

Prefață

„Dacă logica nu îți mai folosește, trebuie să renunți la logică!”, spunea un personaj al unui roman destul de cunoscut. Replica imediată a fost: „Mi se pare logic!”. Deși algoritmica pare a fi o știință exactă, uneori este mai util să renunțăm la logică. Vom economisi astfel timp prețios.

De exemplu, această prefață ar fi trebuit să fie scrisă urmând un fir logic. Nu a fost așa și a fost terminată în zece minute (e sigur fiindcă această frază a fost adăugată la sfârșit). O prefață se scrie foarte greu... dar dacă nu are nici o logică e mai ușor. Așa că nu ar trebui să vă surprindă faptul că uneori, cele spuse par să nu aibă nici o noimă.

Foarte puțini elevi care participă la concursurile de programare sunt pregătiți să renunțe la logică. Foarte puțini dintre cei care citesc acest manual au încredere în „metodele lipsite de logică”. Foarte puțini vor avea curajul să apeleze la astfel de metode. Din acest motiv, o foarte mică parte a acestei cărți nu va fi bazată pe logică.

Și totuși... uneori nu avem altă soluție. Din acest motiv, pe lângă capitolele în care vor fi prezentate diferite noțiuni de algoritmică, cartea conține și trei capitole în care se renunță la aproape orice fel de logică. Vom încerca să renunțăm la logică într-un mod logic. Sună absurd, dar veți vedea citind capitolele respective.

Am amintit aceste capitole încă de la început pentru că algoritmi probabiliști, deși ocupă o foarte mică parte a acestei cărți, reprezintă, în opinia noastră, una dintre cele mai interesante noțiuni prezentate în cadrul ei.

Așa cum am spus, restul cărții se bazează pe logică. Vor fi prezentate noțiuni avansate de teoria grafurilor, detalii referitoare la NP-completitudine (aici vor apărea și algoritmi probabiliști), elemente de geometrie computațională, precum și câțiva algoritmi avansați care nu se încadrează în nici una dintre categoriile amintite.

Cartea începe cu o prezentare a concursurilor de programare la care vor participa o mare parte dintre cititori. Veți găsi în cadrul acestui prim capitol o mulțime de sfaturi referitoare la modul în care trebuie să vă pregătiți pentru concursuri, precum și la modul în care trebuie să abordați concursul propriu-zis. Totuși, nici aici logica nu se aplică. Nu pot exista rețete sigure de succes. Așadar, puteți considera sfaturile prezentate ca fiind doar ceea ce sunt: simple sfaturi. Nu trebuie să urmați pas cu pas instrucțiunile noastre. Acest prim capitol se dorește a fi doar un ghid; el se bazează pe experiența

mai multor elevi care au participat cu succes la o mulțime de concursuri naționale și internaționale.

Al doilea capitol al cărții se dorește a fi o scurtă prezentare a noțiunii de complexitate a unui algoritmi. În cadrul acestuia vor fi prezentate detalii referitoare la modul în care poate fi estimat ordinul de mărime al timpului de execuție al unui algoritm.

În continuare, cartea este structurată în patru părți în cadrul cărora vor fi prezentate diferite noțiuni teoretice, vor fi enunțate probleme a căror rezolvare necesită cunoașterea acestor noțiuni și vor fi descrise soluțiile acestora. La începutul fiecărei părți este prezentă o scurtă prezentare a noțiunilor care vor fi descrise.

Cam atât despre ce conține cartea... Am numit-o carte fiindcă nu suntem siguri dacă este un manual sau o culegere de probleme. În mod sigur ea va fi utilizată ca manual la Centrele de Excelență (acesta este scopul pentru care a fost scrisă). Totuși, numărul relativ mare de probleme propuse și rezolvate o fac să rivalizeze cu o culegere de probleme. De fapt, nu prea contează... Sperăm doar că ea se va dovedi utilă. Puteți să o numiți cum vreți; poate că trebuie să renunțăm la logică și de data aceasta...

Vă dorim mult succes!

Autorul

Programa școlară

1. Generalități

- *Conținutul următoarelor programe (dezvoltate pentru doritorii de cunoștințe solide în programare) nu precizează limbajul de programare. În funcție de cerințele elevilor și disponibilitățile profesorilor toate temele se pot trata realizând aplicații în limbajul **Pascal** și/sau **C++**. De asemenea, se vor face aplicații și în **FreePascal**, respectiv **gnu C**. Evident, nu sunt excluse nici mediile vizuale (Delphi, Visual C++, VisualBasic sau Java).*
- *Conținutul programei este grupat în 23 de săptămâni, urmând ca ocazional să se organizeze „miniconcursuri” din domeniul (domeniile) de cunoștințe abordate deja. Elevii vor lucra în condiții similare viitoarelor concursuri la care vor participa. Evaluarea se va face automat, folosind software-ul de evaluare care va fi folosit și în celelalte săptămâni pentru a verifica lucrările elevilor și a evalua modul în care ei avansează.*

2. Obiective generale

Elevii participanți vor fi capabili:

- să folosească teoria grafurilor pentru rezolvarea diverselor probleme de programare;
- să implementeze eficient algoritmi pe grafuri;
- să transforme enunțul unei probleme în termeni ai teoriei grafurilor;
- să stăpânească noțiunile matematice elementare care stau la baza teoriei numerelor;
- să implementeze eficient algoritmi de potrivire a șirurilor;
- să folosească elemente de geometrie analitică;
- să recunoască faptul că o anumită problemă se reduce la o altă problemă cunoscută despre care se știe că este NP-completă;
- să implementeze algoritmi cât mai rapizi pentru rezolvarea problemelor NP-complete.

3. Gruparea temelor pe săptămâni

Grafuri (I)

Obiective specifice:

Elevii trebuie să își amintească cunoștințele despre grafuri acumulate în clasa a X-a. În acest scop ei vor rezolva câteva probleme care necesită noțiuni de teoria grafurilor, dar și alte cunoștințe prezentate în anul precedent.

Grafuri (II)

Obiective specifice:

Elevii trebuie să înțeleagă diferențele dintre conexitatea și tare-conexitatea unui graf orientat. De asemenea, ei trebuie să înțeleagă modul în care este ascunsă tare-conexitatea în enunțul unei probleme.

1. verificarea tare-conexității unui graf;
2. determinarea componentelor tare-conexe.

Grafuri (III)

Obiective specifice:

Elevii trebuie să cunoască noțiunea de ciclu în graf, să recunoască problemele care necesită găsirea unor astfel de cicluri și să implementeze algoritmi eficienți pentru determinarea unui ciclu sau al unei mulțimi disjuncte de cicluri.

1. verificarea ciclicității;
2. identificarea unui ciclu;
3. identificarea unei mulțimi disjuncte de cicluri.

Grafuri (IV)

Obiective specifice:

Elevii trebuie să cunoască noțiunea de punct de articulație, să recunoască problemele care necesită găsirea unor astfel de puncte în grafuri și să implementeze algoritmi eficienți pentru determinarea unor astfel de puncte.

1. implementarea algoritmului clasic (neeficient) în care se elimină câte un nod și apoi se verifică dacă graful rămas este conex;
2. implementarea algoritmului eficient.

Grafuri (V)

Obiective specifice:

Elevii trebuie să cunoască noțiunea de punte în graf, să recunoască problemele care necesită găsirea unor astfel de muchii și să implementeze algoritmi eficienți pentru determinarea unor astfel de muchii.

1. implementarea algoritmului clasic (neeficient) în care se elimină câte o muchie și apoi se verifică dacă graful rămas este conex
2. implementarea algoritmului eficient

Grafuri (VI)

Obiective specifice:

Elevii trebuie să cunoască noțiunea de componentă biconexă, să recunoască problemele care necesită găsirea unor astfel de componente și să implementeze algoritmi eficienți pentru determinarea lor.

1. implementarea algoritmului eficient.

Grafuri (VII)

Obiective specifice:

Elevii trebuie să înțeleagă noțiunea de rețea de transport și cea de flux în graf și să recunoască problemele în care trebuie determinate fluxuri. De asemenea, ei vor trebui să aleagă algoritmul adecvat problemei date.

1. implementarea algoritmului Ford-Fulkerson;
2. implementarea variantei Edmonds-Karp.

Grafuri (VIII)

Obiective specifice:

Elevii trebuie să înțeleagă noțiunea de cuplaj în general și de cuplaj bipartit în particular. De asemenea, ei trebuie să implementeze eficient cuplajul într-un graf bipartit.

1. implementarea algoritmului de determinare a cuplajului într-un graf bipartit.

NP-completitudine

Obiective specifice:

Elevii trebuie să cunoască noțiunea de NP-completitudine și să aplice metode euristice cât mai eficiente pentru a rezolva probleme NP-complete.

- Definiția NP-completitudinii;
- Reducerea în timp polinomial;
- Probleme NP-complete cunoscute;
- Metode de aproximare;
- Îmbunătățirea performanțelor algoritmilor;
- Algoritmi probabiliști;
- Algoritmi genetici;
- Multi Expression Programming.

Geometrie analitică (I)

Obiective specifice:

Elevii trebuie să cunoască formulele care stau la baza geometriei analitice și să le folosească pentru rezolvarea problemelor.

1. ecuația liniei în plan și spațiu;
2. ecuația planului;
3. aria unui triunghi, volumul unei piramide;
4. punct în interiorul triunghiului, punct în interiorul piramide;
5. verificarea intersecției a două segmente (plan și spațiu).

Geometrie analitică (II)

Obiective specifice:

Elevii trebuie să cunoască noțiunea de înfășurătoare convexă și să fie capabili să implementeze algoritmi eficienți pentru determinarea ei.

1. algoritmul clasic, ineficient;
2. scanarea Graham;
3. potrivirea Jarvis.

Geometrie analitică (III)

Obiective specifice:

Elevii trebuie să recunoască problemele care necesită determinarea unor distanțe minime sau maxime între puncte.

1. cea mai apropiată pereche de puncte;
2. cea mai îndepărtată pereche de puncte.

Teoria numerelor

Obiective specifice:

Elevii trebuie să fie capabili să folosească algoritmi de teoria numerelor pentru rezolvarea anumitor probleme.

1. algoritmul extins al lui Euclid;
2. aritmetică modulară;
3. calculul valorii $a^b \bmod n$.

Potrivirea șirurilor

Obiective specifice:

Elevii trebuie să fie capabili să utilizeze algoritmi eficienți de determinare a apariției unui subșir într-un șir.

1. Algoritmul clasic;
2. Algoritmul Rabin-Karp;
3. Algoritmul Knuth-Morris-Pratt

Arbori indexați binar

Obiective specifice:

Elevii trebuie să înțeleagă algoritmi de modificare și interogare pentru arbori indexați binar indiferent de numărul dimensiunilor în care este definită problema.

1. implementarea operațiilor de modificare și interogare în arbori indexați binar.

Cuprins

Prefață	5
Programa școlară	7
Cuprins	11
Cap. 1 - Concursuri	17
1.1. Concursurile de programare	17
1.2. Pregătirea pentru concursuri	20
1.3. Concluzii	28
Cap. 2 - Analiza complexității	29
2.1. Timpul de execuție	29
2.2. Cazul cel mai defavorabil și cazul mediu	31
2.3. Ordinul de complexitate	32
Partea I - Teoria grafurilor	34
Cap. 3 - Noțiuni elementare	35
3.1. Definiții	35
3.2. Reprezentarea grafurilor	38
3.3. Parcurgerea grafurilor	41
3.4. Arbori parțiali minimi	43
3.5. Drumuri minime	45
3.6. Rezumat	49
3.7. Implementări sugerate	50
3.8. Probleme propuse	50
3.9. Soluțiile problemelor	56
Cap. 4 - Tare-conexitate	61
4.1. Considerații teoretice	61
4.2. Un algoritm ineficient	63
4.3. Algoritmul plus-minus	67
4.4. Algoritmul optim	70
4.5. Rezumat	74
4.6. Implementări sugerate	74
4.7. Probleme propuse	74
4.8. Soluțiile problemelor	78
Cap. 5 - Cicluri	82
5.1. Etajarea grafurilor	82
5.2. O clasificare a muchiilor	84

5.3.	Determinarea unui ciclu	85
5.4.	Cicluri disjuncte	90
5.5.	Rezumat	92
5.6.	Implementări sugerate	93
5.7.	Probleme propuse	93
5.8.	Soluțiile problemelor	98
Cap. 6 - Puncte de articulație		102
6.1.	Considerații teoretice	102
6.2.	Un algoritm simplu	103
6.3.	Algoritmul eficient	104
6.4.	Rezumat	105
6.5.	Implementări sugerate	106
6.6.	Probleme propuse	106
6.7.	Soluțiile problemelor	110
Cap. 7 - Punți în grafuri		113
7.1.	Considerații teoretice	113
7.2.	Un algoritm simplu	113
7.3.	Algoritmul eficient	114
7.4.	Rezumat	116
7.5.	Implementări sugerate	116
7.6.	Probleme propuse	116
7.7.	Soluțiile problemelor	120
Cap. 8 - Biconexitate		123
8.1.	Considerații teoretice	123
8.2.	Determinarea componentelor biconexe	124
8.3.	Rezumat	126
8.4.	Implementări sugerate	127
8.5.	Probleme propuse	127
8.6.	Soluțiile problemelor	133
Cap. 9 - Fluxuri		136
9.1.	Rețele de transport	136
9.2.	Fluxuri maxime	137
9.3.	Algoritmul Ford-Fulkerson	138
9.4.	Algoritmul Edmonds-Karp	141
9.5.	Tăietura minimă	142
9.6.	Rețele de transport particulare	144
9.7.	Rezumat	144
9.8.	Implementări sugerate	145
9.9.	Probleme propuse	145
9.10.	Soluțiile problemelor	149

Cap. 10 - Cuplaje maxime	153
10.1. Cuplaje	153
10.2. Grafuri bipartite	153
10.3. Cuplaje maxime în grafuri bipartite	154
10.4. Rezumat	156
10.5. Implementări sugerate	156
10.6. Probleme propuse	156
10.7. Soluțiile problemelor	160
Partea II - Probleme NP	163
Cap. 11 - NP-completitudine	164
11.1. Clase de probleme	164
11.2. Reductibilitate	166
11.3. Probleme NP-complete	167
11.4. Concluzii	172
11.5. Rezumat	172
11.6. Implementări sugerate	172
11.7. Probleme propuse	172
11.8. Soluțiile problemelor	175
Cap. 12 - Măsurarea timpului	179
12.1. Preliminarii	179
12.2. Preluarea timpului	180
12.3. Setarea timpului	182
12.4. Întreruperea 08h	184
12.5. Locația \$0000:\$046C	186
12.6. Rezumat	187
12.7. Implementări sugerate	187
Cap. 13 - Probleme NP-complete	188
13.1. Preliminarii	188
13.2. Scurtarea timpului de execuție	189
13.3. Ordinea explorării soluțiilor	190
13.4. Particularitățile problemelor	192
13.5. Concluzii	194
13.6. Rezumat	195
13.7. Implementări sugerate	195
13.8. Probleme propuse	195
13.9. Soluțiile problemelor	198
Cap. 14 - Algoritmi probabiliști	200
14.1. Metode probabilistice	200
14.2. Utilizarea nedeterminismului	201
14.3. Noțiuni "avansate"	202

14.4.	Rezumat.....	203
14.5.	Implementări sugerate	203
Cap. 15 - Algoritmi genetici		204
15.1.	Preliminarii.....	204
15.2.	Noțiuni elementare	204
15.3.	Selecția	206
15.4.	Mutația	208
15.5.	Încrucișarea	208
15.6.	Evoluția	210
15.7.	Concluzii	212
15.8.	Rezumat.....	213
15.9.	Implementări sugerate	213
Cap. 16 - Multi Expression Programming		214
16.1.	Structura cromozomilor.....	214
16.2.	Calitatea cromozomilor	219
16.3.	Efectul mutațiilor	222
16.4.	Efectul încrucișărilor	223
16.5.	Concluzii	224
16.6.	Rezumat.....	225
16.7.	Implementări sugerate	225
Partea a III-a - Geometrie computațională		226
Cap. 17 - Linii și segmente		227
17.1.	Ecuția dreptei și a planului	227
17.2.	Pozițiile punctelor față de o dreaptă	230
17.3.	Pozițiile punctelor față de un plan.....	230
17.4.	Coliniaritate și coplanaritate	230
17.5.	Intersecții.....	232
17.6.	Arii și volume	235
17.7.	Convexitate.....	237
17.8.	Rezumat.....	238
17.9.	Implementări sugerate	238
17.10.	Probleme propuse.....	239
17.11.	Soluțiile problemelor	242
Cap. 18 - Înfășurătoarea convexă		246
18.1.	Un algoritm ineficient	246
18.2.	Scanarea Graham	247
18.3.	Potrivirea Jarvis	249
18.4.	Rezumat.....	250
18.5.	Implementări sugerate	250
18.6.	Probleme propuse.....	251
18.7.	Soluțiile problemelor	254

Cap. 19 – Distanțe	258
19.1. Preliminarii	258
19.2. Distanța minimă între două puncte	259
19.3. Distanța maximă între două puncte	263
19.4. Rezumat	265
19.5. Implementări sugerate	265
19.6. Probleme propuse	266
19.7. Soluțiile problemelor	269
Partea a IV-a - Algoritmi avansați	272
Cap. 20 - Teoria numerelor	273
20.1. Algoritmul extins al lui Euclid	273
20.2. Aritmetică modulară	275
20.3. Ridicarea la putere	276
20.4. Rezumat	279
20.5. Implementări sugerate	279
20.6. Probleme propuse	279
20.7. Soluțiile problemelor	283
Cap. 21 - Potrivirea șirurilor	284
21.1. Algoritmul ineficient	284
21.2. Algoritmul Rabin-Karp	286
21.3. Algoritmul Knuth-Morris-Pratt	289
21.4. Rezumat	291
21.5. Implementări sugerate	292
21.6. Probleme propuse	292
21.7. Soluțiile problemelor	296
Cap. 22 - Arbori indexați binar	298
22.1. Preliminarii	298
22.2. Cazul unidimensional	300
22.3. Cazul bidimensional	307
22.4. Cazul tridimensional	315
22.5. Cazul n -dimensional	316
22.6. Rezumat	317
22.7. Implementări sugerate	317
22.8. Probleme propuse	318
22.9. Soluțiile problemelor	323