

Valsts pārbaudes darbi
2022./2023. m.g.

Programmēšana

**Augstākais mācību
satura apguves
līmenis**



CENTRALIZĒTĀ EKSĀMENA
PROGRAMMA



Programmēšana

Augstākais satura mācību apguves līmenis

Centralizētā eksāmena programma

S A T U R S

1.	Eksāmena mērķis un adresāts	2
2.	Vērtēšanas saturs	2
2.1.	Sasniedzamo rezultātu veids un grupa	2
2.2.	Satura moduļi	3
2.3.	Izziņas darbības līmenis	4
3.	Eksāmena darba uzbūve	5
4.	Eksāmena piekļuves nosacījumi	5
5.	Nepieciešamo resursu nodrošinājumu	7
6.	Vērtēšanas kārtība un kritēriji	7
6.1.	Vērtēšanas kārtība	7
6.2.	Vērtēšanas kritēriji	8
7.	Palīgīdzekļi, kurus atļauts izmantot eksāmena laikā	10
	PIELIKUMI	11
1.	pielikums. Programmēšanas labās prakses princips	11
2.	pielikums Rīcības vārdu skaidrojumi ar uzdevumu piemēriem	12
3.	pielikums Projekta izstrādes dokumentācijas snieguma līmeņu apraksts	17

1. Eksāmena mērķis un adresāts

Centralizētā eksāmena (turpmāk – eksāmens) mērķis ir novērtēt skolēnu sniegumu mācību priekšmetā atbilstoši Ministru kabineta 2019. gada 3. septembra noteikumiem Nr. 416 “Noteikumi par valsts vispārējās vidējās izglītības standartu un vispārējās vidējās izglītības programmu paraugiem” (turpmāk – standarts) un standarta 7. pielikumam “Plānotie skolēnam sasniedzamie rezultāti tehnoloģiju mācību jomā” optimālajā un augstākajā mācību satura apguves līmenī, identificēt un izvērtēt, cik lielā mērā ir apgūti plānotie sasniedzamie rezultāti (turpmāk – SR).

Eksāmena adresāts – skolēni, kuri ir apguvuši tehnoloģiju mācību jomas SR optimālajā un augstākajā mācību satura apguves līmenī atbilstoši mācību priekšmetu kursiem Programmēšana I un Programmēšana II (standarta 9. pielikums).

2. Vērtēšanas saturs

Programmēšanas augstākā mācību satura apguves līmeņa eksāmena vērtēšanas saturu raksturo trīs kategorijas:

- 1) sasniedzamo rezultātu veids un grupa;
- 2) satura modulis;
- 3) izziņas darbības līmenis.

Tas nozīmē, ka katru eksāmenu testelementu¹ raksturo noteikts SR veids un grupa, satura modulis un izziņas darbības līmenis.

2.1. Sasniedzamo rezultātu veids un grupa

Skolēnam plānoti triju veidu SR:

- 1) zināšanas un izpratne;
- 2) prasmju grupas;
- 3) zināšanu, izpratnes, prasmju un ieradumu kombinācijas.

Katram SR veidam ir norādītas sasniedzamo rezultātu grupas (sk. 1. tabulu), kuras apkopo standartā noteikto mācību saturu un tiek pārbaudītas eksāmenā.

Zināšanu un izpratnes pārbaudei eksāmenā iekļauti uzdevumi, kuros skolēni:

- 1) atpazīst pamatalgoritmus un izprot to darbības principus,
- 2) salīdzina programmatūras izstrādes modeļus, skaidro programmēšanas jēdzienus.

Prasmju apguvi raksturo trīs SR grupas:

- 1) izstrādā programmatūras prasību specifiku, programmu, izvēršanas plānu u. c.;
- 2) lieto prasmes darbā ar informāciju, piemēram, izstrādājot programmatūru, jēgpilni izmanto esošo bibliotēku piedāvātās iespējas, izmanto programmēšanas valodas dokumentāciju, lai apgūtu programmēšanas valodas iespējas dotās problēmas risināšanā (t. sk. sintakses pārbaudē) u. c.;

¹ Testelementi ir uzdevums vai uzdevuma daļa, kas veidota, lai pēc kritērijiem vērtētu kādu konkrētu skolēnu darbības aspektu.

- 3) lieto labās prakses principus (sk. 1. pielikumu). Vērtējumu par labās prakses principu lietošanu skolēns iegūst eksāmena uzdevumos, kuru vērtēšanas kritērijos norādīts par šo prasmju pārbaudi.

Zināšanu, izpratnes, prasmju un ieradumu kombināciju pārbaudei izmanto uzdevumus, kuri skolēniem liek atrisināt problēmu jaunā situācijā, turklāt to saturs var būt no jebkura satura moduļa (sk. 2. tabulu). Tie ir uzdevumi, kuros tiek kombinēti vairāki programmatūras dzīves cikla posmi: problēmas analīze, programmatūras specifikācija un darba plānošana, programmatūras produkta izstrāde.

1. tabula. SR veidi, grupas un to īpatsvars eksāmenā

SR veids un grupa		Īpatsvars (%)
Zināšanas un izpratne	Atpazīst pamatalgoritmus un izprot to darbības principus, spēj lasīt un izprast blokshēmās un pseidokodā rakstīto.	10 ± 5
	Salīdzina programmatūras izstrādes modeļus, skaidro programmēšanas jēdzienus, raksturo mašīnmācīšanās izmantošanas iespējas u. c.	
Prasmju grupas	Izstrādā programmatūras prasību specifikācijas daļu, programmatūru, izvēršanas plānu u. c.	57 ± 5
	Lieto prasmes darbā ar informāciju.	8 ± 3
	Lieto programmēšanas labās prakses principus.	3 ± 1
Zināšanu, izpratnes, prasmju un ieradumu kombinācijas	Kombinē vairākus programmatūras dzīves cikla posmus: problēmas analīzi, programmatūras specifikāciju un darba plānošanu, programmatūras produkta izstrādi, izvērtē drošības riskus.	16 ± 4

2.2. Satura moduļi

Satura moduļu īpatsvars eksāmenā (sk. 2. tabulu) ir proporcionāls kursa apguvei noteiktajam stundu skaitam.

2. tabula. Satura moduļi un to īpatsvars eksāmenā

Satura modulis	Īpatsvars (%)
Objektorientētā programmēšana un ārējās bibliotēkas	33 ± 5
Datortīkls un datubāze	15 ± 5
Datu struktūras, programsaskarne (API)	33 ± 5
Programmatūras dzīves cikls (problēmas analīze, programmatūras specifikācija un darba plānošana)	15 ± 5

Eksāmena izstrādes procesā tiek saskaņots un nodrošināts sadaļu procentuālais sadalījums gan SR veidiem un grupām, gan satura moduļiem.

2.3. Izziņas darbības līmenis

Eksāmenā iekļautie uzdevumi ir grupēti četros izziņas darbības līmeņos, un to līmeņa noteikšanai izmanto SOLO jeb novēroto mācīšanās rezultātu taksonomiju. SOLO taksonomijā skolēna sniegums tiek raksturots, analizējot ideju jeb struktūrelementu skaitu un saišu kvalitāti starp šiem struktūrelementiem. Vispārīgs izziņas darbības līmeņu apraksts, kas piemērots eksāmenā, apkopots 3. tabulā.

3. tabula. Izziņas darbības līmeņu raksturojums un to īpatsvars eksāmenā.

Izziņas darbības līmenis un tā apraksts		Īpatsvars (%)
I	Atceras, nosauc atsevišķas idejas, izpilda īsas procedūras.	20 ± 5
II	Saista, skaidro, lieto zināšanas vai prasmes pazīstamās situācijās.	30 ± 5
III	Saista, skaidro, lieto zināšanas vai prasmes jaunās situācijās, demonstrējot izpratni (analizējot, pielietojot, pamatojot, salīdzinot).	30 ± 5
IV	Lieto zināšanas un prasmes situācijās ar augstu kompleksuma pakāpi.	11 ± 5

Katram līmenim atbilstošo uzdevumu īpatsvars noteikts, ievērojot eksāmena mērķi un galvenos vērtēšanas principus. Pirmais princips – skolēnu grupai ar zemu un vidēju snieguma līmeni programmēšanā dota iespēja apliecināt savas zināšanas un prasmes pietiekami plašā satura jautājumu lokā, t. sk. uzdevumos, kas mēra zināšanu, izpratnes, prasmju un ieradumu kompleksu lietojumu. Otrais princips – SR veidu “Zināšanas un izpratne” un “Prasmju grupas” vērtēšanai iekļauti testelementi, kas atbilst III izziņas darbības līmenim, tādējādi akcentējot izpratnes vērtēšanu. Trešais princips – visi SR veida “Zināšanu, izpratnes, prasmju un ieradumu kombinācijas” vērtēšanai iekļautie uzdevumi ietver vismaz III līmenim atbilstošus testelementus.

3. Eksāmena darba uzbūve

4.tabula. Eksāmena daļu īpatsvars un izpildei paredzētais laiks

Daļa	Uzdevumu skaits	Maksimālais punktu skaits	Daļas īpatsvars, %	Izpildes laiks, min
1. Datortīkls un datu bāze	1	20	17	70
2. Programmatūras dzīves cikls	1	20	17	
3. Objektorientēta programmēšana un ārējās bibliotēkas	1	38	31	80
4. Datu struktūras, programsaskarne (API) principi	2	42 (28*1,5)	35	80
Kopā	5	120	100	230

Katrā eksāmena daļā iekļauti uzdevumi, kuri pārbauda satura moduļiem un sasniedzamo rezultātu veidiem atbilstošās SR grupas.

1. un 2. daļā var tikt iekļauti dažāda veida uzdevumi, piemēram, atbilžu izvēles uzdevumi (viena pareizā atbilde), īso atbilžu uzdevumi, izvērsto atbilžu uzdevumi, informācijas apkopošana un apstrāde, problēmas analīze, risinājuma vai tā daļu modelēšana u.c. Katra no eksāmena daļām var saturēt visu šo veidu uzdevumus. Katra veida uzdevumu skaits un īpatsvars daļā un eksāmenā kopumā nav noteikts. Uzdevuma veida izvēli nosaka atbilstība sasniedzamajam rezultātam, ko tas pārbauda.

3. un 4. daļā ir iekļauti uzdevumi, kuros skolēnam jāizstrādā programmatūras produkts atbilstoši dotajam problēmas aprakstam, uzdevuma nosacījumiem. Skolēns drīkst izmantot jebkuru programmēšanas valodu un vidi, kuru ir apguvis mācību procesā skolā. Praktiskajās daļās tiks vērtēts programmēšanas labās prakses principu lietojums (sk. 1. pielikumu).

4. Eksāmena piekļuves nosacījumi

Eksāmenu var kārtot jebkurš skolēns, kas izstrādājis programmatūras produktu un tā dokumentāciju, īstenojot visus programmatūras izstrādes dzīves cikla posmus, izņemot lietotāju apmācību un uzturēšanu (līdz 15 A4 formāta lapām, neskaitot titullapu, satura rādītāju un pielikumu).

Piekļuves materiālus no 2023.gada 1. marta, bet ne vēlāk kā astoņas nedēļas pirms eksāmena norises skolēnam jāaugšupielādē Valsts pārbaudījumu sistēmā (<https://eksameni.vps.gov.lv>). Kārtība, kā augšupielādēt piekļuves materiālus, būs pieejama no 1. marta VPS lietotāju atbalsta dienesta tīmekļvietnē (<https://atbalsts.refined.site/space/VPS>).

Piekļuves materiālus izglītības iestādes skolotājs vērtē ballēs atbilstoši izglītības iestādes saistošai izglītojamo mācību sasniegumu vērtēšanas kārtībai un ne vēlāk kā sešas nedēļas pirms eksāmena norises dienas vērtējumu ievada VPS. Izglītojamais eksāmenu drīkst kārtot, ja vērtējums par piekļuves materiālu nav zemāks par četrām ballēm.

Izglītojamie, kuri eksāmenu kārtos augstskolā, piekļuves materiālus neiesniedz.

Dokumentācijā jāiekļauj šādas nodaļas:

Darbā jāietver visi programmatūras dzīves posmi, izņemot lietotāju apmācību un uzturēšanu.

1. Problēmas izpēte un analīze – izpētes metodes izvēle un pamatojums, izpētes procesa apraksts, izpētes datu apkopojums.

SR: analizē dažādus ikdienas darba procesus, saskata tajos vai to daļās automatizācijas iespējas. (T.A.2.4.1.)

2. Programmatūras prasību specifikācija – risinājuma mērķauditorijas izvēle un tās raksturojums, programmatūras produkta un tā funkciju apraksts, programmatūras produkta skice.

SR: ikdienas darba procesos vai to daļās saskata automatizācijas iespējas un to, kā pasūtītājs formulē darba uzdevumu programmatūras izstrādātājam. (T.A.2.4.1.)

SR: sastāda vienkāršotu programmatūras prasību specifikāciju atbilstoši konkrētajam uzdevumam, izvērtējot mērķauditorijas specifiku un vajadzības. (T.A. 2.4.4.)

SR: veic vienkāršotu programmatūras projektēšanu (t. sk. lietotāju saskarnes un vienkāršotu datu modeļa izveidi) atbilstoši programmatūras prasību specifikācijā izvirzītajām funkcionālajām un nefunkcionālajām prasībām.
(T.A. 2.4.5.)

SR: izvēlas programmēšanas valodu un programmatūras izstrādes vidi programmatūras izstrādē atbilstoši uzdevuma specifikai, pamato savu izvēli. (T.A. 2.4.12.)

3. Programmatūras izstrādes plāns.

SR: salīdzina un izvēlas piemērotāko programmatūras izstrādes modeli konkrētā uzdevuma atrisināšanai, pamato savu izvēli. (T.A. 2.4.2.)

SR: veic vienkāršotu programmatūras izstrādes plānošanu (bez darbietilpības novērtējuma).
(T.A. 2.4.5.)

4. Atklādošanas un akcepttestēšanas pārskats.

SR: veic programmatūras vienību izstrādi un vienībutestēšanu, izstrādājamās programmatūras vienību apvienošanu, integrācijas un akcepttestēšanu atbilstoši izstrādātajai programmatūras prasību specifikācijai un projektējuma aprakstam. (T.A. 2.4.6.)

5. Lietotāja ceļvedis.

SR: izstrādā un prezentē izveidotās programmatūras vienkāršotu izvēršanas (t. sk. ieviešanas) plānu, lietotāja ceļvedi un uzturēšanas plānu, ievērojot tās lietotāju mērķauditorijas specifiku. (T.A.2.4.7.)

6. Piemērotās licences pamatojums.

SR: salīdzina atvērto kodu licences un to atšķirības, izmanto un piemēro atbilstošāko no licencēm savam programmatūras projektam. (T.A.3.1.4.)

**7. Programmatūras kods, kas veidots, ievērojot labās prakses principus (sk. 1. pielikumu).
Programmatūras kods pievienojams dokumentācijas pielikumā.**

SR: veido programmu, ievērojot labās prakses pieredzi tās pieraksta strukturēšanā un komentāru veidošanā. (T.A. 2.4.8.)

Minimālās prasības programmatūras produktam:

1. Pielieto datu bāzi ar vairākām tabulām;

SR: 2.3.2. Plāno datubāzi, t. sk. izveido ER modeli konkrētā uzdevuma datu apstrādes risinājumam.

SR: 2.4.17. Izveido vienkāršu datu apstrādes programmatūru (sistēmu), datu uzglabāšanai izmantojot paša veidotu datubāzi ar vairākām tabulām.

2. Pielieto vismaz vienu bibliotēku un/vai API;

SR: 2.4.10. Izmanto programmēšanas valodas un tās bibliotēku dokumentāciju un palīdzības sistēmu, lai patstāvīgi apgūtu citas to piedāvātās iespējas, kas nepieciešamas konkrētās programmatūras izstrādei.

SR: 2.4.11. Meklē un pievieno atvērtā koda bibliotēkas un lieto API (programmsaskarni) specializētu funkciju veikšanai sava programmēšanas projekta īstenošanai.

SR: 2.4.13. Izvēlas un lieto atbilstošas programmēšanas valodas konstrukcijas, datu tipus un dažādas bibliotēkas, veidojot programmas doto uzdevumu un problēmu risinājumam.

3. Pielieto vismaz vienu no minētajām datu struktūrām;

SR: 2.4.14. Izmanto dažādas datu struktūras (t. sk. masīvi, kopas, ieraksti, steks, rinda, saraksts, koks, grafs, datne) un ar tiem saistītos pamatalgoritmus.

SR: 2.4.19. Veido dotā uzdevuma (problēmas) risinājumu, izmantojot gatavus algoritmus un/vai pielāgojot vai kombinējot tos, un/vai izstrādājot jaunus algoritmus. Izprot un skaidro dažādu algoritmu darbību, pielāgo tos dažādām nestandarta situācijām, ja nepieciešams, veidojot jaunas datu struktūras.

4. Ir realizēta lietotāja piekļuves vai datu aizsardzība.

SR: 3.1.2. Izmanto kriptogrāfijas metodes konkrētā uzdevuma risinājumā.

5. Nepieciešamo resursu nodrošinājumu

Eksāmens veicams, izmantojot datoru.

Pie izglītojamajiem un personām, kuras piedalās eksāmena nodrošināšanā, no brīža, kad viņiem ir pieejams eksāmena materiāls, līdz eksāmena norises beigām nedrīkst atrasties ierīces (planšetdators, piezīmjdators, viedtālrunis, viedpulkstenis u. c. saziņas un informācijas apmaiņas līdzekļi), kuras nav paredzētas Valsts pārbaudes darbu norises darbību laikos.

Eksāmena 3. un 4. daļas programmēšanas uzdevumos:

operētājsistēmas pamatprogrammas un biroja programmatūru (piemēram, kalkulatoru, failu pārlūku, izklājprogrammu), interneta pārlūkprogrammu (tikai atļautajām darbībām), izstrādes un atklāšanas vides un kompilatorus.

6. Vērtēšanas kārtība un kritēriji

6.1. Vērtēšanas kārtība

Katrā uzdevumā ir norādīts maksimālais iegūstamo punktu skaits. Eksāmena vērtētājam ir pieejami kritēriji, pēc kuriem nosaka punktu skaitu, ko skolēns ieguvis. Skolēna rezultātus eksāmenā – iegūto punktu summu visā darbā, iegūto punktu summu katrā daļā – izsaka procentuālā novērtējumā. Vidēji 20% eksāmenā iekļauto testelementu reprezentē minimālo prasību kopumu – katra eksāmena satura moduļa izpildi atbilstoši 1. un 2. līmenim SOLO taksonomijā (piemēram, atpazīst pamatalgoritmus, nosauc piemērotāko izpētes metodi. Izpilda vienkāršas darbības,

piemēram, nodrošina datu ievadi/izvadi. Kombinē zināšanas un prasmes standarta situācijās, piemēram, izveido masīvu datu uzglabāšanai).

Atbilstoši Ministru kabineta 2019. gada 3. septembrī noteikumi Nr. 416 “Noteikumi par valsts vispārējās vidējās izglītības standartu un vispārējās vidējās izglītības programmu paraugiem” 25.¹1. punktam valsts pārbaudes darbā vērtējums nav iegūts, ja darba kopvērtējums 2022./2023. mācību gadā ir mazāks nekā 10 procenti.

6.2. Vērtēšanas kritēriji

Skolēnu sniegumu eksāmenā vērtē pēc kritērijiem, kas var būt izteikti kā katram punktam atbilstošu atbilžu, darbību, rezultāta apraksts. Programmēšanas labās prakses principu ievērošana tiek vērtēta pēc snieguma līmeņu apraksta (turpmāk – SLA), katram līmenim piešķirot noteiktu punktu skaitu (sk. 1. pielikumu).

Eksāmenā biežāk lietoto vai mācību procesā nereti dažādi interpretēto rīcības vārdu skaidrojums (sk. 5. tabulu) ir iekļauts, lai veidotu vienotu skolotāju un skolēnu izpratni par šo vārdu nozīmi un tai atbilstošu skolēna sniegumu mācību procesā un eksāmenā.

Rīcības vārdi ar piemēriem skaidroti 2. pielikumā.

5. tabula. Biežāk lietotie rīcības vārdi un to skaidrojums

Rīcības vārds	Skaidrojums
Analizē	Sīki, pamatīgi pēta, raksturo, piemēram, algoritmu, problēmas aprakstu.
Apvieno	Izveido vienu veselu (no atsevišķām vienībām, daļām, piemēram, apvieno programmatūras daļas).
Atpazīst	Pazīst, identificē (kādu/ko pēc noteiktām pazīmēm, parasti starp līdzīgiem, piemēram, OOP darbības principus).
Formulē	Izsaka (domu, atzinumu, lēmumu, jautājumu), piemēram, kādu programmatūras produktu nepieciešams izstrādāt.
Identificē	Nosaka, pazīst; konstatē (ko/kādu pēc kādām raksturīgām pazīmēm, piemēram, lomu sadalījumu projekta komandā).
Ievēro	Rīkojas saskaņā (ar ieteikumiem, norādījumiem u. tml.); iegaumē un izpilda (noteikumus, prasības u. tml.). Piemēram, labās prakses principus, rakstot programmatūras kodu.
Integrē	Iekļauj kādā vienībā; papildina, pilnveido, izveido par veselu vienību (piemēram, programmatūras daļas vienā veselumā).
Izmanto	Lieto (ko) par pamatu, materiālu (piemēram, kā noteikta gatavošanai, izveidošanai), piemēram, programmēšanas valodas dokumentāciju.
Izprot	Pilnīgi izzina un saprot, piemēram, algoritma darbību.
Izveido/Izstrādā	Ar mērķtiecīgu darbību panāk vai organizē, ka, piemēram, datubāze (vai kas cits) iegūst vēlamu veidu un atbilst noteiktām prasībām. Rada prasīto.
Izvēlas	Iegūst lietošanai, atrod, izraugās izmantošanai no kāda kopuma (ko piemērotu, atbilstošu, piemēram, programmatūras izstrādes modeli savam projektam atbilstoši problēmai, nosacījumiem u. tml.).
Izvērtē	Novērtē kādu kopumu, parasti pa tā sastāvdaļām, detaļām; katru faktu, notikumu un tml. noteiktā kopsakarībā (piemēram, mērķauditorijas specifiku un vajadzības).
Kombinē	Veido noteiktā veselumā (no atsevišķām sastāvdaļām), piemēram, vairākas algoritma daļas savienojot vienā veselumā.
Konfigurē	Pielāgo sistēmu (piemēram, datortīklu), uzstādot aparāturu un programmatūru (piemēram, maršrutētāja); noteiktā veidā uzstāda.
Lieto	Rīkojas (ar materiālu, koda fragmentu) mērķtiecīgi, pārdomāti – tā, lai gūtu labumu; izmanto (piemēram, lieto programsaskarni).
Meklē	Cenšas (skatoties u. tml.) dabūt (ko vajadzīgu, nepieciešamu u. tml., piemēram, atvērtā koda bibliotēkas).
Pamato	Min faktus, cēloņus, loģiskus slēdzienus u. tml., kas pierāda (kā) patiesumu, nepieciešamību (piemēram, izvēli lietot konkrētu izpēti metodi lietotāju vajadzību noskaidrošanai).
Pārbauda	Pārlicinās, vai (kas) atrodas noteiktā stāvoklī, kārtībā (piemēram, programmatūras veiktspēju, programmatūras kodu), arī – vai kas ir noticis, izdarīts (piemēram, iegūts pareizais rezultāts, izpildot programmu).
Pielāgo	Veido, izveido (ko, piemēram, algoritmu) atbilstoši izmantošanas mērķim, apstākļiem.
Piemēro	Padara atbilstošu (piemēram, apstākļiem, iespējām programmatūras licenci).

Pievieno	Pieliek (pie kā, kam klāt, piemēram, atvērtā koda bibliotēkas).
Plāno	Veido (kāda objekta, piemēram, datubāzes) plānu, skici, shēmu. Iepriekš sadala (laiku) pa posmiem (kā veikšanai, piemēram, projekta īstenošanai).
Raksturo	Nosaka, apraksta, vērtē (kā) raksturīgās īpašības, pazīmes. Piemēram, mašīnmācīšanās algoritmus.
Salīdzina	Pēta, analizē, novēro u. tml. (divus vai vairākus priekšmetus, blokshēmas u. tml.), lai noteiktu (to) kopīgās vai atšķirīgās īpašības, pazīmes.
Saskata	Skatoties, pilnīgi, precīzi uztver (ko, piemēram, automatizācijas iespējas), atšķir, pazīst (ko); skatoties atklāj, konstatē (ko), secina (ko).
Sastāda	Izveido (piemēram, programmatūras prasību specifikāciju), apvienojot materiālus; uzraksta noteiktā formā, izmantojot kādus materiālus.
Skaidro	Stāstot, rakstot, arī rādot panāk, ka kādam (kas) kļūst zināms, saprotams (piemēram, programmatūras koda darbības principi).
Veic	Ar savu darbību, rīcību panāk, ka īstenojas (piemēram, kāda norise), tiek sasniegts (kāds mērķis). Piemēram, veic akcepttestēšanu.

7. Palīg līdzekļi, kurus atļauts izmantot eksāmena laikā

Skolēnam tiek atļauts izmantot izvēlētās programmēšanas valodas dokumentāciju, kura ir iekļauta datorā instalētajā izstrādes vidē, pašā programmēšanas valodā vai lejupielādēta no programmēšanas valodas izstrādātāju mājas lapas, tīmekļa resurss “w3schools”.

Pie izglītojamajiem un personām, kuras piedalās eksāmena nodrošināšanā, no brīža, kad viņiem ir pieejams eksāmena materiāls, līdz eksāmena norises beigām nedrīkst atrasties ierīces (planšetdators, piezīmjdators, viedpulkstenis u. c. saziņas un informācijas apmaiņas līdzekļi), kuras nav paredzētas Valsts pārbaudes darbu norises darbību laikos.

PIELIKUMI

1. pielikums. Programmēšanas labās prakses princips

Labās prakses principi ir:

- katru atsevišķu priekšrakstu raksta jaunā rindā;
- koda loģiskās daļas (piemēram, zarošanos, ciklu, masīvu) savstarpēji atdala ar tukšu rindu;
- lieto atkāpes, lai vizualizētu priekšrakstu vai struktūru iekļaušanu citās struktūrās;
- izvairās no garām koda rindām. Lieto pārnēsi jaunā rindā, atvieglot koda lasīšanu;
- mainīgo, funkciju u. c. nosaukumus veido jēgpilnus, atvieglot koda uztveri (piemēram, perimetru apzīmējot nevis ar “a”, bet “perim” vai “perimetru”);
- ar komentāriem skaidro programmatūras koda loģiskās daļas (piemēram, zarošanos, ciklu, masīvu), to lomu programmatūrā (piemēram “Datu izvade”).

Kritērijs/Līmenis	Sācis apgūt	Turpina apgūt	Apguvis
Veido programmatūru, ievērojot labās prakses principus tās pieraksta strukturēšanā un komentāru veidošanā.	Lielākajā daļā programmatūras koda lieto labās prakses principus, bet dara to nekonsekventi vai daļēji korekti.	Labās prakses principus programmatūras kodā lieto kopumā korekti un konsekventi, pieļaujot dažas neprecizitātes.	Labās prakses principus programmatūras kodā lieto korekti un konsekventi.

2. pielikums Rīcības vārdu skaidrojumi ar uzdevumu piemēriem

Tabulas virsrakstu skaidrojums:

- Sasniedzamais rezultāts (SR) – atbilstoši standartam.
- SOLO – maksimālais SR pārbaudes dziļums atbilstoši SOLO taksonomijai. Eksāmenā sasniedzamo rezultātu var pārbaudīt zemākā, bet ne augstākā SOLO līmenī.
- Rīcības vārdi – SR iekļautie darbības (rīcības) vārdi.
- Uzdevumu veidu piemēri – rīcības vārdu interpretācija dažādu veidu uzdevumu piemēros. Piemēru saraksts nav pilnīgs.
- Komentāri – papildu informācija uzdevumu veidu piemēriem.

Sasniedzamais rezultāts	SOLO	Rīcības vārdi	Uzdevumu veidu piemēri	Komentāri
2.3.2. Plāno datubāzi, t. sk. izveido ER modeli konkrētā uzdevuma datu apstrādes risinājumam.	3	Plāno	Uzzīmē uz papīra vai lietotnē (t. sk. izklājlappās) datubāzes shēmu ar vairākām tabulām un relācijām.	

Sasniedzamais rezultāts	SOLO	Rīcības vārdi	Uzdevumu veidu piemēri	Komentāri
2.4.1. Analizē dažādus ikdienas darba procesus, saskata tajos vai to daļās automatizācijas iespējas un kā pasūtītājs formulē darba uzdevumu programmatūras izstrādātājam.	3	Analizē un saskata	Iepazīstas ar problēmas aprakstu teksta formātā un uzskaita dažādas procesu automatizācijas iespējas. Ir dots problēmas apraksts un varianti automatizācijas risinājumiem – identificē, kuri no tiem atbilst dotajai situācijai un kuri neatbilst.	Dizaina domāšanas principi un soļi – izpēte, problēmas definēšana, ideju ģenerēšana.
		Formulē	Balstoties uz dotu problēmas aprakstu, formulē, kādu programmatūras produktu nepieciešams izstrādāt. Uzskaita un apraksta tehniskās prasības programmatūras produkta izstrādei.	
2.4.2. Salīdzina un izvēlas piemērotāko programmatūras izstrādes modeli dotā uzdevuma atrisināšanai, pamato izvēli.	3	Salīdzina	Atpazīst un raksturo vienu vai vairākus programmatūras izstrādes modeļus. Uzskaita ieguvumus, riskus konkrētu programmatūras izstrādes modeļu izmantošanā un salīdzina divus vai vairākus programmatūras izstrādes modeļus.	
		Izvēlas un pamato	Ņemot vērā problēmas aprakstu, izvēlas un pamato, kuru programmatūras izstrādes modeli izmantot. Plāno projekta izstrādes procesu, iekļaujot tajā programmatūras izstrādes modeli un paskaidrojumu par tā izvēli.	

Sasniedzamais rezultāts	SOLO	Rīcības vārdi	Uzdevumu veidu piemēri	Komentāri
2.4.4. Sastāda vienkāršotu programmatūras prasību specifikāciju atbilstoši konkrētajam uzdevumam,	4	Izvērtē	Definē un raksturo mērķauditoriju atbilstoši dotajam uzdevumam. Atbilstoši dotajai problēmai identificē un apkopo informāciju par mērķauditorijas specifiku un vajadzībām.	

izvērtējot mērķauditorijas specifiku un vajadzības.			Pielāgo programmatūras produkta ideju dažādām mērķauditorijām.	
		Sastāda programmatūras prasību specifikāciju (PPS)	Balstoties uz dotu problēmas aprakstu, izstrādā PPS vai kādas tās daļas. Papildina iesāktu PPS.	
		Veic akcepttestēšanu	Veic akcepttestēšanu atbilstoši izstrādātajām programmatūras prasībām. Fiksē atbilstību.	

Sasniedzamais rezultāts	SOLO	Rīcības vārdi	Uzdevumu veidu piemēri	Komentāri
2.4.8. Veido programmu, ievērojot labās prakses pieredzi tās pieraksta strukturēšanā un komentāru veidošanā, vienojoties par vienotu stilu ar visiem grupas dalībniekiem.	4	Ievēro labās prakses principus	<p>Veido programmatūru, ievērojot labās prakses principus:</p> <ul style="list-style-type: none"> katru atsevišķu priekšrakstu raksta jaunā rindā; koda loģiskās daļas (piemēram, zarošanos, ciklu, masīvu) savstarpēji atdala ar tukšu rindu; lieto atkāpes, lai vizualizētu priekšrakstu vai struktūru iekļaušanu citās struktūrās; izvairās no garām koda rindām. Lieto pārnesei jaunā rindā, atvieglējot koda lasīšanu; mainīgo, funkciju u.c. nosaukumus veido jēgpilnus, atvieglējot koda uztveri (piemēram, perimetru apzīmējot nevis ar “a”, bet “perim” vai “perimetru”); ar komentāriem skaidro programmatūras koda loģiskās daļas (piemēram, zarošanos, ciklu, masīvu), to lomu programmatūrā (piemēram “Datu izvade”). 	

2.4.10. Izmanto programmēšanas valodas un tās bibliotēku dokumentāciju un palīdzības sistēmu, lai patstāvīgi apgūtu citas to piedāvātās iespējas, kas nepieciešamas konkrētās programmatūras izstrādei.	4	Analizē un izmanto citas piedāvātās iespējas	Izstrādājot programmatūru, jēgpilni izmanto esošo bibliotēku piedāvātās iespējas. Lieto esošo klašu konstruktorus un to metodes, tādējādi saīsinot esošo programmatūras kodu un uzlabojot izstrādājamās programmatūras darba efektivitāti. Analizē doto bibliotēku dokumentāciju un palīdzības sistēmu informāciju, lai atrastu un pielietotu atbilstošu bibliotēku problēmas risināšanā. Izmanto programmēšanas valodas dokumentāciju, lai apgūtu programmēšanas valodas iespējas dotās problēmas risināšanā (t. sk. sintakses pārbaudē).	
2.4.11. Meklē un pievieno atvērtā koda bibliotēkas un lieto API (programmsaskarni) specializētu funkciju veikšanai sava programmēšanas projekta īstenošanai.	4	Meklē	Atrod atvērtā koda bibliotēku, kas veic nepieciešamo funkciju, un pamato izvēli.	Piemēram, pamato atbildi ar saiti un citātu no dokumentācijas.
		Pievieno	Pievieno paša atrastu bibliotēku. Pievieno dotu bibliotēku.	
		Lieto	Pielieto bibliotēku un/vai API dotā uzdevuma risinājumam.	

Sasniedzamais rezultāts	SOLO	Rīcības vārdi	Uzdevumu veidu piemēri	Komentāri
2.4.14. Izmanto dažādas datu struktūras (t. sk. masīvi, kopas, ieraksti, steks, rinda, saraksts, koks, grafs, datne) un ar tiem saistītos pamatalgoritmus.	3	Izmanto	Izveido risinājumu dotajam uzdevumam, izmantojot viendimensijas un divdimensiju masīvus, datnes, vārdnīcas.	

2.4.15. Skaidro objektorientētās programmēšanas pamatprincipus, veido programmas vienā no objektorientētajām programmēšanas valodām.	4	Skaidro OOP pamatprincipus	Skaidro klašu, īpašību un metožu izveides un pielietošanas iespējas: • uz papīra vai lietotnē uzraksta pseidokodu ar piemēriem (dota problēma).	
		Veido programmas	Veido programmatūras produktu, balstoties uz OOP pamatprincipiem – izveido un pielieto klasi un tās īpašības, metodes un konstruktorus.	

Sasniedzamais rezultāts	SOLO	Rīcības vārdi	Uzdevumu veidu piemēri	Komentāri
2.4.17. Izveido vienkāršu datu apstrādes programmatūru (sistēmu).	4	Izveido	Izveido programmatūru.	

3. pielikums Projekta izstrādes dokumentācijas snieguma līmeņu apraksts

SLA var tikt izmantots, izstrādājot projektus vai to daļas optimālajā un padziļinātajā līmenī veicot pašnovērtējumu, summatīvo vai formatīvo vērtēšanu mācību procesā, eksāmena piekļuves nosacījumu izpildes vērtēšanai.

Kritērijs/ Līmenis	Sācis apgūt	Turpina apgūt	Apguvis	Apguvis padziļināti
	Problēmas izpēte un analīze			
Izpētes metodes	Nosauc un raksturo izpētes metodes (piemēram, interviju, priekšizpēti, novērojumu u. c.). Trūkst izpētes metožu pozitīvo iezīmju un trūkumu uzskaitījums. Izpētes metodes pamatojumam nav saistība ar doto problēmu.	Nosauc un raksturo izpētes metodes, minot katras izpētes metodes pozitīvās iezīmes un trūkumus. Izpētes metodes izvēles pamatojums daļēji ir saistīts ar doto problēmu.	Izpētes metodes izvēles pamatojums ir balstīts uz pozitīvo iezīmju un trūkumu salīdzinājumu un ņemot vērā doto problēmu.	

Izpētes process	Izpētes procesa soļi ir sajauktā secībā un/vai neveido loģisku procesu (iztrūkst kāds būtisks posms, nav sākuma vai noslēguma u. tml.).	Aprakstā ir mazāk par pieciem procesa soļiem, tie ir loģiskā secībā un atspoguļo izpētes procesu no sākuma līdz galam – piemēram, sagatavošanās, norise un secinājumu izdarīšana (vai tml.).	Aprakstā iekļautā izpētes procesa gaita ir loģiska, detalizēta un sastāv no vismaz pieciem soļiem – piemēram, sagatavošanās, norise, rezultātu apkopošana, izvērtēšana un secinājumu izdarīšana.	
Izpētes dati	Pievienotie dati daļēji atbilst pētāmajai problēmai un ir pievienoti pielikumā.	Dati atbilst pētāmajai problēmai un ir pievienoti pielikumā.	Dati atbilst pētāmajai problēmai, ir pievienoti pielikumā un ir strukturēti.	
	Programmatūras prasību specifikācija			
Programmatūras produkta mērķauditorija	Mērķauditorija ir definēta, neiedziļinoties dotajā problēmā.	Ir definēta, tomēr iespējams to konkretizēt.	Tiek definēta konkrēta risinājuma mērķauditorija.	
Mērķauditorijas vajadzību raksturojums	Vispārīgs, neiedziļinoties mērķauditorijas vajadzībās.	Daļa no vajadzībām ir saistītas ar mērķauditoriju un doto problēmu.	Vajadzības ir saistītas ar mērķauditoriju un doto problēmu.	

Programmatūras produkta apraksts	Ļoti vispārīgs. Nesniedz ieskatu, kādas tehnoloģijas būtu jāizmanto.	Ir aprakstīts, kas ir izstrādājama produkta, tomēr nav konkrēti uzskaitītas tehnoloģijas, kas tiks izmantotas tā izstrādē.	Ir aprakstīts, kas ir izstrādājama produkta un kādas tehnoloģijas nepieciešamas tā izveidē.	
Programmatūras produkta funkciju apraksts	Funkciju apraksts ir vispārīgs.	Funkcijas ir nosauktas (piemēram, lietotāja autentifikācija, datu atlasē filtrs u. tml.), raksturotas un daļēji atbilst mērķauditorijas vajadzībām.	Funkcijas ir nosauktas, raksturotas un atbilst mērķauditorijas vajadzībām.	
Programmatūras produkta skice	Programmatūras produkta skice daļēji atbilst aprakstam un funkciju prasībām. Tajā iekļautā informācija ir grūti uztverama.	Programmatūras produkta skice daļēji atbilst aprakstam un funkciju prasībām.	Atbilst aprakstam un funkciju prasībām. Atbilstoši konkrētā darba vajadzībām izstrādāts: programmatūras struktūras, procesu plūsmas skice; lietotāja saskarnes skice; datubāzes skice.	Papildu izstrādāts: datu struktūras skice; datu uzglabāšanas un drošības skice.
	Programmatūras izstrādes plāns			

Programmatūras izstrādes plāns	Izstrādāts plāns, norādot programmatūras izstrādes dzīves cikla posmus. Atsevišķiem programmatūras dzīves cikla posmiem norādīti galvenie veicamie uzdevumi vai to nav vispār.	Izstrādāts plāns, norādot atsevišķiem programmatūras izstrādes dzīves cikla posmiem paredzēto laiku. Atsevišķiem programmatūras dzīves cikla posmiem norādīti galvenie veicamie uzdevumi vai to nav vispār.	Izstrādāts plāns, norādot katram programmatūras izstrādes dzīves cikla posmam paredzēto laiku. Katram programmatūras dzīves cikla posmam norādīti galvenie veicamie uzdevumi. Plāns ir realizējams paredzētajā laikā.	Izstrādāts plāns, norādot katram programmatūras izstrādes dzīves cikla posmam paredzēto laiku. Katram programmatūras dzīves cikla posmam norādīti galvenie veicamie uzdevumi, apakšuzdevumi un tiem paredzētais laiks.
	Atklūdošanas un akcepttestēšanas pārskats			
Testēšanas plāns un izpilde	Testēšanas plāns paredz pārbaudīt tikai atsevišķas sistēmas daļas.	Testēšanas plāns ir nepilnīgs, tajā trūkst informācijas par atsevišķu sistēmas daļu testēšanu. Ne visas specifikācijā minētās programmatūras produkta funkcijas tiek pārbaudītas.	Testēšanas plānā noteikts, kuras sistēmas daļas un kādā veidā jātestē. Nosaka nepieciešamo rezultātu atbilstoši specifikācijā esošajam programmatūras produkta funkciju aprakstam.	Testēšanas plānā noteikts, kuras sistēmas daļas un kādā veidā jātestē. Nosaka nepieciešamo rezultātu atbilstoši programmatūras produkta funkciju aprakstam. Satur plāna izpildes atspoguļojumu (piemēram, ekrānuzņēmumus no testēšanas procesa).

Akcepttestēšanas pārskats	Akcepttestēšanas pārskats ir vispārīgs un satur virspusēju programmatūras prasību specifikācijā esošo prasību pārbaudi no lietotāja skatu punkta.	Akcepttestēšanas pārskats ir vispārīgs un satur lielāko daļu programmatūras prasību specifikācijā esošo prasību pārbaudi no lietotāja skatu punkta.	Atspoguļota sistēmas darbības pārbaude no lietotāja skatu punkta atbilstoši specifikācijai.	Atspoguļota sistēmas darbības pārbaude no lietotāja skatu punkta atbilstoši specifikācijai. Atspoguļota sistēmas uzlabojumu veikšana un/vai papildu funkciju izstrāde atbilstoši lietotāju vēlmēm.
	Lietotāja ceļvedis			
Lietotāja ceļvedis	Izveidots lietotāja ceļvedis, kura struktūra ir juceklīga. Dokumenta saturs vispārīgi apskata daļu programmatūras produkta funkciju.	Izveidots lietotāja ceļvedis: kam daļēji pievienota vizuāla informācija; dokumenta struktūra un saturs ir juceklīgs.	Izveidots lietotāja ceļvedis, kas atbilst šādiem kritērijiem: pievienota vizuāla informācija; loģiska dokumenta struktūra, viegli saprotamas sadaļas; iespēja viegli atrast nepieciešamo informāciju.	Izveidots lietotāja ceļvedis, kas atbilst šādiem kritērijiem: iespējami vienkārša un saprotama valoda; pievienota vizuāla informācija; loģiska dokumenta struktūra, viegli saprotamas sadaļas; iespēja viegli atrast nepieciešamo informāciju; iekļauta tikai nepieciešamā informācija.

	Piemērotās licences pamatojums			
Produkta licence	Licence ir norādīta, bet tās izvēle nav pamatota.	Norādīta licence, tās izvēles pamatojums ir vispārīgs.	Norādīta atbilstoša licence un tās izvēle ir pamatota (piemēram, specifikācijā).	
	Programmatūras kods, kas veidots ievērojot labās prakses principus (1. pielikumā)			
Veido programmatūru, ievērojot labās prakses principus 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.	Programmatūras kodā labās prakses principus lieto korekti un konsekventi.	