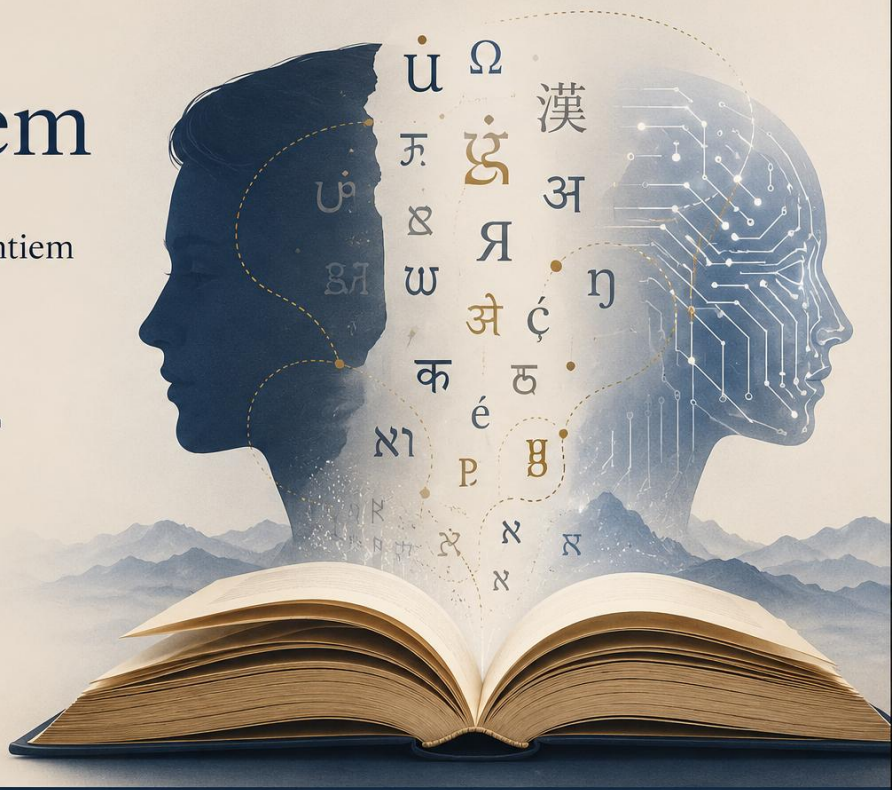
Dainis Zeps

Carpe Diem

spēle Syntra platformā.
Uzdevumu krājums studentiem
un AI rīkiem

- Uzdevumi valodām –
  no zināmām līdz pavisam svešām
- Attīsti lingvistisko intuīciju
  un domāšanu
- Cilvēks un AI –
  partneri valodas izpētē

*"Valoda nav tikai tas, ko mēs zinām,
bet arī tas, ko mēs spējam izprast."*



MĀCIES. UZMINI. PĀRBAUDI. SADARBĪJIES.

Satura rādītājs

Vingrinājumi 5
Vingrinājumi armēņu valodā 10
Uzdevumi 17
8. uzdevums* 23
Uzdevumi Phython programmēšanā 26
Uzdevumi Wolfram Mathematica programmēšanā 29
Uva kā programmēšanas rīks 31
Rekonstrukcijas 32
Literatūra un avoti 34

Carpe Diem spēle Syntra platformā. Uzdevumu krājums studentiem un AI rīkiem

Sastādījis Dainis Zeps, Latvijas Universitāte

Iesākts 2026. gada 11. augustā

Kopsavilkums

Šis darbs tiek veidots kā uzdevumu krājums, kas papildina mācīblīdzekli [1]. Uzdevumi paredzēti gan studentiem, gan AI rīkiem (LLM), turklāt uzdevuma teksts vienlaikus kalpo arī kā prompts LLM. Tādējādi LLM var palīdzēt studentam un piedāvāt iespējamos risinājumus, savukārt students var kritiski izvērtēt, cik sekmīgi LLM risina doto uzdevumu. Uzdevumi ļauj arī salīdzināt un izvērtēt gadījumus, kad LLM, balstoties uz savu uzdevuma interpretāciju, piedāvā būtiski atšķirīgus risinājumus.

Otrs un viens no galvenajiem mērķiem ir piedāvāt studentiem iespēju mācīties pašu izvēlētas valodas, vienlaikus aktīvi iesaistoties arī tādu uzdevumu risināšanā, kuros aplūkotās valodas viņiem ir maz pazīstamas vai pavisam svešas. Šādās situācijās studentam jāliek lietā savas vispārīgās lingvistiskās zināšanas un valodas intuīcija. Viņam jāmācās orientēties situācijā, kurā konkrēto valodu viņš nezina, tomēr, analizējot tās formas, struktūras, paralēles ar citām valodām un AI piedāvātos risinājumus, spēj izvirzīt pieņēmumus un tos pārbaudīt. Tādējādi uzdevumu mērķis nav tikai konkrētas valodas apguve, bet arī vispārīgas lingvistiskās domāšanas attīstīšana — spēja risināt lingvistiskus jautājumus un orientēties valodas situācijās arī tad, ja pati valoda vēl nav apgūta.

Trešais mērķis ir praktiski pētīt un veidot cilvēka un mašīnas sadarbību lingvistisku uzdevumu risināšanā. Students šeit nav tikai AI sniegtās atbildes saņēmējs: viņš formulē uzdevumu, vērtē AI piedāvāto risinājumu, atrod kļūdas un neskaidrības, labo tās un, ja nepieciešams, liek AI meklēt citu risinājumu. Savukārt AI darbojas ne tikai kā informācijas avots, bet kā partneris kopīgā valodas izpētes procesā. Tādējādi šie uzdevumi vienlaikus kļūst arī par eksperimentiem cilvēka un mašīnas sadarbības formu veidošanā.

Abstract

This work is being developed as a collection of exercises accompanying the learning material [1]. The exercises are intended for both students and AI tools (LLMs), with the text of each exercise also serving as a prompt for an LLM. In this way, an LLM can assist the student and suggest possible solutions, while the student can critically evaluate how successfully the LLM solves the given task. The exercises also make it possible to compare and assess cases in which LLMs, based on their own interpretation of a task, propose substantially different solutions.

A second and central objective is to offer students the opportunity to learn languages of their own choice while also actively engaging with tasks involving languages that are only slightly familiar or entirely unknown to them. In such situations, students are encouraged to draw on their general linguistic knowledge and linguistic intuition. They learn to orient themselves in a situation in which they do not know the particular language but can nevertheless formulate and test hypotheses by analysing its forms and structures, comparing them with other languages, and examining solutions proposed by AI. The aim of these exercises is therefore not only the acquisition of a particular language, but also the development of general linguistic reasoning: the ability to address linguistic questions and find one's way through unfamiliar linguistic situations even when the language itself has not yet been learned.

A third objective is to explore and develop, in practical terms, forms of human–machine collaboration in solving linguistic problems. The student is not merely a recipient of answers supplied by AI: the student formulates the task, evaluates the proposed solution, detects errors and ambiguities, corrects them, and, where necessary, prompts the AI to seek an alternative solution. AI, in turn, functions not only as a source of information but as a partner in a joint process of linguistic inquiry. In this sense, the exercises also become experiments in developing forms of collaboration between humans and machines.

Vingrinājumi

Vingrinājumos iekļausim salīdzinoši vienkāršākus uzdevumus, kas vairumā gadījumu neprasīs papildu programmēšanu, bet kas būs visdažādākas dabas un arī ar atšķirīgu grūtības pakāpi, kur šo pēdējo aspektu mēģināsim jau norādīt vingrinājuma tekstā, gan piešķirot (vai nē) vingrinājumiem kādas zvaigznītes (*).

1.vingrinājums

https://lingua.id.lv/armen/carpe-diem-armenian-pero.apburta.princese.html [VAA,]

Jūs kārtojat eksāmenu armēņu valodā, bet neesat bijis ne uz vienu nodarbību, nezināt neko, bet students māk nokārtot jebkuru eksāmenu jebkuros apstākļos. Jums jāizlasa no šīs pasakas fragments n.11. un jāsadala frāzēs, līdzīgi kā tas ir citās pasakās, kur sadalījumi jau ir veikti.

Veiciet sadalījumu!

2.vingrinājums

https://lingua.id.lv/armen/carpe-diem-armenian-pero.apburta.princese.html [VAA,]

Lai skatām fragmentu n.12 pasakā Apburtā princese. Veidojiet sadalījumu. Vai nepietiek armēņu valodas zināšanu vai tādu vispār nav? Jebkurā gadījumā palīgā varat ņemt Syntra ekrānu https://italozeps.github.io/ai-connectors-hub/TRM/charles-perrault/, kur atveriet n.2 šo pašu pasaku un fr. n.12. Tas ir tas pats teksts, bet te jūs katru formu varat atšifrēt. (Ja jums datoram divi ekrāni, uz katra lieciet pa šīs pasakas tekstam.) Jums nebūs pacietības visu tekstu tā pa formai vien izlasīt. Lai orientētos, kur var būt dalījuma vietas, vispirms izvēlieties pieturzīmēs. Teikuma beigas armēņu valodā apzīmē visbiežāk ar kolona zīmīti, ":", to tad meklējiet, vai komatu vai zīmīti "՝", apostrofu, kas arī armēņiem ir pieturzīme, tūlīt aiz pieturzīmes esošo formu tad apskatiet, un mēģiniet tai piemeklēt vārdu latviešu tekstā. Tālāk jau jūsu atjautība, vai uzminēsiet, kā veidot dalījumu.

Savu rezultātu izvadiet ar pogu Eksportēt fragmenta JSON. To jūs pārsūtīsiet pasniedzējam/eksaminatoram. Ko viņš ar to darīs? Apskatiet https://italozeps.github.io/ai-connectors-hub/LEX/cd.teacher/, kas ir skolotāja portāls, tur ar pogu Importēt skolēna JSON viņš saņems jūsu rezultātu. Apskatieties, izmēģiniet ar savu fragmentu. Ievadiet! Vai tur būs tas pats jūsu fragments, proti, portālā? pie kam ar jūsu dalījumu. Re, kā tas viss kopā strādā!

3. vingrinājums

https://lingua.id.lv/alex/syntra_syriac_greek.html

Draugs liek nokārtot eksāmenu viņa vietā. Eju eksaminēties, un te uzdevums – izpildīt uzdevumu trešajā fragmentā. Aizpildīt vismaz desmit formām gramatisko informāciju, izvadīt rezultāta JSON un nosūtīt skolotājam. Tad pamanu, ka draugam neesmu uzprasījis, kas par eksāmenu man jākārto. Nepazīstu ne valodu pa kreisi, ne pa labi. Ko darīt? Jautāšu ChatGPT? Kā lai jautāju? Ko lai prasu? Jā, vispirms, kas tās par valodām? Ko ieteiksiet?

4. vingrinājums

https://lingua.id.lv/alex/Syntra_student_reader_23.psalm.html

Par psalmu 23 jāpilda uzdevums – 4.fragmentā. Aizpildiet pēc iespējas vairāk formas, kur rakstiet ne obligāti piesaistot ebreju klaviatūru, bet brīvi. Ko nozīmē ‘kosī revajā’? Atrodiet to tekstā. Kā grieķu valodā tas atveidots?

5. vingrinājums

https://lingua.id.lv/alex/Syntra_student_reader.aram.targum.23.ps.html

Vai varam iejusties kā senajā Israēlī? Sinagogā ienes rulli – nolasa tekstu senebreju valodā. Vai klausītāji to saprata? Kā kurš un kā kurā reģionā. Daudziem ikdienas valoda jau bija aramiešu. Tāpēc pastāvēja tradīcija Rakstu tekstu arī tulkot un skaidrot aramiešu valodā – Targumā. Nu, lūk, mums vingrinājums, bet te otrādi – no senebreju, kas mums pazīstamāks, uz Targumu, kas mums, jā, ak vai, ne tik pazīstams. Te, lūk, lasīsim un mēģināsim saprast Targumu/tulkojumu, un, ko sapratām, atzīmēt kā saprastas formas Studenta info lodziņā. Mēģiniet izpildīt!

6. vingrinājums

https://italozeps.github.io/ai-connectors-hub/TRM/syntra/

Lai izmantojam Syntra morfoloģisko lasītāju Faidra fabulām, kas augstāk.

https://lingua.id.lv/alex/Syntra_Phaedrus_reader.html

Lai izmantojam Syntra sintaktisko lasītāju, kas ir "apbruņots" ar sintaktiskajām saitēm, ar vārdkopu izveidojumiem.

Lai apskatām piemēra pēc kādu vienu noteiktu fabulu, kaut vai to pašu De Vulpe et Uva. Apskatāmies, piemēram, formu 'elevant' un nospiežam ar peli uz tās – ieraudzīsim bez morfoloģiskās kādas info arī kāda sintatkiskās saites. Piem., qu elevant debebunt, qui verbis elevant, quae pussunt elevant, verbis elevant.

Un tagad apskatīsim to pašu vārdu/formu otrajā lasītājā, sintaktisko saišu/vārdkopu lasītājā. Ko ieraugām? Nospiedīsim ar peli uz formu 'elevant'. Ko ieraudzīsim.

Ak, vai! Izdalās tieši tās pašas vārdkopas kas bija morfoloģiskajā lasītājā. Ko tas nozīmē? Mistika? Nē, atcerēsimies, ka abi šie rīki rekonstruēti no viena avota, no programmatūras Syntra. Nu, te mēs redzam, no kurienes šīs vārdkopas. Tās bija veidotas kā papildinājums pie morfoloģiskās lasīšanas iespējas.

Bet aplūkosim tagad citu fabulu, piemēram, De Simonide. Un šai fabulai mums nav ne morfoloģiskās info, ne vārdkopu sadalījuma. Ko varam vieglāk izveidot? Ja vēlētos attīstīt šo neizpētīto stūrīti? Jā, varam jo viegli izveidot vārdkopu sadalījumu. Bet lai to darītu, mums labi jāpārvalda šīs fabulas teksts. Vai to varam viegli/vienkārši izdarīt? Varbūt tomēr mēģināt morfoloģisko analīzi pievienot fabulai? Jāsauc palīgā AI? Vai vēl kāds vingrinājums mums var palīdzēt? Lai ķeramies pie pētīšanas!

7. vingrinājums

https://lingua.id.lv/alex/Le_chat_botte.fr.html

Lasām Šarla Pero pasaku "Runcis zābakos" **franču** valodā.

Izmantojam pasakas tekstu no Iļjas Franka grāmatiņām, [VA, 1]

8. vingrinājums

https://lingua.id.lv/alex/el_principe_tomasito.html

Lasām pasaku **spāņu valodā** "El príncipe Tomasito y San José" .

Izmantojam pasakas tekstu no Iļjas Franka grāmatiņām, [VA, 1]

9. vingrinājums

https://lingua.id.lv/alex/carpe_diem_czech_tevocis_psihoda..html

Lasām **čehu valodā** Boženas Ņimcovas pasaku Strejček Příhoda.

Izmantojam pasakas tekstu no Iļjas Franka grāmatiņām, [VA, 1]

10. vingrinājums

https://lingua.id.lv/alex/carpe-diem-italian-nonso.html

Lasām itāļu pasaku "Nonsò – Nezinu"

Izmantojam pasakas tekstu no Iļjas Franka grāmatiņām, [VA, 1]

Literatūra un avoti [V]

[V, 1] Метод чтения Ильи Франка, http://www.franklang.ru/index.php/

[V,2] **Franču valoda**, Šarla Pero pasaka "Runcis zābakos", teksts no [VA,1], https://lingua.id.lv/alex/Le_chat_botte.fr.html

[V,3] **Spāņu valoda**, Pasaka "El príncipe Tomasito y San José", teksts no [VA, 1], https://lingua.id.lv/alex/el_principe_tomasito.html

[V,4] **Čehu** valoda, Boženas Ņemcovas pasaka "Strejček Příhoda", https://lingua.id.lv/alex/carpe_diem_czech_tevocis_psihoda..html

[V,5] **Itāļu** valoda, Pasaka itāļu valodā "Nonsò – Nezinu", teksts no [VA,1], https://lingua.id.lv/alex/carpe-diem-italian-nonso.html

[V,6] Programma Syntra, Faidra 44 fabulas ar Gr. Smeltera papildinājumiem – Intepretationes Latinae, formas ar morfoloģisku atšifrējumu + sintaktiskās ķēdes (vārdkopas ar pārklājumiem), https://italozeps.github.io/ai-connectors-hub/TRM/syntra/

[V,7] Programma Syntra (ar **ieskaņojumu**), Faidra 44 fabulas ar Gr. Smeltera papildinājumiem – Intepretationes Latinae, formas ar vārdkopām ar pārklājumiem, **ieskaņojums vārdkopām**, https://lingua.id.lv/alex/Syntra_Phaedrus_reader.html

[V,8] Bibliskā **senebreju** valoda ar tulkojumu uz sengrieķu (LXX), 23. psalms, studentu lasītājs (morfoloģisko info aizpilda students), https://lingua.id.lv/alex/Syntra_student_reader_23.psalm.html

[V,9] Klasiskā **sīriešu** valoda ar tulkojumu sengrieķu valodā (koinē), Jāņa evanģ. 10,1-18, studentu lasītājs (morfoloģisko info aizpilda students), https://lingua.id.lv/alex/syntra_syriac_greek.html

[V,10] Apvērstais tulkojums: **Targumi (aramiešu)**, uz senebreju valodu, studentu lasītājs (morfoloģisko info aizpilda students), https://lingua.id.lv/alex/Syntra_student_reader.aram.targum.23.ps.html

[V,11] **Latīņu** valoda, Studentu lasītājs, Demo versija ar vienu teikumu: Ut panem domum ferant canes, https://lingua.id.lv/alex/Syntra_student_reader.html

[V,12] Ukrainas atbalstam, https://lingua.id.lv/art/publ.war.html

Vingrinājumi armēņu valodā

Šajā vingrinājumu sadaļā iekļauti tie, kur viena no skatāmām valodām ir armēņu valoda. Šie vingrinājumi būtu orientējami diviem mērķiem, gan tiem, kas mācās armēņu valodu vai ir armēņu valodas pārvaldītāji (un tos interesētu pretējā valoda), vai tādiem kam armēņu valoda ir pilnīgi sveša valoda. Tiem šie vingrinājumi kalpos kā tie uzdevumi, kur studentam jādarbojas ar "nepazīstamo" valodu, kur no studenta tiek prasīta cita tipa atjautība – mēģināt orientēties svešas valodas situācijā, lai izpildītu kādu uzdotu vingrinājumu šādā pagriezienā.

1.vingrinājums

https://italozeps.github.io/ai-connectors-hub/TRM/charles-perrault/

Astoņas Šarla Pero pasakas tulkotas armēņu valodā no oriģinālajiem Š.P. franču tekstiem. Pasaku izvēlē labās puses panelī atrodiet vēlamo pasaku, sākot no Sarkangalvīte. Blakus pa labi ir fragmenta izvēle, piemēram, Sarkangalvītei to ir 14. Pa labi un kreisi attiecīgi ir Next un Prev pogas nākošā un iepriekšējā fragmenta izvēlei. Pavisam labajā pusē ir taustiņš – trīs punktiņi , … , to spiežot, atveras vēl rīkjosla, kur podziņa pa kreisi H (Hide) aizver latviešu tulkojumu, un Sh (Show) atkal atver to. Tālāk uz kreiso pusi, Paradigm Editor, Vocabulary Editor, kreisajā pusē, About lodziņš. Ja spiedīsiet … rīkjosla aizvērsies, bet saglabāsies stāvoklis H vai Sh. Latviešu tekstu, piemēra pēc, var aizvērt, kad jau pasakas tekstu daudzmaz esat apguvis armēņu valodā un tulkojums nav vairs vajadzīgs. Aizklātā izvēle vēl noderīga būs eksāmena gadījumā, kad jūs jau kāds eksaminēs, teiksim, online režīmā.

Par galveno paneli pa kreisi, kur pats pasakas teksts armēņu valodā. Uzbraucot ar peli uz kāda vārda/formas, ieraudzīsiet formas atšifrējumu peldošajā ekrāniņā, bet uz to uzklikšķinot ar peli, labajā panelī apakšā parādīsies nofiksējies formas gramatiskais atšifrējums.

2. vingrinājums

Pasaka par Fejām https://italozeps.github.io/ai-connectors-hub/LEX/cd.fejas/

Šarla Pero pasaka par Fejām Carpe Diem panelī. Šeit tas pats teksts, kas 1.vingrinājumā, pie izvēles 3. Bet šeit pasakai ir izdalīti apakšfragmenti/frāzes, kas uz peles kustību iekrāsojas un klikšķinot skandina attiecīgo izvēli armēņu valodā. Valodas skanējumu nodrošina ElevenLab rīks, tā armēņu izrunas kvalitāte ir atkarīga no šī rīka, kaut arī ar laipnu atbalstu no izstrādātāju puses ir klonēta mūsu puses armēņu valodā runātāja balss.

Ar Next pogu skatieties nākamo fragmentu to pašu dariet ar fragmentu izvēli. Tālāk ir Rediģēt/Sadalīt poga, kur atvērtajā dialoga logā varat paši izveidot savu fragmentu izvēli un tad to ierunāt (ar ElevenLab rīku). Par šī paneļa lietošanu/izmantošanu skatīsim citos vingrinājumos. Tam var arī paredzēt eksāmena uzdevumu, kad students veic to pašu online. Ar pogu Saglabāt (visu pasaku) var saglabāt pasaku ar jauno fragmentu/frāžu veidojumu (bet ne online).

https://lingua.id.lv/armen/carpe-diem-armenian-fejas.1.ai.html

3. vingrinājums

https://lingua.id.lv/alex/Syntra_student_reader.html [VA,11]

Iepazīstieties ar šo lasītāju, kas saucas **Syntra student reader**. Šeit šobrīd ievadīts ļoti vienkāršs teksts – Ut panem domum ferant canes ar paralēli latvisko tulkojumu – Lai maizi mājās nes suņi.

Ar peli uz latīņu teksta formu ieraudzīsiet peldošo ekrāniņu, ka forma nav aizpildīta, bet ja ieklikšķināsiet, labajā pusē panelī apakšā atvērsies – Gramatiskā informācija (aizpilda students). Šis ir arī šīs lapas mērķis, lai students aizpilda formas ar gramatisko informāciju. Kad students būs ierakstījis gramatisko info (pēc saviem ieskatiem/cik iemācījies – sic!), un nospiedis saglabāt, tad, nākamreiz uzbraucot ar peli uz šīs formas, parādīsies tas, ko ierakstījis students.

Izmēģiniet!

Piezīme: Šo formu/lapu turpmāk izmantosim daudz. Pamazām iepazīstieties ar to, lai adektvāti to lietotu.

4. Vingrinājums

Pasaka par Sarkangalvīti

https://lingua.id.lv/armen/carpe-diem-armenian-pero.sarkangalvite.html

Oriģinālā Šarla Pero pasaka, ar salinkotām un ieskaņotām frāzēm.

5. Vingrinājums

Pasaka par Apburto princesi

https://lingua.id.lv/armen/carpe-diem-armenian-pero.apburta.princese.html

Oriģinālā Šarla Pero pasaka ar vēl nesalinkotām frāzēm.

6. Vingrinājums

Pasaka par Fejām

https://lingua.id.lv/armen/carpe-diem-armenian-fejas.1.html

Šarla Pero pasakas saīsināts variants ar izveidotām/salinkotām frāzēm.

7. Vingrinājums

Pasaka par Zilbārdi

https://lingua.id.lv/armen/carpe-diem-armenian-zilbardis.html

Šarla Pero pasakas saīsināts variants ar izveidotām/salinkotām frāzēm.

8. Vingrinājums

Pasaka par Fejām

https://lingua.id.lv/armen/carpe-diem-armenian-fejas.html

Šarla Pero pasakas saīsināts variants ar izveidotām/salinkotām frāzēm.

Dodam šīs pasakas atšķirīgus variantus. Noderīgi būs tiem, kas grib to pašu pasaku skatīt ar atšķirīgiem tulkojumiem.

9. Vingrinājums

Runcis Zābakos

https://lingua.id.lv/armen/carpe-diem-armenian-katu-fr.html

Ne oriģinālais Šarla Pero pasakas variants, pie kam paralēlais teksts franču valodā.

10. Vingrinājums

Runcis Zābakos

https://lingua.id.lv/armen/carpe-diem-armenian-runcis.html

Oriģinālais Šarla Pero pasakas variants, ar linkotām frāzēm un ieskaņojumu

11. Vingrinājums

Pasaka par Sarkangalvīti

https://lingua.id.lv/armen/carpe-diem-armenian-sarkangalvite.html

Šarla Pero pasakas saīsināts variants ar izveidotām/salinkotām frāzēm.

12. Vingrinājums

Pasaka par Zilbārdi

https://lingua.id.lv/armen/carpe-diem-armenian-pero.zilbardis.html

Šarla Pero pasakas pilnais variants bez izveidotām/salinkotām frāzēm.

13. Vingrinājums

Pasaka par Īkstīti

https://lingua.id.lv/armen/carpe-diem-armenian-pero.ikstitis.html

Šarla Pero pasakas oriģinālais variants. Nav vēl salinkots.

14. Vingrinājums

Rikē ar matiņu

https://lingua.id.lv/armen/carpe-diem-armenian-pero.rike.ar.matinu.html

Šarla Pero oriģinālā pasaka. Nav vēl salinkota.

15. Vingrinājums

Pasaka par Pelnrušķīti

https://lingua.id.lv/armen/carpe-diem-armenian-pero.pelnruskite.html

Šarla Pero oriģinālā pasaka. Nav vēl salinkota.

Literatūra un avoti [VA]

[VA, 1] Метод чтения Ильи Франка, http://www.franklang.ru/index.php/

[VA,2] **Armēņu** valoda, Astoņu Šarla Pero pasaku (armēņu tulkojumos) lasītājs, morfoloģiskais formu atšifrējums, https://italozeps.github.io/ai-connectors-hub/TRM/charles-perrault/

[VA,] **Armēņu** valoda, Pasaka par Sarkangalvīti, Šarla Pero pasakas tulkojums armēņu valodā, oriģinālais Š.P. teksts, teksts salinkots un **ieskaņots**, https://lingua.id.lv/armen/carpe-diem-armenian-pero.sarkangalvite.html

[VA,] **Armēņu** valoda, Pasaka par Apburto Princesi, Šarla Pero pasakas tulkojums armēņu valodā, oriģinālais Š.P. teksts, https://lingua.id.lv/armen/carpe-diem-armenian-pero.apburta.princese.html

[VA,] **Armēņu** valoda, Pasaka par Fejām, Šarla Pero pasakas tulkojums armēņu valodā, oriģinālais Š.P. teksts, https://italozeps.github.io/ai-connectors-hub/LEX/cd.fejas/ , https://lingua.id.lv/armen/carpe-diem-armenian-fejas.1.ai.html

[VA,] **Armēņu** valoda, Runcis Zābakos, Šarla Pero pasakas tulkojums armēņu valodā, oriģinālais Š.P. teksts, salinkots un **ieskaņots**, https://lingua.id.lv/armen/carpe-diem-armenian-runcis.htm

[VA,] **Armēņu** valoda, Pasaka par Īkstīti, Pasaka par Īkstīti, Šarla Pero pasakas tulkojums armēņu valodā, oriģinālais Š.P. teksts, https://lingua.id.lv/armen/carpe-diem-armenian-pero.ikstitis.html

[VA,] **Armēņu** valoda, Pasaka par Rikē ar matiņu, Šarla Pero pasakas tulkojums armēņu valodā, oriģinālais Š.P. teksts, https://lingua.id.lv/armen/carpe-diem-armenian-pero.rike.ar.matinu.html

[VA,] **Armēņu** valoda, Pasaka par Pelnrušķīti, Šarla Pero pasakas tulkojums armēņu valodā, oriģinālais Š.P. teksts, https://lingua.id.lv/armen/carpe-diem-armenian-pero.pelnruskite.html

[VA,] **Armēņu** valoda, Pasaka par Zilbārdi, Šarla Pero pasakas tulkojums armēņu valodā, oriģinālais Š.P. teksts, https://lingua.id.lv/armen/carpe-diem-armenian-pero.zilbardis.html

[VA,] **Armēņu** valoda, Pasaka par Fejām, Šarla Pero pasakas tulkojums armēņu valodā, saīsināts Š.P. teksts, salinkots un **ieskaņots**, https://lingua.id.lv/armen/carpe-diem-armenian-fejas.1.html

[VA,] **Armēņu** valoda, Pasaka par Fejām, Šarla Pero pasakas tulkojums armēņu valodā, saīsināts Š.P. teksts, salinkots un **ieskaņots**, https://lingua.id.lv/armen/carpe-diem-armenian-fejas.html

[VA,], **Armēņu - franču** valoda, Runcis Zābakos, Šarla Pero pasakas tulkojums armēņu valodā, saīsināts Š.P. teksts, salinkots un **ieskaņots**, https://lingua.id.lv/armen/carpe-diem-armenian-katu-fr.html

[VA,] **Armēņu** valoda, Pasaka par Sarkangalvīti, Šarla Pero pasakas tulkojums armēņu valodā, saīsināts Š.P. teksts, salinkots un **ieskaņots**, https://lingua.id.lv/armen/carpe-diem-armenian-sarkangalvite.html

[VA,] **Armēņu** valoda, Pasaka par Zilbārdi, Šarla Pero pasakas tulkojums armēņu valodā, saīsināts Š.P. teksts, salinkots un **ieskaņots**, https://lingua.id.lv/armen/carpe-diem-armenian-zilbardis.html

[VA,] Ukrainas atbalstam, https://lingua.id.lv/art/publ.war.html

Uzdevumi

Platforma Syntra veidojusies no programmām Borland Delphi vidē, kas iesākta pirms 2000. gada, kur intensīvākais periods bija 1998. gada pavasarī. Bet sākums ir meklējams vēl agrāk, kad radās ideja veidot latīņu valodas lasītāju, kas ieguva reālākus apveidus, kad tika atklāta William Whitaker Word vārdnīca. Tajā laikā arī izveidojās divas programmas Syntra un Carpe Diem, kuras rekonstruējam jau šodienas apstākļos. Pirms tam vēl veidojās interneta lapa Reading Exercises in Latin, kas tika iekļauta latīņu valodas skolotāju apvienotajā lapā The Classic Technology Center CTCWeb, AbleMedia platformā. Balstoties uz šīs lapas materiāla ir arī izveidota prog. Syntra.

Tālāk jau par šo krājumu.

Pie uzdevumiem pieskaitīsim šos studentam adresētos darbus, kas prasa lielāku piepūli, attiecīgu, eventuāli, lielāku AI piesaisti, lai kādu uzdevuma prasību izpildītu. Te izpildes līmeņi var stipri atšķirties no izpildītāja kvalifikācijas gan lingvistiskā plāksnē, gan sagatavotībā programmēšanā, gan no spējas piesaistīt AI rīkus uzdevumu pildīt. Šeit izpildītājam gan jāmēģina izpildīt uzdevumā norādīto, gan censties pēc iespējas produktīvāk izmantot šīs platformas un spēles Carpe Diem iespējas. Lai students pats izdomā, kā attīstīt šīs iespējas, un arī tajos virzienos, kā to atrod sev par nepieciešamāko. Lai izaicina ar savu ieguldījumu AI iespējas attiecīgā jautājuma risināšanā!

1.uzdevums.

Lai veidojam Latīņu mājas lasīšanas paraugam [] interaktīvu lapu pēc parauga [Lx,1], armēņu pasakām. Veidosim Syntra Latin Latvian LATIN, kur ar lielajiem burtiem būs skaņas kolona.

Lai mums teksts jau izraudzīts – lai šajā pirmajā reizē teksts ir viena rindiņa “Ut panem domum ferant canes”, blakus būs latviešu tulkojums “Lai maizi mājās nes suņi”. Mums ir jau arī Whitaker vārdnīcai atbilstošās datu bāzes izveidota python bibliotēka, par to atsevišķi.

Vispirms izveidosim ar python prog. build_batch.py. Šo, kā arī citus python elementus lūgsim veidot mums piesaistītajam AI rīkam. Mēs taču darbojamies kopā. Šai procedūrai jāizveido json fails ar pieciem ierakstiem, piemēram, piemēram pirmajam vārdam atbildīs ieraksts,

"ut": {"lemma": "","morphology": "","latvian": "", "english": "","note": "" }

Kas atbildīs pirmajam latīņu vārdam 'ut', bet pārējie lauki tukši, jo tur ir "", kas nozīmē, šī json forma vēl nav aizpildīta ar gramatiskajām ziņām.

Nākamajā solī mums šī json forma jāizpilda, ko izdarīsim ar Whitaker vārdnīcas palīdzību, kur attiecīgais python modulis mums ir jau izveidots. AI lūgsim izveidot python procedūru fill_vocab.py, kuru izpildot mūsu json forma aizpildīsies ar gramatiskajām ziņām, piemēram, pirmajam ierakstam atbildīs jau šāds elements

"ut": {"ut": "","morphology": "**V - SG PRES IMP ACT 2P**","latvian": "", "english": "use, make use of, enjoy","note": "" } . Latviešu tulkojums paliek tukšs, jo Whitaker tulko tikai angļu valodā. Pēdējo lauku arī neizmantojam šeit. Kad mums ir gatava jau aizpildītā json forma, veicam trešo soli, lūdzam AI izveidot python procedūru generate_latin.py, kas izveido .html lapu pēc vēlamā parauga, kur vēl palūdzam pievienot Read pogu, lai mums pieejamais balss aģents pievieno izrunu. Iesākumā šo soli varam arī izlaist.

Izveidojiem json formu visai dotajai frāzei/teikumam no pieciem vārdiem.

Tagad, ja izveidotajam docx ar tabulu, kur kreisajā rindā latīņu un labajā latviešu tulkojumi, pielietojam visas trīs mūsu iegūtās python procedūras, tad iegūstam šo https://lingua.id.lv//alex/Ut_panem_reader.html

Lai šis kopskats, ko redzam, ir arī piederīgs pie kopējā prompta, ko mēs dosim mūsu palīgiem, AI rīkiem, lai no visa kopā ņemot, viņi izveido šo iznākumu, kā testu, šai pieejai, ko te klāstam. Tālāk jau teksta parauga vietā liekam citus tekstus un izmēģinām. Piemēram, https://lingua.id.lv/alex/Poggii_facetiae_reader.html.

Visas trīs procedūras, ko darbināsim, ir šīs zemāk. (Par Whitaker datu bāzes pievienojamo bibliotēku mums vēl jārunā atsevišķi, ko ņemsim no https://lingua.id.lv/alex/lingua-dicttools.html.)

Šīs procedūras darbinām, lai iegūtu rezultātā vēlamo .html lapu.

python.exe build_batch.py

python.exe fill_vocab.py

python.exe generate_latin.py

Jautājums: piedāvātajā risinājumā kļūda ieviesusies, jo ‘ut’ ir divas interpretācijas iespējas, kā verbu un kā saikli. Sistēma izvēlējusies verbu un to mēs varētu uzlūkot kā valodas kļūdu, bet sistēma nav risinājusi jautājumu, ko darīt šādu paralēl-situāciju gadījumā. Vai to vajag risināt? Un vai ir vienkāršs kāds risinājums, kas derētu šajā situācijā?

2.uzdevums

Lai risinām līdzīgu uzdevumu kā iepriekšējā uzdevumā, bet tagad veidosim

Syntra Greek Latin GREEK izkārtojumu

ar dotu tekstu **:** *Ὅπως οἱ κύνες τὸν ἄρτον οἴκαδε φέρωσιν,* t.i. iepriekšējais teikums tikai grieķu valodā.

Bet atšķirībā no iepriekšējā uzdevuma pieņemsim, ka mums nepieciešamās Greek WDB nav, bet mēs to veidosim paši. Kā mums būs tas jādara (šī uzdevuma rāmjos)?

Izveidojam neaizpildītu json tekstam, tā formām pēc parauga

"*Ὅπως* ": {"lemma": "","morphology": "","latin": "" } .

Tā kā mums WDB vēl nav, tad darbinām tikai pēdējo no trīs procedūrām generate.py, ko pasūtīsim izveidot LLM. Rezultātā mums jāiegūst līdzīga kā iepriekšējā uzdevumā html lapa, kur peldošie ekrāniņi uz teikuma formām rādīs neaizpildītu gramatisko informāciju, jo nelietojām WDB. Bet tagad darbosimies pretēji kā iepriekšējā uzdevumā un paši aizpildīsim WDB informāciju, t.i, uzbraucot ar peli uz formas un tur klikšķinot, gramatiskās informācijas laukā pa labi, kas mums veidojās statistiski, neesošās informācijas vietā ierakstīsim Whitaker gramatisko rindu, e.g. šajā piemērā priekš vārdiņa CONJ vai kaut kā tamlīdzīgi, kāda tā parādījās iepriekšējā uzdevumā vārdiņam ‘ut’ . Kad šo esam ierakstījuši gramatiskajos laukos, lai sistēma to atceras, un kad nākamreizi skatīsim ar peli šo formu, tur parādīsies mūsu ievietotā informācija. Šajā solī mēs vēl nepārbaudām, vai mūsu ierakstītā informācija ir pareiza. Jā, mēs it kā veidojam mūsu darinātās WDB no mūsu gramatisko zināšanu krātuves. Kad aizpildīta gramatiskā informācija visiem pieciem vārdiem, uzdevums ir izpildīts un nododams pārbaudei.

Jautājums: kā saglabāt studenta izveidoto WDB? (Meklējiet atbildi iepriekšējā uzdevumā.)

Jautājums: kuru valodu students zina un kuru mācās pēc šī uzdevuma nostādnes?

3. uzdevums

Lai mums ir divvalodu teksta fragments un vēlamies veidot sintaktisku savienojumu diviem vārdiem, pie kam abu valodu kontekstā, kur šo savienojumu nosauksim pēc to savienojamo elementu lomām teikumā/fragmentā. Piemēram grieķu teikumā

Ἕως μὲν δὴ Ἀλέξανδρος παρὰ θάλατταν ἐπορεύετο, ἡ χώρα· ἐρήμη μὲν ἦν, ἀλλ᾽ οὐκ ἄνυδρος.

Quamdiu quidem Alexander secundum mare iter faciebat, regio deserta quidem erat, sed non inaquosa.

Grieķu teikumā izdalīsim divu vārdus

παρὰ θάλατταν

un latīnu teikumā

secundum mare – gar jūru .

Šo fiksēsim kā sintaktisku saiti un nosauksim, liekot tai vārdu sastāvošu no diviem – PREP ACC, un latīņu teikumā – PREP DAT. Izmantojot šo kā paternu, izveidosim tam atbilstosu json ierakstu.

Lūdzu, uzrakstiet abilstošo json, kas sastāv no astoņiem elementiem. Palūidziet kāda LLM palīdzību, to izdarīt un salīdziniet. Ja izvēlētais pāris nesatur tiešu analoģiju otrā valodā, tad attiecīgāa vieta atstājiet tukšas. Tā arī būs informācija, ka otrajā valodā tīešas ekvivalences izvēlētajai sintatksiskajai saitei nav.

Uzrakstiet kādus piemērus no dotā teikuma, lai aizpildītu jūsu izveidoto json formu, vai lūdziet LLM tos atrast jūsu vietā.

Vēl kāds papildinājums. Šīs sintaktiskās saites būs saistītas ar attiecģo fragmenu, tā ka, ja kāds vārds fragmentā atkārtosies, pie tā json būs jāpieliekt numuriņš,

PIezīme: jautājot LLM, atklājas pirmā kļūda. Gramatiskajai specifikācijai jābūt pietiekamai, ka sintaktistā saite ir nofiksēta, tā nevar saturēt nenoteiktību, jo tad tā nedarbosies, pie kam vienmēr var specificēt saiti, lietojot virākus gramatiskos parametrus. Jāparedz arī tas, ka ieviesīsim podziņas ar attiecīgajiem gramatiskajiem apzīmējumiem, kuras lietosim, nevis vienkārši vārdus liksim studentam rakstīt laukos. Būs mums, piemēram, poga FINIT, kas nozīmēs finītu verba formu.

Te, lūk, ko piedāvā ChatGPT. Gan varat ieraudzīt, kādu json formu viņis piedāvā, gan konkrēti piemēri no mūsu ņemtā teikuma.

{{"w11":"παρά","w12":"θάλαττα ν","r11":"PREP","r12":"ACC","w21":"secundum","w22":"mare","r21":"PREP","r22":"ACC"},

{"w11":"Ἀλέξανδρος","w12":"ἐπορεύετο","r11":"SUBJ","r12":"V","w21":"Alexander","w22":"faciebat","r21":"SUBJ","r22":"V"},

{"w11":"χώρα","w12":"ἐρήμη","r11":"SUBJ","r12":"PRED","w21":"regio","w22":"deserta","r21":"SUBJ","r22":"PRED"},

{"w11":"χώρα","w12":"ἦν","r11":"SUBJ","r12":"V","w21":"regio","w22":"erat","r21":"SUBJ","r22":"V"},

{"w11":"ἐρήμη","w12":"ἦν","r11":"PRED","r12":"V","w21":"deserta","w22":"erat","r21":"PRED","r22":"V"},

{"w11":"","w12":"","r11":"","r12":"","w21":"iter","w22":"faciebat","r21":"ACC","r22":"V FINIT"},

{"w11":"οὐκ","w12":"ἄνυδρος","r11":"NEG","r12":"PRED","w21":"non","w22":"inaquosa","r21":"NEG","r22":"PRED"}}

Cilvēkam tīkamākā formā šī tabula būtu šāda:

παρά θάλατταν	PREP ACC	secundum mare	PREP ACC
Ἀλέξανδρος ἐπορεύετο	SUBJ V	Alexander faciebat	SUBJ V
χώρα ἐρήμη	SUBJ PRED	regio deserta	SUBJ PRED
χώρα ἦν	SUBJ V	regio erat	SUBJ V
ἐρήμη ἦν	PRED V	deserta erat	PRED V
—	—	iter faciebat	OBJ V FINIT
οὐκ ἄνυδρος	NEG PRED	non inaquosa	NEG PRED

Piezīme: Apskatiet šos piemērus. Te ir dažas rindas problemātiskas.

4. uzdevums

Izveidot morfoloģisko datu bāzi!

Par morfoloģisko datu bāzi uzlūkosim tādu, kas fiksētai valodai atpazīst tās formas, nosaka valodas formu morfoloģisko informāciju. Pirmo šāda tipa datu bāzi patapinājām Whitaker Words, bet tad izveidojām savu MS Access tipa datu bāzi. To arī lietojām. Claude izveidojam no šīs wdb.db mūzdienīgā versijā. Citu izveidojām priekš armēņu valodas ar ChatGPT, Gemini un Perplexity palīdzību, vocabulary.json, kas satur visas formas armēņu tulkojumiem Šarla Pero

astoņām pasakām, kas satur 4021 formu. Ja kāda forma šajos tekstos nav kādam vārdam, tad tas netiks atpazīts, lietojot šo. No tās Claude ir izveidojis db sarla_pero_wordforms.db. Varbūt dažkārt mums būs izdevīgi par morfoloģisku datu bāzi json formu ar tikai vienu aizpildījumu, pie kam, studenta aizpildītu un ne obligāti pārbaudītu. Vai varbūt pat tukšu? {}, šādu? Dažādos uzdevumos un vingrinājumos skatīsimies, kas mums sanāk/izdodas.

Bet kā tad risināsim jautājumu par morfoloģisku db, ja tādas kādai valodai vēl nav? Meklēsim, jautāsim caur AI rīkiem. Bet otra pieeju būs, ko lietosim. Mums būs uzdevumi un vingrinājumi, kur meklēsim veidus, kā ar esošajiem līdzekļiem Syntra platformā iegūt, veidot vai sākt veidot savu morfoloģisku db kādai no mūsu valodām. Līdzīgi kā jau to darījām ar pasakām armēņu valodā.

5.uzdevums

https://scireprints.lu.lv/508/

Šis dokuments satur 4 kolonu tabulas, kas atbilst pēc struktūras mūsu Text_reading vai Student_reading rīkiem. To tā arī uzlūkosim un izveidosim šādu Student_reading rīku, bet šoreiz ar četrām kolonām divu vietā. Specificēsim, ka pirmā kolona, kur viscaur teksts klasiskajā sīriešu valodā, ir aktīvais teksts, bet pārējās trīs kolonas – pasīvais teksts. Kā to sapratīsim? Aktīvajā tekstā formas būs salinkotas, ar aizpildītu datu bāzi, vai ar neaizpildītu, vai daļēji, tukšu vai pustukšu, tātad. Bet pasīvie teksti būs tie, kas tādi arī būs, pasīvi, virs kuriem peles kustība neko neredzēs/nerādīs. Ar ko šis dokuments ir īpašs? Te ir dažādas valodas lietotas, ne tikai četras, kā varētu sagaidīt. Visas uzskaitīt pat pagrūti: klasiskā sīriešu, senebreju, latīņu, grieķu (LXX vai koinē), modernā ebreju/ivrīts, targumi, un variācijas. Fragmentus no šīs 4-tabulas izmantosim vingrinājumu veidošanā.

6. uzdevums

Veidosim Student_reading rīku ar divām aktīvām kolonām. Šeit divās kolonās abām valodām būs formu salinkojums, pie kam arī Studentu stūrītī būs divi paralēli panelīši, attiecīgi katrai kolonai/valodai formu specificēšanai. Paredzēsim iespēju, ka vienai no valodām, teiksim, kreisā pusē, būs pilns (daudzmaz) morfoloģisks atpazinums, tad tam studenta rāmīša vietā būs tikai info paplašinājums kā Text_reading rīkā. Šī tipa panelim var norādīt to ar to, ka tiek ievadīta attiecīgā db.

Veidosim šo kā shell tipa rīku, kur varēsim ielādēt jebkuru divkolonu tabulu, kādu mums jau ir pietiekami, lai izvēlētos dažādu valodu pārus. Šo tehnisko rīku mēģināsim izmantot, lai

veidotu/meklētu veidus, kā ar sintaktiskajiem un arī ne tik vienkāršiem gramatiskiem rīkiem radīt morfoloģiskās datu bāzes, to iedīgļus.

7. uzdevums

Veikt eksperimentu ar divvalodu aktīvajiem paneļiem, teiksim, grieķu, latīņu, kur latīņu ar morfoloģisko atbalstu. Meklēsim vārdu pārus no grieķu uz latīņu, kas sakrīt pēc nozīmēm un arī tekstā funkcionē atbilstīgi/saskanīgi. Jautāsim – cik mums tādu saišu uzstādīt, lai grieķu teksts kļūst morfoloģiski atpazīstams kādam 'gudram' gramatiskam rīkam. Jautāsim AI – kāds varētu būt šis 'gudrais gramatiskais rīks'?

Lai mums te arī ir paredzēta iespēja veidot izvēlēto vārdu pāru json fiksējumu, ti, līdzīgi kā uzdevumā 3 esam izveidojuši sintaktisku saiti. Lai to varam arī izvadīt un saglabāt iespējamiem turpmākiem eksperimentiem.

8. uzdevums*

Lai mums dokuments ir gramatikas grāmata kādai valodai, teiksim, gramatika latīņu valodai latviešu valodā. Mēs ar to lai strādājam. Kā iegūt divvalodu tabulu ar divām kolonām priekš Carpe Diem? Lūgsim kreisajā kolonā latīnisko tekstu un labajā kolonā latvisko tekstu, lai AI mums palīdz to veidot, vai paši nemākam saprogrammēt ar python? Bet tālāk prasīsim gramatikas skaidrojošo tekstu pielietot, kā to grāmata prasa.

Lai šajā uzdevumā esam formulējuši vispārīgo uzdevumu, par kuru runājām [18] seminārā angļu valodā kaut kur astoņdesmito gadu beigās, kur spriedām par to, ka pie attīstīta AI mums gramatikas grāmata kļūs aktīva un to lasītājs lasīs atbilstoši savam gramatikas izpratnes līmenim. Šādu pieeju varēsim jau lietot atsevišķos savos vingrinājumos, kur varam ņemt kādas konkrētas gramatikas grāmatas, kādās konkrētās gramatikas sadaļās, kaut vai teiksim kaut kur morfoloģijā, kur skatīsim kādas kategorijas locīšanu, vai skatīsim kādu sintaktisko konstrukciju, teiksim, infinitīvu ar akuzatīvu, vai jebko.

Atcerēsimies – 8. uzdevums, kur mēs noformulējam savu pieeju visvispārīgākajā veidā vai tuvu tai. Vai var vispārīgāk? Kāpēc ne? Uzprasiet to formulēt AI!

9. uzdevums

https://lingua.id.lv/alex/Syntra_Phaedrus_reader.html

Lai mums doti teksti valodā, ko apgūstam vai mācām citiem. Lai veidojam šiem tekstiem vārdkopu virknes. Veidosim vārdkopu vienkāršā veidā, kā to darījām jau kādreiz, piemēram, kad runājām par to Rekonstrukcijā 3. Numurēsim vārdus tekstā pēc to pozīcijas tekstā sākot skaitīt vārdus no kādas fiksētas vietas, piemēram, no fragmenta (vai fabulas, ja tā ir kā dotajā piemērā sākumā) sākuma. Tad frāzes būs kodētas ļoti vienkārši. Teiksim, teikumam

Fame coacta ***vulpes*** *alta* ***in vinea uvam adpetebat****, summis* ***saliens*** *viribus,*

kur mums jau frāze vai teikuma daļa, kas sintaktiski saistīta, to gribam formēt kā vārdkopu mūsu notācijā, tad to darām šādi, ja skaitīsim vārdus no šī teikuma sākuma:

3 **vulpes** 5 **in vinea uvam adpetebat**, 10 **saliens** .

Ko šāds kodējums mums dod? Tas dod viennozīmīgu atbilstību vārdkopā vārdiem pret pamattekstu, jo ievērosim, ka vārdi te vienā tekstā var atkārtoties, bet pozīcijas numurs nofiksē katru formu perfekti. Kā šo vienkāršo atbilstību izmantot? Mēs varam šajā tekstā kopā ar dotajām vārdkopām ar peli sameklēt un fiksēt kādu mūsu izraudzīto vārdkopu. Pie kam to varam darīt no abām pusēm, ja tās izvēlamies tā kā augšā jau realizētā programmā.

Skatīsim no kreisās puses, kur vārdkopu izvēle nav triviāla, jo viena forma var ieiet vairāk kā vienā vārdkopā – vispirms mums visas šīs vārdkopas jāatrod, pēc tam tās kā izvēli jāizkārto zem šīs formas, un tad ļaut lietotājam jau klikšķināt uz kādas vienas vārdkopas izvēlnē. Vienkārši? Bet kā varam te izmantot to, ka mūsu vārdi ir apveltīti ar speciālu numerāciju, pozīcijas numuru? Vispārīgākā gadījumā tā varētu būt arī adrese. Kā mēs atradīsim visas vārdkopas, kur mūsu fiksētais vārds ieiet? Nu, jā, jāmeklē tās vārdkopas, kurās ir numurs, kas atbilst izvēlētajam vārdam. Tas ir, jāiet cauri visam vārdkopu sarakstam un jāatlasa attiecīgas numurs/pozīcija? Jā, ja neko neveidojam un daudz galvu nelauzam. Bet varbūt izveidot struktūru, kas atrod visus vārdus ar šo numuru? Kas tā būtu par struktūru? Labi, te uz vietas to nerisināsim. Uzdosim uzdevumu, noformulēsim to. Tā varētu būt grafiska struktūra vai koks vai kas specifiskāks? Man nāk prātā tādi termini kā heš-tabulas/struktūras. Tie ir koki? Vai te nav vērts ar tādiem krāmēties? Bet mēs te risinām uzdevumus kopā ar AI. Varbūt ka ir vērts krāmēties?

10. uzdevums

Ap 2000. gadu, programmējot Borland Delphi vidē, eksperimentējot tika izveidots brauzeris ‘Uva”, ko veidoja tikai viena B.Delphi V. Comp. Proti, html brauzeris, kurš vienīgi paņēma kādu dotu html linku un caur šo demo brauzeri to izlika internetā.

Šodien šo pašu demo varētu jau veidot ar zināmu lietderīgu funkcionalitāti. Lai šim brauzerim dialoga logi ir tieši tie uzdevumi, kas izveidoti šeit šajā krājumā. E.g. lai vienā nolaižamā ekrānā ir visi vingrinājumi, nākamajā ir armēņu vingrinājumi utt. T.i. brauzeris būs paredzēts tika noteiktiem interneta linkiem, bet tieši tiem, kas atbilstu šī krājuma uzdevumiem un vingrinājumiem. Vai kāds LLM jau nepietektos šādu brauzeri "Uva" realizēt?

https://lingua.id.lv/alex/Uva_browser.html

Šis ir variants, ko izveidoja Claude. Aplūkosiem to, kā pirmo variantu papētīsim, cik mums tas noderīgs, koriģēsim. Palīdziet mums pie tā!!!

11. uzdevums

Ievedot īpašu brauzeri "Uva" kā uzdevumā 10, mēs varam specificēt uzdevumus mūsu Syntra platformas rāmjos. Tie būs gan lingvististiski kā vingrinājumi valodās vai platformas rīku lietošanā, gan programēšanā, lai radītu jaunus papildinājumus platformai un šai pieejai. Vai varam jau runāt par jaunu pieeju uzdevumu specifikācijā, arī kā programēšanas rīku? To lai diskutējam kopā ar AI rīkiem.

Literatūta un avoti [U]

[U,1] Ukrainas atbalstam, https://lingua.id.lv/art/publ.war.html

Uzdevumi Phython programmēšanā

1.uzdevums

Veikt 1. uzdevumu no Wolfram Mathematica programmēšanas sadaļas, tikai kodējot Python.

Proti, no grafa, kas dots adjacences saraksta reprezentācijā, iegūt leksikografiski sakārtotu šo pašu reprezentāciju. Pieņemam, ka dotajā reprezentācijā leksikografiskā kārtība varēja nebūt.

Mūs šis ļoti vienkāršais uzdevums interesē, vai to var izpildīt LLMi, precīzi pieturoties pie uzdevuma specifikācijas.

Bet šis uzdevums kalpos arī kā rosinājums nākamajam uzdevumam par frāžu lasīšanu lasītājā.

2. uzdevums

https://lingua.id.lv/alex/Syntra_Phaedrus_reader.html
Lai aplūjojam Faidra fabulu lasītāju ar dotām izdalītām frāzēm gandrīz visām fabulām šeit. Lai ņemam vērā iepriekšējo grafa adjacences saraksta uzdevumu. Vai te nav līdzīga situācija? Izrādās, ka vairāk nekā līdzība. Atrisiniet šo. Aiciniet AI palīgā.

3. uzdevums

https://lingua.id.lv/alex/Syntra_Phaedrus_reader.html

Lai turpinām no iepriekšējā uzdevuma un mums Faidra teksti, piemēram, šāda rinda Fame coacta vulpes alta in vinea uvam adpetebat un dotas vārdkopas/frāzes no teksta, pie kam formā – 3 alta 4 in 5 vinea, kur vārdiem/formām priekšā ciparai, kas norāda vietu pēc kārtas teikumā. Ievērosim, ka šādi kodējot vārdkopu, cipari būs vienmēr augošā kārtībā, un tas pats būs, ja liksim programmai izanalizēt doto tekstu, katram vārdam/formai tāpat piešķirot numuru, kas korelēs ar vārdkopu kodējumu numuriem. Un, lūk, ja gribēsim tagad izcelt/renderēt vārdkopu no teksta, lietosim kādu ļoti primitīvu algoritmu, nu gluži kā to (tkpat primitīvu), ko devām 1.uzdevumā. Veiciet šo uzdevumu.

Zemāk apskatieties, kā to pasniedz Claude Sonnet 5.

```
import re

TEXT = "Fame coacta vulpes alta in vinea uvam adpetebat"
PHR = "1 Fame 2 coacta\n3 vulpes 5 in\n4 alta 5 in 6 vinea"


def tokenize(text: str) -> list[dict]:
    return [{"pos": i + 1, "text": w} for i, w in enumerate(text.split(" "))]


def find_and_render(words: list[dict], phr: str, pos: int) -> tuple[str, list[int] | None]:
    for line in phr.split("\n"):
        positions = [int(m.group(1)) for m in re.finditer(r"(\d+)\s+(\S+)", line)]
        if pos not in positions:
            continue
        parts = []
        hi = 0
        for w in words:
            if hi < len(positions) and positions[hi] == w["pos"]:
                parts.append(f"*{w['text']}*")
                hi += 1
            else:
                parts.append(w["text"])
        return " ".join(parts), positions

    return " ".join(w["text"] for w in words), None


def main() -> None:
    words = tokenize(TEXT)

    print("Teksts (ar pozīcijām):")
    print(" ".join(f"{w['pos']}:{w['text']}" for w in words))
    print()
    print("Ievadi vārda pozīcijas numuru, lai redzētu tā frāzi (q -- iziet).")

    while True:
        raw = input("\n> ").strip()
        if raw.lower() in ("q", "quit", "exit"):
            break
        if not raw.isdigit():
            print("Lūdzu ievadi skaitli (vai 'q', lai iziet).")
            continue

        pos = int(raw)
        rendered, positions = find_and_render(words, PHR, pos)

        if positions is None:
            print(f"Pozīcijai {pos} nav neviena frāze.")
        else:
            print(f"Frāze: {positions}")
        print(rendered)


if __name__ == "__main__":
    main()
```

Literatūta un avoti [UP]

[UP,1] Ukrainas atbalstam, https://lingua.id.lv/art/publ.war.html

Uzdevumi Wolfram Mathematica programmēšanā

Jau vairāk kā 15 gadus programmēju tikai Wolfram Mathematica vidē.

1.uzdevums

Uzrakstīt kodu WM grafa kaimiņu leksikografiskai sakārtošanai, ja grafs ir kodēts adjacency list reprezentācijā. Piemēram, ja graphs ir dots kā

$$\text{unordgraph} = \{\{2,4,3\},\{4,1,3\},\{2,1,4\},\{3,2,1\}\},$$

leksikografiski sakārtot to, proti

$$\text{unordgraph} = \{\{2,3,4\},\{1,3,4\},\{1,2,4\},\{1,2,3\}\},$$

2. uzdevums

https://lingua.id.lv/alex/Syntra_Phaedrus_reader.html

Risinām to pašu, ko Python kodēšanā skatījām 3. uzdevumā.

Skatāmās/renderējamās frāzes/vārdkopas ir

```
phr = "1 Fame 2 coacta\n3 vulpes 5 in 6 vinea\n3 vulpes 7 uvam 8 adpetebat";
```

Kā to darām WM?

```
text = "Fame coacta vulpes alta in vinea uvam adpetebat";
phr = "1 Fame 2 coacta\n3 vulpes 5 in 6 vinea\n3 vulpes 7 uvam 8 adpetebat";
outputFile = "D:\\taksis1_server\\ai.prog.codes\\wm.minimal_demo_commented.html";

escapeJS[s_String] := StringReplace[s, {
    "\\" -> "\\\\",
    "\"" -> "\\\"",
    "\n" -> "\\n"
  }];

htmlTemplate = "<!DOCTYPE html>
<!--
 Minimal pattern: position-indexed phrase highlighting.

 Two independent pieces of data are given:
  1. `text`  -- a plain sentence.
  2. `phr`   -- a list of \"phrases\" (syntactic word groups), each phrase
          written as a flat sequence of \"<position> <form>\" pairs,
          one phrase per line. The `<form>` part is only there for
          human readability; only the `<position>` numbers are
          actually used by the code below.

 The core idea: every word in `text` gets a 1-based POSITION index just by
 its order in the sentence. A phrase in `phr` refers to specific words not
 by their text (which could repeat) but by that same position index. This
 gives an unambiguous way to link the two independent pieces of data
 together.

 This file was generated by Wolfram Mathematica (WM string export), not
 hand-written -- the underlying HTML/JS is identical to the Python-
 generated version.
-->
```

```
<pre id=\"t\" style=\"font-family:serif;font-size:1.4rem;cursor:pointer\"></pre>
<script>
const text = \"@@TEXT@@\";
const phr = \"@@PHR@@\";

// Step 1 -- tokenize: split the sentence into words, each tagged with its
// 1-based position in the sentence (this numbering must match whatever
// convention was used to write `phr`).
const words = text.split(\" \").map((w, i) => ({ pos: i + 1, text: w }));

// Step 2 -- parse phrases: for each line in `phr`, pull out just the
// position numbers (ignore the word-text part, which is display-only).
const groups = phr.split(\"\\n\").map(line =>
  [...line.matchAll(/(\\d+)\\s+(\\S+)/g)].map(m => +m[1])
);

// Step 2b -- build a position -> group lookup ONCE (same O(n+m) \"bucket\"
// idea as the graph-sorting exercise), instead of scanning every group
// with .find()+.includes() on every single click.
const posToGroup = {};
groups.forEach(g => g.forEach(p => { posToGroup[p] = g; }));

// Step 3 -- render: wrap each word in a clickable <span>, tagging it with
// its own position via a data-pos attribute so it can be looked up later.
document.getElementById(\"t\").innerHTML = words
  .map(w => `<span data-pos=\"${w.pos}\" onclick=\"mark(${w.pos})\">${w.text}</span>`)
  .join(\" \");

// Step 4 -- on click: O(1) lookup of the clicked word's phrase, then a
// single linear pass over the spans (already in ascending-position DOM
// order) merged against the phrase's positions (also ascending) -- no
// generic .includes() scan per span.
function mark(pos) {
  const group = posToGroup[pos] || [];
  let hi = 0;
  document.querySelectorAll(\"span\").forEach(el => {
    const p = +el.dataset.pos;
    if (hi < group.length && group[hi] === p) {
      el.style.background = \"yellow\";
      hi++;
    } else {
      el.style.background = \"\";
    }
  });
}
</script>
";

finalHtml = StringReplace[htmlTemplate, {
      "@@TEXT@@" -> escapeJS[text],
      "@@PHR@@" -> escapeJS[phr]
    }];

Export[outputFile, finalHtml, "Text"];
Print["Izveidots: ", outputFile];
```

Literatūta un avoti [UWM]

[UWM,1] Ukrainas atbalstam, https://lingua.id.lv/art/publ.war.html

Uva kā programmēšanas rīks

Pie uzdevumiem uzdevumā 10 un tālāk uzdevumā 11 ieviesām pārlūku/brauzeri Uva un domu, ka to var izmantot kā programmējamu rīku vai attīstīt ideju par Uva kā uzdevumu specifikācijas rīku ar savu valodu, līdzīgi kā tas notiek WM vidē, kur no vienkāršas rēķināšanas izaug programmēšana un tad arī valoda.

Vai šāda pieeja ir produktīva tādā mini vidē, bet ar valodu apmācības ievirzi, tradīciju, un rekonstruējot vecu (nederīgu?,nestrādājošu? vāji strādājošu) programmatūru, to var risināt, darbojoties kopā ar AI rīkiem. Uzdevumi gan studentiem, gan AI.

1.uzdevums

Rakstīsim Uva[A][6], vai saīsināti Uva[A,6] un esam izsaukuši pasaku armēņu valodā par Fejām. Bet vēl ir jāuzraksta programma/interpretators?, kas šo uva konstrukciju atpazīst un izpilda, vai jāsāk veidot mums mūsu Uva programmēšanas valoda.

2. uzdevums

Var mēģināt domāt, kā Uva attīstīt kā valodu, bet diez ko daudz man prātā nenāk. Izsaukt esošās valodu un uzdevumu saites, lai tās atvērtu, kas ja ir pieejamas tīmeklī, jā, - tā jā. Bet ko izdomāt vairāk? Jādomā!

Bet cita ideja nāk prātā. Tā kā mēs sadarbojamies ar AI un arī ar aģentiem, un arī ar eventuālu iespēju veidot tādus aģentus, tad mums ir iespēja kā parametrus Uva uzdot arī AI piesaisti. Kādu? Lai uzrakstām – Uva[A,6][Gemini] un sagaidām izpildi. Kas notiks? Jā, ko mēs prasīsim interpretatoram, kam būs jāiztulko šī komanda. Izsauksim Gemini aģentu? Jā, kaut ko tā. Vispirms šie aģenti būs jāuzbūvē. Tad vienkārši LLM nosaukuma Gemini vietā parametros rakstīsim aģenta vārdu. Kopā ar studentiem un AI rīkiem to arī darīsim!

Rekonstrukcijas

Pie rekonstrukcijām pieskaitām tos uzdevumus, kurus pamatā jau esam sākuši veikt ar Claude palīdzību, ko turpināsim, jau tālāk ne tikai apmierinoties ar lingvistiskiem darbiem.

Rekonstrukcija 1

https://lingua.id.lv/alex/lingua-dicttools.html

Latin Dictionary Tools Page, tā tika izveidota ap 2000. gadu vai nedaudz vēlāk. Par to tika stāstīts rakstā un Rīgā notikušajā konferencē. Tā tika izveidot asp (active server pages) tehnikā kā aktīva interneta lapa. Tajā laikā jau tā saucamā interneta programēšana notika pilnā sparā, no kā es neko nesapratu, bet šo lapu man palīdzēja veidot profesionāls programmētājs, Uldis Šneiders, parādīja, kas man kur jāraksta un, man akurāti viņa norādījumiem sekojot, tapa interneta lapa. Kādu laiku atpakaļ mūsu LATNETs paziņoja, ka asp lapas netiek uzturētas un man vajadzēja pārprogramēt. Es to neizdarīju, līdz iestājās AI laiki un Claude man šo rekonstrukciju veica. Tā vēl nav nobeigta, jo lapa vēl jātestē. Ne viss ir rekonstruējies līdz tam līmenim, kā tas bija toreiz. Lapa izmanto Whitaker datubāzi, ko mums ar Uldi Šneideru bija jāizveido. Kā tā internetā online darbojas, https://latin-words.com/, varat apskatīt un patestēt. Lai šo dabubāzi veidotu, mums bija jāprasa Viljama Vitakera, atvaļināta NATO pulkveža, atļauju, ko arī saņēmām.

Rekonstrukcija 2

Šī rekonstrukcija turpina pirmo rekonstrukciju, bet te Claude bija prasīts atjaunot, maksimāli pietuvinotos oriģinālajam lapas izskatam. Te, lūk, rezultāts. Lapa apmēram, ne pilnībā, tā arī izskatījās:

https://lingua.id.lv/alex/lingua-historical-2004.html

Rekonstrukcija 3

Izveidojām rekonstrukciju no prog. Syntra vienas sadaļas, kur ņemam ap 44 Faidra papildinātos fabulu tekstus un šo fabulu frāžu sadalījumu. Šie sadalījumi izveidoti vēl pirms 2000.gada

visdrīzāk ar rokām, kaut gan šodien jau grūti to noteikt, jo materiāls tika ņemts no lapas Reading Exercises in Latin. Izmantots tiek veids, kā tika kodētas frāzes. Tika izmantota vienkārša formula, frāzes pierakstot kā virknes <pozīcija forma>, kur pozīciju numerācija tiek ņemta vienas fabulas rāmjos. Frāžu dalījums ietver gan pašu fabulu, gan Smeltera papildinājumu fabulai. Izveidotajā pirmajā rekonstrukcijā morfoloģiskā atbalsta nekāda nav, bet tiek pievienots ieskandinājums vārdkopām.

Rekonstrukcija 4

https://italozeps.github.io/ai-connectors-hub/TRM/syntra/

Syntra pirmā rekonstrukcija, kur morfoloģiskā analīze un sintaktiskā, kur sintatiskā sakrīt ar to, ko atklājam Rekonstrukcijā 3.

Literatūta un avoti [R]

[R,1] Ukrainas atbalstam, https://lingua.id.lv/art/publ.war.html

Literatūra un avoti

Literatūras avotos iekļaujam arī tos, kas avotā [1] jau ierādīti, kas tieši nepieciešami šeit šajā uzdevumu krājumā. Bet jāskata arī citi avoti no [1], kur te visi nav izcelti.

1. Mācīšanās platforma Syntra un lingvistiskā spēle Carpe Diem: piemēri, vingrinājumi, eksaminēšanas plāni un MI atbalstīti programmatūras izstrādes piemēri, The Syntra Learning Platform and the Carpe Diem Language-Learning Game: Examples, Exercises, Assessment Plans, and AI-Assisted Software Development, sast. D.Zeps, https://lingua.id.lv/art/war.101.pdf, https://scireprints.lu.lv/569/, 2026
2. Dainis Zeps, The Learning of Ancient Languages as (super)Human Effort, https://web-archive.southampton.ac.uk/cogprints.org/6624/index.html, https://www.researchgate.net/publication/28765445
3. Dainis Zeps, Latin Dictionary Tools in Internet, https://hal.science/hal00412265v1/file/ladito.zeps.pdf , 2004
4. Dainis Zeps, ELEKTRONISKIE VINGRINĀJUMI ATDZĪVINA STUDENTAM LATĪŅU VALODU, tēzes, HLT conference, 2004, https://www.ltn.lv/~dainize/MathPages/el.ving.lat.val.thesis.pdf
5. Vingrinājumi Sengrieķu/koine valodā, sast. D. Zeps, https://www.ltn.lv/~dainize/inform_theol/Koine%20vingrinajumi.pd
6. William Whitaker's Words, https://en.wikipedia.org/wiki/William_Whitaker%27s_Words
7. Dainis Zeps, Runcis Zābakos. Armēņu gramatika, armēņu gramatiskais atveidojums uz Šarla Pero pasaku tekstiem, https://www.researchgate.net/publication/405287878 , 2026
8. Dainis Zeps, Runcis Zābakos. Armēņu gramatika 2. daļa. Corpus secundum, armēņu gramatiskais atveidojums uz Šarla Pero pasaku tekstiem, https://www.researchgate.net/publication/405288265 , 2026
9. G.J. Cēzars, II. Grāmata, Piezīmes par gallu karu : (Burtisks un literārisks tulkojumi, gramatikas paskaidrojumi, vārdu pamatformas un piezīmes), sastādījis Voldemārs Vītiņš, Rīga : L.ū. Studentu padomes grāmatnīca, 1939. 93 lpp.
10. G.J. Cēzars, IV. Grāmata, Piezīmes par gallu karu : (Burtisks un literārisks tulkojumi, gramatikas paskaidrojumi, vārdu pamatformas un piezīmes), sastādījis Voldemārs Vītiņš, Rīga : L.ū. Studentu padomes grāmatnīca, 1939. 93 lpp.
11. Phaedri Augusti Liberti Fabulae Aesopiae selectae. Faidra fabulas : izlase ; ievads, teksts ar vingrinājumiem, paskaidrojumi, vārdnīca un sintakses pielikums / apstrādājis Gr. Smelters, Rīga : Valters un Rapa, 1939. 152 lpp.

12. Dainis Zeps, LINGUA GRÆCA NOVI TESTAMENTI, TEXTUS LEGENDI IN AUDITORIO, https://www.ltn.lv/~dainize/inform_theol/Koine%20Chrestomatija%20I.pdf
13. Ernst Jenni, HEBRAICUM VETERIS TESTAMENTI, (0-8 lekcijas), Latviešu tulkojums D. Zeps, Rīga, 1988. https://www.ltn.lv/~dainize/hebraicum/Jenni.0-8.latv.pdf
14. Dainis Zeps, Mājas lasīšana, paralēli latīņu-latviešu teksti, https://www.ltn.lv/~dainize/inform_theol/Home_reading.pdf
15. Dainis Zeps, Evaņģeliji un psalmi, https://scireprints.lu.lv/508/ , 2023
16. Sīriešu Jaunās Derības dažu grāmatu leksiskā atslēga, sast. D. Zeps, http://www.ltn.lv/~dainize/gpt.ai.language.model/Syriac/syriac.comment.hal.pdf
17. Readings in Classical Syriac: Parallel Syriac - Greek text. Tutorial for students of Classical Syriac, comp. D. Zeps, Riga, 2006, http://www.ltn.lv/~dainize/inform_theol/Aramaic/evang.vingr.pdf
18. Dainis Zeps, Hendersons, semināri…., karš, https://lingua.id.lv/art/Henders.semi.pdf
19. Ukrainas atbalstam, https://lingua.id.lv/art/publ.war.html

© Dainis Zeps, lingua.id.lv/art/ai.exercises.pdf, 2026