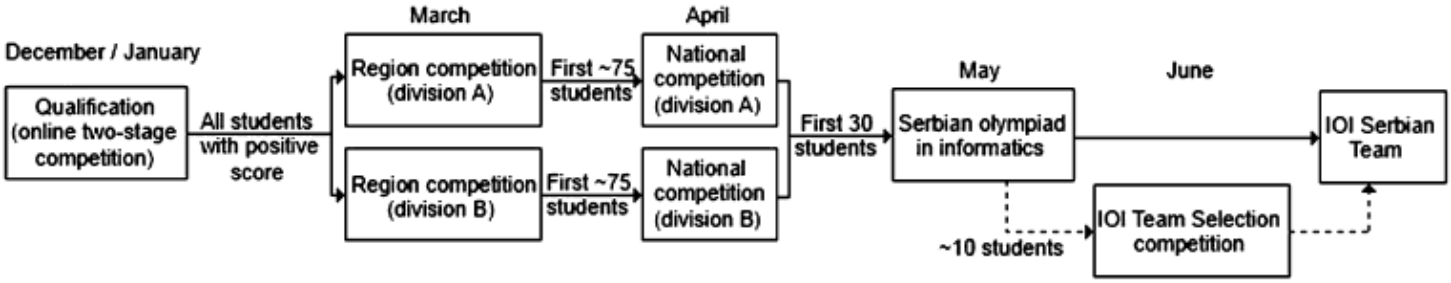
Припремни задаци пред квалификације

http://www.yuoi.nis.edu.rs
http://www.z-trening.com/competitions.php



ON-LINE trening
Z-TRENING http://www.z-trening.com
TIMUS http://acm.timus.ru/
TOPCODER http://www.topcoder.com/tc
USACO http://train.usaco.org/usacogate
COCI http://evaluator.hsin.hr
MENDO http://mendo.mk/Welcome.do
UVA http://acm.uva.es/contest
ACM http://acm.hit.edu.cn/index.php

1. **(Србија 2011, окружно)** Дата су три броја A, B и C. Исписати вредност израза $A - B * C$.
Улаз.
(Улазни подаци се учитавају са стандардног улаза.) У првом реду стандардног улаза налазе се природни бројеви A, B, C
($0 \leq A, B, C \leq 2^{32} - 1$).
Излаз.
(Излазне податке исписати на стандардан излаз.) У први и једини ред исписати вредност израза $A - B * C$.
Ограничења.
У 50% тест примера ће A, B и C имати вредност из интервала [0, 1000].
Пример 1.

стандардни улаз	**стандардни излаз**
1 2 3	-5

Објашњење.
1 - 2 * 3 = 1 - 6 = -5.
Пример 2.

стандардни улаз	**стандардни излаз**
0 0 0	0

Пример 3.

стандардни улаз	**стандардни излаз**
98574892 9473 9387590	-88830065178

Komentar rešenja:
http://www.yuoi.nis.edu.rs/takmicenja/2011.0.kv1/1.pomnozi/pomnozi.solution.pdf

```
#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;
}
```

2. **(Шумен 2012, комбинаторна претрага)** Дате су теразије са два таса и n тегова чије су масе a_1, a_2, …, a_n. Постављамо сваки од датих тегова на теразије, један за другим, тако да у сваком моменту леви тас није никад тежи од десног таса. У сваком кораку, бирамо један тег који још није постављен на теразије, и смештамо га или на леви тас или на десни тас. Понављамо ове кораке све док не употребимо све тегове. Напишите програм **balance**, који израчунава број начина на који можемо то урадити.

Улаз

У првом реду стандардног улаза дат је цео број n ($0 < n < 10$). У другом реду се налази n целих бројева: a_1, a_2, …, a_n ($0 < a_1, a_2, \ldots, a_n < 1000$).

Излаз

Ваш програм мора да испише један број на стандардни излаз – захтевани број начина.

Пример

Улаз

3
1 2 4

Излаз

15

РЕШЕЊЕ

Задатак може да се реши проналажењем свих могућности за постављање тегова под условима описаним у формулацији задатака. Стога, генеришемо све пермутација бројева a1, a2, ... За сваку такви пермутацију покушавамо да стави другу тежину на леви или десни тас, уз проверу да ли укупне тежине на левом тасу нису веће него са десне стране.Предложено решење је да се користити рекурзивна реализација трагања тегова.

```
#include<iostream>
#include<algorithm>
using namespace std;

int a[10], n, ans = 0, lpan, rpan;

void set(int num)
{
   if(num == n)
   {
      ans++;
      return;
   }
   if (lpan + a[num] <= rpan)
   {
      lpan += a[num];
      set(num + 1);
      lpan -= a[num];
   }
```

```
    rpan += a[num];
    set(num + 1);
    rpan -= a[num];
}

int main()
{
    cin >> n;
    for(int i=0; i<n; i++)
      cin >> a[i];
    sort(a, a+n);
    do
    {
       lpan=0; rpan=0;
       set(0);

    }while(next_permutation(a,a+n));
    cout << ans << endl;
    return 0;
}
```

Још неки тест примери

УЛАЗ	ИЗЛАЗ
3 1 2 3	17
9 501 503 504 506 502 505 507 509 508	34812189
9 1 2 3 99 98 97 32 33 34	34843575

3. **(Србија, 2000, општинско, средња школа, динамичко програмирање)** Шумадија, некада пребогата шумама, данас се више тиме не може поносити. Због заштите животне средине покушава се на сваки начин спрецити несавесна сеча дрвећа. Локални, новопечени богаташ, купио је повелико квадратно парче земље, на њему жели да изгради што је веће могућу кућу и то тако да и њена основа буде квадратног облика са странама паралелним странама плаца. Мапа имања је састављена из квадратица димензија 1x1. Једно дрво заузима тачно један такав квадратић. Пошто је имање велико, врло је тешко наћи највећи квадрат са странама паралелним странама плаца на коме нема дрвећа. Ваш задатак је да за задату мапу имања на којој се налазе позиције дрвећа, нађете квадрат чија је страна најдужа на коме нема стабала. Дужина стране је број квадратића 1x1 дуж његове стране.

Са стандардног улаза се учитава следеће: У првом реду број n (n = 200) који је дужина стране имања. Следећи ред садржи број p, број стабала на имању. Наредних p редова садрже по два броја који су координате стабала. На стандардни излаз треба исписати само један број, дужина странице највећег квадрата који не садржи ни једно дрво.

Пример

Улаз:
5
3
2 2
4 2
4 4

Излаз:

3

Координате горњег левог темена траженог квадрата су 1 3, а доњег десног 3 5.

```
program bogatas;
const MaxN=200;

var
  n:integer;
  v:array[1..MaxN,1..MaxN] of integer;

procedure Init;
var
  i,j,x,y,k:integer;

begin
  readln(n);
  for i:=2 to n do
    for j:=2 to n do     v[i,j]:=-1;

  for i:=1 to n do
    begin
      v[i,1]:=1;      v[1,i]:=1;
    end;

  readln(k);
  for i:=1 to k do
    begin
      readln(x,y);
      v[x,y]:=0;
    end;
end;

function min(a,b,c:integer):integer;
begin
  if a>b then a:=b;
  if a>c then a:=c;
  min:=a;
end;

procedure Solve;
var
  max,i,j:integer;

begin
  max:=1;
  for i:=2 to n do
    for j:=2 to n do
      if v[i,j]<>0 then
        begin
          v[i,j]:=min(v[i-1,j],v[i-1,j-1],v[i,j-1]) + 1;
          if v[i,j]>max then max:=v[i,j];
        end;
  writeln(max); readln;
end;

begin
```

```
  Init;
  Solve;
end.
```

4. (Србија 2012, квалификације, средње школе, математички)

За два дата природна броја а и б ($2 \le a \le б < 2^{64}$) одредити природне бројеве х и к тако да важи $a \le x^к \le б$, а к је што веће могуће. У случају да има више решења исписати оно у којем је х најмање.

Улаз.

У првој и јединој линији стандардног улаза се налазе природни бројеви а и б.

Излаз.

На стандардни излаз исписати редом два тражена природна броја х и к.

Пример 1.

Улаз: Излаз:

5 20 2 4

Пример 2.

Улаз: Излаз:

35 50 6 2

РЕШЕЊЕ

Дати проблем на квалификацијама био је математичке природе. Задатак није баш најуспешније решен, јер је просечан број освојених бодова свега 28 од 100. Најпре треба уочити да к може да узм само вредности из скупа {1,2,3,…,63}. Наиме, како су границе сегмента ограничене са 2^{64} , а пошто је х природан број, онда тражени степен к не може бити већи од 63.

У формулацији проблема се захтева минимизација број х за тражено к. Препоставимо да смо степен к фиксирали. Које су могуће вредности за број х? Уколико трансформишемо услов a <= x^k <=b, добијамо да важи

$$\sqrt[k]{a} \le x \le \sqrt[k]{b}$$

Како је х природан број, минимална вредност која задовољава горње неједнакости може имати два облика:

$\lfloor \sqrt[k]{a} \rfloor$ или $\lfloor \sqrt[k]{a} \rfloor + 1$

Једноставним испитивањем да ли први, односно други облик задовољава и горњу границу, налазимо тражено х за дато к. Како к може имати само 63 могућих вредности, испитивањем сваке могућности долазимо до коначног решења. Овде треба напоменути да случај к = 1 треба посебно да се издвоји. Пошто се тражи максимално к, можемо кренути од максималне могуће вреднити, односно 63, и редом испитивати за вредност х. Чим наиђемо на прво к за које постоји х које задовољава услове, програм може прекинути рад и вратити ове вредности.

```
Algoritam: Pseudo kod algoritma problema Stepen
Input: a, b
Output: x, k
for k:=63 to 2 do
    x = ⌊∛a⌋;
      if (x^k<a) x=x+1;
    if (a<=x^k<=b) return(x,k);
end
return (a,1)
```

Дата ограничења захтевају рад са 64-битним типовима података. У C-у се може користити тип *unsugned long long*, а у Pascal-у тип Qword. Оба типа узимају вредности из сегмента [0, 2^{64}-1], што је управо нама и потребно. За рачунање вредности х, односно к-тог корена броја а, користимо функције:

1. у програмском језику C
2. exp(ln(a)/k) у програмском језику Pascal (ово је заправо композиција две функције, јер користимо природни логаритам као међукорак, односно

$$\sqrt[k]{a} = a^{1/k} = e^{\ln(a^{1/k})} = e^{\frac{1}{k}\ln a}$$

Међутим овде се крије један мали проблем. Наиме, при позиву фунцкције pow која као параметар прима тип доубле може се за веће вредности улазних података изгубити на тачности.
Наиме, као у приказаном псеудо-коду проверавамо да ли је pow(x,k)<a.
Ако јесте, повећавамо x за један. Овим додатним испитивањем сигурно добијамо тражену вредност x као цео број (наравно ово не важи за реалну вредност). Велики број такмичара је због овог превида, изгубио поене на неким тест примерима.
Напомена. Друга могућност је била импелементација посебне функције за рaqунање к-тог корена. На овај начин пребацивање у реални тип не би било потребно, тако да се не би губила горе описана презицност. Наведена функција се може имплементирати уз помоћу бинарне претраге.

5. **(Шумен 2012, основна школа, графови)** У земљи Waterland, постоји n језера (нумерисаних од 1 до n) и m канала између њих. Позната је ширина сваког канала (у метрима). Кретање каналима се може извести у оба смера. Познато је да чамац ширине један метар може доспети до ма ког језера, почевши од језера са редним бројем 1.
Написати програм **channels**, који израчунава минимални број канала које треба проширити, тако да чамац ширине k метара може путовати између свака два језера (чамац се може померати од једног језера до другог, ако је његова ширина мања или једнака од ширине канала који повезује језера).
Улаз
У првој линији стандардног улаза су дати цели бројеви n и m ($1 < n \leq 1000$, $1 < m \leq 100000$).
У наредних m линија су дата три цела броја, u, j и w, који указују да постоји канал ширине w ($1 \leq w \leq 200$) између језера, i и j ($1 \leq i, j \leq n$).
У последњој линији је дат цео број k ($1 \leq k \leq 200$).
Излаз
У једином реду стандардног излаза испишите један цео број: минимални број канала који треба проширити.
Пример
Улаз

```
6 9
1 6 1
1 2 2
1 4 3
2 3 3
2 5 2
3 4 4
3 6 2
4 5 5
5 6 4
4
```

Излаз

```
2
```

1. начин

Нађимо максимално покривајуће стабло T_{max} графа језера G и повежимо га каналима. После израчунавамо колко канала из T_{max} су ужи од дате вредности K. Нека је број таквих канала q. Ако се ови канали прошире, чамац ширине K може да прође између свака два језера. Штавише, ако уклонимо све канале уже од K, граф ће се разбити на q+1 компоненти повезаности и минимални број канала који ће повезати све компоненте јесте q.

2. начин

Тражимо број компоненти повезаности графа језера G_K и канала ширине барем K.

Нека је тај број q+1. Пошто је град G повезан, постојаће q тесних канала, који ће при проширивању до ширина K и додавањем у G_K повезати компоненте.
Проналажење компоненти повезаности у G_K може да се обави неким алгоритмом за обилазак графа – BFS или DFS.

```
#include <algorithm>
#include <cstdio>
#include <map>
```

```
#include <vector>
#include <queue>
#include <time.h>
#include <limits.h>
using namespace std;

const int MaxVertex=1001;
int E[MaxVertex][MaxVertex], D[MaxVertex], Pi[MaxVertex];
bool Marked[MaxVertex];
int N,M,K;


void input()
{
  int u,v,w;
  scanf("%d %d", &N, &M);
  for(int i=1; i<=N; i++)
   for (int j=1; j<=N; j++)
    E[i][j]=0;
  for(int i=1; i<=M; i++)
  {
   scanf("%d %d %d", &u, &v, &w);
   E[u][v]=w;
   E[v][u]=w;
  }
  scanf("%d", &K);
}

void CreateMST(int s)
{
  int u,v,w,cnt;
  for(int i=1; i<=N; i++)
   { D[i]=0; Pi[i]=-1; Marked[i]=false; }
  D[s]=INT_MAX; // MaxSpaningTree !!
  cnt=1; // koliko cvorova je u T, pocetak s je u T
  while (cnt<=N) {
   w=0; u=0;
   for(int i=1; i<=N; i++)
     if ((D[i]>w) && (!Marked[i]))
       {w=D[i]; u=i;}
   Marked[u]=true;
//   if (u != s) printf("%d %d %d\n",u, Pi[u], D[u]);
   for(int v=1; v<=N; v++)
     if ((D[v]<E[u][v]) && (!Marked[v]))
       {D[v]=E[u][v]; Pi[v]=u;}
   cnt++;
  }
// uredjeni par <v,Pi(v)> je u T,  sa tezinom d[v], za v != s
}

void SolveP2()
{
  int l;
  l=0;
  for (int v=1; v<=N; v++)
   if ((Pi[v]>0) && (D[v]<K))
    l++;
```

```
  printf("%d\n",l);
}

main()
{
  input();
  CreateMST(1);
  SolveP2();
}
```

Још неки тест примери

УЛАЗ	ИЗЛАЗ
7 11 4 6 3 2 3 1 3 7 5 1 5 3 1 4 7 2 6 6 5 7 5 3 6 4 2 7 7 4 5 2 2 4 2 6	3
8 20 3 6 3 1 8 5 2 4 2 1 5 7 6 7 6 4 7 4 5 8 3 3 5 4 1 4 6 2 7 6 5 6 3 4 6 5 6 8 6 2 5 2 1 3 5 5 7 6 7 8 3 2 8 5 2 6 3 4 8 5 5	0