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 trebui 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 algoritmii 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 algoritmii 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 şi internaționale.

Al doilea capitol al cărții se doreşte 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 cunoaşterea acestor noțiuni şi 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). Totuşi, numărul relativ mare de probleme propuse şi 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 şi de data aceasta…

Vă dorim mult succes!

Autorul

Programa şcolară

1. Generalităţi

- *Conţinutul următoarei programe (dezvoltate pentru doritorii de cunoştinţe solide în programare) nu precizează limbajul de programare. În funcţie de cerinţele elevilor şi disponibilităţile profesorilor toate temele se pot trata realizând aplicaţii în limbajul **Pascal** şi/sau **C++**. De asemenea, se vor face aplicaţii şi în **FreePascal**, respectiv **gnu C**. Evident, nu sunt excluse nici mediile vizuale (Delphi, Visual C++, VisualBasic sau Java).*
- *Conţinutul programei este grupat în 23 de săptămâni, urmând ca ocazional să se organizeze „miniconcursuri" din domeniul (domeniile) de cunoştinţe abordate deja. Elevii vor lucra în condiţii similare viitoarelor concursuri la care vor participa. Evaluarea se va face automat, folosind softwareul de evaluare care va fi folosit şi în celelalte săptămâni pentru a verifica lucrările elevilor şi a evalua modul în care ei avansează.*

2. Obiective generale

Elevii participanţi vor fi capabili:

- să folosească teoria grafurilor pentru rezolvarea diverselor probleme de programare;
- să implementeze eficient algoritmi pe grafuri;
- să transforme enunţul unei probleme în termeni ai teoriei grafurilor;
- să stăpânească noţiunile matematice elementare care stau la baza teoriei numerelor;
- să implementeze eficient algoritmi de potrivire a şirurilor;
- să folosească elemente de geometrie analitică;
- să recunoască faptul că o anumită problemă se reduce la o altă problemă cunoscută despre care se ştie 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 îşi amintească cunoştinţele despre grafuri acumulate în clasa a X-a. În acest scop ei vor rezolva câteva probleme care necesită noţiuni de teoria grafurilor, dar şi alte cunoştinţe prezentate în anul precedent.

Grafuri (II)

Obiective specifice:

Elevii trebuie să înțeleagă diferențele dintre conexitatea şi 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 cicluri şi 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 şi să implementeze algoritmi eficienți pentru determinarea unor astfel de puncte.

1. implementarea algoritmului clasic (neeficient) în care se elimină câte un nod şi 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 şi să implementeze algoritmi eficienți pentru determinarea unor astfel de muchii.

1. implementarea algoritmului clasic (neeficient) în care se elimină câte o muchie şi 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 şi să implementeze algoritmi eficienți pentru determinarea lor.

1. implementarea algoritmului eficient.

Grafuri (VII)

Obiective specifice:

Elevii trebuie să înțeleagă noțiunea de rețea de transport şi cea de flux în graf şi 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 şi 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 şi 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 probabilişti;
- Algoritmi genetici;
- Multi Expression Programming.

Geometrie analitică (I)

Obiective specifice:

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

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

Geometrie analitică (II)

Obiective specifice:

Elevii trebuie să cunoască noțiunea de înfăşurătoare convexă şi 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 mod n.

Potrivirea şirurilor

Obiective specifice:

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

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

Arbori indexați binar

Obiective specifice:

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

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

Cuprins