

Kisakoodarin käsikirja

Antti Laaksonen

4. joulukuuta 2016

Sisältö

Alkusanat	ix
I Perusasiat	1
1 Johdanto	3
1.1 Ohjelmointikielet	3
1.2 Syöte ja tuloste	4
1.3 Lukujen käsittely	6
1.4 Koodin lyhentäminen	8
1.5 Virheen etsiminen	10
1.6 Matematiikka	11
2 Aikavaativuus	15
2.1 Laskusäännöt	15
2.2 Vaativuusluokkia	18
2.3 Tehokkuuden arviointi	19
2.4 Suurin alitaulukko	20
3 Järjestäminen	23
3.1 Järjestämisen teoriaa	23
3.2 Järjestäminen C++:ssa	27
3.3 Binäärihaku	29
4 Tietorakenteet	33
4.1 Dynaaminen taulukko	33
4.2 Joukkorakenne	35
4.3 Hakemisto	36
4.4 Iteraattorit ja välit	37
4.5 Muita tietorakenteita	39
4.6 Vertailu järjestämiseen	41
5 Täydellinen haku	43
5.1 Osajoukkojen läpikäynti	43
5.2 Permutaatioiden läpikäynti	45
5.3 Peruuttava haku	46
5.4 Haun optimointi	47
5.5 Puolivälihaku	49

6	Ahneet algoritmit	51
6.1	Kolikkotehtävä	51
6.2	Aikataulutus	52
6.3	Tehtävät ja deadlinet	54
6.4	Keskiluvut	55
6.5	Huffmanin koodaus	56
7	Dynaaminen ohjelmointi	59
7.1	Kolikkotehtävä	59
7.2	Pisin nouseva alijono	64
7.3	Reitinhaku ruudukossa	65
7.4	Repunpakkaus	66
7.5	Editointietäisyys	68
7.6	Laatoitukset	69
8	Tasoitettu analyysi	71
8.1	Kaksi osoitinta	71
8.2	Lähin pienempi edeltäjä	74
8.3	Liukuvan ikkunan minimi	75
9	Välikyselyt	77
9.1	Staattiset kyselyt	77
9.2	Binääri-indeksipuu	80
9.3	Segmenttipuu	83
9.4	Lisäteknikoita	87
10	Bittien käsittely	89
10.1	Luvun bittiesitys	89
10.2	Bittioperaatiot	90
10.3	Joukon bittiesitys	92
10.4	Dynaaminen ohjelmointi	94
II	Verkkoalgoritmit	97
11	Verkkojen perusteet	99
11.1	Käsitteitä	99
11.2	Verkko muistissa	103
12	Verkon läpikäynti	107
12.1	Syvyyshaku	107
12.2	Leveyshaku	109
12.3	Sovelluksia	111
13	Lyhimmät polut	113
13.1	Bellman-Fordin algoritmi	113
13.2	Dijkstran algoritmi	116
13.3	Floyd–Warshallin algoritmi	119

14 Puiden käsittely	123
14.1 Puun läpikäynti	124
14.2 Lämpimä	125
14.3 Solmujen etäisyydet	126
14.4 Binaäripuut	127
15 Virittävät puut	129
15.1 Kruskalin algoritmi	130
15.2 Union-find-rakenne	132
15.3 Primin algoritmi	134
16 Suunnatut verkot	137
16.1 Topologinen järjestys	137
16.2 Dynaaminen ohjelmointi	139
16.3 Tehokas eteneminen	142
16.4 Syklin tunnistaminen	143
17 Vahvasti yhtenäisyys	145
17.1 Kosarajun algoritmi	146
17.2 2SAT-ongelma	148
18 Puukyselyt	151
18.1 Tehokas nouseminen	151
18.2 Solmutaulukko	152
18.3 Alin yhteinen esivanhempi	155
19 Polut ja kierrokset	159
19.1 Eulerin polku	159
19.2 Hamiltonin polku	163
19.3 De Bruijnin jono	165
19.4 Ratsun kierros	166
20 Virtauslaskenta	167
20.1 Ford-Fulkersonin algoritmi	168
20.2 Rinnakkaiset polut	172
20.3 Maksimiparitus	173
20.4 Polkupeitteet	176
III Lisäaiheita	179
21 Lukuteoria	181
21.1 Alkuluvut ja tekijät	181
21.2 Modulolaskenta	185
21.3 Yhtälönratkaisu	188
21.4 Muita tuloksia	189

22 Kombinatoriikka	193
22.1 Binomikerroin	194
22.2 Catalanin luvut	196
22.3 Inklusio-eksklusio	198
22.4 Burnsiden lemma	200
22.5 Cayleyn kaava	201
23 Matriisit	203
23.1 Laskutoimitukset	203
23.2 Lineaariset rekursioyhtälöt	206
23.3 Verkkojen käsittely	208
24 Todennäköisyys	211
24.1 Tapahtumat	212
24.2 Satunnaismuuttuja	214
24.3 Markovin ketju	216
24.4 Satunnaisalgoritmit	217
25 Peliteoria	221
25.1 Pelin tilat	221
25.2 Nim-peli	223
25.3 Sprague–Grundyn lause	224
26 Merkkijonoalgoritmit	229
26.1 Trie-rakenne	229
26.2 Merkkijonohajautus	230
26.3 Z-algoritmi	234
27 Neliöjuorialgoritmit	239
27.1 Eräkäsittely	240
27.2 Tapauskäsittely	241
27.3 Mo’n algoritmi	241
28 Lisää segmenttipuusta	243
28.1 Laiska eteneminen	244
28.2 Dynaaminen toteutus	247
28.3 Tietorakenteet	249
28.4 Kaksiulotteisuus	250
29 Geometria	253
29.1 Kompleksiluvut	254
29.2 Pisteet ja suorat	256
29.3 Monikulmion pinta-ala	259
29.4 Etäisyysmitat	260

30 Pyyhkäisyviiva	263
30.1 Janojen leikkauspisteet	264
30.2 Lähin pistepari	265
30.3 Konvekssi peite	266

Osa I

Perusasiat

Osa II

Verkkoalgoritmit

Osa III

Lisäaiheita

Luku 30

Pyyhkäisyviiva

Pyyhkäisyviiva on tason halki kulkeva viiva, jonka avulla voi ratkaista useita geometrisia tehtäviä. Ideana on esittää tehtävä joukkona tapahtumia, jotka vastaavat tason pisteitä. Kun pyyhkäisyviiva törmää pisteeseen, tapahtuma käsitellään ja tehtävän ratkaisu edistyy.

Seuraava tehtävä tarjoaa yksinkertaisen esimerkin tekniikasta:

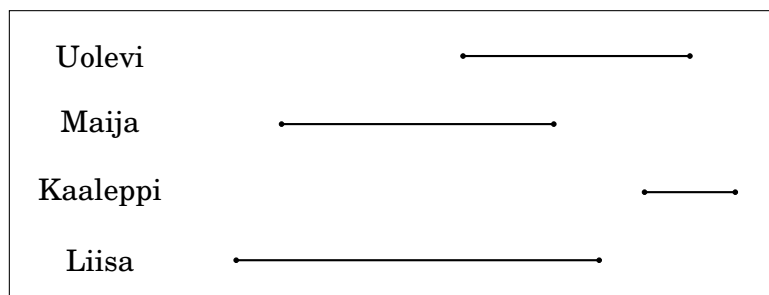
Tehtävä: Yrityksessä on töissä n henkilöä. Tiedät jokaisesta henkilöstä, milloin hän tuli töihin ja lähti töistä tietyinä päivinä. Mikä on suurin määrä henkilöitä, jotka olivat samaan aikaan töissä?

Tehtävän voi ratkaista mallintamalla tilanteen niin, että jokaista henkilöä vastaa kaksi tapahtumaa: tuloaika töihin ja lähtöaika töistä. Pyyhkäisyviiva käy läpi tapahtumat aikajärjestyksessä ja pitää kirjaa, montako henkilöä on töissä milloinkin.

Esimerkiksi tilannetta

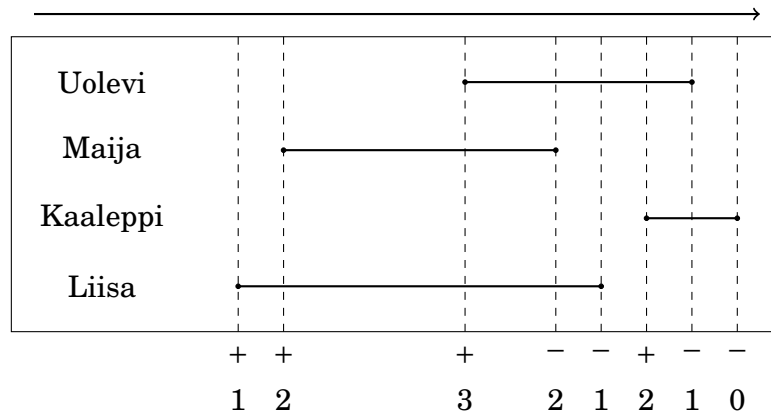
henkilö	tuloaika	lähtöaika
Uolevi	10	15
Maija	6	12
Kaaleppi	14	16
Liisa	5	13

vastaavat seuraavat tapahtumat:



Pyyhkäisyviiva käy läpi tapahtumat vasemmalta oikealle ja pitää yllä laskuria. Aina kun henkilö tulee töihin, laskurin arvo kasvaa yhdellä, ja kun henkilö lähtee töistä, laskurin arvo vähenee yhdellä. Tehtävän ratkaisu on suurin laskuri arvo pyyhkäisyviivan kulun aikana.

Pyyhkäisyviiva kulkee seuraavasti tason halki:



Kuvan alareunan merkinnät + ja – tarkoittavat, että laskurin arvo kasvaa ja vähenee yhdellä. Niiden alapuolella on laskurin uusi arvo. Laskurin suurin arvo 3 on voimassa Uolevi tulohetken ja Maijan lähtöhetken välillä.

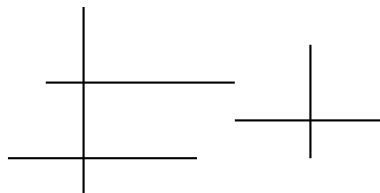
Ratkaisun aikavaativuus on $O(n \log n)$, koska tapahtumien järjestäminen vie aikaa $O(n \log n)$ ja pyyhkäisyviivan läpikäynti vie aikaa $O(n)$.

Tässä luvussa tutustumme kolmeen klassiseen tehtävään, jotka ratkeavat tehokkaasti pyyhkäisyviivan avulla. Ensimmäinen tehtävämme on laskea tassossa olevien janojen leikkauspisteiden määrä.

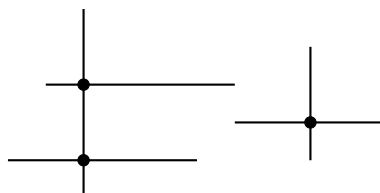
30.1 Janojen leikkauspisteet

Tehtävä: Annettuna on n janaa, joista jokainen on vaaka- tai pystysuuntainen. Monessako pisteessä kaksi janaa leikkaa toisiaan?

Esimerkiksi tilanteessa



leikkauspisteitä on kolme:

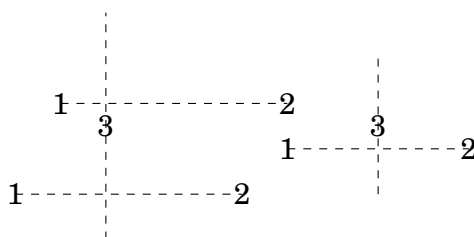


Tehtävä on helppoa ratkaista ajassa $O(n^2)$, koska riittää käydä läpi kaikki mahdolliset janaparit ja tarkistaa, moniko leikkaa toisiaan. Seuraavaksi ratkaisemme tehtävän ajassa $O(n \log n)$ pyyhkäisyviivan avulla.

Ideana on luoda janoista kolmenlaisia tapahtumia:

- (1) vaakajana alkaa
- (2) vaakajana päättyy
- (3) pystyjana

Äskeistä esimerkkiä vastaava pistejoukko on seuraava:



Algoritmi käy läpi pisteet vasemmalta oikealle ja pitää yllä tietorakennetta y-koordinaateista, joissa on tällä hetkellä aktiivinen vaakajana. Tapahtuman 1 kohdalla vaakajanan y-koordinaatti lisätään joukkoon ja tapahtuman 2 kohdalla vaakajanan y-koordinaatti poistetaan joukosta.

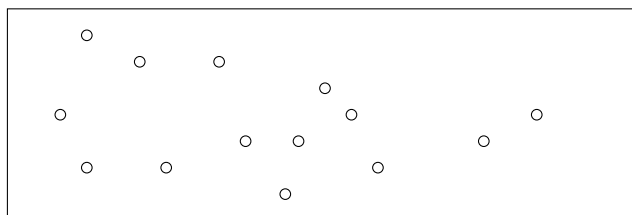
Algoritmi laskee janojen leikkauspisteet tapahtumien 3 kohdalla. Kun pystyjana kulkee y-koordinaattien $y_1 \dots y_2$ välillä, algoritmi laskee tietorakenteesta, monessako vaakajanassa on y-koordinaatti välillä $y_1 \dots y_2$ ja kasvattaa leikkauspisteiden määrää tällä arvolla.

Sopiva tietorakenne vaakajanojen y-koordinaattien tallentamiseen on binääriindeksipuu tai segmenttipuu, johon on tarvittaessa yhdistetty indeksien pakkaus. Tällöin jokaisen pisteen käsittely vie aikaa $O(\log n)$, joten algoritmin kokonaisaikavaativuus on $O(n \log n)$.

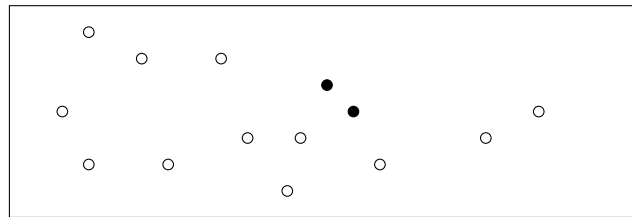
30.2 Lähin pistepari

Tehtävä: Annettuna on n pistettä kaksiulotteisessa tasossa ja tehtäväsi on etsiä kaksi pistettä, jotka ovat mahdollisimman lähellä toisiaan.

Esimerkiksi tilanteessa



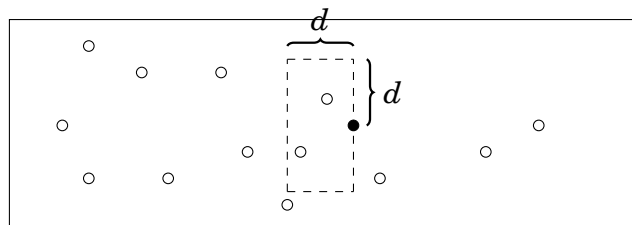
lähin pistepari on seuraava:



Tämäkin tehtävä ratkeaa $O(n \log n)$ -ajassa pyyhkäisyviivan avulla. Algoritmi käy pisteet läpi vasemmalta oikealle ja pitää yllä arvoa d , joka on pienin kahden pisteen etäisyys. Kunkin pisteen kohdalla algoritmi etsii lähimmän toisen pisteen vasemmalta. Jos etäisyys tähän pisteeseen on alle d , tämä on uusi pienin kahden pisteen etäisyys ja algoritmi päivittää d :n arvon.

Jos käsiteltävä piste on (x, y) ja jokin vasemmalla oleva piste on alle d :n etäisyydellä, sen x-koordinaatin tulee olla välillä $[x-d, x]$ ja y-koordinaatin tulee olla välillä $[y-d, y+d]$. Algoritmin riittää siis tarkistaa ainoastaan pisteet, jotka osuvat tälle välille, mikä tehostaa hakua merkittävästi.

Esimerkiksi seuraavassa kuvassa katkoviiva-alue sisältää pisteet, jotka voivat olla alle d :n etäisyydellä tummennetusta pisteestä.



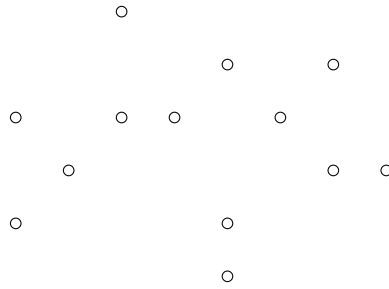
Algoritmin tehokkuus perustuu siihen, että d :n rajoittamalla alueella on aina vain $O(1)$ pistettä. Nämä pisteet pystyy käymään läpi $O(\log n)$ -aikaisesti pitämällä algoritmin aikana yllä joukkoa pisteistä, joiden x-koordinaatti on välillä $[x-d, x]$ ja jotka on järjestetty y-koordinaatin mukaan.

Algoritmin aikavaativuus on $O(n \log n)$, koska se käy läpi n pistettä ja etsii jokaiselle lähimmän edeltävän pisteen ajassa $O(\log n)$.

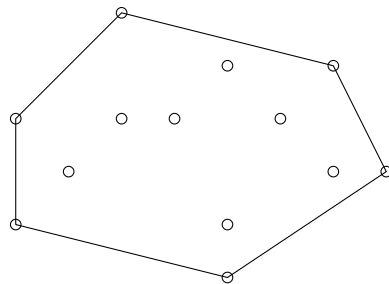
30.3 Konvekksi peite

Konvekssi peite on pienin konvekksi monikulmio, joka ympäröi kaikki pistejoukon pisteet. Konveksius tarkoittaa, että minkä tahansa kahden kärkipisteen välinen jana kulkee monikulmion sisällä. Hyvä mielikuva asiasta on, että pistejoukko ympäröidään tiukasti viritetyllä narulla.

Esimerkiksi pistejoukon



konvekksi peite on seuraava:

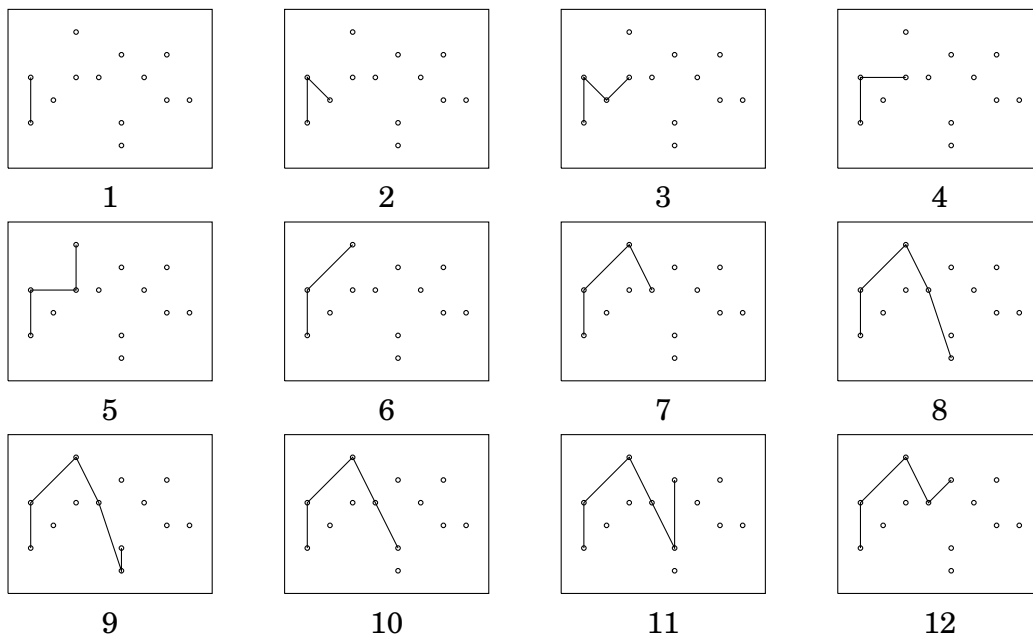


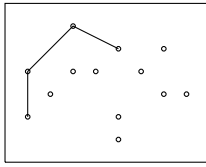
Tehokas ja helposti toteutettava menetelmä muodostaa konvekksi peite on **Andrew'n algoritmi**, jonka aikavaativuus on $O(n \log n)$.

Algoritmi muodostaa konveksin peitteen kahdessa osassa: ensin peitteen yläosan ja sitten peitteen alaosan. Kummankin osan muodostaminen tapahtuu samalla tavalla, ja keskitymme nyt yläosan muodostamiseen.

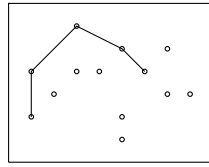
Algoritmi järjestää ensin pisteet ensisijaisesti x-koordinaatin ja toissijaisesti y-koordinaatin mukaan. Tämän jälkeen se käy pisteet läpi järjestyksessä ja liittää aina uuden pisteen osaksi peitettä. Kuitenkin aina kun kolme viimeistä pistettä peitteessä muodostavat vasemmalle kääntyvän osan, algoritmi poistaa näistä keskimmäisen pisteen.

Seuraava kuvasarja esittää Andrew'n algoritmin toimintaa:

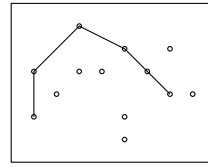




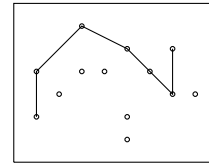
13



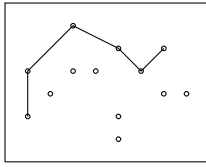
14



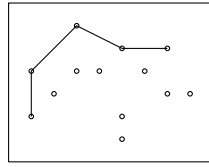
15



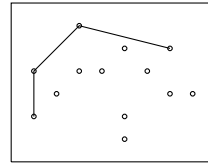
16



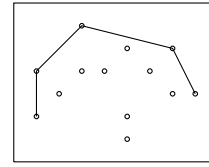
17



18



19



20

Algoritmi tarkastaa ristitulon avulla, muodostavatko kolme pistettä vasemmal-
le kääntyvän osan. Algoritmin aikavaativuus on $O(n \log n)$, koska pisteiden jär-
jestäminen vie aikaa $O(n \log n)$ ja sen jälkeen konveksin peitteen muodostami-
nen vie aikaa $O(n)$.

Hakemisto

#define, 8
bitset, 39
complex, 254
deque, 39
map, 36
multiset, 35
next_permutation, 45
priority_queue, 40
queue, 40
random_shuffle, 37
reverse, 37
set, 35
sort, 27, 37
stack, 40
string, 34
typedef, 8
unordered_map, 36
unordered_multiset, 35
unordered_set, 35
vector, 33
2SAT-ongelma, 148

aakkosto, 229
ahne algoritmi, 51
aikavaativuus, 15
alijono, 229
alin yhteinen esivanhempi, 155
alkuluku, 181
alkulukupari, 183
alkuosa, 229
alkutekijähajotelma, 181
Andrew'n algoritmi, 267
aritmeettinen summa, 13
aste, 101

Bellman-Fordin algoritmi, 113
binääri-indeksipuu, 80
binäärihaku, 29
binäärikoodi, 56
binääripuu, 128

binomijakauma, 215
binomikerroin, 194
bittiesitys, 89
bittijoukko, 39
Burnsiden lemma, 200

Catalanin luku, 196
Cayleyn kaava, 201

de Bruijnin jono, 165
determinantti, 205
Dijkstran algoritmi, 116, 141
Dilworthin lause, 178
Diofantoksen yhtälö, 188
Diracin lause, 164
dynaaminen ohjelmointi, 59
dynaaminen segmenttipuu, 247

editointietäisyys, 68
Edmonds-Karpin algoritmi, 170
ehdollinen todennäköisyys, 213
Eratostheneen seula, 184
esijärjestys, 128
etäisyysmitta, 260
Eukleideen algoritmi, 184, 188
Eukleideen kaava, 190
Euklidinen etäisyys, 260
Eulerin kierros, 160
Eulerin lause, 186
Eulerin polku, 159
Eulerin totienttifunktio, 185

Fermat'n pieni lause, 186
Fibonaccin luku, 14, 190, 206
Floyd-Warshallin algoritmi, 119
Floydin algoritmi, 144
Ford-Fulkersonin algoritmi, 168
funktionaalinen verkko, 142

geometria, 253
geometrinen jakauma, 216

geometrinen summa, 13
 Goldbachin konjektuuri, 183
 Grundy-luku, 225
 Grundyn peli, 227

 häviötila, 221
 hajautus, 231
 hajautusarvo, 231
 hakemisto, 36
 Hallin lause, 174
 Hamiltonin kierros, 164
 Hamiltonin polku, 163
 harmoninen summa, 14, 184
 harva segmenttipuu, 247
 Heronin kaava, 253
 heuristiikka, 166
 Huffmanin koodaus, 56

 indeksien pakkaus, 87
 inklusio-eksklusio, 198
 inversio, 25
 iteraattori, 37

 jälkijärjestys, 128
 järjestäminen, 23
 jakaja, 181
 jakauma, 215
 jakso, 229
 jaollisuus, 181
 jono, 40
 joukko, 11, 35

 käänteismatriisi, 206
 Königin lause, 175
 kaari, 99
 kaarilista, 105
 kaksi osoitinta, 71
 kaksijakoisuus, 102, 112
 kaksiulotteinen segmenttipuu, 250
 keko, 40
 kekojärjestäminen, 26
 kierto, 229
 kiinalainen jäännöslause, 189
 Kirchhoffin lause, 209
 kofaktori, 205
 kokonaisluku, 6
 kombinatoriikka, 193
 kompleksiluku, 254

 konvekksi peite, 266
 koodi, 56
 koodisana, 56
 Kosarajun algoritmi, 146
 Kruskalin algoritmi, 130
 kuplajärjestäminen, 24

 lähin pienempi edeltäjä, 74
 lähin pistepari, 265
 läpimitta, 125
 Lagrangen lause, 189
 laiska eteneminen, 244
 laiska segmenttipuu, 244
 Las Vegas -algoritmi, 217
 laskemisjärjestäminen, 27
 Legendren konjektuuri, 183
 leikkaus, 168
 leikkauspiste, 257, 264
 leksikografinen järjestys, 230
 leveyshaku, 109
 liukuluku, 7
 liukuvan ikkunan minimi, 75
 logaritmi, 12
 logiikka, 11
 lomitussjärjestäminen, 25
 loppuosa, 229
 lukuteoria, 181
 lyhin polku, 113

 makro, 8
 maksimiparitus, 173
 maksimivirtaus, 167
 Manhattan-etäisyys, 260
 Markovin ketju, 216
 matriisi, 203
 matriisipotenssi, 205
 matriisitulo, 204, 218
 merkkijono, 34, 229
 merkkijonohajautus, 231
 mex-funktio, 225
 minimileikkaus, 168
 Mo'n algoritmi, 241
 modulolaskenta, 6
 Monte Carlo -algoritmi, 217
 multinomikerroin, 196
 muutoshistoria, 248

 naapuri, 101

neliöjuuri, 239
 neliömatriisi, 203
 nim-peli, 223

 odotusarvo, 214
 ohjelmointikieli, 3
 Oren lause, 164
 osajono, 229
 osajoukko, 43

 pakka, 39
 permutaatio, 45
 persistentti segmenttipuu, 248
 peruuttava haku, 46
 Pickin lause, 260
 pienin yhteinen moninkerta, 184
 pikajärjestäminen, 26
 pino, 40
 pisin nouseva alijono, 64
 piste, 254
 polkupeite, 176
 polynominen hajautus, 231
 Prüfer-koodi, 201
 prefiksi, 229
 Primin algoritmi, 134
 prioriteettijono, 40
 puolivälihaku, 49
 puu, 123
 Pythagoraan kolmikko, 190
 pyyhkäisyviiva, 263

 ratsun kierros, 166
 rekursioyhtälö, 60, 206
 repunpakkaus, 66
 reuna, 230
 riippumaton joukko, 175
 riippumattomuus, 214
 ristitulo, 256

 satunnaisalgoritmi, 217
 satunnaismuuttuja, 214
 segmenttipuu, 83, 243
 seuraajaverkko, 142
 sisäjärjestys, 128
 solmu, 99
 solmupeite, 175
 SPFA-algoritmi, 116
 Sprague–Grundyn lause, 224

suffiksi, 229
 suhteellinen alkuluku, 185
 suljettu muoto, 193
 sulkulauseke, 196
 summataulukko, 78
 suurin alitaulukko, 20
 suurin yhteinen tekijä, 184
 syöte ja tuloste, 4
 sykli, 111, 137, 143
 syklin tunnistaminen, 143
 syntymäpäiväparadoksi, 233
 syvyyshaku, 107

 täydellinen luku, 182
 törmäys, 233
 tasajakauma, 215
 tasoitettu analyysi, 71
 tekijä, 181
 tietorakenne, 33
 todennäköisyys, 211
 topologinen järjestys, 137
 transpoosi, 203
 trie, 230

 union-find-rakenne, 132

 välikysely, 77
 väritys, 102, 219
 vaativuusluokka, 18
 vahvasti yhtenäisyys, 145
 vektori, 33, 203, 254
 verkko, 99
 vieruslista, 103
 vierusmatriisi, 104
 virittävä puu

- pienin ja suurin, 129
- yhteismäärä, 209

 virtaus, 167
 voittotila, 221

 Warnsdorffin sääntö, 166
 Wilsonin lause, 191

 yhtenäisyys, 100, 111
 ykkösmatriisi, 204

 Z-algoritmi, 234
 Z-tila, 234
 Zeckendorfin lause, 190