Eesti Teadusagentuur
Estonian Research Council

TeaMe

Euroopa Liit
Euroopa Sotsiaalfond

Eesti tuleviku heaks

Rakenduste loomise ja programmeerimise alused

Õpetajaraamat

Kersti Antoi
Jüri Vilipõld

Rakenduste loomise ja programmeerimise alused

valikkursus gümnaasiumitele
õpetajaraamat

Kersti Antoi, Jüri Vilipõld

2013

Kujundanud Viivi Jokk
TTÜ informaatikainstituut

See teos on litsentseeritud Creative Commonsi Autorile viitamine + Mitteäriline eesmärk + Jagamine samadel tingimustel 3.0 Eesti litsentsiga.

Sisu

Sissejuhatus ... 7
Valikkursuse kirjeldus ... 8
Informaatika põhivaldkonnad ... 8
Olukord meil ja mujal ... 8
Programmeerimese õpetamise keeled ... 10
Kursuse eesmärgid ja omandatavad pädevused/õpitulemused ... 11
Õppeprotsessi kirjeldused ... 12
Soovitused valikkursuse osade ja teemade käsitlemisel ... 12
Õppesisu ja põhimõisted ... 14
Õppe alustamise ning kursuse läbimise tingimused ... 15
Soovitusi füüsilise õpikeskkonna laiendamiseks ... 15
Valikuvõimalused kursuse läbimiseks ... 15
Kasutatavad vahendid ja materjalid ... 16
Soovitused kursuse ülesehituseks ... 17
Harjutused ja ülesanded ... 17
Teooriamaterjalid ... 18
Slaidiesitlused ... 18
Õpitulemuste kontroll ... 19
Õpilaste iseseisev töö ... 20
Iseseisva töö ülesande ja hindamismudeli näidis ... 21
Kursuse lõimimine teiste õppeainetega ... 22
Infotehnoloogiaained ... 22
Teised õppeained ... 22
Tunnikavad ... 23
Tunnikavad. Scratch ... 24
Sissejuhatus kursusesse ... 25
Tutvumine Scratchiga ... 26
Scratch. Liikumisplokid ... 27
Scratch. Välimuseplokid ... 28
Projekt „Kass, pall ja papagoi“ 1 ... 29
Projekt „Kass, pall ja papagoi“ 2 ... 30
Scratch. Avaldised ... 31
Scratch. Pliiats ja jälg ... 32
Scratch. Protsesside juhtimine ... 33
Scratch. Muutuja skoop ... 34
Rehmaatika ... 35
Scratch. Arvavaldised ... 36
Arvu arvamine ... 37
Labürint ... 38
Scratch. Tekstavaldised ... 39
Scratch. Loendi kasutamine ... 40
Vestlus ... 41
Sõnastik ... 42
Algoritme loenditega ... 43
Tunnikavad. VBA ... 44
MS Excel ... 45
VBA. Sissejuhatus ... 46

VB protseduurid ... 47
VBA. Arvavaldised ... 48
Exceli objektid *Range* ja *Shape* ... 49
Animatsioon Auto1 ... 50
Animatsioon Hüpe ... 51
Animatsioon Pallid kastis ... 52
Animatsioon Lennuk ja UFO ... 54
VBA. Tekstiavaldised ... 56
VBA. Lahtriploki omadused ... 57
VBA. Lahtriplokk parameetrina ... 59
Rehmaatika ... 61
Sündmusprotseduurid ... 62
Tunnikavad. Python ... 63
Pythoni kasutajaliides. Avaldised, muutujad ... 64
Pythoni laused. Moodulid ... 65
Python. Lihtsa skripti koostamine ... 66
Python. Funktsioonide loomine ... 67
Python. Kordused ... 68
Python. Rehmaatika ... 69
Python. Tekstiavaldised ... 70
Python. Algoritme loenditega ... 71
Python. Sõnastik ... 72
Python. Edetabel ... 73
Python. Ristkülik ja ring ... 74
Python. Funktsiooni parameetrid ... 75
Python. Funktsioonide uurimine ... 76
Näidistööplaanid ... 77

Sissejuhatus

Valikkursuse „Rakenduste loomise ja programmeerimise alused“ puhul on tegemist uue, võiks isegi öelda, eksperimentaalse kursusega, millele täpset analoogi ei ole. Sageli on taolised kursused orienteeritud programmeerimiskeele õpetamisele, aga mitte programmeerimise õpetamisele selle mõiste laiemas tähenduses. Tegelikult peaks õpetus hõlmama rakenduste loomise kõike faase: ülesande püstitust, analüüsi, disaini ja realisatsiooni. Viimases faasis on tegemist peamiselt programmeerimisega, aga ka mitte ainult. Näiteks üheks oluliseks tegevuseks realisatsioonifaasis on kasutajaliidese loomine.

Väga tähtis koht on rakenduste loomisel ja programmeerimisel objekt-orienteeritud modelleerimisel, tulevikus ka agent-orienteeritud modelleerimisel. Objektorienteeritud läheemisviisi olemust ja põhimõisteid on käsitletud kursuse üldosa jaotistes: objektorienteeritud modelleerimine, algoritmimine, rakenduste loomise alused. Neid põhimõtteid on arvestatud ka Scratchi, VBA ja Pythoni materjalides ning tüüptunnikavades ja vastavates töölehtedes. Kuidas ja millisel määral kasutatakse kursuse materjalides pakutavaid põhimõtteid ja lähenemisviise, jääb õpetaja otsustada. Esitatud näiteid, demosid, harjutusi ja ülesandeid saavad õpetajad laiendada ja edasi arendada.

Valikkursuse „Rakenduste loomise ja programmeerimise alused“ mahuks on käesoleval ajal 35 tundi. Olgu märgitud, et see maht ei ole piisav arvestades arengutendentse ja võrreldes sellega, millist suurt tähelepanu osutatakse antud probleemile teiste riikide koolisüsteemides.

Kursuse läbinud õpilane omandab sellised pädevused ja õpitulemused nagu nt oskus kirjeldada rakenduse loomise põhifaase: ülesande püstitus, analüüsi, disaini ja realisatsioon; teab programmide loomise, töötlemise, täitmise, ja silumise ning testimise põhimõtteid jms (vt „omandatavad pädevused/õpitulemused“, lk 11). Rakenduste loomise põhimõtteid ja -faase tutvustatakse praktilise töö kaudu, koostades mõned projektid, mis sisaldavad modelleerimise, analüüsi ning disaini elemente. Modelleerimises on realiseeritavate programmide jaoks esitatud algoritme unifitseeritud modelleerimiskeele UML abil. Tegevusdiagramme tuleks osata lugeda ja koostada. Klassimudelite olemust võib tutvustada võrdlemisi lühidalt ja üldiselt, arvestades, et õppija oskaks lugeda nende abil esitatud süsteemide ja andmete struktuure ning saaks neist aru. Põhiosa ajast kulub programmeerimise ning algoritmimise aluste omandamisele praktilise töö kaudu. Kursuse teemasid ei pruugi läbida järjestikku, vaid sobiva käsitluse valib õpetaja, arvestades kasutatavat programmeerimisvahendit, kursuse suunitlust, oma metoodilisi kogemusi ja eelistusi jmt.
Valikkursuse „Rakenduste loomise ja programmeerimise alused“ alustamiseks spetsiaalseid eelteadmisi ei nõuta. Kuid oleks väga kasulik, kui oleks läbitud sisukas informaatikakursus, mis arvestaks ECDL (*European Computer Driving Licence*) nõudeid ja soovitusi digitaalse kirjaoskuse osas. Kursuses omandatud teadmisi ja oskusi saab kasutada ainevaldkonna „Loodusained” kõikides põhi- ja valikainetes ning matemaatikas. Kursus on kõige otsesemalt seotud karjäärivalikuga, tutvustades informaatika ja infotehnoloogia erialasid.

Valikkursuse kirjeldus

Informaatika põhivaldkonnad

Kõigepealt üritaks täpsustada mõningaid vaadeldava teemaga seotud mõisteid. Selles vallas valitseb üsna suur mitmekesisus erinevates keeltes, riikides ja meil ka erinevates õppeasutustes. Näiteks seda, mida paljudes Euroopa riikides (ka Eestis) nimetatakse **informaatikaks** (*informatics*) nimetatakse USAs **arvutiteaduseks** (*computer science*). Viimase asemel kasutatakse inglise keeles sageli ka mõistet *computing*. Ka asja olemuse ja sisu tõlgendused on sageli üsna erinevad. Ei ole üheseid määratlusi näiteks sellistel mõistel nagu infotehnoloogia ja arendustöö jm. Allpooltoodus ei ole tegemist definitsioonidega või määratlustega vaid selgitustega, mis ehk aitavad järgnevat paremini mõista.

Õpetamise seisukohast võiks eristada informaatikas kolme tihedalt seotud valdkonda.

- **Programmjuhtimisega süsteemide** (**PJS**) **arhitektuur**. PJS: arvutid, arvutivõrgud, robotid, programmjuhtimisega seadmed ja tööpingid, telefonid jm. Arhitektuur: riist- ja süsteemitarkvara, töökorraldus, protokollid, failisüsteemid, andmete esitusviisid jmt.
- **Infotöö**. Informatsiooni kasutamine, loomine, säilitamine ja edastamine programmjuhtimisega süsteemide abil: lugemine, kirjutamine (dokumentide koostamine), arvutamine ja analüüs, joonistamine ja joonestamine, informatsiooni hankimine ja korrastamine, kommunikatsioon, juhtimine, õpetamine ja õppimine jm. Vastab ligikaudu mõistele **info**- ja **kommunikatsioonitehnoloogia** ehk IKT (*Information and Communication Technology - ICT*)
- **Arendustöö**. Rakenduste, infosüsteemide ja tarkvara loomine ning täiendamine ja muutmine: objektide struktuuri, andmete (omaduste) ja tegevuste modelleerimine, süsteemianalüüs, disain, algoritmimine ja programmeerimine (kodeerimine). Kasutatakse ka mõistet **arvutiteadus** (*Computer Science* või *Computing*)

Olukord meil ja mujal

Käesoleval ajal on kõik inimtegevuse valdkonnad, alates teadusest, majandusest ja tehnoloogiast ning lõpetades kunsti, spordi, olme ja meelelahutusega, väga tihedalt seotud infotehnoloogia vahendite kasutamisega. Võiks öelda, et käesoleva ajastu ja tuleviku infoühiskonna keskseteks mõisteteks on informatsiooni töötlemine ja programmjuhtimisega süsteemid. Viimaste hulka kuuluvad arvutid ja arvutivõrgud, programmjuhtimisega tehnoloogilised süsteemid ja seadmed, robotid ning sisseehitatud protsessoritega seadmed, alates autodest ja lõpetades külmkappide ja pesumasinatega. Üheks tähelepanuväärseks nähtuseks, mis toimub just praegu, on infotehnoloogia ja kommunikatsioonitehnoloogia ühtesulamine.

Kõik see eeldab selliste töötajate hulga kasvu, kes suudavad projekteerida ja luua erinevaid programmjuhtimisega seadmeid ning koostada nende jaoks tarkvara ja rakendusi. Üha enam kasvab vajadus spetsialistide järele, kelle põhitööks on arvutitarkvara loomine, haldamine ja hooldamine. Tänapäeval peavad aga ka paljud mitteinformaatika erialade spetsialistid mitte ainult oskama kasutada IKT vahendeid vaid suutma püstitada ülesandeid oma valdkonna probleemide lahendamiseks ja osalema vastavates projektides.

Eestis jätkub e-riigi teenustega seotud rakenduste loomine, enamikes firmades arendatakse infosüsteeme ja luuakse uusi projekteerimise, tootmise ja juhtimisega seotud rakendusi. Eestis hakkas tööle Euroopa IT agentuur ja NATO küberkaitse keskus. Kõik see näitab aine suurt tähtsust ühiskonna jaoks ning järjest suuremat vajadust inimeste järele, kes omavad infotehnoloogialaseid teadmisi ja oskusi. Sellepärast peab juba põhikoolist alates pöörama oluliselt suuremat tähelepanu arendustöö aluste õpetamisele.

Baasteadmisi programmjuhtimisega süsteemide arhitektuurist ja infotöö oskusi on tänapäeval vaja kõigil gümnaasiumi lõpetajatel tööle või õppima asumisel. Käesoleval ajal õpetatakse kutse-, kõrg- ja ülikoolides sageli mitmesuguseid informaatikaaineid, mis sisuliselt kujutavad endast nö tasandusõpet, sest paljude õppimaasujate oskused ei ole nimetatud valdkondades piisavad. Oleks otstarbekas alustada teatud korrastamist ja standardiseerimist juba koolide informaatikast, vähemalt gümnaasiumide lõpetajate jaoks: määratledes vajalikud valdkonnad ja ka nõuded tulemustele.

Standartiseerimise alused on toimiva arvutikasutaja üleeuroopalise ja rahvusvahelise oskustunnistuse süsteemi näol: ECDL – *The European Computer Driving Licence* (http://www.ecdl.ee/AO.html), täiesti olemas. Kahjuks meie koolides seda eriti ei rakendata. Seitse moodulit (Info- ja sidetehnoloogia (IST) mõisted, arvuti kasutamine ja failihaldus, tekstitöötlus, arvutustabelid, andmebaaside kasutamine, esitlus, veebisirvimine ja suhtlus) peaks vastama gümnaasiumi lõpetajate minimaalsete oskuste nõuetele. Infotöö aluste õpetamise korrastamine ja viimine vajalikule tasemele oleks oluliseks ja väga kasulikuks aluseks arendustöö, sh ka vaadeldava aine, õpetamisel.

Arvestades infotehnoloogia arengu ja rakendamise tendentse, muutub **arendustöö** üha aktuaalsemaks. Arendustöö (praktiliselt programmeerimise) õpetamine toimub praegu meil ainult mõnes koolis, kus on infotehnoloogia õppesuund. Gümnaasiumi lõpetajate arv, kes on koolis kokku puutunud arendustöö või programmeerimisega, üha väheneb. Seda on selgelt näha ülikoolidesse astujate seas. Arendustöö aluste tundmine aitab oluliselt kaasa ka infotöö oskuste täiustamisele.

Taolist tendentsi on täheldatud ka mujal maailmas. On jõutud järeldusele, et koolis ei piisa ainult infotehnoloogiavahendite kasutama õppimisest, vaid peab õppima ka nende tööpõhimõtteid ja loomise meetodeid. Antud probleemile pööratakse viimasel ajal eriti tõsist tähelepanu infotehnoloogia juhtriigis **USA**-s. Sealne (ja ka maailma) suurim arvutustehnika ja infotehnoloogiaalast tegevust koordineeriv organisatsioon: *Association for Computing Machinery* ehk **ACM** (http://www.acm.org), koos selle haldusalasse kuuluva arvutiõpetajate assotsiatsiooniga: *Computer Science Teachers Association* ehk **CSTA** (http://csta.acm.org), viisid hiljuti (2010) läbi põhjaliku uuringu arvututehnika ja infotehnoloogiaalaste ainete õpetamisest USA koolides. Tähelepanuväärse pealkirjaga: *„Running on Empty: the Failure to Teach K–12 Computer Science in the Digital Age“* (Tühijooks: …), kokkuvõttes on juhitud tähelepanu tõsistele puudustele selles valdkonnas ja esitatakse ettepanekud asja parandamiseks (vt link). Tuginedes nimetatud ja ka teistele uurimustele on CSTA koostanud rahvusliku standardi arvutiteaduste õpetamisest koolides: *„CSTA K-12 Computer Science Standards“* (vt link) ning tüüpõppekava selle valdkonna jaoks. Viimases on ette nähtud järgmised ained, mille kohta on toodud üsna detailsed kirjeldused (vt link):

- *Human Computer Interaction* (20 tundi – arvutite kasutamise üldised põhimõtted, riistvara jmt)
- *Problem Solving* (21 tundi – probleemide lahendamine arvutiga, andmed ja algoritmid)
- *Web Design* (25 tundi – veebidisain, html, JavaScript jm)
- *Introduction to Programming* (30 tundi – programmeerimine keeles Scratch, algoritmid)
- *Robotics* (35 tundi – robootika, Lego Mindstroms jmt)
- *Computing Applications* (30 tundi – rakenduste loomine erinevate arendusvahendite abil)

CSTAl on olemas spetsiaalne sait: Exploring Computer Science, kus on hulgaliselt õppematerjali ja muud infot nimetatud õppekava kohta (vt link).

Väga aktiivselt tegutsetakse viimasel ajal selles valdkonnas ka **Suurbritannias**. Londoni Kuningliku Seltsi (The Royal Society - sisuliselt Suurbritannia teaduste akadeemia) egiidi all läbi viidud põhjaliku uuringu alusel koostatud mahukas materjalis (2012) *„Shut down or restart? The way forward for computing in UK schools“* on toodud põhjalik analüüs olukorrast ja esitatud ettepanekud arengusuundade kohta (vt link). Selle alusel on spetsiaalne töögrupp (*Computing At School*) koostanud (2012) arvutiteaduse (*computer science*) õppekava kontseptsiooni (vt link). Õppekavas on määratletud näiteks ka nõuded arvutialaste teadmiste ja oskuste kohta erinevate kooliastmete jaoks alates 1. astmest (vanus 5-7 aastat). Ühtlasi on tehtud ettepanek muuta vastav õppeaine koolides kohustuslikuks.

Seoses koolide ja ülikoolide arvutiõpetuse probleemide aruteludega ning uute kontseptsioonide ja suundade määramisega, on USAs ja Suurbritannia haridussüsteemides võetud kasutusele mõiste ***Computational Thinking*** (*CT*) mis on kesksel kohal ka eespool nimetatud standardis ja kontseptsioonis ning paljudes teistes materjalides. Eesti keeles vastavat terminit veel ei ole. Võiks ehk öelda **infotehnoloogiline** või infoloogiline **mõtlemine** või lähenemisviis.

CT (infotehnoloogiline mõtlemise) põhiolemus seisneb probleemide formuleerimises sellisel kujul, et neid oleks võimalik efektiivselt lahendada infotehnoloogia vahendite abil. Sellega seonduvad järgmised mõisted ja tegevused: abstraktsioon ja modelleerimine, analüüs, rakenduste ja infosüsteemide disain, algoritmimine, programmeerimine jm. Seega võib öelda, et infotehnoloogiline lähenemisviis (CT) on kõige otsesemalt seotud arendustööga ja ka vaadeldava õppeainega.

Mõned lingid, kus on võimalik saada täiendavat informatsiooni: Wikipedia, CSTA, ISTE (*The International Society for Technology in Education*), Google, Carnegie Mellon University (CMU). Viimases on lausa olemas CT teaduskeskus, mille asutajaks ja juhiks on Jeannette Wing, kes võttis taaskasutusse (Wing2006, Wing2010) mõiste *Computational Thinking*. Esimesena kasutas seda mõistet Seymour Papert – programmeeriskeelekeele Logo üks loojatest (1967). Tema õpilase Mitchel Resnick juhtimisel on loodud programmeerimissüsteem Scratch ja ka tarkvara õppeotstarbelisele robotikomplektile LEGO Mindstorms.

Haridusspetsialistid arvavad, et CT alused peaks omandama kõik koolide lõpetajad. Kusjuures sellega tutvumine peaks algama võimalikult madalast astmest ning täiustuma ja süvenema järk-järgult.

Programmeerimese õpetamise keeled

Viimasel ajal on järjest tekkinud ning kiiresti arenenud ja levinud programmeerimiskeeled, mis on loodud spetsiaalselt programmeerimise õppimiseks/õpetamiseks: Alice, Scratch, BYOB, AgentSheets, Greenfoot, Kodu, MS Small Basic, Logo, StarLogo, MIT App Inventor jm. Neid nimetatakse ka hariduslikeks programmeerimiskeelteks (vt. Wikipeedia).

Õppeotstarbelisi programmeerimiskeeli on loodud ja kasutatud juba ammu. Esimeste seas olid näiteks BASIC (1964), Logo (1967), Pascal (1970). Neist kasvasid välja ka nö tööstuslikud dialektid. Näiteks Visual Basic (VB) ja VB .NET ning *Visual Basic for Application* (VBA) on enamkasutatavateks programmeerimiskeelteks rakendustarkvara ja dokumentidepõhiste rakenduste loomisel. Samal ajal säilitavad nad tänu väga lihtsalale süntaksile ja struktuursusele ka õpetamise keelte staatuse,.

Uue põlvkonna hariduslikud programmeerimiskeeled on enamikus graafilised ehk visuaalsed programmeerimiskeeled. Olgu märgitud, et sellised keeled nagu Visual C, Visual Basic jmt, on nö osaliselt visuaalsed: ainult kasutajaliidese loomine toimub visuaalselt, programmi sisu on tekstipõhine. Traditsiooniliselt nimetatakse neid uusi vahendeid programmeerimiskeelteks, kuid pigem võiks neid nimetada programmeerimise keskkondadeks, sest neis on palju enamat, kui lihtsalt keel. Taoliste keskkondade ja keelte kasutamisel on programmeerimise olemus muutumas. Rohkem on tegemist rakenduste modelleerimisega ja disainiga kui programmeerimisega traditsioonilises tähenduses.

Tegemist on suhteliselt lihtsa süntaksiga keeltega ning mugava kasutajaliidesega keskkondadega, kus programm pannakse kokku peamiselt hiirega. Taolistes süsteemides on osutatud suurt tähelepanu atraktiivsusele ja multimeedia (graafika, heli, videod) kasutamisele, võimalusele lihtsalt ja kiirelt luua mänge, animatsioone, koomikseid, taieseid, esitlusi jmt ning selle läbi omandada programmeerimise olemuse, põhimõisted ja meetodid märgatavalt kiiremini sellest, kui alustada kohe mingi "suure", proffidele mõeldud keeltega.

Kursuse eesmärgid ja omandatavad pädevused/õpitulemused

Kursuse „Rakenduste loomise ja programmeerimise alused“ mahuks on praegu planeeritud 35 tundi. Olgu märgitud, et see maht ei ole piisav arvestades arengu tendentse ja võrreldes sellega, millist suurt tähelepanu osutatakse antud probleemile teistes riikide koolisüsteemides. Valikkursusega taotletakse, et õpilane:

- arendab loovust, loogilist, analüütilist ja algoritmilist mõtlemist ning süsteemset käsitlusviisi probleeme ja ülesandeid lahendades
- teadvustab ja tunnetab programmjuhtimisega süsteemide tööpõhimõtet ning informatsiooni esitamise ja töötlemise põhiprotsesside olemust
- tunneb rakenduste ja programmide loomise vahendeid ning põhimeetodeid
- omandab programmide ja algoritmide koostamise ning probleemide lahendamise baasoskused programmjuhtimisega süsteemide abil
- tutvub objektorienteeritud modelleerimise, analüüsi ja disaini põhimõtetega
- saab aru objektide ja andmete olemusest, nende omadustest ning nendega täidetavatest tegevustest algoritmides ja programmides
- omandab algoritmimise ja programmeerimise põhikontseptsioonid ja mõisted ning oskused kavandada, koostada, siluda ja testida programme, mis koosnevad mitmest koostoimivast üksusest (protseduurist)
- kasutab objekte, skalaarandmeid ja massiive ning kirjeldab eri liiki protsesse

Valikkursuse läbinud õpilane:

- eristab ja oskab kirjeldada rakenduste loomise põhifaase: ülesande püstitus, analüüs, disain, realisatsioon
- teab ning oskab võrrelda ja hinnata rakenduste loomise erinevaid vahendeid ning meetodeid
- nimetab ja kirjeldab objektorienteeritud modelleerimise, analüüsi ja disaini põhimõisteid ning saab aru unifitseeritud modelleerimiskeeles (UML) esitatud klassi- ja tegevusdiagrammidest
- teab programmide loomise, töötlemise, täitmise, silumise ning testimise põhimõtteid
- eristab ning oskab programmides ja algoritmides kasutada eri liiki andmeid (arve, tekste, tõeväärtusi, graafika- ja heliandmeid) ning on omandanud ettekujutuse nende esitusviisidest arvutis; oskab neid kasutada Scratchi projektides ning VBA või Pythoni protseduurides
- teab ja mõistab konstantide, muutujate, massiivide ja objektide olemust ning kasutab neid otstarbekalt algoritmides ja programmides
- eristab ning oskab kasutada eri liiki operatsioone, avaldisi ja funktsioone väärtuste leidmiseks
- teab ja tunnetab omistamise operatsiooni olemust ning oskab seda kasutada algoritmides ja Scratchi ning VBA või Pythoni programmides
- oskab määrata tegevusi graafikaobjektidega ning tunneb ja oskab kasutada vahendeid ja meetodeid graafiliste kujundite joonestamiseks
- tunneb ning oskab algoritmides ja programmides kasutada protsesside juhtimise vahendeid ning kirjeldada eri liiki protsesse: järjestikused protsessid (jada), tsüklilised protsessid (kordused), hargnevad protsessid (valikud) ja paralleelprotsessid
- oskab programmides luua ja kasutada loendeid ja massiive ning kirjeldada tüüpalgoritme nendega: summade ja keskmiste arvutamine, ekstreemumite leidmine, otsimine ja sorteerimine
- oskab korraldada programmide jaotamist üksusteks (protseduurideks, funktsioonideks või skriptideks) ning korraldada nende vahel koostööd ja andmevahetust

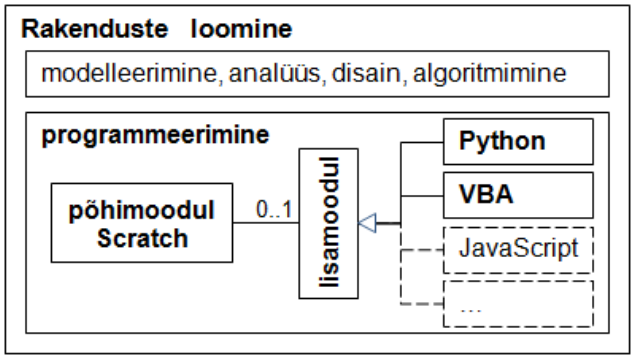
Õppeprotsessi kirjeldused

Valikkursuse „Rakenduste loomise ja programmeerimise alused“ põhiosad

1. rakenduste loomise põhimõtted
2. mudelid ja modelleerimise alused
3. algoritmimise ja programmeerimise alused

Soovitused valikkursuse osade ja teemade käsitlemisel

Kursuse struktuuri ja põhikomponente kajastab allolev joonis.



Kõrvalolev programmeerimise osa skeem demonstreerib modelleerimiskeele UML põhimõtteid süsteemide struktuuri kirjeldamisel. Skeem näitab, et programmeerimise osa koosneb põhimoodulist, milles kasutatakse Scratchi. Sellele võib (aga ei pea: kordus 0..1), järgneda lisamoodul, milles kasutakse Pythonit või VBAd. Tulevikus võivad lisanduda veel teised moodulid: JavaScript jm.

Moodulite orienteeruvad mahud: põhimoodul 15–35 t, lisamoodul 0–20 t. Lisamooduli keele ja moodulite mahud (kokku 35 t) valib kool arvestades õppijate õppesuunast. Lisamoodul võib ka puududa. Lisamooduli kasutamise üheks eesmärgiks on õppijale tekstipõhise programmeerimiskeele tutvustamine, sest Scratchi sarnased graafilised programmeerimiskeeled hakkavad alles levima.

Valikkursuse osasid ning teemasid ei läbita järjestikku, vaid sobiva käsitluse valib õpetaja, arvestades kasutatavat programmeerimisvahendit, kursuse suunitlust, oma metoodilisi kogemusi ja eelistusi jmt.

Rakenduste loomise meetodeid, vahendeid ja põhifaase (ülesande püstitus, analüüs, disain ja programmeerimine) vaadeldakse ülesannete lahendamisel kogu kursuse jooksul. Rakenduste loomise põhimõtteid ja põhifaase tutvustatakse praktilise töö kaudu, koostades projekte, mis sisaldavad modelleerimise, analüüsi ning disaini elemente. Erilist tähelepanu tuleks pöörata analüüsi olemusele ja selle seostamisele modelleerimisega. Õppevahendi osas „Rakenduste loomine“ on vaadeldud rakenduste loomise erinevate faaside olemust ja nende täitmiseks kasutatavaid vahendeid.

Läbivalt käsitletakse ka **modelleerimist** ja **algoritmimist**. Modelleerimise olemuse ja põhimõistetega oleks otstarbekas tutvuda kohe alguses, sest neid kasutatakse praktiliselt kõikides teistes materjalides. Modelleerimises tutvustatakse objektorienteeritud lähenemisviisi ja unifitseeritud modelleerimiskeelt UML. Objekte, klasse ning vastavaid diagramme käsitletakse lühidalt ja lihtsustatult. Õppija peaks oskama lugeda nende abil esitatud süsteemide ja andmete struktuuri. Oluline saavutada, et õpilased teadvustaks objektorienteeritud lähenemisviisi ja modelleerimise olemust ja teaksid põhimõisteid: objekt, objektide omadused ja meetodid. Põhjalikumalt vaadeldakse protsesside ja algoritmide esitamist tegevusdiagrammide ja algoritmikeele (pseudokoodi) abil. Seda tehakse koos programmide koostamisega.

Põhiosa ajast kulub kursuses **programmeerimise ning algoritmimise aluste** omandamisele praktilise töö kaudu. Kursuse eesmärkide saavutamiseks on tähtis teadvustada ning tunnetada programmide ja programmjuhtimise olemust, koostades praktilises töös programme ning realiseerides neid arvutil.

Põhimooduli programmeerimiskeele valimisel on arvestatud, et programmeerimise õppimisel ei ole primaarne programmeerimiskeel, vaid algoritmimine, modelleerimine ja disain. Seepärast peaks esimese keelena kasutama võimalikult lihtsa süntaksi ja struktuuriga programmeerimiskeelt, mis võimaldaks

keskenduda nimetatud tegevustele ning algoritmimise ja programmeerimise üldistele põhimõtetele ja kontseptsioonidele. On arvestatud keele levikut ja sarnasust teiste programmeerimiskeeltega. Põhimoodulis on programmeerimiskeeleks **Scratch**. Selles on pööratud suurt tähelepanu atraktiivsusele ning multimeedia kasutamisele: lihtsalt ja kiiresti on võimalik luua mänge, animatsioone, koomikseid, esitlusi jm. Samas on võimalik realiseerida ka arvutuslike, andmetöötluse ja arvutigraafika algoritme.

Scratchi kodusaidil Massachusettsi Tehnoloogia Instituudis (MIT) on suurel hulgal õppematerjale ning näiteid ja demosid. Scratchi asemel või lisaks sellele võib kasutada programmeerimissüsteemi BYOB (SNAP!), mis on loodud Berkeley Ülikoolis. BYOB on täielikult ühilduv Scratchiga ja omab mõningaid täiendavaid võimalusi. Scratch ja BYOB leiavad üha laiemat kasutamist koolides ja ülikoolides. USA arvutiõpetajate assotsiatsiooni (CSTA) poolt koostatud gümnaasiumite arvutiteaduse õppekavas aine "Sissejuhatus programmeerimisse" tugineb täielikult Scratchil (vt link). Scratch ja BYOB leiavad kasutamist arvutiõpetuse algkursustes ka mitmes maailma juhtivas ülikoolis (Harvard, Berkeley jt). Programmeerimiskeelte kasutamise ulatuse (populaarsuse) nn Tiobe indeksis on Scratch tavaliselt 30 esimese piirimail. Scratchi kasutajaliides, abiinfosüsteem ja kasutamisjuhend on olemas ka eesti keeles.

Lisamoodulites käsitletakse rakenduste loomist erinevate programmeerimiskeelte abil. Üldiselt ei peaks piirduma ainult ühe keelega vaid võiks tutvuda veel vähemalt ühe programmeerimiskeelega, valides lisamooduli. Tulevikus võiks lisamooduleid olla 4–5. Valikus võiksid olla mõned sellised keeled nagu: JavaScript, PHP, Visual Basic jm. Esialgu on materjalid kahe lisamooduli kohta: Pythoni ja VBA.

Python on vabavaraline üldotstarbeline objektorienteeritud, lihtne ja võimas programmeerimiskeel, seda saab kasutada nii lauarvutite kui ka veebirakenduste loomiseks. Keel on loodud 1991. aastal, tema populaarsus ja kasutamise ulatus (kuues-kaheksas koht Tiobe indeksis) on järsult kasvanud just viimastel aastatel. Pythonit kasutatakse paljudes koolides ja ülikoolides nö esimese programmeerimiskeelena. Üheks oluliseks Pythoni omaduseks on lihtsus. Lausete struktuur on selge ja kompaktne, puuduvad igasugused spetsiifilised eraldajad lausete struktuuri määramiseks. Ei ole väärtuste ja muutujate deklareerimist ning struktuurandmete jäiga ja fikseeritud struktuuri kirjeldamist, mis on iseloomulik enamikule programmeerimiskeeltele; andmetüüpide ja andmestruktuuride käsitlemine on lihtne ja dünaamiline.

VBA (*Visual Basic for Application*) kuulub paljudes koolides kasutatava MS Office juurde, pakkudes lihtsaid vahendeid rakenduste loomise, programmeerimise ja modelleerimise õppimiseks objektorienteeritud keskkonnas. VBA põhineb üldotstarbelisel programmeerimissüsteemil Visual Basic (**VB**). Viimane leiab laialdast kasutamist eriti rakendustarkvara ja veebirakenduste loomisel olles kasutusulatuselt enam-vähem samal tasemel nagu PHP ja Python.

VBA on üks enim kasutatav vahend **dokumendipõhiste rakenduste** loomisel ja arendamisel. VBA leiab laialdast kasutust Microsofti toodetes: Excel, Word, PowerPoint, Access, Visio jt. See on kasutusel ka mitmetes teiste firmade toodetes: Corel Office, Corel Draw, AutoCAD, Imagineer jm. VBAga lähedased arendusvahendid on ka süsteemides OpenOffice ja IBM SmartSuite ja paljudes teistes. Siin vaadeldakse VBA kasutamist MS Exceli keskkonnas. Eesmärgiks ei ole Exceli programmeerimine, vaid Scratchis omandatud programmeerimise oskuste ja teadmiste süvendamine ja laiendamine tekstipõhiste, objektorienteeritud programmeerimist toetavate vahendite abil. VBA ja Excel pakuvad selleks suurepäraseid võimalusi.

Õppesisu ja põhimõisted

Valikkursuse „Rakenduste loomise ja programmeerimise alused“ õppesisu ja põhimõisted on järgmised:

- Rakenduste loomise põhimõtted. Rakenduse olemus. Rakenduste loomise meetodid ja vahendid. Üldotstarbelised programmeerimissüsteemid ja -keeled. Rakendusprogrammid ja nende arendusvahendid.
- Rakenduste loomise põhifaasid: ülesande püstitus, analüüs, disain, realisatsioon. *Tunnis peaks tegema mõned projektid, kus detailsemalt käsitletakse rakenduste loomise põhifaase. Näiteid on jaotises „Rakendused“ ja ka teistes materjalides.*
- Objektorienteeritud modelleerimine ja modelleerimiskeel UML. Objektid ja klassid. Objektide omadused ja meetodid. Sündmused. Seosed objektide ja klasside vahel, klassidiagrammid. *Oluline on kohe alguses selgeks teha mõisted objekt, objektide omadused ja meetodid ning objektide klassid.*
- Tegevused ja tegevusdiagrammid. Tegevusdiagrammide ja pseudokoodi kasutamine algoritmide esitamiseks. *Võiks näidata erinevate protsesside kirjeldusi olemasolevate diagrammide abil.*
- Algoritmimise ja programmeerimise alused. Programmi olemus. Programmide loomise, töötlemise ja täitmise põhimõtted. Programmeerimiskeeled ja -süsteemid. Laused ning programmiüksused: protseduurid, funktsioonid ja skriptid. Translaatorid: interpretaatorid ja kompilaatorid.
- Algoritmi olemus. Algoritmide esitusviisid: tegevusskeemid, pseudokoodid ja algoritmikeeled, programmeerimiskeeled jms. *Siin võiks toimida järk-järgult. Alguses on otstarbekas tuua koos algoritmid (UML diagrammid ja/või kirjeldus pseudokoodis) ja vastavad Scratchi skriptid, et näidata ülemineku põhimõtteid. Edasi praktiseerida skriptide loomist etteantud algoritmide alusel ning siis ka algoritmide iseseisvat koostamist*.
- Objektide ja andmete käsitlemine programmides. Objektide omaduste, meetodite ja sündmuste kasutamine. Andmete liigid ja organisatsioon. Konstandid ja muutujad. Massiivid. Operatsioonid andmetega. Avaldised ja funktsioonid. Omistamine. Andmete lugemine väliskeskkonnast ja kirjutamine väliskeskkonda.
- Graafikaandmete kasutamine programmides. Graafikaobjektide importimine. Graafiliste kujundite loomine (joonistamine) programmi poolt. Graafikaobjektide põhiomadused ja nende muutmine. Tüüptegevused graafikaobjektidega. Animatsioon.
- Protsesside liigid algoritmides ja programmides: järjestikused protsessid (jada), tsüklilised protsessid (kordus), hargnevad protsessid (valik), paralleelprotsessid. Eri liiki protsesside kirjeldamise ja täitmise põhimõtted. Korduste liigid: lõpmatu kordus, etteantud kordamisarvuga kordus, eel- ja järelkontrolliga iteratiivsed kordused. Valik ühest, kahendvalik ja mitmene valik.
- Massiivid. Massiivide põhiomadused: nimi, dimensioon, mõõtmed. Massiivide määratlemine ja loomine. Viitamine massiivide elementidele. Tüüpalgoritmid massiividega: summade ja keskmiste arvutamine, ekstreemumite leidmine, otsimine, sorteerimine jmt.
- Koostöö ja andmevahetus programmiüksuste vahel. Pöördumised ja teadete saatmine. Globaalsed ja lokaalsed andmed. Kasutajaliidese loomine. Programmide silumine ja testimine. Programmide vormistamine.

Kursuse materjalides on kirjeldatud mitmeid algoritme, mis on realiseeritud erinevate vahenditega (Scratch, VBA, Python). See on metoodiliselt vajalik mõistmiseks, et iga ülesande juures on oluline analüüs, modelleerimine, algoritmimine ja realiseerimine.

Õppe alustamise ning kursuse läbimise tingimused

Valikkursuse „Rakenduste loomise ja programmeerimise alused“ alustamiseks spetsiaalseid eelteadmisi ei nõuta. Kuid oleks väga kasulik, kui oleks läbitud sisukas informaatikakursus, mis arvestaks ECDL nõudeid ja soovitusi digitaalse kirjaoskuse osas.

Kursuse läbimiseks peab õpilane omandama ainekavas ettenähtud teadmised ja oskused, mis võivad olla täpsustatud õpetaja poolt. Kursuse läbimiseks peab õpilane täitma kursust läbiviiva õpetaja poolt määratud harjutused, iseseisvad tööd, testid ja kontrolltööd.

Eesmärgiks seatud õpitulemuste saavutamiseks sobivad kõige paremini aktiivõppe meetodid. Kõikidel õpilastel peaks arvutiklassis olema kasutada arvuti ja õpetaja poolt esitatu ja ette näidatu saab kohe praktilises töös järele proovida.

Soovitusi füüsilise õpikeskkonna laiendamiseks

Valikkursus „Rakenduste loomise ja programmeerimise alused“ on täielikult seotud ITK vahendite kasutamisega. Selleks peab õppetöös kasutatavatesse arvutitesse olema installeeritud vastav programmeerimise keskkond, milleks on alguses Scratch, hiljem VBA või Python.

Õppetöö läbiviimiseks võiks olla kasutusel vastav õpikeskkond, milleks on sobiv e-õppe keskkond Moodle.

Valikkursust võib läbi viia ka väljaspool kooli vastavalt kooli ja koolituskeskuste vahelistele kontaktidele.

Valikuvõimalused kursuse läbimiseks

Valikkursus „Rakenduste loomise ja programmeerimise alused“ koosneb üldjuhul kahest osast: põhimoodulist ja lisamoodulist.

Põhimooduli programmeerimise osa on orienteeritud programmeerimissüsteemi Scratch kasutamisele.

Lisamoodulis võib kooli valikul praegu olla kasutusel kas arendussüsteem Visual Basic Application (VBA) MS Exceli baasil või programmeerimiskeel Python.

Moodulite maht ja teise programmeerimiskeele valik sõltuvad õppesuunast, õppurite õpimotivatsioonist ja eelteadmistest ning õpetaja soovist ja kogemustest.

Näiteks:

- kasutatakse ainult Scratchi
- põhjalikumalt käsitletakse Scratchi ning tutvustatakse kas VBAd, Pythonit või mõlemat
- Scratchi vaadeldakse lühidalt, põhjalikumalt käsitletakse Pythonit või VBAd.

Lisamooduli valikul võiks otsustada VBA kasuks, kui varasemates informaatika/arvutiõpetuse kursustes on põhjalikumalt käsitletud tabeltöötlust MS Exceli abil. Tuttav keskkond võimaldab kiiremini aru saada VBA protseduuride kasutamisest ja kasutajaliidese loomisest töölehel. Proovida võiks realiseerida eelnevates kursustes õpitud keerulisemaid Exceli valemeid VB funktsioonidena ja tegevusi lahtriplokkidega, näitamaks arendusvahendist tulenevat kasu Exceli rakenduse loomisel. Sarnaselt Scratchile on VBA puhul andmed, programm ja kasutajaliides kõik ühes failis (dokumendipõhine rakendus).

Python on (erinevalt VBst) vabavara (http://www.python.org) ja leiab viimasel ajal koolides ja ülikoolides laialdaselt kasutamist esimese programmeerimiskeelena. Selle kasutamine eeldab õppurilt rohkem teadmisi failisüsteemist. Töö toimub suures osas käsurea vahendusel, graafilise kasutajaliidese loomine on keerulisem (selle kursuse materjalides ei käsitleta). Pythoni õpetamisel võiks kindlasti näidata andmete lugemist tekstifailist ja/või tulemuste kirjutamist tekstifaili.

Erandjuhul võib lisamoodul ka puududa.

Käesoleval ajal viivad erinevaid programmeerimise kursuseid läbi ka mitmed ülikoolid ja koolituskeskused. Kokkuleppel kooliga (õpetajaga) võiks mõni taoline kursus minna arvesse ka lisamoodulina.

Tulevikus võiks valikuvõimaluseks olla valikkursuse või selle osa (lisamoodul) läbimine e-õppena.

Kasutatavad vahendid ja materjalid

Materjalid, juhendid, näidisülesanded ja demod (http://rlpa.ttu.ee) on esitatud ka Moodle keskkonnas kursuses „Rakenduste loomise ja programmeerimise alused (õppekomplekt)“ vt. https://moodle.e-ope.ee/course/view.php?id=3859 registreerumise võti: **opetaja**, külaliskasutaja salasõna: **Tiiger**.

Kõikide harjutuste juurde kuuluvad õpilase töölehed, millel olevad juhendid, vihjed, tegevusskeemid ja lisaülesanded võimaldavad varieerida ülesannete täitmise järjekorda ja aega. Kursuse materjalides Moodles on ka töölehtede lähtetekstid, et harjutused oleksid vastavalt vajadusele kohandatavad õpetaja poolt.

Soovitused kursuse ülesehituseks

Pikaajaline praktika samalaadsete kursuste läbiviimisel Tallinna Tehnikaülikoolis on näidanud, et õppematerjali omandamisel ja õppeprotsessi mõistmisel on oluline õppida praktilise töö käigus. Lihtsates näiterakendustes, mida koostatakse koos õpetajaga, tekitatakse kõigepealt arusaamine õpitavast mõistest või tegevusest (mis see on, kus ja milleks saab kasutada), hiljem tutvutakse teooriamaterjalides antud definitsioonide ja erinevate kasutamisvõimalustega kas õpetaja esituses või iseseisvalt. Teadmised ja oskused kinnitatakse testide ja iseseisvalt tehtavate tööde abil.

Iga algus on raske. Kui valida rakenduste loomise tutvustamisel esimeseks keeleks keerulise süntaksiga tekstipõhine programmeerimiskeel, on õppuri jaoks uusi mõisteid ja omandatavaid töövõtteid alustamisel liiga palju: harjumatu ülesandepüstitus, rakenduse analüüs ja kirjeldamine, objektorienteeritud lähenemine, uus töökeskkond, programmeerimiskeele süntaks ja semantika, vigade avastamine ja parandamine.

Scratchi kasutamise eesmärk selles kursuses ei ole keeruliste ja atraktiivsete animatsioonide ja mängude loomine, vaid programmeerimise põhikontseptsioonidest ülevaate andmine. Scratch on lihtne ja intuitiivselt omandatav, süntaksiprobleeme praktiliselt ei esine. Seepärast ongi just see vahend valitud rakenduse olemusest, objektorienteeritud lähenemisest, muutujate kasutamisest ja juhtimislausetest kiire ülevaate andmiseks. Graafilised käsuplokid annavad hästi meeldejääva pildi liitlausete (valik, kordus) kasutamisest ja tekitavad seega hea ülevaate tegevuste struktuurist. Lisamooduli arendusvahendi saab kasutusele võtta niipea, kui õppuritel on algteadmised ja -oskused omandatud.

Lisamooduli (Python või VBA) harjutusülesannetes on eeldatud, et õppur juba oskab programmeerida (Scratchis). Esimesed harjutusülesanded on teise keele süntaksist ja erisustest ning editori võimalustest ülevaate saamiseks. Kasutatud on kas väga lihtsaid või varem Scratchis tehtud rakendusi, mille algoritmid on varasemast tuttavad.

Põhi- ja lisamoodul ei pea olema üksteisest rangelt eraldatud. Lisamooduli esimesi harjutusi võiks kasutada juba enne põhimooduliga (Scratchiga) lõpetamist. Näiteks võiks teise valitud programmeerimiskeele süntaksit näidata ja mõned Scratchis tehtud tuttavad ülesanded selles realiseerida juba enne loendite tutvustamist. Nii on lihtsam kahe vahendi võimalusi võrrelda. Edaspidi saab siis demonstreerida töötavat algoritmi Scratchis ja lasta õppuritel iseseisvalt teha samalaadne rakendus teises keeles.

Harjutused ja ülesanded

Kursuse raames tehtavad harjutused (tehakse õpetajat jäljendades või tema juhendamisel) ja ülesanded, mida õppur teeb iseseisvalt, võib jaotada kahte gruppi:

Keskkonna tundmaõppimine:

- tehnilised oskused (käsuplokkide kasutamine vs töö programmieditoriga, tegevuste käivitamine, kasutajaliides)
- lihtsate tegevuste realiseerimine mõistmaks keskkonna võimalusi andmete ja objektidega manipuleerimisel ning keele süntaksi ja semantika omandamiseks
- pooliku rakenduse täiendamine või vigade parandamine; eesmärgiks võõrast programmist arusaamine ning vigade avastamine ja programmi silumine

Rakenduse loomine:

- rakenduse loomine etteantud ülesande (kirjelduse) alusel; ette võib olla antud sõnaline kirjeldus, analüüsi ja disaini elemente (tegevusdiagramm jm)
- rakenduse koostamine etteantud tehnilistel tingimustel; näiteks peab rakenduses, mille sisu õppur ise valib, olema kasutusel eelkontrolliga kordus ja kaheosaline valik
- suurem omaloominguline rakendus kohustuslike elementidega, hindamine hindamismudeli abil.

Kursuse jaoks on koostatud näidistunnikavad ja harjutused, mida peaks olema mõnevõrra rohkem, kui aine maht (35 tundi) kasutada võimaldab. Seega on õpetajal võimalus valida konkreetsele õpperühmale sobivaid harjutusi, mida kasutada tunnis või anda iseseisvaks kodutööks. Harjutustele kuluv aeg on märgitud ligikaudsena, tavaliselt sõltub see rühma suurusest, nende eelteadmistest ja motivatsioonist. Vajadusel jõuab harjutusi sooritada ka märgitust lühema ajaga ja seega kombineerida ühes õppetunnis mitu erinevat harjutust ja töölehte. Mõne õpperühmaga võib aega kuluda ka oluliselt kauem, sel juhul tuleks suurendada iseseisva töö osa.

Teooriamaterjalid

Õpiku alguses käsitletakse rakenduste loomise, arendustöö ja programmeerimise üldküsimusi, mis ei ole otseselt seotud konkreetse programmeerimiskeelega. Materjalid on esitatud kolme teemana: objekt-orienteeritud modelleerimine, rakenduste loomise põhimõtted ja algoritmimine.

Programmeerimise aluste tutvustamiseks vaadeldakse kolme erinevat tüüpi keelt (programmeerimis-süsteemi): graafiline programmeerimiskeel Scratch, tekstipõhine üldotstarbeline keel Python ja dokumendipõhine arendussüsteem VBA.

Õpiku materjal ei ole mõeldud järjestikuseks lugemiseks. Esiteks jäävad loomulikult ära mittekasutavate programmeerimiskeelte osad. Teiseks võib pöördumine üldosade (modelleerimine, algoritmimine, rakenduste loomine) poole toimuda korduvalt erinevates kohtades, erinevatel eesmärkidel ja erineva süvenemisastmega. Kohe alguses oleks otstarbekas tutvuda modelleerimise olemuse ja põhimõistetega (objektid, klassid, omadused ja meetodid), sest neid kasutatakse praktiliselt kõikides teistes materjalides. Edaspidi võib vastavate mõistetega erinevate keelte kasutamisel kokku puutudes tulla uuesti sama teema juurde tagasi.

Väga olulist rolli mängivad kursuses praktikumid (probleemorienteeritud ehk projektipõhine lähenemisviis). Paljudel juhtumitel tutvutakse uute mõistetega praktikumides ning vastavalt vajadusele pöördutakse teoreetiliste osade poole.

Erinevate keelte käsitlemisel võib eristada vähemalt kahte taset: üldine ja detailsem.

Võimaldamaks õppijal saada kiiresti ülevaade Scratchist, on esimesel tasemel toodud lühivaade Scratchi kasutamisest rakenduste loomiseks. Sellega on soovitatav tutvuda kohe alguses, paralleelselt esimeste harjutustega. Antud osaga on linkidega seotud lisamaterjalid (teine tase), kust saab täiendavat informatsiooni vastava teema kohta. Lisamaterjalides esinevad lingid üldosadele. Lisamaterjalidele võib olla viiteid ka praktikumide töölehtedelt.

VBA ja Pythoni käsitlemisel eeldatakse, et õppija juba oskab teatud määral programmeerimist ja tunneb selle põhimõisteid, olles eelnevalt töötanud Scratchiga. Sellepärast algab vastav osa nn „kiirtutvusega“. Tuginedes Scratchis varem esinenud näidetele antakse mitteformaalselt lühiülevaade programmeerimise põhimeetodite ja vahendite realiseerimisest antud keeles: objektid, muutujad, avaldised, laused, protsesside juhtimine (jada, kordus, valik), programmi põhiüksused ja koostöö nende vahel, programmide sisestamine, redigeerimine ja silumine. Edaspidi käsitletakse olulisemaid teemasid detailsemalt ja formaalsemalt.

Slaidiesitlused

Moodle kursuses on iga mooduli juures toodud slaidiesitlused olulisemate mõistete, skeemide ja ekraani-kuvadega. Slaidiesitluste kasutamine tunnis uue materjali seletamisel on tõenäoliselt ebavajalik, olulisem on töövahendite ja tegevuste demonstreerimine. Vajadusel saab õpetaja näidata nendest üksikuid teemakohaseid slaide või koostada lühemaid esitlusi.

Õpitulemuste kontroll

- (enese)testid
- pooliku rakenduse täiendamine
- rakenduse koostamine etteantud kirjelduse alusel
- rakenduse koostamine kindlatel tehnilistel tingimustel (vaba sisu)
- suurem omaloominguline rakendus (vaba sisu, hindamismudel)

Testid

Testidega on võimalik eelkõige kontrollida mõistete teadmist ning ka tegevustest arusaamist. Moodle kursuses on näiteteste toodud iga mooduli juures.

Arvutis täidetavad ja automaatselt kontrollitavad testid vajavad arvutitöökohta (tarkvara, võrguühendus jm) ja keskkonda tulemuste automaatseks registreerimiseks. Esimest korda täidetakse test arvutiklassis, selle alusel saab määrata hinde. Hiljem võib õppur testi täitmist korrata õpetaja määratud tingimustel (ka kodus). Testi tulemus võib (aga ei pruugi) mõjutada kursuse lõpphinnet.

Testi kasutamise eelisteks on kiire ja lihtne kasutamine ja automaatne kontroll. Teadmiste kinnistamiseks võib sama testi teha korduvalt, nõuda võib 80-100% tulemust.

Testide puudused:

- saab kontrollida vaid baasteadmisi
- valikvastuste puhul võib õige tabada ka juhuslikult
- heade testide koostamine on väga raske ja mahukas töö

Poolvalmis rakenduse täiendamine

Töö toimub arvutiklassis või iseseisvalt. Poolvalmis rakendus lühendab töö aega. Võimalik on frontaalne meetod: kõik teevad ühesugust tööd; täiendatavad tööd (või ainult täiendused) võivad olla ka veidi erinevad. Valmis tööd võivad kontrollida ka teised õppurid (või võrrelda oma tööd teiste omaga).

Keerulisemad ülesanded sobivad hästi rühmatööks (2-3 õpilast). Rühmatöö puhul võiks lisaülesandeks olla ka tehtud täienduste või leitud vigade loetelu (kirjelduse) esitamine.

Näiteid:

poolvalmis rakendus Scratcis Õhtu linnas täiendamiseks etteantud juhendi järgi.

VBA rakendus vead3.xlsm vigade avastamiseks ja parandamiseks.

Rakenduse koostamine etteantud tehnilistel tingimustel

Sobib kontrollimiseks, kas õpitud objektidest ja tegevustest on aru saadud. Selline lähenemine on õppuri jaoks jõukohasem; kindlasti on omavalitud teemal rakenduse tegemine õppuri jaoks huvitavam.

Sellisena püstitatud mahukam ülesanne võib vajada hindamismudelit, milles kirjeldatakse hea tulemuse ja miinimumtulemuse (võimalikud on ka vahepealsed astmed) nõuded.

Õpilaste iseseisev töö

Kursuses peaks iseseisev töö omama väga olulist rolli, sest rakenduste loomise oskused ja kogemused saab omandada just iseseisvalt ülesandeid lahendades ning programme koostades.

Iseseisvad tööd võib jagada järgmiselt:

- jooksvad, suhteliselt väikesemahulised ülesanded, mida lahendatakse praktiliselt igal nädalal; siia kuulub ka klassis alustatud ülesannete lõpetamine iseseisva tööna
- mahukamad ülesanded („projektikesed“), milles võib esineda kahte liiki ülesandeid: õpetaja poolt etteantud ülesanded ning õpilase poolt vabalt valitud projektid
- veerandi (semestri või terve kursuse) ülesanne, mille kaitsmise (esitluse) alusel saadakse põhiosa veerandi (semestri, kursuse) hindest, võiks olla omaloominguline rakendus (võib ka ette anda teema), mille juures tuleks kasutada kõiki seniomandatud oskusi (ette tuleb anda nõuete loetelu).

Omaloomingulise rakenduse loomisel tuleks õpilastele soovitada kasutada teiste ainete (matemaatika, füüsika, ajalugu, võõrkeel) ülesandeid või teemasid simulatsioonide koostamiseks.

Näiteid:

http://scratch.mit.edu/projects/reinik/1788857

http://www.scribd.com/doc/109600635/iTEC-3-tsukkel-mangu-loomine

Iseseisvalt tehtud tööde tutvustamiseks võiks töid esitleda klassi või väiksema rühma ees. Töö hindamisel saab sel juhul arvestada ka kaasõppurite arvamust.

Omaloominguliste rakenduste juures saab kasutada paaristööd (töö suuremates gruppides pole sellise algkursuse väikesemahuliste rakenduste puhul tõenäoliselt otstarbekas), tulemuse esitlemisel saab näiteks üks õppur kirjeldada rakenduse objekte ja tegevusi ning teine demonstreerida seda ekraanil.

Juhtimaks õppurite tähelepanu olulistele elementidele rakenduses, tuleks iseseisvate tööde kohta koostada ja koos ülesandega kättesaadavaks teha hindamismudel, mis võimaldaks õppuritel ka ise hinnata oma teadmisi ja oskusi ja annaks parema ülevaate õppeprotsessist.

Suurema rakenduse puhul peaks õppur kindlasti koostama ka kasutusjuhendi ning andmete, objektide ja tegevuste (protseduuride) kirjelduse – spetsifikatsiooni. Need nõuded peaks kajastuma ka hindamismudelis.

Õpilaste iseseisvalt tehtud omaloominguliste projektikeste kontrollimine (ja hindamine) õpetaja poolt võiks toimuda kahes etapis:

- pärast projekti esimest esitamist annab õpetaja teada, millised on vead ja puudused ning annab soovitusi parandamiseks ja täiendamiseks;
- täiendatud projektike kas esitatakse uuesti või esitletakse õpetajale ja kaasõppuritele.

Hinnata saab nii esimest kui teist etappi ja ka esitlust. Et õppuritel ei kaoks motivatsioon, ei ole rakenduse esimese variandi eraldi hindamine enamasti otstarbekas; kindlasti tuleks hiljem jälgida, kuidas on tehtud õpetaja poolt soovitatud parandused ja täiendused.

Töö saab hindamiseks anda teis(t)ele õpilas(t)ele; sel juhul peaks antud kommentaarid ja soovitused olema kättesaadavad ka õpetajale. Hinnang kaasõpilase tööle on iseseisev töö ning võib samuti olla hindamise objekt.

Lahendatud ülesanded (või mingi valiku nendest) võib õppur kursuse lõpuks koondada ühte kausta - õpimappi, mida saab kasutada õppeperioodi lõpphinde määramisel.

Iseseisva töö ülesande ja hindamismudeli näidis

Ülesanne

Koostada Scratchis omaloominguline rakendus – animatsioon või mäng. Teema ja sisu võib vabalt valida, kuid rakendus ei tohi kokku langeda või olla väga sarnane ühegi teise tööga või tehtud harjutusega. Hinnatav on teostus, idee uudsus ja töö vastavus nõuetele.

Nõuded rakendusele:

- rakendus peab sisu poolest kujutama **tervikut** (mäng, animatsioon, reaalelulise ülesande lahendamine)
- kasutusel peab olema vähemalt **3-4 aktiivset objekti (spraiti)**, **2-3 muutujat** ning vähemalt üks **loend**
- spraidid peavad **käituma erinevalt** (vähemalt **4-5** tegevust - skripti)
- skriptides peavad esinema **kordused** ja **valikud** ning **koostöö** nende vahel
- **kasutamisjuhend** ja **realiseerimise põhimõtted** peavad olema **projekti märkmetes**: **File**-menüü – **Projekti märkmed**. Kindlasti esitada **objektide** (spraitide) ja **muutujate** kirjeldus.

Tegevused:

1. Õpilastele tutvustatakse tingimusi, tähtaegu, hindamismudelit
2. Õpilased koostavad rakenduse (individuaalselt või grupitööna) ja esitavad selle tähtajaks
3. Õpetaja kontrollib ja annab tagasiside: puudused, soovitused täiendamiseks (kujundav hindamine)
4. Õpilased täiendavad tööd ning esitavad uuesti
5. Tööd võib esitleda, kaasõpilased annavad ka hinnangu
6. Õpetaja hindab tehtud tööd ja esitlust. Hindamisele võib kuuluda ka rakenduse esialgne kavand.

Hindamismudel

Hindamis-kriteeriumid	**5**	**3**	**1**
rakenduse sisu	huvitav ja uudne, hästi läbi mõeldud	sarnaneb varemtehtud rakendustega, aga sisaldab ka uudseid elemente	sarnaneb väga palju varemtehtud rakendustega
objektid	kasutusel on piisav hulk spraite ja muutujaid ning vähemalt üks loend	spraite ja muutujaid on nõutust vähem või puudub loend	spraite ja muutujaid on liiga vähe, puudub loendi kasutamine
tegevused	kasutusel mitmeid erinevaid tegevusi; tegevused on hästi läbi mõeldud	erinevaid tegevusi vähevõitu või lahendused ebaotstarbekad	erinevaid tegevusi liiga vähe
keerukus	kasutusel piisav hulk juhtimis-lauseid, skriptid on koostatud otstarbekalt	korduste ja valikute kasutamine ei ole piisavalt läbi mõeldud; skriptides esineb korduvaid osi, mida saaks koondada	juhtimislauseid on kasutatud liiga vähe või valesti ja ebaotstarbekalt
kasutusjuhend	juhend on põhjalik ja sõnastus korrektne; kirjeldatud on kõiki kasutaja tegevusvõimalusi ja rakenduse reaktsioone	juhend on lünklik, kõiki võimalusi ei ole nimetatud	juhend puudub
spetsifikatsioon	kõiki spraite, nende otstarvet ja tegevusi ning muutujaid on korralikult kirjeldatud	kirjelduses on olulisi puudusi	kirjeldus puudub
rakenduse esitlemine	rakendust esitletakse igakülgselt, põhjalikult ja korrektselt; vastatakse esitatud lisaküsimustele	esitlemine kohmakas, vaja on suunavaid küsimusi	rakendust pole esitletud

Kursuse lõimimine teiste õppeainetega

Kursusel on suur potentsiaal ja ka vajadus lõimimiseks teiste õppeainetega. Lõimimisel võiks eristada kahte õppeainete rühma:

- infotehnoloogiaained: informaatika, robootika, programmeerimist tutvustavad ained algkoolis, veebirakenduste loomine jm
- kõik teised õppeained: matemaatika, füüsika, bioloogia, ajalugu, keeled, kunstiõpetus jne

Infotehnoloogiaained

Ained peaks moodustama ühtse süsteemi, kus kasutatakse sarnast lähenemisviisi, metoodikat jm.

Antud ainele eelnevatest ainetest võiks eeskätt nimetada Informaatikat. Näiteks tabeltöötluse põhjalikum käsitlus – nimede kasutamine, valemid, funktsioonid (sh tingimuslikud, loogika- ja otsimisfunktsioonid) – aitaks kaasa infotöö oskuste parandamisele ja oleks ka oluliseks toeks programmeerimise õppimisel. Üheks kasulikuks vahendiks Informaatika õppimisel oleks modelleerimise ja objektipõhise lähenemisviisi (objektid, klassid, omadused, meetodid, sündmused) kasutamine erinevate programmide töö tutvustamisel (operatsioonisüsteemid, tekstitöötlus, tabeltöötlus, graafika- ja andmebaasiprogrammid).

Ainetes Robootika ja Arvuti kasutamine uurimistöös saaks otseselt ära kasutada antud aines omandatud programmeerimis- ja modelleerimisoskused.

Teised õppeained

Lõimimisel teiste ainetega võib eristada kahte suunda:

- teiste ainete ülesannete kasutamine antud aine praktikumides ja kodutöödes (arvutusülesanded, jooniste koostamine, simulatsioonid jm)
- antud aines omandatud modelleerimise, algoritmimise ja programmeerimise oskuste rakendamine mingi teise aine õppimisel.

Olgu märgitud, et vastavalt infoloogilise lähenemisviisi (CT) arendamise põhimõtetele on primaar-seks programmeerimise ja algoritmimise kasutamine teistes ainetes. See muidugi eeldab vastavate teadmiste ja oskuste olemasolu aineõpetajatel.

Antud kursuse materjalides on pööratud üsna suurt tähelepanu eeskätt matemaatikale, mida kasutatakse ka teistes ainetes. Mõned sellised näited on kirjeldatud õpikus ning tunnikavades ja töölehtedel, vt ka Moodle kursuses iga programmeerimiskeele osa juures olev kaust Lisa (näited ja ülesanded). Näidetes on vaadeldud geomeetriat, joonestamist, funktsioonide uurimist (graafikute tegemine, võrrandite lahendamine, integreerimine jm). On toodud ka mõned füüsikaülesannete demod – Autohüppe ja Visked (VBA).

Mitmeid näiteid (eriti Scratchis) saab kasutada prototüüpidena viktoriinide, testide jm loomiseks erinevates ainetes.

Tunnikavad

Järgnevalt on toodud **näidistunnikavad** valikkursuse „Rakenduste loomise ja programmeerimise alused“ õpetamiseks. Tunnikavad sisaldavad selgitusi ja viiteid kasutatavale materjalile, nõuanded õppeprotsessi korraldamise (sh teemade järgnevus) ning harjutuste ajalise jaotuse ja iseseisva töö kohta. Iga tunni kohta on välja toodud tunni eesmärgid/õpitulemused, õpitavad ja kordamisele tulevad mõisted ning uued mõisted ja oskused, mida selle tunni harjutustes küll kasutatakse, aga põhjalikumalt käsitletakse edaspidi.

Tunnikavadele vastavad töölehed, kus on selgitusi, soovitusi, tegevusskeeme või algoritmide kirjeldusi ja osaliselt ka lahendusi, asuvad Moodle kursuses.

Tegemist on soovitusliku materjaliga. Õpetaja saab neid kindla teema õpetamiseks kasutada või muuta vastavalt oma vajadustele. Teemade järjestuse ja mahu määramisel tuleb ilmselt arvestada õppesuunda, õppegruppi koosseisu, õppurite õpimotivatsiooni ja eelteadmisi, tundide jaotust (üksikud, paaristunnid või pikemad õppesessioonid) ja õpetaja isiklikke eelistusi.

Tundideks vajalikud materjalid, vahendid, tarkvara:

- õpetaja arvuti
- videoprojektor
- vastav õpikeskkond või veebileht
- igal õpilasel arvuti (internetiühendusega)
- tarkvara juhendmaterjali (pdf või tekstifail) lugemiseks.

Põhimooduli ülesannete jaoks Scratch (vabavara, http://scratch.mit.edu)
VBA ülesannete jaoks MS Excel.
Pythoni ülesannete jaoks Python 3.x.x (vabavara, http://www.python.org)
Töölehed võib õppuritele kättesaadavaks teha elektroonsel kujul (pdf, html) või väljaprindituna.

Märkus: seoses sellega, et erinevates süsteemides võivad kasutusel olla erinevad kooditabelid, on töölehtedel nagu ka õpikus olevates programminäidetes sageli välditud täpitähtede kasutamist protseduuride ja muutujate nimedes.

Tunnikavad. Scratch

Scratchi kasutamise eesmärk selles kursuses ei ole programmeerimiskeele Scratch õppimine ega ka keeruliste ja atraktiivsete animatsioonide ning mängude loomine, vaid programmeerimise põhikontseptsioonidest ülevaate andmine.

Scratch on lihtne vahend rakenduste loomise ning programmeerimise ja algoritmimise aluste tutvustamiseks. Scratchi abil on võimalik kiirelt omandada kõik peamised programmeerimise, algoritmimise, modelleerimise ja rakenduste loomise põhimõtted ja tüüptegevused:

- eriliiki andmete (märk-, graafika-, heli- ja videoandmed) olemus ja kasutamise iseärasused
- operatsioonid andmetega ja avaldised
- andmete sisestamine ja väljastamine
- joonistamise põhimõtted
- muutujad ja loendid
- protsesside liigid ja nende kirjeldamine: jada, kordus, valik ja paralleelsus
- programmide jagamine üksusteks ning koostöö ja andmevahetuse korraldamine nende vahel
- rakenduste kasutajaliideste loomise põhimõtted
- objektorienteeritud lähenemisviisi ja modelleerimise olemus.

Scratchi õppematerjalis ja näidisülesannetes on minimaalselt tutvustatud liikumise ja välimuse plokke ning heliandmeid. Põhirõhk on juhtimiskäskudel (valikud, kordused, skriptide käivitamine) ning muutujate ja avaldiste kasutamisel.

Harjutusülesannetes tuleks algusest peale hakata arvestama lisamooduli programmeerimiskeele võimalustega ja valida selliseid näiteid ja harjutusülesandeid, mida oleks mugav ka teises keeles realiseerida ning edasi arendada.

Sissejuhatus kursusesse

Kestus: 45 min

Tunni eesmärgid: kursuse sisu ja materjalide tutvustamine

Õpioskused:

- õpilane on tuttav kursuse sisu ja korraldusega
- õpilane teab mõisteid rakendus, algoritm, programm
- õpilane teab rakenduse loomise etappe

Õpitavad teemad: Rakenduste põhimõisted, rakenduste loomise etapid

Mõisted, mida käsitletakse edaspidi: objektorienteeritud lähenemine

Demod: Pong.sb, kits_kunt_kapsas.sb, liitlahuta.sb

Tunni käik:

- sissejuhatus kursusesse, kursuse läbimise nõuded
- registreerumine kursusele, materjalidega tutvumine,
- esitlus „Rakenduste …“ slaid Kursuse sisu
- näide **Pong** (lihtne rakendus – mõned objektid, paar tegevust): tutvustamine
- õpilased võiks kirjeldada rakenduses nähtud tegevusi samm-sammult
- esitlus „Rakenduste …“ slaid Rakenduste loomise faasid
- näite **kits_hunt_kapsas** tutvustamine
- esitlus „Rakenduste …“ slaid Objektid
- näide **kits_hunt_kapsas**: objektid
- näide **liitlahuta**: tutvustamine
- Scratchi näited, ülevaade

Märkus:

Oluline on kursuse korralduse ja materjalide tutvustamine. Kõiki olulisi mõisteid saab uuesti ja uuesti selgitada ülesannete lahendamise käigus järgnevates tundides.

Tutvumine Scratchiga

Kestus: 45 min

Tunni eesmärgid: tutvustada Scratchi võimalusi

Õpioskused:

- õpilane teab objektorienteeritud lähenemise põhimõisteid
- õpilane tunneb Scratchi põhivõimalusi ja oskab kasutada kasutajaliidest
- õpilane oskab luua spraite ja rakendada neile mõningaid tegevusi

Õpitavad teemad: Objektorienteeritud lähenemine, objekt, omadus, meetod; Scratchi sprait, skript

Mõisted, mida käsitletakse edaspidi: UML klassidiagramm; korduslause; muutuja, juhuarvud

Tunniks vajalikud materjalid:

- tööleht **Kraps123**
- demo **kraps123.sb** (kraps123_s.sb)
- graafikafailid **tiiger.gif** ja **hunt.gif**

Tunni käik:

- Scratch ja esitlus „Rakenduste …“: põhimõisted (slaid 23, ka 24–27)
- Scratch ja Kasutusjuhend „Kasutajaliides“: liidese kirjeldamine (sprait, spraidi omadused: nimi, suund, kostüüm, asukoht (x,y), helid; lava, taust, koordinaadid laval).
- Käsuplokkide tutvustamine (liikumine, kostüümivahetus, kordused, paus, skripti käivitamine)
- Harjutuse **Kraps123** (tööleht ja demo kraps123.sb) tutvustamine.
- Lihtsa skripti koostamine ja käivitamine: kassike Kraps (anda spraidile nimi, määrata pööramise stiil) jalutab laval edasi-tagasi; lisada sammu juhuslik muutumine.
- Kui jõuab ja õpilastel huvi on, näidata ka muutuja loomist, selle väärtuse määramist liuguri abil ja kasutamist sammu pikkuse muutmiseks.
- Uute spraitide loomine: näidata joonistusredaktorit ja vaadata üle kostüümide kaustad.
- Luua uus sprait (ballerina-a), tekib üks kostüüm, lisada ka teised kostüümid ning koostada skript kostüümide vahetamiseks.
- Skripte saab käivitada/peatada hiireklõpsuga. Mitme skripti üheaegseks käivitamiseks valida sobiv päiseplokk. Kõik skriptid peatab punane nupp lava ülaservas.
- Luua uued spraidid valides kostüümid animeeritud gifist (failid tiiger.gif ja hunt.gif tuleb eelnevalt salvestada klassi arvutitesse või võrgukettale).
- Muuta (peegeldada) hundi kõiki kostüüme, et ta ei liiguks tagurpidi.

Scratch. Liikumisplokid

Kestus: 45 min

Tunni eesmärgid: Scratchi liikumiskäskudega tutvumine

Õpioskused:

- õpilane oskab nimetada objekti (spraidi) omadusi
- õpilane oskab koostada lihtsamaid skripte liikumisplokkidest

Kordamisele tulevad mõisted: Scratchi kasutajaliides; sprait, kostüümid, skriptid

Õpitavad teemad: Suund, osuta, pööre. Liikumine, juhuslikkus

Mõisted, mida käsitletakse edaspidi: Kordused, valikud; andurid; muutuja, avaldis

Tunniks vajalikud materjalid: tööleht **Veemaailm**, demo **veemaailm.sb** (veemaailm_s.sb)

Tunni käik:

- Eelmise harjutuse materjali meenutamine: kassike Kraps jalutab laval edasi-tagasi.
- Asendades Krapsu kostüümid krabi omadega (crab1-a ja crab1-b), muutes liikumise aeglasemaks (väiksem samm ja pikem paus) ning määrates lava taustaks *underwater*, tekib Veemaailm.
- Harjutuse **Veemaailm** (tööleht ja demo veemaailm.sb) tutvustamine.
- Scratch ja Kasutusjuhend „Liikumine“: käsuplokkide tutvustamine.
- Luua sprait Kalake, tehes koopia Krabist ning muutes nime, kostüümi ja skripti. *Kopeerimine on oluline oskus, mis võimaldab edaspidi muuta töö spraitide ja käsuplokkidega oluliselt kiiremaks!*
- Luua sprait Hai, kes jälitab kalakest.
- Rakenduse loomise ajal on testimiseks mõistlik skripte käivitada/peatada hiireklõpsuga. Mitme skripti üheaegseks käivitamiseks valida sobivad päiseplokid.
- Hai võiks kalakest puudutades midagi ütelda (valikuplokid ja andurid), teavituse *kala käes* võib lisada ka hiljem. Kui on oht, et aega napib, võib selle tegevuse esialgu ka välja jätta.
- Luua sprait Kala (kopeerida ja muuta spraiti Hai), kes järgneb hiirekursorile.
- Luua sprait Kala2. [liigu antud aja jooksul etteantud kohta] on ebamugav käsk*: liikumise ajal ei saa selles skriptis midagi kontrollida (puude teise spraidiga vm).*
- Luua sprait Kala3 (teeb ringe).
- Sprait Merikarp. Skript käivitub tühikuklahviga. Ülevaade päiseplokkidest
- Sprait Kaheksajalg. Skript reageerib Hai skriptis tekitatud teatele *kala käes*.
- Sprait Vaal. NB! Spraidi suund peab olema -90, määratakse nö „käsitsi“.
- Luua sprait Meritäht ja muutuja S (et on ainuke muutuja, võib olla ka globaalne).
- Lisaülesanded

Märkus:

45 min tund võib selle harjutuse põhjaliku käsitlemise jaoks olla liiga lühike, sest alguses on palju uusi mõisteid ja tehnilisi selgitusi ja õpilased ei ole veel Scratchi spraitide ja skriptide loomisel kuigi vilunud. Lahenduseks võiks olla:

- varuda harjutuse jaoks rohkem aega (2 x 45min)
- õpetaja selgitab tunnis lühidalt harjutuse eesmärke ja sisu, demonstreerib valmis skripte ja nende toimimist, näitab ette mõned töövõtted ning seejärel tegelevad õpilased ülesandega iseseisvalt ja lõpetavad selle vajadusel kodus.
- õpilased teevad selle harjutuse töölehe abil kodus iseseisvalt, tunnis vaadatakse toimivad rakendused koos üle ning õpetaja juhib tähelepanu olulisemale ja jagab tehnilisi selgitusi; ka lisaülesanded võiks teha klassis.

Scratch. Välimuseplokid

Kestus: 45 min

Tunni eesmärgid: Scratchi välimusplokkide tutvustamine

Õpioskused:

- õpilane teab spraidi omadusi ja võimalikke graafikaefekte;
- õpilane oskab koostada lihtsamaid skripte liikumise- ja välimusplokkidest;
- õpilane teab tegevuste käivitamisvõimalusi.

Kordamisele tulevad mõisted: Scratchi kasutajaliides; sprait, kostüümid, skriptid

Õpitavad teemad: Spraidi omadused: suurus, värv, keere jm graafikaefektid, nähtavus. Skriptide käivitamise võimalused.

Mõisted, mida käsitletakse edaspidi: Kordused, valikud; teated, juhuarvud

Tunniks vajalikud materjalid:

- töölehed **efektid** ja **nimi**
- demod **efektid.sb** (efektid_s.sb) ja **nimi.sb** (nimi_s.sb)

Tunni käik:

- Harjutuse **efektid** (tööleht ja demo) tutvustamine.
- Õpetaja annab ülevaate Scratchi välimusplokkidest.
- Õpilased koostavad iseseisvalt skriptikesi graafikefektide proovimiseks.
- Skriptide käivitamiseks kasutada erinevaid võimalusi (päiseplokid).
- Juhuslike arvude genereerimine, nende kasutamine varemkoostatud skriptides.
- Rakenduse koostamine töölehel toodud näite alusel.
- Harjutuse **nimi** (tööleht ja demo) tutvustamine.
- Skriptide käivitamisvõimaluste kordamine. Oluline: skript võib käivituda, kui toimub mingi sündmus (reaktsioonina teatele, mille väljastab mingi teine skript); plokkide [teavita] ja [teavita … ja oota] erinevus.
- Õpilased koostavad rakenduse iseseisvalt kasutades võimalikult palju varemõpitud liikumis- ja välimus-plokke.
- Eraldi tuleks rõhutada, et projekti märkustesse tuleks kirjutada, millele ja kuidas tähed peaks reageerima.

Märkus:

Harjutus **nimi** sobib esimeseks arvestuslikuks (hindeliseks) ülesandeks. Uute teadmiste lisandudes (muutujad, loendid jm) võib lasta õpilastel seda muuta ja täiendada ning uuesti esitada.

Projekt „Kass, pall ja papagoi“ 1

Kestus: 45 min

Tunni eesmärgid: lihtsa tervikliku rakenduse loomine kavandamisest realiseerimiseni (algus)

Õpioskused:

- õpilane oskab luua spraite
- õpilane oskab koostada ja kasutada skripte
- õpilane oskab nimetada sündmusi (puude, serval jm) ning kasutada neid skriptide koostamisel.
- õpilane teab rakenduse loomise faase ja nende järjestust

Kordamisele tulevad mõisted: Objekti liikumine ja välimus, rakenduse loomise faasid

Õpitavad teemad: Valikulaused, võrdlus (loogiline avaldis), andurid. Teated

Mõisted, mida käsitletakse edaspidi: UML tegevusdiagramm, algoritmikeel.

Tunniks vajalikud materjalid: tööleht **Kraps, pall ja papagoi**, demo **kpp_2.sb** (kpp_2_s.sb)

Tunni käik:

- Rakenduse loomise faasid (vt. slaidid).
- Harjutuse **Kraps, pall ja papagoi** (tööleht ja demo, edaspidi KPP) esimeste etappide (1-3) tööülesande tutvustamine.
- KPP 1. etapp: Krapsu skriptide loomine ja testimine.
- KPP 2. etapp: Spraidi Papagoi loomine, tema jaoks Krapsu põhiskripti kopeerimine ja muutmine.
- Valikulausete [kui …] ja [kui … muidu …] tutvustamine ja võrdlemine.
- Papagoi skripti muutmine.
- UML tegevusdiagrammi ja algoritmikeele tutvustamine (vt. tööleht)
- KPP 3. etapp: Palli joonistamine.
- Palli skript – pall lendab üles kuni lava servani, kiiruse määramine (samm ja paus)
- Arutlus puutekontrolli asukoha üle (kas Palli või Papagoi skriptis)
- Palli skripti lõpetamine (teated *Pihtas* ja *Möödas*) ja testimine
- Krapsu ja Papagoi skriptid - reaktsioonid teadetele.

Märkused:

- Helindite ja heliklippide koostamist ning kasutamist kursuse materjalides ei käsitleta (arvutiklassis pole see arusaadavalt mõistlik ega võimalikki). Kui tunnis jääb aega, võiks heliandmeid vähemalt nimetada. Näiteks: Kraps teeb näu, Papagoi naerab, mängitakse mingit nooti, kui Pall tabab Papagoid jm.
- Tegemist on pikema harjutusega, mille jaoks on planeeritud 2x 45min. Esimeses osas (etapid 1-3) peaks õpilasele olema kõik varasematest tundidest enam-vähem tuttav. Teises osas (etapid 4-6) lisanduvad võõramad mõisted ja tegevused (muutuja, avaldis, taimer jm), mille juures võib kuluda rohkem aega. Seetõttu võiks üldise ülevaate muutujatest, nendega seotud käsuplokkidest ja plaanist kasutada neid loendurina anda selle tunni lõpus.

Projekt „Kass, pall ja papagoi“ 2

Kestus: 45 min

Tunni eesmärgid: lihtsa tervikliku rakenduse loomine kavandamisest realiseerimiseni (lõpp)

Õpioskused:

- õpilane tunneb mõistet **muutuja**
- õpilane oskab luua muutujaid Scratchis ja kasutada neid skriptides
- õpilane teab skriptide koostöö võimalusi ja oskab programselt lõpetada skriptide töö.

Õpitavad teemad: Muutuja, avaldis. Liugur; omistamine. Taimeri kasutamine. Tegevuste lõpetamine

Tunniks vajalikud materjalid: tööleht **Kraps, pall ja papagoi**, demo **kpp_2.sb** (kpp_2_s.sb)

Tunni käik:

- Esitlus „Rakenduste …“: **muutujad** (slaidid 33–37)
- KKP 4. etapp: muutujate *viskeid* ja *tabas* loomine
- Muutuja kasutamine loendurina – käskude [muuda … 1 võrra] paigutamine palli skripti, proovimine.
- Muutujatele *viskeid* ja *tabas* mängu algul algväärtuse 0 omistamine (võib olla eraldi skript).
- KKP 5. etapp: muutujate *aeg* ja *max_aeg* loomine. Muutuja *max_aeg* liuguri miinimumi ja maksimumi määramine.
- Muutujatele sobiva asukoha leidmine laval.
- Tutvumine taimeriga. Taimeri nullimine.
- Rakenduse töö lõpetamine käsuga [peata kõik], kui taimeri väärtus ületab muutuja *max_aeg* väärtuse.
- Kui enne rakenduse töö lõppu on vaja midagi teha (väljastada lõpetamisteade, viia spraite mingisse olekusse jm), tuleks enne käsku [peata kõik] anda korraldus [teavita *teade* ja oota]. Sel juhul lõpeb kogu rakenduse töö alles siis, kui kõik teatega **teade** käivitunud skriptid on oma töö lõpetanud.
- Spraidi **Teated** (tekst(id) mängu lõppemise kohta) loomine, mida näidatakse mängu aja lõppemisel, kui saabub **teade**.
- Mängu lõpetamise lubatud tabamiste arvu ja/või lubatud visete arvu täitumisel teevad õpilased iseseisvalt: muutujate *m_viskeid* ja *m_tabas* loomine. Õpetaja peaks juhtima tähelepanu asjaolule, et rakenduse töö lõpetamisel, kui visete arv on võrdne muutuja *m_viskeid* väärtusega, jääb viimase viske tulemus nägemata.
- KKP 6. etapp: palli liikumine koos Krapsuga. Skript Palli paigutamiseks Krapsu suhtes sobivale kohale.
- Arutelu, kuidas peaks selle skripti käivitama. Palli ja Krapsu skriptide täiendamine ja testimine
- Muutuja *lendab* loomine, selle kasutamine palli oleku määramiseks (lendab, ei lenda).
- Skriptide täiendamine ja testimine.
- KPP 7. etapp: rakenduse „dokumentatsioon“: kasutusjuhend ja spetsifikatsioonid.
- Valmis rakenduse objektide ja tegevuste võrdlemine töölehel (etapp 7) toodud kirjeldustega.

Scratch. Avaldised

Kestus: 45 min

Tunni eesmärgid: avaldiste koostamine Scratchis

Õpioskused:

- õpilane oskab koostada lihtsamaid avaldisi
- õpilane oskab luua muutujaid (Scratchis) ja kasutada neid avaldistes
- õpilane oskab koostada skripti tegevusdiagrammi alusel
- õpilane oskab organiseerida koostööd skriptide vahel.

Kordamisele tulevad mõisted: Scratchi kasutajaliides; muutuja

Õpitavad teemad: Avaldis, arv- ja tekstavaldised, funktsioon, argument; küsimus ja vastus

Tunniks vajalikud materjalid: tööleht **Ristkülik** (ristkylik), demo **ristkylik.sb** (ristkylik_s.sb)

Tunni käik:

- Harjutus Ristkülik: ülesande tutvustus
- muutujate *c*, *d*, *S*, *P*, *suhe* ja *Diag* (**NB!** globaalsed muutujad – kõigi spraitide jaoks) ning spraitide Õpetaja ja Õpilane loomine.
- Õpilase skriptina oluliste arvutusavaldiste koostamine *S*, *P*, *suhe* ja *Diag* leidmiseks, nende paigutamine plokki [võta …]. Arvutuste testimiseks võib muutujate *c* ja *d* väärtused esialgu määrata liugurite abil.
- tutvumine plokiga [küsi …] ja sisseehitatud muutujaga ***vastus***.
- Õpetaja skriptina koostada ristküliku külgede küsimine kasutajalt ja muutuja ***vastus*** väärtuse kirjutamine muutujatesse *c* ja *d*.
- Õpetaja skripti algus- ja lõpurepliikide ([ütle …]) ning [teavita arvuta ja oota] lisamine. Päiseplokkide valik Õpetaja ja Õpilase skriptidele. Rakenduse testimine.
- Kasutajaliidese viimistlemine:
    - käskude lisamine muutujate monitoride peitmiseks alguses ja näitamiseks sobival hetkel;
    - tekstispraitide loomine ja skriptide koostamine nende peitmiseks/näitamiseks;
    - Õpetaja skripti teadete lisamine tekstispraitide näitamiseks.

Scratch. Pliiats ja jälg

Kestus: 45 min

Tunni eesmärgid: joonistamise võimalused Scratchis

Õpioskused:

- õpilane tunneb võimalusi joonistamiseks skriptide abil;
- õpilane oskab kasutada juhuarve skriptides

Õpitavad teemad: pliiats ja jälg; juhuarvud

Tunniks vajalikud materjalid:

- töölehed **rist_ring, lilleaed**
- demo **rist_ring.sb** (rist_ring_s.sb), **lilleaed.sb** (lilleaed_s.sb)

Tunni käik:

- Scratchi joonistamisvõimaluste ja jälje tutvustamine.
- Harjutuse **Rist_ring** ülesande tutvustus, analüüs ja arutelu.
- Muutujate *a*, *b*, *S* ja *r* loomine.
- Krapsu skripti koostamine: *a* ja *b* väärtuse küsimine kasutajalt, *S* ja *r* arvutamine
- Joonise tegemiseks spraitide Rist ja Ring loomine.
- Ristküliku joonistamise skripti koostamine.
- Ringi joonistamine hulknurgana: üleminek polaarkoordinaatidelt ristkoordinaatidele.
- Ringi joonestamise skripti koostamine
- Tööleht **Lilleaed**: ülesande tutvustus, analüüs ja arutelu. vt. ka Scratchi näide GardenSecret kaustas Interactive_Art.
- Õpilased koostavad rakenduse ise

Märkused:

- Harjutus Rist_ring: Krapsu tegevuse ja ristküliku joonestamise kohta pole ehk vaja algoritmi ette anda, ringi joonestamiseks on algoritm siiski vajalik.
- Harjutus **Lilleaed** iseseisva tööna sobib arvestuslikuks (hindeliseks) ülesandeks.

Scratch. Protsesside juhtimine

Kestus: 45 min.

Tunni eesmärgid: selgitada protsesside juhtimist, korduste ja valikute kasutamist programmis; anda ülevaade Scratchi juhtimisplokkidest ja nende tööpõhimõttest.

Õpitulemused:

- õpilane tunneb ja oskab kasutada juhtimisvõimalusi (valik, kordus) programmis;
- õpilane tunneb Scratchi juhtimisplokke ja nende kasutamist;
- õpilane oskab kasutada erinevaid skriptide käivitamisvõimalusi

Õpilaste eelteadmised ja -oskused: Scratchi kasutajaliides, spraitide loomine, liikumis- ja välimusplokid

Õpitavad teemad: Kordus, valik; teadete saatmine ja vastuvõtmine

Tunniks vajalikud materjalid: tööleht **Juhtimine**, demo **juhtimine.sb** (juhtimine_s.sb)

Tunni käik:

- Juhtimisplokkide üldine ülevaade; meenutamine, milliseid on varem kasutatud.
- Harjutuse **Juhtimine** (tööleht ja demo) tutvustamine. Rakendust ei hakka koostama tunnis, õpilastele teha kättesaadavaks demo **juhtimine.sb**, selle alusel selgitab õpetaja juhtimisplokkide kasutamist jm
- NB! vt. skriptide täitmine sammhaaval.
- Päiseplokid. Rohelist lippu kasutatakse tavaliselt skriptide käivitamiseks rakenduse töö alguses. Hiireklõps spraidil või klahvivajutus tähendab kasutaja sekkumist. Päiseplokk [kui saabub teade …] on kasutusel, kui sündmust, millele tuleks reageerida, kontrollitakse mõnes teises skriptis. ***NB!*** *Vahel võib olla otstarbekas kasutada skriptide käivitamiseks rakenduse töö alguses klahvi klaviatuurilt (tühik) või lavale paigutatud käsunuppu start. Paljude spraitidega rakenduse puhul võib paljude päiseplokkide väljavahetamine olla tülikas ja tekitada vigu. Kui rakenduse kavandamisel pole sobiv käivitusmehhanism veel selge, võiks korraga käivitatavate skriptide päiseplokiks olla 'kui saabub teade …'; selle teavituse teket on vajadusel lihtne muuta; see võib tulla ka mitmest kohast.* Näites pannakse lillevaas püsti rakenduse alguses ja pärast ümberajamist hiireklõpsu peale.
- Valikuplokid. Valik ühest (tegevus toimub või mitte) ja valik kahe erineva võimaluse vahel.
- Kordused [korda …], [lõputult] ja [korda kuni …] on varasemast tuttavad; meenutada, kus kasutati.
- [Oota … sek] ja [oota kuni …] on varem kasutatud; meenutada. Näites ootab lillevaasiskript, et keegi vaasi ümber ajaks.
- Plokkis [lõputult kui …] on nii kordus kui valik. Kasutatakse ikka ja jälle tekkiva reageerimist nõudva olukorra puhul.
- Viitmuutuja (vt. muutuja *lõhkuja*). Scratchis on viitmuutuja väärtuseks spraidi nimi.
- Spraidi kostüüme saab kasutada nii liikumise imiteerimiseks kui ka spraidi muutumiseks (auto ja avarii), kui mõlemat kuvandit ei ole kunagi vaja näidata samaaegselt.
- Olenevalt arendusvahendi eripärast võib aeg-ajalt vaja minna kummalisi lahendusi. Nii nagu ühel spraidil võib olla mitu skripti, võib vahel ühe tegelase kujutamiseks olla kasutusel mitu spraiti. Vt. spraidid Kraps ja Kraps1; pööratavuse omadust ei saa Scratch 1.4 programselt muuta.

Märkused:

Harjutuse eesmärk on võimaldada õpetajal anda ülevaade protsessi juhtimise võimalustest (juhtimisplokkidest) ja näidata mõningaid lahendusi, mida võiks õpilastel iseseisvate tööde juures vaja minna.

Scratch. Muutuja skoop

Kestus: 45 min

Tunni eesmärgid: kirjeldada muutuja mõjupiirkonda (skoop), selgitada erinevust globaalsete ja lokaalsete muutujate vahel, lokaalsete muutujate kasutamine paralleelsetes protsessides. Eelmistes tundides õpitu kordamine.

Õpitulemused:

- õpilane tunneb mõisteid muutuja skoop (mõjupiirkond), globaalne ja lokaalne muutuja;
- õpilane oskab luua ja kasutada lokaalseid muutujaid Scratchis

Õpilaste eelteadmised ja -oskused: Scratchi kasutajaliides, spraitide loomine, liikumis- ja välimusplokid, juhtimisplokid, muutujate kasutamine

Kordamisele tulevad mõisted: muutuja, avaldis, valik, kordus

Õpitavad teemad: muutuja skoop, globaalsed ja lokaalsed muutujad

Tunniks vajalikud materjalid: tööleht **autod**, demo **autod.sb** (autod_s.sb)

Tunni käik:

- Muutujate skoobi ja lokaalsete muutujate tutvustamine Scratchis.
- Lokaalne muutuja kui spraidi (lisa)omadus.
- Harjutuse **autod** (tööleht ja demo) tutvustamine.
- Spraidi auto ja muutujate *ringe* (globaalne), *ring*, *aeg* (lokaalsed) loomine.
- Arutlus: kui auto kiirus (või baaskiirus) on ka auto omadus, tuleks nüüd teha ka muutuja *kiirus*.
- Auto skripti koostamine ja testimine.
- Autost kahe koopia tegemine; koopiate ja nende muutujate paigutamine sobivatesse kohtades laval.
- Kohtuniku (Krapsu) skripti koostamine.
- Arutelu, kuidas leida kolmest arvust suurim/väikseim (esialgu võiks õpilastele algoritme mitte näidata).
- Väikseima aja ja võidusõidu võitja leidmine.
- Rakenduse viimistlemine ja testimine.

Rehmaatika

Kestus: 90 min (2 x 45 min)

Tunni eesmärgid: tutvustada ja võrrelda UML tegevusdiagrammi elemente ja nn algoritmikeelset protseduuri; mahukama näite baasil kirjeldada ülesande jagamist väiksemateks osadeks ning rakenduse koostamist ja silumist järk-järgult.

Õpitulemused:

- õpilane saab aru etteantud algoritmidest ja tegevusskeemidest
- õpilane oskab koostada kirjelduse/skeemide alusel rakenduse
- õpilane oskab analüüsida ülesande etappe ja koostada rakendust järk-järgult

Õpilaste eelteadmised ja -oskused: Scratchi kasutajaliides, spraitide loomine, liikumis- ja välimusplokid, juhtimisplokid, muutujad, avaldised

Kordamisele tulevad mõisted: muutujad, avaldised, valikud, kordused

Õpitavad teemad: UML tegevusdiagrammi elemendid, nn. algoritmikeelsed laused

Tunniks vajalikud materjalid: tööleht **Rehmaatika,** demo **rehmaatika.sb** (rehmaatika_s.sb)

Tunni käik:

- UML tegevusdiagrammi elementide tutvustamine; algoritmikeele kirjeldamine (vt. slaidid).
- Harjutuse **Rehmaatika** (tööleht ja demo) tutvustamine. Kumb algoritmi esitusviis on paremini mõistetav?
- Kuigi üldpilt on olemas, võiks rakendust realiseerima asuda siiski samm-sammult. Lava skriptides ei saa kasutada plokki [ütle …], seepärast tuleks tehte küsimine ja kommenteerimine realiseerida mõne spraidi, näiteks Krapsu, skriptina.
- Muutujate *a*, *b*, *c*, *tulem* (võib-olla ka tekst) loomine.
- *a*, *b*, ja *c* leidmine; esialgu võivad a ja *b* väärtused olla vahemikus 1…10.
- liitmistehte teksti koostamine (muutuja tekst), õige vastus *tulem* on *c*.
- Vastuse küsimine kasutajalt, vastuse kontroll – ütlemine õige/vale.
- Lahutamistehte lisamine: muutuja tehe (leitakse juhuarvuna 1..2) loomine, valikuploki lisamine, lahutamistehte teksti (*c* – *a* =) koostamine, õige vastus on *b*.
- Muutujate *min* ja *max* loomine (esialgu võib nende väärtusi muuta liuguri abil), nende kasutamine *a* ja *b* genereerimisel.
- Muutujate punkte (punktisumma) ja *p* (punkte ühe tehte eest, esialgu võib muuta liuguri abil) loomine; [võta punkte = 0] ja [muuda punkte p võrra] lisamine skripti.
- Raskusastmed: lisada muutuja tase (liugur, min/max 1..3); arutlemine, milline oleks iga taseme puhul *min* ja *max* ning õiglane punktide arv tehte eest – *p*.
- Koostada skriptid mõlema algoritmivariandi alusel; arutleda, kumb on otstarbekam/arusaadavam. Kasutada enammeeldinud skripti raskusastme kohaldamisel: päiseplokk [kui saabub teade …]. **NB!** põhiskriptis peaks olema [teavita … ja oota].
- Valmisskripti võrrelda etteantud algoritmiga.
- Peita muutujate monitorid, jääb ainult tase.
- Ajakontrolli lisamine (vt. varasemad harjutused)
- Kasutajale vajalik info ja kirjeldused kirjutatakse projekti märkustesse
- Lisaülesanded
- Täiendavaks lisaülesandeks võib olla ka algoritmi (teksti) täiendamine, kui lisaülesanne on Scratchis realiseeritud.

Scratch. Arvavaldised

Kestus: 45 min

Tunni eesmärgid: pikkade ja/või keeruliste arvavaldiste koostamise harjutamine

Õpitulemused:

- õpilane oskab koostada keerulisi arvutusavaldisi;
- õpilane oskab kasutada Scratchi sisefunktsioone;
- õpilane teab võimalusi astendamistehte realiseerimiseks Scratchis

Õpilaste eelteadmised ja -oskused: Scratchi kasutajaliides, juhtimisplokid, muutujad, avaldised

Kordamisele tulevad mõisted: muutuja, avaldis, valik

Õpitavad teemad: Scratchi sisefunktsioonid, astendamine Scratchis $x^y = e^{y \cdot \ln x}$

Tunniks vajalikud materjalid: tööleht **Ideaal**, demo **ideaal.sb** (ideaal_s.sb)

Tunni käik:

- Ülevaade tehetest ja funktsioonidest (plokid grupis **Tehted**).
- Harjutuse **ideaal** (tööleht ja demo) tutvustamine.
- Muutujate loomine.
- Kasutajaliidese koostamine; arutlemine kuidas saadakse algandmed (küsitakse kasutajalt, kasutaja määrab liuguri abi vm), kuhu/kuidas väljastatakse tulemused.
- Paari avaldiste koostamine õpetaja juhendamisel.
- Õpetaja kirjeldab töövõtted avaldiste mugavamaks koostamiseks (tehteploki muutmine, avaldise või selle osa kopeerimine jm).
- Õpilased koostavad ülejäänud avaldised iseseisvalt.
- Rakenduse testimine erinevate andmetega.
- Kasutajale vajalik info ja kirjeldused kirjutatakse projekti märkustesse.

Arvu arvamine

Kestus: 45 min

Tunni eesmärgid: iseseisev töö tunnis: rakenduse koostamine etteantud tegevusskeemi järgi.

Õpitulemused: Õpilane oskab iseseisvalt koostada Scratchi rakenduse etteantud tegevusskeemi alusel.

Õpilaste eelteadmised ja -oskused: algoritmi mõistmine, juhtimisplokid (valik, kordus), muutujad, avaldised; kasutajaliides

Tunniks vajalikud materjalid: tööleht **arvamine**, demo **arvamine.sb** (arvamine_s.sb)

Tunni käik:

Ülesande tutvustamine: nõutud rakenduse ja etteantava skeemi sõnaline kirjeldamine

Ülesande põhiosa (rakenduse koostamine etteantud tegevusskeemi järgi) võiks anda tunnikontrollina.

Hinnata võiks:

- **eelkõige tegevuse vastavust skeemile**
- töö teostamise täpsust ja kiirust
- huvitavat animatsiooni
- lisaülesannete täidetust

Labürint

Kestus: 45 min

Tunni eesmärgid: lihtsa mängu koostamine, varemõpitud oskuste kordamine

Õpitulemused:

- õpilane oskab jaotada tegevusi skriptide vahel;
- õpilane oskab kasutada Scratchi anduriplokke;
- õpilane oskab kasutada lokaalseid muutujaid mitme sarnase spraidi loomisel

Õpilaste eelteadmised ja -oskused: Scratchi kasutajaliides, liikumis- ja välimusplokid, juhtimisplokid, muutujad, avaldised, anduriplokkide kasutamine.

Tunniks vajalikud materjalid: tööleht **Labürint** (labyrint), demo **labyrint.sb** (labyrint_s.sb)

Tunni käik:

- Tööleht **labürint**: ülesande tutvustus, demo näitamine, analüüs ja arutelu.
- Labürindi ja peategelase (Pacman) joonistamine. Labürint on otstarbekas luua lava taustana.
- Pacmani liikumisskriptide koostamine, tegevuste jaotamine skriptide vahel, nn alamskripti koostamise põhjendamine (vrdl Rehmaatika raskusastmete kohaldamine).
- Kulla vm varanduse loomine; kulla skriptis peaks olema kontroll [kas puudutab Pacmani].
- Kõigepealt üks kullasprait korralikult (koos skpripti(de)ga) valmis, siis sellest vajalik hulk koopiaid ja nende paigutamine labürinti.
- Muutujate *summa*, *alles* ja *max_kuld* loomine. Muutuja *max_kuld* väärtuseks peab olema kullakangide arv labürindis; kui rakenduse koostaja lisab või eemaldab mõne kullaspraidi, peab ta muutma ka *max_kuld* väärtust. *Alles* ja *Summa* peaks algväärtuse 0 saama mingis (näiteks lava) skriptis, mis käivitub rakenduse töö alguses, nende väärtusi muudetakse kullakangide skriptides (alles=alles-1, summa=summa+väärtus)
- Kolli ja muude takistuste loomine.
- Ajakontrolli lisamine (vt. varasemad harjutused)
- Rakenduse töö lõpeb, kui kõik kuld on korjatud, Pacman tabatakse või aeg saab otsa. Kontrollida!
- Kasutajale vajalik info ja kirjeldused kirjutatakse projekti märkustesse

Märkus:

Õpilased võiks Kolli tegutsemise kohta joonistada UML tegevusdiagrammi.

Scratch. Tekstavaldised

Kestus: 45 min

Tunni eesmärgid: tutvustada tekstide ühendamist ja sümbolite eraldamist (loenduriga kordus), avaldise väärtuse leidmist keerulisemate skriptide abil; sissejuhatus loendite kasutamisse.

Õpitulemused:

- Õpilane tunneb tekstavaldisi
- õpilane oskab koostada skripte teksti käsitlemiseks sümbolite kaupa

Õpilaste eelteadmised ja -oskused: Scratchi kasutajaliides, juhtimisplokid, muutujad, avaldised

Õpitavad teemad: tekstiavaldised, kordus loenduriga

Mõisted, mida käsitletakse edaspidi: loendid

Tunniks vajalikud materjalid: tööleht **Tekst tagurpidi** (tagurpidi), demo **tagurpidi.sb** (tagurpidi_s.sb)

Tunni käik:

- ülesande **tekst tagurpidi** tutvustus, demoga tutvumine, analüüs ja arutelu
- kasutatavate spraitide valimine. Et lava skriptides ei saa kasutada plokki [ütle …] tuleks tegevus(ed) realiseerida mingi spraidi skriptina.
- muutujate *tekst1* (algne tekst), *tekst2* (tulemustekst), *mitu* (sümbolite arv tekstis) ja *nr* (abimuutuja – loendur) loomine (võivad olla ainult ühe spraidi jaoks).
- Esialgu võiks näidata, kuidas *nr* toimib korduses loendurina (kasvavalt/kahanevalt), kasutada plokki [ütle *nr*]. Seejärel lisada plokid sümbolite eraldamiseks ja ühendamiseks uueks tekstiks, *tekst2* ja *nr* algväärtustamiseks ning *tekst1* küsimine kasutajalt ja selle sümbolite arvu omistamine muutujale *mitu*.
- Skripti testimine
- Loend kui võimalus paiguta iga sümbol eraldi mäluvälja (muutujasse). Luua loend *symb*, teha eelnevast skriptist koopia ja lisada kordusesse plokk sümboli lisamiseks loendisse [lisa *symb*-le s] ja algusesse [eemalda *symb* kõik]; testida tulemust.
- Loendi kasutamise võimalused: loendi pikkus, loendi element (esimene, viimane, etteantud järjenumbriga).
- ülesande **sõnad tagurpidi** tutvustus, demoga tutvumine, analüüs ja arutelu.
- Luua loend sõnad ja koostada skript tekst1 paigutamiseks loendisse sõnade kaupa. Testida.
- Lisada skriptile loendist võetud sõnade ühendamine uueks tekstiks muutujas *tekst2*. See võib olla ka eraldi skriptina, nii on mugavam testida.
- Kasutajale vajalik info ja kirjeldused kirjutatakse projekti märkustesse

Märkus:

Sõnad vastupidisesse järjestusse saab muidugi ka ilma loendita.

Scratch. Loendi kasutamine

Kestus: 45 min

Tunni eesmärgid: selgitada varasema harjutusülesande baasil loendi kasutamise eeliseid rakenduses; näidata väärtuste lisamist loendisse ja nende lugemist sealt.

Õpitulemused:

- õpilane oskab luua loendi Scratchis ja lisada/lugeda loendi väärtusi skriptides;
- õpilane teab, millised on loendi (muutujate massiivi) kasutamise eelised võrreldes hulga skalaarmuutujatega.

Õpilaste eelteadmised ja -oskused: Scratchi kasutajaliides, objektid ja skriptid, valiku ja korduse plokid skriptides

Õpitavad teemad: loend (muutujate massiiv), muutuja (loendur) loendi järjenumbrina.

Tunniks vajalikud materjalid: tööleht **autod2** (autod_loend), demo **autod_loend.sb** (autod_loend_sb.sb)

Tunni käik:

- Rakenduse **autod** objektide ja tegevuste meenutamine (autode võidusõidu võib luua kas uuesti või teha koopia varasemast rakendusest).
- Luua üks auto ja selle skript(id), siis teha sellest spraidist sobiv hulk koopiaid ning muuta koopiate kostüümi ja asukohta.
- Realiseerida võitja teatamine (võitja on loendis esimene).
- Mõelda välja ja realiseerida kasutajapoolne võitja ennustamine.
- Koostada skript ennustatud auto järjenumbri leidmiseks loendis.
- Mõelda välja ja realiseerida ennustatud auto tulemuse teatamine.
- Kasutajale vajalik info ja kirjeldused kirjutatakse projekti märkustesse

Vestlus

Kestus: 45 min

Tunni eesmärgid: näidata Scratchi loendisse väärtuste sisestamist klaviatuurilt; loendi väärtuste kasutamine skriptis.

Õpitulemused:

- õpilane oskab luua ja kasutada loendeid Scratchis
- õpilane oskab kasutada loendurit korduslauses järjenumbrina viitamaks loendi elemendile.

Õpilaste eelteadmised ja -oskused: Scratchi kasutajaliides, objektid ja skriptid, kordus, teated teistele skriptidele, viitamine loendi elemendile.

Kordamisele tulevad mõisted: muutuja, loend, kordus

Õpitavad teemad: teksti pikkus (sümbolite arv tekstis)

Tunniks vajalikud materjalid: tööleht **vestlus**, demo **vestlus.sb** (vestlus_s.sb)

Tunni käik:

- Harjutus **vestlus** (tööleht ja demo) tutvustamine.
- Loendite loomine ja dialoogi sisestamine
- Spraitide ja skriptide loomine.
- Rakenduse testimine.
- Teksti kuvamise aeg pannakse sõltuma teksti pikkusest (*baasaeg* ja sümbolite arv sekundis – *k*); sobiva *baasaja* ja *k* määramine proovimise teel.
- Rakenduse muutmine (teha koopia failist) nii, et dialoogi mõlema osapoole laused on samas loendis.

Sõnastik

Kestus: 45 min

Tunni eesmärgid: näidata Scratchi loendisse väärtuste lugemist failist; loendi väärtuste kasutamine skriptis.

Õpitulemused:

- õpilane oskab luua ja kasutada loendeid Scratchis
- õpilane oskab kasutada loendurit korduslauses järjenumbrina viitamaks loendi elemendile.

Õpilaste eelteadmised ja -oskused: Scratchi kasutajaliides, objektid ja skriptid, kordus, viitamine loendi elemendile.

Kordamisele tulevad mõisted: muutuja, loend, kordus, loenduri kasutamine loendis järjenumbrina

Tunniks vajalikud materjalid:

- Töölehed **Sõnastik. Keele test** (sonastik1), **Sõnastik. Tõlkimine** (sonastik2)
- demod **sonastik1.sb** (sonastik1_s.sb), **sonastik2.sb** (sonastik2_s.sb)

Tunni käik:

- Harjutus **sõnastik1 - test** (tööleht ja demo) tutvustamine, analüüs.
- Loendite loomine; andmete lugemine failist loendisse.
- Spraitide ja (abi)muutujate loomine.
- Skripti (test – kasutajalt etteantud sõna tõlkimise nõudmine ja kontroll) koostamine järk-järgult.
- Tulemuse testimine.
- Harjutus **sõnastik2 - tõlkimine** (tööleht ja demo) tutvustamine, analüüs.
- Spraitide ja (abi)muutujate loomine.
- Skripti (kasutaja antud sõnale vaste leidmine) koostamine järk-järgult.
- Tulemuse testimine.
- Kasutajale vajalik info ja kirjeldused kirjutatakse projekti märkustesse.

Algoritme loenditega

Kestus: 45 min

Tunni eesmärgid: lihtsate algoritmide (summa, keskmise, suurima/väiksema leidmine) tutvustamine

Õpitulemused:

- õpilane oskab koostada skripte tegevusskeemi/algoritmikeelse kirjelduse alusel
- õpilane oskab kirjeldada tegevusi loendi elementidega (summeerimine, loendamine jm)

Õpilaste eelteadmised ja -oskused: Scratchi kasutajaliides, juhtimisplokid, muutuja, loend

Kordamisele tulevad mõisted: muutuja, loend, kordus, muutuja kasutamine loendis järjenumbrina

Tunniks vajalikud materjalid: tööleht **Algoritme loenditega** (algoritme), demo **algoritme.sb** (algoritme_s.sb)

Tunni käik:

- Ülesannetega tutvumine.
- Loendite loomine ja väärtuste sisestamine.
- Hindade keskmine – abimuutujate ning spraidi ja skripti loomine.
- Tulemuse testimine, skripti täitmine sammhaaval.
- Suurima väärtuse ja selle järjenumbri leidmine - abimuutujate ning spraidi ja skripti loomine.
- Tulemuse testimine.
- Etteantud väärtuse järjenumbri leidmine - abimuutujate ning spraidi ja skripti loomine.
- Tulemuse testimine.
- Info ja kirjelduste kirjutamine projekti märkustesse.

Tunnikavad. VBA

MS Office's sisalduv arendusvahend VBA (*Visual Basic for Application*) Exceli baasil on selles kursuses teise arendusvahendina õppimiseks välja pakutud järgmistel kaalutlustel:

- Paljudes koolides kasutatakse infotöö õpetamisel ja kontoripakettide tutvustamisel MS Office vahendeid, st MS Excel on arvutiklassides olemas.
- Kui õpilased on MS Excelit varasemas õppetöös kasutanud, on tuttavas keskkonnas lihtsam; juurde-õpitavad oskused annavad võimaluse koostada keerulisemaid rakendusi, mida ainult Exceli vahenditega realiseerida ei saanud, uue arendusvahendi õppimisel on kohe olemas kasulik praktiline väljund.
- MS Exceli töövihiku näol on tegemist nn dokumendipõhise rakendusega, kus programm, objektid ja andmed on kõik ühes failis. Töölehtedele on mugav kujundada kasutajaliides, kasutada saab töölehe lahtreid ja erinevaid graafilisi objekte (kujundid, vormi elemendid jm).

Asudes õppima VBAd peaks õpilastel olema mõningad eelteadmised MS Exceli kasutamisest:

- töölehtede lisamine, kustutamine, kopeerimine, ümbernimetamine
- ridade-veergude vormindamine, kopeerimine, lisamine, eemaldamine
- lahtriplokkide vormindamine, kopeerimine, kustutamine, lisamine, eemaldamine
- valemite koostamine ja kopeerimine
- lahtriplokile nime määramine.

Lisamooduli VBA harjutusülesannetes on eeldatud, et õppur juba oskab programmeerida (Scratchis). Esimesed harjutusülesanded on keele süntaksist ja erisustest ning editori võimalustest ülevaate saamiseks. Kasutatud on kas väga lihtsaid või varem Scratchis tehtud rakendusi, mille algoritmid on varasemast tuttavad.

Põhi- ja lisamoodul ei pea olema üksteisest rangelt eraldatud. Lisamooduli esimesi ülesandeid võiks kasutada juba enne põhimooduliga (Scratchiga) lõpetamist. Näiteks võiks teise valitud programmeerimis-keele süntaksit näidata ja mõned Scratchis tehtud tuttavad ülesanded selles realiseerida juba enne loendite tutvustamist. Nii on lihtsam kahe vahendi võimalusi võrrelda. Edaspidi saab siis demonstreerida töötavat algoritmi Scratchis ja lasta õppuritel iseseisvalt teha samalaadne rakendus teises keeles.

MS Excel

Kestus: 45 min

Tunni eesmärgid: vaadata üle MS Exceli lihtsamad vahendid ja baasoskused, mis on vajalikud VBA toega rakenduste koostamisel Exceli töölehel.

Õpitulemused:

- õpilane oskab vormindada andmeid MS Exceli töölehel
- õpilane oskab koostada lihtsaid valemeid ja teab tingimusi nende kopeerimiseks
- õpilane oskab määrata lahtritele nimed ja kasutada neid valemites

Õpilaste eelteadmised ja-oskused: töökogemus MS Exceliga

Kordamisele tulevad mõisted: tööleht, lahtriplokk, valem, funktsioon

Tunniks vajalikud materjalid: tööleht **MS Excel**

Tunni käik:

- Ülesandega tutvumine. Tabelis ei pruugi olla õige tulumaksuvaba miinimum ja tulumaksuprotsent, tabelis ei pruugi olla arvutatavaid väärtusi
- Andmete kopeerimine Exceli töölehele, väärtuste kustutamine lahtritest, kus peaks olema valemid.
- Töölehele nime andmine – palgaleht.
- Veergudele sobivate laiuste määramine, tabeli vormindamine (font, raamjooned)
- Arvuvormingu valimine. Tulumaksuprotsent protsendivormingus (näidata õpilastele seda lahtrit ka üldvormingus).
- Õpilased koostavad lihtsamad valemid iseseisvalt (põhipalk, töötasu, maksta, kokkuvõtted). Valemites lahtriaadressid. **NB!** Valemi kopeerimiseks mõned absoluutaadressid, abiks funktsiooniklahv F4.
- Preemia ja tulumaksu valemite koostamine vajadusel õpetaja abiga.
- Koopia tegemine töölehest
- Nimede määramine tabeli tulpadele, tulumaksulahtritele ja preemiasummale. Õpetaja näitab erinevaid võimalusi (nime kirjutamine nimeboksi, nimede määramine veeru- ja reapäise abil).
- Õpilased koostavad valemid uuesti. Valemites nimed.
- Tulpdiagrammi koostamine, erinevad variandid.

Preemia = Põhipalk/põhipalk_kokku*Preemiasumma

Tulumaks = IF(Töötasu<=Tulumaksuvaba;0;(Töötasu-Tulumaksuvaba)*Tulumaksuprotsent)

VBA. Sissejuhatus

Kestus: 45 min

Tunni eesmärgid: tutvustada VBA (*Visual Basic for Applications*) vahendeid ja võimalusi MS Exceli baasil; näidata mõningaid valmisrakendusi ja kirjeldada nende alusel programmide koostamist ja redigeerimist, kasutajaliidese loomist Exceli töölehel ning protseduuride täitmist samm-sammult.

Õpitulemused:

- õpilane tunneb üldjoontes VBA rakenduse koostamise vahendeid MS Office dokumentides
- õpilane teab, kuidas sisestatakse ja salvestatakse VB protseduurid MS Office dokumentides
- õpilane oskab kasutada/käivitada VB protseduure (makrod ja töölehefunktsioonid) MS Excelis

Õpilaste eelteadmised ja-oskused: töökogemus Scratchi rakenduste loomisel (selle asemel sobivad baasteadmised mõnest muust programmeerimissüsteemist)

Kordamisele tulevad mõisted: rakendus**,** muutuja, omistamine, avaldis, valik, kordus

Õpitavad teemad: protseduur, funktsioon, VBA akna elemendid

Tunniks vajalikud materjalid: töövihik **yl1_demod.xls**: lehed **Arva**, **Fun**, **Auto**, **Anima**

Arva – kasutaja saab ära arvata arvu (1…100). Antud on tegevusskeem. Kasutusel: protseduuri ja muutujate kirjeldamine, dialoog kasutajaga (sisend-väljund), omistamine, avaldis, valikud, kordus (do…loop).

Fun – VB funktsioonid kehamassiindeksi ja hinnangu andmiseks inimese pikkuse ja kaalu alusel. Kasutusel: VB funktsiooni, parameetrite, muutujate kirjeldamine, valik, VB funktsiooni kasutamine töölehel (valemis).

Auto ja **Anima** – graafikaobjektide ja lahtrite kasutamine Exceli töölehel animatsioonide loomisel.

Tunni käik:

- Näide **Arvamine**, erinevate keelte käskude võrdlemine.
- VBA ja Exceli töölehe rakenduste üldine tutvustamine. Demod, näitamaks VBA mõningaid kasutusvõimalusi.
- VB protseduuride tekst VB editori aknas. Demo **Arva**. Scratchis kasutatud laused VB protseduuris (omistamine, avaldis, valik, kordus).
- Protseduuride täitmine sammhaaval. Katkestuspunktide lisamine ja eemaldamine. Õpilased proovivad neid tegevusi oma arvutites

NB! Kui VB funktsiooni kasutatakse Exceli töölehel valemis, peab protseduuri tekst asuma üldmoodulis.

VB protseduurid

Kestus: 45 min

Tunni eesmärgid: VB editori võimaluste tutvustamine; protseduuride sisestamine klaviatuurilt

Õpitulemused:

- õpilane tunneb VBA akna elemente ja oskab nendest kujundada mugava töökeskkonna
- õpilane tunneb VB editori põhivõimalusi
- õpilane oskab sisestada lihtsa protseduuri teksti ja parandada lihtsamad süntaksivead
- õpilane oskab kasutada/käivitada protseduure (makrod ja töölehefunktsioonid) MS Excelis

Õpilaste eelteadmised ja -oskused: baasteadmised VBA rakenduste loomise vahenditest MS Excelis

Kordamisele tulevad mõisted: protseduur, funktsioon, muutuja, omistamine, avaldis, valik

Õpitavad teemad: muutuja skoop, parameetrid ja argumendid, valikulause

Tunniks vajalikud materjalid: tööleht **VB protseduurid**

Ülesanded:

- Kaal: koostada skeemi alusel algul makro (VB alamprotseduur) - algandmed küsida kasutajalt - ja hiljem VB funktsioon, mis näitaks etteantud pikkuse ja kaalu alusel, kas inimene on alakaaluline, ülekaaluline või normaalse kaaluga. Kui pikkuse ja kaalu vahe jääb piiridesse 90 kuni 110, on kaal normaalne, kui vahe on väiksem, on kaal liiga suur (ülekaaluline), kui vahe on suurem, on kaal liiga väike (alakaaluline).
- Funktsioon. Koostada VB funktsioonid etteantud avaldiste alusel, kasutada neid töölehel funktsiooni väärtuste leidmisel.

Tunni käik:

- VBA akna elementide kordamine, töökeskkonna kujundamine
- VB editori võimaluste tutvustamine
- **Harjutus Kaal**. Protseduuri lausete sisestamine klaviatuurilt (kirjeldamine, valikulause, erinevad võimalused)
- Protseduuri käivitamine VBA aknast; demonstreerimine, protseduuri täitmine samm-sammult
- Sama ülesanne VB funktsioonina; parameetrid, tagastatav väärtus. Käsuaken (*immediate window*)
- VB funktsiooni kasutamine Exceli töölehel, protseduuri täitmine samm-sammult
- **Harjutus Funktsioon.** Astendamine. Funktsiooni väärtuste tabeli koostamine
- Koostada töölehele tabel ning lahtrid *algus* ja *samm* (võib määrata nimed).
- x arvutamiseks kirjutada valemid (vt. tööleht)
- Koostada VB funktsioonid ja kasutada neid valemites (valemite kopeerimine!)

NB! Kui VB funktsiooni kasutatakse töölehel, peab protseduuri tekst asuma üldmoodulis.

VBA. Arvavaldised

Kestus: 45 min

Tunni eesmärgid: arvavaldiste koostamine, VB funktsioonide kasutamine, valikulause.

Õpitulemused:

- õpilane oskab koostada arvavaldisi VB-s
- õpilane teab mõningaid VB funktsioone
- õpilane oskab koostada valikulauseid

Õpilaste eelteadmised ja -oskused: VBA akna ja VB redaktori kasutamine; baasteadmised VBA rakenduste loomise vahenditest MS Excelis.

Kordamisele tulevad mõisted: muutuja, avaldis, omistamine

Õpitavad teemad: VB sisefunktsioonid, astendamine, valikulause

Tunniks vajalikud materjalid: tööleht **Ideaal**

Tunni käik:

- Ülesande tutvustamine, selline harjutus on tehtud ka Scratchis.
- Kasutajaliidese kujundamine töölehele, lahtritele määrata nimed. Algandmed (sugu, vanus, pikkus, kaal) loetakse töölehelt või sisestatakse klaviatuurilt, tulemused kirjutatakse töölehele. Arvutuste käivitamiseks luua käsunupp või mingi graafikaobjekt.
- VB protseduuri koostamine.
- Hinnangu leidmine võiks olla eraldi funktsioon, mille poole pöördutakse VB protseduurist. Selgitada õpilasele veel kord, mis on funktsioon (protseduur) ja kuidas seda kasutatakse.
- Rakenduse testimine (käskude täitmine samm-haaval, katkestuspunktide lisamine ja eemaldamine), vigade avastamine ja parandamine.

Exceli objektid *Range* ja *Shape*

Kestus: 45 min

Tunni eesmärgid: objektide *Range* ja *Shape* omaduste ja meetodite tutvustamine

Õpitulemused:

- õpilane teab MS Exceli objektide *Range* (lahtriplokk) ja *Shape* (kujund) mõningaid omadusi (*value, left, top, width, height* jm), oskab nende väärtusi programmi abil lugeda ja muuta.
- õpilane oskab töölehele kujundada lihtsa kasutajaliidese

Õpilaste eelteadmised ja -oskused: VBA akna ja VB redaktori kasutamine; baasteadmised VBA rakenduste loomise vahenditest MS Excelis.

Kordamisele tulevad mõisted: objekt, omadus, muutuja, omistamine.

Õpitavad teemad: muutuja tüüp

Tunniks vajalikud materjalid: tööleht **Ristkülik ja ring**

Koostada VBA rakendus, mis etteantud ristküliku külgede pikkuse alusel arvutab läbimõõdu ringi jaoks, mille pindala on võrdne ristküliku pindalaga, leiab ristküliku ümbermõõdu ja ringjoone pikkuse suhte ning muudab ristküliku ja ringi mõõtmeid.

Tunni käik:

- Objektide *Range* ja *Shape* omaduste ja meetodite tutvustamine (vt. ka esitlus). Objektide Range ja Shape omaduste lugemine ja muutmine VB programmis.
- Harjutus **Rist_ring**. Kasutajaliidese ja tegevuste kavandamine.
- Kasutajaliidese loomine: kujundite Rist ja Ring loomine ja nimetamine; lahtrite, kuhu paigutatakse andmed, nimetamine ja vormindamine.
- Programmi sisestamine ja testimine samm-sammult õpetaja juhendamisel. Protseduuri teksti esimese versiooni võib anda õpilastele ka koos tööjuhendiga töölehel.
- Programmi laused on lühemad ja paremini loetavad, kui kasutada muutujaid ja nimega konstante. Protseduurist tehakse koopia (muudetakse nimi) ja muudetakse lauseid.
- Lisaülesanne

```
Sub Rist_Ring()
Range("pind").Value = Range("a").Value
Range("b").Value
Range("ümber") = 2 * (Range("a") + Range("b")
Range("d") = Sqr(4 * Range("pind") / 3.14159)
Range("suhe") = Range("ümber") / (3.14159 * Range("d"))
 ' Kujundite mõõtmete muutmine
  Shapes("rist").Width = Range("a") * 72 / 2.54
  Shapes("rist").Height = Range("b") * 72 / 2.54
  Shapes("ring").Width = Range("d") * 72 / 2.54
  Shapes("ring").Height = Shapes("ring").Width
End Sub
```

```
Sub Rist_Ring()
Dim a, b, d, suhe, s, P
Const cm_p = 72 / 2.54, pi =3.14159
a = Range("a").Value
b = Range("b")        '.Value võib puududa
s = a * b
P = 2 * (a + b)
d = Sqr(4 * s / pi)
Range("d") = d
Range("pind").Value = s ' Pindala
Range("ümber") = P ' Ümbermõõt
Range("suhe") = P / (pi * d)
 ' Kujundite mõõtmete muutmine
Shapes("rist").Width = a * cm_p
Shapes("rist").Height = b * cm_p
Shapes("ring").Width = d * cm_p
Shapes("ring").Height = Shapes("ring").Width
End Sub
```

Animatsioon Auto

Kestus: 45 min

Tunni eesmärgid: korduslausete kasutamine; animatsiooni loomise eripära tutvustamine VBA rakenduses (korraldus DoEvents protseduuris paus), viitmuutuja

Õpitulemused:

- õpilane teab korduse Do .. Loop kasutamise võimalusi VB-s
- õpilane oskab kasutada korduslauset animatsiooni loomisel
- õpilane teab korraldust DoEvents ja oskab kasutada valmis protseduuri paus

Õpilaste eelteadmised ja -oskused: VBA akna ja VB redaktori kasutamine; baasteadmised VBA rakenduste loomise vahenditest MS Excelis; algteadmised animatsiooni põhimõtetest; *Shape*-objekti mõned omadused (*left, top, width, height, rotation* jm)

Kordamisele tulevad mõisted: *Shape*-objekti mõned omadused (*left, top, width, height, rotation* jm)

Õpitavad teemad: korduslaused VB-s

Tunniks vajalikud materjalid: tööleht **Auto** (auto sõit töölehel)

Tunni käik:

- Animatsiooni põhimõtete meenutamine, näited
- Autot kujutava graafikaobjekti loomine (nimi *auto*).
- Protseduuri **Auto1** koostamine etteantud tegevusskeemi alusel. ***Do .. Loop*** korduslause tutvustamine. Programmi sammhaaval täitmine, katkestuspunktide lisamine ja eemaldamine.
- Käsu ***DoEvents*** ja valmisprotseduuri *paus* tutvustamine.
- Protseduurist **Auto1** koopia tegemine (**Auto2**).
- Kujundile auto viitava muutuja kasutamine.
- Objekti joon (nimi *Fin*) ning lahtrite *samm* ja *aeg* lisamine
- Protseduuri täiendamine tegevusskeemi järgi õpetaja juhendamisel.
- **Auto2** testimine sammhaaval, vajadusel vigade parandamine.
- Protseduurist **Auto2** koopia tegemine (**Auto3**). Lahtrite *ringe* ja *ring* määramine töölehel.
- Õpetaja selgitab ***For***-korduslause kasutamist
- Õpilased lisavad korduslause oma protseduuri (vt tegevusskeem) ja testivad sammhaaval.
- Lisada protseduur auto asetamiseks töölehe vasakusse serva ja nupud protseduuride käivitamiseks.
- Lisaülesanne (teise auto lisamine).

```
Sub Paus(pp)
' Paus pikkusega pp sek
  Dim pl  ' pausi lõpp
  pl = Timer() + pp
  Do
    DoEvents
  Loop While Timer() < pl
End Sub

Sub kohale()
Shapes("auto").Left = 0
End Sub

Sub Auto1()
Dim a As Shape
Set a = Shapes("auto")
Do
   a.IncrementLeft 10
   Paus 0.1
Loop Until a.Left > 500
End Sub
```

```
Sub Auto2()
Dim a As Shape, samm, algaeg
Set a = Shapes("auto"):  samm = Range("samm")
algaeg = Timer()
Do
   Shapes("auto").IncrementLeft samm
   Range("aeg") = Timer() - algaeg
   Paus 0.1
Loop Until a.Left > Shapes("fin").Left
End Sub

Sub Auto3()
Dim a As Shape, samm, algaeg, ring
Set a = Shapes("auto")
samm = Range("samm"):   algaeg = Timer()
For ring = 1 To Range("ringe")
   a.Left = 0:   Range("ring") = ring
   Do
      Shapes("auto").IncrementLeft samm
      Range("aeg") = Timer() - algaeg
      Paus 0.1
   Loop Until a.Left > Shapes("fin").Left
Next ring
End Sub
```

Animatsioon Hüpe

Kestus: 45 min

Tunni eesmärgid: animatsiooni loomine Exceli töölehel VBA abil - õpitu kordamine ja kinnistamine

Õpitulemused:

- õpilane oskab kasutada korduslauseid animatsiooni loomisel
- õpilane teab korraldust DoEvents ja oskab kasutada valmisprotseduuri paus() animatsiooni loomisel

Õpilaste eelteadmised ja -oskused: VBA akna ja VB redaktori kasutamine; teadmised animatsiooni loomise võimalustest MS Excelis

Tunniks vajalikud materjalid: tööleht **Hüpe**

Tunni käik:

- Ülesandega tutvumine, Juku tegevuste lahti-rääkimine.
- Protseduuride **hüpe** ja **koju** loomine. Ülesande teksti võiks kopeerida kommentaarideks programmi **hüpe**.
- Viitmuutuja loomine ja sidumine kujundiga:
  Dim J As Shape: Set J = Shapes("Juku")
- Käsud protseduuris koju: Juku pöördenurga määramine ja tema viimine hüppelauale. Proovimine.
- Paar esimest tegevust (hüppamine, liikumine laua servani) tehakse koos õpetajaga. Proovimine sammhaaval.
- Õpilased teevad ülesande lõpuni iseseisvalt. Iga valmis saanud etappi proovida kohe. Arutleda ühiselt, milline peaks olema Juku liikumise samm, pöördenurk, asukoht.
  NB! Kui Juku ujub torni taha (pöördenurk -90), on Juku laiuseks originaalkujundi kõrgus.
- Kui tunni lõpus on aega, võiks kasutusele võtta ühe või mitu nimega konstanti või lahtrit töölehel, et Juku liikumiskiirust (ehk ka pauside pikkust) oleks mugavam muuta.

```
Sub koju()
Dim J As Shape
Set J = Shapes("Juku")
J.Rotation = 0
J.Left = Shapes("laud").Left
J.Top = Shapes("laud").Top - J.Height
End Sub

Sub Paus(pp)
  Dim pl
  pl = Timer() + pp
  Do
    DoEvents
  Loop While Timer() < pl
End Sub
```

```
Sub hüpe()
Dim J As Shape, i
Set J = Shapes("Juku")
'Juku teeb paar hüpet
For i = 1 To 3
  J.IncrementTop -20
  Paus 0.1
  J.IncrementTop 20
  Paus 0.2
Next i
'liigub hoolaua vasemast servast paremale
Do Until J.Left > Shapes("laud").Left + _
                    Shapes("laud").Width
  J.IncrementLeft 10
  Paus 0.1
Loop
'"hüppab" saltosid tehes alla kuni jõuab veeni.
Do Until J.Top + J.Height / 2 > Shapes("vesi").Top
  J.IncrementLeft 5
  J.IncrementTop 10
  J.IncrementRotation 30
  Paus 0.1
Loop
'liigub vertikaalselt (pea allpool) põhjani
J.Rotation = 180
Do Until J.Top + J.Height >= Shapes("põhi").Top
  J.IncrementTop 5
  Paus 0.1
Loop
'pöörab ümber ja tõuseb pinnale
J.Rotation = 0
Do Until J.Top + J.Height / 2 < Shapes("vesi").Top
  J.IncrementTop -5
  Paus 0.1
Loop
'pöörab horisontaalasendisse ja ujub torni varju
J.Rotation = -90
Do Until J.Left < Shapes("torn").Left + _
                Shapes("torn").Width - J.Height
  J.IncrementLeft -5
  Paus 0.1
Loop
'peale väikest pausi ilmub torni.
Paus 1
Call koju
End Sub
```

Animatsioon Pallid

Kestus: 45 min

Tunni eesmärgid: *Shape*-objekti omaduste ja meetodite kordamine; alamprotseduuri koostamine ja kasutamine

Õpitulemused:

- õpilane mõistab muutujate kasutamisvõimalusi objekti omadustena
- õpilane oskab vormistada osa tegevusest alamprotseduurina ja kasutada seda erinevate algandmetega
- õpilane oskab avastada ja parandada vigu programmis.

Õpilaste eelteadmised ja -oskused: VB käskude ja redaktori tundmine; baasteadmised animatsiooni loomise võimalustest MS Excelis

Kordamisele tulevad mõisted: muutuja skoop, parameetrid ja argumendid

Õpitavad teemad: alamprotseduur (sub-protseduur)

Tunniks vajalikud materjalid: tööleht **Pallid** (kujundid liiguvad-veerevad töölehel etteantud piirkonnas).

Tunni käik:

- **Pallid** harjutuse 1 tutvustamine. Sarnaneb meritähe liikumisega Scratchi rakenduses Veemaailm.
- Kujundi (palli joonistamine); arutelu, millistes piirides võiks see liikuda, mis peaks olema liikumise samm ja pöördenurk.
- Protseduuri **veere** koostamine. **If**-lause tingimuses esialgu konstandid.
- Protseduuri töö testimine, vajadusel parandamine.
- Protseduurist veere koopia tegemine (**veere1**). Kujundi **kast** joonistamine.
- Horisontaal- ja vertkaalsuunalise sammu määramine (võib valida juhusliku suuna ja leida etteantud sammu pikkuse järgi sammx ja sammy).
- IF-lause muutmine, uue if-lause lisamine vertikaalsihis liikumise juhtimiseks.
- Protseduuri testimine, vajadusel parandamine. Võib lisada ka veel ühe või rohkem palli.
- Alamprotseduuri **liigu** koostamine. Pöördenurk ei pruugi igal sammul muutuda, võimaldame selle väärtust mitte anda. Parameeter dn on valikuline (*optional*) vaikeväärtusega 0.
- Protseduurist veere1 koopia tegemine (**veere2**). Liikumiskäskude asendamine protseduuri **liigu** poole pöördumisega.
- Protseduuri **liigu_kastis** koostamine, millele antakse parameetritena objekt, mis liigub, ja objekt, mille piirides liikuda tohib, lisaks liikumise sammud horisontaal- ja vertikaalsuunas. Kasutatakse protseduuri **liigu**, lisanduvad valikulaused sammude muutmiseks, kui liikuv objekt ei ole enam lubatud piirides.
- Uute kujundite lisamine töölehele. Protseduuri **liigu_kastis** kasutamine võimaldab kujundi liikuma panna vaid mõne käsu lisamisega programmi (algväärtused muutujatele – sammud horisontaal- ja vertikaalsihis – ja **liigu_kastis** kasutamine).

```
Sub veere()
Dim P As Shape, samm, nurk
samm = 10
nurk = 15
Set P = Shapes("pall")
Do
   P.IncrementLeft samm
   P.IncrementRotation nurk
   Paus 0.1
   If P.Left > 400 Or P.Left < 20
Then
      samm = -samm
      nurk = -nurk
   End If
Loop
End Sub
```

```
Sub veere1()
Dim P As Shape, K As Shape, sammx, sammy, nurk
Const samm = 12
nurk = Rnd() * 2 * (4 * Atn(1)) ' 2*pi radiaani
sammx = samm * Sin(nurk)
sammy = samm * Cos(nurk)

Set P = Shapes("pall")
Set K = Shapes("kast")
Do
   P.IncrementLeft sammx
   P.IncrementTop sammy
   P.IncrementRotation 30
   Paus 0.1
   If P.Left < K.Left Or P.Left + P.Width > K.Left + K.Width Then
      sammx = -sammx
   End If
    If P.Top < K.Top Or P.Top + P.Height > K.Top + K.Height Then
      sammy = -sammy
   End If
Loop
End Sub
```

```
Sub liigu(mis As Shape, dx, dy, Optional dn = 0)
   mis.IncrementLeft dx
   mis.IncrementTop dy
   mis.IncrementRotation dn
End Sub

Sub liigu_kastis(mis As Shape, kus, dx, dy)
Call liigu(mis, dx, dy)
If mis.Left < kus.Left Or mis.Left + mis.Width > kus.Left + kus.Width Then dx = -dx
If mis.Top < kus.Top Or mis.Top + mis.Height > kus.Top + kus.Height Then dy = -dy
End Sub

Sub veere2()
Dim P As Shape, P1 As Shape, P2 As Shape, V As Shape, kx, ky, px, py
Set P = Shapes("Pall")
Set P1 = Shapes("Pall1")
Set P2 = Shapes("Pall2")
Set V = Shapes("kast")
px = 8:  py = 5
p1x = 3:  p1y = -9
p2x = Rnd() * 20 - 10: p2y = Rnd() * 20 - 10
kx = 1: ky = 1

Do
   liigu_kastis P, V, px, py
   liigu_kastis P1, V, p1x, p1y
   liigu_kastis P2, Shapes("kast1"), p2x, p2y
   Paus 0.05
Loop
End Sub
```

Animatsioon Lennuk ja UFO

Kestus: 90 min (2x 45min)

Tunni eesmärgid: rakenduse planeerimine, animatsiooni loomine Exceli töölehel VBA abil – õpitu kordamine ja kinnistamine

Õpitulemused:

- õpilane oskab VBA abil luua animatsiooni Exceli töölehel
- õpilane oskab koostada alamprotseduure
- õpilane oskab kasutada alamprotseduuri erinevate algandmetega.

Õpilaste eelteadmised ja -oskused: VBA akna ja VB redaktori kasutamine; VB laused, kordused, valikud, protseduurid, viitmuutuja kasutamine; teadmised animatsiooni loomise võimalustest MS Excelis**.**

Tunniks vajalikud materjalid: tööleht **Lennuk ja Ufo**

Tunni käik:

- Ülesandega tutvumine. Rakenduse koostamise etappide meenutamine. Edaspidi võiks iseseisvaks tööks olla animatsiooni või mängu loomine.
- Edasise tegevusplaani koostamine (kirjalikult).
- Objektide paigutamine Exceli töölehele, kasutajaliidese kujundamine
- Alamprotseduuride (aseta, aseta_obj, vaheta, liigu) koostamine ja proovimine (seda võib teha ka hiljem, kui selliste tegevuste jaoks tekib vajadus). **NB!** Valikulised (*optional*) parameetrid ja nende vaikeväärtused.
- Protseduuride **start** ja **tulista** kavandamine (tegevused võib kirjutada kommentaaridena), arutlemine.
  - Milline on mängu algseis.
  - Millal lõpevad kordused, mis laused peaks kindlasti olema korduste sees.
  - Kuidas määrata liikumiskiirus, ufo kukkumise kiirus, tiibade pöörlemine. Võimalused: konstant protseduuris, nimega konstant, mis kirjeldatakse väljaspool protseduure, muutuja, väärtuse lugemine töölehelt. **NB!** Muutuja algaeg peaks olema kirjeldatud väljaspool protseduure, selle väärtust peab olema kasutatav protseduurides start ja tulista.
- Protseduuri **start** koostamine, proovimine iga tegevuse lisamise järel.
- Käsunupud töölehele, sidumine protseduuridega; nupp ja protseduur tegevuste katkestamiseks
- Protseduuri **tulista** koostamine, proovimine.

```
Sub aseta(obj As Shape, x, y, Optional nurk = "")
obj.Left = x
obj.Top = y
If Not nurk = "" Then obj.Rotation = nurk
End Sub

Sub aseta_obj(obj1 As Shape, obj2, Optional nurk = "")
obj1.Left = obj2.Left + obj2.Width / 2 - obj1.Width / 2
obj1.Top = obj2.Top + obj2.Height / 2 - obj1.Height / 2
If Not nurk = "" Then obj1.Rotation = nurk
End Sub
```

```
Sub vaheta(obj1 As Shape, obj2 As Shape)
aseta_obj obj2, obj1 'panna obj2 obj1 asukohta
obj1.Visible = False
obj2.Visible = True
End Sub

Sub liigu(obj As Shape, dx, dy, Optional nurk = "")
obj.IncrementLeft dx
obj.IncrementTop dy
If Not nurk = "" Then obj.IncrementRotation nurk
End Sub
```

```
Dim algaeg

Sub start()
'algseis
Range("tabas") = 0
Range("aeg") = 0
algaeg = Timer()
Call aseta_obj(Shapes("lennuk"), Shapes("len_alg"))
      'Shapes("ufo").Rotation = 0
Call aseta_obj(Shapes("ufo"), Shapes("ufo_alg"), 0)
'liikumine, kui lennuk või ufo jõuavad servani, tulevad uuesti
Do Until Range("aeg") >= Range("max_aeg")
   Shapes("lennuk").IncrementLeft 10   'lennuk liigub
   If Shapes("lennuk").Left > Shapes("ufo_alg").Left Then
      Call aseta_obj(Shapes("lennuk"), Shapes("len_alg"))
   End If
   Shapes("ufo").IncrementLeft -8      'ufo liigub
   If Shapes("ufo").Left < Shapes("len_alg").Left Then
      Call aseta_obj(Shapes("ufo"), Shapes("ufo_alg"))
   End If
   'kulunud aja näitamine, tiivad pöörlevad
   Shapes("tiivad").IncrementRotation 10
   Range("aeg") = Timer() - algaeg
   Call Paus(0.1)
Loop
Call MsgBox("Aeg läbi")
End Sub

Sub tulista()
Dim L As Shape: Set L = Shapes("lennuk")
'laseri (joone) näitamine lennuki ees
aseta Shapes("laser"), L.Left + L.Width, L.Top + L.Height / 2
Shapes("laser").Visible = True
Paus 0.05
Shapes("laser").Visible = False
'kui laser puudutab ufot
If Not On_Puude(Shapes("laser"), Shapes("ufo")) Then Exit Sub
Range("tabas") = Range("tabas") + 1 'muuta tabamiste arvu
Do Until On_Puude(Shapes("ufo"), Shapes("maa"))
   'ufo kukub pööreldes alla, lennuk lendab edasi
   L.IncrementLeft 10   'lennuk liigub
   If L.Left > Shapes("ufo_alg").Left Then
      Call aseta_obj(L, Shapes("len_alg"))
   End If
   liigu Shapes("ufo"), -8, 10, -20
   'möödunud aja näitamine, tiivad pöörlevad
   Shapes("tiivad").IncrementRotation 10
   Range("aeg") = Timer() - algaeg
   Call Paus(0.1)
Loop
’plahvatus
vaheta Shapes("ufo"), Shapes("tuli")
Paus 0.05
vaheta Shapes("tuli"), Shapes("ufo")
’ufo algseisu
Shapes("ufo").Fill.ForeColor.SchemeColor = Rnd() * 80
aseta_obj Shapes("ufo"), Shapes("ufo_alg"), 0
End Sub
```

VBA. Tekstiavaldised

Kestus: 45 min

Tunni eesmärgid: VB tekstifunktsioonide ja **for**-korduslause tutvustamine

Õpitulemused:

- õpilane mõistab for-korduslause olemust ja oskab seda kasutada
- õpilane teab mõningaid VB tekstifunktsioone (len, mid jm) ning sidurdamistehet
- õpilane oskab koostada VB funktsioone ja kasutada neid avaldistes väärtuste leidmisel

Õpilaste eelteadmised ja -oskused: VBA akna ja VB redaktori kasutamine; baasteadmised VBA rakenduste loomise vahenditest MS Excelis.

Õpitavad teemad: juhtmuutujaga kordus, sidurdamine

Tunniks vajalikud materjalid: tööleht **Tekstiavaldised**

Tunni käik:

- VB funktsioonide koostamise ja kasutamise meenutamine. Neid funktsioone kasutatakse töölehel, seega peaks programmi tekstid asuma üldmoodulis.
- Harjutus töölehel **Tekstiavaldised**. Tekstifunktsioonide (*len*, *mid*, *left* jm) tutvustamine
- Funktsiooni **loenda** alguslausete koostamine; katkestuspunkti lisamine ja parameetrite väärtuste demonstreerimine õpilastele.
- For- korduslause tutvustamine
- Funktsiooni **loenda** koostamine tegevusskeemi abil
- Funktsiooni proovimine töölehel, vajadusel parandamine.
- Funktsiooni **mitmes** alguslausete koostamine.
- Lause **Exit** tutvustamine (Exit For, Exit Function).
- Funktsiooni **mitmes** koostamine tegevusskeemi abil, funktsiooni proovimine, vajadusel parandamine.
- Tekstide ühendamise (sidurdamistehe) tutvustamine.
- Funktsiooni **tagurpidi** koostamine etteantud algoritmi alusel, proovimine, vajadusel parandamine.
- Lisaülesanne

```
Function loenda(tekst As String, symb As String) As Integer
Dim s As String, mitu As Integer, L As Integer, J
L = Len(tekst)
For J = 1 To L Step 1   'Step 1 võib puududa
   s = Mid(tekst, J, 1)
   If s = symb Then mitu = mitu + 1
Next J
loenda = mitu
End Function

Function mitmes(tekst As String, symb As String) As Integer
Dim s As String, mitu As Integer, J As Integer
mitu = Len(tekst)
For J = 1 To mitu   'Step 1 võib puududa
   s = Mid(tekst, J, 1)
   If s = symb Then
      mitmes = J
      Exit Function
   End If
Next J
mitmes = 0
End Function
```

```
Function tagurpidi(tekst As String) As String
Dim txt As String, s As String, mitu As Integer, J As Integer
For J = Len(tekst)To 1 Step -1
      s = Mid(tekst, J, 1)
      txt = txt & s
Next J
tagurpidi = txt
End Function
```

VBA. Lahtriploki omadused

Kestus: 45 min

Tunni eesmärgid: Lahtriploki (*Range*-objekti) omaduste *CurrentRegion*, *Rows*, *Columns*, *Cells, Select* jm tutvustamine ja kasutamine

Õpitulemused:

- õpilane teab mõningaid lahtri ja lahtriploki omadusi ja oskab neid kasutada lahtriploki suuruse ja asukoha määramiseks
- õpilane mõistab for-korduslause olemust ja oskab kasutada for-korduse juhtmuutujat rea/veeru numbrina tabeli lahtrile viitamisel

Õpilaste eelteadmised ja -oskused: VBA akna ja VB redaktori kasutamine; baasteadmised VBA rakenduste loomise vahenditest MS Excelis; for-korduslause mõistmine

Õpitavad teemad: viitmuutuja, lahtriploki omadused

Tunniks vajalikud materjalid: tööleht **Lahtriploki omadused**

Tunni käik:

- Lahtriploki omaduste tutvustamine
- tööleht **Lahtriploki omadused**. Ülesande tutvustamine, võimalike tegevuste kirjeldamine
- Nime **algus** määramine töölehel. Alates sellest lahtrist võiks klaviatuurilt sisestada mingi hulga (3 x 4) väärtusi.
- Viitmuutuja kasutamine võimaldab lühemaid lauseid (Set prk = Range("algus").CurrentRegion); meetodi *Select* abil saab õpilastele näidata viidatavat lahtrit või lahtriplokki (prk.Select).
- Ridade ja veergude arvu (ridu, veerge) leidmine, kontroll.
- Korduslausete kirjutamine: rea number (rida) muutub 1 kuni ridu (sammuga 1), ühe rea piires tuleb vaadata kõiki lahtreid, st veeru number (veerg) muutub 1 kuni veerge. **NB!** teine kordus on tervenisti esimese sees.
- For- korduslause juhtmuutujat kasutatakse rea/veeru indeksina tabeli lahtrile viitamisel - prk.Cells(rida, veerg). **NB! .Cells** võib puududa. Meetodi *Select* abil saab näidata, kuidas (millises järjekorras) lahtreid vaadatakse.
- Lahtri taustavärvi saab määrata objekti Range omaduse *Interior.ColorIndex* väärtusena (täisarv 0...56), täitevärvist loobumiseks kasutada Excelis määratud konstanti *xlColorIndexNone*. Protseduuri täiendamine.
- Protseduuri testimine, täitmine sammhaaval.
- Väärtuste kirjutamiseks teise piirkonda võiks korduslausetest teha koopia, selle asemel võib need mõned laused lisada ka värvimise juurde.
- Ümberkirjutamisel on vaja loendurit jrk (algul 0, uue sobiva väärtuse leidmisel liidetakse 1), mida saab kasutada kirjutatava väärtuse reanumbri määramisel.

```
Sub värvimine()
Dim prk As Range, ridu, veerge, rida, veerg, värv, jrk
Set prk = Range("algus").CurrentRegion
ridu = prk.Rows.Count          'ridade arv
veerge = prk.Columns.Count     'veergude arv
          'värvimine ja 'ümberkirjutamine
värv = Rnd() * 56
jrk = 0
For rida = 1 To ridu
   For veerg = 1 To veerge
      If prk.Cells(rida, veerg) > 0 Then
         prk.Cells(rida, veerg).Interior.ColorIndex = värv
         jrk = jrk + 1
         prk(jrk, veerge + 2) = prk(rida, veerg)
      End If
   Next veerg
Next rida
End Sub
```

```
Sub kustuta()
Dim prk As Range
Set prk = Range("algus").CurrentRegion
veerge = prk.Columns.Count
prk(1, veerge + 2).CurrentRegion.ClearContents
prk.Interior.ColorIndex = xlColorIndexNone
prk.ClearContents
End Sub
```

```
Sub kustu()
Dim prk As Range
Set prk = ActiveCell.CurrentRegion
prk.Interior.ColorIndex = xlColorIndexNone
prk.ClearContents
End Sub
```

VBA. Lahtriplokk parameetrina

Kestus: 45 min

Tunni eesmärgid: lahtriploki omaduste ja for-korduse kordamine ja kinnistamine

Õpitulemused:

- õpilane teab mõningaid lahtri ja lahtriploki omadusi ja oskab neid kasutada lahtriploki suuruse ja asukoha määramiseks
- õpilane mõistab for-korduslause olemust ja oskab kasutada for-korduse juhtmuutujat rea/veeru numbrina tabeli lahtrile viitamisel

Õpilaste eelteadmised ja -oskused: VBA akna ja VB redaktori kasutamine; VB funktsioonide koostamine ja kasutamine; lahtriploki omadused; for-korduse juhtmuutuja kasutamine rea/veeru numbrina tabeli lahtrile viitamisel, algoritmide summa, loendur, min/max teadmine

Tunniks vajalikud materjalid: tööleht **Lahtriplokk parameetrina**

Tunni käik:

- Ülesanne töölehel **Lahtriplokk parameetrina**. Keskmise leidmise võimalike tegevuste kirjeldamine; varem Scratchis vaadeldud algoritmi meenutamine.
- Funktsioon **Keskmine1**, parameetri **prk** kirjeldamine lahtriplokina (Range), ridade ja veergude arv.
- Katkestuspunkti lisamine, funktsiooni kasutava valemi kirjutamine töölehele.
- Funktsiooni koostamine õpetaja juhendamisel (vt. algoritm ja eelmine harjutus)
- Funktsiooni kasutamine töölehel, kontroll Exceli funktsiooni *Average* abil.
- Teha koopia funktsioonist (**Keskmine2**). Näidata, et for-kordust saaks siin kasutada ka teisiti (iga elemendi jaoks grupis ...). Proovimine töölehel.
- Teha koopia funktsioonist (**Pos_Kesk**). Õpilased lisavad iseseisvalt valikulaused. Proovimine
- Tutvumine ülesandega suurima väärtuse leidmiseks varem Scratchis vaadeldud algoritmi meenutamine.
- Õpilased koostavad funktsiooni õpetaja juhendamisel (vt. sobivad *ColorIndex* väärtused).
- Funktsiooni proovimine töölehel.

```
Function Keskmine1(prk As Range)
Dim summa, ridu, veerge, i, j
ridu = prk.Rows.Count
veerge = prk.Columns.Count
summa = 0
For i = 1 To ridu Step 1
   For j = 1 To veerge 'Step 1 võib puududa
      summa = summa + prk.Cells(i, j)
   Next j
Next i
Keskmine1 = summa / (ridu * veerge)
End Function

Function Pos_Kesk(prk As Range)
Dim summa, mitu, lahter
summa = 0:   mitu = 0
For Each lahter In prk
   If lahter > 0 Then
      summa = summa + lahter
      mitu = mitu + 1
   End If
Next lahter
If mitu = 0 Then
    Pos_Kesk = 0
Else
    Pos_Kesk = summa / mitu
End If
End Function
```

```
Function Keskmine2(prk As Range)
Dim summa, mitu, lahter
summa = 0
mitu = 0
For Each lahter In prk
   summa = summa + lahter
   mitu = mitu + 1
Next lahter
Keskmine2 = summa / mitu
End Function
Function v_maks(prk As Range, värv)
Dim ridu, veerge, i, j, max
ridu = prk.Rows.Count
veerge = prk.Columns.Count
max = -10 ^ 10
For i = 1 To ridu
   For j = 1 To veerge
      If prk(i, j).Interior.ColorIndex = värv
         And prk(i, j) > max Then  max = prk(i, j)
   Next j
Next i
If max = -10 ^ 10 Then
   v_maks = "pole"
Else
   v_maks = max
End If
End Function
```

Rehmaatika

Kestus: 45 min

Tunni eesmärgid: VBA toega rakenduse koostamine Exceli töölehel, varemõpitu kordamine ja kinnistamine

Õpitulemused:

- õpilane teab mõningaid lahtriploki omadusi ja oskab neid kasutada
- õpilane mõistab for-korduslause olemust ja oskab kasutada for-korduse juhtmuutujat rea/veeru numbrina tabeli lahtrile viitamisel

Õpilaste eelteadmised ja -oskused: VBA akna ja VB redaktori kasutamine; lahtriploki omadused; for-korduse kasutamine; varem Scratchis tehtud samalaadse ülesande algoritmi mõistmine.

Tunniks vajalikud materjalid: tööleht **Rehmaatika**

Tunni käik:

- Harjutus **Rehmaatika**. Ülesandega tutvumine, varem Scratchis vaadeldud samalaadse rakenduse meenutamine
- Kasutajaliidese kujundamine: lahtrid töölehel (min, max, mitu, alg, õigeid).
- Koostada protseduuri **Tee_uued** algus, arutleda ühiselt, millistest tegevustest peaks see koosnema (kirjutada kommentaaridena).
- Õpilased koostavad protseduuri õpetaja juhendamisel
- Protseduuri testimine erinevate algandmetega (vajadusel sammhaaval)
- Koostada protseduuri **Kontroll** algus, arutleda ühiselt, millistest tegevustest peaks see koosnema (kirjutada kommentaaridena).
- Õpilased koostavad protseduuri õpetaja juhendamisel
- Protseduuri testimine erinevate algandmetega (vajadusel sammhaaval)

```
Sub Tee_uued()
Dim jrk, tabel As Range, a, b, n, tehe, tekst
' vanade andmete kustutamine
Range("alg").CurrentRegion.Offset(1, 0).Clear
Range("õigeid").ClearContents
'andmed tööhelt
Set tabel = Range("alg")
n = Range("Mitu")
min = Range("min")
max = Range("max")
'tehete kirjutamine tabelisse (kordus tehete arv korda)
For jrk = 1 To n
   tabel(jrk + 1, 1) = jrk
   ‘liidetavate genereerimine
   a = Int(Rnd() * (max - min + 1) + min)
   b = Int(Rnd() * (max - min + 1) + min)
   ’tehe ja õige vastus töölehele
   tehe = Rnd()
   If tehe < 0.5 Then
      tabel(jrk + 1, 2) = a & " + " & b & " = "
      tabel(jrk + 1, 4) = a + b
   Else
      tabel(jrk + 1, 2) = a + b & " - " & b & " = "
      tabel(jrk + 1, 4) = a
   End If
Next jrk
'tabeli kujundamine
tabel(1, 4).EntireColumn.Hidden = True
tabel(2, 2).Resize(n, 1).HorizontalAlignment = xlRight
Range("alg").CurrentRegion.Borders.Weight = xlThin
End Sub
```

```
Sub Kontroll()
Dim prk As Range, jrk, mitu, õigeid
'andmete määramine
Set prk = Range("alg").CurrentRegion
mitu = prk.Rows.Count - 1
õigeid = 0
'kontroll (kordus tehete arv korda)
For jrk = 1 To mitu
   ‘kui vastus õige, loendada,
    ‘kui vale, siis värvida
   If prk(jrk + 1, 3) = prk(jrk + 1, 4) Then
      õigeid = õigeid + 1
   Else
      prk(jrk + 1, 3).Interior.ColorIndex = 3
   End If
Next jrk
'õigete vastuste arv töölehele
Range("õigeid") = õigeid
End Sub
```

Sündmusprotseduurid

Kestus: 90 min (2 x 45 min)

Tunni eesmärgid: töölehe sündmusprotseduuride **Worksheet_Change** ja **Worksheet_SelectionChange** tutvustamine.

Õpitulemused: õpilane teab mõningaid sündmusprotseduure ja oskab neid kasutada

Õpilaste eelteadmised ja -oskused: VBA akna ja VB redaktori kasutamine; lahtriploki omadused

Tunniks vajalikud materjalid: tööleht **Sündmusprotseduurid**

Tunni käik:

- Sündmusprotseduuride **Worksheet_Change** ja **Worksheet_SelectionChange** tutvustamine. Sündmusprotseduuri käivitumise demonstreerimiseks võib kasutada näiteks MsgBox "sündmuse nimi" või katkestuspunkte.
- Töölehe ülesannetega tutvumine.
- **Esimene ülesanne**: protseduuri koostab õpetaja iga käsku demonstreerides ja oma tegevust selgitades.
- **Teine ülesanne**: protseduuri püüavad õpilased koostada õpetaja juhendamisel. Testida iga tingimuse lisamise järel.
- **Kolmas ülesanne**: õpilased koostavad ja testivad kõigepealt protseduuri **arvuta_kast**().
- Sündmusprotseduuri **Worksheet_Change** koostamisel näitab õpetaja, et sündmusprotseduur käivitub uuesti pindala kirjutamisel töölehele (vt ka katkestusrežiimis View – Call Stack ja objekti **Application** omadus **EnableEvents**).
- Funktsiooni **On_Sees** kasutamine väldib tegevuse käivitumist, kui töölehel peaks muudetama muid väärtusi peale ristküliku mõõtmete.

```
Private Sub Worksheet_SelectionChange(ByVal Target As Range)
If Target.Column = 1 And Target.Row > 1 Then
   If IsEmpty(Target.Value) And Not IsEmpty(Target.Cells(0, 1)) Then
      Target.Value = Now()
   End If
End If
End Sub

Private Sub Worksheet_Change(ByVal Target As Range)
If IsEmpty(Target.Cells(1, 1)) Then
   Target.Interior.ColorIndex = 0
ElseIf IsDate(Target.Value) Then
   Target.Interior.ColorIndex = 34
ElseIf IsNumeric(Target) Then
   Target.Interior.ColorIndex = 36
Else
   Target.Interior.ColorIndex = 38
End If
End Sub
```

```
Sub arvuta_kast()
Dim a, b
a = Range("a"):  b = Range("b")
Range("Pindala") = a * b
Range("Ümbermõõt") = 2 * (a + b)
Shapes("kast").Width = a * 72 / 2.54
Shapes("kast").Height = b * 72 / 2.54
Shapes("kast").Fill.ForeColor.SchemeColor = Rnd() * 56
End Sub
```

```
Private Sub Worksheet_Change(ByVal Target As Range)
'If Target.Address = Range("a").Address Or
'   Target.Address = Range("b").Address Then
If On_Sees(Target, Range("mõõdud")) Then
   Application.EnableEvents = False
   arvuta_kast
   Application.EnableEvents = True
End If
End Sub
```

Tunnikavad. Python

Lisamooduli (Python või VBA) ülesannetes on eeldatud, et õppur juba oskab programmeerida (Scratchis). Esimesed harjutusülesanded on teise keele süntaksist ja erisustest ning Pythoni keskkonna (IDLE) võimalustest ülevaate saamiseks. Kasutatud on kas väga lihtsaid või varem Scratchis tehtud rakendusi, mille algoritmid on varasemast tuttavad.

Põhi- ja lisamoodul ei pea olema üksteisest rangelt eraldatud. Lisamooduli esimesi ülesandeid võiks kasutada juba enne põhimooduliga (Scratchiga) lõpetamist. Näiteks võiks Pythoni süntaksit näidata ja mõned Scratchis tehtud tuttavad ülesanded selles realiseerida juba enne loendite tutvustamist. Nii on lihtsam kahe vahendi võimalusi võrrelda. Edaspidi saab siis demonstreerida töötavat algoritmi Scratchis ja lasta õppuritel iseseisvalt teha samalaadne rakendus Pythonis.

Et tegemist on programmeerimise algõpetusega, mitte programmeerimiskeele Ptyhon õppimisega, ei ole siin tutvustatud Pythoni paljusid spetsiifilisi võimalusi. Oluline on programmeerimise põhimõistete omandamine. Seepärast võib mõne ülesande puhul seada isegi piiranguid keele võimaluste kasutamise osas.

Pythoni kasutajaliides. Avaldised, muutujad

Kestus: 45 min

Tunni eesmärgid: Pythoni kasutajaliidese ja lihtsamate avaldiste tutvustamine; skripti koostamine.

Õpitulemused:

- õpilane teab Pythoni kasutajaliidese võimalusi
- õpilane oskab koostada Pythonis arv-, tekst- ja loogikaavaldisi ja omistamislauseid
- õpilane oskab koostada lihtsa Pythoni skripti, seda käivitada ja salvestada faili

Õpilaste eelteadmised ja -oskused: mõistete avaldis, muutuja, omistamine teadmine

Tunniks vajalikud materjalid: tööleht **Python. Avaldised**

Tunni käik:

- Näide 'Arvamine', erinevate keelte käskude võrdlemine.
- Pythoni võimaluste ja ajaloo tutvustamine (vt. slaidid)
- Pythoni kasutajaliidese (IDLE) demonstreerimine ja tutvustamine
- Töölehe tutvustamine
- Lihtsate avaldiste koostamine ja testimine, kindlasti proovida ka ilmselt vigaseid avaldisi, et näha erinevaid veateateid (Scratchis ei olnud midagi sellist).
- Omistamislaused, muutuja tüüp. (vt. ka slaidid) Proovida.
- Editori võimaluste tutvustamine.
- Faili salvestamine. NB! laiend **.py**
- Lihtsa skripti koostamine ja käivitamine.

Pythoni laused. Moodulid

Kestus: 45 min

Tunni eesmärgid: Pythoni lausete tutvustamine, programmimoodulid

Õpitulemused:

- õpilane teab Pythoni lihtlauseid ja oskab avaldistes kasutada funktsioone
- õpilane tunneb Pythoni liitlausete süntaksit ja teab mõningaid valiku- ja korduslauseid

Õpilaste eelteadmised ja -oskused: mõisted avaldis, muutuja, omistamine; Pythoni kasutajaliides

Tunniks vajalikud materjalid: tööleht **Pythoni laused. Moodulid**

Tunni käik:

- Töölehe tutvustamine.
- Omistamis- ja print-laused töölehe näites.
- Näite (võrdse pindalaga ristkülik ja ring) kopeerimine programmiredaktorisse ja käivitamine.
- teistes moodulites olevate funktsioonide ja konstantide kasutamine; käsud *import ...* ja *from ... import* . Funktsiooni dir() saab kasutada nimeruumi (kasutusel olevad nimed: muutujad, objektid, funktsionid jm) demonstreerimiseks erinevate import käskude puhul
- Moodulitest ülevaate saamiseks vt. abiinfo (*Library Reference* Pythoni dokumentatsioonis)
- Asendada *import math* käsuga *from math import ** ja teha vastavad muudatused tekstis. Testida programmi uuesti.
- Lühike ülevaade liitlausetest (valik ja kordus) Pythonis (vt. slaidid).
- Kopeerida näide (arvude keskmised) programmiredaktorisse ja käivitada.
- Vigade parandamine:
  - vale taane lausel np += 1 (selgitada, et edaspidi oleks taande tegemiseks mõistlik kasutada tabulaatorit, tabulaatori suuruse määramiseks vt. Options – Configure IDLE)
  - vale taane lausel else: (redaktor näitab viga järgmises lauses)
  - vale taane viimasel if-lausel
  - arvude arv n peab olema täisarv - viga tekib 4. reas, parandada lauset n = int(input("Mitu arvu "))

Python. Lihtsa skripti koostamine

Kestus: 45 min

Tunni eesmärgid: Õpilased koostavad lihtsa skripti iseseisvalt

Õpitulemused:

- õpilane teab Pythoni lihtlauseid ja oskab avaldistes kasutada funktsioone
- õpilane oskab koostada Pythonis lihtsa skripti
- õpilane oskab kasutada valikulaused
- õpilane oskab avastada ja parandada vigu skriptis

Õpilaste eelteadmised ja -oskused: avaldis, funktsioon, muutuja, omistamine; Pythoni kasutajaliides

Tunniks vajalikud materjalid: tööleht **Python**. **Ideaal**

Tunni käik:

- töölehe tutvustamine.
- Scratchis loodud samalaadse rakenduse meenutamine (kuidas sisestati andmed, kuhu paigutati tulemused).
- Õpilased koostavad skripti etteantud valemite järgi samm-sammult. Tulemuse testimiseks võiks skripti lisada print-lauseid (algandmed, abimuutujad), mis hiljem eemaldatakse.
- Õpilased abistavad üksteist vigade leidmisel ja parandamisel.

Märkused:

Võimalikud vead:

- moodul **math** sidumata
- vale pöördumine funktsiooni poole:
  - olenevalt import-lausest mooduli nimi puudu või ülearu
  - vale argumentide arv või tüüp
- **NB!** muutujate ja funktsioonide nimedes eristatakse suur- ja väiketähti
- **NB!** tekstikonstant jutumärkides (soo väärtuse naine/mees sisestamisel)

Python. Funktsioonide loomine

Kestus: 45 min

Tunni eesmärgid: funktsioonide koostamine ja kasutamine Pythonis

Õpitulemused:

- õpilane teab Pythoni lihtlauseid ja oskab avaldistes kasutada funktsioone
- õpilane oskab koostada Pythonis lihtsa skripti ja seda kasutada
- õpilane oskab avastada ja parandada vigu oma skriptis

Õpilaste eelteadmised ja -oskused: avaldis, funktsioon, muutuja, omistamine; Pythoni kasutajaliides

Tunniks vajalikud materjalid: tööleht **Python. Funktsioonid**

Tunni käik:

- töölehe **Funktsioonid** tutvustamine.
- Funktsiooni koostamise kirjeldamine õpetaja poolt (vt ka slaidid).
- Näitefunktsioonide proovimine.
- Arutlemine rasvaprotsendi ja hinnangu andmise funktsioonide üle. Mille poolest erineb rakendus eelmisest ülesandest?
- Õpilased koostavad skripti etteantud valemite järgi samm-sammult. Tulemuse testimiseks võiks skripti lisada print-lauseid (algandmed, abimuutujad), mis hiljem eemaldatakse.
- Õpilased abistavad üksteist vigade leidmisel ja parandamisel.

Märkused:

Võimalikud vead:

- moodul **math** sidumata
- vale pöördumine funktsiooni poole:
  - olenevalt import-lausest mooduli nimi puudu või ülearu
  - vale argumentide arv või tüüp
- NB! muutujate ja funktsioonide nimedes eristatakse suur- ja väiketähti
- NB! tekstikonstant jutumärkides (soo väärtuse naine/mees sisestamisel)
- vajalikku väärtust ei tagastata – puudub return
- pöördumisel oma funktsiooni poole vale argumentide arv või tüüp.

Python. Kordused

Kestus: 45 min

Tunni eesmärgid: Pythoni korduslausete tutvustamine

Õpitulemused: õpilane teab ja oskab kasutada Pythoni korduslauseid

Õpilaste eelteadmised ja -oskused: avaldis, muutuja, omistamine; Pythoni kasutajaliides

Tunniks vajalikud materjalid: tööleht **Python. Kordused**

Tunni käik:

- **for**-korduse tutvustamine (vt. slaidid).
- for-korduse näidete tutvustamine
- näiteskriptde proovimine ja modifitseerimine
- **while**-korduse tutvustamine (vt. slaidid).
- **while** -korduse näidete tutvustamine
- näiteskriptide proovimine ja modifitseerimine

Python. Rehmaatika

Kestus: 45 min

Tunni eesmärgid: skripti koostamine etteantud algoritmi alusel

Õpitulemused:

- õpilane oskab Pythonis kasutada valikulaused ja korduslauseid
- õpilane oskab koostada skripti etteantud algoritmi alusel
- õpilane oskab avastada ja parandada vigu skriptis

Õpilaste eelteadmised ja -oskused: avaldis, muutuja, omistamine; kordus- ja valikulaused

Tunniks vajalikud materjalid: tööleht **Python**. **Rehmaatika**

Tunni käik:

- Töölehe Rehmaatika ja ülesande tutvustamine. Selline ülesanne on varem Scratchis realiseeritud.
- Arutelu: millised tegevused peaks olema selles skriptis
- Arutelu: muutujad, min ja max on standardfunktsioonid, sellise nimega muutujaid on võimalik kasutada, siis ei saa kasutada neid funktsioone
- õpilased koostavad etteantud algoritmi alusel Pythoni skripti õpetaja juhendamisel.
- Skripti testimine ja vigade parandamine

```
import random, time

def Tee_tase(t):
   if (t == 1):   return 1, 10, 2
   elif(t == 2): return 10, 20, 5
   else:         return 20, 50, 10

def rehmaatika():
   n = int (input("Mitu ülesannet? =>" ))
   tase = int (input(" Tase: 1, 2, 3 =>"))
   mini, maxi, p = Tee_tase(tase)
   punkte = 0
   algaeg = time.clock()
   for k in range(n):
      a = random.randint(mini, maxi)
      b = random.randint(mini, maxi)
      c = a + b
      tehe = random.randint(1, 2)
      if tehe == 1:
        teade = str(a) + " + " + str(b) + " = "
        tulem = c
      else :
        teade = str(c) + " - " + str(b) + " = "
        tulem = a
      vastus = int(input(teade))
      if vastus == tulem:
         punkte += p
      else:
         print ("Vale!")
   aeg = time.clock() - algaeg
   print ("Punkte", punkte, "Aeg:", round(aeg, 2), "sek")

print ("Programm võimaldab harjutada liitmist ja lahutamist.")
print ("Kolm taset. Erinevad arvude suurused")
print ("ja  punktide arvud tehte kohta.")
print ("tase 1: min=1, max=10, punkte=2")
print ("tase 2: min=10, max=20, punkte=5")
print ("tase 3: min=20, max=50, punkte=10")
rehmaatika()
```

Python. Tekstiavaldised

Kestus: 45 min

Tunni eesmärgid: sümbolite eraldamine tekstist, tekstide ühendamine

Õpitulemused:

- õpilane teab võimalusi tekstist sümboli(te) eraldamiseks
- õpilane oskab koostada tekstiavaldisi
- õpilane oskab koostada skripti etteantud algoritmi alusel
- õpilane oskab avastada ja parandada vigu skriptis

Õpilaste eelteadmised ja -oskused: avaldis, muutuja, omistamine; kordus- ja valikulaused

Tunniks vajalikud materjalid: tööleht **Python**. **Tekstiavaldised**

Tunni käik:

- Töölehe **Tekstiavaldised** tutvustamine.
- teksti pikkuse (sümbolite arvu leidmine), proovimine käsuaknas.
- tekstist sümbolite eraldamine, proovimine käsuaknas.
- Funktsiooni Tagurpidi1 koostamine etteantud algoritmi alusel.
- Funktsiooni kasutamine käsuaknas, võimalike vigade avastamine ja parandamine.
- Funktsiooni **Tagurpidi_sõnad** koostamine etteantud algoritmi alusel.
- Funktsiooni kasutamine käsuaknas, võimalike vigade avastamine ja parandamine.

Märkused:

tekstavaldis.strip() – eemaldab teksti lõpust tühikud ja erisümbolid.

tekstavaldis.split() – tekst jagatakse sõnadeks, moodustades loendi.

```
def tagurpidi(t1):
    t2 = ""
    for j in range(len(t1)-1,-1,-1):
        t2 += t1[j]
    return t2

def tagurpidi2(t1):
    t2 = t3 = ""
    for j in range(len(t1)):
        s = t1[j]
        if s != " ":
            t3 += s
        else:
            if t2=="": t2 = t3
            else: t2 = t3 + " " + t2
            t3 = ""
    if t2=="": t2 = t3
    else: t2 = t3 + " " + t2
    return t2
```

Python. Algoritme loenditega

Kestus: 45 min

Tunni eesmärgid: loendite kasutamine Pythonis

Õpitulemused:

- õpilane oskab luua loendi Pythonis
- õpilane oskab lugeda väärtusi loendist
- õpilane oskab lisada ja kustutada loendi elemente
- õpilane oskab avastada ja parandada vigu skriptis

Õpilaste eelteadmised ja -oskused: avaldis, muutuja, omistamine; kordus- ja valikulaused

Tunniks vajalikud materjalid: tööleht **Python**. **Algoritme loenditega**

Tunni käik:

- Loendi loomine ja kasutamine Pythonis. Näited (vt. ka slaidid).
- Õpilased proovivad käske loendisse väärtuste lisamiseks ja eemaldamiseks.
- Töölehe ülesannete tutvustamine.
- Aritmeetilise keskmise leidmise algoritmi realiseerimine Pythoni skriptina.
- Testimine, võimalike vigade avastamine ja parandamine.
- Aritmeetilise keskmise leidmine Pythoni funktsioonina.
- Testimine, võimalike vigade avastamine ja parandamine.
- Suurima väärtuse ja selle järjenumbri saab leida ka lühemalt – hinnad.index(max(hinnad)) – seda võib õpilastele näidata. Tulemuse võiks siiski leida etteantud algoritmi järgi; kui see tundub õpilastele nüüd ebavajalik, võib anda näiteks lisatingimuse, et otsitav hind peab olema mingist väärtusest väiksem.
- Suurima väärtuse ja selle järjenumbri leidmise realiseerimine Pythoni funktsioonina.

```
def keskV(V):
    n =len(V)
    S = 0
    for i in range(n):
      S = S + V[ i]
    return S/n
```

```
def suurim(v):
    maxi = v[0]
    nr = 0
    for i in range(len(v)):
        if v[i]> maxi:
            maxi = v[i]
            nr = i
    return maxi, nr
```

```
def tee_pomod(T, V, arv):
    m = len(V)
    TP = [ ]
    for i in range(m):
        if V[i] > arv: TP.append(T[i])
    return TP
```

Python. Sõnastik

Kestus: 45 min

Tunni eesmärgid: loendite kasutamine Pythonis, väärtuste lugemine tekstifailist

Õpitulemused:

- õpilane oskab lugeda väärtusi loendist, lisada ja kustutada loendi elemente
- õpilane teab, kuidas lugeda loendisse andmeid tekstifailist
- õpilane oskab avastada ja parandada vigu skriptis

Õpilaste eelteadmised ja -oskused: Pythoni laused ja kasutajaliides, etteantud algoritmi mõistmine

Tunniks vajalikud materjalid: tööleht **Python**. **Sõnastik**, tekstifailid **eesti.txt** ja **inglise txt**

Tunni käik:

- Töölehe **Sõnastik** tutvustamine. Sellised ülesanded on Scratchis varem tehtud. Scrathis ei olnud võimalik lugeda sõnu failidest loenditesse programselt.
- Funktsiooni **loefail** võib õpilastele anda valmiskujul, kommenteerides iga lauset.
- Alustada funktsioonist **test_pea** (esialgu võib olla ka skriptina) loendite moodustamseks (print-käsuga saab vaadata , millised väärtused funktsioon loefail hankis) ja kasutajalt tõlkesuuna küsimiseks.
- Funktsioonile test antakse argumentidena küsitavate sõnade arv m ja loendid tõlkesuunaga määratud järjestuses. Esmaversioonis leitakse i väärtus juhuarvuna vahemikus 0...len(V1)-1.
- Testimine, võimalike vigade avastamine ja parandamine.
- Et küsitavad sõnad ei korduks, võiks koostada loendi erinevatest juhuarvudest, vt funktsioon **tee_num**.
- Funktsiooni **tõlgi** koostamine. Proovimine käsuaknas.
- Funktsiooni **tõlkimine** koostamine ja testimine.

Kui on aega ja õpilastel huvi, võib leida sõna vaste teisest loendist ka avaldisega eng[est.index(sõna)]. Veateate vältimiseks (kui etteantud sõna loendis ei esine) tuleks siis selgitada ka *try* ja *except ValueError* kasutamist.

```
import random

def loefail(fn):
  fail = open(fn)
  loend = [ ]
  for rida in fail:
     loend.append(rida.strip())
  fail.close()
  return loend

def test_pea():
    est = loefail("eesti.txt")
    eng = loefail("inglise.txt")
    n = len (est)
    for i in range(n):
       print(est[i],"\t", eng[i])
    v =int(input ("1 - est=>eng, 2 - eng=>est " ))
    if v == 1:    ov = test (5, est, eng)
    else:         ov = test (5, eng, est)
    print ("Õigeid vastuseid oli:", ov)

def test(m, V1, V2):
 n = len(V1)
 nv = tee_num(m, 0, n-1);   print (nv)
 ov = 0
 for k in range(m):
    i = nv[k];  print (V1[i], end = " ");  vastus = input()
    if vastus == V2[i]: ov += 1
    else: print ("Vale!")
 return ov
```

```
def tee_num(m, mini, maxi):
# Teeb loendi: m erinevat arvu [mini, maxi]
 V = []
 for i in range(min(m, maxi-mini+1)):
    while True:
       arv = random.randint(mini,maxi)
       if otsi(arv, V, i)== -1:
          V.append (arv); break
 return V

def otsi(x, V, n):  # otsib x-i V-s, kui pole, tagastab -1
  for i in range(n):
     if (x == V[i]): return i
  return -1

def tõlgi(sõna, sõnad1, sõnad2):
   for k in range(len(sõnad1)):
      if sõnad1[k]==sõna: return sõnad2[k]
   return None

def tõlkimine():
   est = loefail("eesti.txt")
   eng = loefail("inglise.txt")
   s = input("Sisesta sõna tõlkimiseks: ")
   v = tõlgi(s, est, eng)
   if v==None:
      v = tõlgi(s, eng, est)
      if v==None:
         print ("Sõna", s, "ei ole minu sõnastikus")
         return
   print(s, "-", v)
```

Python. Edetabel

Kestus: 45 min

Tunni eesmärgid: loendite kasutamine Pythonis, väärtuste kirjutamine tekstifaili

Õpitulemused:

- õpilane oskab lugeda väärtusi loendist, lisada ja kustutada loendi elemente
- õpilane oskab lugeda andmeid tekstifailist ja kirjutada tulemus tekstifaili
- õpilane oskab avastada ja parandada vigu skriptis

Õpilaste eelteadmised ja -oskused: Pythoni laused ja kasutajaliides, etteantud algoritmi mõistmine

Tunniks vajalikud materjalid: tööleht **Python**. **Edetabel**

Tunni käik:

- Tekstifaili avamise, sulgemise, andmete lugemise ja faili kirjutamise käskude tutvustamine ning proovimine. Proovimiseks luua paari kirjega tekstifail mingi tekstiredaktori abil.
- Skripti koostamine andmete lugemiseks tekstifailist loenditesse *nimed* ja *punktid*.
- Testimine, võimalike vigade avastamine ja parandamine. Print-käsuga võiks kontrolliks väljastada kõik väärtused.
- Loendites sobiva koha otsimine ja uute andmete lisamine loendite keskele või lõppu.
- Loendite väärtuste kirjutamine faili, mis nüüd tuleb avada kirjutamiseks.
- Testimine, võimalike vigade avastamine ja parandamine.
- Lisaülesanded

```
import os.path

def lisa_etab(nimi, punkte):
    nimed = [ ]; punktid = [ ]
    if os.path.exists("etab.txt"):
        fail = open("etab.txt")
        for kirje in fail:
            elem = kirje.split()
            nimed.append(elem[0])
            punktid.append(int(elem[1]))
        fail.close()

    tun = 0
    for k in range(len(nimed)):
        if punkte  > punktid[k]: # vahele
            nimed.insert(k, nimi)
            punktid.insert(k, punkte)
            tun = 1; break
    if tun == 0:  # lõppu
        nimed.append(nimi)
        punktid.append(punkte)
        k = len(punktid)-1

    fail = open("etab.txt", 'w')
    for i in range(len(nimed)):
      fail.write(nimed[i] +  (15 - \
      len(nimed[i]))*  " "   + str(punktid[i]) +
"\n")
    fail.close()
    return k + 1
```

Python. Ristkülik ja ring

Kestus: 45 min

Tunni eesmärgid: tutvumine mooduli turtle.py funktsioonidega

Õpitulemused:

- õpilane teab ja oskab kasutada mooduli turtle.py mõningaid funktsioone
- õpilane oskab avastada ja parandada vigu skriptis

Õpilaste eelteadmised ja -oskused: Pythoni laused ja kasutajaliides; funktsioonide koostamine ja kasutamine

Tunniks vajalikud materjalid: tööleht **Ristkülik ja ring Pythonis**, lisaleht 'Pythoni funktsioonid (meetodid) moodulis turtle.py', *Python Library Reference*

Tunni käik:

- Töölehega tutvumine. Selline ülesanne on tehtud ka Scratchis
- Tutvumine mooduli turtle funktsioonidega töölehel. Rohkem ja täpsemat infot saab Pythoni dokumentatsioonist; kui seda juba varem pole tehtud, on nüüd just aeg tutvuda osaga *Library Reference*.
- Proovida kasutada vahendeid joone ja ringi tekitamiseks, funktsioone *numinput* ja *write*.
- Õpilased koostavad skripti iseseisvalt.
- Et joonis ei tuleks liiga pisike või ülearu suur, võiks kasutusele võtta mastaabiteguri, algul konstandina, hiljem võiks sobiva suuruse arvutada.

```
import math,  turtle

turtle.title("Võrdse pindalaga ristkülik ja ring")
a = turtle.numinput("Andmete sisestamine","laius => ")
b = turtle.numinput ("Andmete sisestamine","kõrgus => ")
# Arvutused ja kirjutamine graafikaaknasse
S = a * b; P = 2 * (a + b)
d = math.sqrt(4 * S / math.pi)
suhe = P / (math.pi * d)
t = "pindala=" + str(round(S,3)) + "\nläbimõõt=" + str(round(d,
3))
t = t + "\nsuhe=" + str(round(suhe, 2))
turtle.write(t,align="center", font=("Arial", 16, "bold"))
# Joonistamine
m = 50;   # m - mastaabi tegur
a = a * m; b = b * m
turtle.penup(); turtle.pensize(3)
turtle.setpos(-a/2, -b/2); turtle.pendown()
turtle.forward(a);  turtle.left(90)  # liigu a1, pööra 90
turtle.forward(b);  turtle.left(90)  # liigu b1, pööra 90
turtle.forward(a);  turtle.left(90)
turtle.forward(b); turtle.penup()
turtle.setpos(0, -m*d/2); turtle.pencolor("red");
turtle.pendown();  turtle.left(90)
turtle.circle(m * d/2)  # ringjooon raadiusega m*d/2
turtle.exitonclick()
```

Python. Funktsiooni parameetrid

Kestus: 45 min

Tunni eesmärgid: funktsioonide loomine ja kasutamine, parameetrite komplekti valik

Õpitulemused:

- õpilane oskab valida omakoostatud funktsiooni jaoks sobivad parameetrid, nende järjestuse ja vaikeväärtused
- õpilane teab ja oskab kasutada mooduli turtle.py mõningaid funktsioone

Õpilaste eelteadmised ja -oskused: Pythoni laused ja kasutajaliides; funktsioonide koostamine ja kasutamine

Tunniks vajalikud materjalid: tööleht **Python**. **Funktsiooni parameetrid**, lisaleht 'Pythoni funktsioonid (meetodid) moodulis turtle.py'

Tunni käik:

- Parameetritest võib sõltuda funktsiooni otstarve ja kasutamise mugavus. Arutleda õpilastega, milliseid parameetrite komplekte on kasutatud varemkoostatud funktsioonides; millal oli parameetreid liiga vähe; kui palju parameetreid on liiga palju (kasutamisel ebamugav).
- Parameetri vaikeväärtus, kasutamine. Näited turtle funktsioonide hulgas (ring jm)
- Tutvumine töölehel toodud näitefunktsioonidega, nende katsetamine paari kujundi joonistamisel. Arutleda, kas oleks mõistlik muuta parameetrite hulka, nimesid, järjestust.
- Õpilased koostavad funktsioone joonistamiseks
- Kui selle harjutuse alusel antakse õpilastele iseseisev töö, võiks koostatud funktsioone teistega jagada ja ühiselt arutleda, milliseid funktsioonid on hästi koostatud ning milliseid võiks parandada.
- Õpilased koostavad paarikümnelauselise skripti, milles on kasutatud loodud funktsioone erinevate argumentidega.

Python. Funktsioonide uurimine

Kestus: 45 min

Tunni eesmärgid: Funktsioonide uurimine numbrilisel meetodil

Õpitulemused:

- õpilane oskab jagada rakenduse töö protseduuride (funktsioonide) vahel
- õpilane teab ja oskab kasutada mooduli turtle.py mõningaid funktsioone
- õpilane oskab avastada ja parandada vigu skriptis

Õpilaste eelteadmised ja -oskused: Pythoni laused ja kasutajaliides, mooduli turtle funktsioonide tundmine

Tunniks vajalikud materjalid: tööleht **Python**. **Funktsioonide uurimine**, lisaleht 'Pythoni funktsioonid (meetodid) moodulis turtle.py'

Tunni käik:

- Töölehe ja ülesande tutvustamine
- Funktsiooni väärtust arvutavate funktsioonide koostamine ja testimine
- Funktsiooni(de) koostamine, mis kirjutaks väärtused loenditesse, testimine
- Käsud funktsioonide suurimate ja väikseimate väärtuste leidmiseks, testimine
- Graafikaakna seadistamine (setworldcoordinates) ja (vajadusel) telgede joonistamine
- Arutlemine, kas joonistada graafikud üksteise järel või korraga, vajaliku hulga objektide (turtle) tekitamine
- Käsud graafikute joonestamiseks, testimine
- Lisaülesanded
- Arutlemine: milliseid matemaatikatundides analüütiliselt leitud funktsiooni karakteristikuid (ja kuidas) oleks võimalik lihtsalt leida numbriliste meetoditega.

```
from math import *
from turtle import *

def fun1(x):
    return 2*cos(pi*x)**3-3*sin(3*pi*x/5)

def fun2(x):
    return 2*sin(pi*x/3)+3*cos(pi*x-2)/(x**2+x+2)**(1/3)

def fun3(x):
    return 3*sin(pi*x/3)+2*sin(3*pi*x/2)

def tee_mas1(x, fn1, fn2, fn3, a, b, n):
    h = (b-a)/n
    for i in range(n+1):
        arg = a+i*h
        x.append(arg)
        fn1.append(fun1(arg))
        fn2.append(fun2(arg))
        fn3.append(fun3(arg))

def tee_mas2(a, b, n):
    h = (b-a)/n
    x=[]; fn1=[]; fn2=[]; fn3=[]
    for i in range(n+1):
        arg = a+i*h
        x.append(arg)
        fn1.append(fun1(arg))
        fn2.append(fun2(arg))
        fn3.append(fun3(arg))
    return x,fn1,fn2,fn3
```

```
print ("Fuktsiooni uurimine")
a = float(input("Lõigu algus => "))
b = float(input("Lõigu lõpp => "))
n = int(input("Jaotisi => "))

#x=[]; y1=[]; y2=[]; y3=[]
#tee_mas1(x,y1,y2,y3,a,b,n)
x,y1,y2,y3 = tee_mas2(a,b,n)

min1=min(y1); max1=max(y1)
min2=min(y2); max2=max(y2)
min3=min(y3); max3=max(y3)
min_k=min(min1, min2, min3)
max_k=max(max1, max2, max3)

setworldcoordinates(a, min_k , b, max_k)
# telgede joonistamine
if a<=0 and b>=0:
    pu(); goto(0,min_k); pd(); goto(0,max_k); pu()
if min_k<=0 and max_k>=0:
    pu(); goto(a,0); pd(); goto(b,0); pu()

t1=getturtle(); pencolor("green"); t1.pu();
t1.goto(a,y1[0]); t1.pd()
t2=clone(); pencolor("blue"); t2.pu();
t2.goto(a,y2[0]); t2.pd()
t3=clone(); pencolor("red"); t3.pu();
t3.goto(a,y3[0]); t3.pd()

for i in range(n+1):
    t1.goto(x[i],y1[i]);
    t2.goto(x[i],y2[i]);
    t3.goto(x[i],y3[i])
exitonclick()
```

Näidistööplaanid

Tööplaan1, lisamooduliks Python (tunnid 4 x 45min, kord nädalas)

nr	Teemad ja ülesanded	Kirjeldus
1	Sissejuhatus	mõisted: rakendus, algoritm, programm; rakenduse loomise etapid; tegevuse samm-sammuline kirjeldamine
2	Tutvumine programmeerimiskeskkonnaga Scratch	objektorienteeritud lähenemise põhimõisted; Scratchi kasutajaliidese osad ja võimalused; spraitide loomine ja lihtsa skripti koostamine
3	Scratch. Liikumisplokid	objekti (spraidi) omadused; lihtsamad skriptid liikumis-plokkidest
4	Scratch. Välimuseplokid	spraidi omadused ja graafikaefektid; lihtsamaid skriptid liikumise- ja välimuseplokkidest; tegevuste käivitamine
	Iseseisev töö: lihtsa rakenduse koostamine iseseisvalt (ülesanne Nimi) Lugeda: Scratch. Objektid, Scratch. Skriptid, Modelleerimine	
5	Test1 Lihtsad skriptid Scratchis. Lihtsa mängu algus (kordamine). Harjutus Kass, pall ja papagoi	spraite loomine; skripte koostamine ja kasutamine; sündmused (puude, serval jm), nende kasutamine skriptides
6	Muutuja loomine ja kasutamine. Mängu lõpetamine (Harjutus Kass, pall ja papagoi)	muutuja: mõiste, loomine, kasutamine skriptides; skriptide koostöö võimalused; töö lõpetamine programmi poolt
7	Muutujate kasutamine: harjutus Ristkülik	lihtsamad avaldised; muutujate kasutamine avaldistes; koostöö skriptide vahel; tehte- ja võrdluseplokid, funktsioonid. Andmete (väärtuste) sisestamine, tulemuste väljastamine.
8	Joonistuskäskude kasutamine Scratchis. (harjutus Rist_ring)	joonistamine skriptide abil
	Iseseisev töö: arvavaldiste koostamine ja kasutamine (ülesanne Ideaal) Lugeda: Scratch. Muutujajad, Scratch. Andmed ja avaldised, Algoritmimine	
9	Test2. Mõisted Algoritmi esitusviise	Skiptide käivitamine, kordused, valikud
10	Rakenduse loomine UML tegevus-diagrammi alusel (harjutus Rehmaatika)	UML diagrammi elementide
11	Lokaalne muutuja kui spraidi omadus (harjutus Autod)	
12	Rakenduse loomine UML tegevus-diagrammi alusel iseseisvalt (harjutus Arvamine)	
	Lugeda: Scratch. Juhtimine, Rakenduste loomine	
13	Test3. Käsud ja muutujad. Harjutus Labürint	Tegevuste jagamine skriptide vahel. Globaalsed ja lokaalsed muutujad
14	Tekstavaldised. Harjutus Tagurpidi	Sümbolite eraldamine tekstist, tekstide ühendamine. Loenduri kasutamine sümboli järjenumbrina.

nr	Teemad ja ülesanded	Kirjeldus
15	Loendid. Loomine ja kasutamine (harjutus Vestlus)	Loenditega seotud käsuplokkide tutvustamine.
16	Loendite kasutamine spraitide asukoha juhuslikul määramisel (näide Segamini)	
	Iseseisev töö: mängu või animatsiooni loomine ja dokumenteerimine. Suurem iseseisev töö, rakenduse esitlemine toimub kursuse lõpus. Lugeda: Scratch. Loendid, Algoritme loenditega	
17	Harjutus Sõnastikud	Loendi elementide lugemine tekstifailist, loendite kasutamine rakendustes. Loenduri kasutamine loendi elemendi järjenumbrina.
18	Tüüpalgoritme loenditega	Skriptide koostamine etteantud algoritmi abil, loendite kasutamine.
19	Tekstipõhised programmeerimiskeeled. Sissejuhatus Pythonisse	Tuttav algoritm realiseerituna erinevates programmeerimis-keeltes.
20	Python. Tutvumine keskkonnaga, lihtsamad arvavaldised.	
	Iseseisev töö: omaloomingulise rakenduse lõpetamine, tutvumine materjalidega Pythoni kohta.	
21	Pythoni laused. Lihtsa skripti koostamine	Omistamine, avaldis; funktsioonide kasutamine, moodulid; liitlaused, valikulaused
22	Lihtsa skripti koostamine iseseisvalt	Harjutus Ideaal
23	Protseduurid. Pythoni funktsioonide loomine	Harjutus Ideaal
24	Korduslaused Pythonis	
	Iseseisev töö: test4 Pythoni laused	
25	Korduslaused (kordamine)	
26	Pythoni „kilpkonnagraafika“ kasutamine (harjutus Ristkülik ja ring)	Tutvumine funktsioonidega moodulis turtle
27	Funktsioonide loomine, parameetrid	Funktsioonide koostamine kujundite joonistamiseks kilpkonnagraafika abil. Funktsiooni parameetrite valik, parameetri vaikeväärtus; funktsioonide kasutamine
28	Funktsioonide loomine ja kasutamine graafilise kompositsiooni loomisel	
	Iseseisev töö: graafilise kompositsiooni loomine	
29	Funktsioonide loomine ja kasutamine kompositsiooni loomisel (kordamine)	
30	Animatsiooni loomisest Pythonis	Moodul turtle
31	Tekstiavaldised (harjutus Tagurpidi)	
32	Loendid (harjutus Sõnastik)	
	Iseseisev töö: graafilise kompositsiooni lõpetamine	
33	Tekstid, loendid (kordamine)	
34	Tekstifailide kasutamine	
35	Scratchi rakenduste esitlemine. Kursusest kokkuvõtete tegemine	

Tööplaan2, lisamooduliks VBA (tunnid 2 x 45min, kord nädalas)

nr	Teemad ja ülesanded	Kirjeldus
1	Sissejuhatus	mõisted: rakendus, algoritm, programm; rakenduse loomise etapid; etteantud tegevuse samm-sammuline kirjeldamine
2	Tutvumine programmeerimiskeskkonnaga Scratch (harjutus Kraps123)	objektorienteeritud lähenemise põhimõisted; Scratchi kasutajaliidese osad ja võimalused; spraitide loomine ja lihtsa skripti koostamine
	Kodus: Tutvuda Scratchi lihtsamate võimalustega, abiks on kasutusjuhend.	
3	Scratch. Liikumisplokid (harjutus Vee-maailm)	objekti (spraidi) omadused; lihtsamad skriptid liikumis-plokkidest
4	Scratch. Liikumisplokid	
	Lugeda: Scratch. Objektid, Scratch. Skriptid	
5	Scratch. Välimuseplokid	spraidi omadused ja graafikaefektid; lihtsamaid skriptid liikumise- ja välimuseplokkidest; tegevuste käivitamis-võimalused
6	Ettevalmistus koduseks tööks	
	Kodus: Lihtsa rakenduse koostamine iseseisvalt (ülesanne1. Nimi)	
7	Test1 Lihtsad skriptid Scratchis. Lihtsa mängu algus (harjutus Kass, pall ja papagoi)	Kordamine: spraite loomine; skripte koostamine ja kasutamine; sündmused (puude, serval jm), nende kasutamine skriptides
8	Muutuja loomine ja kasutamine. Mängu lõpetamine	muutuja: mõiste, loomine, kasutamine skriptides; skriptide koostöö võimalused; töö lõpetamine programmi poolt
	Kodus: Test1 lõpetada, lugeda Modelleerimine	
9	Muutujate kasutamine: harjutus Ristkülik	Lihtsamad avaldised; muutujate kasutamine avaldistes; koostöö skriptide vahel
10	Joonistuskäskude kasutamine Scratchis. Harjutus Rist_ring	joonistamine skriptide abil
	Kodus: Lihtsa mängu koostamine (ülesanne2. Modellid) Lugeda Algoritmimine	
11	Protsesside juhtimine. Juhtimisplokid Scratchis	Skiptide käivitamine, kordused, valikud
12	Näide Juhtimine	
13	Arvavaldiste koostamine ja kasutamine	tehte- ja võrdluseplokid, funktsioonid
14	Harjutus ideaal	Andmete (väärtuste) sisestamine, tulemuste väljastamine
	Kodus: Harjutuse Ideaal lõpetamine	
15	Algoritmi mõistmine, rakenduse reali-seerimine samm-sammult	Algoritmi esitamine: sõnaline kirjeldus, pseudokood, UML tegevusdiagramm
16	Harjutus Rehmaatika	Rakenduse loomine UML tegevusdiagrammi alusel
	Kodus: harjutuse Rehmaatika lõpetamine (lisaülesanded)	
17	Muutuja skoop (harjutus Labürint)	Tegevuste jagamine skriptide vahel. Globaalsed ja lokaalsed muutujad
18	Harjutus Autod	Teise spraidi lokaalse muutuja väärtuse lugemine.

nr	Teemad ja ülesanded	Kirjeldus
	Kodus: tunnis tehtud harjutuste lõpetamine	
19	Arvamine (iseseisev töö tunnis)	Rakenduse loomine UML tegevusdiagrammi alusel
20	Tekstavaldised (harjutus Tagurpidi)	Sümbolite eraldamine tekstist, tekstide ühendamine. Loenduri kasutamine sümboli järjenumbrina.
	Kodus: tunnis tehtud harjutuste lõpetamine	
21	Loendid. Loomine ja kasutamine	Loenditega seotud käsuplokkide tutvustamine.
22	Harjutused Vestlus, Segamini	Loenduri kasutamine loendi elemendi järjenumbrina.
	Kodune töö: mängu või animatsiooni loomine ja dokumenteerimine (suurem iseseisev töö). Rakenduse esitlemine toimub kursuse lõpus.	
23	Harjutus Sõnastikud	Loendi elementide lugemine tekstifailist, loendite kasutamine rakendustes
24	Harjutus Tüüpalgoritme loenditega	Skriptide koostamine etteantud algoritmi abil
25	Tabeltöötlus. Sissejuhatus	Märkandmed töölehe lahtrites. Andmete vormindamine
26	Rakenduse koostamine MS Excelis	Viitamine töölehe lahtritele. Valemite koostamine ja paljundamine. Diagrammi loomine.
27	VBA. Sissejuhatus	VBA laused. VB editori kasutamine
28	VB protseduurid	VB protseduuride sisestamine ja kasutamine (funktsioon, sub-protseduur). Muutuja, avaldis, omistamine.
29	VB avaldiste koostamine. Harjutus Ideaal	Andmete küsimine kasutajalt, tulemuste väljastamine ekraanile. Valikulaused
30	Exceli objektide kasutamine VBA rakenduses	Andmete lugemine töölehelt, tulemuste väljastamine töölehele. Objektide Range (lahter) ja Shape (kujund) omaduste muutmine VBA protseduuris
	Kodus: Tunnis alustatud rakenduste lõpetamine ja täiendamine	
31	VBA. Tekstiavaldised	VB tekstifunktsioonide tutvustamine, tekstist sümbolite eraldamine, tekstide ühendamine.
32	VBA protseduuride koostamine etteantud algoritmi abil	For- korduslause kasutamine.
33	Scratchis iseseisvalt loodud rakenduste esitlemine	
34	Scratchis iseseisvalt loodud rakenduste esitlemine	
35	Kokkuvõtete tegemine kursusest	