Centralizētais eksāmens par vispārējās vidējās izglītības apguvi
PROGRAMMĒŠANA
(augstākais mācību satura apguves līmenis)
Vērtēšanas kritēriji

1. daļa. Programmatūras dzīvescikls (kopā 20 punkti)

Uzd. nr.	Vērtēšanas kritēriji	Punkti	Atbilžu piemēri (no kārtotāju darbiem)	0 punktu par atbildi, ja…
1. uzdevums				
1.	1.1.1. **Identificē problēmsituācijas aprakstā <u>vismaz vienu</u> mērķauditoriju.**	1	• Sportisti • Sporta kluba vadības personāls • Sporta kluba apmeklētāji • Cilvēki, kas ir ieinteresēti apmeklēt zāli *Pieņemt arī citas līdzīgas interpretācijas.*	• Lietotāji
	1.1.2. **Identificē problēmsituācijas aprakstā <u>trīs</u> mērķauditorijas.**	1		
	1.2.1. **Formulē vajadzību <u>vismaz vienai</u> identificētajai mērķauditorijai.**	1	• Pārvaldīt un apskatīt sistēmas un darbinieku datus • Apskatīt treniņu grafiku un sacensību rezultātus • Sekot līdzi savam grafikam • Pieteikties nodarbībām *Pieņemt arī citas līdzīgas interpretācijas.*	• Ir sniegta nepareizi noformulēta atbilde
	1.2.2. **Formulē vajadzības <u>trīs</u> identificētajām mērķauditorijām.**	1		
	1.3.1. **Formulē izpētes jautājumu <u>vismaz vienai</u> identificētajai mērķauditorijai.**	1	• Kādas darbības ar kluba informāciju tiek veiktas visbiežāk? • Ko Jums ir svarīgi zināt, piesakoties nodarbībām? • Kā pieteikšanos nodarbībām padarīt ērtāku? • Kā trūkst treniņu un rezultātu uzskaites sistēmā? *Pieņemt arī citas līdzīgas interpretācijas.*	• Kā programma sportistiem var palīdzēt izplānot savu dienu?
	1.3.2. **Formulē izpētes jautājumus <u>trīs</u> identificētajām mērķauditorijām.**	1		

Uzd. nr.	Vērtēšanas kritēriji	Punkti	Atbilžu piemēri (no kārtotāju darbiem)	0 punktu par atbildi, ja…
	1.4.1. **Izvēlas <u>vismaz vienu</u> piemērotāko izpētes metodi mērķauditorijas vajadzības noskaidrošanai.**	1	• Intervija • Aptauja • Datu analīze *Pieņemt arī citas līdzīgas interpretācijas.*	• Uzrakstīta nevis izpētes metode, bet testēšanas vai prototipēšanas metode • SQL struktūra • Vārdnīca, kas ir aizsargāta
	1.4.2. **Izvēlas <u>trīs</u> piemērotākās izpētes metodes mērķauditoriju vajadzību noskaidrošanai.**	1		
	1.5.1. **Pamato <u>vismaz vienu</u> izvēlēto izpētes metodi.**	1	• Tā kā ir jāaptaujā liela cilvēku grupa, individuāli intervēt katru cilvēku aizņemtu daudz laika, savukārt anketēšana saglabā anonimitāti, paturot spēju katrai personai izpaust savu patieso viedokli *Pieņemt arī citas līdzīgas interpretācijas.*	• Jo vārdnīcā var uzglabāt datus, ko var lasīt u. tml.
	1.5.2. **Pamato <u>trīs</u> izvēlētās izpētes metodes.**	1		
2. uzdevums				
2.	2.1. **Uzraksta, kurā no minētajiem posmiem veikta <u>vajadzību izpēte</u>.**	1	• Prasību analīze	• Jebkurš cits posms no uzdevumā minētajiem • Jebkurš cits posms no uzdevumā neminētajiem
	2.2. **Uzraksta, kas <u>vēl</u> tiek veikts posmā, kurā veikta vajadzību izpēte.**	1	• Apkopo un dokumentē programmatūras prasības *Pieņemt arī citas līdzīgas interpretācijas.*	• Programmēšana • Testēšana • Jebkuras citas darbības, kas saistītas ar citiem posmiem
	2.3. **Uzraksta <u>nākamo</u> posmu un apraksta, kas notiek šajā nākamajā posmā.**	1	• Plānošana – Izveido detalizētu projekta plānu, kurā izklāstīti uzdevumi, laika grafiks un budžets • Dizains un projektēšana – Izveido detalizētu sistēmas specifikāciju, lietotāja saskarni un darbplūsmas	• Apraksts nav saistīts ar uzrakstīto posmu, nav korekts pēc būtības

Uzd. nr.	Vērtēšanas kritēriji	Punkti	Atbilžu piemēri (no kārtotāju darbiem)	0 punktu par atbildi, ja…
3. uzdevums				
3.	3.1. **Piedāvā plānotajam programmrisinājumam <u>vienu</u> funkcionālo prasību un <u>pamato tās nepieciešamību</u>.**	1	• Programma ļauj vadībai pārvaldīt darbinieku datus – Vadībai ir svarīgi mainīt darbinieku datus un uzzināt informāciju par darbinieku nostrādātajām stundām un atvaļinājumiem • Programma ļauj treneriem mainīt treniņu grafikus – Treniņu grafiki var mainīties atkarībā no treneru plāniem, ir svarīgi ātri paziņot šo informāciju un par tās izmaiņām sportistiem • Programma ļauj sportistiem un citiem interesentiem pieteikties nodarbībām – Tā ir viena no kluba vēlmēm, kas ir minēta problēmsituācijas aprakstā	• Piedāvātā prasība nav funkcionālā prasība vai nav saistīta ar doto problēmsituācijas aprakstu • Piedāvātās prasības nepieciešamības pamatojums nav objektīvs, pietiekams vai saistīts ar doto problēmsituācijas aprakstu
	3.2. **Piedāvā plānotajam programmrisinājumam <u>vienu</u> funkcionālo prasību un <u>pamato tās nepieciešamību</u>.**	1		
	3.3. **Piedāvā plānotajam programmrisinājumam <u>vienu</u> funkcionālo prasību un <u>pamato tās nepieciešamību</u>.**	1		

Uzd. nr.	Vērtēšanas kritēriji	Punkti	Atbilžu piemēri (no kārtotāju darbiem)	0 punktu par atbildi, ja…
4. uzdevums				
4.	4.1. **Uzraksta un pamato programmrisinājuma izstrādei piemērotāko programmatūras izstrādes modeli.**	1	• Spējais jeb *Agile* modelis/metodoloģija • Spirāles jeb *Spiral* modelis/metodoloģija • Ūdenskrituma jeb *Waterfall* modelis/metodoloģija • *Scrum* satvars	• Pamatojums nav saistīts ar doto problēmsituācijas aprakstu, nav objektīvs vai, piemēram, ir balstīts tikai uz to, ka uzrakstītais modelis ir plaši izplatīts, populārs u. tml. • Kā ūdenskrituma modeļa pamatojums ir minēta iespēja risināt problēmas uzreiz pēc to rašanās
	4.2. **Apraksta uzrakstītā programmatūras izstrādes modeļa raksturīgākās iezīmes.**	1	• *Agile* – Spēja ātri pielāgoties prasību izmaiņām mazina risku, ja bibliotēkas koncepcija deviņu mēnešu laikā mainās • *Spiral* – Katrs cikls sākas ar risku analīzi un beidzas ar klienta veiktu pārskatīšanu, tāpēc tas ir piemērots plašiem, ar risku piesātinātiem projektiem, kuros jāievēro drošības un autortiesību prasības • Iteratīvais – Cikliskā attīstība ļauj agri pamanīt funkcionālās vai dizaina kļūdas un izlabot tās ar mazākām izmaksām • Ūdenskrituma – Galvenais mērķis un gala termiņš tiek noteikts jau sākumā, un tas palīdz noturēt fokusu laika ierobežotā ieviešanas grafikā *Atbilde ir jāvērtē sasaistē ar 4.1. uzdevumā sniegto atbildi.*	• Modeļa raksturīgās iezīmes nav saistītas ar iepriekš minēto modeli • Kā ūdenskrituma modeļa raksturīgākās iezīmes ir minēts vienkāršā momentā izpildāms darbs

Uzd. nr.	Vērtēšanas kritēriji	Punkti	Atbilžu piemēri (no kārtotāju darbiem)	0 punktu par atbildi, ja…
5. uzdevums				
5.	5.1. **Piedāvā programmrisinājuma implementācijas veidu.**	1	Praktiskās implementācijas veidi, piemēram: • Darbvirsmas lietotne – Tā ļautu kluba administrācijas darbiniekiem ērti pārvaldīt informāciju par darbiniekiem, treniņiem un līgumiem, izmantojot darba vietās izvietotos darba datorus • Tiešsaistes lietotne – Tā nodrošinātu centralizētu piekļuvi kluba informācijai no dažādām ierīcēm un atrašanās vietām, kas ir īpaši svarīgi, jo klubs darbojas vairākās ēkās • Viedtālruņa lietotne – Tā ļautu sportistiem un citiem interesentiem jebkurā vietā ērti apskatīt treniņu grafiku, pieteikties nodarbībām un saņemt tūlītējus paziņojumus • Tīmekļvietne – Tā nodrošinātu plašam interesentu lokam piekļuvi informācijai par kluba pakalpojumiem un iespēju pieteikties nodarbībām neatkarīgi no izmantotās ierīces un operētājsistēmas Projekta implementācijas veidi, piemēram: • Soli pa solim – Pakāpeniska sistēmas funkcionalitātes ieviešana ļautu darbiniekiem pielāgoties jaunajam risinājumam un samazinātu kluba ikdienas darbības traucējumu risku • Sprādzienveida – Ja izstrādātais risinājums ir salīdzinoši vienkāršs un pilnībā pārbaudīts, to varētu ieviest visā klubā vienlaikus, izvairoties no ilgstošas pakāpeniskas ieviešanas	• Norādītas tikai konkrētas programmēšanas valodas, tehnoloģijas vai izstrādes modeļi, metodoloģijas bez sasaistes ar problēmsituācijas aprakstu, piemēram: “*Agile* – Ar *Agile* var ērti strādāt ar klientiem un pielāgot programmatūru mainīgām prasībām” • Pamatojums nesasaista konkrētu implementācijas veidu ar problēmsituācijā aprakstītajām sporta kluba vajadzībām vai tikai vispārīgi raksturo izvēlēto tehnoloģiju priekšrocības, piemēram: “*Python*, *Django*, *React*, *MongoDB* un *SQLite* – Šīs tehnoloģijas ir mūsdienīgas un ar tām ērti strādāt programmētājiem”
	5.2. **Pamato piedāvāto programmrisinājuma implementācijas veidu.**	1		

2. daļa. Datubāzes izstrāde un izmantošana (kopā 20 punkti)

Uzd. nr.	Vērtēšanas kritēriji	Punkti
1. uzdevums		
1.	1.1. **Izveido <u>vismaz divas</u> tabulas.**	1
	1.2. **Nodrošina, ka <u>vismaz divās</u> tabulās ir iekļauti visi minimāli nepieciešamie lauki, kas ļauj saglabāt visu doto informāciju.**	1
	1.3. **Nodrošina, ka <u>vismaz divās</u> tabulās visiem laukiem ir norādīti atbilstošākie datu tipi.**	1
2. uzdevums		
2.	2.1. **Pievieno tabulām "SPORTISTI", "TRENERI" un "SACENSIBAS" primāro atslēgu laukus ar atbilstošākajiem datu tipiem.**	1
	2.2. **Pievieno <u>vismaz divām</u> tabulām relācijas izveidei nepieciešamos laukus un šiem laukiem izvēlas atbilstošāko datu tipu.**	1
	2.3. **Pievieno <u>visām</u> tabulām relāciju izveidei nepieciešamos laukus un šiem laukiem izvēlas atbilstošākos datu tipus.**	1
	2.4. **Norāda <u>pirmajā</u> relācijā iesaistītās tabulas un laukus.**	1
	2.5. **Norāda <u>otrajā</u> relācijā iesaistītās tabulas un laukus.**	1
	2.6. **Norāda <u>trešajā</u> relācijā iesaistītās tabulas un laukus.**	1
	2.7. **Norāda <u>visām</u> relācijām tipus.**	1
3. uzdevums		

3.1.	**Izveido vaicājumu, kas atgriež visu informāciju par treneriem, kuru specializācija ir daudzcīņa.**
	
	Pareizās atbildes piemērs:
	`SELECT *`
	`FROM TRENERI`
	`WHERE specializacija = 'daudzcina';`
		1

3.2.	**Izveido vaicājumu, kas atgriež tāllēkšanas treneru skaitu.**
	
	Pareizās atbildes piemērs:
	`SELECT COUNT(specializacija)`
	`FROM TRENERI`
	`WHERE specializacija = 'talleksana';`
		1

Uzd. nr.	Vērtēšanas kritēriji	Punkti
3.3.	3.3.1. **Izveido vaicājumu, kas atgriež viena jaunākā sportista vārdu un uzvārdu.**	1

	3.3.2. **Nodrošina, ka izveidotais vaicājums atgriež visu (vairāku) jaunāko sportistu vārdus un uzvārdus.**
	
	Pareizās atbildes piemērs:
	`SELECT vards, uzvards`
	`FROM SPORTISTI`
	`WHERE dz_gads = (`
	`  SELECT MAX(dz_gads)`
	`  FROM SPORTISTI`
	`);`
		1

3.4.	3.4.1. **Izveido vaicājumu, kas atgriež treneru un viņu audzēkņu vārdus un uzvārdus.**	1
	3.4.2. **Nodrošina, ka izveidotā vaicājuma rezultātā katram sportistam norādīts viņa treneris.**	1

	3.4.3. **Nodrošina, ka izveidotā vaicājuma rezultāts sakārtots alfabētiskā secībā pēc trenera uzvārda.**
	
	Pareizās atbildes piemērs:
	`SELECT TRENERI.vards, TRENERI.uzvards, SPORTISTI.vards, SPORTISTI.uzvards`
	`FROM TRENERI`
	`JOIN SPORTISTI ON TRENERI.trenera_ID = SPORTISTI.trenera_ID`
	`ORDER BY TRENERI.uzvards;`
		1

3.5.	3.5.1. **Izveido vaicājumu, kas atgriež katra sportista vārdu, uzvārdu un kopējo treniņu laiku.**	1
	3.5.2. **Nodrošina, ka izveidotā vaicājuma rezultātā katram sportistam norādīts viņa kopējais treniņu laiks.**	1

	3.5.3. **Nodrošina, ka izveidotais vaicājums atgriež arī sportistu, kuram kopējais treniņu laiks ir nulle.**
	
	Pareizās atbildes piemērs:
	`SELECT SPORTISTI.vards, SPORTISTI.uzvards, SUM(ilgums)`
	`FROM SPORTISTI`
	`LEFT JOIN TRENINI ON TRENINI.sportista_ID = SPORTISTI.sportista_ID`
	`GROUP BY SPORTISTI.sportista_ID;`
		1

3. daļa. Objektorientētā programmēšana un ārējās bibliotēkas (kopā 38 punkti)

<table>
<tr><th>Uzd. nr.</th><th>Vērtēšanas kritēriji</th><th>Punkti</th><th>Sagaidāmā atbilde</th><th>Komentāri</th></tr>
<tr><td colspan="5">1. uzdevums</td></tr>
<tr><td rowspan="2">1.1.</td><td>1.1.1. Izveido klasi “Speletajs”.</td><td>1</td><td rowspan="2"><pre>class Speletajs:
    def __init__(self, vards="Anonīms",
uzvards="Anonīms", numurs=0, speles=0, punkti=0,
augums=None, svars=None, nujas_tveriens=None):
        self.vards = vards
        self.uzvards = uzvards
        self.numurs = numurs
        self.speles = speles
        self.punkti = punkti
        self.augums = augums
        self.svars = svars
        self.nujas_tveriens = nujas_tveriens</pre></td><td rowspan="2"></td></tr>
<tr><td>1.1.2. Izveido klasei “Speletajs” norādītos atribūtus, ievērojot dotos datu piemērus objektam, kas turpmāk būs jāveido.</td><td>1</td></tr>
<tr><td rowspan="3">1.2.</td><td>1.2.1. Izveido klasei “Speletajs” konstruktoru, kas, ja visi klases atribūti ir norādīti, izveido konkrētā spēlētāja objektu.</td><td>1</td><td rowspan="3"><pre>   def __init__(self, vards="Anonīms",
uzvards="Anonīms", numurs=0, speles=0, punkti=0,
augums=None, svars=None, nujas_tveriens=None):
        print(f"Spēlētājs {self.vards} {self.uzvards}
ar numuru {self.numurs} ir veiksmīgi izveidots.")</pre></td><td rowspan="3">Nav pareizi, ka tiek veidota atsevišķa metode, nevis izvade notiek konstruktorā.</td></tr>
<tr><td>1.2.2. Nodrošina, ka klases “Speletajs” konstruktors izvada atbilstošu ziņojumu, ja visi klases atribūti ir norādīti.</td><td>1</td></tr>
<tr><td>1.2.3. Nodrošina, ka, ja klases “Speletajs” objektam netiek norādīts kāds no obligātajiem atribūtiem, tam(tiem) tiek piešķirtas noklusētās vērtības, ievērojot norādītos datu piemērus.</td><td>1</td></tr>
<tr><td>1.3.</td><td>Izveido klases “Speletajs” objektu “uzbrucejs”, kas satur norādīto informāciju.</td><td>1</td><td><pre>uzbrucejs = Speletajs("Rodrigo", "Ābols", 18, 3, 4,
191, 81, "kreisais")</pre></td><td></td></tr>
<tr><td>1.4.</td><td>1.4.1. Izveido klasei “Speletajs” metodi “pievienot_punktus(punkti)”, kas pievieno konkrētajam spēlētājam norādīto punktu skaitu.</td><td>1</td><td><pre>   def pievienot_punktus(self, punkti):
        self.punkti += punkti</pre></td><td></td></tr>
</table>

Uzd. nr.	Vērtēšanas kritēriji	Punkti	Sagaidāmā atbilde	Komentāri
	1.4.2. **Izmantojot metodi "pievienot_punktus(punkti)", pievieno spēlētājam "uzbrucejs" trīs punktus.**	1	`uzbrucejs.pievienot_punktus(3)`	

1.5.	1.5.1. **Izveido klasei "Speletajs" metodi "aprekinat()", kas pēc norādītās formulas aprēķina un atgriež spēlētāja statistiku (vidējo punktu skaitu spēlē).**	1	`def aprekinat(self):`
			`  if self.speles == 0:`
			`  return 0`
			`  else:`
			`  return round(self.punkti / self.speles, 2)`
				Pārbaude uz dalītāju-nulli (spēļu skaitu) nav obligāta.
				
				Decimāldaļas atdalītājs varētu būt gan komats, gan punkts.

	1.5.2. **Nodrošina, ka metodes "aprekinat()" rezultāts tiek noapaļots līdz diviem cipariem aiz komata.**	1		
1.6.	1.6.1. **Izveido klasei "Speletajs" metodi "izvadit()", kas izvada konkrētā spēlētāja vārdu un uzvārdu, kā arī numuru un statistiku atbilstoši dotajam izvades piemēram.**	1	`def izvadit(self):` `  print(f'Spēlētāja "{self.vards} {self.uzvards} ({self.numurs})" statistika ir vidēji {self.aprekinat()} punkti spēlē.')`	
	1.6.2. **Izmantojot metodi "izvadit()", izvada informāciju par spēlētāju "uzbrucejs".**	1	`uzbrucejs.izvadit()`	

1.7.	1.7.1. **Izveido klases "Speletajs" apakšklasi "Vartsargs" vārtsarga pozīcijas spēlētājiem.**	1	`class Vartsargs(Speletajs):`
			`  def __init__(self, vards="Anonīms", uzvards="Anonīms", numurs=0, speles=0, punkti=0, augums=None, svars=None, nujas_tveriens=None, metieni_kopa=7, atvairitie_metieni=5):`
			`  super().__init__(vards, uzvards, numurs, speles, punkti, augums, svars, nujas_tveriens)`
			`  self.metieni_kopa = metieni_kopa`
			`  self.atvairitie_metieni = atvairitie_metieni`	

	1.7.2. **Izveido apakšklasei "Vartsargs" norādītos papildu atribūtus.**	1		
	1.7.3. **Nodrošina, ka, ja apakšklases "Vartsargs" objektam netiek norādīts kāds no papildu atribūtiem, tam(tiem) tiek piešķirtas noklusētās vērtības, ievērojot norādītos datu piemērus.**	1		
1.8.	**Izveido apakšklases "Vartsargs" objektu "vartsargs", kas satur norādīto informāciju.**	1	`vartsargs = Vartsargs("Artūrs", "Šilovs", 31, punkti=2, augums=193, svars=92, nujas_tveriens="kreisais")`	

Uzd. nr.	Vērtēšanas kritēriji	Punkti	Sagaidāmā atbilde	Komentāri

1.9.	1.9.1. **Apakšklasē "Vartsargs" pārraksta virsklases metodi "aprekinat()", kas pēc norādītās formulas aprēķina un atgriež vārtsarga pozīcijas spēlētāja statistiku (atvairīto metienu procentuālo attiecību), un nodrošina, ka šīs metodes rezultāts tiek noapaļots līdz diviem ciparīem aiz komata.**	1	`def aprekinat(self):`
			`  return round(self.atvairitie_metieni / self.metieni_kopa * 100, 2)`
			
			`def izvadit(self):`
			`  print(f'Vārtsarga "{self.vards} {self.uzvards} ({self.numurs})" statistika ir {self.aprekinat()} %.')`
				Var izpildīt, gan pārraksot virsklases metodi, gan veidojot atsevišķu metodi, gan izmantojot standarta izvades metodi.
				
				Decimāldaļas atdalītājs varētu būt gan komats, gan punkts.

	1.9.2. **Izmantojot virsklases metodi "izvadit()", izvada informāciju par spēlētāju "vartsargs".**	1	`vartsargs.izvadit()`	

1.10.	1.10.1. **Izveido klasi "Komanda".**	1	`class Komanda:`
			`def __init__(self, nosaukums="Valstsvienība", speletaji=None):`
			`  self.nosaukums = nosaukums`
			`  self.speletaji = speletaji if speletaji is not None else []`
			`  self.turnira_punkti = 0 # šis būs čempionāta realizācijai`
			`  print(f'Komanda "{self.nosaukums}" ir veiksmīgi izveidota.')`	

	1.10.2. **Izveido klasei "Komanda" norādītos atribūtus, ievērojot dotos datu piemērus objektam, kas turpmāk būs jāveido.**	1		

1.11.	1.11.1. **Izveido klasei "Komanda" konstruktoru, kas, ja visi klases atribūti ir norādīti, izveido konkrētās komandas objektu.**	1	`def __init__(self, nosaukums="Valstsvienība", speletaji=None):`
			`  self.speletaji = speletaji if speletaji is not None else []`
			`  print(f'Komanda "{self.nosaukums}" ir veiksmīgi izveidota.')`	

	1.11.2. **Nodrošina, ka klases "Komanda" konstruktors izvada atbilstošu ziņojumu, ja visi klases atribūti ir norādīti.**	1		
	1.11.3. **Nodrošina, ka, ja klases "Komanda" objektam netiek norādīts nosaukuma atribūts, tam tiek piešķirta norādītā noklusētā vērtība.**	1		

Uzd. nr.	Vērtēšanas kritēriji	Punkti	Sagaidāmā atbilde	Komentāri
	1.11.4. **Nodrošina, ka, ja klases "Komanda" objektam netiek norādīts spēlētāju atribūts, tam tiek piešķirta tukša datu struktūra.**	1		
1.12.	**Izveido klases "Komanda" objektu "valstsvieniba".**	1	`valstsvieniba = Komanda("Valstsvienība")`	

1.13.	1.13.1. **Izveido klasei "Komanda" metodi "pievienot(speletajs)".**	1	`def pievienot(self, speletajs):`
			`  self.speletaji.append(speletajs)`
			`  print(f'Spēlētājs {speletajs.vards} {speletajs.uzvards} ir veiksmīgi pievienots komandai "{self.nosaukums}".')`	

	1.13.2. **Nodrošina, ka metode "pievienot(speletajs)" pievieno jaunu spēlētāju komandā.**	1		
	1.13.3. **Nodrošina, ka metode "pievienot(speletajs)" izvada atbilstošu ziņojumu, ja spēlētājs ir veiksmīgi pievienots.**	1		
	1.13.4. **Izmantojot metodi "pievienot(speletajs)", pievieno valstsvienības komandai spēlētāju objektus "uzbrucejs" un "vartsargs".**	1	`valstsvieniba.pievienot(uzbrucejs)` `valstsvieniba.pievienot(vartsargs)`	

1.14.	1.14.1. **Izveido klasei "Komanda" metodi "atrast_vertigako()".**	1	`def atrast_vertigako(self):`
			`  vertigakais = max(self.speletaji, key=lambda s: s.punkti)`
			`  print(f'Komandas "{self.nosaukums}" vērtīgākais spēlētājs ir {vertigakais.vards} {vertigakais.uzvards} ({vertigakais.punkti} punkti).')`
			`  return vertigakais`	

	1.14.2. **Nodrošina, ka metode "atrast_vertigako()" atrod vērtīgāko spēlētāju komandā (ar lielāko punktu skaitu).**	1		
	1.14.3. **Nodrošina, ka metode "atrast_vertigako()" izvada atbilstošu ziņojumu ar informāciju par atrasto vērtīgāko spēlētāju komandā.**	1		
	1.14.4. **Izmantojot metodi "atrast_vertigako()", atrod vērtīgāko spēlētāju valstsvienības komandā.**	1	`valstsvieniba.atrast_vertigako()`	

Uzd. nr.	Vērtēšanas kritēriji	Punkti	Sagaidāmā atbilde	Komentāri

1.15.1.	**Nodrošina, ka čempionātam pievienojas četras dažādas komandas.**	1	`def pievienot_komandu(self, komanda):`
			`  self.komandas.append(komanda)`
			
			`cempionats.pievienot_komandu(Komanda("Dinamo Rīga"))`
			`cempionats.pievienot_komandu(Komanda("HK Liepāja"))`
			`cempionats.pievienot_komandu(Komanda("HK Zemgale"))`
			`cempionats.pievienot_komandu(Komanda("HK Mogo"))`	

1.15.2.	**Nodrošina, ka katra komanda izspēlē vienu spēli ar pārējām komandām.**	1	`def izspelet(self):`
			`  speles_nr = 1`
			`  for i in range(len(self.komandas)):`
			`  for j in range(i + 1, len(self.komandas)):`
			`  k1 = self.komandas[i]`
			`  k2 = self.komandas[j]`	

1.15.3.	**Nodrošina, ka spēles rezultāts tiek piešķirts pēc nejaušības principa, ņemot vērā, ka maksimālais iegūstamo vārtu skaits vienā spēlē katrai komandai ir seši.**	1	`import random`
			
			`varti1 = random.randint(0, 6)`
			`varti2 = random.randint(0, 6)`	

1.15.4.	**Nodrošina, ka uzvaras gadījumā komanda iegūst divus punktus, ja rezultāts ir neizšķirts, katrai komandai – viens punkts, ja komandai ir zaudējums – nulle punktu.**	1	`if varti1 > varti2:`
			`  k1.turnira_punkti += 2`
			`  elif varti2 > varti1:`
			`  k2.turnira_punkti += 2`
			`  else:`
			`  k1.turnira_punkti += 1`
			`  k2.turnira_punkti += 1`	

1.15.5.	**Izvada visu spēļu rezultātus atbilstoši dotajam izvades piemēram.**	1	`print(f'{speles_nr}. spēle -`
			`"{k1.nosaukums}":"{k2.nosaukums}"`
			`({varti1}:{varti2})')`	

4. daļa. Datu struktūras un programmsaskarnes (kopā 28 * 1,5 = 42 punkti)

Uzd. nr.	Vērtēšanas kritēriji	Punkti	Sagaidāmā atbilde	Atbilžu piemēri (no kārtotāju darbiem)
1. uzdevums				
1.	**Uzraksta parametru, kas ir API pieprasījuma URL saitē.**	1	Pareizā atbilde: `name`	Piemērs (1 punkts): API pieprasījuma URL saite parametrs ir lietotāja vārds (name). Piemērs (sniegti vairāki varianti) (0 punktu): Iesniegtā atbilde: 1: count 2: name 3: age
2. uzdevums				
2.	**Uzraksta formātu, kādā no API tiek saņemti dati.**	1	Pareizā atbilde: `JSON`	Piemērs (sniegti vairāki varianti) (0 punktu): Iesniegtā atbilde: Json valoda formāts \"Pretty-print\" sauc arī par \"struktūrdruku\"
3. uzdevums				

3.1.	**Izveido pieprasījumu uz doto API.**	1	`response = requests.get(url)`	Piemērs (1 punkts):
				`import requests`
				`atbilde = requests.get("https://prgal26.azurewebsites.net/medals")`
				
				Piemērs (0 punktu):
				\"name\": \"Li Bingjie\",\r\n \"country\": \"China\",
				\"name\": \"Zhang Qiongyue\",\r\n \"country\": \"China\",
				
				\"name\": \"Duarte Alecrim Diana\",\r\n \"country\": \"Brazil\",
				\"name\": \"Dunn Crystal\",\r\n \"country\": \"United States\",

Uzd. nr.	Vērtēšanas kritēriji	Punkti	Sagaidāmā atbilde	Atbilžu piemēri (no kārtotāju darbiem)

3.2.	**Pārbauda servera atbildi un pārliecinās, ka atbildes statuss ir "200 OK".**	1	`if response.status_code == 200:`
			`print("Savienojums izdevās! (Statuss 200 OK)\n")`
				Piemērs (1 punkts):
				`if atbilde.status_code == 200:`
				`  print ("Viss ok")`
				
				Piemērs (1 punkts):
				`if response.status_code == 200:`
				`  print("Veiksmigi pieslegts, status_code 200")`
				`else:`
				`  print("Nesanaca pieslegties")`

3.3.	**Saglabā atbildi izvēlētajā datu struktūrā.**	1	`data = response.json()`	Piemērs (1 punkts):
				`data = response.json()`
				
				Piemērs (1 punkts):
				`if len(atbilde.text)>0:`
				`  print("dati ir saņemti,var stadāt!")`
				`  vardnica=json.loads(atbilde.text)`

3.4.	**Izvada visu sportistu vārdus un valstis.**	1	`for sportists in data:`
			`  print(f"{ sportists ['name']}, ({sportists['country']})")`
				Piemērs (1 punkts):
				`for i in res:`
				`  print(f"vārds: {i['name']}, valsts: {i['country']}")`
				
				Piemērs (izvada visus datus) (0 punktu):
				`try`
				`{`
				`  HttpResponseMessage response = await client.GetAsync(apiUrl);`
				`  if (response.IsSuccessStatusCode)`
				`  {`
				`  Console.WriteLine("Servera atbilde: 200 OK");`
				`  string responseData = await response.Content.ReadAsStringAsync();`
				`  Console.WriteLine("Sanemti dati:");`
				`  Console.WriteLine(responseData);`

<table>
<tr><th>Uzd. nr.</th><th>Vērtēšanas kritēriji</th><th>Punkti</th><th>Sagaidāmā atbilde</th><th>Atbilžu piemēri (no kārtotāju darbiem)</th></tr>
<tr><td rowspan="2">3.5.</td><td>3.5.1. Atlasa datus par sportistiem, kuri ieguva zelta medaļas.</td><td>1</td><td><pre>if sportists["medal_type"] == "gold":</pre></td><td rowspan="2">Piemērs (2 punkti):
<pre>for i in res:
    if i['medal_type'] == "gold":
        print(f"Vārds: {i['name']}, disciplīna: {i['discipline']}")</pre>Piemērs (1 punkts):
<pre>for i in dati:
    if i["medal_type"] == "golds":
        print(i["name"], i["discipline"])</pre>Piemērs (1 punkts):
<pre>for x in vardnica:
    if x["medal_type"]=="bronze":
        print("Vārds:",x['name'])
        print("Uzvārds:",x['discipline'])</pre>Piemērs (tikai atlasa datus) (1 punkts):
<pre>string apiUrlZelta = "https://prgal26.azurewebsites.net/medals?medal_type=gold";
using (HttpClient client = new HttpClient())</pre></td></tr>
<tr><td>3.5.2. Izvada atlasīto sportistu vārdus un disciplīnas.</td><td>1</td><td><pre>for sportists in data:
    if sportists["medal_type"] == "goldl":
        print(f"{ sportists['name']} - { sportists[' discipline]}")</pre></td></tr>
<tr><td rowspan="3">3.6.</td><td>3.6.1. Atlasa visu sportistu unikālās valstis (bez dublikātiem).</td><td>1</td><td><pre>countries = []
for i in data:
    if i[“country”] not in countries:
        countries.append(i[“country”])</pre></td><td rowspan="3">Piemērs (3 punkti):
<pre>countries=[]
count=0
for x in text:
    if x["country"] not in countries:
        countries.append(x["country"])
        count+=1
print(count, countries)</pre></td></tr>
<tr><td>3.6.2. Aprēķina unikālo valstu kopējo skaitu.</td><td>1</td><td><pre>skaits = len(countries)</pre></td></tr>
<tr><td>3.6.3. Izvada unikālās valstis un to kopējo skaitu.</td><td>1</td><td><pre>countries = []
for sportists in data:
    country = sportists["country"]
    if country not in countries:
        countries.append(country)</pre></td></tr>
</table>

<table>
<tr><th>Uzd. nr.</th><th>Vērtēšanas kritēriji</th><th>Punkti</th><th>Sagaidāmā atbilde</th><th>Atbilžu piemēri (no kārtotāju darbiem)</th></tr>
<tr><td></td><td></td><td></td><td>

```
for c in sorted(countries):
    print(c)
print(f"Kopā valstu:
{len(countries)}\n")
```

</td><td>Piemērs (3 punkti):

```
countries = []
for v in data:
    countries.append(v["country"])

countries = set(countries)
total = len(countries)
print(f"Kopa ir {total} valstis")
for country in countries:
    print(country)
```


Piemērs (neatlasa unikālās valstis) (1 punkts):

```
for i in list(tuple(unikalas_valstis)):
    print(i)
print(f"Unikālo valsts skaits ir {len(unikalas_valstis)}")
```

</td></tr>
<tr><td rowspan="3">3.7.</td><td>3.7.1. Atlasa datus par sportistiem no Beļģijas.</td><td>1</td><td>

```
if i[“country”] == “Belgium”:
```

</td><td rowspan="3">Piemērs (3 punkti):

```
for x in vardnica:
    if x["country"]=="Belgium":
        print("Sportista vārds no Beļģijas:",x['name'],"disciplīna:",x['discipline'],"medaļu tips:",x['medal_type'])
```


Piemērs (ir filtrēšana pēc valsts) (1 punkts):

```
string apiUrlBelgium = "https://prga126.azurewebsites.net/medals?country=Belgium";
using (HttpClient client = new HttpClient())
{
    try
    {
        HttpResponseMessage response = await client.GetAsync(apiUrlBelgium);
        if (response.IsSuccessStatusCode)
        {
            Console.WriteLine("Servera atbilde: 200 OK");
            string responseData = await response.Content.ReadAsStringAsync();
            Console.WriteLine("Sportistu vārdi, disciplīnas un iegūtā medaļu tipa informācija par
            Console.WriteLine(responseData);

        }
        else
        {
            Console.WriteLine("Servera atbilde: " + response.StatusCode);
```

</td></tr>
<tr><td>3.7.2. Iegūst atlasīto sportistu vārdus, disciplīnas un iegūto medaļu tipus.</td><td>1</td><td>

```
vards = sportists['name']
disciplina = sportists['discipline']
medalas_tips = sportists['medal_type']
```

</td></tr>
<tr><td>3.7.3. Izvada iegūto informāciju.</td><td>1</td><td>

```
for i in data:
    if i["country"].lower() ==
"belgium":
        print(f"{ sportists['name']} -
{ sportists['discipline']} - {
sportists['medal_type']}.vieta")
```

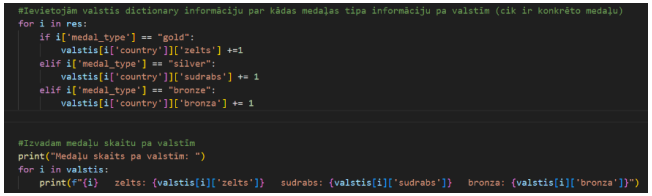
</td></tr>
</table>

<table>
<tr><th>Uzd. nr.</th><th>Vērtēšanas kritēriji</th><th>Punkti</th><th>Sagaidāmā atbilde</th><th>Atbilžu piemēri (no kārtotāju darbiem)</th></tr>
<tr><td rowspan="2">3.8.</td><td>3.8.1. Aprēķina katras valsts iegūto medaļu skaitu pēc kategorijas.</td><td>1</td><td><pre>
medals = {}

for sportists in data:
    valsts = sportists["country"]
    medala = sportists["medal_type"]

    if valsts not in medals:
        medals[valsts] = {"Gold Medal": 0, "Silver Medal": 0, "Bronze Medal": 0}

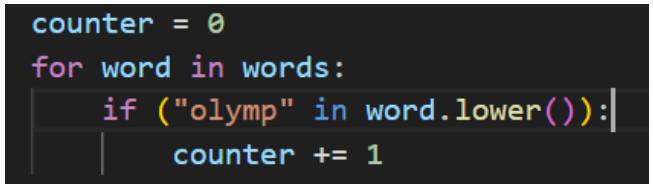
    medals[valsts][medala] += 1
</pre></td><td rowspan="2">Piemērs (3 punkti):

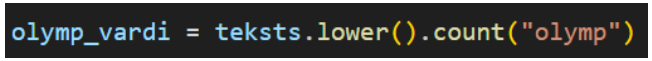
Piemērs (3 punkti):
</td></tr>
<tr><td>3.8.2. Izvada katras valsts iegūto medaļu skaitu pēc kategorijas.</td><td>1</td><td><pre>
for valsts in medalas:
    print(valsts)
    print(" Zelta:", medalas[valsts]["Gold Medal"])
    print(" Sudraba:", medalas[valsts]["Silver Medal"])
    print(" Bronzas:", medalas[valsts]["Bronze Medal"])
</pre></td></tr>
</table>

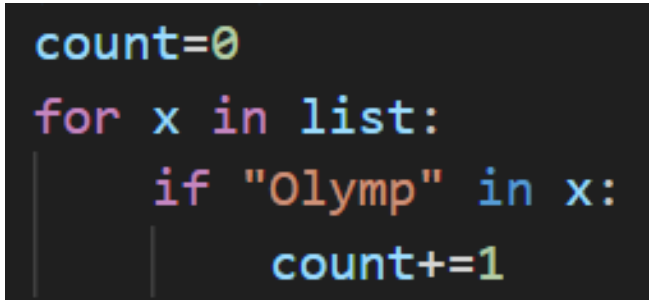
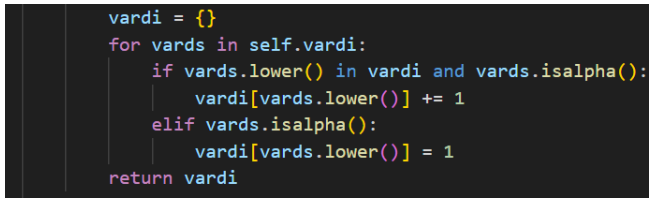
Uzd. nr.	Vērtēšanas kritēriji	Punkti	Sagaidāmā atbilde	Atbilžu piemēri (no kārtotāju darbiem)
4. uzdevums				
4.1.	**Izvada katra nākamā apakšuzdevuma rezultātu atbilstoši dotajam izvades piemēram.**	1		

4.2.	**Aprēķina kopējo vārdu skaitu tekstā, uzskatot katru rakstzīmju virkni, kas atdalīta ar atstarpi, par atsevišķu vārdu.**	1	Pareizā atbilde:
			136
			
			`words = text.split()`
				Piemērs (1 punkts):
				`vardi = teksts.split()`
				`print("vardu skaits teksta", len(vardi))`
				
				Piemērs (1 punkts):
				`#Uzdevums 4.2`
				`print("4.2. uzdevums")`
				`vardi = teksts.count(" ")+1`
				`print(f"Vārdu skaits tekstā: {vardi}")`
				
				Piemērs (0 punktu):
				`a = "The Olympic Games were first held`
				`#4.2. Aprēķini kopējo vārdu skaitu teks`
				`print(f"Vārdu skaits tekstā:{len(a)}")`
				
				Piemērs (0 punktu):
				`print("4.2.uzdevums")`
				`skaits=0`
				`for rinda in rindas.split(","):`
				`  print(len(rindas))`

<table>
<tr><th>Uzd. nr.</th><th>Vērtēšanas kritēriji</th><th>Punkti</th><th>Sagaidāmā atbilde</th><th>Atbilžu piemēri (no kārtotāju darbiem)</th></tr>
<tr><td>4.3.</td><td>Aprēķina, cik reizes tekstā sastopams vārds, kurš satur apakšvirkni “olymp”, neņemot vērā burtu reģistru.</td><td>1</td><td>Pareizā atbilde:
6

count_olymp =
words.lower().count(“olymp”)</td><td>Piemērs (1 punkts):


Piemērs (1 punkts):


Piemērs (meklē tikai “Olymp”) (0 punktu):

</td></tr>
<tr><td rowspan="2">4.4.</td><td>4.4.1. Iegūst visus vārdus, izlaižot jebkāda veida skaitļus, t. sk. gadus un gadsimtus.</td><td>1</td><td>if words.isalpha():</td><td rowspan="2">Piemērs (3 punkti):

</td></tr>
<tr><td>4.4.2. Atlasa no iegūtajiem vārdiem atšķirīgos (unikālos), neņemot vērā burtu reģistru.</td><td>1</td><td>uni_words = {}

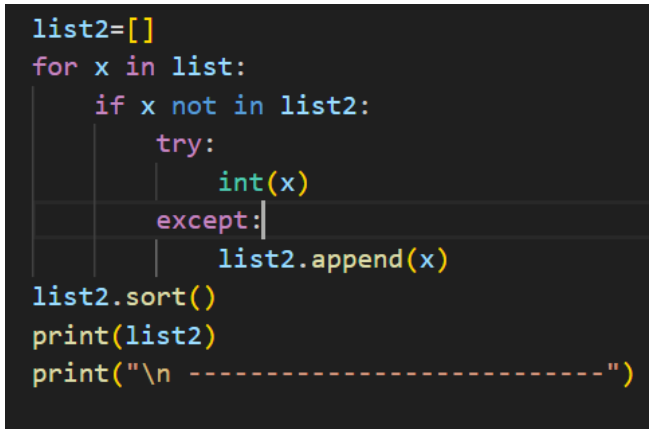
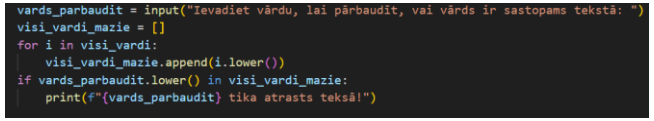
for word in words:
word = word.lower()
if word not in uni_words:
uni_words[word] = 1</td></tr>
</table>

<table>
<tr><th>Uzd. nr.</th><th>Vērtēšanas kritēriji</th><th>Punkti</th><th>Sagaidāmā atbilde</th><th>Atbilžu piemēri (no kārtotāju darbiem)</th></tr>
<tr><td></td><td>4.4.3. Sakārto alfabētiskā secībā atlasītos unikālos vārdus.</td><td>1</td><td><pre>uni_words = {}

for word in words:
    word = word.lower()

    if word in uni_words:
        uni_words[word] += 1
    else:
        uni_words[word] = 1

unique_sorted =
sorted(uni_words.keys())

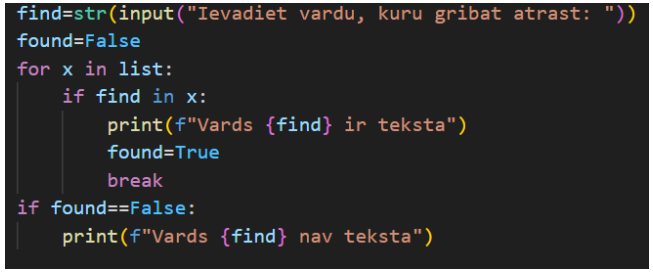
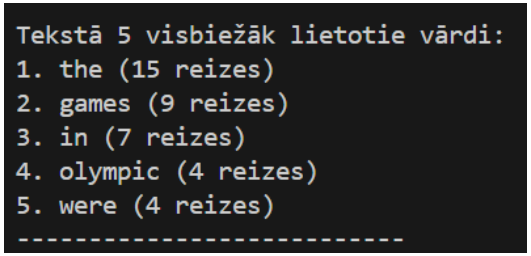
for word in unique_sorted:
    print(word)</pre></td><td>Piemērs (3 punkti):

</td></tr>
<tr><td rowspan="3">4.5.</td><td>4.5.1. Nodrošina lietotāja ievadi.</td><td>1</td><td><pre>user_word = input("Ievadi vārdu, ko
meklēt tekstā: ").lower()</pre></td><td rowspan="3">Piemērs (3 punkti):


Piemērs (3 punkti):

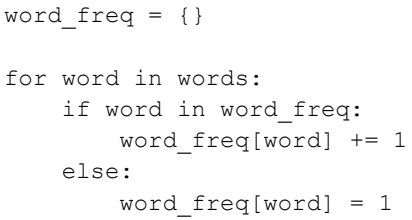
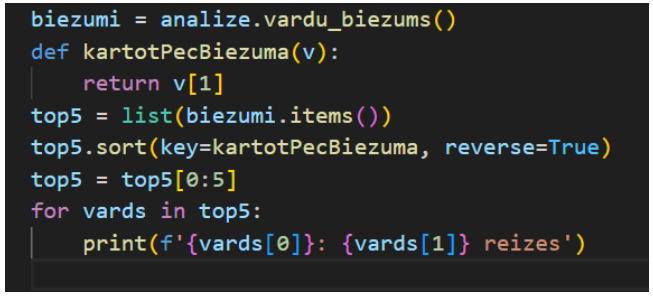
</td></tr>
<tr><td>4.5.2. Pārbauda, vai lietotāja ievadītais vārds ir tekstā, neņemot vērā burtu reģistru.</td><td>1</td><td><pre>found = False
if (user_word in words.lower():
     found = True</pre></td></tr>
<tr><td>4.5.3. Izvada ziņojumu par to, vai lietotāja ievadītais vārds ir vai nav atrasts.</td><td>1</td><td><pre>user_word = input("Ievadi vārdu, ko
meklēt tekstā: ").lower()

found = False
if (user_word in words.lower():
     found = True

if (found == True):
    print(f”Vārds {user_word} ir
atrodams tekstā”)
else:
    print(f”Vārds {user_word} nav
atrodams tekstā”)</pre></td></tr>
</table>

<table>
<tr><th>Uzd. nr.</th><th>Vērtēšanas kritēriji</th><th>Punkti</th><th>Sagaidāmā atbilde</th><th>Atbilžu piemēri (no kārtotāju darbiem)</th></tr>
<tr><td></td><td></td><td></td><td></td><td>Piemērs (ņem vērā burtu reģistru) (2 punkti):

</td></tr>
<tr><td>4.6.</td><td>4.6.1. Izveido datu struktūru, kurā tiek saglabāti vārdi un to lietošanas biežumi.</td><td>1</td><td>Pareizā atbilde:


Komentārs:
Kārtošana dilstošā secībā var būt realizēta vai nu kā atsevišķa darbība, vai nu pie TOP 5 izvades.


</td><td>Piemērs (3 punkti):

</td></tr>
</table>

<table>
<tr><th>Uzd. nr.</th><th>Vērtēšanas kritēriji</th><th>Punkti</th><th>Sagaidāmā atbilde</th><th>Atbilžu piemēri (no kārtotāju darbiem)</th></tr>
<tr><td rowspan="2"></td><td>4.6.2. Sakārto vārdus dilstošā secībā pēc to biežuma.</td><td>1</td><td><pre>sorted_items = sorted(word_freq.items(), key=lambda x: x[1], reverse=True)</pre></td><td rowspan="2">Piemērs (3 punkti):
<pre>rezultats = []
for vards in vardi:
    vards = vards.lower().strip(".,?;:!()")
    atrasts = False
    for ieraksts in rezultats:
        if ieraksts [0] == vards:
            ieraksts[1] += 1
            atrasts = True
    if atrasts == False:
        rezultats.append([vards, 1])

rezultats.sort(key = lambda x: x[1], reverse = True)
print("top 5")
for i in range(5):
    print([i][0], "-", rezultats[i][1])</pre>
Piemērs (TOP 5 izvade realizēta ar kļūdu) (2 punkti):
<pre>words = text.split(" ")
words_set = set(words)
lengths = []
for word in words_set:
    lengths.append(words.count(word))

lengths = sorted(lengths)
top = []
for i in range(len(lengths) - 4, len(lengths)):
    for word in words_set:
        if words.count(word) == lengths[i]:
            top.append(word)

print("4.6. uzdevums")
print("Vārdi, kas satopas visvairāk:")
for j in range(len(top) - 5, len(top)):
    print(top[j])
print("--")</pre></td></tr>
<tr><td>4.6.3. Izvada TOP 5 biežākos vārdus.</td><td>1</td><td><pre>print("TOP 5 vārdi (dilstošā secībā):")

for word, freq in sorted_items[:5]:
    print(f"{word}: {freq}")</pre></td></tr>
</table>