Tartu Ülikool
Loodus- ja täppisteaduste valdkond
Arvutiteaduste instituut
Infotehnoloogia eriala

Priit Vallap

Tekstülesannete automaatne lahendaja

Bakalaureusetöö (9 EAP)

Juhendaja: Sven Aller, MSc

Tartu 2022

Tekstülesannete automaatne lahendaja

Lühikokkuvõte:

Bakalaureusetöö eesmärgiks on luua veebiprogramm, mis võimaldab etteantud matemaatilist tekstülesannet lahendada automaatselt, tagastades nii õige vastuse kui ka lahenduskäigu. Osana tööst on vaja ka piiritleda sobivate ülesannete raskusaste ja vorm. Programmi peamine töövahend on EstNLTK keeletöötluse teekide kogum. Tulenevalt piirangutest EstNLTK poolelt on programm kirjutatud Pythoni programmeerimiskeeles.

Võtmesõnad: Tekstülesannete lahendamine, EstNLTK, Python, keeletehnoloogia

CERCS: P175 Informaatika, süsteemiteooria

Automatic mathematical word problem solver

Abstract:

The aim of the bachelor thesis is to create a web application, which would allow the automatic solving of mathematical word problems. The application will return both the correct answer as well as the solution. Finding out the difficulty and form of solvable problems will be a part of the thesis. The main tool for the application will be the EstNLTK set of libraries for language processing. Due to the restrictions set by EstNLTK, the programming language used for this thesis is Python.

Keywords: Word problem solver, EstNLTK, Python Language Technology

CERCS: P175 Informatics, systems theory

SISUKORD

Sissejuhatus

Matemaatika on õppeaine, millega puutub kokku iga haridusteed läbiv õpilane. Ühe osana õpilaste loogilise mõtlemise oskuse arendamisest ning arengu hindamiseks kasutatakse matemaatilisi tekstülesandeid. Tekstülesanded kujutavad endast tavaliselt paarist lausest koosnevat teksti, milles on esitatud ülesande lahendamiseks vajalikud andmed ning lahendamist vajav probleem või küsimus.

Käesoleva bakalaureusetöö sisuks on luua veebirakendus, mis on võimeline kasutaja poolt vabalt sisestatud, aga siiski tekstülesannete reeglistikele vastavat teksti töötlema. Programm on võimeline leidma sisendina antud tekstülesandest algandmed ning neid vastavalt püstitatud ülesandele töötlema, tagastades nii vastuse kui ka lahenduskäigu. Programm on leitav veebirakendusena aadressil https://www.vallap.eu/lahendaja ning on võimeline lahendama kasutaja poolt vabalt sisestatud tekstülesande põhjal nelja erinevat tüüpi tekstülesandeid.

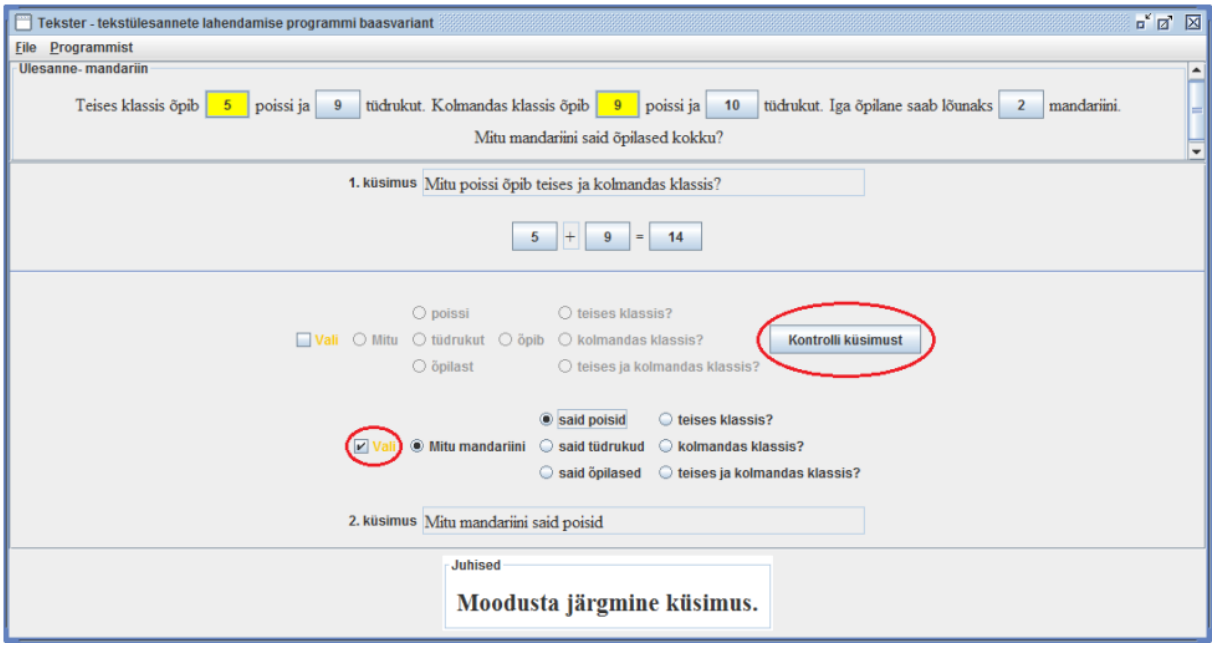
Töö koosneb kaheks suuremaks peatükist. Esimeses peatükis antakse ülevaade sarnastest programmidest, mis on loodud tekstülesannete lahendamiseks. Teises peatükis keskendutakse käesoleva töö raames loodud tekstülesannete automaatsele lahendajale. Peatükis selgitatakse tekstülesannete üldist lahendamisprotsessi, lahendaja peamise tööriista EstNLTK toimimist ning ka loodud lahendaja tööprotsessi.

1. Sarnased lahendused

Tekstülesannete automaatseid lahendajaid on loodud ka varem. Käesoleva töö raames tutvustame täpsemalt neist kolme, millest kaks on inglisekeelsed ning üks eestikeelne. Inglisekeelsetest lahendustest tutvustame Symbolabi ning MathCelebrity veebirakendusi. Eestikeelse tekstülesannete lahendaja näol on tegemist rakendusega, mis on loodud bakalaureusetöö käigus ning saanud ka kaks täiendust, samuti bakalaureusetööde raames.

1.1. Tekster

Teksteri näol on tegemist ainukese eestikeelse tekstülesannete automaatse lahendajaga. Tekster loodi 2006. aastal Evari Koppeli [1] poolt bakalaureusetöö käigus. Programm on hiljemalt saanud ka funktsionaalseid täiendusi esmalt 2013. aastal Joosep Kibali [2] ning seejärel 2014. aastal Katrin Valdsoni [3] poolt. Mõlemad programmi täiendused olid samuti võimelised sisendiks antud tekstülesannetest andmeid eraldama, neid töötlema ja etteantud ülesandeid lahendama.



Joonis 1. Teksteri kasutajaliidese näidis [2, lk 13].

Kuigi pealtnäha sooritavad nii E. Koppeli alustatud ja J. Kibali ning K. Valdsoni täiendatud Tekster kui ka käesoleva töö käigus loodav lahendaja sama ülesannet, siis programmi sisu ja metoodika poolest lähenetakse tekstülesannetele erinevalt. Tekster oli küll võimeline töötlema talle sisendina ette antud tekstülesandeid (joonis 1), kuid töötles üsna rangelt ette määratud andmestikku, kuhu olid muutujad juba ette määratud.

Näide E.Koppeli lõputööst [1] Teksterile püstitatud ülesandest, mida programm oli võimeline algsest tekstifailist töötlema:

```
-=TEKSTER=-
!TEKST=
Teises klassis õpib #a=8 poissi ja #d=10 tüdrukut.
Kolmandas klassis õpib #c=7 poissi ja #d=10 tüdrukut
Mitu õpilast õpib teises ja kolmandas klassis kokku?
!VEERUD=
Mitu
poissi&tüdrukut&õpilast
õpib&magab
teises klassis?&kolmandas klassis?&teises ja kolmandas klassis kokku?
!VASTAVUSED=
a=Mitu poissi õpib teises klassis?
b=Mitu tüdrukut õpib teises klassis?
c=Mitu poissi õpib kolmandas klassis?
d=Mitu tüdrukut õpib kolmandas klassis?
a+c=Mitu poissi õpib teises ja kolmandas klassis?
b+d=Mitu tüdrukut õpib teises ja kolmandas klassis?
a+b=Mitu õpilast õpib teises klassis?
c+d=Mitu õpilast õpib kolmandas klassis?
(a+b) + c+d=Mitu õpilast õpib teises ja kolmandas klassis?
!VASTUS=
A+b+c+d
```

Nagu Teksteri algfaili näitest osutub, on nii andmed muutujate kujul kui ka kõik lahenduskäigud algfailis kindlalt ette struktureeritud ning otseselt määratud.

J. Kibali [2] ja K. Valdsoni [3] loodud täiendused Teksteri programmile ei muutnud programmi fundamentaalset ülesehitust. Endiselt vajas Tekster oma tööks ülaltoodud üsna rangelt ette kirjutatud sisendit. Täiendused puudutasid pigem programmi töö laiendamist ja kasutajaliidese elementide mugavamaks muutmist.

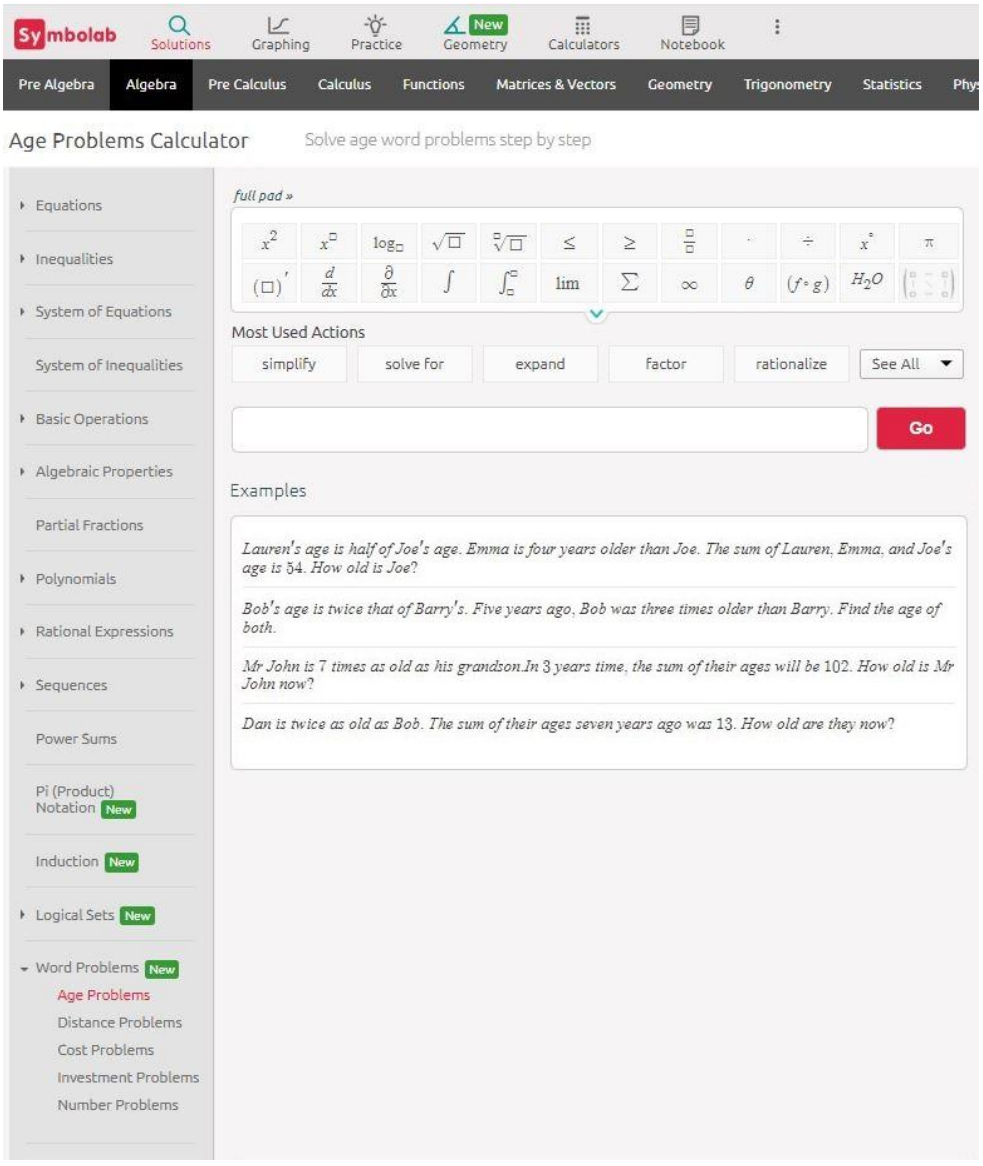
1.2. Symbolab

Teine sarnane lahendus on Symbolab loodud “Word Problems calculator” (“Tekstülesannete kalkukaator”) [4]. Sarnaselt eelmises peatükis kirjeldatud Teksterile on ka Symbolabi lahendus mõeldud pigem lahendama nende poolt juba ettemääratud ülesandeid, kus on võimalik andmete numbrilist väärtust muuta. Vabalt sisestatud ülesandeid ei ole kõnealune programm võimeline töötlema.

Kalkulaatoriga on võimalik lahendada viite alamkategooriasse kuuluvaid tekstülesandeid. Igas kategoorias on ka kindel arv näidisülesandeid, mida kalkulaator lahendab. Nendeks kategooriateks on:

1. “Age problems” - ülesanded inimeste vanustega;
2. “Distance problems” - ülesanded vahemaade ja teepikkustega;
3. “Cost problems” - ülesanded esemete hindade ja maksumustega;
4. “Investment problems” - ülesanded investeeringutega (laenud, intressid);
5. “Number problems” - aritmeetilised ülesanded.

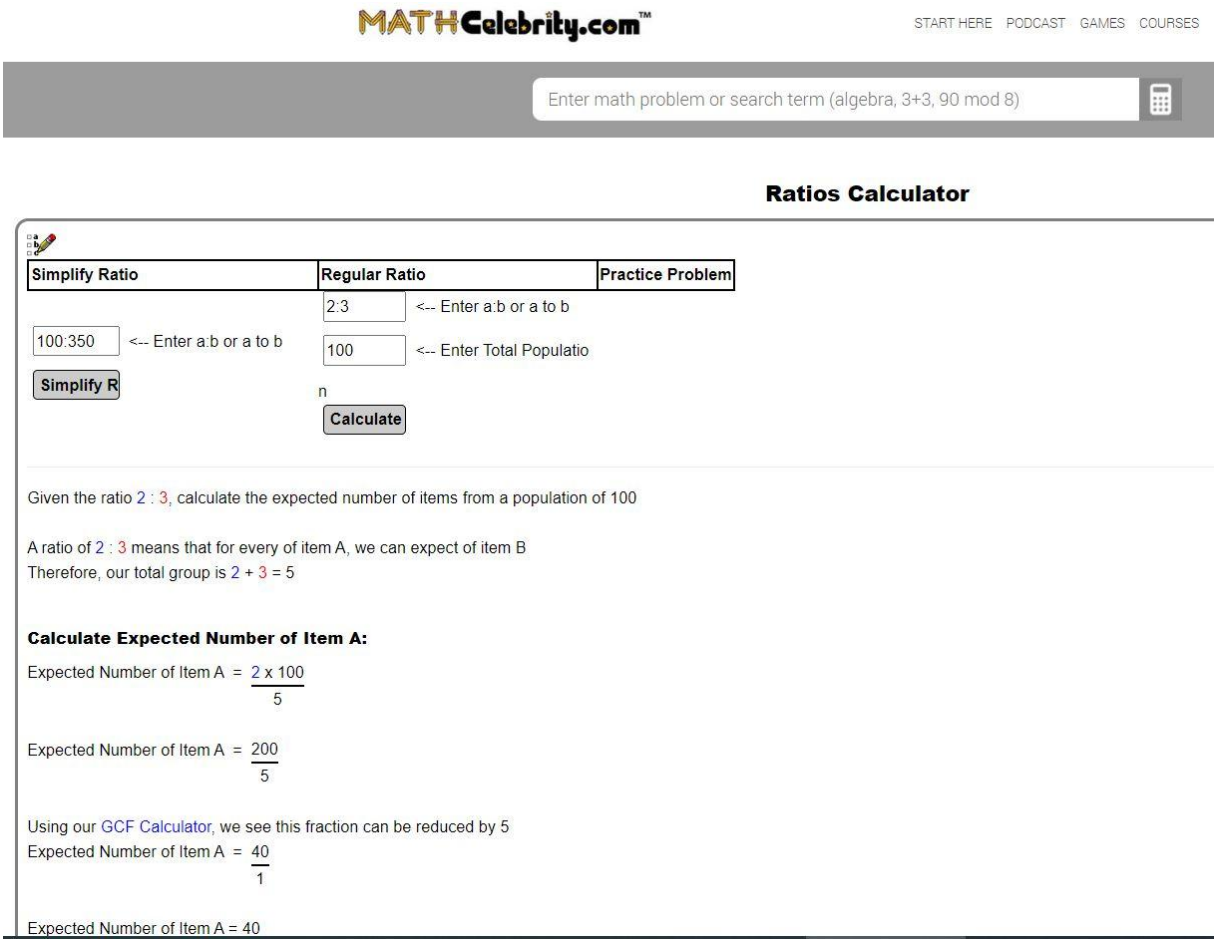
Joonisel 2 on näha kuvatõmmis Symbolabi pakutavast lahendusest ning ülesannete näidistena (“Examples”) on näha vanusega seotud ülesannete mallid.



Joonis 2. Symbolabi vaade ja vanuseülesannete näidised.

1.3. Mathcelebrity

Eelmistele rakendustele sarnast võimekust pakub ka autori ning ettevõtja 2007. aastal Don Sevciku poolt alustatud Mathcelebrity [5]. Autor kirjeldab Mathcelebrityt kui tööriista mis on abiks nii õpilastele kui ka nende lapsevanematele, kes matemaatiliste ülesannetega ise hätta jäävad. Mathcelebrity pakub kokku sadu erinevaid algülesandeid, millest 41 on klassifitseeritud kui tekstülesanded [6]. Sarnaselt eelmistele lahendustele on ka siin iga ülesande jaoks loodud uus lahendus (joonis 3), mis on võimeline töötlema ainult selle ülesande teksti.



Joonis 3. Mathcelebrity näidisülesanne.

Kui võrrelda omavahel Mathcelebrity ja Symbolabi lahendusi, siis Mathcelebrity pakub tunduvalt rohkem algülesandeid. Symbolabi puhul on aga kogu rakenduse lahendus kasutajakogemuse poole pealt palju lihtsam ja paremini läbi mõeldud. Mathcelebrity puhul tundub kasutajaliides olevat üsna konarlik ja mitte väga kasutajasõbralik.

1.4. Eelnevate lahenduste kokkuvõte

Kõigi kolme näite suurim erinevus käesoleva töö raames loodava lahendajaga tuleneb sisendite liigist. Nii MathCelebrity kui ka Symbolabi puhul olid ülesanded ette kirjutatud ja iga ülesande jaoks oli rakendusel ainult üks moodus sisendit töödelda, ehk ülesanded olid väga rangelt programmeeritud. Teksteri puhul oli sisendiks "täidetud lünkadega" sisendfail, kus kõik küsimused ja muutujad olid juba ette määratud. Esitatud näidetest lähimaks käesoleva töö käigus loodava programmiga saab lugeda Teksterit. Põhjuseks fakt, et Tekster oli ainukene eestikeelne sarnase otstarbega rakendus.

Käesoleva töö raames loodav lahendaja peamine erinevus eelmiste näidetega tuleneb sisendist. Siin on võimalik kasutajal sisestada vabalt valitud sisendit, mitte ei pea lahendama ainult ühte etteantud ülesannet. Üheks peamiseks aspektiks on EstNLTK olemasolu, mis töötati välja alles peale Teksteri rakenduse loomist ja täiustamist. EstNLTK annab võimaluse pakutavaid teeke ära kasutades luua olukord, kus sisendit töödeldakse nii, et igast sõnast on võimalik leida algvorm, määrata õige sõnatüüp ja vastavalt sellele sõnu vastavalt vajadusele liigitada. Sõnatüübi määramine saab olema üheks peamistest kasutatavatest tööriistadest sisendülesannete lausete ja sõnade klassifitseerimisel.

2. Tehnilised vahendid

2.1. HTML

Eesliidese ehk kasutajaliidese ülesehitusel on kasutatud peamiselt HTML-märgistuskeelt. Kasutajakogemuse parandamiseks on kujundusel kasutatud CSS-i. Lisaks on vähesel määral kasutusel JavaScript, täpsemalt jQuery. HTML'i eelistati teiste populaarsete veebiarenduse keelte, nagu PHP või JavaScript, ees, seet käesoleva programmi nõudlikus kasutajaliidese ja vajaliku lahenduse keerukuse suhtes ei ole ülemäära suur.

2.2. EstNLTK teekide kogum

EstNLTK on 2016. aastal Eesti Keeletehnoloogia Riikliku Programmi [7] rahastatud ning Tartu Ülikoolis loodud avatud lähtekoodiga teekide kogum, mis on mõeldud eestikeelsete tekstide ja lausete tuvastamiseks ning töötlemiseks. EstNLTK on eestikeelne vaste NLTK (Natural Language ToolKit - naturaalse keele tööriistakomplekt) [8] teekide kogumile, mille töökeel on inglise keel. EstNLTK kui projekti tuumaks on Pythoni EstNLTK teek, milles sisaldub [9]:

- eesti keele sõnestamine ehk sõnapiiride tuvastamine ehk üksustamine (tokeniseerimine);
- eesti keele lausestamine ehk lausepiiride tuvastamine;
- eesti keele osalausestamine ehk osalausepiiride tuvastamine;
- eesti keele lemmatiseerimine ehk sõnade algvormide (lemmade) määramine ning morfoloogiline analüüs ja ühestamine (liidestudes mugavalt vabamorfiga);
- sõnaliikide määramine;
- eesti keele morfoloogiline süntees (etteantud lemma ja grammatilise vormi põhjal õige sõnakuju tuletamine);
- nimeolemite e nimega üksuste tuvastamine eestikeelsest tekstist (ingl *NER* ehk *Named-entity recognition*);
- liidestus Eesti Wordnetiga;
- eestikeelsete ajaväljendite tuvastamine ning nende semantika esitamine (TIMEX3 formaadis);
- pindsüntaktiline analüüs ning sõltuvussüntaktiline analüüs:
    - masinõppepõhine analüüs MaltParseri abil;
    - reeglipõhine analüüs mooduli EstCG abil.

Käesolevas töös on kasutusel peamiselt Maltparseri masinõppepõhine süntaksianalüüs, mille abil on võimalik lauses välja selgitada sõnade omavahelised seosed ja alluvused. Lisaks on oluline lauses olevate sõnade lemmade, käänete ja sõnaliikide tuvastamine ning kasutamine [10]. Kõiki neid võimalusi pakub Maltparseri süntaksianalüüs.

Tabel 1. Käesolevas töös kasutatavad EstNLTK kihid [11].

Kiht:	**Vajab toimimiseks kihte:**	**Kirjeldus:**
tokens	-	Eeltöötlus - eraldab teksti märkideks
compound_tokens	tokens	Eeltöötlus - ühendab märgid liitmärkideks
words	tokens, compound_tokens	Eraldab teksti sõnadeks
sentences	compound_tokens, words	Eraldab teksti lauseteks
morph_analysis	compound_tokens, words, sentences	Sõnade morfoloogiline analüüs, kasutades Vabamorfi.
maltparser_conll_morph	sentences, morph_analysis	Eeltöötlus Maltparseri-põhisele süntaktsianalüüsile
maltparser_syntax	words, sentences, maltparser_conll_morph	Sildistab sõlutuvuste süntaktsianalüüsi kasutades MaltParserit

Enne Maltparseri süntaksanalüüsi tegemist on vajalik töödeldavad (sõne tüüpi) tekstid muuta Text tüüpi objektideks. Text-klassi objektid on EstNLTK töö kesksed komponendid [11]. Text objekti loomise hetkel ei ole selle küljes ühtegi kihti (layer). Kihte lisatakse meetodiga “tag_layer()”. Maltparseri süntaksanalüüsi tekitamiseks tuleb “[‘maltparser_syntax’]” lisada argumendina kihtide lisamise meetodi väljakutsele (“tag_layer([‘maltparser_syntax’])”). Peale meetodi väljakutsumist on lisandunud 7 kihti (Tabel 1) - vaikimisi määratakse Text objektile “tag_layer()” meetodi välja kutsumisel külge viis kihti ning kaks kiht lisanduvad maltparser_syntax kihti tekitades.

2.3. Python ja Flask

Tekstülesannete automaatse lahendaja selgrooks on maailmas populaarsuselt ja tööturu nõudluselt teine [12] programmeerimiskeel Python. Tulenevalt faktist, et EstNLTK on Pythoni põhjal üles ehitatud, ei ole teisi programmeerimiskeeli võimalik käesoleva projekti loomiseks kasutada. Eesliidese (Python) ja tagaliidese (HTML) sidumiseks on kasutusel samuti Pythoni põhjale kirjutatud Flask. Flask on mikroraamistik, mille eesmärgiks on lihtsustada veebirakenduste loomist Pythonis [13].

3. Tekstülesannete automaatne lahendaja

3.1. Tekstülesannete lahendamise üldine protsess

Õpilastele tekstülesandeid esitades oodatakse nende lahendamist üsna lihtsa printsiibi järgi: esmalt tuleb tekstist eraldada andmed, seejärel tuvastada vajalik tegevus ja/või küsimus ning lõpetuseks leida küsimusele vastus. Sama protsessi emuleerib ka loodud lahendaja. Selleks, et tekstülesanded oleksid õpilaste arengut toetavad, peavad nad didaktika vaatenurgast vastama teatud kriteeriumitele [14]:

1. Tekstülesande sisu peab olema tunnetuslikult arendava ja kasvatava iseloomuga.
2. Ülesandes kirjeldatav situatsioon peab olema eakohane, st peab olema lapsele tuttav, lähedane ja huvitav.
3. Arvandmed peavad olema tõepärased, valitud nii, et ülesanne oleks lahendatav. Arvude suurus peab vastama sellele kontsentrile, mida parasjagu õpitakse. Arvandmeid peab olema lahendamiseks piisavalt (st alla 2 neid olla ei saa).
4. Tekstülesannete keel peab olema selge, lakooniline ja selgelt väljendama andmete vahelisi seoseid ning andmete ja otsitava vahelist seost.
5. Küsimus peab olema loogiliselt seotud ülesande sisuga ja vastama arvandmetele.

3.2. Automaatse lahendaja võimekus

Käesoleva töö raames loodud tekstülesannete automaatne lahendaja on võimeline lahendama nelja erineva ülesandepüstitusega tekstülesandeid. Kui andmeid sisaldavate lausete puhul toimivad kõik loodud püstitused sarnaselt, siis andmete kohta tegevust sisaldavast lausest tulenevalt määratakse ka programm lahendama kindlat ülesannet.

Programm on võimeline lahendama järgmiseid ülesandepüstitusi:

1. “Liitmine/lahutamine” - määratakse tegevuslauses, kui sisaldab kahte osalejat, arvulist kogust, elementi ning tegevussõna. Toimub mingi objekti üleviimine ühelt osaliselt teisele (näide: Mari **annab** Tarmole 6 eurot).
2. “Võrra (rohkem/vähem)” - määratakse tegevuslauses, kui sisaldab sõna “võrra” ja määrajat “rohkem” või “vähem”. Leitakse tundmatu objekti kogus liitmise ja lahutamise teel, kus vastus on sõltuv mõnest juba andmetes kirjeldatud osalise objekti kogusest (näide: Karmenil on ananasse 3 **võrra rohkem** kui Andresel).
3. “Korda (rohkem/vähem)” - määratakse tegevuslauses, kui sisaldub sõna “korda” ja määrajat “rohkem” või “vähem”. Leitakse tundmatu objekti kogus liitmise ja

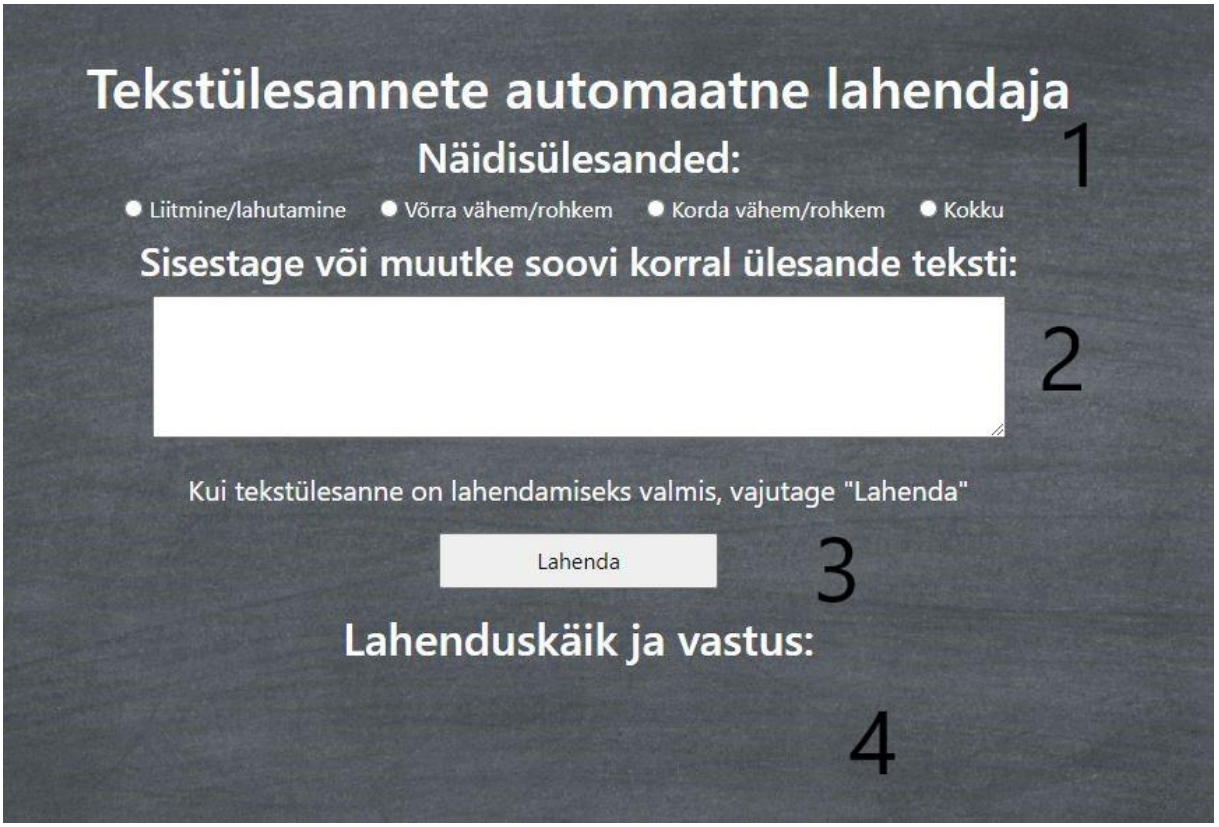
lahutamise teel, kus vastus on sõltuv mõnest juba andmetes kirjeldatud osalise objekti kogusest (näide: Mihklil on autosid 2 **korda rohkem**, kui Marial).

4. “Kokku” - määratakse küsimuslauses, kui küsimuses sisaldub sõna “kokku”. Leitakse kui palju on osalistel küsimuses esitatud objekte kokku (näide: Mitu vihikut on lastel **kokku**?)

Lahendaja arvutab lõppvastuse ainult korrektse sisendi korral. Korrektne sisend tähendab lahendaja jaoks minimaalselt kahte osalejat, kellest igaühele on määratud vähemalt üks positiivse kogusega nimisõnaline element, mille ees võib olla ka kirjeldav omadussõna. Lisaks peab olema selgelt tuvastatav küsimuslause ja (kõigil juhtudel peale “kokku” ülesande) ka tegevuslause.

3.3. Eesliides

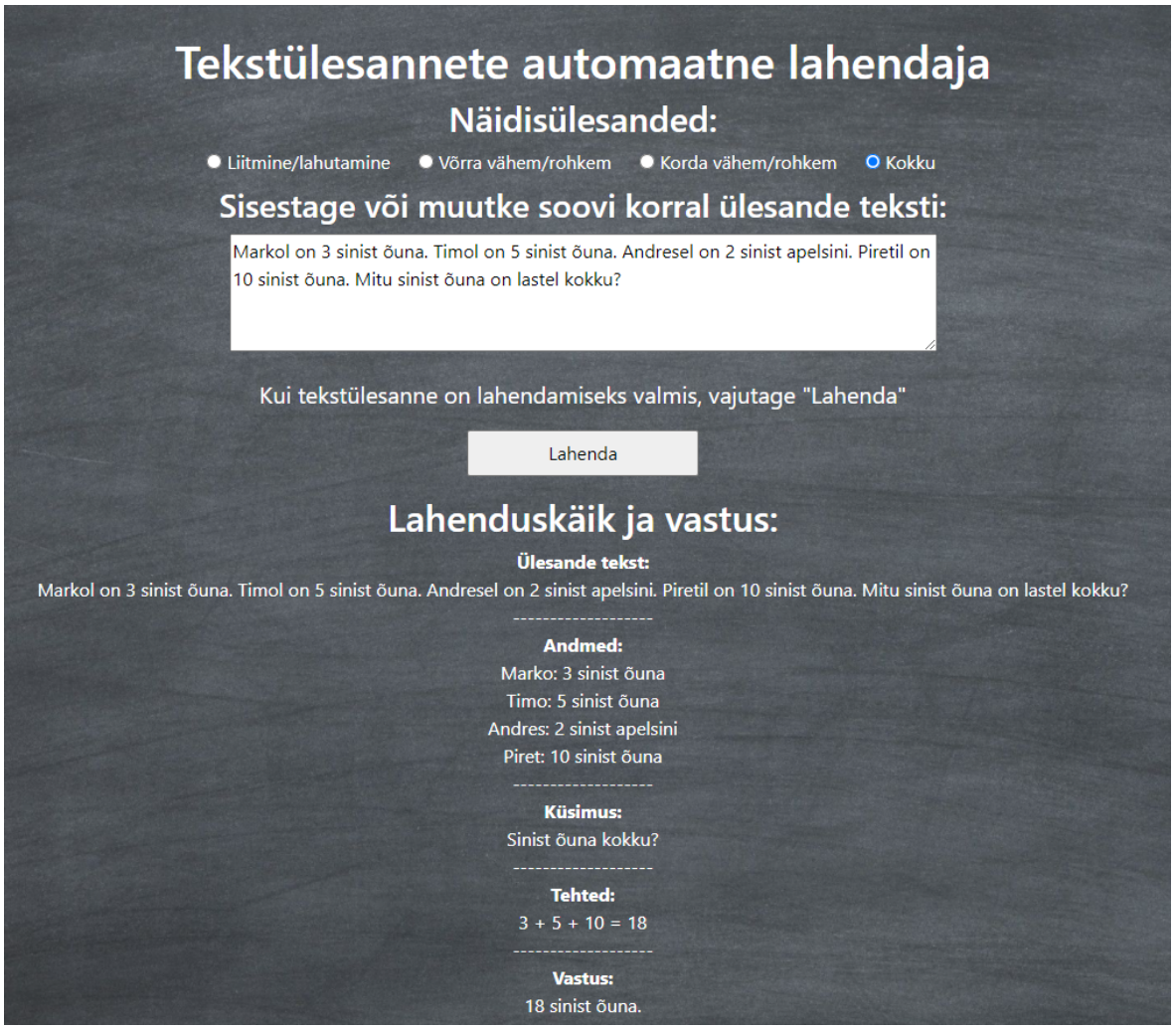
Programmi töö edukaks toimimiseks on vaja kahe olulise poole koostööd. Nendeks on eesning tagaliides. Eesliidese ülesanne on saada kasutajalt sisend ning käsk töö alustamiseks. Seejärel alustab tööd tagaliides, mis töötleb sisendi sobivusel kasutaja poolt ette antud tekstülesannet ning tagastab lahenduskäigu koos vastusega kasutajaliidesesse kasutajale tagasisidena.



Joonis 4. Programmi kasutajaliides: algvaade.

Kasutajaliidest on laias laastus võimalik jagada nelja alamkategooriasse (joonis 4), milleks on:

1. Lehekülje ülemises osas asuvad pealkiri ning nupud, mis võimaldavad kõigi programmi poolt lahendatavate etteantud näidisülesannete kuvamise (joonis 5);
2. Kasutajal ei ole kohustust valida näidisülesannet, vaid on võimalik ka kohe alustada tekstikasti täitmist, mille sisu kasutab programm sisendlausena.
3. Programm alustab tööd, kui on saanud korrektse sisendi ning kasutaja on vajutanud nuppu “Lahenda”. Sisendi mittesobivusel teavitatakse kasutajat puudustega (joonis 6).
4. Sisendi sobivusel alustab tagaliides tööd ning programm genereerib vastuse ja tagastab väljundi. Lahenduskäik ning vastus kuvatakse lehe allosas , kui programm on oma töö lõpetanud.



Joonis 5. Programmi kasutajaliides: “kokku” näidisülesande väljund.

Joonis 6. Programmi kasutajaliides: sobimatu sisendi veateade.

3.4. Tagaliides

Tagaliides on kirjutatud täielikult Pythoni programmeerimiskeeles. Pythoni valiku põhjuseks on asjaolu, et EstNLTK on üles ehitatud ja mõeldud toimima ainult Pythonis. Programmi töö on tekstülesannete lahendamisel jagunenud viieks suuremaks funktsiooniks.

3.4.1. Funktsioon “Käivita”

Programmi peafunktsioon, mida kutsub välja kasutajaliidese HTML-kood. Argumendina saab kasutajaliidesest kaasa kasutaja sisestatud ülesande algteksti. Funktsiooni käigus töödeldakse lause esmalt EstNLTK Text-klassi objektiks ning seejärel rakendatakse objektile “maltparser_syntax” kiht. See funktsioon kutsub oma töö käigus välja ka “lauseTöötleja” ning “Lahendaja” funktsioonid. “Käivita” funktsioon tagastab töö lõpetades kasutajaliidesele HTML-vormingus sobiva vastuse, mis väljastatakse kasutaja ekraanile.

3.4.2. Funktsioon “lauseTöötleja”

Funktsioon kutsutakse välja “Käivita” funktsioonis. Sisendiks saab funktsioon Text-klassi objekti koos “maltparser_syntax” kihiga. Väljundina tagastatakse andmesõnastik, sõnekujul andmelausete järjend, küsimuslause, otsitavate elementide nimistu, tegevuslause ning tehtemääraja. Seda funktsiooni võib pidada programmi kõige keerukamaks, kuid kõige

olulisemaks. Funktsiooni “lauseTöötleja” ülesanne on eraldada etteantud sisendist kõik vastuse leidmiseks vajalikud andmekandjad, mida kasutavad hiljem käivitatavad funktsioonid.

Funktsioon algab tsükliga, mis käib läbi kõik Text-klassi objekti “maltparser_syntax” kihis olevad töödeldud sõnad.

Küsimuslause tuvastatakse lause tagasiulatuval kontrollimisel, kui sõnade läbimise tsükkel jõuab kirjavahemärgini ja vaadeldav töödeldud lause sisaldab küsivat asesõna. Kui kirjavahemärgini jõudes küsivat asesõna tuvastatud ei ole, peetakse seda andmeid sisaldavaks lauseks ning töödeldud lause lisatakse koguandmete nimistusse.

Võtame näiteks lause “Timol on 6 kollast sidrunit.” Kui sõna on lause juursõna (ainuke sõna lauses, millel ei ole vanem-sõna ning millel on alluvad sõnad) ja sõna tüüp on tuvastatud inimese nimena (“**Timo**”), siis määratakse see sõna osaleja nimeks. Mõne Eesti nime eripära tõttu võidakse ka nimi tuvastada nimisõnana (näiteks Mari, Kalju jms).

Osaleja nime tuvastamise järel vaadatakse läbi nimele alluvad sõnad. Kui alluvate hulgas leidub omadussõna, mille sõnatüüp on alluv nimisõna, siis eeldab programm, et tegemist on osalejale kuuluva elemendiga (“**sidrunit**”). Taaskord vaadatakse läbi nimisõnale alluvad sõnad, kuhu kuuluvad omadussõnad (kui neid antud lauses on, näitelause “**kollast**”). Elementide nimed salvestatakse kahekordselt - nii tekstis leitava vormiga kui ka algvormina. Algvorme kasutab hiljem funktsioon “Tehe”. Samuti allub lauses elemendile numbriline väärtus (“**6**”), mis määratakse elemendi koguseks.

Pärast Text-klassi objekti läbimist eraldatakse koguandmetesse lisatud lausetest osalejate nimed, elementide nimed ning kogused ja luuakse nende põhjal kahekordne sõnastiku tüüpi muutuja, millesse salvestatakse kõik mainitud andmed. Esimese taseme võti on osaleja nimi “**Timo**” ning teise taseme võti elemendi nimi (“**kollane sidrun**” või “**kollast sidrunit**”).

Programm käitub iga ülesandetüübi tuvastamise korral erinevalt. Kui lauses tuvastatakse vaadeldava sõna algvormina sõna “kokku”, siis teab programm, et tegevuslauses ei ole vaja ühtegi osalejat. Kõigi teiste ülesandetüüpide juures vajab tehe alati kahte osalejat (“**Mati** võtab **Merjelt**”, “**Indrekul** on x võrra rohkem y-sid kui **Lauril**”). Kahe osalejaga tegevuslausetes otsitakse “võrra” või “korda” märksõnu, mille järgselt määratakse ülesandele vastav tehtemääraja. Kui kumbagi märksõna ei leidu, eeldab programm, et tegemist on tavalise kahe osaleja vahelise liitmise või lahutamisega.

3.4.3. Funktsioon “Lahendaja”

Funktsioon, mille ülesanne on genereerida kasutajale tagastatav väljund. Argumentidena saab kaasa andmesõnastiku, lausest eraldatud andmete järjendi, otsitavate elementide järjendi, tegevuslausete järjendi, alglause sõnena ning tehtemääraja. Funktsiooni lisab väljundisse ülesande alglause, kõik andmed reakaupa ning küsimuslause. Peale seda kutsutakse välja “Tehe” funktsioon ning selle tulemusena lisatakse väljundisse kõik tehted. Viimase sammuna lisatakse “vastuseLeidjast” tuvastatud vastus algsele küsimusele. Kui “Tehe” või “vastuseLeidja” tagastasid vea, siis lõpetatakse programmi töö ning teavitatakse kasutajat vigasest sisendist.

3.4.4. Funktsioon “Tehe”

Funktsioon kutsutakse välja “Lahendaja” funktsiooni sees. Argumentidena saab väljakutsumisel kaasa otsitavate elementide nimistu, tegevuslaused ning andmesõnastiku. Väljundiks on muudetud kujul andmesõnastik ning kõiki tehteid sisaldav sõne, mis lisatakse hiljem programmi väljundisse. Funktsiooni töö käigus eraldatakse tegevuslausest osalejad ning vajalikud teostatavad protsessid:

- “Kokku” arvutamise puhul leitakse kogu andmesõnastikus olevate otsitavate elementide summa ning lisatakse see vastusena andmesõnastikku
- “Võrra”/”korda” tehte puhul tuvastatakse tegevuslausest otsitav osaleja ning vajalik arvväärtus, mille võrra või korda on tarvis väärtust muuta.
- Kahe osaleja vahelist liitmist ning lahutamist eeldava ülesande puhul otsitakse tegevuslausest välja sobiv tegusõna, millele määratakse tegevussõnastikust (tabel 2) õige tehete järjekord.

Tabel 2. Tegevussõnastik tehete järjekorra määramiseks.

Positiivsed tehted (alaleütlevas käändes (Norman viskas **Kellile)** - esimeselt osalejalt lahutatakse, teisele liidetakse)	Negatiivsed tehted (alaltütlevas käändes (Marve ostis **Raunolt)** - esimesele liidetakse, teiselt lahutatakse)
andma	võtma
viskama	krahmama
veeretama	krabama
ulatama	eemaldama

annetama	nõudma
pakkuma	ostma
müüma	saama
loovutama	varastama
kaotama	laenama
kinkima	võitma
maksma	

3.4.5. Funktsioon “vastuseLeidja”

Kutsutakse välja “Lahendaja” funktsiooni lõppfaasis. Argumentidena saab kaasa tuvastatud küsimuse, andmesõnastiku ning tehtemääraja. Kõik vajalikud argumendid tekivad “lauseTöötleja” töö käigus. Andmesõnastikku muudab funktsioon “Tehe”, mis lisab sinna tekstülesande lõppvastuse. “vastuseLeidja” ülesanne on eraldada küsimusest otsitav tehtes osaleja ning otsitav element ja tagastada arvuline väärtus.

4. Võimalused rakenduse arendamiseks

Seni pole autori andmete kohaselt loodud ühtegi sarnast tekstülesannete lahendamise võimekusega rakendust, mis kasutaks tuumikuna EstNLTK-d. Selles tulenevalt leidub ka erinevaid viise, kuidas käesoleva töö raames valminud programmi täiendada.

4.1. Ülesannete keerukuse tõstmine

Ühe võimalusena programmi edasi arendada võiks olla ülesannete keerukuse astme tõstmine. Praegu on programm võimeline edukalt lahendama ühte algandmetes esitatud küsimust korraga, aga on võimalus arendada võimekus lahendada liittekstülesannet, milles esitatakse lahendajale ühe sisendiga mitu küsimust järjestikku.

Lisaks on võimalik programmi keerukust tõsta muutes tegevuslausetes leitava elemendi arvu leidmise omakorda tehteks - näiteks "Matil on 3 korda rohkem eurosid kui Sandril" asemel "Matil on $\sqrt{4^3}$ korda rohkem eurosid kui Sandril". Programmi keerukust on ka võimalik tõsta laiendades programmi võtmesõnastikku.

4.2. Ülesannete tüüpide lisamine

Praegu on lahendaja mõeldud lahendama aritmeetilisi tekstülesandeid. Programmi oleks võimalik edasi arendada viisil, et lahendatavad oleksid ka teist tüüpi, näiteks trigonomeetrilised või algebralised, tekstülesanded. Kuna tekstülesandeid lahendavad matemaatika tundides igas vanuses õpilased 1. kuni 12. klassini, siis on ülesannete potentsiaalne valim väga suur ning potentsiaalne keerukus väga kõrge.

Kokkuvõte

Käesoleva lõputöö eesmärk oli luua tekstülesannete automaatne lahendaja veebirakendus, mis on võimeline kasutaja poolt vabalt sisestatud tekstülesannet töötlema ning tagastama kasutajale nii lahenduskäigu kui ka lõppvastuse. Veebirakenduse tagaliides toetub oma töös suuresti EstNLTK keeletöötluse teekide kogumile ning Pythoni programmeerimiskeelele. Eesliides on loodud kasutades HTML'i, CSS'i, Flaski ning jQueryt. Lõputöö on üles laetud avaliku veebirakendusena ning on nähtav aadressil https://www.vallap.eu/lahendaja.

Kuigi tekstülesannete lahendajaid on loodud ka varem, siis kõik senised lahendused on võimelised töötlema ainult ühte ettemääratud ülesannet korraga. Ka seni ainukene tekstülesannete lahendaja Tekster töötles ainult väga range raamistikuga sisendfaili. Käesolev lõputöö erineb teistest alternatiividest suuresti tänu võimele lahendada tekstülesandeid, mida on võimalik kasutajal vabalt sisestada.

Rakenduse testimise käigus oli tagaliides võimeline korrektselt tuvastama vigase sisendi ning kuvama kasutajale sellekohase veateate. Sisendi muutsid vigaseks puuduvad osalejad, puuduvad andmed, puuduv küsimuslause või vigane tegevuslause. Testides ei suudetud tekitada olukorda, kus vigase sisendiga oleks saadud kasutajale ootamatult mingi vastus. Rakendus on võimeline lahendama kõik rakenduse jaoks korrektselt sõnastatud ja rakenduse skoopi jäävad ülesanded.

Rakenduse peamiste võimalike edasiarendustena näeb autor võimalust teha muutujate väärtuste leidmise protsess omakorda tehteks või laiendada lahendatavate ülesannete baasi, mida saab teha lisades praegustele aritmeetilistele tekstülesannetele ka trigonomeetrilisi või algebralisi tekstülesandeid.

Kasutatud kirjandus

1. E. Koppel. Tekstülesannete lahendamise programmi baasvariant. Bakalaureusetöö. 2006.
2. J. Kibal. Tekstülesannete lahendamise programmi funktsionaalsuse täiendamine. Bakalaureusetöö. 2013.
3. K. Valdson. Tekstülesannete lahendamise programmi täiendamine. Bakalaureusetöö. 2014.
4. Word problem solver, Symbolab. https://www.symbolab.com/solver/word-problems-calculator
5. About us, Mathcelebrity. https://www.mathcelebrity.com/one-second-math.php
6. Word problems calculators, Mathcelebrity. https://www.mathcelebrity.com/lesplan.php?key=wordprob
7. Eesti Keeletehnoloogia Riiklik Programm. https://www.keeletehnoloogia.ee/
8. NLTK. https://www.nltk.org/
9. EstNLTK. https://estnltk.github.io/
10. Süntaksliidese väljundi seletus, Eesti Keeleressursside Keskus. https://korpused.keeleressursid.ee/syntaks/dokumendid/syntaksiliides_ee.pdf
11. Basics of EstNLTK 1.6 / 1.7. https://github.com/estnltk/estnltk/blob/4acdb689eaf51b68d22fd5085362e2e45bac4577/estnltk/tutorials/basics_of_estnltk.ipynb
12. 11 Most In-Demand Programming Languages in 2022. Berkeley Boot Camps. https://bootcamp.berkeley.edu/blog/most-in-demand-programming-languages/
13. Flask. https://flask.palletsprojects.com/en/2.1.x/
14. Tekstülesannete lahendamine, Merike Rand (MA). https://merikerand.com/14-tekstulesannete-lahendamine/

Lisad

I. Programm

1. Eesliides (Python, Flask, HTML, CSS, jQuery)
2. Tagaliides (Python)
3. Veebirakendus aadressil: https://www.vallap.eu/lahendaja

II. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

Mina, Priit Vallap,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose

Tekstülesannete automaatne lahendaja,

mille juhendaja on Sven Aller,

reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada digitaalarhiivi DSpace kuni autoriõiguse kehtivuse lõppemiseni.

2. Annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi DSpace kaudu Creative Commonsi litsentsiga CC BY NC ND 3.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni.

3. Olen teadlik, et punktides 1 ja 2 nimetatud õigused jäävad alles ka autorile.

4. Kinnitan, et lihtlitsentsi andmisega ei riku ma teiste isikute intellektuaalomandi ega isikuandmete kaitse õigusaktidest tulenevaid õigusi.

Priit Vallap
10.05.2022