessed: July, 2021]

[47] A. Wirfs, „Programming Language Standardization“, *5th Asian Conference on Pattern Languages of Programs (AsianPLoP)*. AsianPLoP'2016, February 24-26, Taipei, Taiwan, pp 208-220, 2016.

[48] ISO/IEC, „ ISO, International Organization for Standardization, Store, Standards catalogue.“, [Online], Available: available: https://www.iso.org/standards-catalogue/browse-by-ics.html, [Accessed: April, 2021]

[49] ИСС, Институт за стандардизацију Србије, Напредна претрага, [Online], Available: available: https://iss.rs/sr_Cyrl/project/advanced-search, [Accessed: April, 2021]

[50] *„Међународна класификација стандарда“*, Седмо издање, Институт за стандардизацију Србије (ИСС), 2018.

[51] Ž. Micić and N. Stanković, „Trendovi znanja na platformi standardizacije životne sredine, zaštite zdravlja i bezbednosti*“, 16th International Conference ICDQM-2013*, pp. 519-529, 2013.

[52] N. Stankovic and Z. Micic, „Innovating and management of the knowledge base on the example of IT applications“, *Telematics and Informatics, Elsevier Science BV*, vol. 35, no. 5, pp. 1461 – 1472, Amsterdam, Aug, 2018.

[53] Ž. Micić, N. Stanković, and M. Blagojević, „Clustering of knowledge innovation in standardized „hardware's“ subfields of information technology*“, 5th International conference on Information technology and development of education, Idvor*, 27. June 2014. pp.319-326, 2014.

[54] Ž. Micić and N. Stanković, „Knowledge and innovations trends in metallurgy subfields within standardization platform“, *Metalurgia International, University Bucharest*, No 18 (8), page 154-160, 2013.

[55] Ž. Micić, M. Micić and M. Blagojević, „ICT innovations at the platform of standardisation for knowledge quality in PDCA“, *Computer Standards and Interfaces*, Volume 36, Issue 1, pp. 231-243, 2013.

[56] ISO/IEC, „ISO, International Organization for Standardization, Store, Standards catalogue, By ICS, 35: Information technology“, [Online], Available: available: https://www.iso.org/ics/35/x/, [Accessed: April, 2021]

[57] ISO/IEC, „ISO, International Organization for Standardization, Store, Standards catalogue, By ICS, 35.060: Languages used in Information technology“, [Online], Available: available: https://www.iso.org/ics/35.060/x/p/1/u/0/w/0/d/0, [Accessed: April, 2021]

[58] М. Ковачевић, К. Павловић, В. Шутић, „Употреба ИКТ у Републици Србији, 2018“, Републички завод за статистику, Београд, 2018.

[59] D. Soumitra and B. Lanvin, „The Network Readiness Index 2020: Accelerating Digital Transformation in a post-COVID Global Economy“, Portulans Instiute, Washington, USA, 2020.

[60] Министарство просвете, науке и технолошког развоја Републике Србије, „Нови План и програм за гимназије“, [Online], Available: available: http://www.mpn.gov.rs/novi-plan-i-program-za-gimnazije/, [Accessed: May, 2021]

[61] Министарство просвете, науке и технолошког развоја Републике Србије, „Гимназије – специјализована ИТ одељења“, [Online], Available: available: http://www.mpn.gov.rs/gimnazije-specijalizovana-it-odeljenja/, [Accessed: May, 2021]

[62] „Школски програм за информатику и рачунарство од петог до осмог разреда“, [Online], Available: available: http://www.prvaskola.edu.rs/document/skolski2018_19/Skolski%20program%20informatika%20i%20racunarstvo.pdf, [Accessed: May, 2021]

[63] *„Конкурс за упис ученика у први разред средње школе у Републици Србији за школску 2019/2020*“, „Службени гласник РС“, бр. 55/13 и 101/17 и 27/18 – др. закон, 2019.

[64] *„*Правилник *о наставном плану и програму за гимназију за ученике са посебним способностима за рачунарство и информатику“,* „Службени гласник РС“ - Просветни гласник број 5/17, 2017.

[65] *„Правилник о наставном плану и програму огледа за образовни профил електротехничар информационих технологија*“, „Просветни гласник 004/2012, 014/2015“, 2015.

[66] ЗУОВ, „*Планови и програми наставе и учења (наставни планови и програми) и остали правилници из области образовања“,* [Online], Available: available: http://zuov.gov.rs/nastavni-planovi-i-programi/#1557128435959-aaba5be0-1eed, [Accessed: May, 2021]

[67] The World Universitety Rankings, „World University Rankings 2020 by subject: computer science“, [Online], Available: available: https://www.timeshighereducation.com/world-university-rankings/2020/subject-ranking/computer-science#!/page/0/length/25/sort_by/rank/sort_order/asc/cols/stats, [Accessed: August, 2021]

[68] Edukacija, „Univerziteti u Srbiji“, [Online], Available: available: https://fakulteti.edukacija.rs/univerziteti, [Accessed: June, 2021]

[69] Techrocks.ru, „О типах программистов: специализация и мотивация“, [Online], Available: available: https://techrocks.ru/2018/09/14/programmers-types-and-motivations/, [Accessed: May, 2021]

[70] FTN Čačak, „Kurikulum studijskog programa IT – 2020“, [Online], Available: available: http://www.ftn.kg.ac.rs/studije/program/OAS_IT2020, [Accessed: July, 2021]

[71] D. Shiffman, „The Nature of Code: Simulating Natural Systems with Processing 1st Edition“, D. Shiffman, 2012.

[72] С. Г. Николаева, „*Нейронные сети. Реализация в Matlab*“, Казанский государственный энергетический университет, Казань, 92 стр, 2015.

[73] Б. Маркић, С. Бијакшић и А. Беванда, „Неуронске мреже у прогнозирању економског раста као индикатора конвергенције интеграцијског процеса“, *ACTA ECONOMICA*, година XV, број 26/јун 2017, 2017.

[74] R. J. Hyndman and G. Athanasopoulos, „Forecasting: Principles and practice (2nd ed)“, [Online], Available: available: https://otexts.com/fpp2/nnetar.html, [Accessed: June, 2021]

[75] M. B. Simonović, „*Primena veštačkih neuronskih mreža za kratkoročno predviđanje i analizu sistema daljinskog grejanja*“, doktorska disertacija, Univerzitet u Nišu, Mašinski fakultet, 2016.

[76] D. Mohsin, „Presentation on artificial neural networks“, [Online], Available: available: https://www.scribd.com/document/447012040/ANN-PPT, [Accessed: June, 2021]

[77] C.-T. Li and C. S. G. Lee, „*Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligen*“, Prentice-Hall PTR, Upper Saddle River, New Jersey, 1996.

[78] Е. Н. Горбачевская, „Классификация нейронных сетей“, Вестник Волжского университета имена В. Н. Татищева, №2 (19). - С. 1-6. 92, 2012

[79] F. Marić, „Nadzirano učenje u predviđanju prodaje prehrambenih proizvoda“, Sveučilište u Splitu, Ekonomski fakultet, Split, Hrvatska, 2016.

[80] R. Rojas, „*Neural Networks: A Systematic Introduction*“, Springer-Verlag, Berlin, 1996.

[81] C. Gershenson, „Artificial Neural Networks for Beginners“, *Computer Science*, ArXiv, 2003.

[82] Edureka!, „Backpropagation – Algorithm For Training A Neural Network“, [Online], Available: available: https://www.edureka.co/blog/backpropagation/, [Accessed: June, 2021]

[83] A. Apoorva, „Loss Functions and Optimization Algorithms. Demystified.“, Data Science Group, IITR, [Online], Available: available: https://medium.com/data-science-group-iitr/loss-functions-and-optimization-algorithms-demystified-bb92daff331c, [Accessed: June, 2021]

[84] B. J. Taylor, „*Methods and Procedures for the Verification and Validation of Artificial Neural Networks*“, Springer, 11/01/2005, pages 278, 2005.

[85] American Institute of Aeronautics and Astronautics, „*Guide for the Verification and Validation of Computational Fluid Dynamics Simulations*“, AIAA-G-077-1998, Reston, VA, 2014.

[86] C. Macal, „Model verification and validation“, *Workshop on „Threat Anticipation: Social Science Methods and Models“*, The University of Chicago and Argonne National Laboratory, April 7-9, Chicago, IL, 2005.

[87] E. Alyahyan and D. Düştegör, „Predicting academic success in higher education: literature review and best practices“, *International Journal of Educational Technology in Higher Education, (2020)17:3*, 2020.

[88] K. L. M. Pittman, „*Predicting Student Academic Success in a Developmental English Community College Course*“, Patton College of Education, of Ohio University, 2014.

[89] G. D. Kuh, J. Kinzie, J. A. Buckley, B. K. Bridges and J. C. Hayek, „What matters to student success: A review of the literature“, Commissioned Report for the National Symposium on Postsecondary Student Success: Spearheading a Dialog on Student Success, NPEC, National Center for Education Statistics, U.S. Department of Education July, 2006.

[90] T. T. York, C. Gibson, and S. Rankin, „Defining and Measuring Academic Success. Practical Assessment”, *Practical Assessment, Research & Evaluation*, Vol. 20, Article 5, 2015.

[91] Intellspot, „Key Types of Data Analysis Methods and Techniques“, [Online], Available: available: https://www.intellspot.com/types-data-analysis/, [Accessed: June, 2021]

[92] S.Glen, „Predictive Analytics Techniques in One Picture“, TechTarget, [Online], Available: available: https://www.datasciencecentral.com/profiles/blogs/predictive-analytics-in-one-picture, [Accessed: June, 2021]

[93] D. P. Ausubel, and F. G. Robinson,. „*School learning. An introduction to educational psychology.*“ New York: Holt, Rinehart and Winston, 691 pp, 1969.

[94] Д.Бјекић, „Обликовање наставе на основу стилова учења и мотивације за предмет“, *Васпитање и образовање*, 14(2), 2007.

[95] O. Stanislava, „*Motivacija i uvažavanje stilova učenja kao determinante učeničkog postignuća u hemiji*“, Doktorska disertacija, Univerzitet u Novom Sadu Prirodno-matematički fakultet, Novi Sad, 2016.

[96] D. A. Kolb, and R. Fry, „Toward an applied theory of experiential learning,“ In C. Cooper, Ed., *Theories of Group Process*, John Wiley, London, 33-5, 1975.

[97] A. Y. Kolb and D. A. Kolb, „The Kolb Learning Style Inventory 4.0: Guide to Theory, Psychometrics“, *Research & Applications*, Experience Based Learning Systems, 2013.

[98] Е. Н. Викторовна, „Формирование и развитие компетенций будущих специалистов по связям с общественностью с учетом индивидуальных стилей обучения“, [Online], Available: available: https://sites.google.com/site/konfep/Home/2-sekcia/emelanova, [Accessed: June, 2021]

[99] S. J. Sheel, D. Vrooman, R. S. Renner and S. K. Dawsey, „A comparison of neural networks and classical discriminant analysis in predicting students' mathematics placement examination scores“, *ICCS 2001, LNCS 2074*, pp. 952–957, Springer-Verlag Berlin Heidelberg, 2001.

[100] T. Wang, A Mitrovic, „Using neural networks to predict students' performance“, *Paper presented at the International Conference on Computers in Education*, Auckland, New Zealand, 2002.

[101] M. D Calvo-Flores, E. G. Galindo, M.C Pegalajar Jiménez and O. P. Piñeiro, „Predicting students' marks from Moodle logs using neural network models“, *Current Developments in Technology-Assisted Education*, pp. 586-59, 2006.

[102] M. A. P. Junemann, P. A. S. Lagos and R. C. Arriagada, „Neural networks to predict schooling failure/success“, *Lecture Notes in Computer Science, Nature Inspired Problem-Solving Methods in Knowledge Engineering*. Berlin: Springer, 4528: 571–579, 2007.

[103] I. Zaidah and R. Daliela, „Predicting Students' Academic Performance: Comparing Artificial Neural Network, Decision Tree and Linear Regression“, *21st Annual SAS Malaysia Forum*, 5th September 2007, Shangri-La Hotel, Kuala Lumpur, 2007.

[104] V. O. Oladokun, A. T. Adebanjo, and O. E. Charles-Owaba, „Predicting Students' Academic Performance using Artificial Neural Network: A Case Study of an Engineering Course“, *The Pacific, Journal of Science and Technology*, Volume 9, Number 1, pp. 72-79, 2008.

[105] S. T Karamouzis and A. Vrettos, „An Artificial Neural Network for Predicting Student Graduation Outcomes“, *Proceedings of the World Congress on Engineering and Computer Science, WCECS 2008,* October 22 - 24, San Francisco, USA pp.991-994, 2008.

[106] I. Lykourentzou, I. Giannoukos, G. Mpardis, V. Nikolopoulos and V. Loumos, „Early and Dynamic Student Achievement Prediction in E-Learning Courses Using Neural Networks“, *Journal of the American Society for Information Science and Technology*, 60(2):372-380, 2009.

[107] C. I. Cooper, „A Generalizable Neural Network for Predicting Student Retention“, Professional Papers Proceedings Archive, [Online], Available: available: https://www.asee.org/documents/zones/zone1/2010/professional/A-Generalizable-Neural-Network-for-Predicting-Student-Retention.pdf, [Accessed: April, 2021]

[108] C. E. Moucary, M. Khair and W. Zakhem, „Improving Student's Performance Using Data Clustering and Neural Networks in Foreign-Language Based Higher Education“, *The Research Bulletin of Jordan, ACM*, vol II, pp. 27-34, 2011.

[109] P. Halachev, „Prediction of e-Learning Efficiency by Neural Networks“, *Cybernetics and information technologies*, Volume 12, No 2, Bulgarian academy of sciences, Sofia, 2012.

[110] R. A. Adeleke, A. A Ruzainin and Q. Hongwu, „Risk Status Prediction and Modelling of Students' Academic Achievement-A Fuzzy Logic Approach“, *Research Inventy: International Journal of Engineering and Science*, 3(11):7-14, 2013.

[111] B. Oancea, R. Dragoescu, and S. Ciucu, „Predicting students' results in higher education using a neural network.“, *International Conference on Applied Information and Communication Technologies (AICT2013)*, April, pp. 190-193, 2013.

[112] S. E. Sorour, T. Mine, K. Goda and S. Hirokawa, „Predicting students' grades based on free style comments data by artificial neural network“, *2014 IEEE, Frontiers in Education (FIE) Conference*, 22-25 Oct. 2014, Madrid, Spain, 2014.

[113] G. Lesinski, S. Corns and C. Dagli, „Application of an Artificial Neural Network to Predict Graduation Success at the United States Military Academy“, Procedia Computer Science, Volume 95, Pages 375-382, 2016.

[114] M. Sivasakthi, Classification and prediction based data mining algorithms to predict students' introductory programming performance, *2017 IEEE, International Conference on Inventive Computing and Informatics (ICICI)*, 23-24 Nov. 2017, Coimbatore, India, 2017.

[115] F. Yang and F. W. B. Li, „Study on student performance estimation, student progress analysis, and student potential prediction based on data mining“, *Computers & Education*, 123, 97–108, 2018.

[116] A. D. Kumar, R. P. Selvam and K. S. Kumar, „Review on prediction algorithms in educational data mining“, *International Journal of Pure and Applied Mathematics*, 118, 531–536, 2018.

[117] Z. Ahmad and E. Shahzadi, „Prediction of Students' Academic Performance using Artificial Neural Network“, *Bulletin of Education and Research*, December, Vol. 40, No. 3 pp. 157-164, 2018.

[118] L. K. Hamoud and A. M. Humadi, „Student's Success Prediction Model Based on Artificial Neural Networks (ANN) and A Combination of Feature Selection Methods", *Journal of Southwest Jiaotong University*, vol.54, no.3, June, 2019.

[119] J. Pei and P. Shan, „A Micro-expression Recognition Algorithm for Students in Classroom Learning Based on Convolutional Neural Network", IIETA, Traitement du Signal, Vol. 36, No 9, December, p557-563, 2019.

[120] A. A. M. Beram, N. A. H. Almehthel and S. A. A. Damosh, „Using Artificial Neural Networks to Predict Student Stumbling at Najran University", *International Journal of Engineering Research & Technology (IJERT)*, Vol. 9 Issue 02, February-2020, 2020.

[121] E. T. Lau, L. Sun and Q. Yang, „Modelling, prediction and classification of student academic performance using artificial neural networks", *Springer Nature Journal Applied Sciences (2019)* 1:982, September, 2019.

[122] V. Vijayalakshmi and K. Venkatachalapathy, „Comparison of Predicting Student's Performance using Machine Learning Algorithms", *I. J. Intelligent Systems and Applications, 2019*, 12, 34-45, 2019.

[123] Ş. Aydoğdu, „Predicting student final performance using artificial neural networks in online learning environments", *Education and Information Technologies*, v25 n3 p1913-1927, May, 2020.

[124] A. Çetinkaya, Ö. K. Baykan, „Prediction of middle school students' programming talent using artificial neural networks", *Engineering Science and Technology, an International Journal*, Volume 23, Issue 6, December 2020, Pages 1301-1307, 2020.

[125] C. F. Rodríguez-Hernández, M. Musso, E. Kyndt and E. Cascallar, „Artificial neural networks in academic performance prediction: Systematic implementation and predictor evaluation", *Computers and Education: Artificial Intelligence*, Volume 2, 2021, 100018, 2021.

[126] P. R. Jenkins, W. N. Caballero and R. R.Hill, „Predicting success in United States Air Force pilot training using machine learning techniques", *Socio-Economic Planning Sciences*, Available online 12 July 2021, 101121, 2021.

[127] IBM support, „Downloading IBM SPSS Statistics 20", [Online], Available: available: https://www.ibm.com/support/pages/downloading-ibm-spss-statistics-20, [Accessed: February, 2021].

[128] S. J. Coakes, „*SPSS verzija 20.0 - analiza bez muke*", Kompjuter biblioteka – Beograd, 2013.

[129] J. Pallant, „*SPSS: priručnik za preživljavanje*", prevod 6. izdanja, Mikro knjiga, 2017.

[130] Softonic, „Download Weka for PC", [Online], Available: available: https://weka.en.softonic.com/, [Accessed: February, 2021]

[131] R. R. Bouckaert, E. Frank, M. Hall, R. Kirkby, P. Reutemann, A. Seewald and D. Scuse, „*WEKA Manual for Version 3-7-8*", University of Waikato, Hamilton, New Zealand, 2013.

[132] S. Biswas, „Prediction and Analysis of Student Performance by Data Mining in WEKA", *Report of Project submitted for the partial fulfillment of the requirements for the degree of Bachelor of Technology in Information Technology*, RCC Institute of Information Technology, Canal South Road, Beliaghata, Kolkata, India, 2018.

[133] Microsoft, „Download Visual Studio", [Online], Available: available: https://visualstudio.microsoft.com/vs/older-downloads/, [Accessed: March, 2021]

[134] M. Snell, L. Powers, „Microsoft Visual Studio 2015 Unleashed", 3th Edition. - Sams Publishing, 2015.

[135] Републички завод за статистику, Република Србија, „Општине и региони у Републици Србији, 2020", [Online], Available: available: https://publikacije.stat.gov.rs/G2020/Pdf/G202013047.pdf, [Accessed: April, 2021]

[136] Министарство просвете, науке и технолошког развоја Републике Србије, „Средње школе и образовни профили", [Online], Available: available: https://mojasrednjaskola.gov.rs/Cir/Pocetna, [Accessed: May, 2021]

[137] U. Verma, „Data Cleaning and Preprocessing“, Analytics Vidhya, [Online], Available: available: https://medium.com/analytics-vidhya/data-cleaning-and-preprocessing-a4b751f4066f, [Accessed: May, 2021]

[138] H. Almarabeh, „Analysis of Students' Performance by Using Different Data Mining Classifiers“, *International Journal of Modern Education and Computer Science* 9(8):9-15, August, 2017.

[139] S. Sebastian, J. J. Puthiyidam, „Evaluating Students Performance by Artificial Neural Network using WEKA“, *International Journal of Computer Applications* (0975 – 8887) Volume 119 – No.23, June, 2015.

[140] J. Đ. Novaković, „*Rešavanje klasifikacionih problema mašinskog učenja*“, Reinženjering poslovnih procesa, Monografija, Fakultet tehničkih nauka u Čačku, Univerzitet u Kragujevcu, Čačak, 2013.

[141] N. Z. Vidič, „Primena mašinskog učenja u verifikaciji softvera“, Master rad, Univerzitet u Beogradu, Matematički fakultet, Beograd, 2019.

[142] J. Mohajon, „Confusion Matrix for Your Multi-Class Machine Learning Model“, Towards Data Science, [Online], Available: available: https://towardsdatascience.com/confusion-matrix-for-your-multi-class-machine-learning-model-ff9aa3bf7826, [Accessed: June, 2021]

[143] V. Zablotski, „Confusion Matrix and it's 25 offspring: or the link between machine learning and epidemiology“, https://yury-zablotski.netlify.app/post/confusion-matrix/, Available: available: [Online], Available: available:, [Accessed: June, 2021]

[144] D. E. Parson, „Data Mining and Predictive Analytics Kappa Statistic“, [Online], Available: available: http://faculty.kutztown.edu/parson/fall2019/Fall2019Kappa.html, [Accessed: June, 2021]

[145] B. Shmueli, „Multi-Class Metrics Made Simple, Part III: the Kappa Score (aka Cohen’s Kappa Coefficient)“, Towards Data Science, [Online], Available: available: https://towardsdatascience.com/multi-class-metrics-made-simple-the-kappa-score-aka-cohens-kappa-coefficient-bdea137af09c, [Accessed: June, 2021]

[146] I. Baskin and A. Varnek, „Tutorial on Classification“, Faculté de Chimie de Strasbourg, Laboratoire d‟Infochimie, Strasbourg, France, [Online], Available: available: https://masterchemoinfo.u-strasbg.fr/Documents/TutoChemo/classification.pdf, [Accessed: June, 2021]

[147] K2 Analytics, „Classification Accuracy & AUC ROC Curve“, [Online], Available: available: https://www.k2analytics.co.in/classification-accuracy-auc-roc-curve/, [Accessed: June, 2021]

[148] B. Wang, „ROC vs PRC“, Beverly Wang, [Online], Available: available: https://beverly-wang0005.medium.com/roc-vs-prc-6cd30d287a4b, [Accessed: June, 2021]

[149] M. Döring, „Interpreting ROC Curves, Precision-Recall Curves, and AUCs“, Data Science Blog, [Online], Available: available: https://www.datascienceblog.net/post/machine-learning/interpreting-roc-curves-auc/, [Accessed: June, 2021]

[150] „Gaining an intuitive understanding of Precision and Recall“, Laptrinhx, [Online], Available: available: https://laptrinhx.com/gaining-an-intuitive-understanding-of-precision-and-recall-2631030324/, [Accessed: June, 2021]

[151] D. Chicco, N. Tötsch and G. Jurman, „The Matthews correlation coefficient (MCC) is more reliable than balanced accuracy, bookmaker informedness, and markedness in two-class confusion matrix evaluation“, *BioData Mining 14*, Article number: 13 (2021), 2021.

[152] N. Stanković, M. Blagojević, M. Papić, D. Karuović, „Artificial Neural Network Model for Prediction of Students’ Success in Learning Programming“, *Journal of Scientific and Industrial Research*, Vol. 80, pp. 249-254, March, 2021.

[153] NN-SVG, „Publication-ready NN-architecture schematics“, [Online], Available: available: http://alexlenail.me/NN-SVG/index.html, [Accessed: July, 2021]

ПРЕГЛЕД СЛИКА И ТАБЕЛА

Преглед слика

Преглед табела

ПРИЛОГ

Прилог 1. Упитник „Развој програмерских вештина“, први део – општи подаци о студенту, самопроцене

I. ОПШТИ ПОДАЦИ О СТУДЕНТУ

Заокружите одговор или **упишите** одговор на предвиђено место:

1. Ваше име и презиме: _____
2. За даљи развој овог истраживања, важно је да будем са Вама лично у контакту. Молим Вас да упишите Ваш контакт мејл: _____
3. Пол: Мушки Женски
4. Средња школа (унесите пун назив смера/занимања који сте завршили):
   - Гимназија, смер _____
   - Средња стручна школа, занимање _____
5. Година студија: Прва Друга Трећа Четврта
6. Остварени резултати до сада (закључно са априлским роком):
   - Број освојених ЕСП бодова: _____
   - Просечна оцена свих положених предмета: _____
7. Заокруживањем ДА или НЕ означите предмете, код којих сте предавања/вежбе, у току студирања, посећивали више од 50%. У другом делу означите да ли сте предмете положили.

Предмети	Предавања и вежбе похађано више од 50%		Положено	
1. Увод у програмирање	ДА	НЕ	ДА	НЕ
2. Програмски језици	ДА	НЕ	ДА	НЕ
3. Објектно оријентисано програмирање	ДА	НЕ	ДА	НЕ
4. Веб технологије	ДА	НЕ	ДА	НЕ
5. Интернет програмирање	ДА	НЕ	ДА	НЕ
6. Програмирање база података	ДА	НЕ	ДА	НЕ
7. Софтверско инжењерство	ДА	НЕ	ДА	НЕ
8. Савремене софтверске архитектуре	ДА	НЕ	ДА	НЕ

8. Уредите следеће ставке према важности за Вас, уписујући 1 код најважнијег разлога за упис студијског програма Информационе технологије до 5 код најмање важног разлога:

   _____ стицање општег програмерског знања
   _____ сигурно запослење, добра зарада, могућност рада на даљину („од куће“)
   _____ веб и графички дизајн/програмирање, мобилно рачунарство
   _____ администрирање рачунарских мрежа и база података
   _____ одржавање серверских и клијентских мрежа

   Ако Ваш најважнији разлог није међу наведенима, упишите га на црту: _____
   _____

II. На основу досадашњег искуства о програмирању, **заокружите** исказ који најбоље описује **ВАШУ ТРЕНУТНУ СПРЕМНОСТ ДА ПРОГРАМИРАТЕ**.

1. Нисам сигуран/на да могу самостално да креирам програм.
2. Могу самостално да креирам програм уз нечију помоћ.
3. Могу самостално да креирам једноставан програм, али ми је потребно много времена да то урадим.
4. Могу самостално да креирам сложен програм на најмање једном програмском језику.
5. Могу самостално да креирам сложене програме на више програмских језика.

III. Наведени су одређени програмерски задаци. **Заокруживањем** оцене од 1 (веома тешко) до 5 (веома лако) процените **КОЛИКО БИ ВАМ ТЕШКО/ЛАКО БИЛО** да их решите уз приступ основној литератури за тај задатак.

РЕШАВАЊЕ ПРОГРАМЕРСКИХ ЗАДАТАКА	Веома тешко	Тешко	Осредње, ни тешко ни лако	Лако	Веома лако
1. Писање секвенцијално-структуираних програма у неком од програмских језика	1	2	3	4	5
2. Писање програма у неком од програмских језика који укључује условне и цикличне структуре	1	2	3	4	5
3. Писање програма у неком од програмских језика који укључује комплексне структуре (на пример: показиваче, структуре, рекурзивне функције)	1	2	3	4	5

IV. **Уписивањем** оцене од 1 (највише) до 3 (најмање) процените који **ТИП ПРОГРАМИРАЊА** преферирате:

_____ Објектно оријентисано програмирање (Java, C++, C#, PHP, Python, ...)

_____ Процедурално програмирање (C, Pascal, BASIC, Fortran, Ada, ...)

_____ Функционално и декларативно програмирање (SQL, HTML, XML, LISP, Prolog, ...)

V. **Заокруживањем** оцене од 1 (веома слабо) до 5 (веома добро) процените Ваш **НИВО ЗНАЊА** за наведене програмске језике.

ПРОГРАМСКИ ЈЕЗИЦИ	Веома слабо	Слабо	Средње	Добро	Веома добро
1. Basic	1	2	3	4	5
2. Pascal	1	2	3	4	5
3. Delphi	1	2	3	4	5
4. C	1	2	3	4	5
5. C++	1	2	3	4	5
6. C#	1	2	3	4	5
7. Java	1	2	3	4	5
8. Python	1	2	3	4	5
9. Visual Basic for Applications	1	2	3	4	5
10. Matlab	1	2	3	4	5
11. LabVIEW	1	2	3	4	5
12. _____	1	2	3	4	5
13. _____	1	2	3	4	5

VI. **Заокруживањем** ДА или НЕ означите да ли сте, за наведене програмске језике, имали **ФОРМАЛНО ОБРАЗОВАЊЕ** (образовање у основној, средњој школи, на факултету), односно **ОБРАЗОВАЊЕ ВАН ШКОЛЕ** (курсеви, семинари, радионице, тренинзи, онлајн образовање, волонтирање).

ПРОГРАМСКИ ЈЕЗИЦИ	ФОРМАЛНО ОБРАЗОВАЊЕ		ОБРАЗОВАЊЕ ВАН ШКОЛЕ	
1. Basic	ДА	НЕ	ДА	НЕ
2. Pascal	ДА	НЕ	ДА	НЕ
3. Delphi	ДА	НЕ	ДА	НЕ
4. C	ДА	НЕ	ДА	НЕ
5. C++	ДА	НЕ	ДА	НЕ
6. C#	ДА	НЕ	ДА	НЕ
7. Java	ДА	НЕ	ДА	НЕ
8. Python	ДА	НЕ	ДА	НЕ
9. Visual Basic for Applications	ДА	НЕ	ДА	НЕ
10. Matlab	ДА	НЕ	ДА	НЕ
11. LabVIEW	ДА	НЕ	ДА	НЕ
12. _____	ДА	НЕ	ДА	НЕ
13. _____	ДА	НЕ	ДА	НЕ

VII. Наведите **програмске језике** које сте учили у средњој школи: _____

_____

VIII. Ако би радили у индустрији, **заокруживањем** оцене на скали од 1 (најнижи) до 5 (највиши), процените Ваш тренутни **НИВО ПРОГРАМЕРСКЕ СПРЕМНОСТИ** за рад у индустрији.

1 2 3 4 5

IX. Да ли сте радили неки **ПРОГРАМЕРСКИ ПОСАО** и за кога?

- Врста посла _____
- Фирма _____
- Врста посла _____
- Фирма _____
- Врста посла _____
- Фирма _____

X. Ако сте у току досадашњег школовања били **НАГРАЂИВАНИ** из области програмирања и информационих технологија, наведите награду и годину:

- _____
- _____

Ваш **КОМЕНТАР**: _____

_____

_____

_____

Молимо Вас да проверите да ли сте одговорили на сва питања

ЗАХВАЉУЈЕМО НА САРАДЊИ!

Прилог 2. Упитник „Развој програмерских вештина“, други део – тест знања из области програмирања

I. У следећим задацима **одредите тачност исказа** заокруживањем слова испред одговора.

1. Једном написана функција се НЕ МОЖЕ поново користити у неком другом програму.
   а) Тачно б) Нетачно в) Не знам
2. Унутрашње промењљиве су расположиве и другим функцијама.
   а) Тачно б) Нетачно в) Не знам
3. Функција може имати максимално 12 аргумената.
   а) Тачно б) Нетачно в) Не знам
4. Рекурзија се обично извршава спорије од циклуса.
   а) Тачно б) Нетачно в) Не знам

II. У наредним задацима (1-4) потребно је да **одредите резултате** након извршења кодова датих програма **заокруживањем тачног одговора** од понуђених одговора (узмите у обзир и грешке при компајлирању). Кодови програма су написани у програмским језицима C и Java и, за исти задатак, дају идентична решења. Заокружите програмски језик у коме сте решавали сваки задатак.

ЗАДАТАК БРОЈ 1

Програмски језик C:

```
#include<stdio.h>
int j=3;
int funkcija()
{
   printf("DA ");
}
int main()
{
   while(j<5)
   {
      funkcija();
      j++;
   }
   printf("NE ");
   funkcija();
}
```

Заокружите програмски језик у коме сте решавали задатак:

C Java

Заокружите решење задатка:

а) DA DA NE DA DA

б) DA DA NE DA

в) DA DA NE

г) грешка при компајлирању

д) не знам

Програмски језик Java:

```
static int j=3;
static void funkcija()
{
        System.out.print("DA ");
}

public static void main(String[] args)
{
        while(j<5)
        {
                funkcija();
                j++;
        }
        System.out.print("NE ");
        funkcija();
}
```

ЗАДАТАК БРОЈ 2

Програмски језик C:

```
#include<stdio.h>
void check(int a, int b)
{
   if(a>=5)
      return(b);
   else
      return(a);
}
int main()
{
   int x=15;
   x = check(10, 20);
   printf("%d", x);
}
```

Заокружите програмски језик у коме сте решавали задатак:

C Java

Заокружите решење задатка:

а) 10

б) 20

в) 15

г) грешка при компајлирању

д) не знам

Програмски језик Java:

```
static void check(int a, int b)
{
        if(a>=5)
        {
                return (b);
        }
        else
        {
                return (a);
        }
}
public static void main(String[] args)
{
        int x=15;
        x=check(10, 20);
        System.out.print(x);
}
```

ЗАДАТАК БРОЈ 3

Програмски језик C:

```
#include<stdio.h>
int xyz(int b) {
      b++;
      return (b);
}
void main()
{
      int b=200;
      xyz(b);
      printf("%d ",b);
      b=xyz(b);
      printf("%d",b);
}
```

Заокружите програмски језик у коме сте решавали задатак:

C Java

Заокружите решење задатка:

а) 200 200

б) 200 201

в) 201 201

г) грешка при компајлирању

д) не знам

Програмски језик Java:

```
static int xyz(int b)
{
      b++;
      return (b);
}
public static void main(String[] args)
{
      int b=200;
      xyz(b);
      System.out.print(b+" ");
      b=xyz(b);
      System.out.print(b);
}
```

ЗАДАТАК БРОЈ 4

Програмски језик C:

```
#include<stdio.h>
void f1()
{
      static int s=5;
      ++s;
      printf("%d ",s);
}
main()
{
      int s=3;
      f1();
      f1();
      printf("%d",s);
}
```

Заокружите програмски језик у коме сте решавали задатак:

C Java

Заокружите решење задатка:

а) 5 6 7

б) 6 7 3

в) 6 6 3

г) грешка при компајлирању

д) не знам

Програмски језик Java:

```
static int s=5;
static void f1()
{
      ++s;
      System.out.print(s+" ");
}

public static void main(String[] args)
{
      int s=3;
      f1();
      f1();
      System.out.print(s);
}
```

III. У наредним задацима (5-8) потребно је да **одредите резултате** након извршења кодова датих програма **уписивањем тачног одговора** на предвиђено место (узмите у обзир и грешке при компајлирању). Ако не знате решење, ставите косу црту. Кодови програма су написани у програмским језицима C и Java и, за исти задатак, дају идентична решења. Заокружите програмски језик у коме сте решавали сваки задатак.

ЗАДАТАК БРОЈ 5

Програмски језик C:

```
#include<stdio.h>
int fun1(int k);
int main()
{
   int k=30;
   k = fun1(fun1(fun1(k)));
   printf("%d", k);
   return 0;
}
int fun1(int k) {
   k++;
   return k; }
```

Заокружите програмски језик у коме сте решавали задатак:

C Java

Упишите решење задатка:

_____

Програмски језик Java:

```
public static void main(String[] args)
{
      int k=30;
      k=fun1(fun1(fun1(k)));
      System.out.print(k);
}

static int fun1(int k)
{
      k++;
      return k;
}
```

ЗАДАТАК БРОЈ 6

Програмски језик C:

```
#include<stdio.h>
int xyz(int b) {
      b++;
}
void main()
{
```

Програмски језик Java:

```
static void xyz(int b)
{
      b++;
}
public static void main(String[] args)
{
```

```
    int b=200;
    xyz(b);
    printf("%d ",b);
    b=xyz(b);
    printf("%d",b);
}
```

Заокружите програмски језик у коме сте решавали задатак:

C Java

Упишите решење задатка:

_____

```
    int b=200;
    xyz(b);
    System.out.print(b+" ");
    xyz(b);
    System.out.print(b);
}
```

ЗАДАТАК БРОЈ 7

Програмски језик C:

```
#include<stdio.h>
void f1()
{
    int s=5;
    ++s;
    printf("%d ",s);
}
main()
{
    int s=3;
    f1();
    f1();
    printf("%d",s);
}
```

Заокружите програмски језик у коме сте решавали задатак:

C Java

Упишите решење задатка:

_____

Програмски језик Java:

```
static void f1()
{
    int s=5;
    ++s;
    System.out.print(s+" ");
}

public static void main(String[] args)
{
    int s=3;
    f1();
    f1();
    System.out.print(s);
}
```

ЗАДАТАК БРОЈ 8

Програмски језик C:

```
#include<stdio.h>
#define swap(a,b) a=a+b;b=a-b;a=a-b;
int swap2(int a, int b)
{
   int temp;
   temp=a;
   b=a;
   a=temp;
   return 0;
}
void main()
{
   int x=5, y=10;
   swap (x,y);
   printf("%d %d",x,y);
   swap2(x,y);
 printf("\t%d %d",x,y);
}
```

Заокружите програмски језик у коме сте решавали задатак:

C Java

Упишите решење задатка:

_____

Програмски језик Java:

```
public class Numbers
{
    public int a;
    public int b;
}

static void swap(Numbers num)
{
    num.a=num.a+num.b;
    num.b=num.a-num.b;
    num.a=num.a-num.b;
}
static int swap2(int a, int b)
{
    int temp;
    temp=b;
    b=a;
    a=temp;
    return 0;
}

public static void main(String[] args)
{
    Numbers num = new Numbers();
    num.a=5;
    num.b=10;
    swap(num);
    System.out.print(num.a+" "+num.b);
    swap2(num.a, num.b);
    System.out.print("\t"+num.a+" "+num.b);
}
```

IV. За наредна 2 задатка потребно је **написати одговарајући код** коришћењем програмског језика по Вашем избору. На почетку листинга наведите програмски језик у којем радите задатак.

1. Написати функцију која налази колико има троцифрених бројеве код којих су цифре стотине, и јединице узастопни (растући) бројеви (102, 112, ... 243, 253, ... 677, 687, ..., 889, 899)

   Програмски језик у коме је рађен задатак: _____

2. Написати класу Student са приватним атрибутима: број индекса, име и презиме, просечна оцена. Креирати два објекта класе Student и иницијализовати вредност ових објеката. Написати конструктор, као и функције за приступ приватним атрибутима класе Student.

   Програмски језик у коме је рађен задатак: _____

Молимо Вас да проверите да ли сте урадили све задатке
ЗАХВАЉУЈЕМО НА САРАДЊИ!

Прилог 3. Упитник „Развој програмерских вештина“, трећи део – Колбов инвентар стилова учења

Овај део истраживања је намењен одређивању начина на који Ви лично учите. Не одређује Ваше капацитете за учење и способности, већ само начин учења.

Наведено је девет група са по четири појма и исказа. Означите појмове у једној групи по редоследу који је применљив за Ваш начин учења. Означите у свакој групи исказ који се највише односи на Ваш стил учења бројем 4, а са 1 онај исказ који се најмање односи на Ваш начин учења. У свакој групи исказа упишите сваки број само једном.

ЗАХВАЉУЈЕМО НА САРАДЊИ!

Дакле, на овом листу у девет група су дата 4 типа описа. Означите описе који се највише односе на Ваш начин учења (најбоље Вас описују), а затим и остале, на следећи начин:

4 – за реч која најбоље описује Ваш приступ учењу,

3 – за реч која само мало мање одговара Вашем начину учења,

2 – за реч која још мање одговара,

1 – за реч која најмање одговара Вашем приступу учењу.

Наводимо пример оваквог рангирања неких осећања:

4 срећан	3 брз	1 љут	2 пажљив

1. ред	☐ **диференцирано**	☐ **пробно**	☐ **интегрисано**	☐ **практично**
2. ред	☐ **осетљиво**	☐ **значајно**	☐ **аналитичко**	☐ **непристрасно**
3. ред	☐ **осећати**	☐ **гледати**	☐ **мирисати**	☐ **извести**
4. ред	☐ **прихватити**	☐ **ризиковати**	☐ **вредновати**	☐ **осветити**
5. ред	☐ **интуитивно**	☐ **продуктивно**	☐ **логично**	☐ **истраживачко**
6. ред	☐ **апстрактно**	☐ **опажајно**	☐ **конкретно**	☐ **активно**
7. ред	☐ **садашњост**	☐ **рефлективно**	☐ **будућност**	☐ **прагматично**
8. ред	☐ **искуство**	☐ **размишљање**	☐ **осмишљавање**	☐ **експериментисање**
9. ред	☐ **интензивно**	☐ **резервисано**	☐ **рационално**	☐ **одговорно**

На десној страни је објашњење појмова које у табели рангирате

од 4 – најбоље описује Ваше учење, до 1 – најмање описује Ваше учење:

Објашњење појмова у првом реду:

Диференцирано: У току учења тражим различитости.
Пробно: У току учења прво нешто пробам и покушавам.
Интегрисано: У току учења покушавам да обухватим више ствари.
Практично: У току учења сам усмерен/а на практичну примену.

Објашњење појмова у другом реду:

Осетљиво: У току учења тежим да препознајем мале разлике – осетљив/а сам на мале разлике.
Значајно: У току учења сам фокусиран/а на оно што је релевантно-важно.
Аналитички: У току учења прво анализирам ситуацију.
Непристрасно: У току учења не доносим судове или изборе на основу неких појединачних предности.

Објашњење појмова у трећем реду:

Осећати: У току учења преферирам оно што може да покрене моја осећања.
Гледати: У току учења преферирам оно што могу да видим.
Мирисати: У току учења користим мирисне доживљаје.
Извести: У току учења преферирам оно што могу да изведем (покретом на пример).

Објашњење појмова у четвртом реду:

Прихватити: У току учења тежим да прихватам ствари онаквим какве јесу.
Ризиковати: У току учења сам склон/а да ризикујем у ономе што ћу да урадим или кажем.
Вредновати: У току учења тежим да вреднујем, да доносим вредносне судове.
Освестити: У току учења покушавам да будем свестан/на свега што се догађа.

Објашњење појмова у петом реду:

Интуитивно: У току учења сам склон/а да делујем на основу своје интуиције.
Продуктивно: У току учења сам усмерен/а на крајњи резултат.
Логично: У току учења у првом кораку покушавам да логички мислим.
Истраживачки: У току учења постављам себи питања и дилеме.

Објашњење појмова у шестом реду:

Апстрактно: У току учења покушавам да откријем апстракције и важне формалне појмове.
Опажајно: У току учења сам склон/а да гледам и слушам.
Конкретно: У току учења преферирам оно што је конкретно.
Активно: У току учења тежим да учим кроз активност и практичне методе рада.

Објашњење појмова у седмом реду:

Садашњост: У току учења сам усмерен/а на оно што је овде и сада.
Рефлексивност: У току учења тежим да све филтрирам кроз своје мисли.
Будућност: У току учења сам усмерен/а на оно што може да се догоди у будућности.
Прагматичност: У току учења сам усмерен/а на оно што има прагматичну/практичну природу.

Објашњење појмова у осмом реду:

Искуство: У току учења тежим да учим кроз искуство.
Размишљање: У току учења тежим да размишљам о ономе што слушам и гледам.
Осмишљавање: У току учења тежим да сазнања повежем у интегрисану и структуирану смисаону целину.
Експериментисање: У току учења тежим да проверавам и експериментишем са идејама.

Објашњење појмова у деветом реду:

Интензивно: У току учења тежим да искусим шта се догађа емоционално и интензивно.
Резервисано: У току учења преферирам да задржим дистанцу према ономе што се догађа.
Рационално: Мој приступ ономе што се дешава углавном је са дистанце.
Одговорно: Активно преузимам одговорност за оно што се догађа.

БИОГРАФИЈА

Небојша Станковић рођен је у Гњилану 08. 08. 1966. год. Основну и средњу школу завршио је у Чачку, занимање програмер. Дипломирао је на Техничком факултету у Чачку, 1991. године са темом „Софтверски систем за пројектовање конвертора једносмерне струје“ и стекао звање Дипломирани инжењер електротехнике. Магистарску тезу под насловом „Прилог симулацији рачунарских архитектура“ одбранио је 29. септембра 2009. године на Техничком факултету у Чачку и тиме стекао звање Магистар техничких наука. Ради на Факултету техничких наука у Чачку непрекидно од 1. јануара 1992. године.

Аутор је уџбеника „Мултимедијалне технологије и системи“. Учесник је на пројекту МПНТР Републике Србије под називом „Иновирање курикулума и наставничких компетенција за предмет Мултимедијални системи на основним академским студијама Информационе технологије 2018“. Од 2008. овлашћени је ECDL (European Computer Driving Licence) испитивач. Коаутор је три програма стручног усавршавања наставника акредитованих од стране Министарства просвете и спорта Републике Србије: „Употреби рачунар у настави - диференцирана обука наставника“ (2002-2006); „Диференцирана информатичка обука наставника (2007/2008); „Хипермедија у настави“ (2011/2012). Сарадник је и инструктор на акредитованом програму стручног усавршавања наставника: „Комуникација и учење у настави технике (КУНТ)“ (2002-2006, 2007-2009). Дугогодишњи је члан организационог одбора међународне научне конференције Техника и информатика у образовању (ТИО). Више пута је био предавач на курсевима за информатичку обуку запослених и незапослених за потребе Завода за тржиште рада, извођених у организацији Факултета техничких наука у Чачку.

Кандидат има објављено, као први аутор или коаутор, више од 50 радова у часописима, као и на међународним и домаћим конференцијама.

Образац 1

ИЗЈАВА АУТОРА О ОРИГИНАЛНОСТИ ДОКТОРСКЕ ДИСЕРТАЦИЈЕ

Ја, **Небојша Љ. Станковић**, изјављујем да докторска дисертација под насловом:

ФАКТОРИ УЧЕЊА И ПРЕДВИЂАЊЕ УСПЕШНОСТИ У ПРОГРАМИРАЊУ ПРИМЕНОМ ВЕШТАЧКИХ НЕУРОНСКИХ МРЕЖА

која је одбрањена на **Факултету техничких наука у Чачку**, Универзитета у Крагујевцу представља *оригинално ауторско дело* настало као резултат *сопственог истраживачког рада*.

Овом Изјавом такође потврђујем:

- да сам *једини аутор* наведене докторске дисертације,
- да у наведеној докторској дисертацији *нисам извршио/ла повреду* ауторског нити другог права интелектуалне својине других лица,
- да умножени примерак докторске дисертације у штампаној и електронској форми у чијем се прилогу налази ова Изјава садржи докторску дисертацију истоветну одбрањеној докторској дисертацији.

У **Чачку**, **27. 08. 2021.** године,

потпис аутора

Образац 2

ИЗЈАВА АУТОРА О ИСКОРИШЋАВАЊУ ДОКТОРСКЕ ДИСЕРТАЦИЈЕ

Ја, **Небојша Љ. Станковић**,

☑ дозвољавам

☐ не дозвољавам

Универзитетској библиотеци у Крагујевцу да начини два трајна умножена примерка у електронској форми докторске дисертације под насловом:

ФАКТОРИ УЧЕЊА И ПРЕДВИЂАЊЕ УСПЕШНОСТИ У ПРОГРАМИРАЊУ ПРИМЕНОМ ВЕШТАЧКИХ НЕУРОНСКИХ МРЕЖА

која је одбрањена на **Факултету техничких наука у Чачку**, Универзитета у Крагујевцу, и то у целини, као и да по један примерак тако умножене докторске дисертације учини трајно доступним јавности путем дигиталног репозиторијума Универзитета у Крагујевцу и централног репозиторијума надлежног министарства, тако да припадници јавности могу начинити трајне умножене примерке у електронској форми наведене докторске дисертације путем *преузимања*.

Овом Изјавом такође

☑ дозвољавам

☐ не дозвољавам[1]

[1] Уколико аутор изабере да не дозволи припадницима јавности да тако доступну докторску дисертацију користе под условима утврђеним једном од *Creative Commons* лиценци, то не искључује право припадника јавности да наведену докторску дисертацију користе у складу са одредбама Закона о ауторском и сродним правима.

припадницима јавности да тако доступну докторску дисертацију користе под условима утврђеним једном од следећих *Creative Commons* лиценци:

1) Ауторство

2) Ауторство - делити под истим условима

3) Ауторство - без прерада

4) Ауторство - некомерцијално

5) Ауторство - некомерцијално - делити под истим условима

6) Ауторство - некомерцијално - без прерада[2]

У **Чачку**, **27. 08. 2021.** године,

Ђ. Ђурановић

потпис аутора

[2] Молимо ауторе који су изабрали да дозволе припадницима јавности да тако доступну докторску дисертацију користе под условима утврђеним једном од *Creative Commons* лиценци да заокруже једну од понуђених лиценци. Детаљан садржај наведених лиценци доступан је на: http://creativecommons.org.rs/

Образац 3

ИЗЈАВА О ИСТОВЕТНОСТИ ШТАМПАНЕ И ЕЛЕКТРОНСКЕ ВЕРЗИЈЕ ДОКТОРСКОГ РАДА

Име и презиме аутора: **Небојша Љ. Станковић**

Број индекса: **915/2017**

Студијски програм: **ДАС Информационе технологије**

Наслов рада: **Фактори учења и предвиђање успешности у програмирању применом вештачких неуронских мрежа**

Ментор: **др Марија Благојевић, ванредни професор**

Потписани: **Небојша Љ. Станковић**

Изјављујем да је штампана верзија мог докторског рада истоветна електронској верзији коју сам предао за објављивање на порталу Дигиталног репозиторијума Универзитета у Крагујевцу.

Дозвољавам да се објаве моји лични подаци везани за добијање академског звања доктора наука, као што су име и презиме, година и место рођења и датум одбране рада.

Ови лични подаци могу се објавити на мрежним страницама дигиталне библиотеке, у електронском каталогу и у публикацијама Универзитета у Крагујевцу.

У **Чачку**, **27. 08. 2021.** године,

Потпис докторанда,

Станковић