$\alpha + \omega$, 2002, Nr. 1, 83–88

Lietuvos moksleivių 13-osios informatikos olimpiados uždaviniai

Valentina Dagienė
dagiene@ktl.mii.lt

Straipsnyje pateikiamos 2002 *metų moksleivių informatikos olimpiados baigiamojo rato uždavinių sąlygos.*

Tryliktosios moksleivių informatikos olimpiados baigiamoji dalis vyko 2002 kovo mėn. Palangoje. Paskutiniame etape varžėsi 48 moksleiviai, paskirstyti į dvi grupes: 33 vyresniųjų (X–XII klasių) ir 15 jaunesniųjų (VIII–IX klasių).

Kaip jau daugelį metų įprasta, gausiausiai olimpiečių parengė Panevėžio J. Balčikonio gimnazija — net aštuonis. Iš Vilniaus tiksliųjų, gamtos ir technikos mokslų licėjaus bei Vilniaus Žemynos gimnazijos atvyko po 4 moksleivius.

Suprantama, šios stiprios šalies gimnazijos sutraukia talentus iš daugelio aplinkinių mokyklų ar net apylinkių. Džiugu, kad tose mokyklose yra ir mokytojų entuziastų, kurie skiria savo jėgas ir laisvalaikį darbui su talentingais mokiniais. Toks nenuilstantis mokytojas, gerai pažįstamas daugeliui informatikų (ir matematikų taip pat) yra panevėžietis Benas Budvytis. Už kruopštų darbą rengiant moksleivius olimpiadoms jis buvo apdovanotas Švietimo ir mokslo ministro padėkos raštu.

Pagrindinis olimpiados rėmėjas buvo UAB „Baltic Amadeus“. Kuprines, marškinėlius, tušinukus, lankstinukus bei kitokią medžiagą gavo ne tik laimėtojai, bet ir visi dalyviai, net mokytojai. Knygomis kaip visuomet uoliai parėmė leidykla TEV.

Iš vyresniųjų grupės laimėtojų sudarytos dvi komandos po šešis moksleivius varžytis Baltijos šalių informatikos olimpiadoje. Ji šiemet vyko Vilniuje balandžio 23–28 dienomis.

Olimpiados turinys beveik nepasikeitė nuo pernai vykusios (nebent tik tai, kad vietoje *Turbo Paskalio* buvo pasirinktas atviro kodo laisvai platinamas kompiliatorius *Free Pascalis*, taip pat leidžiama naudotis ir *Virtual Pascaliu*). Dvi dienas jaunesniųjų ir vyresniųjų grupių moksleiviai sprendė skirtingo sunkumo uždavinius. Visų uždavinių sprendimai buvo programos, kurias vertinimo komisija tikrino su iš anksto parengtais duomenų rinkiniais — testais.

UŽDAVINIAI

Tradiciniai klausimai olimpiadoje bei parvykus iš jos būna apie uždavinius. Kokie jie? Ar sunkūs? Kuo įdomūs? Kuo skiriasi nuo pernykščių?

Informatikos olimpiadai uždaviniai kuriami žvalgantis į tarptautines olimpiadas — juk norima ten laimėti. Kol kas vyrauja klasikiniai algoritmai, jų įvairios modifikacijos, taikymai, žaidimų strategijos, darbas su įvairiais moduliais.

Uždavinius nėra lengva sugalvoti, nepriekaištingai suformuluoti, parašyti jų programas, parengti keliolika testų. Visuomet trūksta naujų idėjų. Per mažai jaunų žmonių įsitraukia į šią veiklą. Suprantama, juk ir kryžiažodžius spręsti norinčių yra daugiau, negu juos sudaryti, dailinti...

Džiugu, kad rengiant uždavinius talkina studentai, buvę olimpiadininkai. Šiemet uždavinius padėjo rengti Vilniaus universiteto studentai: Justas Kranauskas, Remigijus Staniulis, Antanas Kompanas, visi ne kartą dalyvavę tarptautinėse informatikos olimpiadose.

Pateikiame vyresniųjų grupės uždavinių sąlygas.

Lapeliai su skaičiais

Atliekamas toks eksperimentas. Imama N sunumeruotų lapelių ir ant kiekvieno lapelio užrašomas bet kuris natūralusis skaičius nuo 1 iki 100. Užrašomi skaičiai gali kartotis. Lapeliai sumetami į urną.

Po to iš urnos atsitiktinai traukiama po tris lapelius ir kur nors atskirame popieriaus lape pasižymima: 1) visų trijų lapelių numeriai; 2) lapeliuose užrašytų natūraliųjų skaičių suma. Pirmas iš tų trijų lapelių išmetamas, o antrasis ir trečiasis įmetami atgal į urną.

Šitaip kartojama tol, kol urnoje lieka tik du lapeliai. Tada apskaičiuojama ant dviejų likusių lapelių užrašytų skaičių sandauga ir suma. Lapelių numeriai neužrašomi. Tuo eksperimentas baigiamas.

Užduotis. *Parašykite programą, kuri pagal eksperimento užrašus nustatytų, koks skaičius buvo užrašytas ant kiekvieno lapelio. Jeigu įmanomi keli sprendiniai, raskite tą, kuriame skaičius, užrašytas ant pirmojo lapelio, yra mažiausias.*

Pradiniai duomenys įrašyti byloje `LAPELIAI.DAT`. Pirmoje bylos eilutėje pateiktas lapelių skaičius N, $3 \leqslant N \leqslant 10000$. Tolesnėse $N-2$ eilučių įrašyta po keturis skaičius. Pirmasis skaičius yra pirmojo traukimo pirmojo lapelio (vėliau išmesto) numeris; antrasis ir trečiasis skaičiai — antrojo ir trečiojo lapelių (įmetamų atgal į urną) numeriai, o ketvirtasis skaičius — lapeliuose užrašytų skaičių suma. Kitoje eilutėje įrašyti duomenys apie antrojo traukimo lapelius, dar kitoje — apie trečiojo ir t. t. Paskutinėse dviejose N-oje ir $(N+1)$-oje bylos eilutėse įrašytos skaičių, užrašytų ant dviejų paskutinių lapelių, sandauga ir suma.

Rezultatus reikia įrašyti į bylą `LAPELIAI.REZ`. Byloje turi būti N eilučių po vieną skaičių kiekvienoje. Į pirmąją eilutę rašomas skaičius, užrašytas ant pirmojo lapelio, į antrą — ant antrojo ir t. t.

Pavyzdys

LAPELIAI.DAT	LAPELIAI.REZ	Paaiškinimas
5	10	
2 4 5 130	30	Ištraukus lapelius pirmą kartą, gaunama suma 130 $(30+40+60=130)$
4 3 1 80	30	Ištraukus antrą kartą, gaunama suma 80 $(30+40+10=80)$
3 5 1 100	40	Ištraukus trečią kartą, gaunama suma 100 $(30+60+10=100)$
600 70	60	Dviejų paskutinių lapelių skaičių sandauga lygi 600 $(60\times 10=600)$, o suma — 70 $(60+10=70)$

Inžinerinis parengimas

Antroji diena poligone. Kariūnai mokomi rankiniu būdu minuoti vietovę prieštankinėmis minomis. Seržantas R. kariūnams davė žemėlapį bei keletą minų, kurias iki minuojamų objektų reikės neštis kuprinėse. Prieštankinė priešvikšrinė-priešdugninė mina TM-72 sveria kelis kilogramus.

Kariūnams reikia užminuoti vietovę. Jos žemėlapis padalytas į vienetinius kvadratėlius. Susitarkime vietovę, atitinkančią kvadratėlį žemėlapyje, vadinti *lauku*. Lauke gali būti objektas arba laukas gali būti tuščias. Objektai yra dviejų tipų:

minuojamas, t. y. objektas, kurį reikia apsaugoti nuo galimų antpuolių (reikia užminuoti);

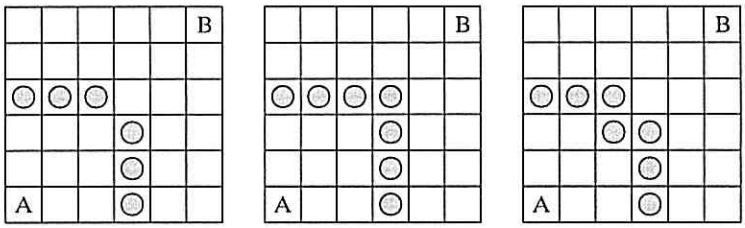
neminuojamas, t. y. objektas, kurio užminuoti negalima.

Todėl nepatirdami nuostolių priešai gali judėti laukais, kuriuose yra neminuojami objektai.

Objektai išsidėstę taip, kad gretimi aštuoni kiekvieną objektą (minuojamą ir neminuojamą) supantys laukai yra tušti. Užminuoti galima tik tuščius laukus.

Tankai ir kita šarvuota priešo technika iš vieno lauko gali pervažiuoti į kitą, jei žemėlapyje laukus atitinkantys kvadratėliai liečiasi briaunomis arba kampais. Priešai be nuostolių gali keliauti tik neužminuotais laukais.

Tarkime, priešas atkeliavo į lauką A ir nori užimti objektą lauke B. Panagrinėkime tris atvejus:



a) b) c)

a) atveju priešo tankai ir kita šarvuota technika be nuostolių gali nuvykti iš A į B; b) ir c) atvejais be nuostolių to padaryti neįmanoma.

Tariama, kad priešo stovykla yra už žemėlapio ribų, t. y. puldamas priešas pirmiausia patenka į kurį nors kraštinį žemėlapio kvadratėlį atitinkantį lauką.

Kariūnai išanalizavo žemėlapį ir įvertino situaciją: visi minuojami objektai išdėstyti arti vienas kito, todėl užtenką juos apjuosti viena ištisine minų linija.

Užduotis. *Reikia parašyti programą, kuri apskaičiuotų, kiek mažiausiai minų kuopa turi paimti į žygį, kad nereikėtų grįžti į aprūpinimo punktą papildyti atsargų.*

Pradiniai duomenys pateikti tekstinėje byloje `MINOS.DAT`. Pirmoje bylos eilutėje įrašyti du sveikieji skaičiai: žemėlapį sudarančių eilučių N ir stulpelių M, $3 \leqslant N, M \leqslant 100$. Likusiose N eilučių įrašyta po M žemėlapį nusakančių simbolių. Galimi tokie simboliai: **X** — minuojamas objektas; **1** — neminuojamas objektas; 0 — tuščias laukas, kurį galima minuoti.

Rezultatus reikia įrašyti į bylą `MINOS.REZ`. Pirmoje bylos eilutėje turi būti įrašomas vienas sveikasis skaičius — mažiausias minavimui reikalingų minų kiekis.

Pavyzdys

MINOS.DAT	MINOS.REZ	Paaiškinimas
10 11	30	
00000000000		000***00000
0100X000000		010*X*00000
00000000000		0***0*00000
00X01000000		0*X01****00
0000000X000		0*00000X*00
00X00100000		0*X001***00
00000000000		0***0**0000
0000X001000		000*X*01000
00000000000		000***00000
00000000000		00000000000

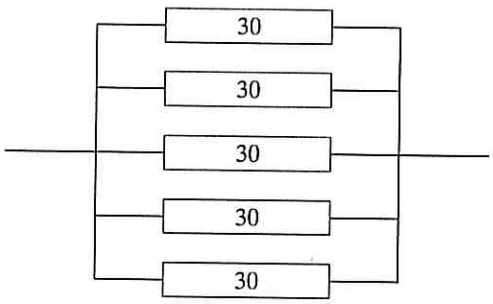
Schemos

Jaunasis kompiuterininkas gamina kompiuterio priedą. Jam prireikia rezistoriaus, kurio varža būtų lygi R. Deja, tokio rezistoriaus jis neturi, tačiau turi daug vienodų rezistorių, kurių kiekvieno varža yra r.

$\alpha + \omega$

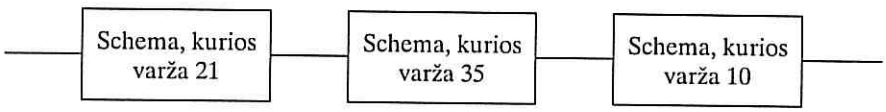
Prisiminęs fizikos žinias, vaikinas kimba į darbą: išrenka kelis rezistorius ir sujungia juos lygiagrečiai. Tokiu būdu gauna schemą, kurios varža tiek kartų mažesnė, kiek rezistorių sujungta lygiagrečiai. Į schemą įtraukus tik vieną rezistorių, jos varža lygi r.

Pavyzdžiui, lygiagrečiai sujungus penkis rezistorius, kurių kiekvieno varža lygi 30, gaunama schema, kurios varža lygi 6 ($\frac{30}{5} = 6$) (1 pav.).



1 pav.

Taip kompiuterininkas pasigamina reikiamą skaičių lygiagrečiai sujungtų rezistorių grupelių. Šias grupeles sujungęs nuosekliai, jis gauna reikiamos vertės rezistorių, t. y. schemą, kurios varža lygi R. Jungiant rezistorių grupeles nuosekliai, visa grupelių varža yra lygi grupelių varžų sumai. Pavyzdžiui, nuosekliai sujungus tris schemas, kurių varžos yra 21, 35 ir 10, gaunama schema, kurios varža lygi 66 ($21 + 35 + 10 = 66$) (2 pav.).



2 pav.

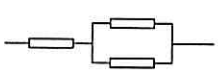
Užduotis. *Reikia parašyti programą, kuri apskaičiuotų, kiek mažiausiai r varžos rezistorių reikia paimti, norint gauti R varžos schemą. Rezistoriai jungiami tokiu būdu, kaip juos jungė jaunasis kompiuterininkas.*

Pastaba. Schemų varžos turėtų būti skaičiuojamos paprastosiomis trupmenomis, paklaidos yra neleistinos.

Pradiniai duomenys — du tarpu atskirti sveikieji skaičiai R ir r — įrašyti vienintelėje bylos `SCHEM.DAT` eilutėje. Galioja ribojimai $1 \leqslant r$, $R \leqslant 300$.

Rezultatą reikia įrašyti į bylą `SCHEM.REZ`. Jeigu iš duotų r varžų rezistorių galima sudaryti R varžos schemą ir mažiausias reikiamų rezistorių skaičius neviršija 1000, tuomet į bylą įrašomas reikiamas vertės r rezistorių skaičius. Jei R varžos schemą galima sudaryti, tačiau reikalingų rezistorių skaičius viršija 1000, į rezultatų bylą įrašoma: `NEGAMINTI`. Jei R varžos schemos sudaryti negalima, į rezultatų bylą įrašomas žodis `NEGALIMA`.

Pavyzdys

SCHEM.DAT	SCHEM.REZ	Paaiškinimas
2 3	3	Dvi grupelės iš vieno ir dviejų rezistorių, kurių kiekvieno varža lygi 2:  2 + 1 = 3

Lygiagretūs procesoriai

Turime P lygiagrečiai veikiančių procesorių. Jiems reikia atlikti U užduočių. Kai kurios užduotys yra tarpusavyje priklausomos, t. y. konkrečią užduotį galima vykdyti tik tada, kai atlikta viena ar kelios kitos užduotys. Viena iš užduočių yra *paskutinė*, t. y. ją galima atlikti tik tada, kai atliktos visos kitos.

Jei užduotį a būtina atlikti prieš pradedant vykdyti užduotį b, tai sakysime, kad užduotys a ir b sudaro *priklausomą porą* (a, b).

Kiekvienas procesorius bet kurią užduotį atlieka per laiko vienetą. Užduočių rinkinys yra korektiškas, t. y. jame nėra tokios priklausomų užduočių porų grandinės, kuri sudarytų ciklą. Bet kuriai užduočiai a (išskyrus paskutinę) egzistuoja viena užduotis b, kad a ir b sudarytų priklausomą porą (a, b).

Užduotis. *Procesorių bei užduočių skaičius ir priklausomų užduočių poros yra žinomos. Reikia parašyti programą, kuri apskaičiuotų, per kiek mažiausiai laiko vienetų bus atliktos visos užduotys.*

Pradiniai duomenys pateikiami byloje `PROC.DAT`. Pirmoje eilutėje įrašyti trys sveikieji skaičiai: P — procesorių ($1 \leqslant P \leqslant 50$), U — užduočių ($1 \leqslant U \leqslant 2000$), N — priklausomų porų. Užduotys sunumeruotos nuo 1 iki U.

Likusiose N eilučių įrašyta po du sveikuosius skaičius a ir b, $1 \leqslant a, b \leqslant U$. Šie skaičiai reiškia, kad užduotys a ir b sudaro priklausomą porą, t. y. užduotį b galima pradėti vykdyti tik tada, jei užduotis a yra atlikta.

Rezultatas — vienas sveikasis skaičius (laiko, per kurį procesoriai gali įvykdyti visas užduotis, mažiausias vienetų skaičius) įrašomas į bylą `PROC.REZ`.

Pavyzdys

PROC.DAT	PROC.REZ	Paaiškinimas
2 4 3 6 3 1 4 1	3	Per pirmą laiko momentą atliekama antroji užduotis, per antrąjį — trečioji ir ketvirtoji, per trečiąjį — pirmoji užduotis

Dėžės

Turime N įvairaus dydžio tuščių dėžių. Dėžę nusako jos plotis, ilgis, aukštis ir sienelės storis. Visų dėžių sienelių storis vienodas ir lygus 1 cm. Dėžės atsidaro viršuje.

Visas N dėžių reikia pervežti. Mažesnes dėžes sudėjus į didesnes, mašinų joms pervežti reiktų mažiau. Dedant dėžes vieną į kitą, jų negalima nei vartyti, nei pasukti. Vieną dėžę įdėjus į kitą, didesnioji dėžė būtinai turi užsidaryti.

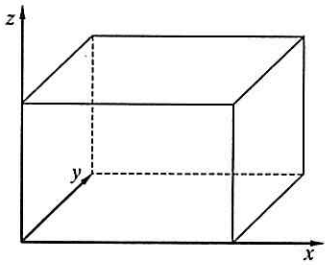
Į vidų dėžes galima dėti vieną šalia kitos ir vieną ant kitos. Sudėjus dėžes vieną į kitą, dėžė, neįdėta į jokią kitą, vadinama konteineriu. Konteineryje gali būti kitų dėžių arba jis gali būti tuščias.

Užduotis. *Parašykite programą, kuri dėžes vieną į kitą sudėtų taip, kad galutinis konteinerių skaičius būtų kuo mažesnis.*

Pradiniai duomenys įrašyti byloje `DEZES.DAT`. Pirmoje eilutėje nurodytas dėžių skaičius N, ($2 \leqslant N < 80$). Dėžės sunumeruotos nuo 1 iki N. Kitose N eilučių įrašyti dėžių matmenys, vienai dėžei skiriant po vieną eilutę: dėžės numeris nr, dėžės plotis w, ilgis l, aukštis h. Galioja ribojimai: $1 \leqslant w, l, h \leqslant 100$ (visi trys dydžiai išreikšti centimetrais).

Rezultatus įrašykite į tekstinę bylą DEZES.REZ. Pirmoje eilutėje nurodomas gautų konteinerių skaičius K. Kitose aprašomos konteineriuose esančios dėžės. Kiekvieno konteinerio aprašas pradedamas atskira eilute, kurios pirmose dviejose pozicijose įrašytos žvaigždutės **. Tolesnių aprašančių konteinerio sudėtį eilučių pirmas skaičius atitraukiamas per dvi pozicijas (t. y. spausdinant nurodomas formatas 5).

Aprašant konteinerio sudėtį, pirmiausia atskira eilute nurodomas dėžės numeris ir jos viduje esančių dėžių skaičius (jeigu konteineris tuščias, tai rašomas 0). Toliau atskiromis eilutėmis išvardijamos viduje esančios dėžės, nurodant jų numerius ir koordinates: tris sveikuosius skaičius x, y, z, nusakančius *mažesniosios dėžės padėtį didesniojoje.*



Koordinačių pradžios taškas yra didesniosios dėžės priekinis kairysis apatinis kampas. Koordinatė x atitinka plotį, y — ilgį ir z — aukštį.

Jeigu konteineryje esančių dėžių viduje yra kitų dėžių, tai jų sudėtys nurodomos taip pat, kaip ir paties konteinerio, t. y. dėžės numeris, viduje esančių dėžių skaičius ir jų padėtys, kiekvienai dėžei skiriant po atskirą eilutę.

Pavyzdys

DEZES.DAT	DEZES.REZ	Paaiškinimai
13	2	Dėžės sudėtos į du konteinerius
12 4 3 5	**	
6 13 5 3	13 4	Pirmame konteineryje (13-oje dėžėje) yra 11 dėžių Į 13-ąją dėžę įdėtos keturios dėžės: 6, 2, 1 ir 7
5 4 5 4	6 1 1 1	
13 23 24 24	2 1 1 4	
4 6 4 3	1 1 10 1	
1 12 10 10	7 1 1 13	
2 9 9 9	2 1	Į 2-ąją įdėta viena dėžė — 12
10 3 3 3	12 1 1 1	
3 3 5 3	1 5	Į 1-ąją įdėtos penkios dėžės: 10, 5, 11, 4 ir 3
7 15 15 5	10 1 1 1	
9 25 25 12	5 1 1 4	
8 15 15 5	11 1 4 1	
11 3 3 3	4 4 1 1	
	3 5 1 4	
	**	
	9 1	Antrame konteineryje (9-oje dėžėje) yra dvi dėžės 9-oje dėžėje yra 8-oji dėžė