DESCRIEREA SOLUȚIILOR
OLIMPIADA NAȚIONALA DE INFORMATICĂ
CLASA A IX-A
ETAPA NAȚIONALĂ

Problema 1: Colibri

Propusă de: stud. Alexandru Petrescu, Universitatea din Oxford

Pregătită de: stud. Andrei Arhire, Universitatea A.I. Cuza, Iași

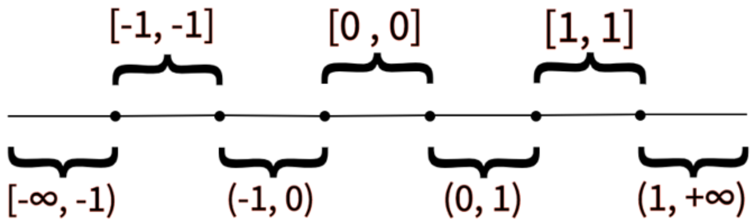
Editorial redactat de: stud. Andrei Arhire, Universitatea A.I. Cuza, Iași

Soluția 1. Vom încerca să găsim subșirul de termeni cu cel mai mare produs presupunând că produsul aparține intervalului $[1, +\infty)$. În cazul în care nu există un astfel de subșir vom căuta în mod similar pentru intervalele $(0, 1)$, $[0, 0]$ și $(-\infty, 0)$.

- O variantă de subșir care este soluție și are produsul în $[1, +\infty)$ conține toate fracțiile pozitive supraunitare și pozitive echiunitare. De asemenea conține și cele mai mici $2 \cdot K$ fracții negative cu mențiunea că cele mai mari două dintre acestea formează un produs mai mare sau egal cu 1.
- În continuare știm că nu exista soluție în $[1, +\infty)$ și încercăm să găsim una în $(0, 1)$. Dacă există atunci conține cel mult 2 fracții. Fie este formată din cea mai mare fracție pozitiva, fie este formată din cele mai mici 2 fracții negative.
- Cel mai mare produs al unui subșir este 0 atunci când nu exista nicio fracție pozitivă, există cel mult o fracție negativă și cel puțin o fracție cu rezultatul 0.
- Produsul aparține intervalului $(-\infty, -1)$ când există o singură fracție cu valoare negativă.

Plecând de la aceste observații problema se poate rezolva în $\mathcal{O}(N \log N)$. Mai întâi se verifică dacă produsul este cel mult 0. Se sortează fracțiile după rezultat și se mențin 2 pointeri, fiecare la un capăt. Cel din capătul stâng se va deplasa cu câte 2 poziții spre dreapta cât timp produsul fracției indicate cu cel al fracției următoare este cel puțin 1 și ambele fracții sunt negative, iar cel din capătul drept se va deplasa cu câte o poziție spre stânga cât timp fracția indicată este cel puțin 1. Dacă în urma operațiilor ambii pointeri nu se află în pozițiile inițiale atunci nu există soluție cu produs în $[1, +\infty)$. Altfel subșirul este format din primele două sau ultima fracție.

Soluția 2.



Pentru a rezolva problema vom împarți fracțiile în 7 categorii în funcție de rezultat. Mai departe ne vom referi prin

- **Categoria I** la mulțimea fracțiilor cu rezultatul în $(-\infty, -1)$
- **Categoria II** la mulțimea fracțiilor cu rezultatul în $[-1, -1]$
- **Categoria III** la mulțimea fracțiilor cu rezultatul în $(-1, 0)$
- **Categoria IV** la mulțimea fracțiilor cu rezultatul în $[0, 0]$
- **Categoria V** la mulțimea fracțiilor cu rezultatul în $(0, 1)$
- **Categoria VI** la mulțimea fracțiilor cu rezultatul în $[1, 1]$
- **Categoria VII** la mulțimea fracțiilor cu rezultatul în $(1, +\infty)$

O idee de rezolvare este să considerăm fiecare din cele 7 intervale în ordine descrescătoare și să verificăm dacă există un subșir de fracții care înmulțite dau o valoare în intervalul la care ne referim.

- $(1, +\infty)$. Pentru a obține un produs mai mare ca 1 e nevoie ca cel puțin una din următoarele propoziții sa fie adevărate:
  - există o fracție în VII;
  - există două fracții în I;
  - există o fracție în I și una în II;
  - există o fracție în I și una în III care înmulțite dau mai mult decât 1.

  Pentru a obține valoarea maxima se aleg în această ordine:
  - toate fracțiile din VII.
  - cele mai mici fracții din mulțimea I două câte două. Astfel, după alegere, în I fie nu rămâne nimic (dacă $|I|$ este număr par), fie rămâne cea mai mare fracție ($|I|$ este impar).
  - dacă a rămas o fracție în I și există o fracție în II atunci ambele sunt alese.
  - dacă a rămas o fracție în I și există o fracție în III al căror produs este > 1 atunci ambele sunt alese.
- $[+1, +1]$.
  Din moment ce suntem aici înseamnă că nu există un subșir ale cărui fracții să dea un produs > 1. Pentru a obține un produs egal cu 1 e nevoie ca cel puțin una din următoarele sa fie adevărată:
  - exista o fracție în VI;
  - exista 2 fracții în II;
  - exista o fracție în I și una în III care înmulțite dau 1.
- $(0, +1)$.
  Din moment ce suntem aici înseamnă că nu există un subșir ale cărui fracții să dea un produs ≥ 1. Pentru a obține un produs din $(0, +1)$ e nevoie ca cel puțin una din următoarele sa fie adevărată:
  - exista o fracție în V;
  - exista o fracție în II și una în III;
  - exista o fracție în I și una în III care înmulțite aparțin $(0, +1)$;
  - exista 2 fracții în III.

  Se alege subșirul care obține cel mai mare produs raportat la cele 4 cazuri de mai sus.
- $[0, 0]$.
  Dacă există cel puțin o fracție cu numărătorul 0 atunci răspunsul este 0.
- $(-1, 0)$ **sau** $[-1, -1]$ **sau** $(-\infty, -1)$.
  Dacă suntem în acestă situație atunci cu siguranță testul conține o singură fracție și aceasta este negativă. Prin urmare aceasta reprezintă și soluția problemei.

Problema 2: Geogra

Propusă de: stud. Gheorghe Liviu Armand, Universitatea din București

Pregătită de: stud. Gheorghe Liviu Armand, Universitatea din București

Editorial redactat de: stud. Gheorghe Liviu Armand, Universitatea din București

Pentru a rezolva problema, trebuie ca mai întâi sa calculăm șirul T.

Ordonăm șirurile R găsite de Alex în fiecare runda crescător după numărul de valori diferite de -1 din cadrul fiecărui șir. Întrucât știm că Alex respectă șirul T în fiecare rundă, putem să deducem un șir T posibil doar uitându-ne la șirurile R în ordinea stabilită anterior. Astfel, când vedem că numărul de valori diferite de -1 diferă de la un șir R la următorul șir R din ordinea stabilită, ne putem da seama că pozițiile pe care în primul sir sunt valori de -1 și în al doilea șir sunt valori diferite de -1 vor urma în șirul T după pozițiile pe care în primul sir sunt valori diferite de -1. Pentru a determina un șir T posibil este îndeajuns ca de fiecare dată când găsim astfel de poziții să le adaugăm în șirul T în ordinea în care le găsim. Șirul T obținut nu este mereu unic și este posibil să difere de cel al lui Alex, însă soluția nu este afectată de acest fapt, după cum vom vedea în continuare.

Pe baza acestui sir T obținut, putem rearanja valorile din fiecare sir R și Z_i. Astfel, valorile din fiecare sir vor fi în ordinea în care Alex le va afla.

Pentru a putea răspunde eficient la întrebări, vom construi $L+1$ vectori în care vom reține cele N locații. Al j-lea vector dintre aceștia, $0 \le j \le L$, va conține cele N locații ordonate după numărul format din primii j biți din cadrul șirului Z_i asociat fiecărei locații.

Pentru fiecare întrebare, ne vom uita în al nr-lea vector dintre cei $L+1$ formați anterior, unde nr reprezintă numărul valorilor diferite de -1 din șirul R asociat rundei respective. În acest vector, putem căuta binar cea mai din stânga și cea mai din dreapta apariție a numărului format din biții ce reprezinta primele nr valori din cadrul șirului R din runda actuala. Astfel, vom obține un interval nevid ce reprezinta toate locațiile care respectă șirul R aflat de Alex în runda actuala.

Cerința 1 – 37 de puncte

Pentru a rezolva aceasta cerința, trebuie doar să afișăm lungimea intervalului găsit anterior.

Cerința 2 – 63 de puncte

Pentru a rezolva această cerința, mai întâi trebuie să observăm că putem calcula A independent de B.

Pentru a calcula A, trebuie sa ne uitam la coordonatele X ale locațiilor din intervalul găsit anterior. Dacă valorile W ale locațiilor din interval ar fi 1, atunci A ar fi valoarea mediană a coordonatelor X ale locațiilor din interval. În cazul în care șirul are 2 mediane, atunci se alege valoarea mai mica, întrucât A trebuie sa fie minim. Dacă A ar fi o valoare mai mică decât cea mai mică mediană sau mai mare decât cea mai mare mediană, atunci d_p ar crește. Pe cazul general, unde valorile W ale locațiilor din interval nu sunt toate egale cu 1, ne putem imagina că fiecare locație apare de W-ul ei ori. Astfel, răspunsul ar fi tot mediana acestui nou (și mult mai lung) sir, adică a $sumW/2 + sumW\%2$ valoare X in ordine crescătoare, unde $sumW$ reprezinta suma valorilor W ale locațiilor din interval. Deci, dacă ne-am ordona după X locațiile din intervalul găsit cu cele doua căutări binare, atunci A va fi coordonata X a primei locații în care suma valorilor W a locațiilor de dinaintea ei în ordinea stabilita devine mai mare sau egala cu $sumW/2 + sumW\%2$. Pentru a reuși să calculăm acest lucru, vom ordona locațiile din interval după coordonata X, vom face sume parțiale după W și vom parcurge locațiile din interval până găsim valoarea căutată.

Pentru a calcula B se procedează asemănător, singura diferența fiind că B trebuie calculat în funcție de coordonatele Y ale locațiilor din intervalul găsit în runda actuală.

Pentru a obține 100 de puncte, trebuie să calculăm o singură dată rezultatul din runde care au același sir R. Astfel, fiecare interval găsit pentru un sir R va fi parcurs o singură data. Întrucât un sir R cu un anumit număr de valori diferite de -1 determină un interval disjunct de intervalul determinat de un sir R diferit cu același număr de valori diferite de -1, fiecare element din cei $L+1$ vectori făcuți anterior va fi parcurs o singura data în total.

PROBLEMA 3: SCHI

Propusă de: prof. Stelian Ciurea, Universitatea "Lucian Blaga", Sibiu

Pregătită de: stud. Ioan-Bogdan Iordache, Universitatea din București

Editorial redactat de: stud. Ioan-Bogdan Iordache, Universitatea din București

Soluție în complexitate pătratică — 24 de puncte. Cutiile sunt procesate în ordinea în care sunt adăugate în turn. Notăm cutiile cu $c_1, c_2, \ldots, c_N$.

Fie două cutii c_i, c_j cu $i < j$. Considerăm că c_i a fost deja plasată în turn și cunoaștem top_i înălțimea la care se află fața superioară a acestei cutii (eventual înălțimea la care se află fața corespunzătoare capacului lipsă dacă c_i este o cutie de tipul 2, "cu golul în sus"). Știind top_i și implicit poziția cutiei c_i în spațiu, putem determina o restricție pentru top_j, considerând că nu mai există alte cutii în afară de c_i care să întrerupă căderea lui c_j.

Se disting mai multe cazuri în funcție de tipul și orientarea cutiilor (M_i, M_j) și diferența dintre dimensiunile laturilor capacelor (L_i, L_j). Vom arăta mai jos câteva cazuri mai interesante:

- dacă $M_i = 2$, $M_j = 1$ și $L_i < L_j$, obținem restricția $top_j \geq top_i + H_j$
- dacă $M_i = 2$, $M_j = 1$ și $L_i > L_j$, obținem restricția $top_j \geq top_i - H_i + H_j$
- dacă $M_i = 1$, $M_j = 0$ și $L_i < L_j$, obținem restricția $top_j \geq top_i$

De asemenea, cutiei c_j i se mai impune o restricție și de către podea: $top_j \geq H_j$.

Calculând pentru cutia c_j restricțiile impuse de toate cutiile $c_1, c_2, \ldots, c_{j-1}$ și de către podea, putem afla top_j ca fiind maximul dintre acestea.

Înălțimea turnului va fi în final $\max\{top_1, top_2, \ldots, top_N\}$.

Pentru a determina care cutii sunt vizibile din lateral, vom defini "intervalul de vizibilitate" al unei cutii intervalul de înălțimi la care acea cutie este vizibilă (evident ne va interesa pentru câte cutii acest interval are lungime nenulă).

La început, pentru toate cutiile de forma c_i inițializăm intervalul de vizibilitate cu $[top_i - H_i, top_i]$ (intervalul de înălțimi pe care îl ocupă cutia c_i). Cutia c_i poate fi ascunsă/acoperită parțial sau total doar de cutii c_j, pentru care $L_j > L_i$. Așadar, ne vom uita la astfel de cutii c_j, care fie nu acoperă în niciun fel cutia c_i, fie o acoperă total, deci deja putem stabili că c_i nu este vizibilă din lateral, fie parțial, caz în care c_j acoperă fie un prefix, fie un sufix al intervalului de vizibilitate al lui c_i.

Matematic, dacă notăm viz_i intervalul de vizibilitate al lui c_i calculat până la un pas intermediar și dorim să actualizăm acest interval pe baza unei cutii c_j cu $L_j > L_i$, realizăm diferența de intervale/mulțimi: $viz_i \leftarrow viz_i \setminus [top_j - H_j, top_j]$. Este ușor de observat din procesul de plasare al cutiilor, că această diferență va rezulta mereu într-un interval.

Rezolvând astfel ambele cerințe ale problemei, obținem complexitatea $\mathcal{O}(N^2)$.

Soluție în complexitate liniară — 100 de puncte. Pentru a obține 100 de puncte, va fi nevoie de o rafinare a soluției de mai sus. Pentru a rezolva prima cerință vom încerca să calculam din nou top_i pentru fiecare cutie c_i. Observația care va reduce complexitatea este că nu avem nevoie să calculăm restricțiile impuse de $c_1, \ldots, c_{i-1}$, cutiei c_i, ci doar de către o parte dintre ele.

Notăm $c_{i_1}, c_{i_2}, \ldots, c_{i_k}$ ($1 \leq i_1 < i_2 < \ldots < i_k < i$) cutiile care ar putea întrerupe căderea lui c_i. Observăm ca acest șir de cutii este descrescător după L: dacă am avea c_{i_j} și $c_{i_{j+1}}$ cu $L_{i_j} < L_{i_{j+1}}$, cutia c_{i_j} ar fi inutilă (oricum ar cădea, c_i se va lovi mai întâi de $c_{i_{j+1}}$, nu de c_{i_j}).

O altă observație utilă este că pentru două cutii c_{i_j} și $c_{i_{j+1}}$ cu $L_{i_j} > L_{i_{j+1}} > L_i$, restricția impusă de $c_{i_{j+1}}$ cutiei c_i va fi cel puțin la fel de mare ca restricția impusă de c_{i_j}. Cu alte cuvinte, în subșirul descrescător de cutii $c_{i_1}, c_{i_2}, \ldots, c_{i_k}$, pentru a determina top_i vom calcula restricțiile doar cu un sufix al acestui subșir (cutiile cu L mai mic decât L_i, și cutia cu L minim mai mare decât L_i).

După ce am calculat top_i, eliminăm din subșir sufixul format din cutii cu L mai mic decât L_i și adăugăm cutia c_i la finalul subșirului. Astfel per total numărul de restricții calculat pentru toate cutiile este direct proporțional cu numărul de ștergeri din acest subșir, iar din moment ce o cutie este adăugată și ștearsă din subșir cel mult o dată, complexitatea va fi liniară.

Pentru a determina vizibilitatea cutiilor, vom folosi noțiunea de interval de vizibilitate descrisă mai sus. Vom actualiza aceste intervale pentru toate cutiile mai întâi considerând doar acoperirile realizate de cutiile de tipul 2 (cu golul în sus) și apoi separat cele realizate de cutii de tipul 0 (cu golul în jos).

Vom explica în continuare procesul pentru acoperirile produse de cutii de tipul 2. Vom procesa cutiile în ordinea în care au fost adăugate în turn. În momentul de față suntem la cutia c_i de un tip oarecare și vrem să vedem ce cutii de tipul 2 o acoperă parțial/total. Evident aceste cutii, ca să poată acoperi c_i, trebuiau adăugate înainte și să aibă L mai mare decât L_i. Dintre cutiile de tip 2 care respectă aceste condiții, ne interesează *top*-ul maxim (extremitatea superioară maximă) a acestor cutii care poate acoperi un prefix al intervalului de vizibilitate pentru c_i.

O observație care ne va duce la soluția dorită este că dacă după o cutie de tip 2 se adaugă la un pas ulterior o cutie cu L mai mare (de orice tip), cutia de tip 2 nu va mai acoperi pe nimeni începând cu acest moment.

De aceea, la pasul i putem menține subșirul de cutii de tip 2: $c_{i_1}, c_{i_2}, \ldots, c_{i_k}$ $(1 \leq i_1 < i_2 < \ldots < i_k < i)$ cu proprietatea că $L_{i_1} > L_{i_2} > \ldots > L_{i_k}$. Când procesăm cutia c_i eliminăm din subșir toate cutiile cu L mai mic decât L_i (acestea nu pot acoperi c_i și nicio altă cutie adăugată în viitor). Presupunem că rămânem acum cu k' cutii în subșir, pentru a determina ce prefix din intervalul de vizibilitate al lui c_i ar fi acoperit, trebuie să determinăm $max\{top_{i_1}, top_{i_2} \ldots, top_{i_{k'}}\}$. Acesta nu este altceva decât un maxim pe prefix pentru subșirul nostru, pe care îl putem menține în complexitate constantă atunci când modificăm subșirul. După acest pas, în cazul în care c_i este de tipul 2 o adăugăm la finalul subșirului și continuăm cu cutia $i+1$.

Pentru determinarea acoperirilor produse de cutii de tipul 0 (cu golul în jos) strategia este similară, însă cutiile vor trebui parcurse în ordinea inversă a adăugării lor în turn (”de sus în jos”).

În final, complexitatea obținută pentru ambele cerințe este $\mathcal{O}(N)$.

Comisia științifică

- prof. Nodea Gheorghe-Eugen - Colegiul Național ”Tudor Vladimirescu”, Tg. Jiu
- prof. Cheșcă Ciprian - Liceul Tehnologic ”Grigore C. Moisil”, Buzău
- prof. Candale Silviu - Colegiul Național ”Liviu Rebreanu”, Bistrița Năsăud
- prof. Ciurea Stelian - Universitatea ”Lucian Blaga”, Sibiu
- student Arhire Andrei - Universitatea A.I. Cuza, Iași
- student Cismaru Mihaela - NetRom Software, Universitatea din Craiova
- asist. doctorand Constantinescu Andrei Costin - ETH Zürich
- student Gheorghe Liviu Armand - Universitatea din București
- student Iordache Ioan-Bogdan - Universitatea din București
- student Petrescu Alexandru - Universitatea din Oxford
- student Popescu Mihai Cristian - Universitatea Babeș-Bolyai Cluj-Napoca