

Võistlusprogrammeerimine

ehk kuidas kirjutada kiiret koodi

Targo Tennisberg, Katrin Gabrel

0	Sissejuhatus.....	10
0.1	Mis on võistlusprogrammeerimine?	10
0.2	Mida siit õpikust leida võib?.....	11
0.3	Kontrollülesannete lahendamine	13
0.4	Kasutatud ülesanded ja pildid	13
0.5	Autoritest.....	13
0.6	Tänuavaldused.....	14
0.7	Veaparandused ja soovitused	14
1	Programmide sisemaailm.....	15
1.1	Programmi elutsükkel.....	16
1.1.1	Lähtekood.....	16
1.1.2	Lähtekoodi transleerimine	16
1.1.3	Teegid	17
1.1.4	Linkimine	18
1.1.5	Laadimine ja täitmine	18
1.2	Andmete hoidmine ja töötlemine arvutis	18
1.2.1	Protsessori tööpõhimõte.....	18
1.2.2	Protsessori jõudlus	19
1.2.3	Käsukskonveier	20
1.2.4	Hargnemise ennustamine	21
1.2.5	Andmete liikumine	23
1.2.6	Mälu.....	23
1.3	Andmetüübid.....	25
1.3.1	Täisarvud	25
1.3.2	Ujukomaarvud	26
1.3.3	Püsikomaarvud	31
1.3.4	Märgid	31
1.3.5	Tõeväärtused.....	31
1.3.6	Viidad.....	32
1.3.7	Struktuursed tüübid ehk liitüübid	32
1.3.8	Kirjed.....	33
1.3.9	Stringid.....	33
1.3.10	Massiivid.....	33
1.4	Operatsioonid stringidega	34
1.4.1	Märk kohal i.....	34
1.4.2	Stringide võrdlemine	34
1.4.3	Stringide ühendamise ehk liitmine	35
1.5	Sisend-väljund	37
1.5.1	Standardvood	37

1.5.2	Failidest lugemine ja kirjutamine	38
1.5.3	Sisendi töötus	39
1.5.4	Väljund.....	40
1.5.5	Jooksvalt töötamine	41
1.6	Programmeerimiskeelte võrdlus	42
1.6.1	C++	42
1.6.2	Java	43
1.6.3	Python	44
1.7	Testimine	46
1.7.1	Piirjuhud	46
1.7.2	Õigsuse kontroll.....	47
1.7.3	Algoritmi jõudluse kontroll.....	48
1.8	Silumine	49
1.8.1	Arenduskeskkonnad	49
1.8.2	Aja mõõtmine	50
1.9	Kontrollülesanded	51
1.9.1	Järelmaks	52
1.9.2	Kell	52
1.9.3	Tigu	53
1.9.4	3D printer	53
1.9.5	Mõttemeister	54
1.9.6	Male.....	54
1.9.7	Riigihanked	55
1.9.8	Bender	56
1.9.9	Interpretaator.....	56
1.9.10	Pangatellerid.....	57
1.10	Viited lisamaterjalidele.....	58
2	Läbivaatus- ja otsingualgoritmid	59
2.1	Alamprogrammid	59
2.1.1	Pinumälu.....	59
2.1.2	Funktsiooni kohalikud andmed	60
2.1.3	Pinu ületäitumine	60
2.1.4	Pinu kasutamine protsessoris.....	61
2.1.5	Kuhimälu.....	62
2.1.6	Funktsiooni parameetrid	63
2.1.7	Objektide edastamine parameetritena	64
2.2	Rekursioon.....	66
2.2.1	Hargnemiseta rekursioon	66
2.2.2	Rekursioonivalem	67
2.2.3	Hargnemistega rekursioon	68
2.2.4	Sabarekursioon ja väljakutsete optimeerimine.....	70
2.3	Variantide läbivaatamine	72
2.3.1	Tagurdusmeetod	72
2.3.2	Iteratsioonimeetod.....	72
2.3.3	Tagurduse ja iteratsiooni võrdlus	73
2.3.4	Variantide läbivaatamine tagurdusmeetodiga.....	73
2.3.5	Variantide läbivaatamine iteratsioonimeetodiga.....	74
2.3.6	Variantide läbivaatus programmeerimisvõistlustel	74

2.4	Läbivaatuse optimeerimine.....	75
2.4.1	Lihtne tagurdusmeetod.....	75
2.4.2	Andmete optimeerimine.....	76
2.4.3	Ettearvutamine.....	77
2.4.4	Otsingupuu ahendamine.....	78
2.4.5	Sisemiste tsüklite optimeerimine.....	79
2.4.6	Andmestruktuuride optimeerimine.....	80
2.4.7	Rekursiooni eemaldamine.....	82
2.5	Kahendotsing.....	83
2.5.1	Vastuse kahendotsing.....	83
2.5.2	Kahendotsing andmetest.....	85
2.5.3	Topeltkahendotsing.....	86
2.5.4	Otsing kahemõõtmelisest tabelist sadulameetodil.....	87
2.6	Jaga ja valitse.....	91
2.7	Sissejuhatus kombinatoorikasse.....	92
2.7.1	Permutatsioonid.....	92
2.7.2	Permutatsioonide konstrueerimine.....	93
2.7.3	Kombinatsioonid.....	95
2.8	Kontrollülesanded.....	97
2.8.1	Autoturg.....	97
2.8.2	Kaupmees ja maksukoguja.....	98
2.8.3	Lippude vasturünnak.....	98
2.8.4	Akulaadija.....	99
2.8.5	Pildi pakkimine.....	99
2.8.6	Stern-Brocot' puu.....	100
2.8.7	Viinamarjad.....	101
2.8.8	Erinevad summad.....	102
2.8.9	Kodumasinat näidikud.....	103
2.8.10	Graafi värvimine.....	104
2.9	Viited lisamaterjalidele.....	104
3	Algoritmi keerukus ja põhilised andmestruktuurid.....	105
3.1	Algoritmi keerukus.....	105
3.1.1	Keskmine ja halvim keerukus.....	106
3.1.2	Asümptootiline hinnang.....	106
3.1.3	Kasvuseoste omadused.....	108
3.2	Algoritmi keerukuse hindamine.....	108
3.2.1	Hargnemiste keerukuse hindamine.....	108
3.2.2	Tsüklite keerukuse hindamine.....	109
3.2.3	Mitmekordsete tsüklite keerukus.....	109
3.2.4	Rekursiooni keerukuse hindamine.....	110
3.3	Tavalised keerukusklassid.....	110
3.3.1	Keerukusklass ja ajalimiit.....	112
3.4	Andmestruktuurid.....	112
3.4.1	Andmestruktuuride klassifitseerimine.....	113
3.5	Massiiv.....	113
3.5.1	Massiivid Pythonis.....	113
3.5.2	Massiivi võimalused ja piirangud.....	114
3.5.3	Mitmemõõtmelised massiivid.....	114

3.5.4	Dünaamilised massiivid	115
3.6	Ahel.....	115
3.7	Pinu.....	117
3.8	Järjekord	118
3.8.1	Kaheotsaline järjekord.....	119
3.8.2	Eelistusjärjekord	119
3.9	Pinu ja järjekorra kasutamine.....	119
3.10	Kahendpuu	123
3.10.1	Kahendpuu esitus.....	124
3.10.2	Kahendotsingu puu.....	124
3.11	Kujutis.....	124
3.11.1	Kujutis programmeerimiskeeltes	125
3.12	Praktiline ülesanne	126
3.13	Kuhi.....	127
3.14	Sortimine	128
3.14.1	Mullimeetod	129
3.14.2	Valikmeetod	130
3.14.3	Pistemeetod	131
3.14.4	Põimemeetod	131
3.14.5	Kiirmeetod	133
3.14.6	Võrdlustel põhineva sortimise keerukuse alampiir	134
3.15	Sortimise erimeetodid.....	135
3.15.1	Loendamismeetod.....	135
3.15.2	Positsioonimeetod.....	136
3.15.3	Kimbumeetod.....	136
3.16	Mitme kriteeriumi järgi sortimine	136
3.17	Programmeerimiskeelte standardtegid	136
3.17.1	C++	137
3.17.2	Java	137
3.17.3	Python	137
3.18	Kontrollülesanded	139
3.18.1	Vabrikud	139
3.18.2	Jalgpalliturniir	140
3.18.3	Erdöse arvud.....	141
3.18.4	Programmeerimisvõistlus.....	142
3.18.5	Pannkoogid.....	142
3.18.6	Kaartide segamine	143
3.18.7	Kass klaviatuuril.....	143
3.18.8	Pinugrammid	144
3.18.9	Parv.....	145
3.18.10	Ahelmurrud	146
3.19	Viited lisamaterjalidele.....	146
4	Arvuteooria.....	147
4.1	Jaguvus ja jääk	147
4.1.1	Jaguvus	147
4.1.2	Jääk	148
4.1.3	Jäägi leidmine programmeerimiskeeltes.....	148
4.1.4	Ülesanne: loosiümbrikud.....	149

4.1.5	Arvutamine moodulitega.....	150
4.1.6	Ülesanne: anagrammid 2.....	152
4.2	SÜT ja VÜK.....	154
4.2.1	Eukleidese algoritm suurima ühisteguri leidmiseks.....	155
4.2.2	Vähima ühiskordse leidmine.....	155
4.2.3	Ülesanne: hammasrattad.....	156
4.2.4	Diofantilised võrrandid.....	157
4.3	Algarvud.....	158
4.3.1	Eratosthenese sõel.....	158
4.3.2	Algarvulisuse kontroll.....	160
4.3.3	Ülesanne: algarvu-Scrabble.....	162
4.3.4	Arvu algtegurid.....	165
4.4	Positsioonilised arvusüsteemid.....	166
4.4.1	Süsteemide vahel teisendamine.....	166
4.4.2	Ülesanne – positsioonilised arvusüsteemid.....	168
4.4.3	Arvutamine kahendsüsteemis.....	173
4.5	Suurte arvudega arvutamine.....	173
4.5.1	Suurte arvude esitus.....	173
4.5.2	Suurte arvude liitmine ja lahutamine.....	174
4.5.3	Suurte arvude korrutamine ja astendamine.....	176
4.5.4	Efektne astendamine.....	177
4.5.5	Suurte arvude jagamine.....	178
4.5.6	Ülesanne – anagrammid 3.....	178
4.5.7	Suured arvud erinevates programmeerimiskeeltes.....	179
4.5.8	Logaritm ja eksponent suurte arvudega arvutamisel.....	180
4.6	Kontrollülesanded.....	182
4.6.1	Suur arv.....	182
4.6.2	Faktoriaali lõpp.....	182
4.6.3	Goldbachi hüpotees.....	183
4.6.4	Klaaskuulid.....	183
4.6.5	VÜK-võimsus.....	184
4.6.6	Suurim algtegur.....	184
4.6.7	Vahetusraha.....	185
4.6.8	Taandumatud murrud.....	185
4.6.9	Mertensi funktsioon.....	186
4.6.10	Mõrvamüsteerium.....	186
4.7	Viited lisamaterjalidele.....	187
5	Dünaamiline planeerimine algajatele.....	188
5.1	Sissejuhatav ülesanne – Fibonacci jada.....	188
5.1.1	Tavaline rekursioon ehk jõumeetod.....	189
5.1.2	Mäluga rekursioon.....	190
5.1.3	Alt üles DP.....	191
5.1.4	Kokkuhoidlik DP.....	192
5.1.5	DP retsept.....	193
5.2	Lineaarne väärtuste tabel - optimaalne maksimine.....	193
5.2.1	Ahne algoritm.....	193
5.2.2	Kõigi variantide läbivaatamine (rekursioon).....	194
5.2.3	DP lahendus.....	195

5.3	Pikima kasvava osajada leidmine	197
5.3.1	Kõigi võimaluste läbivaatus	197
5.3.2	DP lahendus.....	198
5.3.3	Tagurdusmeetodiga lahendus.....	199
5.4	Kahemõõtmeline väärtuste tabel – pikim ühine osajada.....	200
5.4.1	DP lahendus.....	200
5.5	Ristsummade loendamine.....	202
5.5.1	Ristsummade arvutamine	202
5.5.2	DP lähenemine	202
5.5.3	DP Exceliga.....	203
5.5.4	Tabeli koostamine programselt.....	204
5.5.5	Ülesande lahendus (tabeli kasutamine).....	205
5.6	Mitmemõõtmeline DP tabel – Miljonär ja vaeslapsed.....	206
5.6.1	Kõikide võimaluste läbivaatus	207
5.6.2	Korduvad harud	207
5.6.3	DP tabeli koostamine	208
5.6.4	Lahendus	209
5.7	Tõeväärtuste tabeliga DP - õiglase jagamine.....	211
5.7.1	Kõik kombinatsioonid	211
5.7.2	DP lahendus.....	211
5.8	Bitimaskide põhine väärtuste tabel - moekunstnik ja koiliblikas	213
5.8.1	Kõigi läbivaatuste puu	214
5.8.2	Tee DP poole.....	215
5.8.3	Bitimaskide kasutamine tabeli indeksitena.....	215
5.8.4	DP lahendus.....	217
5.9	Kuidas ja millal DP-d kasutada.....	218
5.9.1	Ülalt alla vs alt üles DP.....	219
5.9.2	Kidunud DP	219
5.10	DP praktilised kasutusalaad	220
5.11	Kontrollülesanded	221
5.11.1	Aktsiaturg	221
5.11.2	Protsessori planeerimine.....	221
5.11.3	Maksmise võimalused	222
5.11.4	Statistika manipuleerimine.....	222
5.11.5	Lennukikütus	223
5.11.6	Kahjuritõrje.....	224
5.11.7	Torni ehitamine	225
5.11.8	Putin sukeldumas	226
5.11.9	Arvude liitmine	226
5.11.10	Templisambad	227
5.12	Viited lisamaterjalidele.....	227
6	Sissejuhatus graafiteooriasse	228
6.1	Graafiteooria terminid.....	229
6.1.1	Suunatud ja suunamata servad	229
6.1.2	Teed graafis	229
6.1.3	Sidususkomponendid	230
6.2	Graafide esitusviisid	230
6.2.1	Naabrusmaatriks	231

6.2.2	Tippude loend.....	231
6.2.3	Servade loend.....	232
6.2.4	Regulaarsete servadega graafid	233
6.3	Graafi läbimine	233
6.3.1	Sügavuti läbimine	233
6.3.2	Ülesanne: Ratsu teekond.....	235
6.3.3	Laiuti läbimine	238
6.3.4	Ülesanne: Ratsu teekond 2.....	239
6.4	Sidususkomponentide leidmine	241
6.4.1	Ülesanne: Ratsud.....	241
6.5	Üleujutamine.....	243
6.5.1	Ülesanne: Veekogud.....	243
6.6	Kahealuseline graaf	244
6.7	Topoloogiline sorteerimine	245
6.7.1	Ülesanne: Loomad.....	246
6.8	Kaalutud servadega graafid.....	248
6.9	Dijkstra lühima tee leidmise algoritm	248
6.9.1	Ülesanne: sõiduaeg	251
6.10	Puud.....	253
6.10.1	Puud kui graafid.....	253
6.10.2	Puu definitsioon	253
6.10.3	Graafitöötusalgoritmid puul	253
6.10.4	Kahendpuu sügavuti läbimine	254
6.10.5	Puu laiuti läbimine.....	255
6.10.6	Ülesanne: Kahendpuud	255
6.11	Kontrollülesanded	258
6.11.1	Robotivõistlus	258
6.11.2	Civilization	259
6.11.3	Dominod.....	259
6.11.4	Projektiplaan	260
6.11.5	Sõnateisendused	260
6.11.6	Kahevärviprobleem	261
6.11.7	Joonejälgija robot.....	262
6.11.8	Numbriruudud	263
6.11.9	Optimaalne ruuting	264
6.11.10	Liftid.....	265
6.12	Viited lisamaterjalidele.....	266
7	Efekttiivne programmeerimistehnika.....	267
7.1	Koodistiil.....	268
7.1.1	Pikaajaline ja lühiajaline lähenemine	268
7.1.2	Alamprogrammid.....	268
7.1.3	Näide: Tankid.....	270
7.1.4	Kommentaarisid.....	274
7.1.5	Nimed	276
7.1.6	Funktsioonide nimetamine.....	277
7.2	Testimine	277
7.2.1	Testide koostamine	278
7.2.2	Übermööd	278

7.2.3	Testide genereerimine	281
7.2.4	Käsuinterpretaatorid ja skriptimine	283
7.2.5	Valideerimine	285
7.3	Tunne oma keelt.....	286
7.3.1	Teegid	286
7.3.2	Andmestruktuurid	286
7.3.3	Makrod	287
7.4	Võistluste strateegia.....	288
7.5	Ahned algoritmid.....	288
7.5.1	Mündid	289
7.5.2	Kingid	290
7.5.3	Majakavahid	293
7.5.4	Veel ahnetest algoritmidest	295
7.5.5	Suveniirid.....	295
7.6	Kontrollülesanded	299
7.6.1	Onu Robert	299
7.6.2	Bussijuhid.....	300
7.6.3	Hernehirmutised	300
7.6.4	Kolimine.....	301
7.6.5	Krokodillid.....	301
7.6.6	Lohe Gorõnitš	302
7.6.7	Fraktsioonid.....	303
7.6.8	Bitimask	303
7.6.9	LED-lambid.....	304
7.6.10	Antimonotoonne jada	305
7.7	Viited lisamaterjalidele.....	305
8	Dünaamiline planeerimine edasijõudnutele	306
8.1	DP kui graafi lineariseerimine.....	306
8.2	Seljakoti pakkimine.....	306
8.3	Edit distance	307
8.4	Geenijärjendite leidmine.....	307
8.5	Rändkaupmehe ülesanne	307
8.6	Dynamic Time Warping	307
8.7	Maatriksite korrutamine	307
8.8	Floyd-Warshalli algoritm	307
8.9	Kontrollülesanded	307
9	Graafiteooria	308
9.1	Toesepuu	308
9.2	Euleri tsükli leidmine	308
9.3	Kahealuselised graafid.....	308
9.4	Maksimaalne voog ja minimaalne lõige	308
9.5	Kontrollülesanded	308
10	Andmestruktuurid edasijõudnutele	309
10.1	Fenwicki puu.....	309
10.2	2D Fenwicki puu	309
10.3	Lõikude puu	309
10.4	Laisk väärtustamine lõikude puudes	309
10.5	O(log n) operatsioonid puudel	309

10.6	Kontrollülesanded	309
11	Tekstialgoritmid.....	310
11.1	Teksti parsimine	310
11.2	Räsid	310
11.3	Knuth-Morris-Pratti algoritm.....	310
11.4	Aho-Corasicki algoritm	310
11.5	Sufiksipuud	310
11.6	Kontrollülesanded	310
12	Arvutusgeomeetria.....	311
12.1	Samal joonel asumise kontroll	311
12.2	Paralleelsuse ja samasihilisuse kontroll	311
12.3	Lõikude lõikumine	311
12.4	Kujundi pindala.....	311
12.5	Punkti sisaldumine kujundis	311
12.6	Koordinaatide pakkimine	311
12.7	Kumer kate	311
12.8	Sweeping line	311
12.9	Bresenhami algoritm	311
12.10	Magic missile	311
12.11	Kontrollülesanded	311

0 SISSEJUHATUS

0.1 MIS ON VÕISTLUSPROGRAMMEERIMINE?

Programmeerimise eesmärk on panna arvuti tegema midagi kasulikku. Enamasti on see mingi tegevus, mille muidu oleks ära teinud inimesed. Nead saavad nüüd aga teha midagi meeldivamat – las masin töötab, tema on rauast. Distsipliinina on programmeerimine üsna noor, kuid ometi läbi teinud pead pööritama paneva arengu, eelkõige kasutatavate vahendite (arvutid ja programmeerimist abistav tarkvara) osas. Kaasaegne riistvara võimaldab „rauast masinal“ sooritada teatud ülesandeid miljardeid kordi kiiremini kui ükski inimene suudaks. Samas mõeldakse pidevalt välja uusi ja keerulisemaid ülesandeid, mis lükkavad tagant ka programmeerimise sisemist võimekust ehk algoritmide ja meetodite arengut.

Uute algoritmide ja meetodite väljamõtlemine loob aga viljaka pinnase programmeerijatele üksteisega mõõduvõtmiseks. Kes kirjutab kiiremini efektiivsema programmi etteantud ülesande lahendamiseks? Selles vallas toimub palju võistlusi, kus pannakse proovile osalejate teadmised algoritmide alal, võimekus oma mõtteid kiiresti programmideks realiseerida ning oskus kirjutada veavaba koodi.

Eestis on selles vallas peamine võistlus Eesti Informaatikaolümpiaad, mis on suunatud eelkõige kooliõpilastele, kuid mille raames toimub ka kõigile avatud üritusi. Edukamatel kooliõpilastel on võimalus osaleda regionaalsel Balti Informaatikaolümpiaadil (kus osalevad Läänemere äärsed riigid) ning Rahvusvahelisel Informaatikaolümpiaadil (IOI).

Ülikoolitasemel on peamine võistlus ACM International Collegiate Programming Contest (ACM-ICPC). Neil, kes on koolid juba lõpetanud, on võimalik osaleda erafirmade korraldatud võistlustel, nagu Google Code Jam või Facebook Hacker Cup. Peale selle on olemas hulk internetipõhiseid võistluskeskkondi, kus toimuvad regulaarsed üritused. 2017 alguse seisuga on neist tuntumad CodeForces, TopCoder, HackerRank ja CodeChef.

Programmeerimisvõistlustel on üldjuhul ajapiirang (näiteks 2-5 tundi), mille jooksul on antud lahendamiseks hulk ülesandeid (enamasti 3-6). Ülesannetele antakse sisendandmed, mille põhjal tuleb leida väljundandmed, mille õigsust kontrollitakse.

Lahendus võib olla iseseisev programm (sel juhul loeb ta sisendit ja väljundit failist või standardsisendist ja kirjutab väljundi teise faili või standardväljundisse) või siis lihtsalt funktsioon, mis lingitakse testprogrammi külge ja mida testprogramm välja kutsub, andes sisendi ette funktsiooni parameetritena.

Suurim väljakutse on sageli mitte lihtsalt vastuse leidmiseks sobiva lahenduse leidmine, vaid sobiva **kiire** lahenduse leidmine. Lahendustel on antud ajalimiit, mille jooksul tuleb vastus väljastada, tänapäeval tavaliselt sekund või paar.

Siin on üks lihtne näide võistlusstiilis ülesandest:

Malelaual peab ratsu liikuma N käiguga ühelt etteantud väljalt teisele. Mitu erinevat teekonna võimalust tal selleks on?

Kui võistleja on lahenduse valmis programmeerinud, esitab ta selle testimiseks. Olenevalt võistluse formaadist võib testimine toimuda kas kohe või lahendamisaja lõpus. Kohese testimise puhul

öeldakse võistlejale, kas programm läbis testid või mitte, nii et tal on võimalus oma lahendust parandada.

Testimisel on üldjuhul neli võimalikku tulemust:

- OK – kõik töötas õigesti.
- Vale vastus – programm ei väljastanud oodatud tulemust.
- Ajalimit ületatud – programm ei lõpetanud ajapiiri jooksul oma tööd.
- Käivitusaja viga – programmi töös juhtus viga (näiteks mittelubatud mälu piirkonnast lugemine või nulliga jagamine), mis ei lasknud tal tööd lõpetada.

Mõnedel võistlustel on ülesande eest punktide saamiseks vaja korrektselt läbida kõik testid, teistel saab vastavalt läbitud testide arvule osalise arvu punkte. Suurima punktide arvuga võistleja saab auhinna või siis lihtsalt au ja kuulsust.

Selle teksti kirjutamise ajal on maailma parim võistlusprogrammeerija Gennadi Korotkevitš Valgevenest, kes praegu esindab St.Peterburgi ITMO ülikooli. Korotkevitš hakkas olulisi võistlusi võitma juba 11-aastaselt, sai IOI-lt kokku kuus kuldmedalit ning on hetkel olulisematel võistlussaitidel maailma kõrgeima reitinguga.

Eesti seni edukaim võistlusprogrammeerija on olnud Martin Pettai, kes sai aastatel 1999-2002 IOI-lt kolm kuld- ja ühe hõbemedali.

0.2 MIDA SIIT ÕPIKUST LEIDA VÕIB?

Käesoleva õpiku peamine eesmärk on ehitada sild programmeerimisoskuse ja teoreetilise arvutiteaduse vahele. Eesti keeles on mitmesuguseid programmeerimist õpetavaid raamatuid, kus keskendutakse programmeerimistehnikale. Samuti on ilmunud arvutiteaduse sissejuhatavaid õpikuid, kus räägitakse mitmesugustest teoreetilistest küsimustest.

Ka isiklikult olen ma elus kohanud paljusid inimesi, kes jäävad kas ühte või teise serva: ühel pool on puhtad praktikud, kes pole põhjalikumalt teooriat omandanud või on selle unustanud kui „mittevajaliku“. Teisel pool on teoretikud, kes tunnevad paljusid algoritme, kuid kellele kuluks ära rohkem praktilist kogemust nende realiseerimisel.

Siin raamatus tuuakse teoreetiline ja praktiline pool kokku ja räägitakse mõlemast. Käsitletakse nii võtteid efektiivsemaks programmeerimiseks kui ka algoritme, mis võimaldavad esmapilgul võimatuid ülesandeid mõistliku ajaga lahendada.

Tutvustatakse mitmeid algoritme ja andmestruktuure, millest mõned kuuluvad arvutiteaduse parimate saavutuste sekka, kuid teoreemide ja tõestuste asemel on siin hulk praktiliselt läbiprogrammeeritavaid näiteid. Raamatuga kaasas olevas lisas on võimalik tutvuda paljude näidisprogrammidega ja lahendada mitmesuguse raskusega katseülesandeid enesekontrolliks.

Õpiku ülesehitus võimaldab seda kasutada struktureeritud kursustel, kus õpetatakse programmeerimist või arvutiteadust üldiselt või siis tegeletakse spetsiifiliselt programmeerimisvõistlusteks ettevalmistamisega. Teiselt poolt võiks see olla kasutatav praktilise käsiraamatuna, kust on vajadusel võimalik ühe või teise algoritmi kohta infot leida.

Tegemist ei ole programmeerimise algkursusega, eelduseks on, et lugejad on algtasemel programmeerimist juba õppinud ning oskavad oma lemmikkeeles põhiasjadega toime tulla.

Raamatust võivad kasu saada mitmesugused huvilised:

Esiteks **kooliõpilased**, kellel on olemas teadmised programmeerimise põhialustest ja kes soovivad teistega mõõtu võtta. Raamat sisaldab suurel hulgal teadmisi, mida on vaja informaatikaolümpiaadil. 2016. aasta lõpu seisuga on raamatu esimese osa läbitöötamine piisav, et olla edukas Eesti olümpiaadidel ja teine osa võimaldaks koju tuua medali rahvusvahelistelt võistlustelt.

Teiseks **üliõpilased ja kraadiõppurid**, kes soovivad täiendada oma teadmisi arvutiteaduses, neid samal ajal praktiliste oskustega seostades. Õpik pakub mitmeid kasulikke lähenemisviise, mis võimaldavad uurimistöodes vajalikke programme efektiivsemalt kirjutada.

Kolmandaks **professionaalsed programmeerijad**, kes soovivad omandada algoritmiliselt keerulisemate ülesannete lahendamise oskust. Võistlustel ja olümpiaadidel programmeerimine on nagu rallisõit, kus läheb vaja spetsiifilisi oskusi. Igapäevane tavaprogrammeerimine mõnes ettevõttes on selle kõrval nagu liinibussi või takso juhtimine – sarnaste elementidega, aga siiski mitte päris sama. On aga olukordi, kus rallioskused kuluvad ka igapäevaelus marjaks ära. Olen ise oma karjääri jooksul nii mõnelgi korral suutnud „võimatut“ teha ja mõne asja kümneid või sadu kordi kiiremini tööle panna, samal ajal kui teised olid juba valmis massiivseks täiendava riistvara ostuks. Ja kes ei tahaks vahel kangelane olla :)

Neljandaks **õpetajad ja õppejõud**, kes soovivad oma õpilasi olümpiaadideks ja võistlusteks ette valmistada. Raamat on üles ehitatud struktuurilt, iga järgnev peatükk kasutab ja täiendab eelmistes õpitut. Tekstis on palju ülesandeid, mida võib anda lahendamiseks nii grupidena kui individuaalselt.

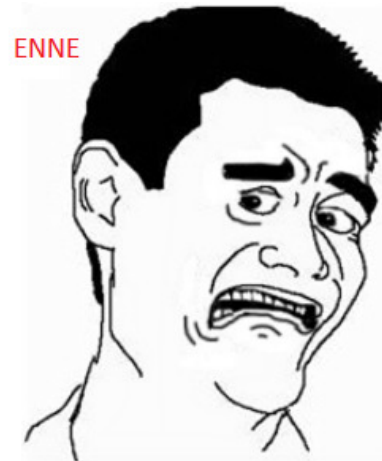
Viiendaks **ambitsioonikad spetsialistid**, kes soovivad minna tööle mõnesse maailma „kõrgliiga“ tehnoloogiaettevõttesse. Ettevõtetes nagu Google, Facebook või Palantir on tavaline, et tööintervjuudel antakse kandidaatidele ülesandeid näiteks graafiteooria või dünaamilise planeerimise vallast. Seega on intervjuu läbimiseks vaja ka võistlusprogrammeerimise alaseid teadmisi. Eestis pole nii põhjalik valmistumine tööintervjuudeks veel väga levinud, aga maailma tippasemel juhtub sageli, et kandidaadid õpivad intervjuudeks nädalaid või isegi kuid.

Ja lõpuks kõik ülejäänud, kellele meeldib programmeerimine ning teadmiste omandamine.

Raamat on jagatud kaheks osaks ja neist kumbki kuueks peatükiks, millest igaüks käsitleb konkreetset teemat.

Iga peatükk sisaldab järgmisi osi:

1. Tekst, mis avab teema olemust ja teoreetilist tausta.
2. Näidisülesanded, kus põhjalikult selgitatakse, kuidas kirjutada programme vastavate algoritmide realiseerimiseks.
3. Veel mõned ülesanded, kus on antud lahenduse idee ja viited näidislahendustele.
4. Ülesanded enesekontrolliks (või kontrollitöödeks akadeemilises keskkonnas). Kõigi ülesannete lahendusi saab esitada *online* võistluskeskkonnas, kus neid automaatselt testitakse.



Kõik näited on saadaval kolmes programmeerimiskeeles:

- C++ kui võistlustel ja olümpiaadidel enim kasutatav keel
- Java kui Eesti tarkvaratööstuses enim kasutatav keel
- Python kui hariduses ja programmeerimise õppimisel populaarne keel.

Tekstis toodud näited kasutavad C++ varianti, teistes keeltes kirjutatud vasted on saadaval lisamaterjalina.

0.3 KONTROLLÜLESANNETE LAHENDAMINE

Iga peatüki lõpus on toodud rida kontrollülesandeid. Neid on võimalik lahendada omal käel või kui raamatut kasutatakse õppetöös, saab kursuse juhendaja korraldada nende põhjal kontrolltöö.

Ülesandeid saab lahendada online võistluskeskkonnas CodeForces, kus:

- Igale ülesandele on üles seatud automaatsed testid.
- Kasutajad saavad veebibrauseri kaudu esitada oma koodi ning saavad kohest tagasisidet selle kohta, kas programm läbis testid või ei.

Ülesannete lahendamiseks:

- Loo CodeForces'i konto (või logi sisse Google'i kontoga)
- Ühine Eesti Võistlusprogrammeerimise grupiga aadressil <http://codeforces.com/group/jODaK73gf0/>
- Kontrolltööde jaoks on loodud võistlused, vali neist asjakohane

Lahendused peavad lugema andmed standardsisendist ja kirjutama vastuse standardväljundisse. Kui väljundisse kirjutada veel midagi muud, on tulemuseks tõenäoliselt *Wrong Answer*. Kõik testid vastavad ülesande kirjelduses toodud spetsifikatsioonile ja nende korrektsust eraldi kontrollima ei pea.

0.4 KASUTATUD ÜLESANDED JA PILDID

Näidisülesanded kuuluvad enamikus arvutiteaduse klassikasse loodud spetsiaalselt käesoleva raamatu jaoks, osad on ka taaskasutatud mõnelt Eesti olümpiaadilt. Teistest allikatest pärit ülesanded on vastavalt viidatud.

Kontrolltööde ülesanded on loodud kas selle raamatu jaoks või kohandatud teiste riikide võistlustelt. Kuna enamasti on teiste võistluste ülesanded toodud koos lahendustega, pole neid raamatust otse viidatud, viite saab küsimise peale autoritelt.

Kasutatud pildid on võetud vabadest pildipankadest nagu Pixabay, Pexels, Wikimedia Commons jt, kus nad on antud maailmale kasutamiseks CC0 litsentsi alusel.

0.5 AUTORITEST

Targo õppis programmeerima aastal 1990, mil tal oli ainult tekstirežiimi võimaldav kasutatud IBM XT arvuti, millel polnud mängu, mida mängida. Lahenduseks oli ise mängud kirjutada, mis eeldas omakorda programmeerimise äraõppimist. Pärast seda on ta programmeerinud igasuguseid asju klaasilõikepinkidest ja lennuki elektrisüsteemidest dokumendihalduse ja mobiilirakendusteni. Praegu

juhhib Targo ka Eesti Informaatikaolümpiaadi. Kuna talle meeldib rohkem endast rääkida, tähistab tekstis „mina“ enamasti Targot.

Katrin huvitus mängudest ise vähem, kuid kirjutas neid oma nooremale vennale. Hiljem on ta õppinud arvutuslingvistikat ja töötanud süsteemianalüütiku ning programmeerijana, samuti tegelenud võistlusprogrammeerimise õpetamisega Eesti Informaatikaolümpiaadi raames.

0.6 TÄNUAVALDUSED

Tahame tänada paljusid inimesi, kes raamatu valmimisele kaasa aitasid:

- Mihkel Kree ja Jaak Vilo organisatsioonilise toetuse eest,
- Härmel Nestra, Tähvend Uustalu, Liisa Jõgiste, Oliver-Matis Lill, Tiina Kull ja Heno Ivanov sisuliste soovitude ja veaparanduste eest,
- Toomas Tennisberg ja Oliver Tennisberg ülesannete läbilahendamise ja testide koostamise ning kontrollimise eest.

0.7 VEAPARANDUSED JA SOOVITUSED

Tegemist on elava dokumendiga, millest loodame aja jooksul uusi versioone luua. Valdkond areneb kiiresti, samuti võib esile tulla vajadusi materjali täpsustamiseks, mitmesuguseid vigu jne.

Kõik parandused ja soovited võib saata aadressil targot@gmail.com.