|  |
| --- |
| RĪGAS TEHNISKĀ UNIVERSITĀTE |
| Datorzinātnes un informācijas tehnoloģijas fakultāte |
| Informācijas tehnoloģijas institūts |
| **<studējošā vārds, uzvārds>**  <bakalaura/maģistra akadēmisko/profesionālo, pirmā/otrā līmeņa studiju programmas> students, stud. apl. nr. <.......> |
| **<DARBA/PROJEKTA TEMATIKAS NOSAUKUMS>** |
| **< BAKALAURA/MAĢISTRA DARBS, DIPLOMDARBS, DIPLOMPROJEKTS, INŽENIERPROJEKTS,**  **KVALIFIKĀCIJAS DARBS>** |
| Zinātniskais vadītājs  <zinātniskais grāds, akadēmiskais amats>  <vārds, uzvārds> |
| RĪGA <20….> |

*Šī lapa netiek izdrukāta – tās vietā vēlāk gatavam darbam tiek pievienota uzdevumu lapa.*

|  |
| --- |
| RĪGAS TEHNISKĀ UNIVERSITĀTE |
| DATORZINĀTNES UN INFORMĀCIJAS TEHNOLOĢIJAS FAKULTĀTE |
| Informācijas tehnoloģijas institūts |
| INFORMĀCIJAS TEHNOLOĢIJAS ATTĪSTĪBA ELEKTRONISKAJĀ TIRDZNIECĪBĀ |
| Edvīns Mališevskis |
| Anotācija |

|  |
| --- |
| RIGA TECHNICAL UNIVERSITY |
| FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY |
| Institute of information technology |
| DEVELOPMENT OF INFORMATION TECHNOLOGY IN ELECTRONIC COMMERCE |
| Edvīns Mališevskis |
| Annotation |

Goals of the work, results and their estimations.

The work contains \_\_\_ p., \_\_\_ tables, \_\_\_\_ figures and \_\_\_ appendixes.

saturs

[Ievads 3](#_Toc444598137)

[1. SAGATAVOŠANĀS DARBI UN IETEIKUMI 3](#_Toc444598138)

[1.1. Sagatavošanās darbu secība 3](#_Toc444598139)

[1.1.1. Rezerves kopijas izveidošana 3](#_Toc444598140)

[1.1.2. Virsrakstu izveidošana objektiem 3](#_Toc444598141)

[1.2. Ieteikumi drukas teksta formatēšanai 3](#_Toc444598142)

[2. VISPĀRĒJIE NORĀDĪJUMI 3](#_Toc444598143)

[3. TEKSTA NOFORMĒŠANA 3](#_Toc444598144)

[3.1. Sadaļu numerācija 3](#_Toc444598145)

[3.2. Sadaļu virsraksti 3](#_Toc444598146)

[3.2.1. Nodaļas virsraksts 3](#_Toc444598147)

[3.2.2. Apakšnodaļas virsraksts 3](#_Toc444598148)

[3.2.3. Punktu virsraksts 3](#_Toc444598149)

[3.2.4. Apakšpunktu virsraksts 3](#_Toc444598150)

[4. OBJEKTU IEKļAUšANA TEKSTĀ 3](#_Toc444598151)

[4.1. Attēli 3](#_Toc444598152)

[4.2. Tabulas 3](#_Toc444598153)

[4.3. Formulas 3](#_Toc444598154)

[5. izmantotās literatūras SARAKSTA sakārtoŠANA un atsauces 3](#_Toc444598155)

[6. PIELIKUMI 3](#_Toc444598156)

[rezultāti un SECINĀJUMI 3](#_Toc444598157)

[Izmantotās literatūras saraksts 3](#_Toc444598158)

[pielikumi 3](#_Toc444598159)

[1. pielikums. Noslēguma darba vāks 3](#_Toc444598160)

[2. pielikums. Pielikumā iekļauto objektu noformēšanas paraugi 3](#_Toc444598161)

[3. pielikums. Atsauču noformēšana 3](#_Toc444598162)

Ievads

Šī veidne ir veidota saskaņā ar RTU Informācijas tehnoloģijas institūta noslēguma darbu izstrādes noteikumiem un ietver sevī dažādus automatizācijas elementus, kas vienkāršos darba noformēšanu un novērsīs problēmas darba satura izmaiņu gadījumos.

Dokumenta saturs ietver sevī arī pašus Noformēšanas norādījumus, kurus ir ieteicams izlasīt, lai izvairītos no pretrunām un nesaskaņām nākotnē, lietojot dažādas *Word* versijas un valodas uzstādījumus.

Lietotājam ieteicams glabāt neskartu veidnes versiju, jo tas satur dažādus elementus, kuru pavairošana veicama tikai ar kopēšanas palīdzību.

# Programmēšanas valodas

## Jaunās programmēšanas valodas

Pasaulē eksistē milzīgs daudzums programmēšanas valodu, taču tas netraucē pastāvīgi rasties jaunām valodām. Rodas pilnīgi dabisks jautājums- kādēļ tās rodas. Eksistē taču tādi industrijas titāni kā “Java”, “C”, “C++”,” Objective-C”, kā arī ne tik veci, taču ieņēmuši specifiskas nišas- “JavaScript”, “PHP”,”Ruby”. Tam ir vairāki iemesli un pirmais ir valodu fokuss. Ir jāpiekrīt, ka zemu līmeņu mehanizmu programmēšanai labāk izmantot C, bet interaktīvus vebelemetu- Javascript. Otrs iemesls ir valodas arhitektūra. Ne visas valodas var pielietot smagi piekrautās programmās ar lielu paralēlu skaitļošanu daudzumu. Arhitektūras specifikas kā “null[[1]](#footnote-1)” un ”Generics”. Daudzi programmētāji uzskata, ka null ir milzīga programmēšanu valodu kļūda. Un, piemēram, Java veidotāji atvainojas par Generics ieviešanu. Taču tā kā lielākā daļa programmēšanas valodu ir atgriezeniski saderīgas(Backward compatibility)[[2]](#footnote-2) , nav iespējams izmainīt vai uzlabot šīs daļas. Tieši šis ir iemesls, kādēļ rodas jaunas programmēšanas valodas, kuras ne atkārto savu priekšnācēju kļūdas un optimizējas jaunajām programmēšanas tendencēm. Zemāk būs apskatītas dažas programmēšanas valodas, kuras izveidojās pēdējo trīs gadu laikā.

### Google Go

Go- arī pazīstams kā golang tika palaists 2009. Gadā. Tā veidotāji ir trīs Google darbinieki Robert Griesemer, Rob Pike, and Ken Thompson. Šī publiskās piekļuves(open source) valoda tika raksturota kā ātra un viegli izmantojama, salīdzinot ar eksistējošām valodām kā Java un C. Tieši šīs valodas tika izmantotas kā Go pamats. Tā tiek izmantota visdažādākās organizācijās sākot ar BBC, SoundCloud un Facebook, līdz pat Lielbritānijas valdības mājaslapai GOV.UK. Tā tiek arī izmantota Docker tehnoloģijās. Go veidotāji vēsta: „Go ir mēģinājums kombinēt programmēšanas vieglumu dinamiskā valodā ar statisku, kompilētu valodu efektivitāti un drošību”

### Swift

Swift tika atklāts Apple WWDC conference laikā 2014. Gadā. Tās veidošanas nolūks bija Objective-C valodas aizvietošana priekš OSX un iOS attīstībā. Apple šo valodu publiskoja 2015. gada Decembrī zem Apache licenzijas. Tas nozīmē, ka viss avota kods(source code) ir atklāti pieejams iz.maiņām un uz tā pamatiem var tikt veidotas programmas, nesaistītas ar Apple. Swift, kuram piemīt līdzība ar modernajām Ruby un Python valodām, tika pakļauts neizmērojamai izaugsmei kopš tā palaišanas.”Swift iekļauj sevī drošu programmēšanu un modernus uzlabojumus, lai padarītu programmēšanu vieglu, dinamisku un interesantu” saka Apple.

### Scala

Scala ir funkcionāla un objekt-orientēta valoda, no kuras vārda var secināt par tās mērogojamību(scalability). Tā tiek pielietota labi pazīstamās organizācijās. Tā ir cieši saistīta ar JVM[[3]](#footnote-3). Tā tiek pielietota lielās uzdevumu kritiskās sistēmās(large mission critical systems) kā piemēram Twitter, LinkedIn un Intel.

### Rust

Rust 1.0 tika izveidots 2014. gadā Mozilla kompānijas iniciatīvas dēļ un uzlabots šo dažu gadu laikā. Dažos aspektos Rust ir līdzīgs C un C++, un Mozilla apraksta to kā „ jaunu programmēšanas valodu, kura fokusējas uz procesu izpildi, paralelizēšanu un atmiņas drošību”. Veidojot jaunu valodu un pieliekot elementus no mūsdienu programmēšanas valodu dizaina, Rust veidotāji izvairās no liela daudzuma atgriezeniski saderības prasībām, kurām tradicionālās valodas velta lielu uzmanību.

### Ceylon

Ceylon ir Objekt-orientēta augsta līmeņa valoda ar prasīgu statisku tipizāciju un SDK[[4]](#footnote-4), kuru pilnveido Red Hat kompānija. Pirmā informācija par šo valodu parādijās 2011. gadā, kad tās veidotāji izveidoja blogu, kur dalījās ar valodas panākumiem un sasniegumiem. Pirmā pilnā versija tika laista klajā 2013. gada 12. Novembrī. Šī valoda var tikt kompilēta ar JVM vai arī JavaScript.

## Specificētas un universālas valodas

Katra programmēšanas valoda var tikt izmantota vienā vai vairākās nozarēs. Pamatnozares iekļauj sevī- Web, Desktop, Mobile,Embedded.

* Web – Valodas aplikācijām, kurām ir piekļuve caur internetu un kuras strādā attālinātā serverī. Simbols- aplis ar līniju noklājumu.
* Desktop – Valodas programmām, kuras tiek instalētas un izmantotas uz konkrēta datora. Pašlaik tās ir pārsvarā tikai zinātkiska programmas. Simbols- monitors
* Mobile – Valodas applikācijām, kuras strādā uz telefoniem(smartfoniem). Simbols- smartfons.
* Embedded –Programmēšanas valodas zemu līmeņu ierīcēm. Simbols-mikročips.



1.1. att. Valodu specialitāte un popularitāte

Kā to var redzēt attēlā 1.1, eksistē valodas ar ļoti šauru specialitāti kā R un arī pietiekami universālas valodas kā Java un C. Savukārt Web nodaļa dalās 2 daļās backend [[5]](#footnote-5)un frontend[[6]](#footnote-6). Parasti backend un frontend tiek rakstīti dažādās valodās. Parasti backend programmēšanā izmanto Java, C# vai PHP, taču frontend programmēšanā izmanto Javascript, tomēr gandrīz visās valodā ir freimvorks(???)[[7]](#footnote-7) frontend programmēšanai. Programmēšanas valoda, kura tiks apskatīta pirmā, saucas Ceylon. To var uzskatīt par ļoti jaunu valodu, kura tiks vaļā no problēmām, kuras roda Java programmētājiem, vēsta tās izveidotāji.

# CEYLON

## Почему Ceylon

На JVM основано достаточно много языков а именно : Scala, Groovy, Clojure, Kotlin, Ceylon, Xtend, X10. Все эти языки имеют доступ ко всей, нажитой долгими годами, инфрастуктуре Java. Ко всем библиотекам, фреймворкам и прочему. Это одна из причин по которой я считаю, что языки основанные на JVM, будут иметь успех в одной из самых популярных индустрий. Речь идет о Web разработке, больших сложных систем, которые используются в больших предприятиях. Ведь Java сейчас занимает лидирущую позицию в этой области. Мой же выбор пал на Ceylon. Я считаю, что Ceylon самый перспективный язык из списка JVM языков. Ceylon не является функциональным языком программирования, что как я считаю, является плюсом, современные системы достаточно сложны и без функицональных элементов. Ceylon может быть скомпилирован в JavaScript, что является уникальной особенностью среди JVM языков. Это позволяет писать части программы которые будут выполняться непосредственно у пользователя в браузере, а не на сервере.

### Red Hat

Red Hat это самая крупная компания занимающаяся разработкой Open-Source Software[[8]](#footnote-8) продуктов, основонная в 1993 году. Это компания которая создала бесплатную и открытую операционную систему, которая в свою очередь используется по всему миру – Linux. Red Hat также занимается поддержкой и предоставлением услуг связанных со своими продуктами. Это ещё одна причина по которой я считаю, что Ceylon самый перспективный язык. Ведь, кто если не люди создавшие Linux, и множество других продуктов, люди который сотрудничают с ведущими компаниями в своих индустриях, могут создать лучший язык на основе JVM.

## Альтернативы JVM

Хотелось бы детальнее рассмотреть другие языки JVM и попробовать оценить их как альтернативу устоявшейся Java. Как уже было сказано выше Java является лидирующим языком для разработки серверной части тяжеловесных, больших и серьезных приложений для корпораций или банков.

### Scala

Scala is a statically typed programming language that fuses the object-oriented model and functional programming ideas. That means, in practice, that you can declare classes, create objects, and call methods just like you would typically do in Java. However, Scala also brings popular features from functional programming languages such as pattern matching[[9]](#footnote-9) on data structures, local type inference[[10]](#footnote-10), persistent collections[[11]](#footnote-11), and tuple[[12]](#footnote-12) literals. Я считаю, что Scala не очень подходит для больших проектов в которых работает много людей, только если требуется большая масштабируемость. Scala подойдет для проектов с очень высокой нагрузкой, и очень хорошим быстродействием. Имеет все возможности Java а так же возможность работать почти со всеми существующими библиотеками.

### Groovy

[Groovy](http://groovy.codehaus.org/) is a dynamically typed object-oriented language. Groovy’s dynamic nature lets you manipulate your code in powerful ways. For example, you can expand objects at runtime [[13]](#footnote-13) - by adding fields or methods. С моей точки зрения Groovy не подходит для больших проектов и комманд. Динамическая типизация не дает уверенности, когда много людей работает над проектом и каждый человек не знает всех частей. Зато на Groovy очень удобно покрывать тестами код. Тесты получаются лаконичнее. Имеет все возможности Java, некоторые из них даже дополнены, например работа с коллекциями.

### Clojure

[Clojure](http://www.clojure.org/) is a dynamically typed programming language that can be seen as a modern take on Lisp[[14]](#footnote-14). It is radically different from what object-oriented programmers might be used to. In fact, Clojure is a fully functional programming language, and as a result, it is centered on immutable data structures, recursion, and functions. Я считаю, что Clojure подходит только для научной деятельности, как и любой другой полностью функциональный язык программирования и абсолютно не применим в корпоративной разработке. Имеет доступ к Java экосистеме, по скольку язык не объектно – ориентированный, то функциональность может быть ограничена.

### Kotlin

[Kotlin](http://kotlin.jetbrains.org/) is a statically typed object-oriented language. Its main design goals are to be compatible with Java’s API, have a type system that catches more errors at compile time, and be less verbose than Java. Kotlin’s designers say that Scala is a close choice to match its design goals, but they dislike Scala’s complexity and long compilation time compared to Java. Kotlin aims to tackle these issues. По моему мнению Kotlin лучше всех предидущих языков подходит для больших проектов. Уменьшенное время компиляции на ряду с системой отлова большей части исключений во время компиляции кода, позволит допускать меньше ошибок. Разработчиком Kotlin является компания – Jetbrains, которая разработала ряд IDE[[15]](#footnote-15) для самых разных языков. IDE для Java не так давно стала самой популярной для этого Языка.

### Xtend

[Xtend](https://www.eclipse.org/xtend/) is a statically typed object-oriented language. One way it differs from other languages is that it compiles to pretty-printed Java code rather than bytecode. As a result, you can also work with the generated code.

Xtend supports two forms of method invocation: default Java dispatching and multiple dispatching. With multiple dispatching, an overloaded method is selected based on the runtime type of its arguments. Xtend provides many other popular features available in other languages such as operator overloading and type inference. Интересная особенность языка это - активные аннотации. Активные аннотации это способ использовать метапрограммирование[[16]](#footnote-16) во время компиляции кода. Мне кажется, главная особенность этого языка является его главным недостатком, зачем программировать на языке который в следствии переводится в Java код. Считаю, что этот язык будет сложно и не удобно использовать в большом корпоративном проекте.

### X10

[X10](http://x10-lang.org/) is an experimental object-oriented language that IBM developed. It supports features such as first-class functions[[17]](#footnote-17) and is designed to facilitate efficient programming for high-performance parallel computing. To this end, the language is based on a programming model called the “partitioned global address space”. In this model, each process shares a global address space, and slices of this space are allocated as private memory for local data and access. To work with this model, X10 offers specialized built-in language constructs to work with concurrency and distributed execution. По скольку IBM сами называют этот язык эксперементальным, использовать его в корпоративных проектах – глупо. Я считаю, что этот язык будут использовать сами IBM для создания какой-нибудь очень сложной системы на подобии IBM Watson[[18]](#footnote-18).

# TEKSTA NOFORMĒŠANA

Rakstot tekstu, jālieto fonts *Times New Roman*,– 12. lielums, ar 1,5 intervālu starp rindām[[19]](#footnote-19). Rindkopas pirmo rindiņu sāk ar atkāpi. Rindkopas kārto bez atstarpēm. Darba sadaļas sakārto šādā secībā:

* titullapa,
* darba uzdevums,
* anotācijas (latviešu un angļu valodā),
* saturs,
* ievads,
* darba pamatsadaļas,
* rezultāti un secinājumi,
* izmantotās literatūras saraksts,
* pielikums(i),
* darba novērtējuma lapa.

Darba vāku, titullapu un satura rādītāju noformē atbilstoši ITI noslēguma darbu noformēšanas norādījumos dotajiem paraugiem, vai skatoties šī dokumenta – veidnes – attiecīgas daļas.

## Sadaļu numerācija

Darba galvenās sadaļas numurē un piešķirtos numurus raksta pirms sadaļas virsraksta. Nodaļas numurē ar arābu cipariem, piemēram: 1., 2. utt. Apakšnodaļas numurē attiecīgās nodaļas ietvaros ar diviem arābu cipariem, piemēram: 1.2., 2.3. utt. Apakšnodaļas virsraksts un teksts seko uzreiz pēc iepriekšēja teksta tajā pašā lappusē. Ja apakšnodaļa sastāv no vairākiem punktiem, tad tos numurē attiecīgās apakšnodaļas ietvaros ar trim arābu cipariem, piemēram: trešās nodaļas otrās apakšnodaļas punkti tiks numurēti 3.2.1., 3.2.2. utt.

Anotācijām, saturam, ievadam, secinājumiem un priekšlikumiem, literatūras sarakstam kārtas **numurus nepiešķir**.

* Ja darba teksta sadaļās paredzēts veidot sarakstus, to apzīmējumam lieto aizzīmi (*bullets*) • vai ciparus (*numbering*).
* Saraksta punktu numerācijai var izmantot vai nu arābu ciparus ar punktu vai iekavu, piemēram: 1., 2. vai 1), 2) utt., vai arī lielos latīņu burtus ar punktu vai mazos burtus ar iekavu, piemēram: A., B. vai a), b) utt.
* Tekstu aiz punkta sāk ar lielo burtu, bet aiz iekavas – ar mazo burtu.

## Sadaļu virsraksti

Darbā var būt izdalītas šādas sadaļas: 1. līmenis – nodaļa, 2. līmenis – apakšnodaļa, 3. līmenis – punkts.

Virsraksta attālums no iepriekšēja un turpmākā teksta ir 12 mm. Virsrakstos vārdus pārnest nav atļauts. Virsrakstus nepasvītro, tā beigās punktu neliek. Virsrakstu rakstīšanai ir ieteicams lietot MS Word stilus *Heading 1* (nodaļām), *Heading 2* (apakšnodaļām) utt. Tas nodrošina automātisku satura radītāja izveidošanu.

### Nodaļas virsraksts

Nodaļu (pirmā līmeņa) virsrakstus raksta ar izceltiem **1. LIELAJIEM BURTIEM** (**Bold**, 14. lielums, novietojums – lapas vidū). Katru nodaļu sāk ar jaunu lappusi.

### Apakšnodaļas virsraksts

Apakšnodaļu (otrā līmeņa) virsrakstus raksta ar izceltiem **1.2. Mazajiem burtiem** un lielo sākuma burtu (**Bold**, 14. lielums, novietojums – lapas vidū).

### Punktu virsraksts

Punktu (trešā līmeņa) virsrakstus raksta ar izceltiem ***1.2.1. Mazajiem burtiem*** un lielo sākuma burtu (**Bold**, *Italic*, 13. lielums, novietojums – lapas kreisajā malā).

### Apakšpunktu virsraksts

Ja nepieciešams, teksta strukturizēšanai iespējams izmantot arī 4. līmeņa sadalījumu – apakšpunktus ar nosaukumiem, bet tos nekļauj satura rādītājā.

Apakšpunktu (ceturtā līmeņa) virsrakstus raksta ar izceltiem ***1.2.1.1. Mazajiem burtiem*** un lielo sākuma burtu (**Bold**, *Italic*, 12. lielums, novietojums – lapas kreisajā malā).

# OBJEKTU IEKļAUšANA TEKSTĀ

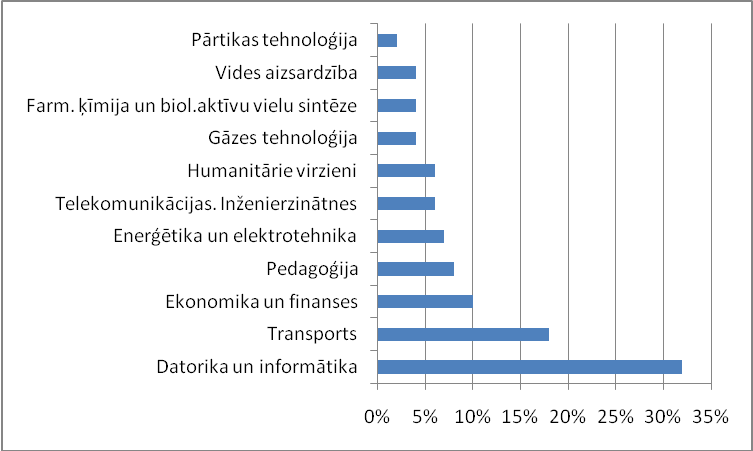
Par iekļautiem noslēguma darbā objektiem var būt uzskatīti attēli, tabulas un matemātiskās izteiksmes.

Pirmajā nodaļā tika aprakstīti sagatavošanās darbi objektu izmantošanai un automātiskai numurēšanai, skat. 1.1.2. punktu. Ja tā netika izlasīta un attiecīgie pasākumi netika veikti, tad tas ir jāizdara tagad.

Kā jau tika minēts, katru no objekta veidiem ieteicams noformēt ar eksistējošu piemēru kopēšanas palīdzību, vēlāk nomainot attiecīgās objekta saturu un virsrakstu. Atsauces ievietojamas ar opcijas *References→Cross-reference* palīdzību, no objektu saraksta izvēloties attiecīgo tipu (*Reference Type*). Lai ievietotu tikai attiecīgā objekta numuru, tad jāizvēlas opcija *Only label and number* no *Insert reference to* saraksta.

## Attēli

Darbā ievietoto ilustrāciju – fotogrāfiju, shēmu, grafiku, diagrammu un tml. – apzīmēšanai izmanto vienu un to pašu terminu "attēls". Attēli tiek numurēti nodaļas ietvaros, un katram no tiem ir jābūt savam nosaukumam. Attēla numuru un nosaukumu raksta simetriski zem ilustrācijas. Nosaukumu raksta treknrakstā ar maziem burtiem un lielo sākuma burtu, bez punkta nosaukuma beigās. Zemāk redzams 4.1. attēla piemērs.



4.1. att. **Bakalaurantu un maģistrantu stipendiju piešķīrumi pa nozarēm**

Ja attēlam nepieciešami paskaidrojumi, tos raksta zem nosaukuma centrā (paskaidrojumus var rakstīt, lietojot mazāka lieluma fontu), šajā gadījumā aiz nosaukuma liek kolu.

* Tekstā, kura ilustrēšanai izmantots attēls, attiecīgajā vietā uz to jānorāda atsauce, piemēram: “... uzskatāmi ilustrē 4.1. attēlā sniegtais zīmējums...” vai "4.1. att. sniegtais...". Tas ir viegli izdarāms ar automātisko atsauču sistēmu, par kuru minēts **Error! Reference source not found.**. nodaļas sākumā.
* Atsaucei uz ilustrāciju tekstā pirmoreiz jāparādās **pirms** attēla.
* Ievietojot atsauci, attēla **paskaidrojumam tekstā jāatšķiras no attēla virsraksta** (tas pats attiecas arī uz tabulām!).
* Attēlus ieteicams izveidot un iekļaut tekstā kā veselus objektus, proti, caur izvēlni *Insert→Object→Microsoft Visio* vai arī iekļaut attēlu no citiem grafiskajiem redaktoriem, lietojot opciju *Create from File*.
* Ja grafiskie objekti (attēli) tiek veidoti, lietojot opciju *Insert→Shapes*, tad visiem attēla elementiem jābūt sagrupētiem, lai rezultātā attēls būtu kā viens **vesels** bloks.
* Šāds attēls kā vesels bloks tiek ievietots darbā atsevišķajā rindā, piemēram, lietojot stilu Picture.
* Ja attēls (vai tabula) ir ņemts no kāda avota, tad tas jāiekļauj bibliogrāfijas sarakstā un attiecīgajā vietā obligāti jānorāda **atsauce uz oriģinālo** bibliogrāfijas avotu (pie paša attēla vai tabulas nosaukuma).

## Tabulas

Katrai darbā ievietotajai tabulai jābūt numurētai un ar savu nosaukumu. Tabulas numurē ar arābu cipariem **tikai nodaļas** ietvaros, piemēram: 1.1. tabula, 2.3. tabula, utt.

Tabulas numuru raksta labajā pusē virs tabulas nosaukuma. Tabulas nosaukumu raksta simetriski virs tabulas ar izceltiem maziem burtiem un lielo   
sākuma burtu (**Bold**, 12. lielums, novietojums − lapas **vidū**), bez punkta nosaukuma beigās.

Zemāk redzams tabulu noformēšanas paraugs ar bieži lietojamu saīsinājumu un to atšifrējumu aprakstu (4.1. tab.).

4.1. tabula

Tradicionālie saīsinājumi, kurus noslēguma darbā nav nepieciešams atšifrēt

| N.p.k. | Saīsinājums | Atšifrējums |
| --- | --- | --- |
| 1. | lpp. | lappuse |
| 2. | n.p.k. | numurs pēc kārtas |
| 3. | sk. vai skat. | skatīt |
| 4. | š.g. | šā gada; šī gada |
| 5. | t.i. | tas ir |
| 6. | u.c. | un citi |
| 7. | u.tml. | un tamlīdzīgi |
| 8. | utt. | un tā tālāk |

Tekstā attiecīgā vietā jādod atsauce uz tabulu, piemēram: “… kā rāda 4.1. tabulas dati...”

* Atsaucei uz tabulu tekstā pirmoreiz jāparādās **pirms** tabulas.
* Tabulas nosaukumu **nepasvītro**.
* Tabulas nosaukumā vēlams **izvairīties no komatiem**.
* Tabulas **galvenei** jābūt iezīmētai kā *Header Row*. Ja tabula aizņem vairākas lapaspuses, galveni atkārto.
* Tukšā tabulas rindā (ailē, šūniņā) ievelk svītriņu vai liek daudzpunktus.
* Nākamai pēc tabulas rindkopai jālieto stils „Normal pēc tab/form” vai „Normal pēc tab/form ar Tab”,, kas nodrošina attālumu pirms rindkopas sākuma.

## Formulas

Darbā ievietotās matemātiskās formulas iekļauj tekstā, taču katru no tām raksta savā atsevišķā rindiņā. Formulas numurē nodaļas ietvaros ar arābu cipariem, kurus raksta aiz formulas rindiņas labajā pusē, piemēram (1.1), (2.3) utt.

Numuru ieslēdz parastajās (apaļās) iekavās un raksta vienā rindā ar formulu lapas labajā malā. Aiz formulas kārtas numura punktu neliek, piemēram:

|  |  |
| --- | --- |
| , | (4.1) |

|  |  |  |  |
| --- | --- | --- | --- |
| kur |  | – | nepieciešamais materiāla daudzums gadā, kg; |
|  |  | – | materiāla patēriņa norma, kg/gab.; |
|  |  | – | gada ražošanas apjoms, gab. |

* Formulu labākam noformējumam ieteicams lietot **neredzamas tabulas**: pašai formulai – viena rinda ar divām kolonnām; eksplikācijai – tabula ar 4 kolonnām.
* Tekstā, atsaucoties uz kādu no formulām, tās numuru raksta tāpat kā aiz formulas – apaļajās iekavās, piemēram: "Izmantojot formulu (4.1) …".
* Atsaucei uz formulu pirmoreiz jāparādās **pirms** formulas.
* Ja formula aizņem vairākas rindiņas, numuru raksta pēdējās rindiņas labajā pusē.
* Formula parasti ir teikuma neatņemama daļa, un tajā teikumā korekti jālieto pieturzīmes.
* Formulā izmantotajiem simboliem jābūt atšifrētiem eksplikācijā, kuru raksta tūlīt aiz formulas, pirmo rindiņu sākot ar vārdu "kur". To raksta lapas kreisajā malā, kolu aiz tā neliek.
* Simboliem / atšifrējumiem tekstā / eksplikācijā **jāizskatās tāpat** kā pašā vienādojumā.
* Ja formulā izmantotiem elementiem ir zināmas mērvienības, tām jābūt iekļautām eksplikācijā.
* Formulu un vienādojumu rakstīšanai obligāti jāizmanto formulu **redaktors**, piemēram, *Microsoft Equation* vai *Math Type Equation*.

# izmantotās literatūras SARAKSTA sakārtoŠANA un atsauces

Diakopoulos, N., Cass,S. Interactive: *The Top Programming Languages 2016* [online].

IEEE SPECTRUM, 2016 [skatīts 2017. g. 3. majs.]. Available from: http://spectrum.ieee.org/static/interactive-the-top-programming-languages-2016

Finnegan, M. 10 up-and-coming programming languages developers should get to know [online].

Tech World, 2017 [skatīts 2017. g. 4. majs.]. Available from:

http://www.techworld.com/picture-gallery/apps/10-up-coming-programming-languages-for-developers-get-grips-with-3621455/

Raoul-Gabriel, U. Alternative Languages for the JVM [online].

Java Magazine 2014 [skatīts 2017. g. 5. majs.]. Available from:

http://www.oracle.com/technetwork/articles/java/architect-languages-2266279.html

Ja darbā ir atreferēts, citēts vai citādi tieši vai netieši izmantots materiāls no kādas grāmatas vai cita avota, tad tekstā ir jāatsaucas uz šo avotu. Darba autora pienākums ir precīzi norādīt informācijas avotu un autoru. Pareizi noformēta atsauce informē lasītāju par to, kādus avotus autors izmantojis rakstot darbu, un palīdz lasītājam sameklēt citēto materiālu. Atsauču noformēšanas pamatā ir divi elementi – atsauce tekstā un izmantotās literatūras saraksts jeb bibliogrāfija. Pastāv dažādas sistēmas (stili) bibliogrāfijas un atsauču noformēšanai, precīzāk par tām minēts RTU ITI noformēšanas norādījumos[[20]](#footnote-20).

Noslēguma darbos tiek lietojami šādi ieteikumi izmantotās literatūrassaraksta veidošanai un atsauču noformēšanai:

* Literatūras avotus apraksta **oriģinālvalodā** un tos sarindo alfabēta secībā pēc autora uzvārda / nosaukuma, ja autors nav minēts.
* Sarakstu kārto alfabēta secībā – **vispirms** uzrāda norādes **latīņu** cilmes valodās (latviešu, angļu, vācu utt. – latīņu alfabēta secībā) un pēc tam – norādes **slāvu** cilmes valodās (slāvu alfabēta secībā).
* Literatūras avotiem jābūt **numurētiem**.
* Literatūras noformēšanas **paraugi** sniegti 3. pielikumā.
* ***Izmantotās literatūras sarakstu*** ievieto darba beigās pēc „Rezultātiem un secinājumiem” un pirms pielikumiem.
* Uz katru no literatūras sarakstā minētajiem avotiem pamattekstā jābūt atsaucei. To izdara, tekstā aiz attiecīgā materiāla **kvadrātiekavās** ierakstot literatūras kārtējo numuru, piemēram, [1], [2], [2, 6] vai [3–5].
* Literatūras avota numurs kvadrātiekavās ir neatņemama teikuma daļa, un tam jāatrodas **pirms punkta** teikuma beigās.
* Veidojot literatūras sarakstu, tos ieteicams numurēt automātiski, lai izmaiņu gadījumā saglabātos pareiza secība, kā arī tiktu pareizi atjaunotas atsauces. Veidojot atsauces – jāveido *Cross Reference* uz numuru (*numbered item*) [[21]](#footnote-21), kas dotajā brīdī norāda konkrēto literatūras avotu (*Paragraph Number*), līdz ar to, mainoties literatūras sarakstam, automātiski mainīsies arī atsauces. Piemēram: teksts no literatūras avota A [1]; teksts no žurnāla B [3].
* Ja tabulas vai attēli tiek ņemti (pat ar savu nosaukumu) no kāda literatūras avota bez izmaiņām, tad obligāti atsauce tiek pievienota tabulas vai attēla nosaukuma beigās šādā veidā: [4, 56. lpp], ja literatūras avotā ir zināma attiecīgā objekta izvietošanas lappuse, vai parasti [3], piemēram, no tīmekļa resursiem, kur lappuses var būt nezināmas.

# PIELIKUMI

Dažādus palīgmateriālus, kas neiekļaujas darba pamatsaturā, pievieno darbam kā pielikumus ar kopīgu virsrakstu **PIELIKUMI** uz atsevišķas lapas (horizontāli un vertikāli pa vidu).

Katru pielikumu **sāk ar jaunu lapu**, lapas labajā augšējā stūrī uzrādot tā kārtas numuru, piemēram: 1. pielikums, 2. pielikums utt. Zem šī uzraksta, nākamās rindiņas vidū, raksta pielikuma nosaukumu. Pielikuma nosaukumu raksta ar izceltiem burtiem (**Bold**, lieliem vai maziem) simetriski tekstam.

* Ja pielikumā ir lietderīgi pievienot kādu svešvalodā izstrādātu informatīvi normatīvo materiālu, tad to nav nepieciešams tulkot valsts valodā (drīkst pievienot oriģinālvalodā).
* Tekstā attiecīgā vietā jādod **atsauce** uz pielikumu, piemēram: 1. pielikumā pievienoti dati par...
* Ja darbam ir viens pielikums, tad kopīgo virsrakstu PIELIKUMI neraksta un pielikumam numuru nepiešķir.
* Pielikumu virsraksti ir veidoti tā, lai tie saturā parādītos kā otrā līmeņa virsraksti. Papildus tie satur arī apslēptus laukus, kas nodrošina pareizu ilustrāciju numerāciju. Līdz ar to pielikuma virsrakstus nepieciešams kopēt un mainīt tikai pielikuma nosaukumu.
* Tā kā ilustrāciju numerācija nedaudz atšķiras, tad arī šajā gadījumā jāizmanto 2. pielikuma sniegtie piemēri, kurus iespējams nokopēt un mainīt saturu. Savukārt atsauču ievietošana neatšķiras no parastās.

rezultāti un SECINĀJUMI

Izmantotās literatūras saraksts

1. Literatūras avots A
2. Publikācija B
3. Žurnāls C

pielikumi

1. pielikums. Noslēguma darba vāks

|  |
| --- |
| **RĪGAS TEHNISKĀ UNIVERSITĀTE** |
| **<NOSLĒGUMA DARBA VEIDS>**  (no saraksta: bakalaura darbs, maģistra darbs, diplomdarbs, diplomprojekts, inženierprojekts, kvalifikācijas darbs) |
| RĪGA <20….> |

2. pielikums. Pielikumā iekļauto objektu noformēšanas paraugi

Pielikumā ievietoto objektu noformējums atšķiras no pamatsaturā ievietojamajiem tikai ar „P” burtu, kas liekams attiecīgā objekta numura priekšā. Zemāk ievietotos objektus jāizmanto kā paraugs saviem objektiem (jauna objekta ievietošana veicama ar kopēšanas palīdzību, nomainot attiecīgo saturu un virsrakstus).

P2.1. attēlā ir skatāms ...



P2.1. att. Attēla nosaukums

P2.1. tabulā ir redzami dati, kas atspoguļo...

P2.1. tabula

Tabulas nosaukums

| Nr. | Vērtība | Procenti | Anuitāte |
| --- | --- | --- | --- |
| 1. | ... |  |  |
| 2. | ... |  |  |

P2.1. tabulas dati atspoguļo...

No formulu iekļaušanas pielikumā labāk izvairīties, jo jebkādas matemātiskas izteiksmes var būt izskatītas kā neatņemama daļa teorētisko algoritmu izklāstam un tiem jābūt ievietotiem darba pamatdaļā.

3. pielikums. Atsauču noformēšana

**LVS ISO 690:2010** *[adaptēts]*

**1. ZINĀTNISKĀ MONOGRĀFIJA / MĀCĪBU GRĀMATA / LEKCIJU KONSPEKTS**

Autora uzvārds**,** Iniciāļi**.,** Nākamā autora uzvārds**,** Iniciāļi**.** *Nosaukums oriģinālvalodā* *(Slīprakstā)***.** Izdevums**.** Daļa (sējums)**.** Izdošanas vieta**:** Izdevējs**,** gads**.** Lappušu skaits**.** Pieejams: doi**:**[[22]](#footnote-22)

Author**.** *Title* *(Italic[[23]](#footnote-23))***.** Edition**.** Volume (if more than one)**.** Place**:** Publisher**,** date**.** Pages**.** Available from: doi**:**

* ***Ja autoru skaits ir lielāks par četriem, tad aiz ceturtā autora jāieraksta "u.c." vai "et al."***

**Piemēri:**

Šenfelde, M. *Makroekonomika.* 4. izd. Rīga: RTU izdevniecība, 2012. 244 lpp.

Barkans, J., Zalostiba, D. *On the Global Climate Change*. Riga: RTU Publishing House, 2010. 82 p.

Djukendžijevs, J. *Cilvēka balsta-kustību un manipulāciju aparāta protezēšana*. 1. daļa, 2. sēj., 6. grām. Rīga, 2000. 164 lpp.

Platais, I., Graudiņš, P. *Gāzapgāde.* 2. daļa. Dabasgāzes gāzapgādes sistēmu izveide, ierīkošana un apkalpe. Rīga: RTU, 2008. 219 lpp.

Маталин, А.А. *Технология машиностроения*. Ленинград: Машиностроение, 1985. 512 с.

**2. PROMOCIJAS DARBS**

Autora uzvārds**,** Vārds**.** *Nosaukums oriģinālvalodā* *(slīprakstā)***.** Promocijas darbs**.** Izdošanas vieta**:** Izdevējs**,** gads**.** Lappušu skaits**.**

**Piemērs:**

Poļaka, Inese. *Klašu blīvuma struktūras izmantošana lēmumu koku klasifikatoru ansambļu evolucionārajā indukcijā*. Promocijas darbs. Rīga: RTU, 2014. 141 lpp.

**3. PROMOCIJAS DARBA KOPSAVILKUMS**

Autora uzvārds**,** Vārds**.** *Nosaukums latviešu valodā* *(slīprakstā)***.** Promocijas darba kopsavilkums**.** Izdošanas vieta**:** Izdevējs**,** gads**.** Lappušu skaits**.**

Author**.** *Title of the theses* *(Italic)***.** Summary of Promotion Thesis**.** Place**:** Publisher**,** date**.** Pages**.**

**Piemēri:**

Poļaka, Inese. *Klašu blīvuma struktūras izmantošana lēmumu koku klasifikatoru ansambļu evolucionārajā indukcijā*. Promocijas darba kopsavilkums. Rīga: RTU Izdevniecība, 2014. 38 lpp.

Poļaka, Inese. *Evolutionary Induction of Decision Tree Classifier Ensembles using Class Density Structure*. Summary of Promotion Thesis. Riga: RTU Press, 2014. 37 p.

**4. RAKSTS KONFERENČU TĒŽU KRĀJUMĀ /  
RAKSTS PILNA TEKSTA KONFERENČU RAKSTU KRĀJUMĀ**

Autora uzvārds**,** Iniciāļi**.,** Nākamā autora uzvārds**,** Iniciāļi**.** Raksta nosaukums oriģinālvalodā**. No:** *Krājuma un konferences nosaukums***,** *Valsts***,** *Pilsēta***,** *Datums* *(slīprakstā)***.** Daļa (sējums)**.** Izdošanas vieta**:** Izdevējs**,** gads**,** lappuses**.** Pieejams: doi**:**

Author**.** Title of the conference paper**.** **In:** *Title of the conference Proceedings/Abstracts book (incl. conference name)*, *Place and date of conference* *(Italic)***.** Volume (if more than one)**.** Place**:** Publisher**,** date**,** page numbers**.** Available from: doi**:**

**Piemēri:**

Buliņš, Z., Šitikovs, V. Programmatūras paplašināšana, izmantojot MySQL piedāvātās iespējas. No: *Lietišķās datorsistēmas: 52. RTU studentu zinātniskās un tehniskās konferences rakstu krājums, Latvija, Rīga, 2011. g. aprīlis.* Rīga: RTU Izdevniecība, 2011, 84.−91. lpp.

Zicans, J., Kalnins, M., Bledzki, A.K., Jablonskis, I., et al. Tensile Properties of Irradiated Binary Heterogeneous Blends Based on Poly (ethylene terephtalate) and Polyethylene. In: *Materials Engineering & BALTTRIB* − *2001: Materials of the X–th International Baltic Conference*, *Latvia, Jurmala, 27*−*28 September 2001*. Riga: RTU Publishing House, 2001, pp. 120−121.

**5. PUBLIKĀCIJA ZINĀTNISKO RAKSTU KRĀJUMĀ**

Autora uzvārds**,** Iniciāļi**.,** Nākamā autora uzvārds**,** Iniciāļi**.** Raksta nosaukums oriģinālvalodā**. No:** *Krājuma nosaukums* *(slīprakstā)***.** Daļa (sējums)**.** Izdošanas vieta**:** Izdevējs**,** gads**,** lappuses**.** Pieejams: doi**:**

Author of the article**.** Article title**. In:** *Title of the book (Italic)***.** Volume (if more than one)**.** Place**:** Publisher**,** date**,** page numbers**.** Available from: doi**:**

**Piemēri:**

Zigmunde, A., Ķestere, I. Latvijas Universitātes Pedagoģijas nodaļas pirmsākumi, studiju process, mācībspēki un studenti. No: *Pedagoģijas vēsture: 15 jautājumi: Zinātnisko rakstu krājums*. Rīga: RaKa, 2010, 176.−203. lpp.

Počs, R. Regulations and Requirements for Development of Promotion Theses in Latvia. In: *Overcoming the Hindrance in Writing Doctoral Theses: Collection of Scientific Articles*. Riga: RTU Publishing House, 2009, pp. 7−25.

**6. NODAĻA ZINĀTNISKAJĀ MONOGRĀFIJĀ**

Autora uzvārds**,** Iniciāļi**.,** Nākamā autora uzvārds**,** Iniciāļi**.** Raksta nosaukums oriģinālvalodā**. No:** Monogrāfijas autora vai redaktora uzvārds**,** Iniciāļi**.** *Monogrāfijas nosaukums* *(slīprakstā)***.** Izdevums**.** Daļa (sējums)**.** Izdošanas vieta**:** Izdevējs**,** gads**,** lappuses**.** Pieejams: doi**:**

Author of chapter**.** Chapter title**. In:** Author or editor of book**.** *Title of book* *(Italic)***.** Edition**.** Volume (if more than one)**.** Place**:** Publisher**,** date**,** page numbers**.** Available from: doi**:**

**Piemēri:**

Ketners, K. Nodokļu evolūcija. No: Krastiņš, A., Andrējeva, V., Ketners, K. *Ievads nodokļu administrēšanas specialitātē*. Rīga: RTU Izdevniecība, 2007,   
10.−16. lpp.

Merkuryev, Yu., Burinskiene, A., Merkuryeva, G. Warehouse Order Picking Process. In: Yu. Merkuryev, G. Merkuryeva, eds. *Simulation-Based Case Studies in Logistics: Education and Applied Research.* London: Springer, 2009,   
pp. 147−165. Available from: doi: 10.1007/978-1-84882-187-3\_9.

**7. PUBLIKĀCIJAS ZINĀTNISKAJOS ŽURNĀLOS**

Autora uzvārds**,** Iniciāļi**.,** Nākamā autora uzvārds**,** Iniciāļi**.** Raksta nosaukums oriģinālvalodā**.** *Žurnāla nosaukums* *(slīprakstā)***.** Izdošanas gads**,** sējums (numurs)**,** lappuses**.** Pieejams: doi**:**

Author**.** Article title**.** *Journal title* *(Italic)***.** Date**,** volume number (issue)**,** page numbers**.** Available from: doi**:**

**Piemēri:**

Krēsliņš, A., Borodiņecs, A. Dzīvojamo ēku ventilācijas sistēmas. *Latvijas Būvniecība*. 2010, Nr. 1, 38.−39. lpp.

Haritonovs, V., Smirnovs, J., Naudžuns, J. Prediction of Rutting Formation in Asphalt Concrete Pavement. *The Baltic Journal of Road and Bridge Engineering*. 2010, vol. 5, no. 1, pp. 38−42. Available from: doi: 10.3846/bjrbe.2010.05.

* ***Iespējami 2 varianti numerācijas un lpp. atveidošanā:***

***2010, vol. 5, no. 1, pp. 38***−***42.***

***2010, 5(1), 38***−***42.***

**8. PUBLIKĀCIJAS IZDEVUMA „RTU ZINĀTNISKIE RAKSTI” ŽURNĀLOS**

Autora uzvārds**,** Iniciāļi**.,** Nākamā autora uzvārds**,** Iniciāļi**.** Raksta nosaukums oriģinālvalodā**.** *Žurnāla nosaukums* *(slīprakstā)***.** Izdošanas gads**,** sējuma numurs**,** lappuses**.** Pieejams: doi**:**

Author**.** Article title**.** *Journal title* *(Italic)***.** Date**,** volume number**,** page numbers**.** Available from: doi**:**

**Piemēri:**

Berziša, S., Grabis, J. Projekta fāžu atkarīga projektu vadības informācijas sistēmu konfigurācija. *Informācijas tehnoloģija un vadības zinātne.* 2012, 15. sēj., 105.−110. lpp. Pieejams: doi:10.2478/v10313-012-0011-x (angļu valodā).

Kirshners, A., Polaka, I., Aleksejeva, L. Gastric Cancer Risk Analysis in Unhealthy Habits Data with Classification Algorithms.  Information Technology and Management Science. 2015, vol. 18, pp. 97−102. Available from: doi:10.1515/itms-2015-0015.

**9. TĪMEKĻA RESURSI**

Autora uzvārds**,** Iniciāļi**.,** Nākamā autora uzvārds**,** Iniciāļi**.** *Nosaukums oriģinālvalodā* *(slīprakstā)* [tiešsaiste]**.** Izdevējs**,** gads [skatīts 2014. g. 21. febr.]. Pieejams: <URL>

Author**.** *Title* *(Italic)* [online]**.** Publisher**,** date [viewed 21 February 2014]. Available from: <URL>

* ***Bieži nav iespējams noteikt izdošanas vietu un izdevēju, tādēļ šie elementi ir fakultatīvi.***

**Piemēri:**

Ribickis, L. *VIEDIE TĪKLI – jaunās tehnoloģijas drošai elektroapgādei* [tiešsaiste]. Dienas Bizness, 2011 [skatīts 2014. g. 21. febr.]. Pieejams: http://konferences.db.lv/wp-content/uploads/2011/12/4\_Ribickis.pdf

Janusevskis, J., Le Riche, R. *Simultaneous Kriging-Based Sampling for Optimization and Uncertainty Propagation* [online]. CCSd, 2010 [viewed   
21 February 2014]. Available from: http://hal.archives-ouvertes.fr/hal-00506957

**10. LATVIJAS PATENTI**

PATENTA PIETEICĒJS vai ĪPAŠNIEKS**.** *Nosaukums (slīprakstā)***.** Vārds [*vai* Iniciālis] Uzvārds**,** Vārds [*vai* Iniciālis] Uzvārds (izgudrotāji)**.** Int. Cl.: [Starptautiskās klasifikācijas indekss]**.** Iesniegšanas datējums [gggg-mm-dd]**.** Patenti un Preču Zīmes [Avota nosaukums]**.** Patenta numurs ar valsts kodu**.** Publicēšanas datējums [gggg-mm-dd]**.**

**Piemērs:**

RĪGAS TEHNISKĀ UNIVERSITĀTE. *Zemtemperatūras keramika ar hidrauliskām īpašībām*. Laimonis Bīdermanis, Linda Krāģe, Andris Cimmers, Lauma Lindiņa, Ingunda Šperberga, Laimons Timma (izgudrotāji). Int. Cl.: C04B33/00. Iesniegšanas datējums 2012-06-01. Patenti un Preču Zīmes. LV14562B. 2013-01-20.

Darba izpildes un novērtējuma lapa

Es, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, bakalaura studiju programmas „Informācijas tehnoloģija” III RDBI0\_\_ grupas students(e), ar savu parakstu apstiprinu, ka esmu izstrādājis(-usi) doto bakalaura darbu, kas iesniegts Rīgas Tehniskajā universitātē inženierzinātņu bakalaura grāda datorvadībā un datorzinātnē iegūšanai.

Bakalaura darbs ir izpildīts pilnīgi patstāvīgi un satur visas nepieciešamās atsauces uz darbā izmantotajiem materiāliem.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Bakalaura darbs izstrādāts \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ katedrā

Darba autors: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 201\_.g. \_\_\_. \_\_\_\_\_\_\_\_\_\_\_

Zinātniskais vadītājs: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 201\_.g. \_\_\_. \_\_\_\_\_\_\_\_\_\_\_

Bakalaura darbs pielaists aizstāvēšanai:

\_\_\_\_\_\_\_ katedras vadītājs: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 201\_.g. \_\_\_ .\_\_\_\_\_\_\_

Bakalaura darbs aizstāvēts Informācijas tehnoloģijas institūta bakalaura darbu aizstāvēšanas komisijas 201\_. gada \_\_\_.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ sēdē, protokola Nr. \_\_\_\_\_ un novērtēts ar atzīmi \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

Informācijas tehnoloģijas institūta bakalaura darbu aizstāvēšanas komisijas sekretāre: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Dr.math. V.Minkēviča/

1. Null – Tukšuma apraksts programmēsāna valodās [↑](#footnote-ref-1)
2. Backward compatibility – programmu un interfeisu eksistēšana jaunākos datoru modeļos, kutas tika izmantotas vecākos modeļos, kas atvieglina programmu(vai cilvēku) darba turpināšanu bez drastiskas pārmācīšanās. [↑](#footnote-ref-2)
3. JVM – Java Virtual Machine veic bait kodu Java, kurš tika kompilēts no Java valodas koda. JVM var arī veikt darbības ar programmām, kuras tiek rakstītas citās programmēšanas valodās. [↑](#footnote-ref-3)
4. SDK – software development kit rīku komplekts, kurš ļauj speciālistiem veidot aplikācijas noteiktam programmu veidam [↑](#footnote-ref-4)
5. Backend – Datora programmas vai sistēmas daļa, kura strādā ar datiem un kurai lietotājam nav piekļuves. [↑](#footnote-ref-5)
6. Frontend – Datora sistēmas daļa vai aplikācija, kurai lietotājam ir tieša piekļuve. [↑](#footnote-ref-6)
7. Фреймворк – programmas platforma, kura nosaka programmas sistēmas struktūru; programmatūra, kura atvieglo dažādu liela projekta komponenšu izstrādi un apvienošanu. [↑](#footnote-ref-7)
8. Open-Source Software - is [computer software](https://en.wikipedia.org/wiki/Computer_software) with its [source code](https://en.wikipedia.org/wiki/Source_code) made available with a [license](https://en.wikipedia.org/wiki/Open-source_license) in which the [copyright](https://en.wikipedia.org/wiki/Copyright) holder provides the rights to study, change, and distribute the software to anyone and for any purpose. Open-source software may be developed in a [collaborative public manner](https://en.wikipedia.org/wiki/Collaborative_software_development_model). [↑](#footnote-ref-8)
9. Pattern matching -  is the act of checking a given sequence of tokens for the presence of the constituents of some [pattern](https://en.wikipedia.org/wiki/Pattern). [↑](#footnote-ref-9)
10. Type inference - в программировании возможность компилятора самому логически вывести тип значения у выражения. [↑](#footnote-ref-10)
11. Persistent collections -  is a [collection](https://en.wikipedia.org/wiki/Data_structure) that always preserves the previous version of itself when it is modified. [↑](#footnote-ref-11)
12. Tuple – A tuple is a class that can contain a miscellaneous collection of elements. [↑](#footnote-ref-12)
13. Runtime - the period during which a computer program is executing [↑](#footnote-ref-13)
14. Lisp – Язык программирования появившийся в 1958 году, научный язык в большинстве своем используемый для разработки ИИ. [↑](#footnote-ref-14)
15. IDE – Integrated Development Environment is a [software application](https://en.wikipedia.org/wiki/Application_software) that provides comprehensive facilities to [computer programmers](https://en.wikipedia.org/wiki/Computer_programmer) for [software development](https://en.wikipedia.org/wiki/Software_development). [↑](#footnote-ref-15)
16. Метапрограммирование -  вид [программирования](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), связанный с созданием [программ](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0), которые порождают другие программы как результат своей работы. в частности, на стадии [компиляции](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) их [исходного кода](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4), либо программ, которые меняют себя во время выполнения [↑](#footnote-ref-16)
17. First-class Functions -  В частности, это означает, что язык поддерживает передачу функций в качестве аргументов другим функциям, возврат их как результат других функций, присваивание их переменным или сохранение в структурах данных. [↑](#footnote-ref-17)
18. IBM Watson – суперкомпьютер оснащенный вопросно-ответной системой искусственного интеллекта. Основная задача Watson – понимать вопросы на людском языке, и находить на них ответы в базе данных. [↑](#footnote-ref-18)
19. Fonta „*Times New Roman*” pielietošana attiecās ne tikai uz pašu tekstu, bet arī uz visiem elementiem – tabulām, attēliem un formulām. [↑](#footnote-ref-19)
20. Norādījumi noslēguma darba noformēšanai studiju programmā „Informācijas tehnoloģija” / Izstr. L. Aleksejeva, J. Grabis, J. Merkurjevs, I. Upīte. Rīga: RTU, 2016. 20 lpp. [↑](#footnote-ref-20)
21. Lai saīsinātu numurēto ierakstu sarakstu un atvieglotu iespēju atsaukties uz literatūras avotiem, ***ieteicams izveidot iezīmes nosaukumu arī literatūras avotam***, analoģiski attēlu, tabulu vai formulu variantam. [↑](#footnote-ref-21)
22. Krāsas lietotas tikai, lai izceltu būtiskākos atribūtus; tās nav jāizmanto bibliogrāfijas noformēšanā. [↑](#footnote-ref-22)
23. *Slīprakstā* visos piemēros tiek atzīmēti ***augstākā līmeņa*** elementu nosaukumi – grāmatas, zinātniskās monogrāfijas, promocijas darba vai promocijas darba kopsavilkuma, rakstu krājuma, žurnāla, utt. nosaukumi. [↑](#footnote-ref-23)