# Eessõna

Käesoleval ajal on kõik inimtegevuse valdkonnad, alates teadusest ja tehnoloogiast ning lõpetades kunsti, spordi, olme ja meelelahutusega, väga tihedalt seotud informatsiooni ja infotehnoloogia kasutamisega. Võiks öelda, et praeguse ajastu ja tuleviku infoühiskonna olulisteks ja keskseteks nähtusteks on infotöötlus ja programmjuhtimisega süsteemid: arvutid ja arvutivõrgud, programmjuhtimisega seadmed ja tööpingid, robotid ja sisseehitatud protsessoritega seadmed, telefonid ja muud sideseadmed jm. Koos sellega kasvab vajadus infotehnoloogiaspetsialistide järele, kes projekteerivad, loovad ja hooldavad erinevaid IKT süsteeme ja nende tarkvara. Tänapäeval peavad praktiliselt kõikide erialade spetsialistid mitte ainult oskama kasutada infotehnoloogia vahendeid, vaid suutma ka püstitada ülesandeid oma valdkonna probleemide lahendamiseks ning vajadusel osalema vastavates projektides näiteks oma valdkonna mudelite ja algoritmide loomisel. Samal ajal täheldatakse nii meil kui mujal, et kooli lõpetajate huvi infotehnoloogia erialade õppimiseks ei ole piisav ja õppima asujate ettevalmistuse tase on puudulik.

Viimasel ajal on toimumas mitmes riigis (näit. USA, Suurbritannia jm) aktiivsed arutelud ja on läbi viidud põhjalikud uuringud olukorrast infotehnoloogia ja arvutiteaduse õpetamisest koolides. Praeguseks on jõutud järeldusele, et olukord ei vasta aja nõuetele ja arengutendentsidele ning õpetamise sisu vajab olulisi muutusi. Nende uuringute alusel on loodud koolide arvutiõpetuse jaoks uued kontseptsioonid ja standardid ning soovitused õppekavade uuendamiseks. On jõutud järeldusele, et koolis ei piisa ainult infotehnoloogia vahendite kasutama õppimisest, vaid peab tundma ka nende töö põhimõtteid ja loomise meetodeid. Vt näiteks USA arvutiõpetajate assotsiatsiooni ([CSTA](http://www.csta.acm.org/)) poolt koolide jaoks koostatud arvutiteaduse rahvuslikku [standardit](http://csta.acm.org/Curriculum/sub/K12Standards.html) ja [tüüpõppekava](http://www.exploringcs.org/) ning Suurbritannia koolide arvutiteaduse õppekava [kontseptsiooni](http://www.computingatschool.org.uk/data/uploads/ComputingCurric.pdf). Viimases on määratletud ka nõuded arvutialaste teadmiste ja oskuste kohta erinevate kooliastmete jaoks: alates 1. astmest (vanus 5-7 aastat). Materjalides on tehtud ettepanek muuta vastav õppeaine koolides kohustuslikuks.

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

Infotehnoloogilise lähenemisviisi (CT) **põhiolemus seisneb oskuses formuleerida probleeme sellisel kujul, et neid oleks võimalik efektiivselt lahendada infotehnoloogia vahendite abil**. Sellega seonduvad järgmised mõisted ja tegevused: abstraktsioon, süsteemide struktuuri, oleku ja käitumise modelleerimine, süsteemianalüüs, rakenduste ja infosüsteemide disain, algoritmimine ja programmeerimine jm.

Mõned lingid, kus on võimalik saada täiendavat informatsiooni: [Wikipedia](http://en.wikipedia.org/wiki/Computational_thinking), [CSTA](http://csta.acm.org/Curriculum/sub/CompThinking.html), [ISTE](http://www.iste.org/learn/computational-thinking) (*The International Society for Technology in Education*), [Google](http://www.google.com/edu/computational-thinking/what-is-ct.html).

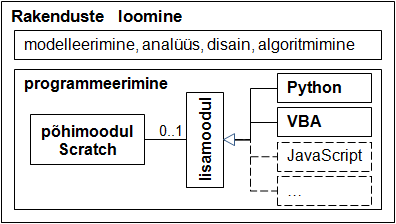
Üheks tähelepanuväärseks nähtuseks on viimasel ajal järjest tekkivad ning kiiresti arenevad ja levivad programmeerimis[keeled](http://en.wikipedia.org/wiki/Category:Educational_programming_languages), mis on loodud spetsiaalselt [programmeerimise](http://en.wikipedia.org/wiki/Category:Educational_programming_languages) õppimiseks/[õpetamise](http://en.wikipedia.org/wiki/Category:Educational_programming_languages)ks: [Alice](http://en.wikipedia.org/wiki/Alice_%28software%29), [Scratch](http://en.wikipedia.org/wiki/Scratch_(programming_language)), [BYOB](http://byob.berkeley.edu/), [AgentSheets](http://en.wikipedia.org/wiki/AgentSheets), [Greenfoot](http://en.wikipedia.org/wiki/Greenfoot), [Kodu](http://en.wikipedia.org/wiki/Kodu_Game_Lab), [MS Small Basic](http://en.wikipedia.org/wiki/Microsoft_Small_Basic), [Logo](http://en.wikipedia.org/wiki/Logo_%28programming_language%29), [StarLogo](http://en.wikipedia.org/wiki/StarLogo), [MIT App Inventor](http://en.wikipedia.org/wiki/App_Inventor_for_Android) jm. Neid nimetatakse ka hariduslikeks programmeerimiskeelteks (vt. [Wikipeedias](http://en.wikipedia.org/wiki/Category:Educational_programming_languages)) Õppeotstarbelisi programmeeri­mis­keeli on loodud ja kasutatud juba ammu. Esimeste seas olid näiteks [BASIC](http://en.wikipedia.org/wiki/BASIC) (1964), [Logo](http://en.wikipedia.org/wiki/Logo_programming_language) (1967), [Pascal](http://en.wikipedia.org/wiki/Pascal_(programming_language)) (1970). Mõnedest neist kasvasid välja ka nö tööstuslikud variandid näiteks Visual Basic ([VB](http://en.wikipedia.org/wiki/Visual_Basic)) ja Visual Basic for Application ([VBA](http://en.wikipedia.org/wiki/Visual_Basic_for_Applications)).

Uue põlvkonna hariduslikud programmeerimiskeeled on enamikus graafilised ehk [visuaalsed programmeerimiskeeled](http://en.wikipedia.org/wiki/Visual_programming_language). Tegemist on lihtsa süntaksiga (või süntaksita) 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.

Ülaltooduga on arvestatud ka antud kursuse sisu, struktuuri ja kasutatavate vahendite valimisel.

Kursuse põhieesmärgiks on omanda rakenduste loomise ja programmeerimise põhimõtted. Rõhuasetus on just rakenduste loomisel, mitte lihtsalt programmeerimisel mingis keeles. Rakenduste loomine on mõnevõrra laiem valdkond ja sisaldab muuhulgas ka programmeerimist. See on tänapäeval tihedalt seotud selliste tegevustega nagu modelleeri­mine: süsteemide ja objektide struktuur, omadused ja tegevused, rakenduste disain: andmed, kasutajaliides, protseduurid ja koostöö nende vahel, algoritmide koostamine. Samal ajal aga peab rõhutama, et programmeerimise oskus vähemalt ühes programmeerimiskeeles (parem kui mitmes) on absoluutselt vajalik. Arvestades kursuse mahtu, kulub suurem osa ajast siiski programmeerimise põhikontseptsioonide ja oskuste omandamisele.

Kursuse struktuuri ja põhikomponente kajastab allolev joonis.



See programmeerimise osa skeem tutvustab veidi 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.

Moodulite orienteeruvad mahud: põhimoodul 15–35 t, lisamoodul 0–20 t. Lisamooduli keele ja moodulite mahud (kokku 35 t) valib kool arvestades õppijate õppesuunda. Lisamoodul võib ka puududa. Lisamooduli kasutamise üheks eesmärgiks on õppijale tekstipõhise programmeerimiskeele tutvustamine.

Rakenduste loomise meetodeid, vahendeid ja põhifaase (ülesande püstitus, analüüs, disain ja programmeerimine) vaadeldakse ülesannete lahendamisel kogu kursuse jooksul. Läbivalt käsitletakse ka modelleerimist ja algoritmimist. Modelleerimises tutvustatakse objektorienteeritud lähenemisviisi ja unifitseeritud modelleerimiskeelt UML. Objekte, klasse ning vastavaid diagramme käsitletakse lühidalt ja lihtsustatult. Põhjalikumalt vaadeldakse protsesside ja algoritmide esitamist tegevusdiagrammide ja algoritmikeele (pseudokoodi) abil. Kursuse üldosa materjalid on eraldi failides: [modelleerimine](http://rlpa.ttu.ee/Modelleerimine.pdf), [rakenduste loomine](http://rlpa.ttu.ee/Rakendused.pdf) ja [algoritmimine](http://rlpa.ttu.ee/Algoritmimine.pdf). Nende poole tuleb pöörduda kursusejooksul korduvalt. Kohe alguses oleks otstarbekas tutvuda modelleerimise olemuse ja põhimõistetega, sest neid kasutatakse praktiliselt kõikides teistes materjalides.

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 arvutuslikke, andmetöötluse ja arvutigraafika algoritme.

Scratchi [kodusaidil](http://scratch.mit.edu/) Massachusettsi Tehnoloogia Instituudis (MIT) on suurel hulgal [õppematerjale](http://scratched.media.mit.edu/resources/harvard-cs50-introduction-programming-and-scratch) ning näiteid ja demosid. Scratchi asemel või lisaks sellele võib kasutada programmeerimissüsteemi [BYOB](http://byob.berkeley.edu/) (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](http://www.exploringcs.org/curriculum)). Scratch ja BYOB leiavad kasutamist arvutiõpetuse algkursustes ka mitmes maailma juhtivas ülikoolis ([Harvard](http://scratched.media.mit.edu/resources/harvard-cs50-introduction-programming-and-scratch#comments), [Berkeley](http://bjc.berkeley.edu/) jt). Programmeerimiskeelte kasutamise ulatuse (populaarsuse) nn [Tiobe indeksis](http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html) on Scratch tavaliselt 30 esimese piirimail. Scratchi kasutajaliides, abiinfosüsteem ja [kasutamisjuhend](http://www.tud.ttu.ee/~vilip/Scratch/Juhend/Scr_juhend14_P.html) on olemas ka eesti keeles.

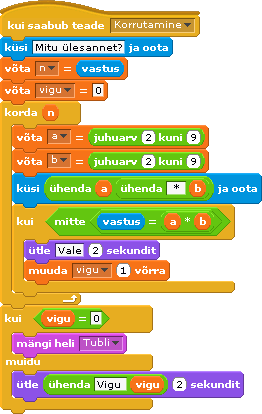
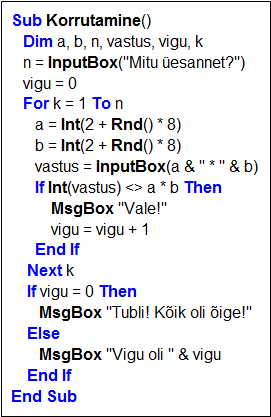
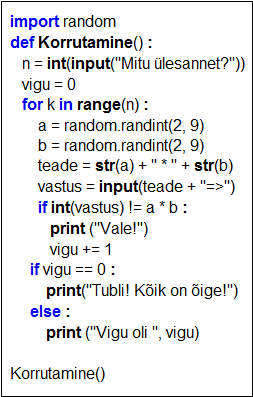
Lisamoodulites käsitletakse rakenduste loomist erinevate programmeerimiskeelte abil. Üldiselt ei peaks piirduma ainult ühe keelega vaid võiks tutvuda veel vähemalt ühe programmeerimis­keelega, valides lisamooduli. Tulevikus võiks lisamooduleid olla 4–5. Valikus võiksid olla mõned sellised keeled nagu[: JavaScript](http://et.wikipedia.org/wiki/JavaScript), [PHP](http://et.wikipedia.org/wiki/PHP), [Visual Basic](http://en.wikipedia.org/wiki/Visual_Basic) jm. Esialgu on materjalid kahe lisamooduli kohta: Python ja VBA.

[Python](http://et.wikipedia.org/wiki/Python_(programmeerimiskeel)) on vabavaraline üldotstarbeline objektorienteeritud, lihtne ja võimas programmeerimiskeel. Keel on loodud 1991. aastal, selle populaarsus ja kasutamise ulatus (kuues-kaheksas koht [Tiobe indeksis](http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html)) 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](http://en.wikipedia.org/wiki/Visual_Basic_for_Applications) (*Visual Basic for Application*) kuulub paljudes koolides kasutatava kontoripaketi MS Office juurde, pakkudes lihtsaid vahendeid rakenduste loomise, programmeerimise ja modelleerimise õppimiseks objektorienteeritud keskkonnas. VBA põhineb üldotstarbelisel programmeeri­missüsteemil [Visual Basic](http://en.wikipedia.org/wiki/Visual_Basic) (**VB**), mis on kasutusulatuselt enam-vähem samal tasemel nagu PHP ja Python.

VBA on üks enim kasutatav vahend **dokumendipõhiste rakenduste** loomisel ja arendamisel 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.

Õpikus vaadeldakse VBA kasutamist MS Exceli keskkonnas. Eesmärgiks ei ole Exceli programmeerimine, vaid Scratchis omandatud programmeerimis­oskuste ja -teadmiste süvendamine ja laiendamine tekstipõhiste, objektorienteeritud programmeerimist toetavate vahendite abil. VBA ja Excel pakuvad selleks suurepäraseid võimalusi.

Allpool on võrdluseks toodud näide Scratchis, VBAs ja Pythonis. Programm esitab ***n*** korrutamise ülesannet, kontrollib kasutaja vastuseid ning annab üldhinnangu.