Zadaci sa rešenjima Kvalifikacije I 2011.

zadatak: Pomnozi

Data su tri broja *A*, *B* i *C*. Ispisati vrednost izraza *A* - *B* * *C*.
Ulaz.
(Ulazni podaci se učitavaju sa standardnog ulaza.) U prvom redu standardnog ulaza nalaze se prirodni brojevi *A*, *B* i *C* ($0 \leq A, B, C \leq 2^{32} - 1$).
Izlaz.
(Izlazne podatke ispisati na standardan izlaz.) U prvi i jedini red ispisati vrednost izraza *A* - *B* * *C*.
Ograničenja.
U 50% test primera će *A*, *B* i *C* imati vrednost iz intervala [0, 1000].
Primer 1.

standardni ulaz	standardni izlaz
1 2 3	-5

Objašnjenje.
1 - 2 * 3 = 1 - 6 = -5.
Primer 2.

standardni ulaz	standardni izlaz
0 0 0	0

fajl: pomnozi.cpp

```
#include <iostream>
#include <cstdio>

using namespace std;

int main(){
  unsigned long long A, B, C;
  scanf("%llu %llu %llu", &A, &B, &C);
  B *= C;
  if (B > A)
    printf("-%llu\n", B - A);
  else
    printf("%llu\n", A - B);

  return 0;
}
```

zadatak: Frekventna suma

Dat je niz brojeva $A = (a_1, ..., a_N)$. Posmatrajmo skup svih suma uzasotpnih članova $S = \{a_i + ... + a_j \mid 1 \leq i \leq j \leq N\}$.
Ispisati vrednost iz skupa *S* koja se najčešće pojavljuje, kao i koliko puta se pojavljuje. U slučaju da ima više takvih, ispisati onu čija je vrednost najveća.
Ulaz.
(Ulazni podaci se učitavaju sa standardnog ulaza.) U prvom redu standardnog ulaza nalazi se prirodan broj *N* ($1 \leq N \leq 3000$). U sledećem redu nalazi se *N* prirodnih brojeva, redom a_1 , ..., a_N, svaki iz intervala [0, 3000].
Izlaz.
(Izlazne podatke ispisati na standardan izlaz.) U prvi i jedini red standardog izlaza ispisati dva prirodna broja, koji redom predstavljaju, broj koji se najšeće pojavljuje u skupu *S*, i koliko puta se pojavljuje. U slučaju da postoji više takvih brojeva, ispisati onaj koji ima najveću vrednost.
Ograničenja.
U 30% test primera će biti $1 \leq N \leq 100$.
Primer 1.

standardni ulaz	standardni izlaz
3 1 2 3	3 2

Objašnjenje.

Skup S = {1, 1 + 2, 1 + 2 + 3, 2, 2 + 3, 3} = {1, 3, 6, 2, 5, 3}.

Primer 2.

standardni ulaz	**standardni izlaz**
8 17 13 17 13 5 6 5 6	30 3

Objašnjenje.

Primetimo da se i suma 11 pojavljuje 3 puta, ali je 30 veća suma.

fajl: freksuma.cpp

```
#include <iostream>
#include <cstdio>
#define ffor(_a,_f,_t) for(int _a=(_f),__t=(_t);_a<__t;_a++)
#define SET(__set, val) memset(__set, val, sizeof(__set))
#define FOR(__i, __n) ffor (__i, 0, __n)

using namespace std;

int cnt[9000001], sum[3001];

int main(){
  int n, val;
  scanf("%d", &n);
  sum[0] = 0;
  FOR (i, n){
    scanf("%d", &val);
    sum[i + 1] = sum[i] + val;
  }

  SET(cnt, 0);

  ffor (i, 1, n + 1){
    val = sum[i];
    FOR (j, i)
      cnt[val - sum[j]]++;
  }
  int mm = 0, ret = -1;

  FOR (i, 9000001)
    if (cnt[i] >= mm){
      mm = cnt[i];
      ret = i;
    }

  printf("%d %d\n", ret, mm);

  return 0;
}
```

zadatak: Will Rogers

Will Rogers fenomen se dobija kada se element iz jednog skupa prebaci u drugi, pri čemu se srednje vrednosti oba skupa povećaju. Baziran je na citatu Willa Rogera:
Kada su Okie (domorovci) napustile Oklahomu i preselili se u Kaliforniju, podigli su prosečan nivo inteligencije u obe države.
Data su dva skupa prirodnih brojeva *a* i *b* veličine *n* odnosno *m*. Naći broj elemenata skupa *a* koji prebacivanjem u skup *b* povećavaju prosečne vrednosti oba skupa.

Ulaz.

(Ulazni podaci se učitavaju sa standardnog ulaza.) U prvom redu standarnog ulaza nalaze se dva prirodna broja *n* i *m* ($2 \leq n, m \leq 5000$). Naredna dva reda sadrže po *n* i *m* prirodnih brojeva koji predstavljaju elemente nizova *a* i *b*, redom. Elementi nizova su iz segmenta [1, 1000].

Izlaz.

(Izlazne podatke ispisati na standardan izlaz.) U prvom i jedinom redu standardnog izlaza ispisati traženi broj.

Primer 1.

standardni ulaz	**standardni izlaz**
3 3 6 5 4 1 2 3	1

Objašnjenje.

Jedini element koji zadovoljava uslove je broj: 4.

fajl: willrogers.cpp

```
#include<stdio.h>
#define MAX_N 5005

int n, m, a [MAX_N], b [MAX_N], sol;
double avgA, avgB;

  void input()
  {
    scanf ("%d %d", &n, &m);
    for (int i = 0; i < n; i++)
      scanf ("%d", &a [i]);
    for (int i = 0; i < m; i++)
      scanf ("%d", &b [i]);
  }

  void solve()
  {
    avgA = 0;
    for (int i = 0; i < n; i++)
      avgA += a [i];
    avgA = avgA / n;

    avgB = 0;
    for (int i = 0; i < m; i++)
      avgB += b [i];
    avgB = avgB / m;

    sol = 0;
    for (int i = 0; i < n; i++)
      if ((a [i] < avgA) && (avgB < a [i]))
        sol++;
  }

  int main()
  {
    input();
    solve();
    printf ("%d\n", sol);

    return 0;
  }
```

zadatak: Segmenti

Dato je n segmenata i m tačaka na x-osi. Za svaku od datih m tačaka odrediti broj segmenata kojima ona pripada. Tačka x pripada segmentu $[a, b]$ ako je $a \leq x \leq b$.

Ulaz.

(Ulazni podaci se učitavaju sa standardnog ulaza.) U prvom redu standradnog ulaza nalaze se dva prirodna broja $n \leq 10^5$ i $m \leq 10^5$ - broj segmenata i broj tačaka, redom. U sledećem redu se nalaze m brojeva razdvojenih razmakom - koordinate tačaka. U sledećih n redova se nalaze po dva broja razdvojena

razmakom - leva i desna koordinata odgovarajućeg segmenta (leva koordinata je strogo manja od desne). Sve koordinate su prirodni brojevi ne veći od 10^9.

Izlaz.

(Izlazne podatke ispisati na standardan izlaz.) Na standardni izlaz za svaku tačku ispisati broj segmenata kojima ona pripada, svaki broj u posebnom redu i u redosledu kojim su tačke date na ulazu.

Primer 1.

standardni ulaz	**standardni izlaz**
3 4 5 1 8 9 6 7 4 9 2 5	2 0 1 1

fajl: segmenti.pas

```
const
  MaxN = 100010;
  MaxM = 100010;

type
  Point = Record
    x : longint;
    t : longint;
    num : longint;
  end;

var
  a : array[0..2*MaxN + MaxM] of Point;
  sol : array[0..MaxM] of longint;
  n, m, i, sum : longint;
  x, y : Point;


procedure QS(l, r : longint);
var
  i, j: longint;
begin
  i := l; j := r; x := a[(l + r) DIV 2];
  repeat
    while ((a[i].x < x.x) or ((a[i].x = x.x) and (a[i].t > x.t))) do i := i + 1;
    while ((x.x < a[j].x) or ((x.x = a[j].x) and (x.t > a[j].t))) do j := j - 1;
    if (i <= j) then begin
      y := a[i]; a[i] := a[j]; a[j] := y;
      i := i + 1; j := j - 1;
    end;
  until (i > j);
  if (l < j) then QS(l, j);
  if (i < r) then QS(i, r);
end;

begin

   readln(n, m);
   for i := 0 to m - 1 do begin
     read(a[i].x);
     a[i].t := 0;
     a[i].num := i;
   end;
   for i := 0 to n - 1 do begin
     readln(a[m + 2*i].x, a[m + 2*i + 1].x);
     a[m + 2*i].t := 1;
     a[m + 2*i + 1].t := -1;
   end;
```

```
   QS(0, m + 2 * n - 1);
   sum := 0;
   for i := 0 to m + 2 * n - 1 do begin
     if (a[i].t = 0) then
       sol[ a[i].num ] := sum
     else
       sum := sum + a[i].t;
   end;

   for i := 0 to m - 1 do
     writeln(sol[i]);

end.
```

fajl: segmenti.cpp

```
#include <cstdlib>
#include <cstdio>
#include <algorithm>

using namespace std;

const int MaxN = 100010;
const int MaxM = 100010;


int n, m, sum;

struct Point {
  int x;      // type =  1 => levi kraj segmenta
  int type;    // type =  0 => jedna od m tacaka
  int num;    // type = -1 => desni kraj segmenta
};

Point a[2 * MaxN + MaxM];
int sol[MaxM];

bool cmp(Point A, Point B) {
  return ((A.x < B.x) || (A.x == B.x && A.type > B.type));
}

int main() {

  scanf("%d%d", &n, &m);
  for (int i = 0; i < m; i++) {
    scanf("%d", &a[i].x);
    a[i].type = 0;
    a[i].num = i;
  }
  for (int i = 0; i < n; i++) {
    scanf("%d%d", &a[m + 2 * i].x, &a[m + 2 * i + 1].x);
    a[m + 2 * i].type = 1;
    a[m + 2 * i + 1].type = -1;
  }

  sort(a, a + m + 2 * n, cmp);

  sum = 0;
  for (int i = 0; i < m + 2 * n; i++) {
    if (a[i].type == 0)
      sol[ a[i].num ] = sum;
    else
      sum += a[i].type;
  }

  for (int i = 0; i < m; i++)
```

```
    printf("%d\n", sol[i]);

  return 0;
}
```

zadatak: O-Timberlend

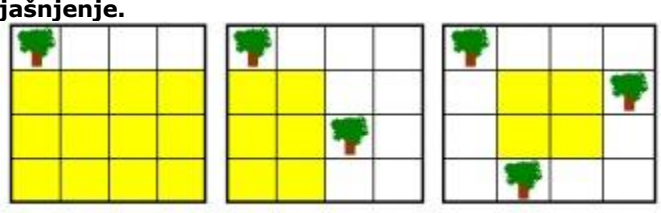
Glavna zanimacija stanovnika Timberlenda je timberbay. Svake godine se proglašava pobednik, a to je stanovnik koji je imao najviše tačnih odgovora. Sem stanovnika, koji su aktivni danju, u Timberlendu žive još growlini i cutlini koji su aktivni noću. Samo tada oni izlaze u timberdolinu i menjaju njen izgled. Growlini sade drveće, posipaju ih magičnim prahom i drveća do jutra porastu. Cutlini opet pojedu celo drvo ne ostavljajući panjeve za sobom.
Rano u jutro dok još nije svanulo stanovnici Timberlenda izvlače iz šešira brojeve Q_i, a čim svane moraju brzo da podignu jednu od dve palice: "yes" ili "no" palicu. Palicu "yes" treba podići ako je u Timberdolini moguće postaviti pravugaoni teren za timberbol površine Q_i izmedju drveća, a palicu "no" ako to nije moguće. Svaki tačan odgovor donosi poen za godišnju nagradu. (Pod pravougaonim terenom se podrazumevaju samo tereni koji imaju stranice paralelne granicama Timberdoline i kojima su stranice celobrojne.)
Ulaz.
(Ulazni podaci se učitavaju sa standardnog ulaza.) Iz prvog reda standardnog ulaza čitaju se dva broja N i Q. N predstavlja dimenziju kvadratne Timberdoline ($2 \le N \le 100$). Q predstavlja broj dogadjaja. U sledećih Q redova nalaze se dogadjaji. Oni mogu biti:

- **grow *x y*** Growlini su posadili drvo na polju (x, y)
- **cut *x y*** Cutlini su pojeli drvo na polju (x,y)
- **vote *p*** Pitanje "da li je u timberdolinu moguće postaviti pravougaoni teren površine p"

($1 \le Q \le 50000$), ($1 \le p \le N^2$), ($1 \le x, y \le N$), i neće biti više od 365 dogadjaja "vote p" (za svaki dan u godini po jedno odgovor. Setite se proglašenje pobednika je na kraju godine).
Izlaz.
(Izlazne podatke ispisati na standardan izlaz.) Na standardnom izlazu, za svaki dogadjaj "vote p" potrebno je u posebnom redu ispisati "yes" ili "no".
Napomena.
U početku nema nijednog drveta.
Primer 1.

standardni ulaz	standardni izlaz
4 11	no
grow 1 4	yes
vote 10	no
vote 12	yes
grow 3 2	yes
vote 7	no
vote 6	
grow 4 3	
grow 2 1	
cut 3 2	
vote 4	
vote 9	

Objašnjenje.



vote 10 - odgovor je "no" jer nije moguće postaviti pravougaonu oblast povrsine 10
vote 12 - odgovor je "yes" jedna mogucnost prikazana je na slici "Day 1"

vote 7 - odgovor je "no" jer nije moguće postaviti pravougaonu oblast povrsine 7
vote 6 - odgovor je "yes" jedna mogucnost prikazana je na slici "Day 2"
vote 4 - odgovor je "yes" jedna mogucnost prikazana je na slici "Day 3"
vote 9 - odgovor je "no" jer nije moguće postaviti pravougaonu oblast površine 9

fajl: o-timberlend.pas

```
var Polje, Pom   : Array[0..102, 0..102] of Word;
    Faktx, FaktY : Array[1..100] of Word;
    Dokle, Xmin, Ymin, N, P, Q, i, j, k, x, y, w, Area, BrFakt : Word;
    c: Char;

Function MozeLi:String;
var i, j : Integer;
Begin
  For k := 1 to BrFakt do
    if (FaktX[k] <= n) and (Fakty[k] <= n)
      then for i := 0 to N-1 do
             for j := 0 to N-1 do
               if (i+FaktX[k] <= n) and (j+FaktY[k] <= n)
                 then if Pom[i+FaktX[k], j+FaktY[k]] + Pom[i, j] =
                         Pom[i, j+FaktY[k]] + Pom[i+FaktX[k], j]
                        then Begin
                               MozeLi := 'yes';
                               exit;
                             end;
  MozeLi := 'no';
end;

begin
  readln(N, Q);
  For w := 1 to Q do
  Begin
    Read(c);
    Case C of
    'g' : Begin
            Read(c);Read(c);Read(c);
            Readln(x, y);
            Polje[x,y] := 1;
          end;
    'c' : Begin
            Read(c);Read(c);
            Readln(x, y);
            Polje[x,y] := 0;
          end;
    'v' : Begin
            Read(c);Read(c);Read(c);
            Readln(Area);

          { rastavljanje na parove do korena }

            j := 0;
            For i := 1 to Trunc(Sqrt(Area)) do
              if (Area mod i = 0)
                then Begin
                       Inc(j);
                       FaktX[j] := i;
                       FaktY[j] := Area div i;
                     end;
          { formiranja parova preko korena }

            For i := 1 to j do
            Begin
              Inc(j);
              FaktX[j] := FaktY[i];
```

```
              FaktY[j] := FaktX[i];
            end;
            BrFakt := j;

          { Upisujem koliko drveca ima u pravoug. oblasti
            od donjeg levog ugla do [i,j] (dinamika) }

            For i := 1 to N do
              For j := 1 to N do
                Pom[i,j] := Polje[i,j] + Pom[i-1,j] + Pom[i,j-1] + Polje[i,j] - Pom[i-1,j-
1];

          { u funkciji MozeLi, vrsi se provera ima li drveca unutar
            svakog pravougaonika koji staje u matrice }

            Writeln(MozeLi);
          end;
    end;
  end;
end.
```