МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ
УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»

Основы информатики.
Задачи для практических занятий

Электронные методические указания

САМАРА
2011

УДК СГАУ: 004 (075)
ББК 22.18
О-753

Составитель: Пшеничников Виктор Владимирович

Основы информатики. Задачи для практических занятий [Электронный ресурс]: электрон. метод.. указания / Минобрнауки России, Самар.гос. аэрокосм. ун-т им. С.П. Королева (нац. исслед. ун-т); сост В. В. Пшеничников– Электрон. текстовые и граф. дан. (2.75 п.л., 0.45 МВ). – Самара, 2011. – 1 эл. опт. диск (CD ROM). – Систем. требования : ПК Pentium; Window 98 или выше

Ориентировано на обеспечение сквозного индивидуального обучения студентов факультета информатики по курсу «Основы информатики» 1 семестра направления 010400.62 «Прикладная математика и информатика». Методические указания содержат задачи по всем темам курса для самостоятельного решения и правила их оформления при сдаче на автоматизированной системе тестирования.

Подготовлено на кафедре геоинформатики и информационной безопасности СГАУ

Как писать решения на Паскале

Программы на Паскале компилируются на сервере с помощью 32-битного компилятора **FreePascal 2.0.4**. Он настроен в режиме совместимости с Borland Delphi и запускается со следующими параметрами:

```
ppc386 %1 -WC -Ci-o-r-t- -Xs -Sdgich -Se10 -l- -vwnh -n -dONLINE_JUDGE
```

Вы можете скачать FreePascal и прочитать online документацию на сайте www.freepascal.org.

Пример решения задачи

Вот пример решения задачи A + B на Паскале:

```
var
   a, b: integer;
begin
   readln(a, b);
   writeln(a + b);
end.
```

Что нового в 32-битном компиляторе по сравнению с 16-битным

Если вы привыкли писать программы в старых 16-битных компиляторах для DOS, например, в Borland Pascal 7.0, то вам будет полезно знать следующее:

- Можно использовать массивы размером более 64 КБ.
- Тип `integer` является 32-битным и совпадает с типом `longint`. 16-битный целый знаковый тип называется`smallint`.
- Строки теперь не ограничены 255 символами. Не следует обращаться к нулевому символу, чтобы узнать длину строки, используйте для этого функцию `length(s)`. Установить длину строки можно при помощи процедуры`setlength(s, n)`.

Кроме того, появилось много новых ключевых слов. Следующие слова не могут быть переопределены или использованы как идентификаторы:

```
as              false           new
class           finalization    on
dispose         finally         property
except          initialization  raise
exit            is              true
exports         library         try
```

В дополнение к указанным выше, слова `protected` и `published` стали зарезервированными внутри описания объекта. Также имя `result` в теле функции зарезервировано: значением соответствующей переменной является результат работы функции, и объявление второго такого имени приведёт к

ошибке компиляции. Например, следующие определения функций эквивалентны:

```
function Sum(a, b: integer): integer;
begin
   Sum := a + b;
end;

function Sum(a, b: integer): integer;
begin
   result := a + b;
end;
```

А вот такой код не скомпилируется:

```
function Sum(a, b: integer): integer;
var
   result: integer;
begin
   result := a + b;
   Sum := result;
end;
```

Особенности компилятора по сравнению с другими 32-битными компиляторами Паскаля

Есть некоторые особенности компилятора, используемого на сервере, которые полезно знать.

- На сервере нет модулей `crt` и `wincrt`, так как процедуры, содержащиеся в них, не нужны для решения задач.
- При динамическом выделении памяти размер блока округляется до кратного 16 байтам. Например, если вы выделяете много блоков по 4 байта, то ваша программа будет использовать в 4 раза больше памяти, чем надо. Пользуйтесь статическими структурами данных, они лишены этого недостатка и, как правило, работают быстрее динамических.
- Множества могут занимать лишь 4 или 32 байта, что также может привести к значительному перерасходу памяти. Часто хорошей альтернативой множествам оказывается использование битов целого числа (или массива целых чисел).
- Нельзя явно приводить типы переменных целочисленного типа и типа с плавающей точкой. Выражение«`x := extended(i)`», где `x` имеет тип `extended`, а `i` — `integer`, является ошибочным. Тогда как выражение«`x := i`» — правильное.

Нельзя менять значение переменной цикла `for` внутри цикла. Следующий код не скомпилируется:

```
for i := 1 to 10 do
begin
   if i mod 2 = 0 then inc(i);
```

```
   writeln(i);
end;
```

Более того, после цикла `for` значение переменной цикла не определено. Т.е. ее значение нельзя использовать, например, в целях поиска. Следующий пример сработает по-разному на разных компиляторах:

```
var
   a: array[1..10] of integer;
   i: integer;
begin
   for i:=1 to 10 do
      a[i] := i*i;
   for i:=1 to 10 do
      if a[i] = 17 then break;
   writeln('i = ', i);
end.
{
   Delphi output:      i = 11
   FreePascal output:  i = 10
}
```

Как пользоваться 64-битными целыми типами данных

Компилятор полностью поддерживает 64-битные целые, как со знаком, так и без знака. 64-битное целое со знаком имеет диапазон значений от –9223372036854775808 до 9223372036854775807, без знака — от 0 до 18446744073709551615. Знаковый тип называется `int64`, беззнаковый — `qword`. Следующий пример иллюстрирует использование 64-битных целых типов:

```
var
   a: int64;
   b: qword;
begin
   read(a);
   read(b);
   writeln(a);
   writeln(b);
end.
```

Как считать данные до конца входного потока

Для решения некоторых задач требуется уметь читать входные данные до конца входного потока. Следующие примеры показывают, как можно организовать ввод до конца входного потока в случаях, если входные данные состоят из чисел, из строк или из отдельных символов.

```
{numbers}
var n: integer;
...
while not seekeof do
begin
   read(n);
```

```
   ...
end;

{lines}
var line: string;
...
while not eof do
begin
   readln(line);
   ...
end;

{characters}
var c: char;
...
while not eof do
begin
   read(c);
   ...
end;
```

Прочие замечания

Иногда вердикт Wrong Answer на самом деле означает Crash. Это связано с тем, что FreePascal самостоятельно перехватывает некоторые ошибки выполнения программ и выводит сообщение в выходной поток.

Для того, чтобы увеличить объем стека и избежать его переполнения при использовании «глубокой» рекурсии, следует использовать специальную директиву (в примере размер стека устанавливается равным 16 МБ):

```
{$M 16777216}
```

Удобно использовать для отладки решений условную директиву `ONLINE_JUDGE`:

```
var
   a, b: longint;
begin
{$IFNDEF ONLINE_JUDGE}
   assign(input, 'input.txt');
   reset(input);
   assign(output, 'output.txt');
   rewrite(output);
{$ENDIF}
   readln(a, b);
   writeln(a + b);
{$IFNDEF ONLINE_JUDGE}
   close(input);
   close(output);
{$ENDIF}
end.
```

Такая программа на вашем компьютере будет читать данные из файла `input.txt` и выводить результат в файл`output.txt`. На сервере же она будет пользоваться стандартными входным и выходным потоками.

Как писать решения на Java

С 1 июня 2006 года Timus Online Judge поддерживает язык программирования Java. Для компиляции и запуска решений на Java используется **J2SE Development Kit (JDK) 6.0 Update 11** (до 2 февраля 2009 использовался JDK 5.0 Update 15). Вы можете скачать JDK и прочитать online документацию на сайте java.sun.com.

Программы на Java запускаются на сервере с командной строкой:

```
java -Xmx64m -Xms64m -Xss64m -DONLINE_JUDGE YourClassName
```

Как должно выглядеть решение на Java

Программа на Java, посылаемая на проверку, должна содержать ровно один public класс. Этот класс может называться как угодно и он должен содержать метод "`public static void main(String[] args)`". Кроме того, программа может содержать любое количество вложенных классов и глобальных непубличных классов. Вот пример решения задачи A + B:

```
import java.io.*;
import java.util.*;

public class Sum
{
   public static void main(String[] args)
   {
      Scanner in = new Scanner(System.in);
      PrintWriter out = new PrintWriter(System.out);

      int a = in.nextInt();
      int b = in.nextInt();
      out.println(a + b);

      out.flush();
   }
}
```

Это решение тоже является правильным:

```
import java.util.*;

class YouCanUseSuchClasses {}
public class Sum2
{
   class AndSuchClassesToo {}
   public static void main(String[] args)
   {
      Scanner in = new Scanner(System.in);
      System.out.println(in.nextInt() + in.nextInt());
   }
}
```

Время работы и используемая память

Решения на Java почти всегда работают достаточно большое время и требуют много памяти по сравнению с другими языками независимо от сложности задачи. Такова особенность этого языка. Но, тем не менее, гарантируется, что почти все задачи на Timus Online Judge могут быть сданы на Java без особых проблем по сравнению с другими языками. Вот список всех задач, для которых это не гарантируется: 1220, 1275, 1306.

Локализация

По умолчанию на сервере используется локаль `Locale.US`. Если вы установите в своей программе другую локаль, это может повлиять на способность Вашей программы читать/писать числа с плавающей точкой.

Ввод/вывод

Ввод/вывод в Java может стать очень медленным, если пользоваться им неправильно. Вот несколько правил, соблюдая которые, вы сможете избежать проблем, связанных со вводом/выводом:

- `Scanner` является самым удобным средством для чтения входных данных в большинстве задач, но скорость его работы оставляет желать лучшего. Используйте его только для чтения небольших входных данных.
- `BufferedReader` обеспечивает достаточно быстрый ввод для большинства задач. Но самостоятельно этот класс позволяет лишь читать отдельные символы и строки. Для чтения токенов и чисел используйте`StringTokenizer` или `StreamTokenizer`.
- `PrintWriter` подходит для всех случаев и работает достаточно быстро. Но его метод `printf` работает медленно; также медленно работают вызовы типа `println(a + " " + b)`. Выводите по одной переменной за раз, и тогда вы добьетесь максимальной эффективности.

Вот пример эффективного использования классов `StreamTokenizer` и `PrintWriter` (задача — вывести через пробел сумму и разность чисел A и B, записанных во входных данных):

```
import java.io.*;
import java.util.*;

public class SumDif
{
   public static void main(String[] args) throws IOException
   {
      new SumDif().run();
   }

   StreamTokenizer in;
   PrintWriter out;
```

```
   int nextInt() throws IOException
   {
      in.nextToken();
      return (int)in.nval;
   }

   void run() throws IOException
   {
      in = new StreamTokenizer(new BufferedReader(new
InputStreamReader(System.in)));
      out = new PrintWriter(new OutputStreamWriter(System.out));
      solve();
      out.flush();
   }

   void solve() throws IOException
   {
      int a = nextInt();
      int b = nextInt();
      out.print(a + b);
      out.print(" ");
      out.println(a - b);
   }
}
```

Для того чтобы читать и выводить символы с кодами больше 127 также как и в других языках программирования следует использовать следующие конструкторы:

```
Scanner scanner = new Scanner(System.in, "ISO-8859-1");
BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in, "ISO-8859-1"));
PrintWriter writer = new PrintWriter(new
OutputStreamWriter(System.out, "ISO-8859-1"));
```

Это нужно делать, так как насчет кодировки по умолчанию на сервере ничего не гарантируется. Скорее всего, использование кодировки по умолчанию приведет к тому, что часть символов с кодами 128-255 будут преобразованы в символы Unicode с кодами больше 255. А при выводе эти символы могут замениться, например, на вопросительные знаки.

Использование свойства ONLINE_JUDGE

Запуск программы на сервере осуществляется с определенным свойством `ONLINE_JUDGE`. С его помощью в коде программы можно определить, что она запущена на сервере. Например, это можно использовать для решения, использовать ли для работы файлы или стандартный ввод/вывод:

```
boolean oj = System.getProperty("ONLINE_JUDGE") != null;
Reader reader = oj ? new InputStreamReader(System.in) : new
FileReader("input.txt");
Writer writer = oj ? new OutputStreamWriter(System.out) : new
FileWriter("output.txt");
StreamTokenizer in = new StreamTokenizer(new BufferedReader(reader));
```

```
PrintWriter out = new PrintWriter(writer);
```

Как писать решения на C/C++

Программы на C и C++ компилируются на сервере с помощью 32-битного компилятора **Microsoft Visual C++ 2010**. До 3 августа 2010 года использовался компилятор Intel C++ Compiler 7.0. Вы можете бесплатно скачать Microsoft Visual Studio 2010 Express на этой странице. Компилятор запускается со следующими параметрами:

```
// C
cl /TC /MT /EHsc /O2 /W3 /Za /D "_CRT_SECURE_NO_WARNINGS"
   /D "_CRT_SECURE_NO_DEPRECATE" /D "ONLINE_JUDGE"

// C++
cl /TP /MT /EHsc /O2 /W3 /Za /D "_CRT_SECURE_NO_WARNINGS"
   /D "_CRT_SECURE_NO_DEPRECATE" /D "ONLINE_JUDGE"
```

Всю необходимую документацию по Visual C++ 2010 вы найдете в соответствующем разделе MSDN.

Пример решения задачи

Вот пример решения задачи A + B на языке C:

```
#include <stdio.h>
int main()
{
   int a, b;
   scanf("%d%d", &a, &b);
   printf("%d\n", a + b);
   return 0;
}
```

И пример решения той же задачи на языке C++:

```
#include <iostream>
int main()
{
   int a, b;
   std::cin >> a >> b;
   std::cout << a + b << std::endl;
   return 0;
}
```

Ввод/вывод

Пример того, как надо пользоваться вводом/выводом, приведен выше. У тех, кто пишет на C++, всегда есть выбор из двух библиотек ввода/вывода — стандартного (`scanf`, `printf`) и потокового (`cin`, `cout`). Потоковый ввод/вывод удобен в использовании, но работает гораздо медленнее стандартного. Это даже может стать причиной получения вердикта Time limit exceeded. Поэтому, если

вы видите, что в задаче надо считывать много входных данных (скажем, больше мегабайта) или много выводить, то не следует использовать потоковый ввод/вывод.

Не забывайте, что функции чтения символов возвращают отрицательные результаты для символов с кодами 128-255. Если во входных данных могут присутствовать такие символы, то лучше явно привести тип считанного символа к `unsigned char`, чем получить из-за этого вердикты Wrong answer или Crash (access violation).

Для решения некоторых задач требуется уметь читать входные данные до конца входного потока. Следующие примеры показывают, как можно организовать ввод до конца входного потока в случаях, если входные данные состоят из чисел, из строк или из отдельных символов.

```
#include <stdio.h>
...
// numbers
int n;
while (scanf("%d", &n) != EOF)
{
   ...
}
// lines
char line[1000];
while (gets(line))
{
   ...
}
// characters
int c;
while ((c = getchar()) != EOF)
{
   ...
}
```

```
#include <iostream>
...
// numbers
int n;
while (std::cin >> n)
{
   ...
}
// lines
std::string line;
while (std::getline(std::cin, line))
{
   ...
}
// characters
char c;
while (std::cin.get(c))
{
   ...
}
```

Особенности компилятора по сравнению с другими 32-битными компиляторами C/C++

Есть некоторые особенности компилятора, используемого на сервере, которые следует знать, чтобы не получать вердикт Compilation error.

- Нельзя вызывать функции из заголовочного файла `math.h` от целочисленных параметров. Такой код не скомпилируется: `sqrt(2)`. Правильно писать `sqrt((double)2)` или `sqrt(2.0)`.
- В заголовочном файле `math.h` определены функции `j0`, `j1`, `jn`, `y0`, `y1`, `yn`. В результате нельзя объявлять глобальные переменные с такими именами, если используется `math.h`. В заголовочных файлах библиотеки STL определены функции `next`, `prev`, `begin`, `end`, `rank`, `count`, `hash`. Эти имена нельзя использовать для именования глобальных переменных и функций.
- Тип `long double` совпадает с типом `double`. Т.е. `sizeof(long double) = sizeof(double) = 8`. Для решения задач всегда достаточно

типа `double` (если только не требуется своя реализация чисел с плавающей точкой).
- Не поддерживается функция `itoa`. Пользуйтесь функцией `sprintf`.
- Область видимости переменной, объявленной в заголовке оператора `for`, ограничена телом этого оператора.

Как пользоваться 64-битными целыми типами данных

Компилятор полностью поддерживает 64-битные целые, как со знаком, так и без знака. 64-битное целое со знаком имеет диапазон значений от –9223372036854775808 до 9223372036854775807, без знака — от 0 до 18446744073709551615. Объявить 64-битную переменную можно следующим способом:

```
long long a;
unsigned long long b;
```

Читать и выводить 64-битные переменные также можно двумя способами в зависимости от используемой библиотеки ввода/вывода:

```
#include <stdio.h>
...
scanf("%lld", &a);
scanf("%llu", &b);
printf("%lld", a);
printf("%llu", b);

#include <iostream>
...
std::cin >> a;
std::cin >> b;
std::cout << a;
std::cout << b;
```

Библиотека STL

STL — стандартная библиотека шаблонов (Standard Template Library), неотъемлемая часть C++, предоставляющая в распоряжение программиста набор контейнеров, итераторов и алгоритмов. Умелое использование этой библиотеки помогает сократить объем кода и уменьшить время на написание программы.

Контейнеры STL представляют собой готовые реализации часто использующихся структур данных. Самыми полезными при решении задач являются:

- динамический массив `vector`;
- сортированный ассоциативный массив `map` и сортированное множество элементов `set`, основанные на красно-черных деревьях (операции

удаления, вставки и поиска элементов выполняются за логарифмическое время);
- очередь `queue`, очередь с приоритетами `priority_queue`, дек `deque` и стек `stack`.

Итератор в STL — это абстракция, позволяющая обходить элементы какого-либо контейнера. С точки зрения программиста, итератором является любой объект, поддерживающий операторы разыменования указателя на текущий элемент (`*`, `->`) и перемещения к следующему элементу (`++`) (обычный указатель тоже является итератором). Все контейнеры STL предоставляют итераторы для обхода своих элементов; они доступны через обращение`container_type::iterator`, где `container_type` — это тип контейнера.

Самый часто применяемый алгоритм STL — это быстрая сортировка `sort`, работающая за время O(N × log N). Всего в библиотеке определено около 60 алгоритмов, включая бинарный поиск `binary_search`, генерацию следующей перестановки элементов `next_permutation`, линейный поиск `find` и многие другие.

Рассмотрим пример решения задачи с использованием STL. Пусть дан набор строк, требуется вывести все различные строки в лексикографическом порядке и для каждой указать, сколько раз она встретилась в наборе.

```
#include <iostream>
#include <map>
#include <string>

int main()
{
   std::map<std::string, int> words;
   std::string word;

   while (std::getline(std::cin, word))
      words[word]++;
   for (std::map<std::string, int>::iterator it = words.begin(); it != 
words.end(); it++)
      std::cout << it->first << " - " << it->second << std::endl;

   return 0;
}
```

Так как операции с контейнером `map` выполняются за логарифмическое время, сложность решения — O(N × log N), где N — объем входных данных.

Прочие замечания

В решениях задач нельзя кидать исключения (даже внутри блока `try … catch`) — это будет проинтерпретировано проверяющей системой как Crash.

Для того, чтобы увеличить объем стека и избежать его переполнения при использовании «глубокой» рекурсии, следует использовать специальную директиву (в примере размер стека устанавливается равным 16 МБ):

```
#pragma comment(linker, "/STACK:16777216")
```

Удобно использовать для отладки решений условную директиву `ONLINE_JUDGE`:

```
#include <stdio.h>
int main()
{
#ifndef ONLINE_JUDGE
   freopen("input.txt", "rt", stdin);
   freopen("output.txt", "wt", stdout);
#endif
   int a, b;
   scanf("%d%d", &a, &b);
   printf("%d\n", a + b);
   return 0;
}
```

Обратите внимание на то, что функцию `freopen` можно применять и при использовании потокового ввода/вывода:

```
#include <iostream>
int main()
{
#ifndef ONLINE_JUDGE
   freopen("input.txt", "rt", stdin);
   freopen("output.txt", "wt", stdout);
#endif
   int a, b;
   std::cin >> a >> b;
   std::cout << a + b << std::endl;
   return 0;
}
```

Как писать решения на C#

С 31 декабря 2006 года Timus Online Judge принимает решения на языке C#. Программы компилируются на сервере с помощью **Microsoft Visual C# 2010 Compiler версии 4.0.30319.1**. До 4 августа 2010 года использовался компилятор Microsoft Visual C# 2008 Compiler версии 3.5.30729.1; до 8 февраля 2009 года использовался компилятор Microsoft Visual C# 2005 версии 8.00.50727.42. Компилятор запускается со следующими параметрами:

```
csc /o+ /d:ONLINE_JUDGE /r:System.Numerics.dll
```

Примеры решения задач

Вот пример решения задачи A + B на языке C#:

```
using System;

public class Sum
{
    private static void Main()
    {
        string[] tokens = Console.ReadLine().Split(' ');
        Console.WriteLine(int.Parse(tokens[0]) + int.Parse(tokens[1]));
    }
}
```

Далее приведен пример простого решения задачи Обратный корень на языке C#. Этот пример иллюстрирует некоторые возможности языка и не является самым эффективным. При посимвольном вводе можно добиться гораздо меньшего времени работы и объема использованной памяти.

```
using System;
using System.Globalization;

public class ReverseRoot
{
   private static void Main()
   {
      NumberFormatInfo nfi = NumberFormatInfo.InvariantInfo;
      string[] input = Console.In.ReadToEnd().Split(
         new char[] {' ', '\t', '\n', '\r'},
StringSplitOptions.RemoveEmptyEntries);
      for (int i = input.Length - 1; i >= 0; i--)
      {
         double root = Math.Sqrt(double.Parse(input[i], nfi));
         Console.WriteLine(string.Format(nfi, "{0:F4}", root));
      }
   }
}
```

Время работы и используемая память

Решения на C# по времени работы и по объему используемой памяти сравнимы с решениями на Java. Поэтому, почти все задачи на Timus Online Judge могут быть сданы на C# без особых проблем по сравнению с C/C++ и Pascal. Для решения следующих задач лучше не использовать Java и C#: 1220, 1275, 1306.

Размер стека по-умолчанию установлен равным 64 МБ.

Ввод/вывод

Для ввода/вывода на C# не всегда достаточно стандартных методов `Console.ReadLine`, `String.Split` и`Console.WriteLine`. В некоторых задачах потребуется собственная быстрая реализация разбора входных данных и форматирования выходных.

В некоторых задачах числа разделяются не ровно одним пробелом. Поэтому вместо

```
Console.ReadLine().Trim().Split(' ')
```

следует использовать

```
Console.ReadLine().Split(new char[] {' ', '\t'},
StringSplitOptions.RemoveEmptyEntries)
```

Не забывайте, что культура по умолчанию может быть любой. Это важно, если вам требуется считать или вывести число с плавающей точкой: разделитель целой и дробной части может быть задан в системе как «.» или как «,». В настоящий момент на сервере настроен разделитель «.», однако это может измениться в будущем. Чтобы не сталкиваться с подобными проблемами, указывайте культуру явно при каждой операции ввода/вывода или установите культуру по умолчанию для всей своей программы:

```
Thread.CurrentThread.CurrentCulture = CultureInfo.InvariantCulture;
```

1025. Демократия в опасности

Ограничение времени: 1.0 секунды
Ограничение памяти: 16 МБ

Вступление

В одном из островных государств Карибского бассейна все решения традиционно принимались простым большинством голосов на общем собрании граждан, которых, к счастью, было не очень много. Одна из местных партий, стремясь прийти к власти как можно более законным путем, смогла добиться некоторой реформы избирательной системы. Главным аргументом было то, что население острова в последнее время значительно возросло, и проведение общих собраний перестало быть легкой задачей.

Суть реформы состояла в следующем: с момента введения ее в действие все избиратели острова делились на K групп (необязательно равных по численности). Голосование по любому вопросу теперь следовало проводить отдельно в каждой группе, причем считалось, что группа высказывается «за», если «за» голосует более половины людей в этой группе, а в противном случае считалось, что группа высказывается «против». После проведения голосования в группах подсчитывалось количество групп, высказавшихся «за» и «против», и вопрос решался положительно в том и только том случае, когда групп, высказавшихся «за», оказывалось более половины общего количества групп.

Эта система вначале была с радостью принята жителями острова. Когда первые восторги рассеялись, очевидны стали, однако, некоторые недостатки новой системы. Оказалось, что сторонники партии, предложившей систему, смогли оказать некоторое влияние на формирование групп избирателей. Благодаря этому, они получили возможность проводить некоторые решения, не обладая при этом реальным большинством голосов!

Пусть, например, на острове были сформированы три группы избирателей, численностью в 5, 5 и 7 человек соответственно. Тогда партии достаточно иметь трех сторонников в каждой из первых двух групп, и она сможет провести решение всего шестью голосами «за», вместо девяти, необходимых при общем голосовании.

Задача

Вам надо написать программу, которая определяет по заданному разбиению избирателей на группы минимальное количество сторонников партии, достаточное, при некотором распределении их по группам, для принятия любого решения.

Исходные данные

В первой строке записано нечётное число K — количество групп избирателей ($1 \le K \le 101$). Во второй строке через пробел записаны K нечётных чисел, которые

задают количество избирателей в группах. Население острова не превосходит 9999 человек.

Результат

Выведите минимальное количество сторонников партии, способное решить исход голосования.

Пример

исходные данные	результат
3 5 7 5	6

Автор задачи: Леонид Волков

Источник задачи: Второе командное соревнование школьников Свердловской области по программированию, 7 октября 2000

1068. Сумма

Ограничение времени: 2.0 секунды

Ограничение памяти: 16 МБ

Всё, что от вас требуется — найти сумму всех целых чисел, лежащих между 1 и N включительно.

Исходные данные

В единственной строке расположено число N, по модулю не превосходящее 10000.

Результат

Выведите целое число — ответ задачи.

Пример

исходные данные	результат
-3	-5

Источник задачи: 2000-2001 ACM Northeastern European Regional Programming Contest (test tour)

1084. Пусти козла в огород

Ограничение времени: 1.0 секунды

Ограничение памяти: 16 МБ

Козла пустили в квадратный огород и привязали к колышку. Колышек воткнули точно в центре огорода. Козёл голоден, как волк, прожорлив, как бык, и ест всё, до чего дотянется, не перелезая через забор и не разрывая веревку. Какая площадь огорода будет объедена?

Исходные данные

Длина стороны огорода и длина верёвки в метрах (положительные целые числа, не превосходящие 100, расположенные в одной строке через пробел).

Результат

Площадь части огорода (в квадратных метрах, с точностью до 3 знаков после десятичной точки), объеденной козлом.

Пример

исходные данные	результат
10 6	95.091

Автор задачи: Ирина Данилина

Источник задачи: Третье командное соревнование школьников Свердловское области по программированию, 4 марта 2001

1194. Рукопожатия

Ограничение времени: 1.0 секунды

Ограничение памяти: 16 МБ

После вечеринки в трактире «Гарцующий пони» хоббиты расходились по домам. На первом перекрёстке вся компания разделилась на несколько групп, каждая из которых пошла своей дорогой. Разумеется, соблюдая правила хорошего тона, хоббиты перед расставанием обменялись рукопожатиями (каждый хоббит пожал руку каждому, с которым он расставался, вне зависимости от пола). Каждая группа, дойдя до следующего перекрёстка, в свою очередь разделилась на меньшие группы, также обменявшись рукопожатиями. Этот процесс продолжался до тех пор, пока все холостые хоббиты и супружеские пары не разошлись по домам. Другими словами, группы разбивались до тех пор, пока не осталось ни одной группы из трёх и более хоббитов. Ваша задача — подсчитать количество сделанных рукопожатий.

Исходные данные

Пронумеруем группы хоббитов так, что первая (та, которая вышла из трактира) получит номер 1, а все остальные — различные последовательные номера от 2 и далее. В первой строке записаны два числа N и K — общее количество хоббитов, вышедших из трактира и количество супружеских пар. Эти числа удовлетворяют ограничениям: $2 < N \le 20000$; $0 \le 2K \le N$. В каждой следующей строке записан номер группы, затем — количество групп, на которые она разбивается, потом идут пары чисел, разделенные пробелами — номер и численность каждой новой группы. Гарантируется, что если группа Y получилась в результате разбиения группы X, то описание разбиения группы Xбудет раньше описания разбиения группы Y. Это, в частности, означает, что описание группы 1 расположено во второй строке. Если группа Y получилась в результате разбиения группы X, но её описания не последовало, — это означает, что группа Y больше не разбивалась.

Результат

Выведите количество сделанных рукопожатий.

Пример

исходные данные	результат
3 0 1 2 2 2 3 1 2 2 4 1 5 1	3

Автор задачи: Леонид Волков

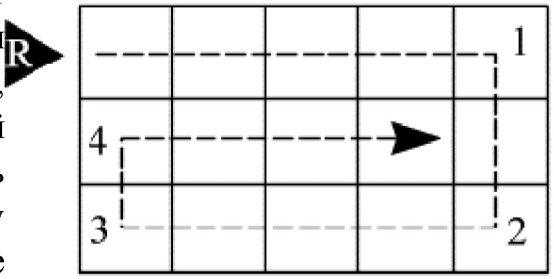
Источник задачи: V командное первенство школьников по программированию (2 марта 2002)

1224. Спираль

Ограничение времени: 1.0 секунды

Ограничение памяти: 16 МБ

Новейший робот-сапёр способен производить разминирование в прямоугольной области целочисленной высоты и ширины (N и M соответственно). Перед началом работы робота помещают около левой верхней ячейки прямоугольника таким образом, что изначально он поедет вправо. Затем робот начинает двигаться и нейтрализовывать мины по пути, спирально закручивающемуся по часовой стрелке (см. рисунок). Спираль закручивается по направлению к центру области, в конечном итоге покрывая все клетки. Область считается безопасной, когда все клетки посещены и проверены роботом.



$N = 3$
$M = 5$

Ваша задача — определить количество поворотов, которые робот совершит во время выполнения своей задачи.

Исходные данные

Ввод состоит из двух целых чисел в следующем порядке: N, M ($1 \le N, M \le 2^{31} - 1$).

Результат

Вывод состоит из одного целого числа — количества поворотов.

Пример

исходные данные	результат
3 5	4

Источник задачи: Четвертьфинальные соревнования ACM ICPC 2002–2003 в центральном регионе России, Рыбинск, октябрь 2002

1243. Развод семи гномов

Ограничение времени: 1.0 секунды

Ограничение памяти: 16 МБ

Все мы знаем, чем закончилась история про Белоснежку и семь гномов — Белоснежка уехала с женихом, бросив всех тех, кто бескорыстно помог ей в трудную пору. После её отъезда гномы стали ссориться — каждый считал, что другие чем-то обидели Белоснежку.

Чтобы не доводить вечные ссоры до кровопролития, некогда дружные гномы решили расстаться, поделив все свое добро, от кружек до алмазов, согласно старинным гномьим законам о разводе. По этим законам, все имущество должно быть самым справедливым образом поделено между гномами, а то, что поделить поровну нельзя, не должно достаться никому из них. Бережливые гномы решили, что неразделённые вещи выкидывать не будут, а отдадут Белоснежке в качестве приданого.

Например, у каждого из гномов с рождения имелось по две пары ботинок, к моменту появления этой задачи самый старый гном одну свою пару износил, значит, после справедливого раздела оставшихся 26 ботинок, каждый гном получит по 3 ботинка, а Белоснежку осчастливят 5 ботинками.

Отметим, что некоторых вещей у гномов очень много — одних только маковых зёрнышек накопилось 123456123456 штук. Гномы потратили немало времени, пока сосчитали, что Белоснежка получит всего одно маковое зёрнышко. Ваша задача — помочь гномам рассчитать долю Белоснежки.

Исходные данные

Единственная строка содержит число N одинаковых вещей, которые хотят поделить гномы ($1 \le N \le 10^{50}$).

Результат

Единственное число — количество вещей, которые в результате справедливого раздела, перейдут к Белоснежке.

Пример

исходные данные	результат
123456123456	1

Автор задачи: Станислав Васильев

Источник задачи: Ural State University Personal Programming Contest, March 1, 2003

1264. Трудовые будни

Ограничение времени: 1.0 секунды

Ограничение памяти: 16 МБ

После успеха предыдущей программы Васечкина, позволившей подвести итоги выборов всего за два дня, Васечкин был назначен начальником отдела. Не правда ли, успех? В данный момент Артемий Сидорович готовит техническое задание для своего подчиненного — программиста Петечкина. Задание заключается в написании крайне полезной функции, которая намного облегчит жизнь всем программистам отдела. Для каждого числа от 0 доM функция будет подсчитывать число раз, которое число встречается в N-элементном массиве. Артемий Сидорович полагает, что функция должна работать следующим образом (пример кода для $N = 3$, $M = 1$):

C	Pascal

`if (arr[0]==0) ++count[0];`
`if (arr[0]==1) ++count[1];`
`if (arr[1]==0) ++count[0];`
`if (arr[1]==1) ++count[1];`
`if (arr[2]==0) ++count[0];`
`if (arr[2]==1) ++count[1];`
	`if arr[0]=0 then count[0] := count[0] + 1;`
	`if arr[0]=1 then count[1] := count[1] + 1;`
	`if arr[1]=0 then count[0] := count[0] + 1;`
	`if arr[1]=1 then count[1] := count[1] + 1;`
	`if arr[2]=0 then count[0] := count[0] + 1;`
	`if arr[2]=1 then count[1] := count[1] + 1;`

Артемий Сидорович хочет оценить время, за которое Петечкин справится с заданием. Известно, что Петечкин пишет одну строчку кода за одну секунду (не правда ли, очень быстро?). Артемий Сидорович точно не знает, какимиM и N можно ограничиться. Ваша цель — написать программу, которая подсчитает число секунд, которые потребуется Петечкину на написание кода.

Исходные данные

Единственная строка содержит целые числа N ($0 \le N \le 40000$) и M ($0 \le M \le 40000$).

Результат

Выведите число секунд, требуемых Петечкину на написание функции.

Пример

исходные данные	результат
3 1	6

Автор задачи: Ден Расковалов

Источник задачи: Открытое командное соревнование школьников Свердловской области по программированию, 11 октября 2003 года

1293. Эния

Ограничение времени: 1.0 секунды

Ограничение памяти: 16 МБ

Неспокойно сейчас на стапелях шестого дока межгалактического порта планеты Торна. Всего через месяц закончится реконструкция малого броненесущего корвета «Эния». И снова этому боевому кораблю и его доблестной команде предстоят тяжёлые бои за контроль над плутониевыми рудниками Сибелиуса. Работа не прекращается ни на секунду, лазерные сварочные аппараты работают круглые сутки. От непрерывной работы плавятся шарниры роботов-ремонтников. Но задержаться нельзя ни на секунду.

И вот в этой суматохе обнаруживается, что термозащитные панели корвета вновь требуют срочной обработки сульфидом тория. Известно, что на обработку одного квадратного метра панели требуется 1 нанограмм сульфида. Всего необходимо обработать N прямоугольных панелей размером A на B метров. Вам необходимо как можно скорее подсчитать, сколько всего сульфида необходимо на обработку всех панелей «Энии». И не забудьте, что панели требуют обработки с обеих сторон.

Исходные данные

Единственная строка содержит целые числа N ($1 \le N \le 100$), A ($1 \le A \le 100$), B ($1 \le B \le 100$).

Результат

Выведите вес необходимого для обработки сульфида тория в нанограммах.

Пример

исходные данные	результат
5 2 3	60

Автор задачи: Ден Расковалов

Источник задачи: IX Открытое командное соревнование школьников по программированию (13.03.2004)

1313. К вопросу о спорте

Ограничение времени: 0.5 секунды

Ограничение памяти: 16 МБ

Уральские медики очень озабочены состоянием здоровья молодежи. Как показали исследования, многие талантливые студенты вместо того, чтобы играть в футбол, кататься на коньках или велосипедах, занимаются олимпиадами по программированию. Так они это еще и спортивным программированием называют! Ну какой же это спорт, сидеть перед монитором и думать по пять часов в день! И ладно бы только на соревнованиях два раз в год, так ведь при подготовке к очередному чемпионату у них на сидение за компьютером уходит по несколько дней в неделю! И пусть бы это были разгильдяи какие-то, так ведь лучшие студенты, говорят, что и на мировом уровне результаты показывают приличные — жалко молодежь!

Чтобы отучить студентов от вредной привычки часами сидеть у компьютеров, уральские медики придумали принципиально новую разработку — монитор с диагональной разверткой! В скором будущем его будут вручать победителям и призерам чемпионатов Урала. В специально разработанном квадратном мониторе электронный луч должен проходить экран не по горизонталям, а по диагоналям. Поскольку длины диагоналей разные, нестандартные параметры мерцания и нелинейные искажения изображения быстро отучат оболтусов пялиться в экран. Разумеется, им ничего не останется делать, кроме как идти и заниматься спортом. Проблема в том, что большинство видеокарт подают на вход монитору нормальную развертку. Для успешного внедрения нового типа монитора нужна программа-адаптер, которая преобразует квадратное изображение в формат, требуемый для ввода в квадратный монитор. Программа должна быть надежной и быстрой, поэтому её разработкой будут заниматься самые лучшие программисты — участники чемпионата Урала по спортивному программированию.

Исходные данные

В первой строке записано целое число N ($1 \le N \le 100$) — количество пикселей на стороне квадратного монитора. Далее идут N строк, в каждой по N целых чисел в пределах от 1 до 100, разделенных пробелом. Это изображение, которое выводит обычная видеокарта (как видно, глубина цветности у нового монитора небольшая, все равно нормальному программисту больше ста цветов и не надо).

Результат

На выходе преобразователя должна получиться последовательность для ввода в новый монитор. Пиксели нумеруются от левого верхнего угла, по диагоналям, слева-направо и снизу-вверх. Подробнее объяснять тут долго и неинтересно, посмотрите на пример, дальше сами догадаетесь.

Пример

исходные данные	результат
4 1 3 6 10 2 5 9 13 4 8 12 15 7 11 14 16	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

Автор задачи: Идея — Станислав Васильев, подготовка — Станислав Васильев, Александр Клепинин

Источник задачи: VIII Командный студенческий чемпионат Урала по программированию. Екатеринбург, 11-16 марта 2004 г.

1327. Предохранители

Ограничение времени: 1.0 секунды

Ограничение памяти: 16 МБ

Янус Полуэктович (не помню уже, А или У) воспользовался машиной только один раз. Он принес с собой небольшую полупрозрачную коробочку, которую присоединил к "Алдану". Примерно через десять секунд работы с этой приставкой в машине полетели все предохранители, после чего Янус Полуэктович извинился, забрал свою коробочку и ушел.

Нравится Саше Привалову – молодому программисту НИИ Чародейства и Волшебства – в Соловце. Что может быть лучше, чем после института и скучной работы в Ленинграде, оказаться единственным программистом на такой машине как Алдан-3. Единственная проблема – это Янус Полуэктович.

В первый же рабочий день Янус задал Алдану задачу о свертке пространства заклинаний по четырем измерениям. ЭВМ задумалась, пошуршала лентами, поморгала лампочками и отключилась. Сгорел предохранитель. Поменять его даже для программиста – минутное дело. Все бы ничего, если бы Янус не был болезненно рассеян. Эта задача его интересовала настолько, что он помнил про сгоревший предохранитель только один день. На третий день Янус снова спалил Алдан. Четвертый прошел спокойно. Пятый снова подарил Алдану посредством Януса новый предохранитель.

Саша уже и думать об этом перестал, нужно только запастись заранее достаточным числом предохранителей.

Ваша цель помочь ему подготовить заказ для отдела снабжения. Заказ на предохранители составляется на некоторый промежуток времени от A-го Сашиного рабочего дня, до B-го включительно. Помогите Саше посчитать, сколько раз спалит Янус злосчастный предохранитель за этот промежуток времени.

Исходные данные

Первая строка содержит целое число A. Вторая строка содержит целое число B. $1 \le A \le B \le 10000$.

Результат

Выведите единственное число – количество предохранителей, которые спалит Янус в интервал времени с A-го рабочего дня по B-й.

Примеры

исходные данные	результат
1 5	3
100 200	50

Автор задачи: Ден Расковалов

Источник задачи: Десятый командный чемпионат школьников Свердловской области по программированию (16 октября 2004 года)

1409. Два бандита

Ограничение времени: 1.0 секунды

Ограничение памяти: 16 МБ

Бандиты Гарри и Ларри отдыхали на природе. Решив пострелять, они выставили на бревно несколько банок из-под пива (не больше 10). Гарри начал простреливать банки по порядку, начиная с самой левой, Ларри — с самой правой. В какой-то момент получилось так, что они одновременно прострелили одну и ту же последнюю банку.

Гарри возмутился и сказал, что Ларри должен ему кучу денег за то, что тот лишил его удовольствия прострелить несколько банок. В ответ Ларри сказал, что Гарри должен ему еще больше денег по тем же причинам. Они стали спорить кто кому сколько должен, но никто из них не помнил сколько банок было в начале, а искать простреленные банки по всей округе было неохота. Каждый из них помнил только, сколько банок прострелил он сам.

Определите по этим данным, сколько банок не прострелил Гарри и сколько банок не прострелил Ларри.

Исходные данные

В единственной строке записано 2 числа — количество банок, простреленных Гарри и Ларри соответственно.

Результат

Выведите 2 числа — количество банок, не простреленных Гарри и Ларри соответственно.

Пример

исходные данные	результат
4 7	6 3

Автор задачи: Ден Расковалов

Источник задачи: XII командный чемпионат школьников Свердловской области по программированию (15 октября 2005 года)

1446. Волшебная шляпа

Ограничение времени: 1.0 секунды

Ограничение памяти: 16 МБ

В начале каждого учебного года в Хогвартсе происходит исключительно важное событие — распределение юных колдунов по факультетам. Находчивые и изобретательные попадают в Слизерин, отважные — в Гриффиндор, трудолюбивые — в Хаффлпаф, а сообразительные — в Рэйвенкло. Распределение проходит следующим образом: когда называется фамилия очередного первокурсника, он выходит в центр зала и надевает знаменитую шляпу Гриффиндора. Шляпа, оценив обстановку в голове колдуна или ведьмы, громко выкрикивает факультет. Специальный эльф ведет протокол распределения. После распределения ему надо срочно составить список учеников по факультетам. Члены общества Против Угнетения Колдовских Народов-Изгоев предлагают вам помочь уставшему за много лет эльфу в этом нелегком деле.

Исходные данные

В первой стоке указано количество учеников-первогодников N ($1 \le N \le 1000$). В следующих $2N$ строках указаны их имена и факультеты, которые назвала шляпа. Имя ученика может содержать прописные и заглавные буквы латинского алфавита, пробелы и знак дефиса. Длина каждого имени не больше 200 символов.

Результат

Выведите список учеников по факультетам: сначала название факультета, потом двоеточие, в следующих сроках список учеников по одному в строке. Первым должен идти список Слизерина, потом — Хаффлпаф, Гриффиндор, и Рэйвенкло. Между списками факультетов надо оставлять пустую строку. Ученики в списках по факультетам должны идти в том же порядке, в котором их вызывали на примерку шляпы. Известно, что каждый год на каждый факультет попадает по крайней мере один ученик.

Пример

исходные данные	результат
7 Ivan Ivanov Gryffindor Mac Go Nagolo Hufflepuff Zlobeus Zlei Slytherin Um Bridge Slytherin Tatiana Henrihovna Grotter Ravenclaw Garry Potnyj Gryffindor Herr Mionag-Ranger Gryffindor	Slytherin: Zlobeus Zlei Um Bridge Hufflepuff: Mac Go Nagolo Gryffindor: Ivan Ivanov Garry Potnyj Herr Mionag-Ranger Ravenclaw: Tatiana Henrihovna Grotter

Автор задачи: Станислав Васильев
Источник задачи: X командный Чемпионат Урала по спортивному программированию, 24-25 марта 2006 года

1493. В одном шаге от счастья

Ограничение времени: 1.0 секунды

Ограничение памяти: 64 МБ

Вова купил билет в трамвае 13-го маршрута и сразу посчитал суммы первых трёх цифр и последних трёх цифр номера билета (номер у билета шестизначный). Оказалось, что суммы отличаются ровно на единицу. «Я в одном шаге от счастья», — подумал Вова, — «или предыдущий или следующий билет точно счастливый». Прав ли он?

Исходные данные

В единственной строке записан номер билета. Номер состоит ровно из шести цифр, среди которых могут быть и нули. Гарантируется, что Вова умеет считать, то есть суммы первых трёх цифр и последних трёх цифр отличаются ровно на единицу.

Результат

Выведите «Yes», если Вова прав, и «No», если нет.

Примеры

исходные данные	результат
715068	Yes
445219	No
012200	Yes

Подсказка

Трамвайный билет называется счастливым, если сумма его первых трёх цифр равна сумме его последних трёх цифр.

Автор задачи: Владимир Яковлев

Источник задачи: XIII командный чемпионат школьников Свердловской области по программированию (14 октября 2006 года)

1567. SMS-спам

Ограничение времени: 1.0 секунды

Ограничение памяти: 64 МБ

Студент Петя решил открыть свой бизнес — он предлагает арендаторам офисов в только что открывшемся небоскрёбе Призма услуги SMS-рекламы. Услуга заключается в том, что заказчик придумывает речёвку про свою фирму, а Петя со своего сотового телефона рассылает ее как SMS-сообщение тысячам жителей Екатеринбурга, используя купленную у пиратов базу телефонных номеров горожан. Стоимость каждой речёвки определяется как сумма стоимостей каждого символа в ней, а стоимость символов Петя определяет по незатейливой схеме: за каждое свое нажатие на кнопку телефона он берёт по 1 рублю.

Петин телефон не поддерживает T9 и имеет только английскую раскладку:

1	2 abc	3 def
4 ghi	5 jkl	6 mno
7 pqr	8 stu	9 vwx
	0 yz	# .,!

Символом «_» в таблице обозначен пробел. Например, чтобы набрать букву «a», надо нажать один раз на «1», букву «k» — два раза на «4», «!» — три раза на «0» и т.д.

Чтобы узнать, какой гонорар он должен получить за рекламную речёвку, которую в данный момент рассылает, Пете необходимо посчитать её стоимость по этому простому алгоритму. А поскольку Петя очень занятой и вообще не умеет считать, так как учится на философском факультете, вы, как его самый лучший друг, готовы ему помочь.

Исходные данные

В единственной строке записана рекламная речёвка, состоящая из слов, пробелов, запятых, точек и восклицательных знаков. Все слова состоят из строчных английских букв. В речёвке не более 1000 символов.

Результат

Выведите единственное число — стоимость речёвки в петином понимании.

Пример

исходные данные	результат
pokupaite gvozdi tolko v kompanii gvozdederov i tovarischi!	114

Автор задачи: Денис Мусин

Источник задачи: XII Чемпионат УрГУ по программированию, 6 октября, 2007

1581. Работа в команде

Ограничение времени: 1.0 секунды

Ограничение памяти: 64 МБ

Вася и Петя решили принять участие в командной олимпиаде по информатике. Но они слушали рассказы ветеранов спортивного программирования, а поэтому знали, что, помимо умения хорошо программировать, очень важна сыгранность участников. Поэтому они решили отрабатывать навыки работы в команде.

Вася написал на бумажке последовательность натуральных чисел и стал диктовать её Пете. Причём, для краткости, он диктует её так: сначала говорит количество подряд идущих одинаковых чисел, а затем говорит, что это за числа. Например, последовательность «1 1 2 3 3 3 10 10» он продиктует как «*две единицы, одна двойка, три тройки, две десятки*». Петя же, тоже для краткости, записывает вместо слов числа, которые говорит Вася. В указанном примере Петя запишет на бумажку: «2 1 1 2 3 3 2 10».

Отработав этот навык, друзья решили ещё потренироваться в программировании и научить компьютер преобразовывать последовательность Васи в последовательность Пети.

Исходные данные

В первой строке находится целое число N — количество чисел, записанных Васей ($1 \le N \le 1000$). Во второй строке через пробел записаны эти числа. Все числа целые, положительные и не превосходят 10.

Результат

В единственной строке выведите через пробел числа, которые должен был бы записать на бумажке Петя.

Пример

исходные данные	результат
8 1 1 2 3 3 3 10 10	2 1 1 2 3 3 2 10

Автор задачи: Алексей Самсонов

Источник задачи: XIV Открытое командное первенство школьников Свердловской области по программированию

1585. Пингвины

Ограничение времени: 1.0 секунды

Ограничение памяти: 64 МБ

Программист Денис с детства мечтал побывать в Антарктиде, но почему-то регулярных рейсов туда нет. Поэтому Денис все лето изучал Антарктиду с помощью соседнего кинотеатра. Теперь он знает, что в Антарктиде водится несколько видов пингвинов:

- Императорские пингвины (Emperor Penguins) — любители петь;
- Малые пингвины (Little Penguins) — любители потанцевать;
- Пингвины Макарони (Macaroni Penguins) — любители сёрфинга.

К сожалению, в мультфильмах не было сказано, какой вид пингвинов самый многочисленный. Денис решил выяснить это: он посмотрел эти мультфильмы еще раз, и каждый раз, когда видел пингвина, записывал в блокнот название его вида. Сейчас Денис дал вам блокнот с просьбой выяснить, какой вид пингвинов самый многочисленный.

Исходные данные

В первой строке записано целое число n — количество записей в блокноте ($1 \le n \le 1000$). В каждой из следующих nстрок записано по одному виду пингвинов. Среди видов встречаются только «Emperor Penguin», «Little Penguin» и «Macaroni Penguin».

Результат

Выведите самый популярный вид пингвинов. Гарантируется, что такой вид только один.

Пример

исходные данные	результат
7 Emperor Penguin Macaroni Penguin Little Penguin Emperor Penguin Macaroni Penguin Macaroni Penguin Little Penguin	Macaroni Penguin

Автор задачи: Владимир Яковлев

Источник задачи: ACM ICPC 2007–2008. NEERC. Восточный подрегион. Екатеринбург, 27 октября 2007 г.

1601. АнтиКАПС

Ограничение времени: 0.5 секунды

Ограничение памяти: 64 МБ

У блондинки Анжелы новое увлечение — интернет-чаты. Она, как любая уважающая себя блондинка, привыкла писать свои сообщения в верхнем регистре. Вам, модератору чата, в котором любит «зависать» Анжела, надоело уже выносить ей предупреждения. Вы решили сделать простенький антикапс-корректор, который бы приводил сообщения Анжелы к читабельному виду.
Правила исправлений очень просты:

1. Предложения в сообщении состоят из слов, пробелов и знаков препинания.
2. Слова состоят из букв латинского алфавита.
3. Предложения оканчиваются точкой, восклицательным или вопросительным знаком.
4. Первое слово в предложении должно начинаться с заглавной буквы, все остальные буквы в предложении должны быть строчными.

Исходные данные

Входные данные содержат сообщение Анжелы, которое состоит из латинских букв в верхнем регистре, пробелов, переводов строк и знаков препинания: точек, запятых, тире, двоеточий, восклицательных и вопросительных знаков. Общая длина сообщения не превосходит 10000 символов.

Результат

Выведите исправленное сообщение Анжелы.

Пример

исходные данные	результат
HI THERE! HOW DID YOU KNOW I AM A BLONDE?	Hi there! How did you know i am a blonde?

Автор задачи: Денис Мусин

Источник задачи: Девятое открытое личное первенство УрГУ (1 марта 2008)

1639. Шоколад 2

Ограничение времени: 1.0 секунды

Ограничение памяти: 64 МБ

Малыш обиделся на Карлсона. Тот без спроса съел в доме всё сладкое и даже к соседям залезть успел. И вновь — только родители принесли домой большую шоколадку размера $m \times n$, а Карлсон тут как тут! Малыш твёрдо решил, что на этот раз ни кусочка шоколада этому обжоре не достанется. Известно, что кроме еды у Карлсона есть ещё одна слабость: он любит играть в азартные игры и иногда ради возможности сыграть даже готов пожертвовать своим желудком. Поэтому Малыш предложил Карлсону следующую игру: за один ход игрок должен взять один из кусков шоколада и разломить его на два куска вдоль линий, разделяющих дольки. Игроки ходят по очереди; проигрывает тот, кто не может сделать ход, так как остались только куски шоколада размера 1×1. Победитель забирает весь шоколад себе.

Однако Карлсона так просто не проведёшь! Он быстро смекнул, должен ли он ходить первым или уступить право первого хода Малышу, чтобы гарантированно выиграть. А Вы бы смогли определить это на его месте?

Исходные данные

В единственной строке через пробел записаны целые числа m и n ($1 \leq m, n \leq 50$) — длина и ширина шоколадки в дольках.

Результат

Если для того, чтобы выиграть, Карлсону нужно ходить первым, выведите в единственной строке «[:=[first]», иначе выведите «[second]=:]».

Примеры

исходные данные	результат
2 4	[:=[first]
1 3	[second]=:]

Автор задачи: Александр Пронченков (подготовка — Иван Бурмистров)

Источник задачи: Осеннее первенство школьников 2008

1642. Одномерный лабиринт

Ограничение времени: 1.0 секунды

Ограничение памяти: 64 МБ

В одномерном государстве жили одномерные люди. И всё остальное в этом государстве было тоже одномерное. Всё в их одномерном мире было просто и понятно: всего одна ось координат и всего два направления движения — вперёд и назад. Но даже в одномерном мире есть проблемы. Например, как найти выход из лабиринта? Это нам одномерный лабиринт кажется по меньшей мере странным, а для одномерных людей поиск выхода из лабиринта — очень сложная и насущная задача. Решают её одномерные люди следующим образом.

Одномерный человек выбирает направление движения: вперёд (в сторону увеличения координаты) или назад (в сторону её уменьшения) и идёт в выбранном направлении. Если он находит точку выхода, то сразу покидает лабиринт. Если же он наталкивается на препятствие, то разворачивается и идёт в обратном направлении.

Почувствуйте, как нелегко жить в одномерном мире: попробуйте реализовать функцию, определяющую расстояние, которое пройдёт одномерный человек, прежде чем найдёт выход, в зависимости от выбора начального направления движения.

Исходные данные

В первой строке через пробел записаны два целых числа n и x — количество препятствий и координата точки выхода, соответственно ($0 \le n \le 100$). Одномерный человек располагается в начале координат. Во второй строке через пробел записаны n различных целых чисел — координаты препятствий. Все координаты, включая x, не равны нулю и не превосходят по модулю 1000. Гарантируется, что ни одно препятствие не расположено в точке выхода. Гарантируется, что вне зависимости от выбора начального направления движения одномерный житель рано или поздно найдёт выход или натолкнётся на препятствие.

Результат

Выведите через пробел два числа — расстояние, которое пройдёт одномерный человек до выхода из лабиринта, если сначала пойдёт вперёд и если сначала он пойдёт назад, соответственно. Если из-за препятствий дойти до выхода невозможно, выведите «Impossible».

Примеры

исходные данные	результат
3 -2 -10 -4 2	6 2
3 -2 10 -1 2	Impossible

Автор задачи: Даниил Айзенштейн

Источник задачи: Осеннее первенство школьников 2008

1683. Холодильник

Ограничение времени: 0.5 секунды

Ограничение памяти: 64 МБ

Лежал как-то раз Иванушка на печи да думу думал. И придумал: решил холодильник купить. Сказано — сделано. Купил он холодильник и поставил рядом с печкой. Да только шатается холодильник, ведь пол в избе неровный. Иванушка не растерялся и решил что-нибудь подложить под одну из ножек холодильника, чтобы тот принял устойчивое положение.

Недолго думая, решил Иванушка использовать для этого свёрнутую бумажку. Он взял бумажную полоску шириной 1 сантиметр и длиной n сантиметров и теперь хочет свернуть её так, чтобы получился бумажный квадрат со стороной 1 сантиметр и толщиной в n слоёв (если сделать больше или меньше, чем n слоёв, холодильник по-прежнему будет шататься). Иванушка сгибает бумажную полоску по следующему алгоритму: от левого края полоски отмеряет линейкой целое число сантиметров и загибает левый край направо (в результате левый край полоски смещается на отмеренное число сантиметров). Затем он вновь отмеряет некоторое количество сантиметров от нового левого края и опять загибает его направо, и так далее, пока в результате не получится полоска длиной в 1 сантиметр.

Определите, какое наименьшее количество сгибов бумаги должен сделать Иванушка.

Исходные данные

В единственной строке записано целое число n ($1 \le n \le 10^9$).

Результат

В первой строке выведите минимальное количество сгибов бумаги, необходимое для получения нужного количества слоёв. Во второй строке выведите через пробел последовательность целых чисел — длины в сантиметрах, которые Иванушка отмерял перед каждым из сгибов.

Пример

исходные данные	результат
12	4 5 3 2 1

Автор задачи: Станислав Васильев

Источник задачи: Открытое личное первенство УрГУ 2009 (28 февраля 2009)

1197. Один в поле воин

Ограничение времени: 1.0 секунды

Ограничение памяти: 16 МБ

Условие этой задачи очень простое: вам всего лишь надо определить, сколько клеток находится под боем шахматного коня, одиноко стоящего на шахматной доске. На всякий случай напомним, что конь ходит буквой «Г» — на две клетки по горизонтали или вертикали в любом направлении, и потом на одну клетку в направлении, перпендикулярном первоначальному.

Исходные данные

В первой строке находится единственное число N, $1 \le N \le 100$ — количество тестов. В каждой из последующих Nстрок содержится очередной тест: два символа (маленькая латинская буква от 'a' до 'h' и цифра от 1 до 8) — стандартное шахматное обозначение клетки, на которой стоит конь. При этом буква обозначает вертикаль, а цифра — горизонталь.

Результат

Выведите N строк: в каждой из них должно находиться единственное число — количество клеток шахматной доски, находящихся под боем коня.

Пример

исходные данные	результат
3 a1 d4 g6	2 8 6

Автор задачи: фольклор

Источник задачи: V командное первенство школьников по программированию (2 марта 2002)

1826 Минное поле

Ограничение времени: 0.5 секунды

Ограничение памяти: 64 МБ

Во время выполнения задания разведгруппа из n человек переходит минные заграждения противника. Поскольку миноискатель у группы всего один, перемещаться решено следующим образом: в сторону противника переходят парой, затем один человек возвращается, чтобы принести оставшейся группе миноискатель.

Каждый человек переходит минные заграждения со своей скоростью. Скорость пары определяется скоростью более медленного её члена.

Найдите минимальное время, за которое вся группа сможет преодолеть минные заграждения.

Исходные данные

В первой строке записано целое число n ($2 \le n \le 100$). В i-й из следующих n строк записано положительное целое число, не превосходящее 600 — время, необходимое i-му члену группы на преодоление минных заграждений.

Результат

Выведите минимальное суммарное время, которое потребуется разведгруппе для преодоления заграждений противника.

Пример

исходные данные	результат
4 1 10 5 2	17

Автор задачи: Дмитрий Теряев

Источник задачи: XII открытое личное первенство УрГУ (19 марта 2011)

1877. Велосипедные коды

Ограничение времени: 0.5 секунды

Ограничение памяти: 64 МБ

У Дена есть два четырёхзначных кодовых замка для велосипеда. Каждый вечер он ставит велосипед на сигнализацию и пристёгивает к специальной стойке одним из замков. Ден никогда не использует один и тот же замок два вечера подряд. В некоторую ночь злоумышленник попытался с помощью кода 0000 открыть висящий на велосипеде замок. Сработала сигнализация, и вор поспешил скрыться. На следующую ночь он решил попробовать код 0001, затем 0002 и так далее в порядке возрастания номера.

Известно, что Ден не меняет кодов и в ночь, когда вор пришёл впервые, велосипед был пристёгнут первым замком.

Исходные данные

В первой строке записан код, установленный на первом замке, во второй строке — код, установленный на втором замке. Оба кода — строки длины 4, состоящие из цифр от 0 до 9.

Результат

Выведите «yes», если злоумышленник рано или поздно взломает замок, и «no» в противном случае.

Примеры

исходные данные	результат
0001 0000	no
0002 0001	yes

Автор задачи: Денис Мухаметьянов

Источник задачи: Уральская региональная командная олимпиада по программированию 2011

1880. Собственные числа Psych Up

Ограничение времени: 0.5 секунды

Ограничение памяти: 64 МБ

Шёл очередной контест Петрозаводских сборов. Игроки команды Psych Up быстро нашли простую задачу, и Федя сел за компьютер. Через пять минут решение было готово. Не тратя времени на тестирование, Федя отправил его на проверку и через несколько секунд получил вердикт Time Limit Exceeded.

Федя скомкал условие задачи и вышел из класса, хлопнув дверью. Что-то шло не так, нужно было развеяться. По пути в туалет он услышал разговор авторов контеста. Паша обсуждал со своим другом решение той самой задачи. Федя смог разобрать из их разговора только слова «собственные числа».

Федя подумал и решил, что у него, конечно же, есть собственные числа. Например, дата рождения, номер квартиры, оценка на последнем экзамене, количество поездок на соревнования. Но ведь контест командный. А что такое собственные числа команды? Естественно, что число является собственным для команды, если оно собственное для каждого из её участников. С такими радостными мыслями Федя направился назад в аудиторию.

Исходные данные

Входные данные состоят из трёх блоков по две строки. Первая строка каждого блока содержит целое число n —количество собственных чисел очередного игрока ($1 \le n \le 4\,000$). Во второй строке блока записано n целых различных чисел в порядке возрастания — собственные числа очередного игрока. Все собственные числа — целые, положительные и не превосходят 10^9.

Результат

Выведите количество собственных чисел команды Psych Up.

Пример

исходные данные	результат
5 13 20 22 43 146 4 13 22 43 146 5 13 43 67 89 146	3

Автор задачи: Денис Мухаметьянов

Источник задачи: Уральская региональная командная олимпиада по программированию 2011

Задачи для практических занятий с сайта acm.timus.ru				
	Номер	**Название**	**Авторы**	**Сложность**
1	1000	A+B Problem	43013	18
2	1001	Обратный корень	9461	28
3	1785	Трудности локализации	2626	37
4	1409	Два бандита	8171	40
5	1820	Уральские бифштексы	1815	44
6	1068	Сумма	14425	46
7	1293	Эния	8807	47
8	1787	Поворот на МЕГУ	1870	51
9	1639	Шоколад 2	3074	58
10	1025	Демократия в опасности	10943	64
11	1264	Трудовые будни	6948	64
12	1877	Велосипедные коды	894	64
13	1567	SMS-спам	3279	69
14	1581	Работа в команде	3292	69
15	1083	Факториалы!!!	8897	73
16	1585	Пингвины	3054	73
17	1880	Собственные числа Psych Up	779	73
18	1005	Куча камней	9390	75
19	1313	К вопросу о спорте	5497	75
20	1545	Иероглифы	3216	77
21	1493	В одном шаге от счастья	3521	78