Latvijas Republikas izglītības un zinātnes ministrija

Daugavpils Tehnoloģiju un tūrisma tehnikums

**PRAKTISKAIS DARBS**

Programmēšanas tehniķis

specialitāte

1. projekta darbs

temats

Testēšanas sistēmas izveidošana

PASKAIDROJOŠAIS RAKSTS

**DTTT.P.D.00XX.0XXPR**

Profesijas kods 33 481 031 Programmēšanas nodaļa

PR-21. grupa

Izveidoja Artjoms Sidorevics

paraksts V.,Uzvārds

Pārbaudīja L.Stašanova

paraksts V.,Uzvārds

2024./2025. m.g.

**Saturs**

[Ievads 2](#_Toc196807867)

[1. Uzdevuma nostādne 4](#_Toc196807868)

[1.1. Sistēmas modelis 4](#_Toc196807869)

[1.1.1. Priekšmetiskās jomas informācijas modelis 4](#_Toc196807870)

[1.1.2. Sistēmas apkārtnes shēma 4](#_Toc196807871)

[1.1.3. Viedokļu analīze 5](#_Toc196807872)

[1.1.4. Datu modelēšana 6](#_Toc196807873)

[1.1.5. Datu vārdnīca 7](#_Toc196807874)

[1.2. Sistēmas evolūcija 7](#_Toc196807875)

[1.3. Sistēmas funkcionālās prasības 7](#_Toc196807876)

[1.4. Sistēmas nefunkcionālās prasības 10](#_Toc196807877)

[2. Priekšmetiskās jomas klašu sistēmas izstrāde 12](#_Toc196807878)

[2.1. Klašu diagramma 12](#_Toc196807879)

[2.2. Klašu realizēšana Java valodā 14](#_Toc196807880)

[3. Testa programmas izstrāde 15](#_Toc196807881)

[3.1. Testēšanas metodikas 15](#_Toc196807882)

[3.2. Testēšanas programmas struktūra 15](#_Toc196807883)

[3.3. Testēšanas rezultāti 16](#_Toc196807884)

[4. Lietotāja rokasgrāmata 17](#_Toc196807885)

[4.1. Lietotāja instrukcija 17](#_Toc196807886)

[4.2. Palīdzības sistēma (Help) 17](#_Toc196807887)

[Secinājumi 17](#_Toc196807888)

[Izmantotie avoti 18](#_Toc196807889)

[Pielikums A. Klašu sistēma 19](#_Toc196807890)

[Pielikums B. testa programmas kods 20](#_Toc196807891)

# Ievads

Projekta darbā ir aprakstīta testēšanas programma, kura izpilda sekojošas funkcijas:

* lietotāju reģistrēšana
* datu ievade
* datu saglabāšana
* datu rediģēšana
* testa pildīšana
* rezultātu aprēķināšana

Sistēma nodrošina audzēkņa testēšanu par noteiktu tēmu/vielu, tas ir domāts, lai skolotājs varētu viegli izlikt atzīmi skolēnam digitālā veidā attālināti vai klātienē. Šajā gadījuma pirmā testa tēma ir ” Metal Gear Solid” video spēles sēriju.

Šī lietotne var atvieglo skolotāja darbu, automatizējot pareizo atbildi saskaitīšanu.

# Uzdevuma nostādne

## Sistēmas modelis

### Priekšmetiskās jomas informācijas modelis

Apskatāmā sistēmā tika izdalīti objekti: Datu bāze (Fails ar atbildēm), Tests, Rezultāti, Administrators / Skolotājs, Lietotājs. Šie objekti un saites starp tiem ir paradīti (1.1. att.).

Datu bāze

Rezultāti

Tests

Lietotājs

Administrātors/ Skolotājs

att 1.1 Vienkāršots testēšanas sistēmas modelis

Saišu nozīme vienkāršotā nodaļas modelī:

* Skolotājs ir testa veidotājs un pārbaudītājs, kuram ir pieeja pie atbildēm, testam un lietotāja rezultātiem;
* Datu bāze satur failus ar jautājumiem un ar pariezām atbildēm;
* Lietotājs ir testa pildītājs, kas beigās var apskatīt savus rezultātus, nepieciešamības gadījumā viņš varēs sazināties ar skolotāju;
* Tests ir instruments, kas palīdzēs pārbaudīt lietotāju zināšanas un novērtēt to ar atzīmi;
* Rezultāti parāda atzīmi, pareizās un nepareizās atbildes skaitliski un procentuāli.

### Sistēmas apkārtnes shēma

Dati, kā atbildes, ko ievada audzēknis tiek saglabāti failā. Tomēr dažreiz atbildes netiek automātiski novērtētas un skolotājam ir iespēja manuāli novērtēt iesniegto atbildi. Saskarni ar lietotājiem nodrošinās interfeisa modulis (1.2. att.).

Lietotajs

Lietotāja saskarne

Sistēma

Datu bāze

Lietotāja dati un rezultāti

att 1.2 Sistēmas apkārtnes modeļa piemērs

### Viedokļu analīze

Savāktie un identificētie viedokļi par sistēmu ir attēloti ar burbuļdiagrammas palīdzību (1.3. att.), tas faktiski ir viedokļu kopums, kur katra viedokļa nosaukums ir ierakstīts atsevišķa elipsē.

Nefunkcionālie viedokli

Funkcionālie viedokli

Viedokli par lietotaju

Datu viedokli

att 1.3 Viedokļu burbuļu diagramma

Otrais etaps viedokļu analīzē ir viedokļu klasifikācija, kad viedokļi ir sagrupēti grupās (1.4. att.).

att 1.4 Viedokļu hierarhiskā struktūra

### Datu modelēšana

Datu modelēšanai ir tiek izmantots ER modelis. Testēšanas sistēmas ER modelis (1.5. att.) ietver datu entītiju kopumu un relāciju kopumu starp datu entītijām, pie kam ar šo modeli var attēlot dažāda tipa relācijas starp datu elementiem.

Audzēkni

Pilda

Tests

Satur

Jautājums

Skolotājs

Izveido

att 1.5 Vienkāršots testēšanas sistēmas ER modelis

### Datu vārdnīca

|  |  |  |
| --- | --- | --- |
| Datu tips | Nosaukums | Apraksts |
| String | **name** | Audzēknis ievada savu vārdu un uzvārdu |
| String | **questionText** | Neliels teksta apgabals, kas apraksta jautājumu |
| String[] | **answersText** | Atbilžu variantu masīvs. Tiek piedāvāti vairāki atbilžu varianti, kur ir jāizvēlas viens pareizs |
| int | **correctAnswer** | Aile, kas nosaka, vai tika izvēlēts pareizs atbildes variants |
| float | **result** | Audzēkņa saņemto punktu skaits (0-100) |
| int | **mark** | Audzēkņa atzīme (0-10) |

## Sistēmas evolūcija

Testēšanas sistēma ir neatkarīga un pašpietiekama programma. Produktam nav paredzētas

vairākas sastāvdaļas.

Programmai ir trīs ārējās saskarnes, kuras var apskatīt 1.4. nodaļā – viena paredzēta reģistrēšanai sistēmā, otrā – ielogošanai sistēmā un trešā – testa pildīšanai. Grafisko lietotāja saskarni var pilnveidot, papildinot to ar dažādām krasu un noformējuma shēmām.

Testēšanas sistēmā paredzēti divu lietotāju veidi: administrators un lietotājs, kurš pilda testu. Nākotnē plānots pievienot vēl vienu lietotāju grupu ar funkciju veidot jaunus testus un pārbaudīt izpildītos testus, līdz ar to lietotājiem būs divas lomas: audzēknis, kurš pilda testu un skolotājs, kurš veido un pārbauda testus. Administratoram plānots realizēt funkciju mainīt lietotāja vārdu, uzvārdu, lietotājvārdu vai paroli, ka arī pievienot un dzēst lietotājus un mainīt vīnu lomas.

Programma satur vienu testu, kurā ir 10 jautājumi, tomēr nākotnē sistēmu var pilnveidot, pievienojot jaunas testa tēmas, kļūdu uzrādīšanu un kļūdu skaidrojumu pēc testa izpildes, lietotāju reitingu, lietotāju datu izvadi lietotāja izvēlnē vai administratora izvēlnē.

Sistēmas dažiem datu tipiem tiks izmantoti dati, kuri tiek saglabāti teksta failā, nākotnē programmu var pilnveidot darbam ar datu bāzi (datu bāzes savienojums, testa jautājumi un atbildes uz tiem u.c.)

## Sistēmas funkcionālās prasības

**1. Funkcija “Reģistrēties sistēmā”**

**ID: T1**

**Ievads**: Ļauj ievadīt nepieciešamus datus, lai izveidotu savu kontu. Visi ievadīti dati tiek saglabāti failā (datu bāzē).

**Ievade:**

1) Lietotāja vārds

2) Lietotāja uzvārds

3) Lietotāja lietotājvārds

4) Lietotāja parole

5) Ievadītās paroles apstiprināšana.

**Apstrāde:**

1) Lietotāja vārda saglabāšana failā.

3) Lietotāja lietotājvārds saglabāšana failā.

4) Lietotāja paroles saglabāšana failā.

Ja ievadītā parole ir nepareiza, tad tiek norakstīts kļūdiņa teksts reģistrēšanas logā.

**Izvade:** JLabel teksts, kurā uzrakstīts “Lietotājs tiek pievienots!”

**2. Funkcija** “ **Ielogoties sistēmā**”

**ID: T2**

**Ievads:** Ļauj ievadīt nepieciešamus datus, lai ieiet sava kontā.

**Ievade:**

1) Lietotāja lietotājvārds

2) Lietotāja parole

**Apstrāde:**

1) Lietotāja lietotājvārda pārbaude un atrašana DB.

2) Lietotāja paroles pārbaude un atrašana DB

Ja ievadītā parole vai/un lietotājvārds ir nepareiza, tad tiek atvērts kļūdas dialoglodziņš

**Izvade:** JLabel teksts, kurā uzrakstīts lietotāja lietotājvārds.

**3. Funkcija “Pildīt testu”**

**ID: T3**

**Ievads**: Ļauj lietotājam sākt testa izpildi.

**Ievade**: Lietotājs noklikšķina uz pogu “Pildīt testu”.

**Apstrāde**: Ja tests ir pieejams izpildei, tad pēc testa izvēles tiek parādīts dialoglodziņš ar tekstu

"Tests ir pieejams! Vai vēlaties izpildīt testu?". Lietotājam ir divi varianti atbildēm "Jā" vai "Nē". Ja tests nav pieejams izpildei, tad pēc testa izvēles tiek parādīts dialoglodziņš ar tekstu "Tests nav pieejams!"

**Izvade**: Lietotājā ekrānā tiek parādīts dialoglodziņš ar uzrakstu "Veiksmi testā!"

4**. Funkcija “Atbildēt uz testa jautājumu”**

**ID: T4**

**Ievads**: Ļauj ievadīt atbildi uz testa jautājumus.

**Ievade**: Lietotājs izvēlas 1 no 4 piedāvātajām atbildēm uz uzdoto jautājumu.

**Apstrāde**:

Sistēma pārbauda atbildi uz jautājumu. Pēc katras pareizās atbildes sistēma pievieno 1.

**Izvade**: Pēc izvēlētās atbildes lietotājs noklikšķina uz pogas "Nākamais", pēc kura tiek parādīts nākamā izvēlne ar testa jautājumiem

5. **Funkcija “Parādīt rezultātu”**

**ID: T5**

**Ievads**: Ļauj uzzināt nokārtotā testa rezultātus.

**Ievade**: Lietotājs atbilde uz pēdējo jautājumu testā.

**Apstrāde**:

Sistēma saskaita pareizo atbilžu skaitu.

**Izvade**: Tiek atvērts dialoglodziņš, kurā tiek rakstīts pareizo atbilžu skaits.

## Sistēmas nefunkcionālās prasības

1. Prasības produktam:
   1. Lietotāju saskarne ar sistēmu notiek latviešu valodā.
   2. Produktam nav paredzētas vairākas sastāvdaļas. Produktam ir trīs ārējās saskarnes: reģistrēšanai sistēmā, ielogošanai sistēmā, testa pildīšanai.
   3. Programma ir paredzēta vienam lietotājam, kurš saprot latviešu valodu. Lietotāja vecuma vai izglītības ierobežojumi nav paredzēti.
2. Ārējās saskarnes prasības
   1. Lietotāja saskarne: produkts paredz vienotu lietotāja saskarni.
   2. Ekrāna formāti: minimālais ekrāna izmērs ir 10” ar minimālo izšķirtspēju –800x600 punkti.
   3. Lietotāja saskarne “Lietotāja reģistrēšanas saskarne”. Saskarnes struktūra: skat. 1.6. att.

Vārds:

Uzvārds:

Lietotājvārds:

Parole:

Parole (atkārtoti):

Reģistrēties

att 1.6 Lietotāja reģistrēšanas saskarne

**Saskarnes elementi:**

1. teksta lauks “Vārds”
2. teksta lauks “Uzvārds”
3. teksta lauks “Lietotājvārds”
4. teksta lauks ‘Parole”
5. teksta lauks “Parole (atkārtoti)”
6. poga “Reģistrēties”
   1. Lietotāja saskarne “Lietotāja ielogošanas saskarne”. Saskarnes struktūra: skat. 1.7. att.

Lietotājvārds:

Parole:

Reģistrēties

Ielogoties

att 1.7 Lietotāja ielogošanas saskarne

**Saskarnes elementi:**

1. teksta lauks “Lietotājvārds”

2. teksta lauks ‘Parole”

3. poga “Ielogoties”

4. poga “Reģistrēties”

* 1. Lietotāja saskarne “Testa izpilde”. Saskarnes struktūra: skat. 1.8. att.

Izveleties testu:

Tests 1

a)

„Jautājuma teksts”

1. atbilde

2. atbilde

3. atbilde

4. atbilde

b)

Tālāk

Tests 2

Tests 3

Tests 4

att 1.8. Testa izpildes saskarne: testa izvēle/sakums; b) atbilde uz jautājumu/-iem ; c) rezultāti

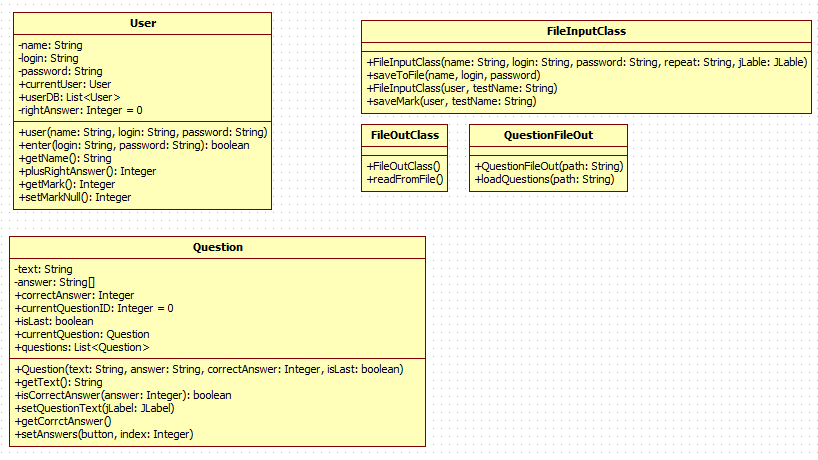
**Saskarnes elementi**

* + 1. A- testa izvēles logs
    2. B- testa logs un atbildes varianti

# Priekšmetiskās jomas klašu sistēmas izstrāde

## Klašu diagramma

Klašu diagrammā ir realizēts vispārīgs klašu apraksts – klašu hierarhijas vispārējā struktūru, to sadarbība, atribūti, metodes, saskarnes un to savstarpējās attiecības, izmantojot UML valodu.



att 2.1 Testēšanas sistēmas klašu diagramma UML valodā

Balstoties uz izstrādāto sistēmas modeli var veikt sekojošu klašu iedalījumu (2.1. att.):

1. Klase **User** – ir atbildīga par darbu ar lietotājiem.
   * User (String name, String login, String password) – konstruktors, kurš izveido jaunu lietotāju ar norādītiem datiem;
   * enter (String login, String password) – metode, kura nodrošina lietotāju pieslēgšanu sistēmai:
   * getName()- parada lietotāja vardu:
   * plusRightAnswer()- pievieno lietotajam pareizo atbildu skaiti:
   * getMark()- izvada lietotāja atzīmi:
2. Klase **Question** – ir atbildīga par darbu ar testa jautājumiem, to izvadi un ievadītas atbildes pārbaudi
   * Question(String text, String[] answers, int correctAnswer, boolean isLast)- konstruktors, kas ļauj izveidot jautājumu un pareizo atbildi:
   * setQuestionText(JLabel jlable)- pievieno jautājuma tekstu pie testa logā:
   * setAnswers(JRadioButton button, int index)- pievieno atbildes tekstu pie atbildes variantiem:
   * getText(): String- iegūst jautājuma tekstu:
   * isCorrect(in answer:String): boolean- pārbauda, vai atbilde ir pareiza:
   * getCorrctAnswer()- iegūst pareizo atbildi uz jautājumu:
   * isCorrectAnswer(int answer)- pārbauda pareizo jautājumu:
   * isLastQuestion()- pārbaude vai jautājums ir pēdējais:
3. FileInputClass- ir atbildigs par lietotaja saglabāšanu DB
   * FileInputClass(String name, String login, String password, String repeat, JLabel jLable)- konstruktors, kas saglaba dati pie DB:
   * saveToFile(String name, String login, String password)- saglaba vardu, lietotājvārdu un paroli failā:
   * saveMark(User user)- saglaba lietotaja vardu un atzimi atsiviška failā:
4. FileOutClass- ir atbildīgs par lietotāju datu izlasīšanu no DB:
   * FileOutClass()- konstruktors, kas izjaukt visi metodī:
   * readFromFile()- lāsa lietotaja dati no DB:
5. QuestionFileOut- ir atbildīgs par jautājumu datu nolasīšanu:
   * QuestionFileOut(String path)- konstruktors,kas izjaukt metodi:
   * loadQuestions(String path)- metode, kas lasa visus jautājumus failā un saglaba tos ArrayList:

## Klašu realizēšana Java valodā

Izstrādātās klašu diagrammas (2.1. att.) realizēšanai **NetBeans** vidē tika izveidots projekts ar nosaukumu **TestingSystem**, kurā izstrādātās klases tika aprakstītas **Java** valodā. Šeit tiek piedāvāts klašu vispārīgā apraksta (klašu deklarācijas) programmas kods un metožu apraksti. Pilns avota programmas kods ar klašu metožu definīcijām ir ievietots pielikumā “[Pielikums A. Klašu sistēma](#_Pielikums_A._Klašu)”, bet programmas kods, kas realizē šīs klašu sistēmas testēšanu (klase ar grafisko lietotāja saskarni) ir ievietots pielikumā “[Pielikums B. testa programmas kods](#_Pielikums_B._testa)”.

class User{

private String name;

private String login;

private String password;

private void User(String name, String login, String password){

}

private boolean enter(String login, String password){

return true;

}

}

# Testa programmas izstrāde

## Testēšanas metodikas

Pirms priekšmetiskās jomas klašu sistēmas testēšanas programmas izstrādes, ir jānosaka testēšanas metodika.

Ir jāatšķir koda testēšana un atkļūdošana. Atkļūdošanu veic programmētājs, izmantojot iebūvētos izstrādes vides rīkus un balstoties uz pieredzi programmas koda rakstīšanā. Būtībā runa iet par sintaktisko un semantisko kļūdu identificēšanu programmas tekstā.

Testēšana ir process, kurā nepieciešams plānot un īstenot vairākas sākotnējās procedūras, no kurām galvenā ir testa piemēru kopas izstrāde, kas veido testa plānu. Testpiemēri vairumā gadījumu ir balstīti uz sistēmas funkcionālajām prasībām un var ietekmēt dažādus attīstības līmeņus (vienības testēšana, integrācijas testēšana, sistēmas testēšana).

Projekta darba laikā nepieciešams veikt vienkāršotu vienības testēšanas versiju, kas realizē visu izstrādāto klašu metožu testēšanu. Ar testēšanas metodiku projekta darbā pieņem dažādu klases metožu izsaukšanas iespēju sarakstu ar gaidāmajiem rezultātiem un šo testēšanas izsaukumu izpildes kārtību.

## Testēšanas programmas struktūra

Testēšanas programmas projektēšana un programmatūras izstrāde ir ļoti līdzīga iepriekš aprakstītajam klašu sistēmas izstrādes procesam.

Jāatzīmē tie momenti un prasības, kas pastāv, izstrādājot testēšanas programmu.

* + Testēšanas programmai ir loga interfeiss, kas ietver sevī izvēlnes un interaktīvā interfeisa rīkus, ar kuru palīdzību var pārbaudīt izveidoto klašu sistēmu.
  + Interfeisa elementu (pogas, saraksti, ievades lauki, dialoglodziņi, grafiskie elementi utt.) komplekts tiek noteikts, pamatojoties uz iepriekšējā sadaļā aprakstīto testēšanas metodiku.
  + Testēšanas programmas interfeisa kvalitātei jābūt vienkāršai lietošanā un interaktīvai. Galvenā prasība ir spēja realizēt testēšanas metodes.

1. Ielogošanas procesa apraksts:
   * Ielogošanas process paredzēts reģistrēto lietotāju pievienošanai sistēmā. Lai lietotājs varētu piekļūt testam viņam pēc ir nepieciešams autorizēties. Ielogošanas procesā lietotājs ievada iepriekš reģistrētos datus – lietotājvārdu un paroli (3.1. att. ). Ja lietotāja 15 dati sakrīt ar validācijas datiem, tad lietotājam tiek atvērts programmas galvenais logs (3.x. att.).
   * Ielogošanas procesā ir iespējamas vairākas kļūdas – neaizpildīti lauki, neprecīzi ievadīti lietotāja dati.

Login Form

Login

Password

Cancel

Login

att 3.1 Testēšanas sistēmas ielogošanas forma

1. Reģistrācijas procesa apraksts:

Lietotajs ievieto savu vardu, uzvardu, loginu, paroli un atkarto paroli, ja paroli ir sakrit, tad dati pievienojas DB

## Testēšanas rezultāti

Testēšanas laika nekādas lielas problēmas nebija atrasti.

# Lietotāja rokasgrāmata

## Lietotāja instrukcija

1. Lietotājs palaiž programmu.
2. Piesakās sistēmā, izmantojot savu lietotājvārdu un paroli, ja lietotājam nav sava konta, tad viņš var reģistrēt to.
   * 1. Reģistrācijas logā lietotājs aizpilda visus nepieciešamus datus un noklikšķina uz pogas Reģistrēties.
3. Lietotāja izvēlnē ir 2 pogas.
   1. Poga «Pildīt testu» ļauj sākt pildīt testu.
      1. Pēc testa izpildīšanas tiek paradīts testa izpildes rezultāti.
   2. Poga „Atpakaļ”
      1. Aizvērt testa logu un izlogos no programma

## Palīdzības sistēma (Help)

Lai sākt pildīt testu, Jums ir nepieciešams ieiet savā kontā, ievadot lietotājvārdu un paroli. Programmā jau ir palīdzības logi

# Secinājumi

Tika izstrādāts projekts «Testēšanas sistēma» NetBeans vidē. Tika izpildītas visas programmas bāzes prasības. Projekta izpildīšanas laikā radās dažādas problēmas, kas neļāva veikt sākotnējo vēlamu rezultātu, bet tajā pašā laikā bija interesanti uzzināt jaunu informāciju, kas noderēs turpmāk. Ceru, jo tālāk, jo labāk!

# Izmantotie avoti

1. **I.Dortiņa un L.Stašanova.** 3.1. DATU STRUKTŪRAS UN ALGORITMI. [Tiešsaiste] https://skolo.lv/pluginfile.php/90483437/mod\_resource/content/5/3.1.Datu\_strukturas\_un\_algoritmi\_2024-1.pdf.

2. **I.Dortiņa un Dortiņš, A.** 1.2.3. JAVA IEVADES/IZVADES SISTĒMA. [Tiešsaiste] https://skolo.lv/pluginfile.php/86388487/mod\_resource/content/3/1.2.3\_Java\_faili\_2024.pdf.

3. **I.Dortiņa un L.Stašanova.** 2.1.3. JAVA PAKOTNES UN INTERFEISI. [Tiešsaiste] https://skolo.lv/pluginfile.php/89797907/mod\_resource/content/3/2.1.3\_Java\_pakotnes\_interfeisi\_2024.pdf.

4. —. 2.1.2. JAVA MANTOŠANA UN METOŽU PĀRDEFINĒŠANA. [Tiešsaiste] https://skolo.lv/pluginfile.php/89012925/mod\_resource/content/6/2.1.2\_Java\_Mantosana\_met\_pardefinesana\_2024.pdf.

5. **I.Dortiņa un Stašanova, L.** 4.1.1. JAVA GRAFISKĀ LIETOTĀJA SASKARNE. [Tiešsaiste] https://skolo.lv/pluginfile.php/90918436/mod\_resource/content/1/4.1.1.Java\_GUI\_2025.pdf.

6. —. 2.1.1.2. JAVA. METODES UN KONSTRUKTORI. [Tiešsaiste] https://skolo.lv/pluginfile.php/88005418/mod\_resource/content/1/2.1.1.2\_Java\_Metodes\_konstruktori\_2024.pdf.

7. **Dortiņa, I. un A. Dortiņš, A.** 1.1.2. Ievads Java. Java pamati. [Tiešsaiste] 2024. gada 14. 2. [Citēts: 2023. gada 16. 2.] https://skolo.lv/mod/resource/view.php?id=82126301.

8. **Dortiņa, I. un Dortiņš, A.** 2.1.1.1. JAVA OOP KONCEPCIJA. [Tiešsaiste] https://skolo.lv/pluginfile.php/86576110/mod\_resource/content/5/2.1.1.1\_Java\_OOP\_2024.pdf.

9. **s I.Dortiņa, L. Stašanova.** 4.2. JAVA VIENLAICĪGA UN SADALĪTA PROGRAMMĒŠANA. [Tiešsaiste] https://skolo.lv/pluginfile.php/91087357/mod\_resource/content/9/4.2.Java\_pavedieni\_2025.pdf.

# Pielikums A. Klašu sistēma

**QUESTION**

package classPackage;

import java.util.ArrayList;

import java.util.List;

import javax.swing.JLabel;

import javax.swing.JRadioButton;

/\*\*

\*

\* @author Artjoms Sidorevics

\*/

public class Question {

private final boolean isLast;

public static Question currentQuestion;

public static int currentQuestionID=0;

public static List<Question> questions = new ArrayList();

private final String text;

private final String[] answers;

private final int correctAnswer;

public Question(String text, String[] answers, int correctAnswer, boolean isLast){

this.text = text;

this.answers = answers;

this.correctAnswer = correctAnswer;

this.isLast = isLast;

}

public void setQuestionText(JLabel jlable){

String t = this.text;

jlable.setText(t);

}

public void setAnswers(JRadioButton button, int index){

String a = answers[index];

button.setText(a);

}

public String getText(){

return text;

}

public int getCorrctAnswer(){

return this.correctAnswer;

}

public boolean isCorrectAnswer(int answer){

return correctAnswer == answer;

}

public boolean isLastQuestion() {

return currentQuestionID == questions.size();

}

}

**User**

package classPackage;

import java.util.ArrayList;

import java.util.List;

/\*\*

\*

\* @author Artjoms Sidorevics

\*/

public class User {

private final String name;

private final String login;

private final String password;

public static User currentUser;

private static int rightAnswers = 0;

public static List<User> userDB = new ArrayList();

public User(String name, String login, String password){

this.name = name;

this.login = login;

this.password = password;

}

public boolean enter(String login, String password){

return login.equals(this.login) && password.equals(this.password);

}

public String getName(){

return this.name;

}

public int plusRightAnswer(){

return this.rightAnswers++;

}

public int setMarkNull(){

return this.rightAnswers=0;

}

public int getMark(){

return this.rightAnswers;

}

}

**FileInputClass**

package classPackage;

import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;

import javax.swing.JLabel;

/\*\*

\*

\* @author Artjoms Sidorevics

\*/

public class FileInputClass {

public FileInputClass(String name, String login, String password, String repeat,JLabel jLable){

if(password.equals(repeat)){

saveToFile(name, login, password);

jLable.setText("Lietotajs tiek pievienots");

}

else

jLable.setText("Paroles nesakrit");

}

public FileInputClass(User user){

saveMark(user);

}

private void saveToFile(String name, String login, String password){

try {

BufferedWriter wr = new BufferedWriter(new FileWriter("UserDB.txt",true));

wr.write(name+"/"+login+"/"+password);

wr.newLine();

wr.close();

}catch (IOException ex)

{}

}

public void saveMark(User user){

try {

BufferedWriter wr = new BufferedWriter(new FileWriter("UserDBMark.txt"));

wr.write(user.getName()+" :Mark: "+user.getMark());

wr.close();

}catch (IOException ex)

{}

}

}

**FileOutClass**

package classPackage;

import java.io.BufferedReader;

import java.io.FileInputStream;

import java.io.IOException;

import java.io.InputStreamReader;

/\*\*

\*

\* @author Artjoms Sidorevics

\*/

public class FileOutClass {

public FileOutClass(){

readFromFile();

}

private void readFromFile(){

try{

BufferedReader fileout = new BufferedReader(new InputStreamReader(new FileInputStream("UserDB.txt")));

String strLine;

while ((strLine = fileout.readLine()) != null)

{

String[] dati = strLine.split("/");

String name = dati[0];

String login = dati[1];

String password = dati[2];

User.userDB.add(new User(name, login, password));

}

fileout.close();

}catch (IOException ex)

{}

}

}

**QuestionFileOut**

package classPackage;

import java.io.BufferedReader;

import java.io.FileInputStream;

import java.io.IOException;

import java.io.InputStreamReader;

/\*\*

\*

\* @author Artjoms Sidorevics

\*/

public class QuestionFileOut {

public QuestionFileOut(String path){

loadQuestions(path);

}

private void loadQuestions(String path) {

try {

Question.questions.clear();

BufferedReader fileout = new BufferedReader(new InputStreamReader(new FileInputStream(path)));

String line;

while ((line = fileout.readLine()) != null) {

String[] parts = line.split(":");

if (parts.length < 6) continue;

String text = parts[0];

String[] answers = { parts[1], parts[2], parts[3], parts[4] };

int correct = Integer.parseInt(parts[5]);

boolean isLast = !fileout.ready();

Question.questions.add(new Question(text, answers, correct, isLast));

}

fileout.close();

} catch (IOException ex)

{}

}

}

# Pielikums B. testa programmas kods

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

System.exit(0);

}

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

jRegistration.setBounds(400,300, 495, 470);

jRegistration.setVisible(true);

}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

String login = jLoginLoginField.getText();

String password = jLoginPasswordField.getText();

FileOutClass out = new FileOutClass();

for(User u: User.userDB){

if(u.enter(login, password)){

User.currentUser = u;

jLabel15.setText(login);

jQuizChoice.setVisible(true);

jQuizChoice.setBounds(400,300, 495, 470);

}

else{

jSysSignal.setText("Wrong Login or Password");

}

}

}

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

String name = jTextName.getText();

String login = jTextLogin.getText();

String password = jTextPassword.getText();

String password\_repeat = jTextPasswordRepeat.getText();

FileInputClass file = new FileInputClass(name, login, password, password\_repeat, jLabelErr);

}

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {

jRegistration.setVisible(false);

jTextName.setText("");

jTextLogin.setText("");

jTextPassword.setText("");

jTextLastName.setText("");

jTextPasswordRepeat.setText("");

jLabelErr.setText("");

}

private void jTestOneActionPerformed(java.awt.event.ActionEvent evt) {

jMGS1Test.setVisible(true);

jMGS1Test.setBounds(400,300, 495, 470);

QuestionFileOut test = new QuestionFileOut("MGS1QuestionDB.txt");

testOneBGroup.add(jTest1AnswerOne);

testOneBGroup.add(jTest1AnswerTwo);

testOneBGroup.add(jTest1AnswerThree);

testOneBGroup.add(jTest1AnswerFour);

for(Question q: Question.questions){

Question.currentQuestion = q;

q.setQuestionText(jTestOneQuestionText);

q.setAnswers(jTest1AnswerOne, 0);

q.setAnswers(jTest1AnswerTwo, 1);

q.setAnswers(jTest1AnswerThree, 2);

q.setAnswers(jTest1AnswerFour, 3);

q.getCorrctAnswer();

}

}

private void jTestOneNextButtonActionPerformed(java.awt.event.ActionEvent evt) {

int answerTestOne = 0;

if (!(jTest1AnswerOne.isSelected() ||

jTest1AnswerTwo.isSelected() ||

jTest1AnswerThree.isSelected() ||

jTest1AnswerFour.isSelected()))

{

return;

}

if (jTest1AnswerOne.isSelected()) {

answerTestOne = 1;

Question.currentQuestionID++;

}

else if (jTest1AnswerTwo.isSelected()) {

answerTestOne = 2;

Question.currentQuestionID++;

}

else if (jTest1AnswerThree.isSelected()) {

answerTestOne = 3;

Question.currentQuestionID++;

}

else if (jTest1AnswerFour.isSelected()) {

answerTestOne = 4;

Question.currentQuestionID++;

}

if (Question.currentQuestion.isCorrectAnswer(answerTestOne)) {

User.currentUser.plusRightAnswer();

}

if (Question.currentQuestionID <= Question.questions.size()-2) {

Question q = Question.questions.get(Question.currentQuestionID);

Question.currentQuestion = q;

q.setQuestionText(jTestOneQuestionText);

q.setAnswers(jTest1AnswerOne, 0);

q.setAnswers(jTest1AnswerTwo, 1);

q.setAnswers(jTest1AnswerThree, 2);

q.setAnswers(jTest1AnswerFour, 3);

testOneBGroup.clearSelection();

if (Question.currentQuestionID == Question.questions.size()-2) {

jTestOneNextButton.setText("End");

}

}

else {

jMGS1Test.setVisible(false);

jResultDialog.setBounds(400,300, 495, 470);

jResultDialog.setVisible(true);

jTestResultLabel.setText(User.currentUser.getName()+" Your result is: "+User.currentUser.getMark());

FileInputClass mark = new FileInputClass(User.currentUser, "MGS1 Test");

Question.currentQuestionID=0;

User.currentUser.setMarkNull();

}

}

private void jTestTwoActionPerformed(java.awt.event.ActionEvent evt) {

jMGS2Test.setVisible(true);

jMGS2Test.setBounds(400,300, 495, 470);

QuestionFileOut test = new QuestionFileOut("MGS2QuestionDB.txt");

testOneBGroup.add(jTest2AnswerOne);

testOneBGroup.add(jTest2AnswerTwo);

testOneBGroup.add(jTest2AnswerThree);

testOneBGroup.add(jTest2AnswerFour);

for(Question q: Question.questions){

Question.currentQuestion = q;

q.setQuestionText(jTestTwoQuestionText);

q.setAnswers(jTest2AnswerOne, 0);

q.setAnswers(jTest2AnswerTwo, 1);

q.setAnswers(jTest2AnswerThree, 2);

q.setAnswers(jTest2AnswerFour, 3);

q.getCorrctAnswer();

}

}

private void jTestTwoNextButtonActionPerformed(java.awt.event.ActionEvent evt) {

int answerTestTwo = 0;

if (!(jTest2AnswerOne.isSelected() ||

jTest2AnswerTwo.isSelected() ||

jTest2AnswerThree.isSelected() ||

jTest2AnswerFour.isSelected()))

{

return;

}

if (jTest2AnswerOne.isSelected()) {

answerTestTwo = 1;

Question.currentQuestionID++;

}

else if (jTest2AnswerTwo.isSelected()) {

answerTestTwo = 2;

Question.currentQuestionID++;

}

else if (jTest2AnswerThree.isSelected()) {

answerTestTwo = 3;

Question.currentQuestionID++;

}

else if (jTest2AnswerFour.isSelected()) {

answerTestTwo = 4;

Question.currentQuestionID++;

}

if (Question.currentQuestion.isCorrectAnswer(answerTestTwo)) {

User.currentUser.plusRightAnswer();

}

if (Question.currentQuestionID <= Question.questions.size()-1) {

Question q = Question.questions.get(Question.currentQuestionID - 1);

Question.currentQuestion = q;

q.setQuestionText(jTestTwoQuestionText);

q.setAnswers(jTest2AnswerOne, 0);

q.setAnswers(jTest2AnswerTwo, 1);

q.setAnswers(jTest2AnswerThree, 2);

q.setAnswers(jTest2AnswerFour, 3);

testOneBGroup.clearSelection();

if (Question.currentQuestionID == Question.questions.size()-1) {

jTestTwoNextButton.setText("End");

}

}

else {

jMGS2Test.setVisible(false);

jResultDialog.setBounds(400,300, 495, 470);

jResultDialog.setVisible(true);

jTestResultLabel.setText(User.currentUser.getName()+" Your result is: "+User.currentUser.getMark());

FileInputClass mark = new FileInputClass(User.currentUser, "MGS2 Test");

Question.currentQuestionID=0;

User.currentUser.setMarkNull();

}

}

private void jLabel12MouseClicked(java.awt.event.MouseEvent evt) {

jHelpWindowLogin.setBounds(400,300, 495, 470);

jHelpWindowLogin.setVisible(true);

}

private void jLabel13MouseClicked(java.awt.event.MouseEvent evt) {

jHelpWindowRegist.setBounds(400,300, 495, 470);

jHelpWindowRegist.setVisible(true);

}

private void jLabel14MouseClicked(java.awt.event.MouseEvent evt) {

jHelpWindowQuiz.setBounds(400,300, 495, 470);

jHelpWindowQuiz.setVisible(true);

}

private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {

jQuizChoice.setVisible(false);

jLoginLoginField.setText("");

jLoginPasswordField.setText("");

User.currentUser = null;

}

private void jTestThreeActionPerformed(java.awt.event.ActionEvent evt) {

jMGS3Test.setVisible(true);

jMGS3Test.setBounds(400,300, 495, 470);

QuestionFileOut test = new QuestionFileOut("MGS3QuestionDB.txt");

testOneBGroup.add(jTest3AnswerOne);

testOneBGroup.add(jTest3AnswerTwo);

testOneBGroup.add(jTest3AnswerThree);

testOneBGroup.add(jTest3AnswerFour);

for(Question q: Question.questions){

Question.currentQuestion = q;

q.setQuestionText(jTest3QuestionText);

q.setAnswers(jTest3AnswerOne, 0);

q.setAnswers(jTest3AnswerTwo, 1);

q.setAnswers(jTest3AnswerThree, 2);

q.setAnswers(jTest3AnswerFour, 3);

q.getCorrctAnswer();

}

}

private void jTestThreeNextButtonActionPerformed(java.awt.event.ActionEvent evt) {

int answerTestTwo = 0;

if (!(jTest3AnswerOne.isSelected() ||

jTest3AnswerTwo.isSelected() ||

jTest3AnswerThree.isSelected() ||

jTest3AnswerFour.isSelected()))

{

return;

}

if (jTest3AnswerOne.isSelected()) {

answerTestTwo = 1;

Question.currentQuestionID++;

}

else if (jTest3AnswerTwo.isSelected()) {

answerTestTwo = 2;

Question.currentQuestionID++;

}

else if (jTest3AnswerThree.isSelected()) {

answerTestTwo = 3;

Question.currentQuestionID++;

}

else if (jTest3AnswerFour.isSelected()) {

answerTestTwo = 4;

Question.currentQuestionID++;

}

if (Question.currentQuestion.isCorrectAnswer(answerTestTwo)) {

User.currentUser.plusRightAnswer();

}

if (Question.currentQuestionID <= Question.questions.size()-1) {

Question q = Question.questions.get(Question.currentQuestionID - 1);

Question.currentQuestion = q;

q.setQuestionText(jTest3QuestionText);

q.setAnswers(jTest3AnswerOne, 0);

q.setAnswers(jTest3AnswerTwo, 1);

q.setAnswers(jTest3AnswerThree, 2);

q.setAnswers(jTest3AnswerFour, 3);

testOneBGroup.clearSelection();

if (Question.currentQuestionID == Question.questions.size()-1) {

jTestThreeNextButton.setText("End");

}

}

else {

jMGS3Test.setVisible(false);

jResultDialog.setBounds(400,300, 495, 470);

jResultDialog.setVisible(true);

jTestResultLabel.setText(User.currentUser.getName()+" Your result is: "+User.currentUser.getMark());

FileInputClass mark = new FileInputClass(User.currentUser, "MGS3 Test");

Question.currentQuestionID=0;

User.currentUser.setMarkNull();

}

}

private void jTestFourNextButtonActionPerformed(java.awt.event.ActionEvent evt) {

int answerTestTwo = 0;

if (!(jTest4AnswerOne.isSelected() ||

jTest4AnswerTwo.isSelected() ||

jTest4AnswerThree.isSelected() ||

jTest4AnswerFour.isSelected()))

{

return;

}

if (jTest4AnswerOne.isSelected()) {

answerTestTwo = 1;

Question.currentQuestionID++;

}

else if (jTest4AnswerTwo.isSelected()) {

answerTestTwo = 2;

Question.currentQuestionID++;

}

else if (jTest4AnswerThree.isSelected()) {

answerTestTwo = 3;

Question.currentQuestionID++;

}

else if (jTest4AnswerFour.isSelected()) {

answerTestTwo = 4;

Question.currentQuestionID++;

}

if (Question.currentQuestion.isCorrectAnswer(answerTestTwo)) {

User.currentUser.plusRightAnswer();

}

if (Question.currentQuestionID <= Question.questions.size()-1) {

Question q = Question.questions.get(Question.currentQuestionID - 1);

Question.currentQuestion = q;

q.setQuestionText(jTest4QuestionText);

q.setAnswers(jTest4AnswerOne, 0);

q.setAnswers(jTest4AnswerTwo, 1);

q.setAnswers(jTest4AnswerThree, 2);

q.setAnswers(jTest4AnswerFour, 3);

testOneBGroup.clearSelection();

if (Question.currentQuestionID == Question.questions.size()-1) {

jTestFourNextButton.setText("End");

}

}

else {

jMGS4Test.setVisible(false);

jResultDialog.setBounds(400,300, 495, 470);

jResultDialog.setVisible(true);

jTestResultLabel.setText(User.currentUser.getName()+" Your result is: "+User.currentUser.getMark());

FileInputClass mark = new FileInputClass(User.currentUser, "MGS4 Test");

Question.currentQuestionID=0;

User.currentUser.setMarkNull();

}

}

private void jTestFourActionPerformed(java.awt.event.ActionEvent evt) {

jMGS4Test.setVisible(true);

jMGS4Test.setBounds(400,300, 495, 470);

QuestionFileOut test = new QuestionFileOut("MGS4QuestionDB.txt");

testOneBGroup.add(jTest4AnswerOne);

testOneBGroup.add(jTest4AnswerTwo);

testOneBGroup.add(jTest4AnswerThree);

testOneBGroup.add(jTest4AnswerFour);

for(Question q: Question.questions){

Question.currentQuestion = q;

q.setQuestionText(jTest4QuestionText);

q.setAnswers(jTest4AnswerOne, 0);

q.setAnswers(jTest4AnswerTwo, 1);

q.setAnswers(jTest4AnswerThree, 2);

q.setAnswers(jTest4AnswerFour, 3);

q.getCorrctAnswer();

}

}

private void jResultExitButtonActionPerformed(java.awt.event.ActionEvent evt) {

jResultDialog.setVisible(false);

}