



Padziļinātais kurss Programmēšana II augstākajā mācību satura apguves limenī

Valsts pārbaudes darba paraugs

Padziļinātais kurss Programmēšana II augstākajā satura apguves līmenī

Valsts pārbaudes darba paraugs

Valsts pārbaudes darba paraugs ir izstrādāts Eiropas Sociālā fonda projektā "Kompetenču pieeja mācību saturā" (turpmāk – Projekts).

Valsts pārbaudes darbu satura, programmu un paraugu izstrādi Projektā vadīja **Pāvels Pestovs**.

Valsts pārbaudes darba parauga izstrādi un sagatavošanu publicēšanai vadīja **Kaspars Antonevičs**.

Valsts pārbaudes darba paraugu izstrādāja **Ance Kancere** un **Edgars Kupčs**.

Valsts pārbaudes darba parauga mācību satura recenzents: **Sergejs Zembkovskis**.

Valsts pārbaudes darba parauga zinātniskais recenzents: **Kristaps Ozols**.

Projekts izsaka pateicību visām Latvijas izglītības iestādēm, kas piedalījās valsts pārbaudes darba parauga aprobācijā.

Saturs

| | |
|--|----|
| Ievads | 4 |
| 1. Valsts pārbaudes darba parauga uzdevumi | 6 |
| 1. daļa. Datortikls un datubāze | 6 |
| 2. daļa. Problēmas analīze, programmatūras specifikācija un darba plānošana, akcepttestēšana, atklādošana | 8 |
| 3. daļa. Objektorientētā programmēšana un ārējās bibliotēkas | 11 |
| 4. daļa. Datu struktūras, programmsaskarne (API) un mašīnmācīšanās principi | 12 |
| 2. Vērtēšanas kritēriji | 13 |
| 1. daļa. Datortikls un datubāze | 13 |
| 2. daļa. Problēmas analīze, programmatūras specifikācija un darba plānošana, akcepttestēšana, atklādošana | 14 |
| 3. daļa. Objektorientētā programmēšana un ārējās bibliotēkas | 16 |
| 4. daļa. Datu struktūras, programmsaskarne (API) un mašīnmācīšanās principi | 22 |
| 3. Valsts pārbaudes darba paraugā iekļauto uzdevumu raksturojums | 26 |
| PIELIKUMI | 28 |
| 1. pielikums. Programmēšanas augstākā līmeņa valsts pārbaudes darba parauga uzdevumu atrisinājumi un atbildes | 28 |
| 1. daļa. Datortikls un datubāze | 28 |
| 3. daļa. Objektorientētā programmēšana un ārējās bibliotēkas | 30 |
| 4. daļa. Datu struktūras, programmsaskarne (API) un mašīnmācīšanās principi | 33 |

Ievads

Programmēšanas padziļinātā līmeņa valsts pārbaudes darba (turpmāk – VPD) paraugs veidots atbilstoši pilnveidotajam mācību saturam un VPD programmai – 1. tabulā parādīts, kā VPD vērtēšanas saturu un uzbūvi ar punktu (īpatsvara) sadalījumu reprezentē VPD parauga uzdevumi.

Būtiskākie akcenti VPD saturā un uzbūvē:

- 1) darbam ir četras daļas. Katrā eksāmena daļā iekļauti uzdevumi, kas pārbauda uz tiem attiecināmās sasniedzamo rezultātu (turpmāk - SR) grupas;
- 2) 1. un 2. daļā var tikt iekļauti dažādu veidu uzdevumi, piemēram, atbilžu izvēles uzdevumi (viena pareizā atbilde), īso atbilžu uzdevumi, izvērsto atbilžu uzdevumi, informācijas apkopošana un apstrāde, problēmas analīze, programmatūras produkta vai to daļu modelēšana u. c.;
- 3) 3. un 4. daļā ir iekļauti uzdevumi, kuros skolēnam jāizstrādā programmatūra atbilstoši dotajam problēmas aprakstam un uzdevuma nosacījumiem. Ja VPD ir iekļauts temats par mašīnmācīšanās principiem un izmantošanas iespējām, tas tiek pārbaudīts līdzīgi kā 1. un 2. daļā;
- 4) programmas koda rakstīšanas labās prakses principu ievērošanas vērtēšanai izmanto snieguma līmeņu aprakstu (turpmāk - SLA).

VPD paraugu papildina uzdevumu atrisinājumi (1. pielikums) un skolēnu uzdevumu risināšanas piemēri un to vērtējums (2. pielikums).

1. tabula. VPD vērtēšanas saturs un uzbūve.

| | VPD daļa | 1.daļa | 2.daļa | 3.daļa | 4.daļa | |
|---|--|------------------------|--|---|---|------------------------|
| SR veids | Satura modulis | Datortīkls un datubāze | Problēmas analīze, programmatūras specifikācija un darba plānošana, akcepttestēšana, atklūdošana | Objektorientētā programmēšana un ārējās bibliotēkas | Datu struktūras, programsaskarne (API) un mašīnmācīšanās principi | SR grupu īpatsvars (%) |
| | SR grupa | | | | | |
| Zināšanas un izpratne | Atpazīst pamatalgoritmus un izprot to darbības principus, spēj lasīt un izprast blokshēmās un pseidokodā rakstīto. Izprot kriptogrāfijas metožu nepieciešamību, datortīkla uzbūvi u. c. | | 5 | 7 | | 10 ± 5 |
| | Salīdzina programmatūras izstrādes modeļus, skaidro programmēšanas jēdzienus, raksturo mašīnmācīšanās izmantošanas iespējas u. c. | | | | | |
| Prasmju grupas | Izstrādā programmatūras prasību specifikāciju, programmatūru, izvēršanas plānu u. c., veic atklūdošanu. | 17 | | 22 | 23 | 57 ± 5 |
| | Lieto prasmes darbā ar informāciju. | | | | 10 | 8 ± 3 |
| | Lieto labās prakses principus. | | | 4 | | 3 ± 1 |
| Zināšanu, izpratnes, prasmju un ieradumu kombinācijas | Kombinē vairākus programmatūras dzīves cikla posmus: problēmas analīzi, programmatūras specifikāciju un darba plānošanu, programmatūras produkta izstrādi, akcepttestēšanu, atklūdošanu, ieviešanu un uzturēšanu. Saskata algoritmu optimizācijas iespējas, izvērtē drošības riskus. | | 12 | | | 16 ± 4 |
| | Satura moduļu un VPD daļu īpatsvars (%) | 15 ± 5 | 15 ± 5 | 33 ± 5 | 33 ± 5 | 100 |

1. Valsts pārbaudes darba parauga uzdevumi

Iepazīsties ar norādījumiem!

- Valsts pārbaudes darbam (turpmāk – VPD) ir četras daļas.

| VPD daļa | Uzdevumu skaits | Punktu skaits | Laiks |
|---|-----------------|---------------|--------|
| 1. daļa. Datortīkls un datubāze | 1 | 10 | 40 min |
| Starpbrīdis | | | 10 min |
| 2. daļa. Problēmas analīze, programmatūras specifikācija un darba plānošana, akcepttestēšana, atklūdošana | 7 | 10 | 40 min |
| Starpbrīdis | | | 40 min |
| 3. daļa. Objektorientētā programmēšana un ārējās bibliotēkas | 5 | 19 | 80 min |
| Starpbrīdis | | | 10 min |
| 4. daļa. Datu struktūras, programmsaskarne (API) un mašīnmācīšanās principi | 2 | 19 | 80 min |




- Katram atbilžu izvēles uzdevumam ir tikai viena pareizā atbilde.
- Īso un izvērsto atbilžu uzdevumos raksti atbildi tam paredzētajā vietā!

1. daļa. Datortīkls un datubāze

Šajā uzdevumā tev būs jāmodelē datubāze.

Ievēro:

- Neveido vairāk tabulu un lauku kā nepieciešams uzdevuma izpildei!
- Tabulu, lauku nosaukumus un datu piemērus raksti latviski (izņēmums – datu tips). Nosaukumos nelieto latviešu valodas diakritiskās zīmes (piemēram, “āķis” vietā raksti “akis”)!
- Izmanto kādu no 1. attēlā dotajiem relāciju tipu apzīmējumiem!

| | | |
|--|---|---|
|  Viens pret daudziem (<i>one to many</i>). Izmantojot šo relāciju tipu, jāizvēlas vajadzīgais virziens! |  Viens pret vienu (<i>one to one</i>) |  Daudzi pret daudziem (<i>many to many</i>) |
|--|---|---|

1. att. Relāciju tipi un to apzīmējumi

1. uzdevums (10 punkti) Analizē doto problēmu un zemāk dotajās tabulās izplāno relāciju datubāzes struktūru (aizpildi tikai tās tabulas un laukus, kuri ir nepieciešami):

- izplāno atbilstošus laukus, kas atvieglo datu apstrādi, (3 punkti)
- uzraksti atbilstošus datu tipus, (2 punkti)
- ieraksti vienu datu piemēru katrā no laukiem, (1 punkts)
- sasaisti tabulas ar atbilstošu relāciju tipu. (4 punkti)

Problēmas apraksts:

Celtniecības instrumentu un aprīkojuma nomas uzņēmumam ir jāizveido datu bāze klientu (klienti ir tikai fiziskas personas), iznomājamās un iznomātās tehnikas uzskaitē.

Datu bāzei jāsaturs:

- informācija par nomnieku (piemēram, Jānis Kociņš), kurš ir viennozīmīgi identificējams un sazvānāms;
- informācija par iznomājamiem instrumentiem, piemēram, urbi, leņķa slīpmašīnu, lāpstu, trepēm. Tajā skaitā – jāsaturs informācija, kad katrs instruments iegādāts un kāda bija katra instrumenta nomas cena dienā.

Vieta uzdevuma atbildei:

Izvēlies un pielieto kādu no relāciju tipiem (sasaisti izvēlētos laukus, aizvelkot un novietojot vajadzīgo bultu):



Tabula **nomnieks**

| Lauka nosaukums | Datu tips | Datu piemērs |
|-----------------|-----------|--------------|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

Tabula:

| Lauka nosaukums | Datu tips | Datu piemērs |
|-----------------|-----------|--------------|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

Tabula **instrumenti**

| Lauka nosaukums | Datu tips | Datu piemērs |
|-----------------|-----------|--------------|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

Tabula:

| Lauka nosaukums | Datu tips | Datu piemērs |
|-----------------|-----------|--------------|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |







2. daļa. Problēmas analīze, programmatūras specifikācija un darba plānošana, akcepttestēšana, atklūdošana

PROBLĒMAS APRAKSTS:

Cēsu Tūrisma informācijas centrs (TIC) regulāri atjauno statistiku, kas satur dažādu Cēsu viesus raksturojošu informāciju, piemēram, par viesu izraudzītajiem galamērķiem, valstīm, no kurienes viesi ieradusies, ierašanās veidu, viesiem sniegto informācijas tematiku, informācijas kanāliem u. tml. Statistika par vairākiem gadiem ir pieejama daudzās lielās savstarpēji nesaistītās *.csv tabulās vietnē *data.gov.lv* (sk. 2. attēlu), līdz ar to informāciju ir grūti pārskatīt un analizēt.

Tūrisma informācijas centram būtu noderīgi izveidot digitālu rīku pārskatu veidošanai ar dažādām datu atlasēs iespējām un vizualizācijām, kas konkrētos datus attēlotu lietotājam saprotamā un ērtā formātā. Papildus Tūrisma informācijas centra pārstāvji būtu priecīgi, ja ar digitālā rīka palīdzību varētu izveidot "Cēsu viesu" profilu, kas palīdzētu regulāri vērtēt centra darbību un uzlabot vispieprasītākos pakalpojumus.

Dati un resursi

| | | |
|---|---|-----------------------------|
|  | Cēsu TIC sniegtās informācijas tematika un kanāli 🔥 | ↩ Izpētīt ▾ |
|  | Cēsu TIC sniegtās informācijas ģeogrāfija | ↩ Izpētīt ▾ |
|  | Cēsu TIC apmeklētāji: valstis 🔥 | ↩ Izpētīt ▾ |
|  | Cēsu TIC apmeklētāji: ar ko kopā ceļo | ↩ Izpētīt ▾ |
|  | Cēsu TIC apmeklētāji: ierašanās veids | ↩ Izpētīt ▾ |
|  | Cēsu TIC: galamērķu apmeklējums | ↩ Izpētīt ▾ |

2. att. Cēsu TIC datu tabulas

Izpēte, mērķauditorija

1. **uzdevums.** (2 punkti) Uzraksti vienu piemērotāko izpētes metodi, ar kuru noskaidrot prasības programmatūras produkta funkcionalitātei! Pamato, kāpēc izvēlētā izpētes metode ir piemērotākā dotajai problēmai!

Izpētes metode: _____

Pamatojums: _____

2. **uzdevums.** (1 punkts) Definē mērķauditoriju, kas tiks iesaistīta izpētes procesā, jo ikdienā savā darbā izmantos izstrādāto programmatūras produktu!

Izpētes mērķauditorija: _____

3. uzdevums. (2 punkti) Sastādi **izpētes procesa plānu**, uzrakstot un īsi paskaidrojot 1. līdz 5. izpētes procesa soli! Sestais solis – secinājumu formulēšana – jau ir dots. Plāna izveidē ņem vērā 1. un 2. uzdevumā tevis sniegtās atbildes.

1) _____

2) _____

3) _____

4) _____

5) _____

6) Formulēju secinājumus, aprakstot Cēsu Tūrisma informācijas centra ikdienas procesus un vajadzības saistībā ar programmatūras produkta izstrādi.

Programmatūras prasību specifikācija programmatūras produkta izstrādei

4. uzdevums. (1 punkts) Ņemot vērā uzdevumā doto problēmu, apraksti, **kādu programmatūras produktu ir nepieciešams izstrādāt** (kurā programmēšanas valodā un ko tu izstrādāsi). (maksimālais apjoms 200 rakstzīmes)

5. uzdevums. (1 punkts) Apraksti programmatūras produkta datu uzglabāšanas veidu un datu fizisko izvietojumu.

Datu uzglabāšanas veids: _____

Datu fiziskais izvietojums: _____

6. uzdevums. (1 punkts) Kurš programmatūras izstrādes modelis ir piemērotākais konkrētā programmatūras produkta izstrādē? Pamato, kāpēc! Sasaisti savu **pamatojumu ar doto problēmu**.

7. uzdevums. (2 punkti) Nosauc un īsi raksturo piecas **būtiskākās funkcijas**, ko nepieciešams izstrādāt programmatūras produktam, ņemot vērā problēmas aprakstā un 2. uzdevumā tevis definētās mērķauditorijas specifiku un vajadzības (pieņemam, lietotāja autorizāciju), kā arī likumdošanas prasības!

Funkcijas, kas minētas problēmas aprakstā:

1.1. _____

1.2. _____

1.3. _____

Papildu funkcijas (*kuras tu ieteiktu papildus problēmas aprakstā vai uzdevuma nosacījumos minētajām*):

1.4. _____

1.5. _____

3. daļa. Objektorientētā programmēšana un ārējās bibliotēkas

1. uzdevums (1 punkts) Kāds ir viens no veidiem, kā uzstādīt objekta īpašību (*attributes*) sākuma vērtības?

Atbilde: _____

2. uzdevums (1 punkts) Ir izveidota klase "Klients". Izvēlies programmēšanas valodu un uzraksti tajā konstruktoru ar īpašību (*attribute*) "vards". Nosauc tavā piemērā lietoto programmēšanas valodu (ja valoda nav nosaukta, atbilde netiek labota)!

Atbilde: _____

3. uzdevums (1 punkts) Dots pseidokoda fragments klasei "Prece" (sk. uzdevuma piemēru). Sevis izvēlētā programmēšanas valodā uzraksti objektu dotajai klasei, piešķirot brīvi izvēlētu vērtību klases īpašībai (*attribute*). Nosauc savā piemērā lietoto programmēšanas valodu (ja valoda nav nosaukta, atbilde netiek labota)!

Uzdevuma piemērs

```
class Prece
    integer cena;
```

Atbilde: _____

4. uzdevums (1 punkts) Atzīmē pareizo atbildi (drīkst būt viena pareiza atbilde)!

Ko dara atslēgvārds "this" ("self") programmēšanā?

- A Atslēgvārds izveido jaunu konstruktoru.
- B Atslēgvārds izveido jaunu mainīgo.
- C Atslēgvārds atgriež mainīgā vērtību.
- D Atslēgvārds norāda saiti uz pašreizējo objektu.

5. uzdevums (15 punkti) Vienā no objektorientētajām programmēšanas valodām izveido programmu, kurā:

- Lietotājs var ievadīt, apskatīt un labot informāciju par kādu no personālā datora sastāvdaļām. Katrai sastāvdaļai ir trīs īpašības – veids, modelis un cena. Ievadot vai labojot informāciju, jābūt aizpildītām visu īpašību vērtībām. Personālā datora sastāvdaļu piemēri doti uzdevuma tabulā. Katras īpašības vērtībai, piemēram, "Corsair Vengeance LPX 16GB" vai "RAM", jāglabājas atsevišķi no citām vērtībām (piemēram, nedrīkst sapludināt vienā īpašībā "RAM Corsair Vengeance LPX 16GB 99,99"). (5 punkti)
- Programmas vadība notiek, lietojot grafisko lietotāja saskarni. (3 punkti)
- Informāciju par personālā datora sastāvdaļu ir iespējams saglabāt teksta datnē šādā formātā (sk. 3. attēlā). (2 punkti)

```
-Personālā datora sastāvdaļa-
Veids: RAM
Modelis: Corsair Vengeance LPX 16GB
Cena: 99,99 EUR
```

3. attēls. Teksta datnes satura piemērs

Ievēro digitāla produkta dizaina, objektorientētas programmēšanas valodas un koda rakstīšanas labās prakses pamatprincipus! (5 punkti)

Uzdevuma tabula. Personālā datora sastāvdaļu piemēri.

| Veids | Modelis | Cena |
|-------|-----------------------------|--------|
| RAM | Corsair Vengeance LPX 16GB | 99,99 |
| GPU | Gigabyte GeForce GT 710 2GB | 75,50 |
| CPU | AMD Ryzen 7 5800X 3,8GHz | 657,80 |

4. daļa. Datu struktūras, programmsaskarne (API) un mašīnmācīšanās principi

Dots API domēns: <http://universities.hipolabs.com/>

Lai meklētu datus pēc universitātes nosaukuma, tiek veidots šāds pieprasījums:

<http://universities.hipolabs.com/search?name=poland>.

Papildus var veidot pieprasījumu pēc valsts nosaukuma. Tad tiek izmantots parametrs "country".

1. uzdevums (6 punkti) Izveidot API pieprasījumu, kas:

- atlasa Latvijas universitātes, (3 punkti)
- sakārto universitātes pēc to nosaukuma alfabētiskā secībā, (1 punkts)
- izvada universitāšu nosaukumus vienu zem otra. (2 punkti)


2. uzdevums (13 punkti) Dota teksta datne *teksts.txt*. Izveido programmu, kas saskaita vārdu biežumu datnē dotajā tekstā un izvada piecus biežāk minētos vārdus. Atslēgas vārda garums nedrīkst būt īsāks par četriem burtiem.

2. Vērtēšanas kritēriji

Skolēna sniegumu vērtē, izmantojot vērtēšanas shēmu (katram punktam aprakstot konkrētu darbību un rezultātu) un snieguma līmeņa aprakstu (skolēna sniegums tiek aprakstīts dažādās kvalitātes gradācijās). 2., 3., 4., 6., 7. tabulā ir apkopotas vērtēšanas shēmas. 5. tabulā ir snieguma līmeņa apraksts (SLA) programmēšanas labās prakses principu ievērošanai.

1. daļa. Datortīkls un datubāze

2. tabula. 1. daļa. Datortīkls un datubāze.

| Uzd. | Kritēriji | Punkti | Piemēri | SOLO līmenis |
|------------|---|--------|--|--------------|
| 1. | Izveidots vismaz viens korekts lauks katrā no tabulām nomnieks un instrumenti | 1 | <ul style="list-style-type: none"> vards, uzvards (tabula nomnieks); instruments (tabula instrumenti); | 2 |
| | Katrā no tabulām nomnieks un instrumenti izveidoti trīs korekti lauki | 1 | <ul style="list-style-type: none"> vards, uzvards (tabula nomnieks); personas kods/klientu kartes Nr (tabula nomnieks) – noteikti jābūt izveidotam laukam, lai viennozīmīgi identificētu nomnieku; talruna_numurs (tabula nomnieks); instruments (tabula instrumenti); datums (tabula instrumenti); cena (tabula instrumenti). | 2 |
| Datu tipi | Uzrakstīts vismaz viens korekts datu tips katrā no tabulām nomnieks un instrumenti | 1 | Tekstam – text Skaitlim – int, money, float Datumam – date | 1 |
| | Laukam, kas saglabā informāciju par cenu datu tips atļauj decimāldaļas | 1 | cena (tabula instrumenti) - money, float vai tml. | 2 |
| Relācija | Katrā no tabulām instrumenti un nomnieks izveidots lauks unikālai ierakstu identifikācijai | 1 | instrumenta_id un nomnieks_id vai līdzīgi | 2 |
| Testa dati | Visiem laukiem ir ievadīti korekti testa datu piemēri | 1 | <ul style="list-style-type: none"> vards, uzvards (tabula nomnieks) – Jānis Bērziņš; personas kods/klientu kartes Nr (tabula nomnieks) – 45; talruna_numurs (tabula nomnieks) – 26101010; instruments (tabula instrumenti) – lāpsta; datums (tabula instrumenti) – 05.03.2022; cena (tabula instrumenti) – 60,50. | 2 |
| | Ierakstu unikālās identifikācijas lauku id vērtības tiek automātiski ģenerētas ar gadījuma vērtībām, lai apgrūtinātu to vērtību uzminēšanu. T. i., tās nav, piemēram, vērtības 1 2 3 utt. | 1 | Lauka instrumenta_id datu tips text ar vērtību htuy87rd-65tf-gt55-kj88-jhc678gd54st | 3 |
| | Ir izveidota trešā tabula, kas nodrošina iznomāto instrumentu un iznomātāju uzskaiti ar atbilstošiem laukiem un datu tiptiem | 1 | Tabula noma ar laukiem: <ul style="list-style-type: none"> id; instrumenta_id; nomnieks_id. | 3 |
| | Izveidota relācija starp tabulu atbilstīgajiem laukiem | 1 | Starp tabulu nomnieks un instrumenti atbilstīgajiem laukiem, ja ir tikai divas tabulas. Starp tabulām nomnieks-noma-instrumenti , ja ir izveidotas trīs tabulas | 2 |
| | Relācijai ir lietoti atbilstoši apzīmējumi | 1 |  | 3 |

2. daļa. Problēmas analīze, programmatūras specifikācija un darba plānošana, akcepttestēšana, atklūdošana

3. tabula. 2. daļa. Problēmas analīze, programmatūras specifikācija un darba plānošana, akcepttestēšana, atklūdošana.

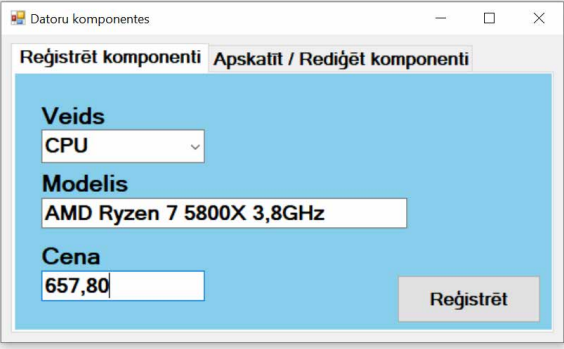
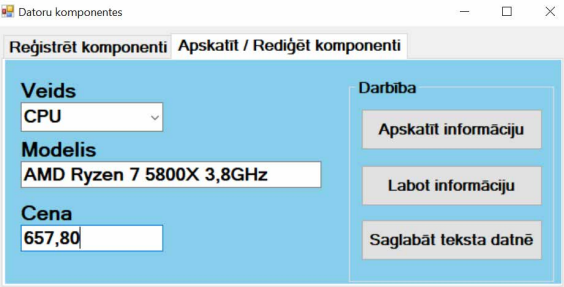
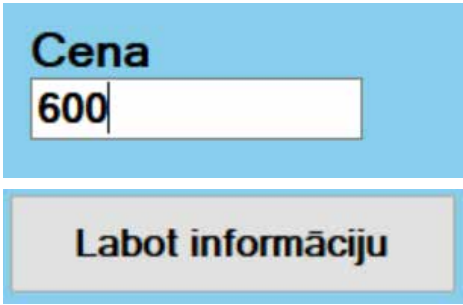
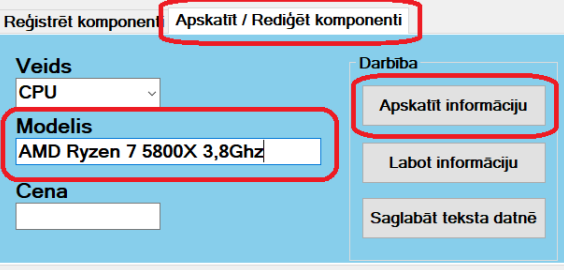
| Uzd. | Kritēriji | Punkti | Piemēri (satur fragmentus no skolēnu atbildēm) | SOLO līmenis |
|------|---|--------|---|-----------------|
| 1. | Nosaukta konkrēta izpētes metode (piemēram aptauja, novērojumu veikšana, lietotāja dienasgrāmatas izveide u. c.) | 1 | 1. piemērs: Manuprāt, visizdevīgāk būtu veikt aptauju, jo tad varētu noskaidrot centra pārstāvju viedokļus un tad varētu veikt secinājumus un izdomāt risinājumu viņu problēmai. | 1 |
| | Pamato izvēlētajā izpētes metodi. Ir skaidri redzama saistība ar doto problēmsituāciju. | 1 | 2. piemērs: Manuprāt, visefektīvākā izpētes metode būtu aptauja/ intervija, jo ar tās palīdzību varētu noskaidrot visu informācijas centra pārstāvju (turpmāk tekstā ICP) viedokļus un balstoties uz tiem es varētu veikt secinājumus un izdomāt, piedāvāt ICP atbilstošu risinājumu viņu problēmām. | 3 |
| 2. | Definēta precīza mērķauditorija, kas saistīta ar Cēsu Tūrisma informācijas centra darbiniekiem. Piemēri: tūrisma informācijas centra darbinieki; tūrisma informācijas centra direktors; tūrisma informācijas centra konsultanti; tūrisma informācijas centra gidi u. tml. | 1 | 1. piemērs: Cēsu Tūrisma informācijas centra darbinieki. 2. piemērs: Cēsu Tūrisma informācijas centra vadītājs un klientu konsultanti. | 1 |
| 3. | Nosaukti un īsi paskaidroti divi izpētes procesa soļi, tiem ir saistība ar 1. un 2. uzdevumā sniegto informāciju (iespējamie izpētes procesa soļi – sagatavošanās, norise, rezultātu apkopošana, izvērtēšana un secinājumu izdarīšana. Skolēnam nav obligāti jālieto minētā soļa nosaukums, bet pietiek ar solim raksturīgajām darbībām). | 1 | 1. Aptaujas izveide. Izveidot aptauju, lai iegūtu vajadzīgo informāciju par lietotāja vajadzībām. 2. Ievāc informāciju no mērķauditorijas. Tiek aptaujāti cilvēki, kas varētu lietot šo programmu, lai uzzinātu kādas funkcijas un informācijas ieguves veidus tiem būtu vieglāk uztvert un izmantot, piemēram, datu atlases veidus. Uzzināt, kā paātrināt un atvieglot informācijas meklēšanas un izmantošanas procesu. Kā arī citu nepieciešamu informāciju, kas var noderēt programmas izveidē. | 1 |
| | Nosaukti un īsi paskaidroti pieci izpētes procesa soļi, tiem ir saistība ar 1. un 2. uzdevumā sniegto informāciju. | 1 | 3. No aptaujas rezultātiem noskaidrošu lietotāju nepieciešamības. Es izskatīšu iegūto informāciju un noskaidrošu, kas ir pieprasītākais starp potenciālajiem lietotājiem. 4. Apkopo izvērtēto informāciju. Piedāvāt Tūrisma centra vadītājam apkopotās informācijas galējo rezultātu. 5. Izdaru secinājumus, lai varētu veikt turpmākās darbības. | 1 |

| Uzd. | Kritēriji | Punkti | Piemēri (satur fragmentus no skolēnu atbildēm) | SOLO līmenis |
|------|---|--------|--|-----------------|
| 4. | Ir skaidri un saprotami pateikts, kas tiks izstrādāts un kāds ir risinājuma formāts (piemēram, "Adaptīvs tīmekļa risinājums dažādu ierīču ekrānu izmēriem Tūrisma informācijas centra datu glabāšanai, apstrādei un apskatei"). | 1 | <p>1. piemērs: Cēsu Tūrisma centra IS risinājums, kurš darbosies izmantojot tīmekļa tehnoloģijas datu attēlošanai un apstrādei (HTML, CSS, JS, PYTHON, MySQL).</p> <p>2. piemērs: Ir nepieciešams izstrādāt mobilo aplikāciju ņemot vērā lietotāju specifiku. Programma ir jāuzglabā nepieciešamajā mobilajā aplikācijas formātā Appstore vai GooglePlay.</p> <p>3. piemērs: Tiks izveidota viegli pārskatāma mājaslapa (HTML), kurā varēs apskatīt visa veida datus un kuru varēs patstāvīgi atjaunot, lai dati būtu aktuāli.</p> | 2 |
| 5. | Norāda uzglabāšanas veidu un datu fizisko izvietojumu (piemēram, SQL datubāze, kas izvietota pilsētas serverī). | 1 | <p>1. piemērs: Oracle datubāze, kas izvietota uzņēmuma serverī.</p> <p>2. piemērs: Lokālajā failu serverī ar datubāzi.</p> <p>3. piemērs: Datus uzglabā MySQL datubāzē, kas izvietota nomas hostinga serverī.</p> | 2 |
| 6. | Nosauc programmatūras izstrādes modeli (piemēram, Agile, Spējais, Ūdenskrituma). Pamatojumā argumenti ir saistīti ar doto problēmsituāciju, mērķauditoriju u. tml. | 1 | <p>1. piemērs: Manuprāt, vispiemērotākais programmatūras izstrādes modelis ir Agile, jo lietotājs var sekot līdzi sava produkta attīstībai, pastāvīgi iesaistoties procesā.</p> <p>2. piemērs: Agile, jo būtu visu laiku ar pasūtītāju procesā līdz pašam gala variantam.</p> | 3 |
| 7. | Funkciju sarakstā ir iekļautas vajadzības, kas minētas problēmsituācijas aprakstā (datu atlase, datu vizualizēšana, Cēsu viesu profila izveide). | 1 | <p>Piemērs funkciju uzskaitījumam:</p> <p>1. Ir nepieciešams izstrādāt datu atlasī.</p> <p>2. Ir nepieciešams izstrādāt Cēsu viesu profila izveidi.</p> <p>3. Ir nepieciešams izstrādāt grafisko attēlojumu.</p> <p>4. Ir nepieciešams izstrādāt lietotāju tiesības.</p> <p>5. Ir nepieciešams izstrādāt maināmu izskatu.</p> | 3 |
| | Funkciju sarakstā ir iekļautas arī papildu funkcijas, kas šādam programmvadāmajam risinājumam būtu būtiskas (piemēram, lietotāju lomas, personalizēt krāsas, fonta izmēru u. c.) | 1 | | 4 |

3. daļa. Objektorientētā programmēšana un ārējās bibliotēkas

Piemērs realizēts izmantojot C# programmēšanas valodu. Pilnu uzdevuma risinājumu skatīt 1. pielikumā.

4. tabula. 3. daļa. Objektorientētā programmēšana un ārējās bibliotēkas.

| Uzd. | Kritēriji | Punkti | Piemēri (satur fragmentus no skolēna risināta uzdevuma) | SOLO līmenis |
|------|--|--------|--|-----------------|
| 1. | Pareiza atbilde. | 1 | Izveidot konstruktoru. | 1 |
| 2. | Pareiza atbilde bez sintakses kļūdām. | 1 | public Klients(string x) { vards = x}; C# programmēšanas valoda | 1 |
| 3. | Pareiza atbilde bez sintakses kļūdām. | 1 | Prece x = new Prece(); x.cena = 1; C# programmēšanas valoda | 1 |
| 4. | Pareiza atbilde. | 1 | d. Atslēgvārds norāda saiti uz pašreizējo objektu. | 1 |
| 5. | Programmas vadība notiek, lietojot grafisko lietotāja saskarni – tiek paredzēta datu ievade, piemēram, ar ievadlaukiem un pogu. | 1 |  | 2 |
| | Programmas vadība notiek, lietojot grafisko lietotāja saskarni – tiek paredzēta datu izvade, piemēram, ar datu izvades laukiem un pogu. | 1 |  | 2 |
| | Programmas vadība notiek, lietojot grafisko lietotāja saskarni – tiek paredzēta datu labošana, piemēram, ar datu ievades laukiem un pogu. | 1 |  | 2 |
| | Izveidota lietotājam draudzīga grafiskā saskarne (piemēram, objekti nepārklājas un ir labi saskatāmi, ievadāmie dati ir redzami pilnā apjomā) un grafisko objektu funkcijas ir paskaidrotas. | 1 |  | 3 |

| Uzd. | Kritēriji | Punkti | Piemēri (satur fragmentus no skolēna risināta uzdevuma) | SOLO līmenis |
|------|---|--------|--|-----------------|
| 5 | Lietotājs var ievadīt (1 punkts), apskatīt (1 punkts) un labot (1 punkts) informāciju par kādu no datora komponentēm. | 3 | <p>1. Datu ievade un saglabāšana objektā.</p> <pre> private void button1_Click_1(object sender, EventArgs e) { // dati tiek saglabāti (reģistrēti) try { if (String.IsNullOrEmpty(comboBox1.Text)) { MessageBox.Show("Nav izvēlēts komponentes veids!"); } else if (String.IsNullOrEmpty(textBoxModelis.Text)) { MessageBox.Show("Nav ievadīts komponentes modelis!"); } else if (String.IsNullOrEmpty(textBoxCena.Text)) { MessageBox.Show("Nav ievadīta komponentes cena!"); } else { pc = new komponentes(comboBox1.SelectedItem.ToString(), textBoxModelis.Text, Convert.ToDouble(textBoxCena.Text)); MessageBox.Show("Dati ir reģistrēti!"); } } catch (Exception ex) { MessageBox.Show("Radusies kļūda reģistrējot komponentes datus: {0}", ex.ToString()); } } </pre> | 3 |

| Uzd. | Kritēriji | Punkti | Piemēri (satur fragmentus no skolēna risināta uzdevuma) | SOLO līmenis |
|------|-----------|--------|--|-----------------|
| 5 | | | <p>2. Datu izvade (datu nolasīšana no objekta)</p> <pre>private void button2_Click(object sender, EventArgs e) { // datu nolasīšana no objekta try { comboBox2.SelectedItem = pc.veids; textBoxModelis2.Text = pc.modelis; textBoxCena2.Text = pc.cena.ToString(); } catch (Exception ex) { MessageBox.Show("Radusies kļūda nolasot komponentes datus: {0}", ex.ToString()); } }</pre> <p>3. Esošā objekta datu labošana</p> <pre>private void button3_Click_1(object sender, EventArgs e) { // datu labošana try { if (String.IsNullOrEmpty(comboBox2.Text)) { MessageBox.Show("Nav izvēlēts komponentes veids!"); } else if (String.IsNullOrEmpty(textBoxModelis2.Text)) { MessageBox.Show("Nav ievadīts komponentes modelis!"); } else if (String.IsNullOrEmpty(textBoxCena2.Text)) { MessageBox.Show("Nav ievadīta komponentes cena!"); } else { pc.veids = comboBox2.SelectedItem.ToString(); pc.modelis = textBoxModelis2.Text; pc.cena = Convert.ToDouble(textBoxCena2.Text); MessageBox.Show("Dati ir laboti!"); } } }</pre> | |

| Uzd. | Kritēriji | Punkti | Piemēri (satur fragmentus no skolēna risināta uzdevuma) | SOLO līmenis |
|------|--|--------|--|-----------------|
| 5 | Datu validācija – neļauj saglabāt datus, ja nav norādīta kāda no parametru vērtībām. | 1 |  <p>The screenshot shows a form titled 'Reģistrēt komponenti' with tabs 'Apskatīt / Rediģēt komponenti'. The form has fields for 'Veids' (highlighted with a red box), 'Modelis' (AMD Ryzen 7 5800X 3,8Ghz), and 'Cena' (75,55). A dialog box is open with the message 'Nav izvēlēts komponentes veids!' and an 'OK' button. A 'Saglabāt teksta datnē' button is visible at the bottom right.</p> | 3 |
| | Informāciju par komponenti ir iespējams saglabāt teksta datnē (1 punkts), informācija teksta datnē ir saglabāta uzdevuma attēlā dotajā formātā (1 punkts). | 2 |  <pre>// datu saglabāšana ārējā teksta datnē public void saveToTxt() { StreamWriter writer = new StreamWriter("komponente.txt"); writer.WriteLine("-Datora komponente-"); writer.WriteLine("Veids: {0}", veids); writer.WriteLine("Modelis: {0}", modelis); writer.WriteLine("Cena: " + cena.ToString() + " EUR"); writer.Close(); MessageBox.Show("Dati ir saglabāti datnē: komponentes.txt!"); }</pre> <p>-Personālā datora sastāvdaļa- Veids: RAM Modelis: Corsair Vengeance LPX 16GB Cena: 99,99 EUR</p> | 3 |

| Uzd. | Kritēriji | Punkti | Piemēri (satur fragmentus no skolēna risināta uzdevuma) | SOLO līmenis |
|------|---|--------|---|-----------------|
| 5 | Balstoties uz OOP pamatprincipiem, izveidota vismaz viena klase ar tās īpašībām (1 punkts) un metodēm (1 punkts). | 2 | <pre>// izveidota klase ar īpašībām, konstruktors un metode, kura saglabā datus teksta datnē public class komponentes { public string veids; public string modelis; public double cena; // konstruktors public komponentes(string veids1, string modelis1, double cena1) { veids = veids1; modelis = modelis1; cena = cena1; } // datu saglabāšana ārējā teksta datnē public void saveToTxt() { StreamWriter writer = new StreamWriter("komponente.txt"); writer.WriteLine("-Datora komponente-"); writer.WriteLine("Veids: {0}", veids); writer.WriteLine("Modelis: {0}", modelis); writer.WriteLine("Cena: " + cena.ToString() + " EUR"); writer.Close(); MessageBox.Show("Dati ir saglabāti datnē: komponentes.txt!"); } }</pre> | 2 |
| | Sasaista klases īpašības un metodes ar vismaz pusi (50%) no izveidotajiem grafiskās saskarnes elementiem. | 1 | <p>Pēc attiecīgo pogu nospiešanas tiek izsaukta metode vai izveidots objekts, dati tiek nolasīti no ievadlaukiem.</p> <pre>pc = new komponentes(comboBox1.SelectedItem.ToString(), textBoxModelis.Text, Convert.ToDouble(textBoxCena.Text)); MessageBox.Show("Dati ir reģistrēti!");</pre> <p>Dotajā piemērā dati tiek nolasīti no ievadlaukiem un saglabāti, lietojot esošo konstrukturu jaunajā "pc" objektā.</p> | 3 |

| Uzd. | Kritēriji | Punkti | Piemēri (satur fragmentus no skolēna risināta uzdevuma) | SOLO līmenis |
|------|---|--------|--|-----------------|
| 5 | Datu apmaiņa programmā notiek, lietojot OOP pieeju. | 1 | Datu ielasīšana no izveidotā objekta "pc" dotajos ievadlaukos. <pre>// datu nolasīšana no objekta try { comboBox2.SelectedItem = pc.veids; textBoxModelis2.Text = pc.modelis; textBoxCena2.Text = pc.cena.ToString(); }</pre> | 3 |
| | Rakstot kodu, izmanto labās prakses principus vismaz SLA 2. līmenī (sk. 5. tabulu). | 1 | Katrs priekšraksts sākas jaunā rindā. Lietotas atkāpes. Jēgpilni klases un īpašību nosaukumi. Lietoti komentāri. <pre>public string veids; public string modelis; public double cena; // konstruktors public komponentes(string veids1, string modelis1, double cena1) { veids = veids1; modelis = modelis1; cena = cena1; }</pre> | 3 |

5. tabula. 3. daļa. Objektorientētā programmēšana un ārējās bibliotēkas – programmēšanas labās prakses principu SLA.

| Kritērijs/Līmenis | Sācis apgūt | Turpina apgūt | Apguvis |
|--|---|--|---|
| Veido programmu, ievērojot labās prakses pieredzi tās pieraksta strukturēšanā un komentāru veidošanā | Lielākajā daļā programmatūras koda lieto labās prakses principus, bet nekonsekventi vai daļēji korekti. | Kopumā korekti un konsekventi programmatūras kodā lieto labās prakses principus, pieļaujot dažas neprecizitātes. | Korekti un konsekventi programmatūras kodā lieto labās prakses principus. |

4. daļa. Datu struktūras, programsaskarne (API) un mašīnmācīšanās principi

Piemēri realizēti izmantojot Python programmēšanas valodu. Pilnus uzdevumu risinājumus skatīt 1. pielikumā.

6. tabula. 4. daļa. Datu struktūras, programsaskarne (API) un mašīnmācīšanās principi.

| Uzd. | Kritēriji | Punkti | Piemēri (satur fragmentus no skolēna risināta uzdevuma) | SOLO līmenis |
|------|--|--------|---|-----------------|
| 1. | Nomaina parametru uz <i>Country</i> . | 1 | <pre>result = requests.get("http://universities.hipolabs.com/search?country=latvia")</pre> | 2 |
| | Nomaina parametra vērtību uz <i>Latvia</i> . | 1 | <pre>result = requests.get("http://universities.hipolabs.com/search?country=latvia")</pre> | 2 |
| | Veic programsaskarnes pieprasījumu. | 1 | <pre>result = requests.get("http://universities.hipolabs.com/search?country=latvia")</pre> | 2 |
| | Sakārto iegūtos datus pēc nosaukuma. | 1 | <pre># Sort uni_list.sort()</pre> | 2 |
| | Izveda iegūtos datus. | 1 | <pre># Print for uni in uni_list: print(uni)</pre> | 1 |
| | Katra universitāte ir izvadīta jaunā rindā. | 1 | <pre>Baltic International Academy Daugavpils University Higher School of Psychology Latvian Academy of Arts Latvian Academy of Culture</pre> | 2 |
| 2. | Iegūst lietotāja datnes saturu. | 1 | <pre>faila_saturs = [] with open('teksts.txt','r',encoding="utf8") as f: # atver failu ar utf8, jo eksāmens latviešu valodā... faila_saturs = f.readlines() # ielasa faila saturu uzreiz listā. Listā viens elements.</pre> | 2 |
| | Veic vārdu saglabāšanu masīvā. | 1 | <pre>vardu_masivs = teikums.split(" ")</pre> | 2 |
| | Sadalīšana veikta pēc atstarpes. | 1 | <pre>for simbols in spec_simboli: teikums = teikums.replace(simbols, " ") # aizvietojam speciālos simbolus ar tukšumu, lai vēlāk varam sasist vārdus listā. vardu_masivs = teikums.split(" ")</pre> | 1 |

| Uzd. | Kritēriji | Punkti | Piemēri (satur fragmentus no skolēna risināta uzdevuma) | SOLO līmenis |
|------|--|--------|--|-----------------|
| 2 | Veic iegūto vārdu attīrīšanu no liekām rakstzīmēm – lieto ciklu. | 1 | <pre> from string import punctuation # ārējās bibliotēkas. spec_simboli = set(punctuation) # šo var ar roku aizvietot, bet izmantojam ārējo bibliotēku un iegūstam speciālos simbolus teikums = faila_satur[0] # listā viens elements, to arī paņemam kā teikumu, lai vieglāk tīrīt. for simbols in spec_simboli: teikums = teikums.replace(simbols, " ") # aizvietojam speciālos simbolus ar tukšumu, lai vēlāk varam sasist vārdus listā. vardu_masivs = teikums.split(" ") for vards in vardu_masivs[:]: # ejam cauri masīvam un notīram visas atstarpes un vārdus, kuri ir mazākl par 4 burtiem. if len(vards) < 4: vardu_masivs.remove(vards) </pre> | 3 |
| | Veic iegūto vārdu attīrīšanu no liekām rakstzīmēm – nosaka liekās rakstzīmes (punktu, komatu, semikolu). | 1 | <pre> from string import punctuation # ārējās bibliotēkas. spec_simboli = set(punctuation) # šo var ar roku aizvietot, bet izmantojam ārējo bibliotēku un iegūstam speciālos simbolus teikums = faila_satur[0] # listā viens elements, to arī paņemam kā teikumu, lai vieglāk tīrīt. for simbols in spec_simboli: teikums = teikums.replace(simbols, " ") # aizvietojam speciālos simbolus ar tukšumu, lai vēlāk varam sasist vārdus listā. vardu_masivs = teikums.split(" ") for vards in vardu_masivs[:]: # ejam cauri masīvam un notīram visas atstarpes un vārdus, kuri ir mazākl par 4 burtiem. if len(vards) < 4: vardu_masivs.remove(vards) </pre> | 4 |

| Uzd. | Kritēriji | Punkti | Piemēri (satur fragmentus no skolēna risināta uzdevuma) | SOLO līmenis |
|------|--|--------|---|-----------------|
| 2 | Veic iegūto vārdu attīrīšanu no liekām rakstzīmēm – teksts ir attīrīts no liekajām rakstzīmēm. | 1 | <pre> from string import punctuation # ārējās bibliotēkas. spec_simboli = set(punctuation) # šo var ar roku aizvietot, bet izmantojam ārējo bibliotēku un iegūstam speciālos simbolus teikums = faila_saturs[0] # listā viens elements, to arī paņemam kā teikumu, lai vieglāk tīrīt. for simbols in spec_simboli: teikums = teikums.replace(simbols, " ") # aizvietojam speciālos simbolus ar tukšumu, lai vēlāk varam sasist vārdus listā. vardu_masivs = teikums.split(" ") for vards in vardu_masivs[:]: # ejam cauri masīvam un notīram visas atstarpes un vārdus, kuri ir mazākl par 4 burtiem. if len(vards) < 4: vardu_masivs.remove(vards) </pre> | 4 |
| | Izveidots vārdu uzskaites algoritms – algoritms veic skaitīšanu. | 1 | <pre> for vards in vardu_masivs[:]: # ejam cauri masīvam un notīram visas atstarpes un vārdus, kuri ir mazākl par 4 burtiem. if len(vards) < 4: vardu_masivs.remove(vards) vardu_skaits = {} #izveidojam vārdnīcu, kurā skaitīsim vārdus. for vards in vardu_masivs: vardu_skaits[vards] = vardu_masivs.count(vards) sakaartots = sorted(vardu_skaits, key=vardu_skaits.get, reverse=True) # sakārtojam vārdnīcu pēc vārdu skaita un noglabājam. </pre> | 3 |
| | Izveidots vārdu uzskaites algoritms – skaitīšana korekti parāda vārdu skaitu tekstā. | 1 | <pre> for vards in vardu_masivs[:]: # ejam cauri masīvam un notīram visas atstarpes un vārdus, kuri ir mazākl par 4 burtiem. if len(vards) < 4: vardu_masivs.remove(vards) vardu_skaits = {} #izveidojam vārdnīcu, kurā skaitīsim vārdus. for vards in vardu_masivs: vardu_skaits[vards] = vardu_masivs.count(vards) sakaartots = sorted(vardu_skaits, key=vardu_skaits.get, reverse=True) # sakārtojam vārdnīcu pēc vārdu skaita un noglabājam. </pre> | 4 |

| Uzd. | Kritēriji | Punkti | Piemēri (satur fragmentus no skolēna risināta uzdevuma) | SOLO līmenis |
|------|--|--------|--|-----------------|
| 2 | Nodrošina vārdu neatkārtošanos. | 1 | <pre>vardu_skaits = {} #izveidojam vārdnīcu, kurā skaitīsim vārdus. for vards in vardu_masivs: vardu_skaits[vards] = vardu_masivs.count(vards) sakaartots = sorted(vardu_skaits, key=vardu_skaits.get, reverse=True) # sakārtojam vārdnīcu pēc vārdu skaita un noglabājam.</pre> | 4 |
| | Sakārto datus pēc vārdu biežuma tekstā – tiek izmantota kārtošana. | 1 | <pre>vardu_skaits = {} #izveidojam vārdnīcu, kurā skaitīsim vārdus. for vards in vardu_masivs: vardu_skaits[vards] = vardu_masivs.count(vards) sakaartots = sorted(vardu_skaits, key=vardu_skaits.get, reverse=True) # sakārtojam vārdnīcu pēc vārdu skaita un noglabājam.</pre> | 3 |
| | Sakārto datus pēc vārdu biežuma tekstā – vārdi ir sakārtoti pēc atkārtotības biežuma | 1 | <pre>vardu_skaits = {} #izveidojam vārdnīcu, kurā skaitīsim vārdus. for vards in vardu_masivs: vardu_skaits[vards] = vardu_masivs.count(vards) sakaartots = sorted(vardu_skaits, key=vardu_skaits.get, reverse=True) # sakārtojam vārdnīcu pēc vārdu skaita un noglabājam.</pre> | 4 |
| | Pārskatāmi izvada biežāk sastopamos vārdus – notiek datu izvade | 1 | <pre># izdruka print("Vārds : atkārtotības reizes") for key in sakaartots[:5]: print(key, " : ", vardu_skaits[key])</pre> | 1 |
| | Pārskatāmi izvada pārskatāmi biežāk sastopamos vārdus – izvada biežāk lietotos atslēgas vārdus | 1 | <pre>Vārds : atkārtotības reizes atslēgas : 3 vārdus : 3 veidotu : 2 punkti : 2 tekstam : 1</pre> | 2 |

3. Valsts pārbaudes darba paraugā iekļauto uzdevumu raksturojums

Lai nodrošinātu VPD atbilstību izvirzītajam mērķim – pārbaudīt standartā noteikto SR apguvi un iegūt iespējami reprezentatīvus datus par skolēnu sniegumu valsts pārbaudes darbā –, katrs VPD uzdevums tiek raksturots vairākās kategorijās (sk. 8. tabulu).

8. tabula. VPD parauga uzdevumu raksturojums.

| Uzd. | Sasniedzamais rezultāts | Standarta SR kods | SR grupa | Satura modulis | Izziņas darbības līmenis (SOLO) |
|---------|--|-------------------------|--|--|---------------------------------|
| 1. daļa | | | | | |
| 1. | Plāno datubāzi, t. sk. izveido ER modeli konkrētā uzdevuma datu apstrādes risinājumam. Izveido vienkāršu datu apstrādes programmatūru (sistēmu), datu uzglabāšanai izmantojot paša veidotu datubāzi ar vairākām tabulām. | T.A.2.3.2., T.A.2.4.17. | Izstrādā programmatūras prasību specifikāciju, programmatūru, izvēršanas plānu u. c., veic atklādošanu. | Datortīkls un datubāze | 1-3 |
| 2. daļa | | | | | |
| 1. | Izvēlas izpētes metodi dotajai problēmsituācijai, pamato izpētes metodes izvēli. | T.A.2.4.1. | Atpazīst pamatalgoritmus un izprot to darbības principus, spēj lasīt un izprast blokshēmās un pseidokodā rakstīto. Izprot kriptogrāfijas metožu nepieciešamību, datortīkla uzbūvi u. c. Kombinē vairākus programmatūras dzīves cikla posmus. Saskata algoritmu optimizācijas iespējas, izvērtē drošības riskus. | Problēmas analīze, programmatūras specifikācija un darba plānošana, akcepttestēšana, atklādošana | 1-3 |
| 2. | Definē mērķauditoriju. | T.A.2.4.4. | Kombinē vairākus programmatūras dzīves cikla posmus. Saskata algoritmu optimizācijas iespējas, izvērtē drošības riskus. | | 1 |
| 3. | Izveido dotajai problēmsituācijai atbilstošu izpētes procesa plānu. | T.A.2.4.1. | Kombinē vairākus programmatūras dzīves cikla posmus. Saskata algoritmu optimizācijas iespējas, izvērtē drošības riskus. | | 1 |
| 4. | Definē programmvadāmo risinājumu. | T.A.2.4.1. | Kombinē vairākus programmatūras dzīves cikla posmus. Saskata algoritmu optimizācijas iespējas, izvērtē drošības riskus. | | 2 |
| 5. | Apraksta programmvadāmā risinājuma datu uzglabāšanas nosacījumus. | T.A.2.4.1. | Kombinē vairākus programmatūras dzīves cikla posmus. Saskata algoritmu optimizācijas iespējas, izvērtē drošības riskus. | | 2 |
| 6. | Izvēlas piemērotāko programmatūras izstrādes modeli konkrētā uzdevuma atrisināšanai, pamato izvēli. | T.A.2.4.2. | Salīdzina programmatūras izstrādes modeļus, skaidro programmēšanas jēdzienus, raksturo mašīnmācīšanās izmantošanas iespējas u. c. Kombinē vairākus programmatūras dzīves cikla posmus. Saskata algoritmu optimizācijas iespējas, izvērtē drošības riskus. | | 3 |

| Uzd. | Sasniedzamais rezultāts | Standarta SR kods | SR grupa | Satura modulis | Izziņas darbības līmenis (SOLO) |
|---------|---|-----------------------------|--|--|---------------------------------|
| 7. | Definē programmvadāmā risinājuma funkcijas. | T.A.2.4.1. | Salīdzina programmatūras izstrādes modeļus, skaidro programmēšanas jēdzienus, raksturo mašīnmācīšanās izmantošanas iespējas u. c. Kombinē vairākus programmatūras dzīves cikla posmus. Saskata algoritmu optimizācijas iespējas, izvērtē drošības riskus. | | 3-4 |
| 3. daļa | | | | | |
| 1. | Skaidro objektorientētās programmēšanas pamatprincipus. | T.A.2.4.15. | Atpazīst pamatalgoritmus un izprot to darbības principus, spēj lasīt un izprast blokshēmās un pseidokodā rakstīto. Izprot kriptogrāfijas metožu nepieciešamību, datortīkla uzbūvi u. c. | Objekto-orientētā program- mēšana un ārējās biblio- tēkas | 1 |
| 2. | Atpazīst objektorientētās programmēšanas pamatprincipus. | T.A.2.4.15. | Atpazīst pamatalgoritmus un izprot to darbības principus, spēj lasīt un izprast blokshēmās un pseidokodā rakstīto. Izprot kriptogrāfijas metožu nepieciešamību, datortīkla uzbūvi u. c. | | 1 |
| 3. | Atpazīst objektorientētās programmēšanas pamatprincipus. | T.A.2.4.15. | Atpazīst pamatalgoritmus un izprot to darbības principus, spēj lasīt un izprast blokshēmās un pseidokodā rakstīto. Izprot kriptogrāfijas metožu nepieciešamību, datortīkla uzbūvi u. c. | | 1 |
| 4. | Atpazīst objektorientētās programmēšanas pamatprincipus. | T.A.2.4.15. | Atpazīst pamatalgoritmus un izprot to darbības principus, spēj lasīt un izprast blokshēmās un pseidokodā rakstīto. Izprot kriptogrāfijas metožu nepieciešamību, datortīkla uzbūvi u. c. | | 1 |
| 5. | Veido programmas vienā no objektorientētajām programmēšanas valodām. Izmanto programmēšanas valodas un tās bibliotēku dokumentāciju un palīdzības sistēmu, lai patstāvīgi apgūtu citas to piedāvātās iespējas, kas nepieciešamas konkrētās programmatūras izstrādei. | T.A.2.4.15., T.A.2.4.10. | Izstrādā programmatūras prasību specifikāciju, programmatūru, izvēršanas plānu u. c., veic atklādošanu. Lieto labās prakses principus. | | 2-3 |
| 4. daļa | | | | | |
| 1. | Meklē un pievieno atvērtā koda bibliotēkas un lieto API (programmsaskarni) specializētu funkciju veikšanai. | T.A.2.4.11. | Lieto prasmes darbā ar informāciju. | Datu struk- tūras, prog- rammsa- skarne (API) un mašīn- mācīšanās principi | 1-2 |
| 2. | Izmanto dažādas datu struktūras un ar tiem saistītos pamatalgoritmus. | T.A.2.4.14. | Izstrādā programmatūras prasību specifikāciju, programmatūru, izvēršanas plānu u. c., veic atklādošanu. | | 1-4 |

PIELIKUMI**1. pielikums.****Programmēšanas augstākā līmeņa valsts pārbaudes darba parauga uzdevumu atrisinājumi un atbildes****1. daļa. Datortīkls un datubāze**

Uzdevuma realizācijas piemērs, modelējot datubāzi tekstapstrādes lietotnes tabulās.

| Tabula nomnieks | | |
|-----------------|-----------|---------------------------------------|
| Lauka nosaukums | Datu tips | Datu piemērs |
| id | text | htuy87rd-22tf-gt55-kj88-jhc678-gd65kj |
| vards | text | Jānis |
| uzvards | text | Bērziņš |
| talruna_numurs | integer | 29010101 |
| | | |

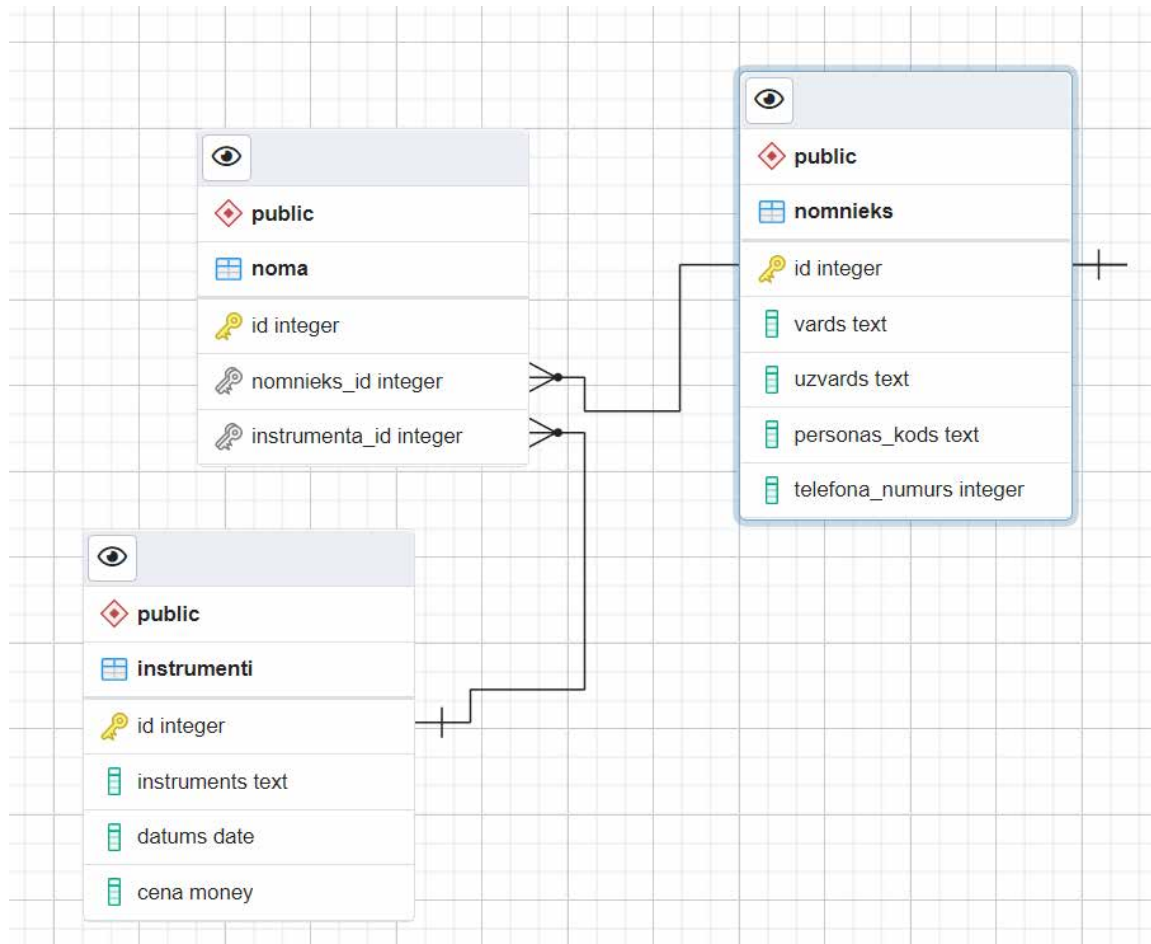
| Tabula noma | | |
|-----------------|-----------|--------------------------------------|
| Lauka nosaukums | Datu tips | Datu piemērs |
| id | text | htuy11rd-11tf-gu88-kj88-jhc878gd65aa |
| instrumenta_id | text | htuy87rd-65tf-gt55-kj88-jhc678gd54st |
| nomnieka_id | text | htuy87rd-22tf-gt55-kj88-jhc678gd65kj |
| | | |

| Tabula instrumenti | | |
|--------------------|-----------|--------------------------------------|
| Lauka nosaukums | Datu tips | Datu piemērs |
| id | text | htuy87rd-65tf-gt55-kj88-jhc678gd54st |
| instruments | text | grābeklis |
| datums | date | 2020-05-10 |
| cena | money | 10.50 |
| | | |

| Tabula: | | |
|-----------------|-----------|--------------|
| Lauka nosaukums | Datu tips | Datu piemērs |
| | | |
| | | |
| | | |
| | | |

Uzdevuma realizācijas piemērs, izveidojot datubāzi PostgreSQL (satur fragmentus no skolēna risināta uzdevuma. Datubāzes izveide uzdevumā netika prasīta).

Relāciju vizualizācija (ar atšķirīgu id datu tipu):



Datu bāzes darbības pārbaude (satur fragmentus no skolēna risināta uzdevuma. Darbības pārbaude uzdevumā netika prasīta):

```

1 SELECT nomnieks.vards,nomnieks.uzvards, instrumenti.instruments
2 FROM noma
3 JOIN nomnieks
4 ON nomnieks.id = noma.nomnieks_id
5 JOIN instrumenti on instrumenti.id = noma.instrumenta_id

```

Data Output Explain Messages Notifications

| | vards text | uzvards text | instruments text |
|---|---------------|-----------------|---------------------|
| 1 | Jānis | Bērziņš | grābeklis |

3. daļa. Objektorientētā programmēšana un ārējās bibliotēkas**5. uzdevums**

Vienā no objektorientētajām programmēšanas valodām izveido programmu, kurā:

- Lietotājs var ievadīt, apskatīt un labot informāciju par kādu no personālā datora sastāvdaļām. Katrai sastāvdaļai ir trīs īpašības – veids, modelis un cena. Ievadot vai labojot informāciju, jābūt aizpildītām visu īpašību vērtībām. Personālā datora sastāvdaļu piemēri doti uzdevuma tabulā. Katras īpašības vērtībai, piemēram, "Corsair Vengeance LPX 16GB" vai "RAM", jāglabājas atsevišķi no citām vērtībām (piemēram, nedrīkst sapludināt vienā īpašībā "RAM Corsair Vengeance LPX 16GB 99,99").
- Programmas vadība notiek, lietojot grafisko lietotāja saskarni.
- Informāciju par personālā datora sastāvdaļu ir iespējams saglabāt teksta datnē šādā formātā (sk. 3. attēlā).

-Personālā datora sastāvdaļa-
Veids: RAM
Modelis: Corsair Vengeance LPX 16GB
Cena: 99,99 EUR

3. attēls. Teksta datnes satura piemērs

Ievēro digitāla produkta dizaina, objektorientētas programmēšanas valodas un koda rakstīšanas labās prakses pamatprincipus!

Uzdevuma tabula. Personālā datora sastāvdaļu piemēri

| Veids | Modelis | Cena |
|-------|-----------------------------|--------|
| RAM | Corsair Vengeance LPX 16GB | 99,99 |
| GPU | Gigabyte GeForce GT 710 2GB | 75,50 |
| CPU | AMD Ryzen 7 5800X 3,8GHz | 657,80 |

Piemērā izmantots skolēna risināts uzdevums C# programmēšanas valodā.

```
using System;
using System.IO;
using System.Windows.Forms;

namespace OOP_GUI
{
    public partial class Form1 : Form
    {
        // ir izveidots objekts no klases komponentes
        public komponentes pc;
        public Form1()
        {
            InitializeComponent();
        }
        private void button2_Click(object sender, EventArgs e)
        {
            // datu nolasīšana no objekta
            try
            {
                comboBox2.SelectedItem = pc.veids;
                textBoxModelis2.Text = pc.modelis;
                textBoxCena2.Text = pc.cena.ToString();
            }
            catch (Exception ex)
            {
                MessageBox.Show("Radusies kļūda nolasot komponentes datus: {0}",
                    ex.ToString());
            }
        }
        private void button1_Click_1(object sender, EventArgs e)
        {
            // dati tiek saglabāti (reģistrēti)
            try
            {
                if (String.IsNullOrEmpty(comboBox1.Text))
                {
                    MessageBox.Show("Nav izvēlēts komponentes veids!");
                }
                else if (String.IsNullOrEmpty(textBoxModelis.Text))
                {
                    MessageBox.Show("Nav ievadīts komponentes modelis!");
                }
                else if (String.IsNullOrEmpty(textBoxCena.Text))
                {
                    MessageBox.Show("Nav ievadīta komponentes cena!");
                }
                else
                {
                    pc = new komponentes(comboBox1.SelectedItem.ToString(),
                        textBoxModelis.Text,
                        Convert.ToDouble(textBoxCena.Text));
                    MessageBox.Show("Dati ir reģistrēti!");
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show("Radusies kļūda reģistrējot komponentes datus: {0}",
                    ex.ToString());
            }
        }
    }
}
```

```

private void button3_Click_1(object sender, EventArgs e)
{
    // datu labošana
    try
    {
        if (String.IsNullOrEmpty(comboBox2.Text))
        {
            MessageBox.Show("Nav izvēlēts komponentes veids!");
        }
        else if (String.IsNullOrEmpty(textBoxModelis2.Text))
        {
            MessageBox.Show("Nav ievadīts komponentes modelis!");
        }
        else if (String.IsNullOrEmpty(textBoxCena2.Text))
        {
            MessageBox.Show("Nav ievadīta komponentes cena!");
        }
        else
        {
            pc.veids = comboBox2.SelectedItem.ToString();
            pc.modelis = textBoxModelis2.Text;
            pc.cena = Convert.ToDouble(textBoxCena2.Text);
            MessageBox.Show("Dati ir laboti!");
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Radusies kļūda veicot izmaiņas: {0}", ex.ToString());
    }
}

private void button4_Click(object sender, EventArgs e)
{
    // datu izvade ārējā teksta datnē
    pc.saveToTxt();
}

// izveidota klase ar īpašībām, konstruktors un metode, kura saglabā datus teksta datnē
public class komponentes
{
    public string veids;
    public string modelis;
    public double cena;
    // konstruktors
    public komponentes(string veids1, string modelis1, double cena1)
    {
        veids = veids1;
        modelis = modelis1;
        cena = cena1;
    }
    // datu saglabāšana ārējā teksta datnē
    public void saveToTxt()
    {
        StreamWriter writer = new StreamWriter("komponente.txt");
        writer.WriteLine("-Datora komponente-");
        writer.WriteLine("Veids: {0}", veids);
        writer.WriteLine("Modelis: {0}", modelis);
        writer.WriteLine("Cena: " + cena.ToString() + " EUR");
        writer.Close();
        MessageBox.Show("Dati ir saglabāti datnē: komponentes.txt!");
    }
}
}

```


4. daļa. Datu struktūras, programmsaskarne (API) un mašīnmācīšanās principi

1. uzdevums

Izveidot API pieprasījumu, kas:

- atlasa Latvijas universitātes,
- sakārto universitātes pēc to nosaukuma alfabētiskā secībā,
- izvada universitāšu nosaukumus vienu zem otra.

Piemērā izmantots skolēna risināts uzdevums Python programmēšanas valodā.

```
import requests
import json

# Get data
result = requests.get("http://universities.hipolabs.com/search?country=latvia")
universities = json.loads(result.content)
uni_list = []
for uni in universities:
    uni_list.append(uni['name'])
uni_list = list(dict.fromkeys(uni_list))
# Sort
uni_list.sort()
# Print
for uni in uni_list:
    print(uni)
```

2. uzdevums

Dota teksta datne *teksts.txt*.

Izveido programmu, kas saskaita vārdu biežumu datnē dotajā tekstā un izvada piecus biežāk minētos vārdus. Atslēgas vārda garums nedrīkst būt īsāks par četriem burtiem.

Piemērā izmantots skolēna risināts uzdevums Python programmēšanas valodā.

```
from string import punctuation # ārējās bibliotēkas.
spec_simboli = set(punctuation) # šo var ar roku aizvietot, bet izmantojam ārējo bibliotēku un iegūstam speciālos simbolus
faila_saturs = []
with open('teksts.txt','r',encoding="utf8") as f: # atver failu ar utf8, jo eksāmens latviešu valodā...
    faila_saturs = f.readlines() # ielasa faila saturu uzreiz listā. Listā viens elements.
teikums = faila_saturs[0] # listā viens elements, to arī paņemam kā teikumu, lai vieglāk tīrīt.
for simbols in spec_simboli:
    teikums = teikums.replace(simbols, " ") # aizvietojam speciālos simbolus ar tukšumu, lai vēlāk varam sasist vārdus listā.
vardu_masivs = teikums.split(" ")
for vards in vardu_masivs[:]: # ejam cauri masīvam un notīram visas atstarpes un vārdus, kuri ir mazāki par 4 burtiem.
    if len(vards) < 4:
        vardu_masivs.remove(vards)
vardu_skaits = {} #izveidojam vārdnīcu, kurā skaitīsim vārdus.
for vards in vardu_masivs:
    vardu_skaits[vards] = vardu_masivs.count(vards)
sakaartots = sorted(vardu_skaits, key=vardu_skaits.get, reverse=True) # sakārtojam vārdnīcu pēc vārdu skaita un noglabājam.
# izdruka
print("Vārds : atkārtošanās reizes")
for key in sakaartots[:5]:
    print(key, " : ", vardu_skaits[key])
```

**DOMĀT.
DARĪT.
ZINĀT.**

Valsts izglītības satura centra īstenotā projekta "Kompetenču pieeja mācību saturā" mērķis ir izstrādāt, aprobēt un pēctecīgi ieviest Latvijā tādu vispārējās izglītības saturu un pieeju mācīšanai, lai skolēni gūtu dzīvei 21. gadsimtā nepieciešamās zināšanas, prasmes un attieksmes.

Projekts Nr. 8.3.1.1/16/I/002 Kompetenču pieeja mācību saturā



NACIONĀLAIS
ATTĪSTĪBAS
PLĀNS 2020



EIROPAS SAVIENĪBA
Eiropas Sociālais
fonds

IEGULDĪJUMS TAVĀ NĀKOTNĒ