empowersoft

**Concours online de informatică**
**Categoria PROGRAMARE**

**Secţiunea 7-8 avansaţi**

## DESCRIERE SOLUŢII SECŢIUNEA 7-8 AVANSAŢI

**PROBLEMA 1** **autor prof.Constantin Scheau**

**BARA**

Se vor elimina pe rând piesele aflate la distanţe minime de găurile din bară. Pentru fiecare piesa se va calcula distanţa faţa de gaura cea mai apropiată şi se vor contoriza deplasările.

**PROBLEMA 2** **propunator prof. Luminita Nastase**

**CLĂTITE**

Parcurgem toate numerele dintre N și M, iar pentru fiecare număr calculăm simultan valorile pentru primele 3 tipuri de clătite astfel: descompunem în factori primi numărul (mergând pana la radical din acesta) și ne folosim de următoarele formule/ proprietăți:

Tipul 1:
Dacă nu am găsit niciun divizor al numărului, acesta e *prim* și îl contorizăm.

Tipul 2:
Un număr este *pătrat perfect* dacă toți exponenții ($d_1$, $d_2$, ... $d_k$) sunt divizibili cu 2.
Un număr este *cub perfect* dacă toți exponenții ($d_1$, $d_2$, ... $d_k$) sunt divizibili cu 3.
Folosind aceste două relații putem determina pe parcurs dacă numărul este sau nu pătrat respectiv cub perfect.

Tipul 3:
Calculam pe parcurs suma divizorilor numărului cu următoarea formulă: fie numărul N și descompunerea sa în factori primi, de forma: $\boldsymbol{p_1^{d1} * p_2^{d2} * \ldots * p_k^{dk}}$.

Suma divizorilor numărului N este dată de relația : $\boldsymbol{\frac{p_1^{d1+1}-1}{p_1-1} * \frac{p_2^{d2+1}-1}{p_2-1} * \ldots * \frac{p_k^{dk+1}-1}{p_k-1}}$.
Astfel, verificăm dacă suma calculată – numărul propriu-zis = numărul propriu-zis (suma de mai sus include și numărul).

Dacă niciuna dintre relațiile de mai sus nu este îndeplinită verificăm pur și simplu paritatea numărului (tipul 4 dacă e *par* respectiv tipul 5 dacă e *impar*).

Complexitatea totală: O(N sqrt(N))

*Ploiești, 6 iunie 2015*

**PROBLEMA 3** **autor prof. Radu Visinescu**

## INTERVALE

Descriere soluție: Se sortează crescător acele elemente distincte. Apoi vom folosi două cicluri while, incluse unul in altul. În cadrul primului ciclu, vom determina pozițiile de început ale secvențelor de numere consecutive. În ciclul interior parcurgem, una câte una, aceste secvențe. La sfârșitul unei astfel de parcurgeri, dacă secvența este formată din cel putin două numere, se va tipări și ultimul element al secvenței. Programul este următorul:

```
{sortarea vectorului X}
i:=1;
while i <= n do
        begin
        write('[',x[i]);
        L:=0;
        while x[i]+1=x[i+1] then
                begin
                i:=i+1;
                L:=L+1
                end;
        if L >0 then write ('..',x[i]);
        write(']+');
        i:=i+1
        end;
```

**PROBLEMA 4**

## HODOR

**Autori**
**Ilie Andrei Cătălin (enunț, descriere soluție, sursă oficială)**
**Vlad Costin Alexandru (sursă oficială, generare teste, verificare surse)**

Ideea de bază constă în sortarea castelelor cu apărători după unghiul pe care acestea le fac cu axa Ox (unghiurile dintre semidreapta obținută prin unirea castelelor cu originea și axa Ox).
Astfel, printr-o sortare de complexitate n log n, vom obține un vector în care castelele se află în ordinea în care acestea ar fi întâlnite de o semidreaptă ce se rotește în jurul originii, începând de la un unghi făcut cu axa Ox de 0 grade și terminând la un unghi de 180 de grade. După ce avem castelele sortate după unghiul pe care acestea le fac cu axa Ox, problema se reduce la următoarea: fiind dat un vector v de dimensiune n , aflați indicele i strict cuprins între 0 si n+1, astfel încât modulul diferenței dintre suma elementelor de la poziția 1 până la poziția i-1 și suma elemntelor de la poziția i+1 până la poziția n să fie minimă (cu cazurile particulare atunci când i este chiar 1 sau n).
Rezolvarea acestei probleme este elementară, rezolvându-se liniar. Astfel, complexitatea optimă a algoritmului este n log n.
Soluțiile de n^2 obțin 50 de puncte, acestea făcându-se sau fără sortare, sau cu o sortare ineficientă.

**PROBLEMA 5** **propunator prof. Luminita Nastase**

**CIOCOLATA**

Problema cere, de fapt, determinarea primei submatrice a cărei elemente au suma maximă (prima = cea care respectă condițiile de coordonate minime, conform enunțului).

1. Metoda naivă este aceea de a fixa toate perechile de indici (X, Y) atât pentru colțul stânga sus al submatricei cât și pentru colțul dreapta jos și de a itera între acestea pentru a calcula suma. Această soluție are complexitate O(N^6) și ar trebui să obțina 20p.

2. Putem rafina puțin ideea de mai sus, ținand o matrice de sume parțiale. Astfel, după ce setăm cei doi indici, putem calcula suma în timp constant. Complexitate se reduce la O(N^4) și ar trebui să obțină 40p.

3. Soluția optimă folosește următoarea strategie: Calculăm sume parțiale pe coloane, pornind din elementul 1 și coborând spre elementul N. Fixăm apoi două linii (nu neaparat distincte), fie acestea x1 și x2 (x1 <= x2). Se poate observa că cele două linii alese delimitează un "culoar". Putem considera acest culoar ca fiind un șir de M numere unde fiecare număr este de fapt suma pe coloană între cei doi indici. Pe "șirul" nou format se va aplica algoritmul clasic de găsire a unei subsecvenţe de sumă maximă și se memorează maximul pentru toate perechile (x1, x2).
Solutia are complexitatea O(N^3) și obține 100p.