ЗАДАЧІ ІІІ ЕТАПУ ВСЕУКРАЇНСЬКОЇ ОЛІМПІАДИ З ІНФОРМАТИКИ В ЖИТОМИРСЬКІЙ ОБЛАСТІ У 2016 РОЦІ ТА РЕКОМЕНДАЦІЇ ЩОДО ЇХ РОЗВ'ЯЗУВАННЯ

Жуковський Сергій Станіславович,
доцент кафедри прикладної математики та інформатики Житомирського державного університету імені Івана Франка, кандидат педагогічних наук.

Матвійчук Сергій Володимирович,
учитель інформатики Ружинської гімназії.

Третій етап Всеукраїнської олімпіади в Житомирській області традиційно проходив з використанням Інтернет-порталу e-olymp.com. Пропонуємо завдання олімпіади і рекомендації щодо їх розв'язання. Задачі можна спробувати розв'язати й перевірити, використовуючи автоматичну систему перевірки. Номери відповідних задач на сайті e-olymp.com вказано біля назви.

ДЕНЬ 1

1. Олімпіада
(https://www.e-olymp.com/uk/problems/7503)

На олімпіаду з інформатики прибули **N** команд по **A[i]** учасників в кожній **(i=1..N)**. Для проведення змагань приготували класи з кількістю **M** комп'ютерів у кожному. Яку мінімальну кількість класів потрібно задіяти за умови, що в кожному класі мають бути представники різних команд.

Вхідні дані. Числа **N** і **M** у першому рядку. У другому **N** чисел **A[i] (i=1...N)**. Числові значення цілі, невід'ємні, не перевищують **100**.

Вихідні дані. Одне число — необхідна кількість класів.

Приклад введення	Приклад виведення
5 3 2 3 4 1 2	4

Розв'язання задачі

Розглянемо частинний випадок — 3 команди, у кожній команді по 3 учасники, у класах по 2 учасники.

Клас 1	Клас 2	Клас 3	Клас 4	Клас 5
Команда 1	Команда 1	Команда 1	Команда 2	Команда 2
Команда 2	Команда 3	Команда 3	Команда 3	

Рис. 1

Спробуємо розмістити учасників у 5 класів.

Якщо розміщувати так, як показано на малюнку, то для розв'язання задачі потрібно знайти загальну кількість учасників **S** (1 — тут і далі число в дужках — номер мітки біля рядка коду в програмі), поділити її на кількість комп'ютерів **M** у класі (2).

Якщо при діленні утворилась остача, то до результату додати 1 (3). Якщо кількість класів, яку ми знайшли, менша, ніж кількість учасників у найбільшій команді, то кількість класів дорівнюватиме кількості учасників у найбільшій команді (4).

```
#include<iostream>
#include<stdio.h>
using namespace std;
int main()
{
//перенаправляємо потік введення з консолі
  на введення з файлу
freopen(«input.txt»,»r»,stdin);
// перенаправляємо потік виведенення
   в консоль на виведення у файл
freopen(«output.txt»,»w»,stdout);
int n, m, a, s=0, mk = 0, k, i;
cin>>n>>m;
for(i=1; i<=n; i++)
{
 cin>>a;
 s+=a;     // (1)
 mk = max(a, mk);// знаходження команди
          з найбільшою кількістю учасників
}
k=s/m; // (2)
if ( s % m != 0) k++;              // (3)
if ( k < mk ) k=mk;    // (4)
cout << k << endl;
return 0;
}
```

2. Три прямокутники
(https://www.e-olymp.com/uk/problems/7504)

На білому аркуші паперу в клітинку намалювали три зафарбованих прямокутники так, що їхні сторони лежать на лініях сітки, а вершини мають відомі цілі координати. Знайти загальну кількість зафарбованих клітинок.

Вхідні дані. У трьох рядках по чотири цілих числа — координати двох протилежних вершин кожного прямокутника (значення по модулю не перевищують **100**).

Вихідні дані. Одне число — кількість зафарбованих клітинок.

Приклад введення	Приклад виведення
2 2 5 6 3 3 7 1 6 4 4 7	22

Розв'язання задачі

Розв'язок 1

Проаналізуємо умову задачі. Вхідні дані є цілими числами, значення яких по модулю не перевищують 100 (це означає, що дані можуть бути **додатні** та **від'ємні**). Оскільки числа цілі, то можна створити масив розміром 200×200. Мовою С++: **int mas[200][200]={0}**.

Прямокутники задані координатами вершин однієї з діагоналей. Після введення координат діагоналі прямокутника впорядкуємо їх так, щоб перша координата прямокутника відповідала лівій нижній вершині, а друга координата — правій верхній вершині (5, 6).

Після цього заповнюємо комірки пам'яті масиву, які відповідають клітинкам,ю що лежать в середині прямокутників числом 1 (рис. 2.) (9) . У зв'язку з тим, що в мові С++ нумерація масиву розпочинається з 0, то ті, хто створює програму цією мовою, значення всіх координат збільшують на 100 (7, 8).

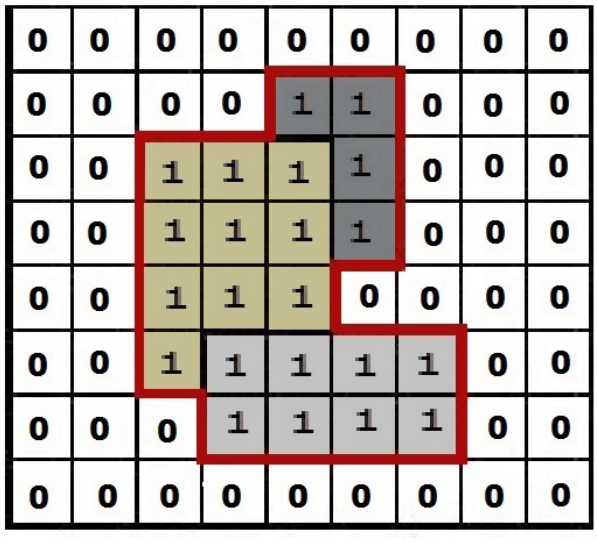

Рис. 2

Далі підрахуємо кількість одиниць в утвореному масиві (10). Це і є відповіддю до задачі.

```
#include <bits/stdc++.h>
#define N 200
using namespace std;
int main()
{
   freopen(«input.txt»,»r»,stdin);
   freopen(«output.txt»,»w»,stdout);
   int x,y,x1,y1,x2,y2,i,s=0;
   int mas[N][N]={0};
  for(i=1;i<=3;i++)
  {
     cin>>x1>>y1>>x2>>y2;
     if(x1>x2)  swap(x1,x2);  // (5)
     if(y1>y2)  swap(y1,y2);  // (6)
      x1+=N/2; y1+=N/2;      // (7)
      x2+=N/2; y2+=N/2;      // (8)
     for(x = x1; x < x2; x++)
       for(y=y1; y<y2; y++)
        mas[x][y]=1;          // (9)
  }
     for(x=0; x<N; x++)
       for(y=0; y<N; y++)
        s += mas[x][y];  // (10)
      cout<<s<<endl;
return 0;
}
```

Розв'язок 2

Аналізуючи попередній розв'язок, можна помітити, що при накладанні прямокутників можна зустріти перетин двох прямокутників, і перетин трьох прямокутників.

Знайдемо площі прямокутників, від суми віднімемо площі попарних перетинів прямокутників, а далі додамо площу прямокутника, який утворений перетином всіх трьох прямокутників (11).

Даний розв'язок буде швидко працювати для великих обмежень (у межах $2*10^9$), навіть для дійсних чисел.

```
#include <bits/stdc++.h>
using namespace std;
struct rect
{
  int x1, y1, x2, y2;
 void input()
 { // уведення координат прямокутників
    cin>>x1>>y1>>x2>>y2;
 // упорядкування координат
    if(x1>x2) swap(x1,x2);
    if(y1>y2) swap(y1,y2);
 }
  int Str()
  {
   // знаходження площі прямокутника
    int dx,dy;
    if(x1>x2) dx=0; else dx=x2-x1;
    if(y1>y2) dx=0; else dy=y2-y1;
    return dx*dy;
  }
};

void peretin(rect A, rect  B, rect &D)
{
// знаходження координат прямокутника,
//що утворений перетином  двох прямокутників
 int dx,dy;
  D.x1=max(A.x1,B.x1);
  D.y1=max(A.y1,B.y1);
  D.x2=min(A.x2,B.x2);
```

```
    D.y2=min(A.y2,B.y2);
}
int main()
{
freopen(«input.txt»,»r»,stdin);
freopen(«output.txt»,»w»,stdout);
   rect  A,B,C,AB,AC,BC,ABC;
   int Sab,Sbc,Sac,Sabc,S,Sa,Sb,Sc,Sres;
   A.input();   B.input();   C.input();
   peretin(A,B,AB);
   peretin(A,C,AC);
   peretin(B,C,BC);
   peretin(AB,AC,ABC);
   Sres=A.Str()+B.Str()+C.Str()-AB.Str()-
        AC.Str()-BC.Str()+ABC.Str(); // (11)
   cout<<Sres<<endl;
   return 0;
}
```

3. Послідовність кратних (https://www.e-olymp.com/uk/problems/7505)

У послідовності натуральних чисел $A_1, A_2, A_3,\ldots$ будь-яке з чисел A_k — найменше натуральне, яке ділиться без остачі на кожне з перших **k** натуральних чисел **1, 2, 3, ..., k**. Для заданого **N** вказати найменше **K** таке, що всі **N** чисел послідовності, починаючи з A_K — однакові.

Вхідні дані. Натуральне число **N (N<100)**.

Вихідні дані. Відповідь до задачі.

Приклад введення	Приклад виведення
2	5

Пояснення: Початок ряду 1 2 6 12 60 60 420 840 2520 …

Розв'язання задачі

Розв'язок 1

Зробимо цикл від 1, доки не знайдемо N однакових найменших спільних кратних для послідовних натуральних чисел.

Даний розв'язок дає 25% від загальної кількості балів, за умови, що ми використаємо 64 розрядний тип даних (int64 або long long).

Якщо використати мову програмування Java або python з уже реалізованим довгим типом, то можна набрати 80% від загальної кількості балів.

Розв'язок 2

Дослідимо числа нашої послідовності.

1	2	3	4	5	6	7	8	9	10
1	2	6	12	60	60	420	840	2520	2520

Можна помітити, що нове число послідовності отримаємо тоді, коли відповідне натуральне число просте, або степінь простого числа.

Для розв'язання даної задачі, спочатку знайдемо прості числа на деякому діапазоні, виконавши алгоритм «Решето Ератосфена» (12), де в масиві типу bool відмітимо всі прості числа значенням 0 (false), а не прості числа — 1 (true). Далі пройдемося по даному масиву і відмітимо всі степені простих чисел також 0 (13).

Залишилося пройтися по масиву і знайти послідовність, яка розпочинається з 0, а далі іде N–1 одиниця (14). Номер комірки пам'яті, де знаходиться 0, і є відповіддю до задачі.

Реалізація програми

```
#include<iostream>
#define N 500000
using namespace std;
bool p[N+100]={0};
int m[N]={0};
void resheto()  // (12)
{
      long long i,j;
      for(i =2; i<N; i++)
      {
      if(!p[i])
       for(j=i*i; j<N; j+=i )
         p[j]=true;
  }
      for(i=2;i<N;i++) //(13)
    if(!p[i])
    for(j=i*i;j<N;j*=i)
        p[j]=false;
}

int main()
{
      int n,i,in=1,k=1;
 cin>>n;
 if(n==1){cout<<1<<endl;  return 0;
 }
 resheto();
 for(i=2;i<N;i++){   // (14)
  if ( p[i] )  k++;
else  {k=1;in=i;}
 if(k==n)
{
 cout<<in<<endl;
 return 0;
}
 }
return 0;
}
```

4. Подорож у місті (https://www.e-olymp.com/uk/problems/7506)

У деякому місті будинки, що знаходяться по одну сторону єдиної вулиці, пронумеровані послідовними числами від **1** до **N**. Відстань між сусідніми будинками достатньо велика, тому мешканці звикли пересуватись по місту на маршрутних таксі, яких усього **M**. Поїздка по одному маршруту на будь-яку відстань коштує лише один долар, але кожне таксі має зупинятись тільки біля строго визначених (але не менше двох) будинків. На зупинки вказує номер маршруту — **A B (A** — найменший номер будинку, де зупиняється таксі, **B** — період зупинок).

Наприклад, маршрутне таксі з номером **2 3** при **N=11** зупиняється так: **2 5 8 11 8 5 2 ...**, тому деякі будинки можуть бути незадіяними.

Знаючи значення **N** і **M** та номери всіх маршрутів, можно знайти:

- кількість **K** будинків, де не зупиняється жодне таксі;
- скільки найменше доларів потрібно витратити, щоб дістатися з першого до **N**-го будинку або вивести **0**, якщо це неможливо.

Числові значення **N, M, A** і **B** — натуральні, 1 J **N** J 200, 1 J **A,B,M** J 20.

Приклад введення	Приклад виведення
11 2 2 3 1 4	5 2

Пояснення: На таксі **1 4** рухаємось до будинку **5**, потім на таксі **2 3** дістаємося до будинку **11**.

Розв'язання задачі

Для знаходження будинків, біля яких не зупиняється маршрутне таксі, створимо масив color, заповнимо його нулями, і під час введення даних можна пройтися циклом від найменшого номеру будинку A, з кроком B в масив color занесемо значення 1 (16).

Комірки пам'яті, у яких залишаться нулі і будуть відповідати будинкам, біля яких не зупиняється маршрутне таксі.

Для знаходження найменшої вартості проїзду, потрібно створити матрицю суміжності (масив розміром N*N, де 1 позначимо переїзд від будинку до будинку за допомогою одного маршрутного таксі (17), Попередньо в масив занесемо велике число, наприклад 10000000 (15). Після чого виконаємо алгоритм Флойда-Уоршела (18) (пошук найкоротшого шляху на графі)

```
 #include <bits/stdc++.h>
#define oo 10000000
using namespace std;
int main()
{
int a,b,n,m,j1,j2,i,k,j;
int d[205][205]={0},color[205]={0};
freopen(«input.txt»,»r»,stdin);
freopen(«output.txt»,»w»,stdout);
cin>>n>>m;
for(j1=1;j1<=n;j1++)
      for(j2=1; j2<=n; j2++)
 if(j1!=j2) d[j1][j2]=oo; // (15)
for(i=1; i<=m; i++)
{
      cin>>a>>b;
      for(j1=a; j1<=n; j1+=b)
      {
            color[j1]=1; // (16)
       for(j2=a; j2<=n; j2+=b)
       if(j1!=j2) // (17)
{ d[j1][j2]=1; d[j2][j1] =1; }
    }
}
for(k=1; k<=n; k++) // (18)
 for(i=1; i<=n; i++)
  for(j=1; j<=n; j++)
  d[i][j]= min(d[i][k] + d[k][j], d[i][j]);
int kil=0,res;
for(i=1;i<=n;i++)
 if(color[i]==0) kil++;
 if(d[1][n]==oo) res=0 ; else res= d[1][n];
 cout<<kil<<' '<<res<<endl;
return 0;
}
```

ДЕНЬ 2

1. Добуток
(https://www.e-olymp.com/uk/problems/7507)

Маємо **N** цілих чисел. Який найбільший добуток можна отримати, використавши тільки три з цих чисел?

Вхідні дані: У першому рядку ціле невід'ємне число **N. 3≤N≤10^5.** У другому рядку **N** цілих чисел, кожне по модулю не перевищує **10^5.**

Вихідні дані: Значення найбільшого добутку трьох з них.

Приклад введення	Приклад виведення
9 3 5 -9 7 4 0 9 -3 5	315

Розв'язання задачі

Відсортуємо масив чисел (19). Результатом буде максимальне значення серед добутку трьох найбільших чисел, або добутку двох найменших чисел (якщо вони від'ємні) на найбільше число (20).

Щоб не сортувати, можна знайти три найбільші і два найменші числа, після чого знайти max

(max1*max2*max3, min1*min2*max3),

де max1<=max2<=max3. Можна зберігати тільки 5 комірок max1,max2,max3,min1,min2,max3 під час зчитування даних.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
freopen(«input.txt»,»r»,stdin);
freopen(«output.txt»,»w»,stdout);
      long long a[100005],i,n,res;
      cin>>n;
      for(i=0;i<n;i++)
      cin>>a[i];
      sort(a,a+n); // (19)
      res=max(a[0]*a[1]*a[n-1],  a[n-1]*a[n-2]*a[n-
3]); // (20)
      cout<<res<<endl;
      return 0;
}
```

2. Номер тижня
(https://www.e-olymp.com/uk/problems/7508)

Виробники комп'ютерної пам'яті записують датою виготовлення DDR-модуля як номер тижня року і року, тому день **5.03.2016** буде записаний як 9 тиждень 2016 року. Відповідно до стандарту ISO першим вважається тиждень, який містить найперший четвер (або 4 січня) року, тому декілька початкових днів можуть залишитись в останньому тижні попереднього року. Далі нумерація відбувається звичайним способом, отже, кожен рік має 52 або 53

тижні. Потрібно враховувати, що крайні дні року інколи потрапляють до першого тижня наступного року. Новорічний тиждень відноситься до того року, куди попадає його четвер.

Для даної дати потрібно вивести номер тижня і рік, відповідно.

Вхідні дані: У єдиному рядку календарна дата у форматі **D M Y** (день місяць рік). **1≤D≤31**, **1≤M≤12**, **2001≤Y≤2020**.

Вихідні дані: Номер тижня і рік для даної дати.

Приклад введення	Приклад виведення
5 3 2016	9 2016

Розв'язання задачі

Згенеруємо масив, номер першого тижня для кожного року з діапазону умови задачі із 2001 по 2021 роки (21), врахувавши високосний рік, а потім для даної дати, врахувавши перший тиждень року розрахувати номер тижня (22).

```
Var d,m,y,i,j,p,q,w : longint;
                a,b : longint;
                 f1,f2:text;
          x : array [1..7777,1..2] of integer;
{перевірка року на високосність}
Function VVV (y :integer) : integer;
begin
  if (y mod 4 =0) and (y mod 100 <>0) or (y mod 400=0)
 then VVV:=1 else VVV:=0;
end;
{кількість днів у місяці}
Function MMM (m,y : longint) : longint;
var w : longint;
begin
  w:=31;
  if m=2 then w:=28+VVV(y);
  if (m=4) or (m=6) or (m=9) or (m=11) then w:=30;
  MMM:=w;
end;
Function Count (d,m,y : longint): longint;
var  x,k,s : longint;
begin
  s:=0;
  for x:=1900 to y-1 do s:=s+365+VVV(x);
  for x:=1 to m-1 do s:=s+MMM(x,y);
  Count:=s+d;
end;

begin
  assign(f1,'input.txt');reset(f1);
  assign(f2,'output.txt');rewrite(f2);
   Readln(f1,d,m,y);
   j:=Count(31,12,2000);
   q:=2001;
   w:=1;
    for i:=Count(1,1,2001) to Count(1,1,2021) do
begin {    (21)  }
     x[i-j,1]:=w;
     x[i-j,2]:=q;
     if i mod 7=0 then begin
       if Count(31,12,q)<i+4 then begin
         w:=1; q:=q+1;
       end
       else w:=w+1;
     end;
   end;
   p:=Count(d,m,y); { (22) }
   writeln(f2,x[p-j,1],' ',x[p-j,2]);
close(f1);  close(f2);
end.
```

3. Одиниці і сімки

(https://www.e-olymp.com/uk/problems/7508)

Знайти **N**-й член зростаючої послідовності натуральних чисел:

1 7 11 17 71 77 111 117 171 177 711 717..

Вхідні дані: Натуральне число **N. 1≤N≤10000.**

Вихідні дані: Відповідь до задачі

Приклад введення	Приклад виведення
7	111

Розв'язання задачі

Запишемо перші натуральні числа у 2-ковій системі числення.

0, 1, 10,11, 100, 101, 110, 111,

Подивившись на послідовність, що дано в умові задачі, можна помітити, що на першому місці знаходиться як і 1, так і 7 як значущі цифри, а в 2-ковій системі на першому місці 0 — не є значущим.

Подивимося на послідовність чисел двійкової системи числення, починаючи з числа 2, і відкинемо першу значущу цифру. Можна помітити, що числа відповідають нашій послідовності, замінивши 0 на 1, а 1 на 7, не враховуючи першу значущу цифру.

Отже, для розв'язання задачі нам потрібно до введеного числа додати 1, перевести його в 2-кову систему числення (не враховуючи першу цифру) і замінити 0 на 1, а 1 на 7.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
bitset <32> a;
int t;
freopen(«input.txt»,»r»,stdin);
freopen(«output.txt»,»w»,stdout);
cin>>t;
a=t+1;
string s=a.to_string(); // переведення числа в двійкову систему числення
while(s[0]=='0') s.erase(0,1); // виділимо ведучі (незначущі) нулі у двійковому записі числа
  s.erase(0,1); // видалення першої одиниці двійкового запису числа
 for(int i=0; i<s.length();i++) // заміна цифри 0 на 1 і 1 на 7.
 if (s[i]=='0') s[i]='1'; else s[i]='7';
 cout<<s<<endl;
 return 0;
}
```

4. Арифметичний ребус (https://www.e-olymp.com/uk/problems/7510)

Арифметичний ребус — це запис вигляду **M&N=P**, де **M,N,P** — послідовності з цифр **0,1,2,..,9** та/або латинських букв **a, b, c, .., j**, які позначають невідомі цифри, а & — знак однієї з операцій **+** або *****.

Для розв'язання ребусу потрібно кожну букву замінити цифрою, щоб отримати правильну рівність, за такими правилами:

- однакові букви в усьому ребусі замінюються однією цифрою, якої немає серед відомих цифр;
- перша цифра у числах **M, N, P** — не нуль, а кожне з чисел **M, N, P** не перевищує **999999**.

Вхідні дані: У єдиному рядку записано один арифметичний ребус.

Вихідні дані: Будь-який розв'язок ребусу, або **0** (нуль), якщо розв'язку немає.

Приклад введення	Приклад виведення
ab*ab=6ab	25*25=625

Розв'язання задачі

Визначимо кількість різних невідомих цифр, після чого зробимо всі можливі перестановки цифр, підставивши їх у математичний вираз і знайдемо той, який відповідає умові задачі.

```
#include<bits/stdc++.h>
using namespace std;
int p[10]={0},z[10]={0},nn,a,b,c,kil=0,nz,nd;
char ss[100];
char s[100]=»»,s1[100], zn,d;
bool res=true;
void pp(int n) // перевірка певної підстановки
                    цифр
//замість букв на правильність виразу
{
      strcpy(s1,s);
   for(int i=0;i<n;i++)
     for(int j=0;j<nn;j++)
     {
       if(s1[j]==ss[i]) s1[j]=(z[i]+'0');
     }
     if(s1[0]=='0'|| s1[nz+1]=='0'||s1[nd+1]=='0')
       return;
     for(int j=0;j<nn;j++)
     if(s1[j]>='a'&&s1[j]<='j') return;
sscanf(s1,»%d%c%d%c%d»,&a,&zn,&b,&d,&c);
     if(zn=='+')
        {if(a+b==c) {
      cout<<a<<»+»<<b<<»=»<<c<<endl;
      res=false;
      exit(0);
      }
      else
       {if(a*b==c) {
cout<<a<<»*»<<b<<»=»<<c<<endl;
       res=false;
       exit(0);
      }
   }
}

void perebor(int k,int n) // генерація всіх
  можливих підстановок
//цифр замість букв з виразу
{
      if(k==n) { pp(n); kil++;}
      for(int i=0;i<10;i++)
      {if(p[i]==0)
               {
               z[k]=i;p[i]=1;
               perebor(k+1,n);
               p[i]=0;
               }
      }
}

int main()
{
freopen(«input.txt»,»r»,stdin);
freopen(«output.txt»,»w»,stdout);
cin>>s;
 int i,n=0;
 nn=strlen(s);
 for(i=0;i<nn;i++)// визначення букв у виразі
  та математичної операції
 {
  if(s[i]>='a' &&s[i]<='j') { ss[n]=s[i];  n++; }
  if(s[i]=='+') nz=i;
  if(s[i]=='*') zn=i;
  if(s[i]=='=') nd=i;
  if(s[i]>='0'&& s[i]<='9') p[s[i]-'0']=1;
 }
   ss[n]='\0';
  sort(ss,ss+n);
  int j=0;
 for(i=1;i<n;i++)
 if(ss[j]!=ss[i]) {j++;ss[j]=ss[i];}
 ss[j+1]='\0';
 n=j+1;
 perebor(0,n);
 if(res) cout<<0<<endl;
 return 0;
}
```