

Võistlusprogrammeerimine

ehk kuidas kirjutada kiiret koodi

Targo Tennisberg, Katrin Gabrel

0	Sissejuhatus.....	12
0.1	Mis on võistlusprogrammeerimine?	12
0.2	Mida siit õpikust leida võib?.....	13
0.3	Kontrollülesannete lahendamine	15
0.4	Kasutatud ülesanded ja pildid	15
0.5	Autoritest.....	15
0.6	Tänuavaldused.....	16
0.7	Veaparandused ja soovitused	16
1	Programmide sisemaailm.....	17
1.1	Programmi elutsükkel.....	18
1.1.1	Lähtekood.....	18
1.1.2	Lähtekoodi transleerimine	18
1.1.3	Teegid	19
1.1.4	Linkimine	20
1.1.5	Laadimine ja täitmine	20
1.2	Andmete hoidmine ja töötlemine arvutis	20
1.2.1	Protsessori tööpõhimõte.....	20
1.2.2	Protsessori jõudlus	21
1.2.3	Käsukskonveier	22
1.2.4	Hargnemise ennustamine	23
1.2.5	Andmete liikumine	25
1.2.6	Mälu.....	25
1.3	Andmetüübid.....	27
1.3.1	Täisarvud	27
1.3.2	Ujukomaarvud	28
1.3.3	Püsikomaarvud	33
1.3.4	Märgid	33
1.3.5	Tõeväärtused.....	33
1.3.6	Viidad.....	34
1.3.7	Struktuursed tüübid ehk liitüübid	34
1.3.8	Kirjed.....	35
1.3.9	Stringid.....	35
1.3.10	Massiivid.....	35
1.4	Operatsioonid stringidega	36
1.4.1	Märk kohal i.....	36
1.4.2	Stringide võrdlemine	36
1.4.3	Stringide ühendamise ehk liitmine	37
1.5	Sisend-väljund	39
1.5.1	Standardvood	39

1.5.2	Failidest lugemine ja kirjutamine	40
1.5.3	Sisendi töötus	41
1.5.4	Väljund.....	42
1.5.5	Jooksvalt töötamine	43
1.6	Programmeerimiskeelte võrdlus	44
1.6.1	C++	44
1.6.2	Java	45
1.6.3	Python	46
1.7	Testimine	48
1.7.1	Piirjuhud	48
1.7.2	Õigsuse kontroll.....	49
1.7.3	Algoritmi jõudluse kontroll.....	50
1.8	Silumine	51
1.8.1	Arenduskeskkonnad	51
1.8.2	Aja mõõtmine	52
1.9	Kontrollülesanded	53
1.9.1	Järelmaks	54
1.9.2	Kell	54
1.9.3	Tigu	55
1.9.4	3D printer	55
1.9.5	Mõttemeister	56
1.9.6	Male.....	56
1.9.7	Riigihanked	57
1.9.8	Bender	58
1.9.9	Interpretaator.....	58
1.9.10	Pangatellerid.....	59
1.10	Viited lisamaterjalidele.....	60
2	Läbivaatus- ja otsingualgoritmid	61
2.1	Alamprogrammid	61
2.1.1	Pinumälu.....	61
2.1.2	Funktsiooni kohalikud andmed	62
2.1.3	Pinu ületäitumine	62
2.1.4	Pinu kasutamine protsessoris.....	63
2.1.5	Kuhimälu.....	64
2.1.6	Funktsiooni parameetrid	65
2.1.7	Objektide edastamine parameetritena	66
2.2	Rekursioon.....	68
2.2.1	Hargnemiseta rekursioon	68
2.2.2	Rekursioonivalem	69
2.2.3	Hargnemistega rekursioon	70
2.2.4	Sabarekursioon ja väljakutsete optimeerimine.....	72
2.3	Variantide läbivaatamine	74
2.3.1	Tagurdusmeetod	74
2.3.2	Iteratsioonimeetod.....	74
2.3.3	Tagurduse ja iteratsiooni võrdlus.....	75
2.3.4	Variantide läbivaatamine tagurdusmeetodiga.....	75
2.3.5	Variantide läbivaatamine iteratsioonimeetodiga.....	76
2.3.6	Variantide läbivaatus programmeerimisvõistlustel	76

2.4	Läbivaatuse optimeerimine.....	77
2.4.1	Lihtne tagurdusmeetod.....	77
2.4.2	Andmete optimeerimine.....	78
2.4.3	Ettearvutamine.....	79
2.4.4	Otsingupuu ahendamine.....	80
2.4.5	Sisemiste tsüklite optimeerimine.....	81
2.4.6	Andmestruktuuride optimeerimine.....	82
2.4.7	Rekursiooni eemaldamine.....	84
2.5	Kahendotsing.....	85
2.5.1	Vastuse kahendotsing.....	85
2.5.2	Kahendotsing andmetest.....	87
2.5.3	Topeltkahendotsing.....	88
2.5.4	Otsing kahemõõtmelisest tabelist sadulameetodil.....	89
2.6	Jaga ja valitse.....	93
2.7	Sissejuhatus kombinatoorikasse.....	94
2.7.1	Permutatsioonid.....	94
2.7.2	Permutatsioonide konstrueerimine.....	95
2.7.3	Kombinatsioonid.....	97
2.8	Kontrollülesanded.....	99
2.8.1	Autoturg.....	99
2.8.2	Kaupmees ja maksukoguja.....	100
2.8.3	Lippude vasturünnak.....	100
2.8.4	Akulaadija.....	101
2.8.5	Pildi pakkimine.....	101
2.8.6	Stern-Brocot' puu.....	102
2.8.7	Viinamarjad.....	103
2.8.8	Erinevad summad.....	104
2.8.9	Kodumasinat näidikud.....	105
2.8.10	Graafi värvimine.....	106
2.9	Viited lisamaterjalidele.....	106
3	Algoritmi keerukus ja põhilised andmestruktuurid.....	107
3.1	Algoritmi keerukus.....	107
3.1.1	Keskmine ja halvim keerukus.....	108
3.1.2	Asümptootiline hinnang.....	108
3.1.3	Kasvuseoste omadused.....	110
3.2	Algoritmi keerukuse hindamine.....	110
3.2.1	Hargnemiste keerukuse hindamine.....	110
3.2.2	Tsüklite keerukuse hindamine.....	111
3.2.3	Mitmekordsete tsüklite keerukus.....	111
3.2.4	Rekursiooni keerukuse hindamine.....	112
3.3	Tavalised keerukusklassid.....	112
3.3.1	Keerukusklass ja ajalimiit.....	114
3.4	Andmestruktuurid.....	114
3.4.1	Andmestruktuuride klassifitseerimine.....	115
3.5	Massiiv.....	115
3.5.1	Massiivid Pythonis.....	115
3.5.2	Massiivi võimalused ja piirangud.....	116
3.5.3	Mitmemõõtmelised massiivid.....	116

3.5.4	Dünaamilised massiivid	117
3.6	Ahel.....	117
3.7	Pinu.....	119
3.8	Järjekord	120
3.8.1	Kaheotsaline järjekord.....	121
3.8.2	Eelistusjärjekord.....	121
3.9	Pinu ja järjekorra kasutamine.....	121
3.10	Kahendpuu	125
3.10.1	Kahendpuu esitus.....	126
3.10.2	Kahendotsingu puu.....	126
3.11	Kujutis.....	126
3.11.1	Kujutis programmeerimiskeeltes	127
3.12	Praktiline ülesanne	128
3.13	Kuhi.....	129
3.14	Sortimine	130
3.14.1	Mullimeetod	131
3.14.2	Valikmeetod	132
3.14.3	Pistemeetod	133
3.14.4	Põimemeetod	133
3.14.5	Kiirmeetod	135
3.14.6	Võrdlustel põhineva sortimise keerukuse alampiir	136
3.15	Sortimise erimeetodid.....	137
3.15.1	Loendamismeetod.....	137
3.15.2	Positsioonimeetod.....	138
3.15.3	Kimbumeetod.....	138
3.16	Mitme kriteeriumi järgi sortimine	138
3.17	Programmeerimiskeelte standardtegid	138
3.17.1	C++	139
3.17.2	Java	139
3.17.3	Python	139
3.18	Kontrollülesanded	141
3.18.1	Vabrikud	141
3.18.2	Jalgpalliturniir	142
3.18.3	Erdöse arvud.....	143
3.18.4	Programmeerimisvõistlus.....	144
3.18.5	Pannkoogid.....	144
3.18.6	Kaartide segamine.....	145
3.18.7	Kass klaviatuuril.....	145
3.18.8	Pinugrammid	146
3.18.9	Parv.....	147
3.18.10	Ahelmurrud	148
3.19	Viited lisamaterjalidele.....	148
4	Arvuteooria.....	149
4.1	Jaguvus ja jääk	149
4.1.1	Jaguvus	149
4.1.2	Jääk	150
4.1.3	Jäägi leidmine programmeerimiskeeltes.....	150
4.1.4	Ülesanne: loosiümbrikud.....	151

4.1.5	Arvutamine moodulitega.....	152
4.1.6	Ülesanne: anagrammid 2.....	154
4.2	SÜT ja VÜK.....	156
4.2.1	Eukleidese algoritm suurima ühisteguri leidmiseks.....	157
4.2.2	Vähima ühiskordse leidmine.....	157
4.2.3	Ülesanne: hammasrattad.....	158
4.2.4	Diofantilised võrrandid.....	159
4.3	Algarvud.....	160
4.3.1	Eratosthenese sõel.....	160
4.3.2	Algarvulisuse kontroll.....	162
4.3.3	Ülesanne: algarvu-Scrabble.....	164
4.3.4	Arvu algtegurid.....	167
4.4	Positsioonilised arvusüsteemid.....	168
4.4.1	Süsteemide vahel teisendamine.....	168
4.4.2	Ülesanne – positsioonilised arvusüsteemid.....	170
4.4.3	Arvutamine kahendsüsteemis.....	175
4.5	Suurte arvudega arvutamine.....	175
4.5.1	Suurte arvude esitus.....	175
4.5.2	Suurte arvude liitmine ja lahutamine.....	176
4.5.3	Suurte arvude korrutamine ja astendamine.....	178
4.5.4	Efektiivne astendamine.....	179
4.5.5	Suurte arvude jagamine.....	180
4.5.6	Ülesanne – anagrammid 3.....	180
4.5.7	Suured arvud erinevates programmeerimiskeeltes.....	181
4.5.8	Logaritm ja eksponent suurte arvudega arvutamisel.....	182
4.6	Kontrollülesanded.....	184
4.6.1	Suur arv.....	184
4.6.2	Faktoriaali lõpp.....	184
4.6.3	Goldbachi hüpotees.....	185
4.6.4	Klaaskuulid.....	185
4.6.5	VÜK-võimsus.....	186
4.6.6	Suurim algtegur.....	186
4.6.7	Vahetusraha.....	187
4.6.8	Taandumatud murrud.....	187
4.6.9	Mertensi funktsioon.....	188
4.6.10	Mõrvamüsteerium.....	188
4.7	Viited lisamaterjalidele.....	189
5	Dünaamiline planeerimine algajatele.....	190
5.1	Sissejuhatav ülesanne – Fibonacci jada.....	190
5.1.1	Tavaline rekursioon ehk jõumeetod.....	191
5.1.2	Mäluga rekursioon.....	192
5.1.3	Alt üles DP.....	193
5.1.4	Kokkuhoidlik DP.....	194
5.1.5	DP retsept.....	195
5.2	Lineaarne väärtuste tabel - optimaalne maksimine.....	195
5.2.1	Ahne algoritm.....	195
5.2.2	Kõigi variantide läbivaatamine (rekursioon).....	196
5.2.3	DP lahendus.....	197

5.3	Pikima kasvava osajada leidmine	199
5.3.1	Kõigi võimaluste läbivaatus	199
5.3.2	DP lahendus.....	200
5.3.3	Tagurdusmeetodiga lahendus.....	201
5.4	Kahemõõtmeline väärtuste tabel – pikim ühine osajada.....	202
5.4.1	DP lahendus.....	202
5.5	Ristsummade loendamine.....	204
5.5.1	Ristsummade arvutamine	204
5.5.2	DP lähenemine	204
5.5.3	DP Exceliga.....	205
5.5.4	Tabeli koostamine programselt.....	206
5.5.5	Ülesande lahendus (tabeli kasutamine).....	207
5.6	Mitmemõõtmeline DP tabel – Miljonär ja vaeslapsed.....	208
5.6.1	Kõikide võimaluste läbivaatus	209
5.6.2	Korduvad harud	209
5.6.3	DP tabeli koostamine	210
5.6.4	Lahendus	211
5.7	Tõeväärtuste tabeliga DP - õiglase jagamine.....	213
5.7.1	Kõik kombinatsioonid	213
5.7.2	DP lahendus.....	213
5.8	Bitimaskide põhine väärtuste tabel - moekunstnik ja koiliblikas	215
5.8.1	Kõigi läbivaatuste puu	216
5.8.2	Tee DP poole.....	217
5.8.3	Bitimaskide kasutamine tabeli indeksitena.....	217
5.8.4	DP lahendus.....	219
5.9	Kuidas ja millal DP-d kasutada.....	220
5.9.1	Ülalt alla vs alt üles DP.....	221
5.9.2	Kidunud DP	221
5.10	DP praktilised kasutusalaad	222
5.11	Kontrollülesanded	223
5.11.1	Aktsiaturg	223
5.11.2	Protsessori planeerimine.....	223
5.11.3	Maksmise võimalused	224
5.11.4	Statistika manipuleerimine.....	224
5.11.5	Lennukikütu	225
5.11.6	Kahjuritõrje.....	226
5.11.7	Torni ehitamine	227
5.11.8	Putin sukeldumas	228
5.11.9	Arvude liitmine	228
5.11.10	Templisambad	229
5.12	Viited lisamaterjalidele.....	229
6	Sissejuhatus graafiteooriasse	230
6.1	Graafiteooria terminid.....	231
6.1.1	Suunatud ja suunamata servad	231
6.1.2	Teed graafis	231
6.1.3	Sidususkomponendid	232
6.2	Graafide esitusviisid	232
6.2.1	Naabrusmaatriks	233

6.2.2	Tippude loend.....	233
6.2.3	Servade loend.....	234
6.2.4	Regulaarsete servadega graafid	235
6.3	Graafi läbimine	235
6.3.1	Sügavuti läbimine	235
6.3.2	Ülesanne: Ratsu teekond.....	237
6.3.3	Laiuti läbimine	240
6.3.4	Ülesanne: Ratsu teekond 2.....	241
6.4	Sidususkomponentide leidmine	243
6.4.1	Ülesanne: Ratsud.....	243
6.5	Üleujutamine.....	245
6.5.1	Ülesanne: Veekogud.....	245
6.6	Kahealuseline graaf	246
6.7	Topoloogiline sorteerimine	247
6.7.1	Ülesanne: Loomad.....	248
6.8	Kaalutud servadega graafid.....	250
6.9	Dijkstra lühima tee leidmise algoritm	250
6.9.1	Ülesanne: sõiduaeg	253
6.10	Puud.....	255
6.10.1	Puud kui graafid.....	255
6.10.2	Puu definitsioon	255
6.10.3	Graafitöötusalgoritmid puul	255
6.10.4	Kahendpuu sügavuti läbimine	256
6.10.5	Puu laiuti läbimine.....	257
6.10.6	Ülesanne: Kahendpuud	257
6.11	Kontrollülesanded	260
6.11.1	Robotivõistlus	260
6.11.2	Civilization	261
6.11.3	Dominod.....	261
6.11.4	Projektiplaan	262
6.11.5	Sõnateisendused	262
6.11.6	Kahevärviprobleem	263
6.11.7	Joonejälgija robot.....	264
6.11.8	Numbriruudud	265
6.11.9	Optimaalne ruuting	266
6.11.10	Liftid.....	267
6.12	Viited lisamaterjalidele.....	268
7	Efekttiivne programmeerimistehnika.....	269
7.1	Koodistiil.....	270
7.1.1	Pikaajaline ja lühiajaline lähenemine	270
7.1.2	Alamprogrammid.....	270
7.1.3	Näide: Tankid.....	272
7.1.4	Kommentaarisid.....	276
7.1.5	Nimed	278
7.1.6	Funktsioonide nimetamine.....	279
7.2	Testimine	279
7.2.1	Testide koostamine	280
7.2.2	Übermööd	280

7.2.3	Testide genereerimine	283
7.2.4	Käsuinterpretaatorid ja skriptimine	285
7.2.5	Valideerimine	287
7.3	Tunne oma keelt.....	288
7.3.1	Teegid	288
7.3.2	Andmestruktuurid	288
7.3.3	Makrod	289
7.4	Võistluste strateegia.....	290
7.5	Ahned algoritmid.....	290
7.5.1	Mündid	291
7.5.2	Kingid	292
7.5.3	Majakavahid	295
7.5.4	Veel ahnetest algoritmidest	297
7.5.5	Suveniirid.....	297
7.6	Kontrollülesanded	301
7.6.1	Onu Robert	301
7.6.2	Bussijuhid.....	302
7.6.3	Hernehirmutised	302
7.6.4	Kolimine.....	303
7.6.5	Krokodillid.....	303
7.6.6	Lohe Gorõnitš	304
7.6.7	Fraktsioonid.....	305
7.6.8	Bitimask	305
7.6.9	LED-lambid.....	306
7.6.10	Antimonotoonne jada	307
7.7	Viited lisamaterjalidele.....	307
8	Dünaamiline planeerimine edasijõudnutele	308
8.1	Seljakoti pakkimine.....	308
8.1.1	Ülesanne: Kohver	308
8.1.2	Dünaamiline planeerimine	309
8.1.3	Rekursioonivalem	309
8.1.4	Tabeli täitmine.....	309
8.1.5	Kood.....	310
8.1.6	Tee taastamine	310
8.1.7	Keerukus.....	311
8.1.8	Seljakoti erinevad variatsioonid	311
8.2	Rändkaupmehe ülesanne	311
8.2.1	Ülesanne: kaupmees	312
8.2.2	Rekursioonivalem	312
8.2.3	Valemi kasutus.....	312
8.2.4	DP tabel	313
8.2.5	Kood.....	314
8.2.6	Tee taastamine	315
8.2.7	Keerukus.....	315
8.3	Levenshteini kaugus	315
8.3.1	Ülesanne: Levenshteini kaugus	316
8.3.2	Rekursioonivalem	317
8.3.3	DP tabel	317

8.3.4	Teisenduse taastamine tabelist.....	318
8.3.5	Ajaline ja mahuline keerukus	318
8.3.6	Levenshteini kauguse omadused	319
8.3.7	Levenshteini kauguse variatsioonid	319
8.4	Geenijärjestuste võrdlemine	319
8.4.1	Ülesanne: Sarnasus.....	320
8.4.2	Ülesanne: Mis on ühist?	321
8.4.3	Rekursioonivalem	321
8.4.4	Tabel	322
8.4.5	Kood.....	322
8.4.6	Tee taastamine	322
8.5	Dünaamiline Aja painutamine (Dynamic Time Warping)	323
8.5.1	Ülesanne: helid.....	323
8.5.2	Rekursioonivalem	324
8.5.3	Tabel	324
8.5.4	Kood.....	324
8.6	Lõikudega ülesanded.....	325
8.6.1	Ülesanne: palgi lõikamine	325
8.6.2	Tee DP-ni	325
8.6.3	Rekursioonivalem	325
8.6.4	Tabeli täitmine.....	325
8.6.5	Kood.....	327
8.6.6	Tee taastamine	327
8.6.7	Keerukus.....	328
8.7	Maatriksite korrutamine	328
8.7.1	Ülesanne: maatriksid.....	328
8.7.2	Võimaluste arv.....	328
8.7.3	Tee DPni.....	329
8.7.4	Tabeli täitmine.....	329
8.7.5	Kood.....	330
8.7.6	Keerukus.....	330
8.8	Ülesanne: mäed.....	331
8.8.1	Rekursioonivalem	332
8.8.2	Tabeli täitmine.....	333
8.8.3	Kood.....	333
8.8.4	Keerukus.....	334
8.9	Kontrollülesanded	335
8.9.1	Torn	335
8.9.2	Bussireisid.....	336
8.9.3	Sillad	337
8.9.4	Mäng.....	338
8.9.5	Uuring.....	339
8.9.6	Alkeemik.....	340
8.9.7	Puhkus	341
8.9.8	Tango	342
8.9.9	Valimised	343
8.9.10	Plaan	344
8.10	Viited lisamaterjalidele.....	344

9	Graafiteooria	345
9.1	Floyd-Warshalli algoritm	345
9.1.1	Ülesanne: tuleb hiljem.....	345
9.1.2	Rekursioonivalem	346
9.1.3	Tabeli täitmine.....	346
9.1.4	Floyd-Warshalli algoritmi kasutusi	348
9.2	Toese mõiste	349
9.2.1	Servade liigitamine	349
9.2.2	DFS koos servade liigitamisega.....	350
9.3	Sillad ja Artikulatsioonipunktid.....	351
9.3.1	Ülesanne: Muinastulede öö	352
9.3.2	Sildade leidmine	356
9.4	Euleri Graaf.....	357
9.4.1	Euleri tsükli leidmine	357
9.4.2	Loomanimed.....	358
9.5	Minimaalne Toes	360
9.5.1	Minimaalse toese leidmine	361
9.5.2	Ülesanne: Elektrikatkestus	362
9.6	Maksimaalne voog ja minimaalne lõige	364
9.6.1	Võrk	364
9.6.2	Voog.....	365
9.6.3	Maksimaalne voog.....	365
9.6.4	Maksimaalse voo leidmine	366
9.6.5	Mitme lähte ja suudmega ülesanded	368
9.6.6	Kaaludega tipud võrgus	368
9.6.7	Lõige	369
9.6.8	Vangid.....	370
9.7	klikid	373
9.7.1	Bron-Kerboschi algoritm.....	373
9.7.2	Sõbrad.....	374
9.8	Kontrollülesanded	376
9.8.1	Laadimispunktid	376
9.8.2	Internetiühendus.....	377
9.8.3	Kiviheitemasin	378
9.8.4	Uus tee.....	379
9.8.5	Mõjuvõim	380
9.8.6	Civilization	381
9.8.7	Põgenemine.....	382
9.8.8	Teepuhastus	383
9.8.9	Doomino	383
9.8.10	Valimised	384
9.9	Viited lisamaterjalidele.....	384
10	Andmestruktuurid edasijõudnutele	385
10.1	Fenwicki puu.....	385
10.2	2D Fenwicki puu	385
10.3	Lõikude puu	385
10.4	Laisk väärtustamine lõikude puudes	385
10.5	O(log n) operatsioonid puudel	385

10.6	Kontrollülesanded	385
11	Tekstialgoritmid.....	386
11.1	Teksti parsimine	386
11.2	Räsid	386
11.3	Knuth-Morris-Pratti algoritm.....	386
11.4	Aho-Corasicki algoritm	386
11.5	Sufiksipuud	386
11.6	Kontrollülesanded	386
12	Arvutusgeomeetria.....	387
12.1	Samal joonel asumise kontroll	387
12.2	Paralleelsuse ja samasihilisuse kontroll	387
12.3	Lõikude lõikumine	387
12.4	Kujundi pindala.....	387
12.5	Punkti sisaldumine kujundis	387
12.6	Koordinaatide pakkimine	387
12.7	Kumer kate	387
12.8	Sweeping line	387
12.9	Bresenhami algoritm	387
12.10	Magic missile	387
12.11	Kontrollülesanded	387

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:

Malelaua 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.
- Ajalimiit 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 läbi programmeeritavaid 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 gruppitööna 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.