Управління освіти і науки
Чернігівської обласної державної адміністрації

Чернігівський обласний інститут післядипломної
педагогічної освіти імені К.Д. Ушинського

ЗБІРНИК ЗАДАЧ ТА РОЗВ'ЯЗКІВ ІІІ ЕТАПУ ВСЕУКРАЇНСЬКОЇ УЧНІВСЬКОЇ ОЛІМПІАДИ З ІНФОРМАТИКИ

2012-2013 НАВЧАЛЬНОГО РОКУ

Чернігів – 2015

Збірник задач та розв'язків ІІІ етапу Всеукраїнських учнівських олімпіад з інформатики 2012-2013 навчального року / укл. О.Є. Баранова, Ю.М. Літош, В.В. Зуб, С.М. Бондаренко, О.М. Смірнова. – Чернігів: ЧОІППО імені К.Д. Ушинського, 2015. – 50 с.

Укладачі:

Баранова О.Є., методист відділу інформатики, інформаційних технологій та дистанційного навчання Чернігівського обласного інституту післядипломної педагогічної освіти імені К.Д. Ушинського

Літош Ю.М., завідувач відділу інформатики, інформаційних технологій та дистанційного навчання Чернігівського обласного інституту післядипломної педагогічної освіти імені К.Д. Ушинського

Зуб В.В., директор Прилуцької загальноосвітньої школи І-ІІІ ступенів № 7 Прилуцької міської ради Чернігівської області, вчитель-методист

Бондаренко С.М, учитель математики та інформатики Прилуцької загальноосвітньої школи І-ІІІ ступенів № 7 Прилуцької міської ради Чернігівської області, учитель-методист

Смірнова О.М., методист відділу інформатики, інформаційних технологій та дистанційного навчання Чернігівського обласного інституту післядипломної педагогічної освіти імені К.Д. Ушинського

Рецензенти:

Горошко Ю.В., завідувач кафедри інформатики та обчислювальної техніки Чернігівського національного педагогічного університету імені Т.Г. Шевченка, доктор педагогічних наук

Покришень Д.А., завідувач кафедри інформатики та інформаційно-комунікаційних технологій в освіті Чернігівського обласного інституту післядипломної педагогічної освіти імені К.Д. Ушинського, кандидат педагогічних наук, доцент

Рекомендовано до друку
вченою радою Чернігівського обласного інституту післядипломної педагогічної освіти імені К.Д. Ушинського (протокол № 6 від 17.09.2015 р.)

ЗМІСТ

ВСТУП

Даний посібник містить задачі та розв'язки III етапу Всеукраїнської учнівської олімпіади з інформатики, що проводилась у Чернігівській області у 2012-2013 навчальному році.

III етап олімпіади проводився синхронно з іншими областями з використанням центрального серверу прийому і автоматизованої перевірки розв'язків. У збірнику викладені алгоритми розв'язання задач, реалізовані мовами програмування Pascal і C++.

Алгоритми розв'язання задач, що представлені в збірнику, запропоновані вчителями та учнями загальноосвітніх навчальних закладів області, які мають позитивний досвід участі в олімпіадному русі. Програми-розв'язки перевірялися на множині тестів, підготовлених журі олімпіади. Переважна більшість наведених розв'язків є коректними, але якщо програма не виконує всі тести, запропоновані журі, то в поясненні розв'язку це вказується.

Матеріали допоможуть учителю і учню системно організувати підготовку до олімпіад з інформатики, підвищити свій рівень знань із програмування. Постійна практика розв'язання задач із програмування формує в учнів навички класифікувати задачі, бачити спільні та відмінні елементи, вносити можливі покращення в існуючі алгоритми розв'язання.

Збірник розрахований на вчителів та учнів, які готуються до участі в інтелектуальних змаганнях з інформатики, а також додатково займаються програмуванням.

Посібник націлений на розвиток здібностей відповідної категорії учнів – підлітків із розвиненим логічним мисленням, підготовленим математичним апаратом, зацікавлених у формуванні себе як майбутніх програмістів. Саме такі учні, як правило, беруть участь в олімпіаді з інформатики.

8-11 клас
I тур

A – Неуважність

Ім'я файлу, який містить вхідні дані: text.in

Ім'я вихідного файлу: text.out

Обмеження часу: 100 мс

Обмеження пам'яті: 128 M

Степан вдало пройшов співбесіду і ось уже як чотири місяці працює на одній із найпрестижніших IT-компаній. Прийшов час здавати проект менеджеру і Степан, як істинний студент, усе виконує в останню ніч перед здачею. Набирає текст Степан звичайно дуже швидко, але неуважно. От і цього разу останню частину тексту він набрав, не звернувши уваги, що випадково натиснув клавішу Caps Lock. Таким чином, великі букви були набрані маленькими, а маленькі – великими. Інші символи він набрав вірно. Степан настільки стомився, що немав сил виправити помилки, і він вирішив кілька годин поспати. Допоможіть Степану, доки він спить. Напишіть програму, яка виправляє неуважно набраний текст.

Формат вхідних даних: перший рядок вхідного файлу містить неуважно набраний Степаном текст, який містить не більше 500 символів.

Формат вихідних даних: вихідний файл має містити виправлений текст.

Приклади

Вхідні дані розміщені у файлі text.in	Результат роботи знаходиться у файлі text.out
sCHOOL	School

Ідея розв'язку задачі Хорошка С.В., учителя вищої категорії Коропської ЗОШ І-ІІІ ст. імені Т.Г. Шевченка

Посимвольно зчитати дані до кінця рядка у файлі і, перевіривши належність зчитаного символу до великих або до малих літер латинського алфавіту, виконати відповідний перевід символів. Для переведення малих букв у великі можна скористатися стандартною функцією **UpCase**. Для переведення з великих у малі можна використати особливість системи **ASCII**. У цій системі кодування текстової інформації всі символи стоять під визначеними номерами. Якщо знайти різницю в коді малої літери «а» від великої та, додавши до неї значення коду поточної великої літери, знайдемо код шуканої малої літери **ord(x) +ord('a')-ord('A')**. Використавши обернену функцію **chr,** знайдемо шукану велику букву.

Розв'язок

```
program A;
      var   x:char;
begin
     assign(input,'text.in');
     reset(input);
     assign (output,'text.out');
     rewrite(output);
     while not eoln(input) do
      begin
     read(x);
      if x in ['a'..'z'] then x:=UpCase(x) else
       if   x in ['A'..'Z'] then  x:=chr(ord(x) +ord('a')-ord('A'));
   write(x);
 end;
    writeln
end.
```

Ідея розв'язку задачі Зуба В.В., директора ЗОШ І-ІІІ ст. № 7 м. Прилук, учителя математики, учителя-методиста; Бондаренка С.М., учителя математики та інформатики ЗОШ І-ІІІ ст. № 7 м. Прилук, учителя-методиста

Завдання на роботу з текстовими величинами. Один із можливих варіантів роботи полягає у символьному зчитуванні тексту і зміні регістру символів із наступним записом у текстовий файл.

Розв'язок

```
  var t:Char; f1,f2:text;
Begin
 Assign(f1,'text.in');Reset(f1);
 Assign(f2,'text.out');Rewrite(f2);
Read(f1,t); {читання першого символу тексту}
 While Ord(t)<>13 do {перевірка кінця рядка}
  Begin
   If (Ord(t)>=65) and (Ord(t)<=90)
Then Write(f2,Chr(Ord(t)+32)); {якщо символ верхнього регістру, то перевести його у нижній регістр}
   If (Ord(t)>=97) and (Ord(t)<=122)
Then Write(f2,Chr(Ord(t)-32)); {якщо символ нижнього регістру, то перевести його у верхній регістр}
   If (Ord(t)<65)or(Ord(t)>122)or((Ord(t)>90)and(Ord(t)<97))
Then Write(f2,t); {якщо символ не є літерою латинського алфавіту, то його не змінюємо}
   Read(f1,t); {читання наступного символу тексту}
  End;
 Close(f1);Close(f2);
End.
```

B – День святого Валентина

Ім'я файлу, який містить вхідні дані: holy.in

Ім'я вихідного файлу: holy.out

Обмеження часу: 500 мс

Обмеження пам'яті: 128 M

Скоро день святого Валентина і, Степану як великому прихильнику даного свята, доручили вибрати кульки для прикраси зали. Профорг університету, де навчається Степан, веде строгий перелік усіх кульок, згідно якому в наявності є N однокольорових (що поробиш – бідні студенти) кульок, діаметр i-ї кульки ($1 \le i \le N$) дорівнює Di міліметрів. Згідно новим вимогам профкому, залу необхідно прикрасити не менше ніж K кульками. Оскільки профоргу університету не подобається свято закоханих, то вона ввела своє поняття – так званий показник некрасивості – рівний максимально можливому числу Di – Dj при $1 \le i, j \le M$, де M – кількість кульок для зали, а Di – їх діаметр. Допоможіть Степану із N іграшок вибрати M ($M \ge K$) так, щоб для вибраних M кульок показник некрасивості був мінімальним.

Формат вхідних даних: перший рядок вхідного файлу містить два натуральних числа N ($2 \le N \le 100\,000$) і K ($2 \le K \le N$) відповідно. Другий рядок містить N цілих чисел Di ($1 \le Di \le 10^9$) – діаметр i-ї кульки.

Формат вихідних даних: вихідний файл має містити значення показника некрасивості, вибраних M кульок.

Пояснення: Приклад 1. Існує кілька різних варіантів вибору. Степан може вибрати, наприклад, 6 кульок: 3, 5, 6, 4, 7 і 8 Приклад 2. Степан вибере 4 кульки: 1, 5, 3 і 6.

Приклади

Вхідні дані розміщені у файлі holy.in	Результат роботи знаходиться у файлі holy.out
8 5 10 20 10 20 10 10 20 20	10

6 4 21 12 17 28 16 21	5

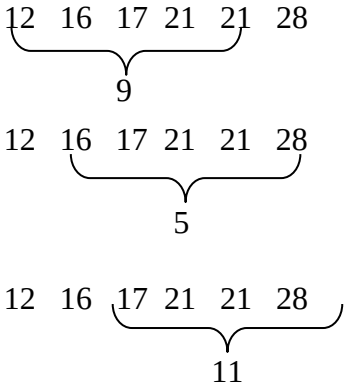
Ідея розв'язку задачі Хорошка С.В., учителя вищої категорії Коропської ЗОШ І-ІІІ ст. імені Т.Г. Шевченка

Потрібно відсортувати масив із N цілих чисел (можна використати швидке сортування).

Масив після сортування:

12 16 17 21 21 28

Потім потрібно знайти показники некрасивості для кожних К кульок (різниця між останньої і першою кулькою з К кульок)



Серед знайдених показників некрасивості знайти найменший.

Розв'язок

```
program b;
var x:array[1..1000000] of integer;
   i:integer;
   n,k,min:longint;
//швидке сортування
procedure sort(l,r:integer); {l лівий кінець масиву,r-правий кінець}
var
```

```
  i,j,x1,y1,m: integer;
begin
  i:=l;
  j:=r;
  m:=round ((l+r)/2);{середній елемент}
  x1:=x[m];
  repeat
    while x[i]<x1 do inc(i);{поки лівий менше середнього, зміщуємо лівий край вправо }
    while x[j]>x1 do dec(j);{поки правий більше середнього, зміщуємо правий край вліво}
    if i<=j then {якщо лівий і правий  перетнулися}
     begin
      y1:=x[i];
      x[i]:=x[j];{міняємо лівий і правий}
      x[j]:=y1;
      inc(i); {лівий вправо}
      dec(j); {правий вліво}
     end;
  until i>j;{кінець однієї перестановки}
  if l<j then sort(l,j);{рекурсивно сортуємо}
  if i<r then sort(i,r);{або ліву або праву частини}
end;

begin
      assign(input, 'holy.in');
      reset(input);
      assign(output,'holy.out');
      rewrite(output);
      readln(n, k);
for i:=1 to n do{зчитує масив}
  begin
   read(x[i]);
  end;
       sort(1,n); {виклик процедури швидкого сортування}
```

```
    min:= x[k]-x[1]; {знаходження мінімального показника
    некрасивості}
    for i:=1 to n-k+1  do
        if min>x[i+k-1]-x[i] then min:=x[i+k-1]-x[i];
    write(min); {вивід результату}
end.
```

Програма правильно виконує перші 10 тестів із 99, запропонованих журі.

Ідея розв'язку задачі Зуба В.В., директора ЗОШ І-ІІІ ст. № 7 м. Прилук, учителя математики, учителя-методиста; Бондаренка С.М., учителя математики та інформатики ЗОШ І-ІІІ ст. № 7 м. Прилук, учителя-методиста

Спочатку необхідно відсортувати масив по зростанню. Потім брати послідовно *K* кульок і шукати різницю діаметрів найменшої (*i*-ї) та найбільшої (*k*-ї). Серед отриманих різниць вибрати найменшу.

Розв'язок:

```
type massiv = array[1..100000] of longint;
var n,k,i,j,min:longint; t:massiv; f1,f2:text;

Procedure QuickSort(var a: massiv; l, r: longint);
 var m, x, y: longint;
Begin
 m := ((l + r) div 2);
 i := l; j := r; x := a[m];
 while i <= j do
 begin
   While a[i] < x do inc(i);
   While a[j] > x do dec(j);
```

```
   If i <= j Then
   Begin
     y:=a[i];a[i]:=a[j];a[j]:=y;
     inc(i);
     dec(j);
   End;
  End;
  If l < j Then QuickSort(a, l, j);
  If r > i Then QuickSort(a, i, r);
 End;

Begin
 Assign (f1,'holy.in');Reset(f1);
 Assign (f2,'holy.out');Rewrite(f2);
 Readln (f1,n,k);
 For i:=1 to n do Read(f1,t[i]);{введення даних у масив}
 QuickSort (t, 1, n); {сортування масиву}
 min:=t[k]-t[1]; {початкове мінімальне значення}
 For i:=1 to n-k do {пошук меншої різниці}
   If t[i+k]-t[i+1]<min Then min:=t[i+k]-t[i+1];
 Write(f2,min);
 Close(f1);Close(f2);
End.
```

C- Степан і Пари

Ім'я файлу, який містить вхідні дані:	pair.in
Ім'я вихідного файлу:	pair.out
Обмеження часу:	1 с
Обмеження пам'яті:	128 M

Останнім часом Степан дуже цікавиться парами чисел. А крім пар чисел, його цікавить найбільший спільний дільник пари чисел, позначимо його як НСД(x, y). Зараз у Степана є ціле число n і його цікавить така інформація: скільки існує пар цілих чисел (i, j), таких що $1 \le i, j \le n$ і виконується рівність i = НСД(i, j). Допоможіть йому у вирішенні нелегкої задачі.

Формат вхідних даних: у першому рядку дано ціле число n
($1 \le n \le 10^6$).

Формат вихідних даних: єдиний рядок має містити відповідь на задачу.

Зауваження: У першому прикладі підходящою парою є пара (1, 1), так як НСД(1, 1) = 1.У другому прикладі підходять 8 пар чисел:
(1, 1), (1, 2), (1, 3), (1, 4), (2, 2), (2, 4), (3, 3), (4, 4).

Приклади

Вхідні дані розміщені у файлі pair.in	**Результат роботи знаходиться у файлі pair.out**
1	1
4	8
10	27

Ідея розв'язку задачі Савченка Анатолія, учня 10 класу Корюківської ЗОШ І-ІІІ ст. № 1

Мова програмування C++, компілятор g++

Розв'язок

```
#include <iostream>
#include <algorithm>
using namespace std;
int main() {
	int n;
	ifstream cin("pair.in");
```

```
	ofstream cout("pair.out");
	cin >> n;
	unsigned int counter = 0;
	n++; // Чтобы каждую итерацию не выполнять n + 1 в условии циклов
	for (int i = 1; i < n; i++) {
		for (int j = i; j < n; j += i) {
			if ( i == __gcd(i, j) ) { // __gcd из algorithm
				counter++;
			}
		}
	}
	cout << counter << endl;
	return 0;
}
```

Ідея розв'язку задачі Таішева Фердинанда, учня 11 класу Радянськослобідської ЗОШ І-ІІІ ст. Чернігівського району (Трейтяка О.В., учитель інформатики та трудового навчання вищої категорії)

Програма на мові Pascal. Використовувався компілятор FreePascal. Створена для пошуку пар чисел, які задовольняють умову задачі.

Розв'язок:

```
var
i,n,j, count: longint; f1,f2:text;
begin
assign (f1,'pair.in');
reset (f1);
readln (f1,n);
close (f1);
```

```
count:= 0;
for i := 1 to n do begin
for j:=1 to n do begin
if n mod j = 0 then inc(count)
end;
n:=n-1;
end;
assign(f2,'pair.out');
rewrite(f2);
writeln(f2,count);
close(f2);
end.
```

Програма правильно виконує перші 10 тестів, а з 11 до 50 - обмеження за часом.

Ідея розв'язку задачі Зуба В.В., директора ЗОШ І-ІІІ ст. № 7 м. Прилук, учителя математики, учителя-методиста; Бондаренка С.М., учителя математики та інформатики ЗОШ І-ІІІ ст. № 7 м. Прилук, учителя-методиста

Розглянемо всі числа i та j ($1 \le i \le n$, $i \le j \le n$), щоб $НСД(i, j) = i$. Перевіряти всі числа i, починаючи з числа 2-го й обраховувати, скільки цілих чисел, не менших за n, діляться націло на i.

Розв'язок

```
var n,i,min:longint; f1,f2:text;
Begin
 Assign(f1,'pair.in'); Reset(f1);
 Assign(f2,'pair.out'); Rewrite(f2);
 Readln(f1,n);
```

```
 min:=n; {оскільки 1 є НСД для кожного з n чисел, що дані в умові задачі, то  min=n}
 For i:=2 to n do min:=min + n div i; {до отриманого значення min додаємо число дільників числа n на і}
 Write(f2,min);
 Close(f1);Close(f2);
End.
```

D – Спадок Степана

Ім'я файлу, який містить вхідні дані: legacy.in

Ім'я вихідного файлу: legacy.out

Обмеження часу: 2 с

Обмеження пам'яті: 128 M

Степан отримав у спадок від дідуся стоянку із N місць, пронумерованих від 1 до N. Стоянка розбита на дві частини. Перші M місць знаходяться з лівого боку, а інші N – M місць з правого. Кожного дня N жителів цього району паркуються на стоянці Степана. Відомо, що перший житель приходить раніше всіх, потім другий, і так далі, тобто k-й приходить k-м. Також для кожного жителя відомо, скільки він буде платити, якщо його машину поставлять на j-е місце. Степан придбав розподільник місць, який кожному автомобілю, що приїздить вказує, на який бік паркуватися, після чого автомобіль паркується на мінімальне за номером вільне місце відповідного боку. При цьому Степан вирішив зекономити і не придбав програмне забезпечення для розподільника, тому він працює не оптимально. Степан просить вас написати програму для цього розподільника, яка максимізує доходи Степана.

Формат вхідних даних: у першому рядку записані два цілих числа N ($2 \le N \le 1000$) і M ($1 \le M < N$) – загальна кількість місць на стоянці і кількість місць із лівого боку відповідно. У

кожному із наступних N рядків записано по N цілих додатних чисел. j-е число і-го рядка означає, скільки буде платити і-ий житель за місце з номером j на цій стоянці. Кожне з цих чисел не перевищує 10^6.

Формат вихідних даних: єдиний рядок має містити одне число – максимальний прибуток стоянки.

Зауваження: Не менш чим у 50% тестів N ≤ 30.

Приклади

Вхідні дані розміщені у файлі legacy.in	**Результат роботи знаходиться у файлі legacy.out**
2 1 3 2 6 4	8
4 1 4 3 1 1 3 1 1 1 1 1 4 1 1 1 1 2	12

Ідея розв'язку задачі Зуба В.В., директора ЗОШ І-ІІІ ст. № 7 м. Прилуки, учителя математики, учителя-методиста

Розв'язок

D – Масив вартості стоянки для і-го жителя. Усього 10 стоянок, із них 2 ліві.

	Л1	**Л2**	**П1**	**П2**	**П3**	**П4**	**П5**	**П6**	**П7**	**П8**
1-й	19	5	5	1	9	2	4	15	7	1
2-й	5	2	3	6	3	19	4	6	6	5
3-й	1	5	5	10	6	9	5	10	9	3
4-й	7	10	7	5	9	10	9	3	1	9
5-й	9	15	6	2	6	5	19	5	9	2

6-й	6	2	10	8	8	3	3	10	3	1
7-й	8	10	3	19	10	3	1	8	2	5
8-й	1	1	3	9	10	10	5	10	4	10
9-й	2	10	10	5	2	5	9	8	5	6
10-й	3	1	6	8	1	9	8	6	8	2

B – Масив накопичувальної вартості

	Ліва		Права стоянка								Дохід Степана після того, як
	Л1	**Л2**	П1	П2	П3	П4	П5	П6	П7	П8	
Блок 1	19		5								1 житель поставив машину на Л1 або П1
	10	21	22	11							2 житель поставив машину
	12	27	26	32	17						3 житель поставив машину
Блок 2	24	42		32	41	27					4 житель поставив машину
	36	56			48	46	46				5 житель поставив машину
	52	48				59	49	56			6 житель поставив машину
	64	62					60	60	58		7 житель поставив машину
	59	65						72	68	68	8 житель поставив машину
	70	78							77	74	9 житель поставив машину
Блок 3		**75**								**80**	10 житель поставив машину

```
type massiv = array[1..1000,1..1000] of longint;
 var n,m,k,i,j,x,y,z,t:longint; f1,f2:text; max:int64;
    b,d:massiv;
Begin
 Assign(f1,'legacy.in');Reset(f1);
 Assign(f2,'legacy.out');Rewrite(f2);
{Введення даних із файла. Ліва стоянка завжди має менше місць ніж права}
 ReadLn(f1,n,m);
 If m>n-m Then x:=n-m Else x:=m; {x – кількість місць зліва}
 For i:=1 to n do For j:=1 to n do
 Begin
```

```
  If x=m Then Read(f1,d[i,j]) {d – масив вартості стоянки для
і-го жителя}
        Else
        Begin
         If j<=m Then Read(f1,d[i,j+x]) Else Read(f1,d[i,j+x-n]);
        End;
 End;
 m:=x;
{Обробка даних. Пошук результату. Заповнення масиву B.}
b[1,1]:=d[1,1]; b[1,m+1]:=d[1,m+1];
 For i:=2 to n do
 Begin
{Блок 1, номер машини менший або рівний числу лівих
стоянок +1}
If (i<=m+1) and (m<=n-m) Then
  Begin
   k:=i;
   For j:=1 to k do
   Begin
    If j=1 ThenBegin b[i,k]:=d[i,k]+b[i-1,k-1];
b[i,m+1]:=d[i,m+1]+b[i-1,k-1]; End;
    If j=k ThenBegin b[i,1]:=d[i,1]+b[i-1,m+k-1];
b[i,m+k]:=d[i,m+k]+b[i-1,m+k-1]; End;
    If (j<>1) and (j<>k) Then
     Begin
      If b[k-1,j-1]>=b[k-1,m+k-j] Then x:=b[k-1,j-1] Else x:=b[k-
1,m+k-j];
      b[k,j]:=d[k,j]+x; b[k,m+k-j+1]:=d[k,m+k-j+1]+x;
     End;
End;
End;
```

{Блок 2, номер машини більший числа лівих стоянок + 1, але менший або рівний числу правих стоянок + 1}

```
If (i>m+1) and (i<=n-m+1) and (m<=n-m) Then
  Begin
   k:=i;
   For j:=1 to k do
   Begin
    If j=k ThenBegin b[i,1]:=d[i,1]+b[i-1,m+k-1];
b[i,m+k]:=d[i,m+k]+b[i-1,m+k-1]; End;
    If (j<>1) and (j<>k) Then
     Begin
      If b[k-1,j-1]>=b[k-1,m+k-j] Then x:=b[k-1,j-1] Else x:=b[k-
1,m+k-j];
      If j>m Then b[k,m+k-j+1]:=d[k,m+k-j+1]+x
    Else Begin b[k,j]:=d[k,j]+x; b[k,m+k-j+1]:=d[k,m+k-j+1]+x;End;
     End;
End;
End;
```

{Блок 3, номер машини більший числа правих стоянок +1}

```
If (i>n-m+1) and (m<=n-m) Then
  Begin
   k:=i;
   For j:=i+m-n to m+1 do
   Begin
    If b[k-1,j-1]>=b[k-1,m+k-j] Then x:=b[k-1,j-1] Else x:=b[k-
1,m+k-j];
     If j=m+1 Then b[k,m+k-j+1]:=d[k,m+k-j+1]+x;
     If j=i+m-n Then b[k,j]:=d[k,j]+x;
     If (j<>m+1) and (j<>i+m-n) ThenBegin b[k,j]:=d[k,j]+x;
b[k,m+k-j+1]:=d[k,m+k-j+1]+x;End;
   End;
```

```
  End;
 End;
{Вивід результату}
 If b[n,m]>=b[n,n] Then max:=b[n,m] Else max:=b[n,n];
 WriteLn(f2,max);
 Close(f1);Close(f2);
End.
```

E – Конфетна проблема Степана

Ім'я файлу, який містить вхідні дані: problem.in

Ім'я вихідного файлу: problem.out

Обмеження часу: 500 мс

Обмеження пам'яті: 128 M

Степан закохався і вирішив привернути увагу дівчини великою коробкою цукерок. За порадою друзів він поїхав унайвідомішу кондитерську фабрику ShenRo і дізнався, що великі коробки цукерок мають трикутну форму. Цукерки в цих коробках розташовуються в кілька рядів. У першому ряду знаходиться одна цукерка, у другому – дві, у третьому – три цукерки і так далі. На фабриці випускаються коробки цукерок із будь-яким числом рядів у межах від 1 до N. Степан хоче купити одну із таких коробок. Але є одна проблема: його дівчина засмутиться, якщо кількість цукерок у коробці не буде ділитись націло на M, тому що в цьому випадку комусь із друзів дівчини дістанеться більше цукерок, чим іншим, або ж якісь цукерки залишаться зайвими. Тому Степан вирішив, що число цукерок у коробці має обов’язково ділитись націло на M.

При виборі подарунка Степан зіткнувся з проблемою придбання відповідної коробки цукерок, оскільки можливих варіантів вибору коробки цукерок виявилося надто багато. Не довго думаючи, Степан вирішив звернутись за допомогою до учасників олімпіади.

Вам необхідно по заданих числах N і M знайти число способів вибору коробки цукерок із множини коробок з кількістю рядів від 1 до N. Способи вважаються різними, якщо їм відповідають коробки з різною кількістю рядів цукерок.

Формат вхідних даних: перший рядок вхідного файлу містить два цілих числа N – максимальна кількість рядів цукерок у коробці і M – кількість друзів дівчини Степана ($1 \le N, M \le 2*10^9$) відповідно.

Формат вихідних даних: вихідний файл має містити одне ціле число – кількість різних способів вибору коробки цукерок.

Оцінювання: N, M ≤ 1000 – не менше 35 балів, N, M ≤ 10^5 – не менше 55 балів.

Приклади

Вхідні дані розміщені у файлі problem.in	**Результат роботи знаходиться у файлі problem.out**
20 10	4
53 199	0
5705 145	157

Ідея розв'язку задачі Савченка Анатолія, учня 10 класу Корюківської ЗОШ І-ІІІ ст. № 1

Мова програмування C++, компілятор g++

Розв'язок

```
#include <iostream>
#include <cmath>

using namespace std;

int sequenceSolve ( int n, int m ) {
  int numWay = 0;

  int seq[4];
```

```
  int prevBoxAmount = 0;
  n++;
  int i = 1;
  for (; i < n && numWay <= 3; i++) {
     int currBoxAmount = prevBoxAmount + i;
     if (currBoxAmount % m == 0 ) {
        seq[numWay++] = i;
        //cout << i << endl;
     }
     prevBoxAmount = currBoxAmount;
  }
  if (i >= n) {
     return numWay;
  }

  int seq4 = seq[3];

  int newSeqEl = 0;
  while (true) {
     for (int i = 0; i < 4; i++) {
        newSeqEl = seq[i] + seq4;
        //cout << i << " " << seq[i] << " + " << seq4 << " = " << newSeqEl
<< endl;
        if (newSeqEl < n) {
           numWay++;
           seq[i] = newSeqEl;
        } else if (i == 0) {
           //cout << "Ends " << newSeqEl << "; Seq4 " << seq4 << endl;
           return numWay;
        }
     }
  }
  return numWay;
}

int mathSolve ( int n, int m ) {
  int maxBoxAmount = (( n * (n + 1) ) / 2); // Сумма первых n чисел N
ряда
```

```
  int numWay = 0;

  maxBoxAmount++;
  for (int i = m; i < maxBoxAmount; i += m) {
    int tmp = 8 * i + 1;

    if ( pow( trunc(sqrt(tmp)), 2 )  == tmp ) {
      numWay++;
    }
  }
  return numWay;
}

int progSolve ( int n, int m ) {
  int numWay = 0;

  int prevBoxAmount = 0;
  n++;
  for (int i = 1; i < n; i++) {
    int currBoxAmount = prevBoxAmount + i;

    if (currBoxAmount % m == 0 ) {
      numWay++;
    }
    prevBoxAmount = currBoxAmount;
  }
  return numWay;
}

int main() {
  int n, m;

  ifstream cin("problem.in");
  ofstream cout("problem.out");
  cin >> n >> m;

  cout << sequenceSolve(n, m) << endl;
  return 0;
}
```

Програма правильно виконує 10 тестів із 23, запропонованих журі

Ідея розв'язку задачі Зуба В.В., директора ЗОШ І-ІІІ ст. № 7 м. Прилук, учителя математики, учителя-методиста

Розв'язок

До оптимізації,
починаючи з тесту 11 і до 20 - обмеження за часом

```
var m,k,i,j:longint; f1,f2:text; x,y,z,n:int64;
    a:array[1..1000000] of longint;
Begin
 Assign(f1,'problem.in');Reset(f1);
 Assign(f2,'problem.out');Rewrite(f2);
 ReadLn(f1,n,m);
 x:=1;
 for i:=2 to n do
 Begin
   x:=x+i;
  If x mod m=0 Then inc(z);
 End;
 WriteLn(f2,z);
 Close(f1);Close(f2);
End.
```

Після оптимізації виконуються всі тести

```
var m,k,i,j:longint; f1,f2:text; x,y,z,n:int64;
    a:array[1..1000000] of longint;
Begin
 Assign(f1,'problem.in');Reset(f1);
 Assign(f2,'problem.out');Rewrite(f2);
```

```
 ReadLn(f1,n,m);
  If n<2*m Then x:=n Else x:=2*m;
  y:=1;j:=0;
  For i:=2 to x do
  Begin
    y:=y+i;
    If y mod m=0 Then Begin inc(z); inc(j); a[j]:=i;End;
 End;
  y:= n div (2*m);
  If y>0 Then z:=z*y;
  If n>2*m Then
     For i:=1 to j do
        If a[i]+y*2*m<=n Then inc(z) Else Break;
 WriteLn(f2,z);
 Close(f1);Close(f2);
End.
```

8-11 клас
II тур

А – Арифметика

Ім'я файлу, який містить вхідні дані: count.in

Ім'я вихідного файлу: count.out

Обмеження часу: 100 мс

Обмеження пам'яті: 128 М

Молодший брат Степана Мишко навчається в першому класі. Він дуже допитливий і постійно дістає Степана запитаннями: А скільки? А чому? Сьогоднішній день не виключення. Мишко каліграфічно виписує цифри в ряд і запитує: А скільки різних цифр у записі цього числа. На перші приклади Степан швидко знаходив відповідь. Але Мишко чим далі, тим більші числа записував. Це стало для Степана проблемою. Допоможіть Степану. Напишіть програму, яка визначає кількість різних цифр у числі Мишка.

Формат вхідних даних: перший рядок вхідного файлу містить одне ціле число N ($1 \le N \le 10^{1000}$), записане Мишком.

Формат вихідних даних: вихідний файл має містити одне число – кількість різних цифр у числі.

Приклади

Вхідні дані розміщені у файлі count.in	Результат роботи знаходиться у файлі count.out
1233	3

Ідея розв'язку задачі Черевка В.В., учителя Носівської ЗОШ І-ІІІ ст. № 1 та Гавриленка І.Б., учителя Носівської ЗОШ І-ІІІ ст. № 2

Текстовий рядок вводиться з вхідного файлу і обробляється посимвольно. Після аналізу прочитаного символу відповідним чином заповнюються елементи додатково

створеного масиву (finded – за прикладом програми). Для обчислення потрібного значення переглядаються елементи цього масиву.

Розв'язок

```
var
      finded:array ['0'..'9'] of boolean;
      in_f, out_f:text;
      c:char;
      i:byte;
begin
  for c:= '0' to '9' do
    finded[c]:= false;
  assign(in_f, 'count.in');
  reset(in_f);
  while not eof(in_f) do
  begin
    read(in_f, c);
    finded[c]:=true;
  end;
  close(in_f);
  i:=0;
  for c:= '0' to '9' do
    if finded[c] then inc(i);
  assign(out_f, 'count.out');
  rewrite(out_f);
  write(out_f, i);
  close(out_f);
end.
```

Ідея розв'язку задачі Зуба В.В., директора ЗОШ І-ІІІ ст. № 7 м. Прилук, учителя математики, учителя-методиста, Бондаренка С.М., учителя математики та інформатики ЗОШ І-ІІІ ст. № 7 м. Прилук, учителя-методиста

Так як число багатоцифрове (>255 символів), то скористаємося прийомом читання цифр числа посимвольно. Для початку створимо рядкову величину, яка буде містити всі 10 цифр. А потім будемо з неї видаляти цифри, які є в даному числі. Кількість цифр, що залишаться в рядковій величині потрібно відняти від числа 10 – це і буде шукана кількість цифр.

Розв'язок

```
Var t:char; n,k,i:longint; f1,f2:text; a:string;
Begin
 a:='0123456789'; {всі можливі цифри}
 Assign(f1,'count.in');Reset(f1);
 Assign(f2,'count.out');Rewrite(f2);
 While (not EOF(f1)) do
 Begin
  Read(f1,t); k:=0; k:=Pos(t,a);{пошук символів, що належать ряду a}
  If k<>0 Then Delete(a,k,1); {видалення символу з ряду a}
 End;
 WriteLN(f2,10-Length(a));
 Close(f1);Close(f2);
End.
```

B – Степан і сірники

Ім’я файлу, який містить вхідні дані: matches.in

Ім’я вихідного файлу: matches.out

Обмеження часу: 100 мс

Обмеження пам’яті: 128 M

Степан дуже полюбляє гратись із сірниками. Але він не балується ними, не розпалює вогонь, а розв'язує різні головоломки. Наприклад, він уміє прирівнювати число дев'ять до числа одинадцять, переклавши лише один сірник. Нещодавно батьки Степана подарували йому декілька наборів, кожен з яких складається з дванадцяти сірників. Хлопчик почав збирати з них різні геометричні фігури. Він уже зібрав багато різних фігур, але тепер йому стало цікаво: з яких наборів можливо склеїти каркас паралелепіпеда за допомогою дванадцяти сірників з набору та клею? Ламати сірники не можна і жоден із сірників не повинен виступати за каркас.

Ваше завдання полягає в тому, щоб за відомими довжинами сірників для кожного набору перевірити, чи можна з них склеїти каркас паралелепіпеда.

Формат вхідних даних: перший рядок вхідного файлу містить одне ціле число N ($1 \leq N \leq 100$), яке представляє кількість наборів. Далі йдуть N рядків, кожен з яких містить у собі опис набору сірників – дванадцять цілих додатніх чисел не перевищують 10^9.

Формат вихідних даних: вихідний файл має містити N рядків. Для кожного набору сірників виведіть “yes”, якщо з нього можливо склеїти каркас паралелепіпеда, і “no” в іншому випадку.

Приклади

Вхідні дані розміщені у файлі matches.in	Результат роботи знаходиться у файлі matches.out
2 1 1 1 1 2 2 2 2 3 3 3 3 1 1 1 1 2 2 2 2 3 3 3 4	yes no

Ідея розв'язку задачі Черевка В.В., учителя Носівської ЗОШ І-ІІІ ст. № 1 та Гавриленка І.Б., учителя Носівської ЗОШ І-ІІІ ст. № 2

Потрібно ввести із файлу в масив дані про розміри сірників у наборі і впорядкувати їх за зростанням значень.

Переглянути елементи масиву і переконатися в наявності трьох четвірок однакових значень. При цьому, оскільки елементи масиву впорядковані, доцільно перевіряти тільки перший та останній елемент кожної четвірки.

Розв'язок

```
#pragma hdrstop

#include <iostream>
#include <fstream>
#include <vector>
#include <algorithm>

using namespace std;

int main()
{
        ifstream cin(«matches.in»);
        ofstream cout(«matches.out»);
        int n;
        cin >> n;

        vector <long> a(12);
```

```
    for (int  k = 0; k < n; k++) {
        for (int  i = 0 ; i < 12; i++) {
            cin >> a[i];
        }
        sort(a.begin(), a.begin() + 12);
        int  m = 3;
        int j = 0;
        while (m < 12 && a[j] == a[m]) {
            j += 4;
            m += 4;
        }

        if (m < 12) {
            cout <<«no»;
        } else {
            cout <<«yes»;
        }

        cout << endl;
    }

    return 0;
}
```

Ідея розв'язку задачі Зуба В.В., директора ЗОШ І-ІІІ ст. № 7 м. Прилук, учителя математики, учителя-методиста; Бондаренка С.М., учителя математики та інформатики ЗОШ І-ІІІ ст. № 7 м. Прилук, учителя-методиста

Як відомо, паралелепіпед має 12 ребер – по 4 на кожен вимір. Отже, паралелепіпед можна склеїти з сірників, якщо є 3 групи по 4 рівних сірники. Відсортуємо розміри сірників за зростанням, і перевіримо, щоб сірники з 1-го по 4-й, з 5-го по 8-й та з 9-го по 12-й були рівними.

Розв'язок

```
type massiv = array[0..12] of longint;
var t:Char; n,k,i,j,x,y:longint; f1,f2:text;a:massiv;

Procedure QuickSort(var a: massiv; l, r: longint);
 var m, x, y: longint;
Begin
 m := ((l + r) div 2);
 i := l; j := r; x := a[m];
 while i <= j do
 begin
   While a[i] < x do inc(i);
   While a[j] > x do dec(j);
   If i <= j Then
   Begin
     y:=a[i];a[i]:=a[j];a[j]:=y;
     inc(i);
     dec(j);
   End;
 End;
 If l < j Then QuickSort(a, l, j);
 If r > i Then QuickSort(a, i, r);
End;

Begin
 Assign(f1,'matches.in');Reset(f1);
 Assign(f2,'matches.out');Rewrite(f2);
 ReadLn(f1,n);
 For x:=1 to n do
 Begin
  For y:=1 to 12 do Read(f1,a[y]); {зчитування 12 розмірів сірників}
```

```
 QuickSort(a, 1, 12);k:=0; {сортування масиву за зростанням}
 For y:=1 to 3 do If a[1+4*(y-1)]=a[4+4*(y-1)] Then k:=k+1;
{перевірка рівності 1-го елемента масиву з 4-м, 5-го з 8-м, 9-го з
12-м}
 If k<>3 Then WriteLN(f2,'no') Else WriteLN(f2,'yes');
End;
Close(f1);Close(f2);
End.
```

C – Задача від Степана

Ім'я файлу, який містить вхідні дані: task.in

Ім'я вихідного файлу: task.out

Обмеження часу: 750 мс

Обмеження пам'яті: 128 M

Перебираючи свої дитячі іграшки, Степан знайшов набір із N різних прямокутників і згадав задачу, яку йому колись задав старенький учитель математики. Назвемо прямокутник маленьким, якщо знайдеться інший прямокутник з даного набору, яким можна повністю накрити цей прямокутник. При цьому прямокутники можна повертати, але відповідні сторони мають бути паралельними.

Наприклад, прямокутник зі сторонами 1 і 10 можна повністю накрити прямокутником 10 і 3, але не можна накрити прямокутником зі сторонами 9 і 9. Прямокутники зі сторонами 10 і 3, а також зі сторонами 9 і 9 накрити не можна, відповідно в наборі із цих трьох прямокутників тільки один маленький. Напишіть програму, яка вирішить згадану Степаном задачу – визначити кількість маленьких прямокутників у даному наборі.

Формат вхідних даних: перший рядок вхідного файлу містить одне ціле число N ($2 \le N \le 200000$). У кожному з наступних N рядків міститься два цілих додатних числа –

розміри одного прямокутника. Усі розміри не перевищують 1000000. Серед даних прямокутників немає однакових.

Формат вихідних даних: вихідний файл має містити одне ціле число – кількість маленьких прямокутників у даному наборі.

Приклади

Вхідні дані розміщені у файлі task.in	Результат роботи знаходиться у файлі task.out
3 1 10 9 9 10 3	1
4 1 7 2 6 3 5 4 4	0

Ідея розв'язку задачі Черевка В.В., учителя Носівської ЗОШ І-ІІІ ст. № 1

Оскільки дитячі іграшки – прямокутники, необхідно перевіряти пари сторін. При повному порівнянні всіх прямокутників, у випадку великих чисел, алгоритм не задовольняє часові обмеження.

Тому для уникнення повного перебору пропонується:

- початковий масив даних формувати спеціальним чином (перший розмір прямокутника не більший другого);
- зменшити кількість порівнянь створюючи масив великих прямокутників.

Розв'язок:

```
#pragma hdrstop

#include <iostream>
#include <fstream>
#include <vector>
```

```
#include <algorithm>
#include <map>
using namespace std;
bool func(pair <long, long> a, pair <long, long> b) {
	if (a.first > b.first) return true;
	if (a.first == b.first && a.second > b.second ) return true;
	return false;
}
int main()
{
	vector < pair<long, long>> b;
	ifstream cin("task.in");
	ofstream cout("task.out");
	long n;
	cin >> n;

	vector < pair <long, long>> a;
	for (int  i = 0; i < n; i++) {
	  pair <long, long> temp;
	  cin >> temp.first >> temp.second;
	  if (temp.first > temp.second) {
	    long x = temp.first;
	    temp.first = temp.second;
	    temp.second = x;
	  }
	  a.push_back(temp);
	}
	int count = 0;
	sort(a.begin(), a.begin() + n, func);

	for (int i = 0; i < n; i++) {
	  bool flag = true;
	  for (int j = 0; j < count; j++) {
	    if (a[i].first <= b[j].first && a[i].second <= b[j].second) {
	      flag = false;
	      break;
```

```
            }
          }
          if (flag) {
            b.push_back(a[i]);
            count++;
    }
  }
  cout << n - count << endl;
  return 0;
}
```

Програма правильно виконує 23 тести із 45,далі – обмеження за часом

Ідея розв'язку задачі Зуба В.В., директора ЗОШ І-ІІІ ст. № 7 м. Прилук, учителя математики, учителя-методиста

Для початку розвернемо прямокутники так, щоб перший вимір завжди був не більшим за другий. Потім у подвійному циклі почнемо перебирати прямокутники. Якщо прямокутник із першого циклу "маленький", то збільшуємо лічильник і робимо наступну ітерацію. Якщо ж маленьким виявиться прямокутник із другого циклу, то збільшуємо лічильник і міняємо прямокутники місцями, щоб більше його не враховувати в підрахунках.

Розв'язок

```
type massiv = array[1..200000,1..2] of longint;
var t:Char; n,k,i,j,x,y:longint; f1,f2:text;
    a:massiv; ok:boolean;

Begin
 Assign(f1,'task.in');Reset(f1);
 Assign(f2,'task.out');Rewrite(f2);
 ReadLn(f1,n);
```

```
 For i:=1 to n do {введення прямокутників та систематизація їх вимірів за зростанням}
 Begin
  ReadLn(f1,x,y);
  If x<y Then Begin a[i,1]:=x; a[i,2]:=y; End
        Else Begin a[i,1]:=y; a[i,2]:=x; End;
 End;
 For i:=1 to n-1 do {i – номер першого прямокутника}
  For j:=i+1 to n do {j – номер другого прямокутника}
 Begin
  ok:=(a[i,1]-a[j,1]<=0) and (a[i,2]-a[j,2]<=0); {умова, що перший прямокутник менший за другий}
  If ok Then Begin k:=k+1; Break; End;
  ok:=(a[i,1]-a[j,1]>=0) and (a[i,2]-a[j,2]>=0); {умова, що другий прямокутник менший за перший}
  If ok Then
  Begin
    k:=k+1;
    x:=a[i,1]; y:=a[i,2];{обмін прямокутників місцями, }
    a[j,1]:=x; a[j,2]:=y;                          {щоб не враховувати їх }
    a[i,1]:=a[j,1]; a[i,2]:=a[j,2];{в подальшому                          }
    Break;
  End;
 End;
 WriteLn(f2,k);
 Close(f1);Close(f2);
End.
```

D – Штрафи

Ім'я файлу, який містить вхідні дані: penalty.in
Ім'я вихідного файлу: penalty.out
Обмеження часу: 500 мс
Обмеження пам'яті: 128 M

Степан нещодавно купив автомобіль, але водійські права ще не отримав. У зв’язку з цим він не має права на ньому їздити. Але його дружина вже спланувала вихідні, і поїздка до столиці входить у ці плани. Недовго думаючи, Степан знайшов вихід. Відомо, що ДАІ стоять не на всіх дорогах, а лише на тих, які обминути не можна, тому що так вони спіймають більше правопорушників. Відомо, що в країні Степана N міст, і вони з’єднані M дорогами. Зрозуміло, ніякі дві дороги не з’єднують одну й ту саму пару міст (у країні ж розумні люди працюють). Степан живе в місті A, а столиця знаходиться в місті 1. За відсутність водійських прав штраф складає 1000 карбованців. Скажіть, скільки в нього має бути при собі грошей, щоб він міг виплатити всі штрафи.

Формат вхідних даних: Перший рядок містить два числа N, M ($2 \le N \le 10^5$, $1 \le M \le 10^5$). Інші M рядків містять два числа Xi і Yi, які описують дорогу між містом Xi і містом Yi. В останньому рядку написано число A ($2 \le A \le N$) – місто в якому живе Степан.

Формат вихідних даних: Виведіть в одному рядку єдине число – кількість карбованців, які Степан має мати при собі.

Приклади

Вхідні дані розміщені у файлі penalty.in	Результат роботи знаходиться у файлі penalty.out
6 7 1 2 2 3 3 1 3 4 4 5 4 6 5 6 6	1000

Ідея розв'язку задачі Черевка В.В., учителя Носівської ЗОШ І-ІІІ ст. № 1

Тривіальним розв'язком є пошук усіх можливих траєкторій від вихідного міста до вхідного з обчисленням кількості проходження по кожній дорозі між містами у випадку доходження до міста призначення. Після аналізу всіх траєкторій обчислюється кількість ділянок, на яких кількість проходжень співпадає з кількістю траєкторій. Отримане число множиться на 1000.

Розв'язок

```
program Project2;

{$APPTYPE CONSOLE}
uses
  SysUtils;

Type  TPat = record
        X,Y,count:Longint;
          End;
Var
  f1,f2:Text;
  N,M:Longint;
```

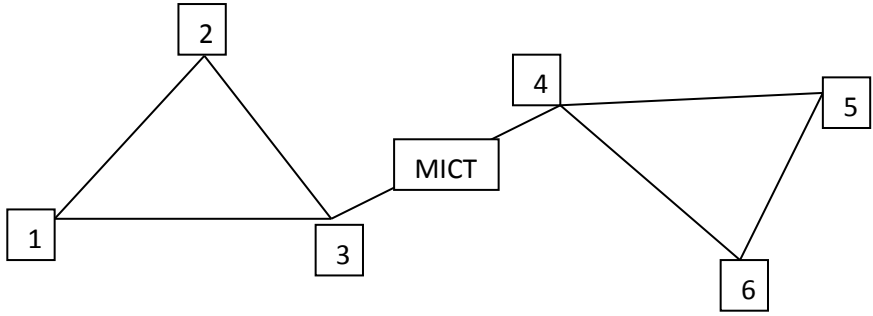
```
  i:Longint;
  Pats:Array[1..100000] of TPat;
  Tr:Array[1..100000] of Longint;
  k:Integer;
  A:Longint;
  Kp,Kt:LongInt;
  Shtr:Longint;
Procedure Search(Pt,A,Cp:Longint);
Var i,j:longint;
    oc:Longint;
    f:Boolean;
Begin
   If Pt=A then Begin Kt:=Kt+1;
   For i:=1 To Cp do inc(Pats[Tr[i]].count);
              End
   Else
   Begin
     For i:=1 to M do Begin
       f:=False;
       if (Pt = Pats[i].x) then begin oc:=Pats[i].Y; f:=True; end;
       if (Pt = Pats[i].Y) then begin oc:=Pats[i].X; f:=True; end;
       if f Then For j:=1 to Cp do If Tr[j]=i then f:=False;
       if f Then begin Tr[Cp+1]:= i; Search(oc,A,Cp+1); End;
   End;              End;
End;
begin
  { TODO -oUser -cConsole Main : Insert code here }
   Assign(f1,'penalty.in');
   Reset(f1);
   Assign(f2,'penalty.out');
   Rewrite(f2);

   Readln(f1,N,M);

   For i:=1 to M do ReadLn(f1,Pats[i].X,Pats[i].Y);
```

```
  Readln(f1,a);
  close(f1);
  Kt:=0;
  Search(1,A,0);
  Shtr:=0;
  For i:=1 to M do if (Pats[i].count=Kt) then Shtr:=Shtr+1000;
  Write(f2,Shtr);
  close(f2);
end.
```

Програма правильно виконує перші 2 тести, далі – обмеження за часом

Ідея розв'язку задачі Зуба В.В., директора ЗОШ І-ІІІ ст. № 7 м. Прилуки, учителя математики, учителя-методиста

Ідея розв'язку задачі полягає в пошуку «мостів» графа на шляху від вершини 1 до міста проживання Степана. Міст графа – це ребро, видалення якого збільшує число зв'язних компонентів графа. Малюнок до умови задачі.




```
varn,m,k,i,j,sum:longint; x,y:int64; f1,f2:text;
a:array of array of longint;
   p,low,pre:array[1..100000] of longint;
```

{low, pre – таблиці часу входження процедурою DFS у вершину графа, P[i] – таблиця вершин графа, в яку можна попасти з і-тої

```
вершини графа}
b:array[1..100000,1..2] oflongint; { Таблиця мостів графа}

{Процедура пошуку всіх мостів графа}
ProcedureDFS(v:longint);
var t: longint;
 Begin
  pre[v]:=j; low[v]:=j; inc(j);
  For t:=1 to a[v,0] do
   If a[v,t]<>0 then
    If pre[a[v,t]]=0 then
    Begin
     p[a[v,t]]:=v;
     DFS(a[v,t]);
     If low[v]<low[a[v,t]] Then low[v]:=low[v] Else
low[v]:=low[a[v,t]];
     If low[a[v,t]]>pre[v] Then
     Begin
       b[x,1]:=v; b[x,2]:=a[v,t]; inc(x); inc(sum);
     End;
    End
    else If p[v]<>a[v,t] Then
     If low[v]<pre[a[v,t]] Then low[v]:=low[v] Else
low[v]:=pre[a[v,t]];
 End;

Begin
 Assign(f1,'penalty.in');Reset(f1);
 Assign(f2,'penalty.out');Rewrite(f2);
 ReadLn(f1,n,m);
{Ініціалізація таблиць}
```

```
 SetLength(a,n+1);
 For i:=1 to n do SetLength(a[i],1);
 For i:=1 to n do Begin pre[i]:=0; low[i]:=0; p[i]:=0; End;
{Побудова таблиці інциденції даного графа, як одного із способів описання графа використовується для великих графів. Структура таблиці а динамічна з використанням процедури SetLength }
Fori:=1 tomdo
Begin
  ReadLn(f1,x,y);
  a[x,0]:=a[x,0]+1;
  SetLength(a[x],a[x,0]+1);
a[x,a[x,0]]:=y;
  a[y,0]:=a[y,0]+1;
  SetLength(a[y],a[y,0]+1);
  a[y,a[y,0]]:=x;
 End;
 ReadLn(f1,k);
 sum:=0; x:=1; y:=1; DFS(1);
{Пошук кількості мостів на шляху від вершини k до вершини 1}
x:=0;i:=k;
 While (i<>1) and (y<=n) do
 Begin
  For j:=1 to sum do
  Begin
   If (i=b[j,1]) and (p[i]=b[j,2]) Then Begin x:=x+1;Break End;
   If (i=b[j,2]) and (p[i]=b[j,1]) Then Begin x:=x+1;Break End;
  End;
  i:=p[i]; inc(y);
 End;
{Якщошлях не існує то вивід – 1, інакше вивід суми штрафу}
```

```
 If y>n Then WriteLn(f2,'-1') Else WriteLn(f2,x*1000);
 Close(f1);Close(f2);
End.
```

E – Ремонт

Ім'я файлу, який містить вхідні дані: repair.in
Ім'я вихідного файлу: repair.out
Обмеження часу: 1 с
Обмеження пам'яті: 64 M

Степан придбав нову квартиру і до приїзду батьків вирішив поклеїти шпалери. На перший погляд усе просто, але, коли він приступив до роботи, виявилась невелика проблема – необхідно вирівнювати малюнки на сусідніх смугах шпалер. Як визнаний програміст, Степан сформулював задачу таким чином. Кожну смугу шпалер можна описати її частиною – прямокутником довжиною N і шириною M (щоб отримати повну полосу, цей прямокутник можна багато разів домалювати до самого себе справа і зліва). Для простоти подумки поділимо цей прямокутник на рівні клітинки так, щоб утворилось N рядків і M стовпців. Щоб було ще простіше, рисунок на шпалерах позначимо символами ”.” і ”*” (крапка і зірочка), по одному символу в кожній клітинці.

Вам дано опис двох смуг шпалер. Допоможіть Степану. Напишіть програму, яка визначатиме, на яку мінімальну кількість клітинок потрібно змістити другу смугу вправо, щоб її малюнок співпав із малюнком на першій смузі. Степан придбав такі шпалери, що гарантовано завжди можна це зробити.

Формат вхідних даних: перший рядок вхідного файлу містить два цілих числа N і M($1 \le N \le 20$, $1 \le M \le 100000$). Наступні N рядків містять по M символів кожна – опис першої смуги шпалер. Наступні N рядків містять по M символів кожна – опис другої смуги шпалер. Кожен рядок опису шпалер містить тільки символи ”.” і ”*”.

Формат вихідних даних: вихідний файл має містити одне число – на яку мінімальну кількість клітинок потрібно змістити другу смугу вправо, щоб її малюнок співпав із малюнком на першій смузі.

Приклади

Вхідні дані розміщені у файлі repair.in	Результат роботи знаходиться у файлі repair.out
2 5 .*.*. *.*.* *.*.. .*.**	1
1 5 ***.. *..**	2

Ідея розв'язку задачі Черевка В.В., учителя Носівської ЗОШ І-ІІІ ст. № 1

Для розв'язку цієї задачі формуємо першу та другу смуги шпалер як масиви чисел. На наступному кроці здійснюється порівняння цих масивів на відповідність та оптимальний пошук необхідного зсуву.

Розв'язок

```
#include <iostream>
#include <fstream>

using namespace std;

const int MAXN = 100000;
int a[MAXN * 2];
	int b[MAXN];
	char s[MAXN + 1];
	int f[MAXN];
int main() {

	ifstream cin("repair.in");
```

```
ofstream cout("repair.out");
	int n, m;
	cin >> n >> m;
	for (int i = 0; i < n; ++i) {
		cin >> s;
		for (int j = 0; j < m; ++j) {
			if (s[j] == '*') a[j] |= 1 << i;
		}
	}
	for (int i = 0; i < n; ++i) {
		cin >> s;
		for (int j = 0; j < m; ++j) {
			if (s[j] == '*') b[j] |= 1 << i;
		}
	}

	for (int i = 0; i < m; ++i) {
		a[m + i] = a[i];
	}
	f[0] = 0;
	int j = 0;
	for (int i = 1; i < m; ++i) {
		while (j > 0 && b[j] != b[i]) {
			j = f[j - 1];
		}
		if (b[j] == b[i]) ++j;
		f[i] = j;
	}
	j = 0;
	for (int i = 0; i < 2 * m; ++i) {
		while (j > 0 && b[j] != a[i]) {
			j = f[j - 1];
		}
		if (b[j] == a[i]) ++j;
		if (j == m) {
			cout << i - m + 1 << endl;
```

```
                    break;
                }
        }
        return 0;
}
```

Ідея розв'язку задачі Зуба В.В., директора ЗОШ І-ІІІ ст. № 7 м. Прилуки, учителя математики, учителя-методиста

Розв'язок

```
var n,m,k,i,j,x,y,z,t:longint; f1,f2:text; max:int64;
    s:char;
    a,b,c,d,aa,bb:array[1..100000] of longint;
 Function stepin(nn:longint):longint; {функція піднесення 2 до степеня}
  var jj:longint;
 Begin
   stepin:=1;
   For jj:=1 to nn do stepin:=stepin*2;
 End;

Begin
 Assign(f1,'repair.in');Reset(f1);
 Assign(f2,'repair.out');Rewrite(f2);
ReadLn(f1,n,m);
Fori:=1 tondo  {формування малюнку першої смуги в цифровому форматі}
Begin
   For j:=1 to m do
   Begin
```

```
    Read(f1,s);
    If s='*' Then c[j]:=c[j]+stepin(n-i)
  End;
  Readln(f1);
 End;
```
Fori:=1 tondo{**формування малюнку другої смуги в цифровому форматі**}
```
Begin
  For j:=1 to m do
  Begin
    Read(f1,s);
    If s='*' Then d[j]:=d[j]+stepin(n-i)
  End;
  Readln(f1);
 End;
```
{**Блок визначення здвигу другої смуги**}
```
j:=1;
 For i:=1 to m do If c[i]<>0 Then Begin a[j]:=i; aa[j]:=c[i];inc(j);End;
 j:=1;
 For i:=1 to m do If d[i]<>0 Then Begin b[j]:=i;
bb[j]:=d[i];inc(j);End;
i:=1; max:=a[1]-b[1]; t:=0;
 While i<j do
 Begin
  If a[i]-b[i]<>max Then
  Begin
   x:=a[1]; z:=aa[1];
   For y:=1 to j-2 do Begin a[y]:=a[y+1]; aa[y]:=aa[y+1]; End;
   a[j-1]:=x+m;
   aa[j-1]:=z;
```

```
   max:=a[1]-b[1];
   i:=1;
  End;
  If aa[i]<>bb[i] Then
  Begin
   x:=aa[1];
   For y:=1 to j-2 do Begin aa[y]:=aa[y+1]; End;
   aa[j-1]:=x; inc(t);
  End;
  inc(i);
End;
```
{**Вивід результату здвигу другої смуги**}
```
If max<0 Then y:=m+max else y:=max;
 If (t>0) and (j-1=m) Then y:=t;
 WriteLn(f2,y);
 Close(f1);Close(f2);
End.
```