Задания заключительного тура

Всероссийского конкурса научных работ школьников «Юниор» 2014-2015 учебного года

секция информатики, 10 класс

Задача A. «Опечатка»

Максимальное время работы на одном тесте: *2 секунды*
Максимальный объем используемой памяти: *64 мегабайта*

Начинающий хакер Петя очень занят разработкой плана по взлому Самого Главного Сервера Интернета. Петя давно бы уже разработал этот план, если бы ему не приходилось постоянно отвлекаться на всякие глупости – на то, чтобы убраться в своей комнате, почистить зубы или – вот как сейчас – на школьные домашние задания.

Сегодняшнее домашнее задание Пети по информатике состоит в том, чтобы написать программу, проверяющую, возможно ли для трех введенных целых чисел a, b, c построить невырожденный треугольник с соответствующими длинами сторон. Вспомнив, что учитель информатики говорил что-то про использование неравенства треугольника, Петя быстренько написал свой вариант программы для решения этой задачи. К сожалению, программа Пети возвращает неправильный ответ для второго теста из условия задачи. Поскольку Пете требуется незамедлительно вернуться к планированию взлома Самого Главного Сервера, он просит Вас доделать вместо него его домашнее задание.

Ниже приведен листинг программы, написанной Петей.

```
Program Triangle;
var
  a,b,c:integer;
begin
  readln(a,b,c);
  if ((a-b+c) > 0) and
      ((a-c+b) > 0) and
      ((a-b-c) > 0)
  then
    writeln('YES')
  else
    writeln('NO');
end.
```

Формат входных данных

В единственной строке входных данных содержатся три целых числа a,b,c, разделенных пробелами: $0 < a,b,c \leq 1000$

Формат результата

В единственную строку результата вывести "YES" (без кавычек, заглавными буквами), если из отрезков с длинами a,b,c можно составить невырожденный треугольник, и "NO" (без кавычек, заглавными буквами) в противном случае.

Пример входных данных и результата:

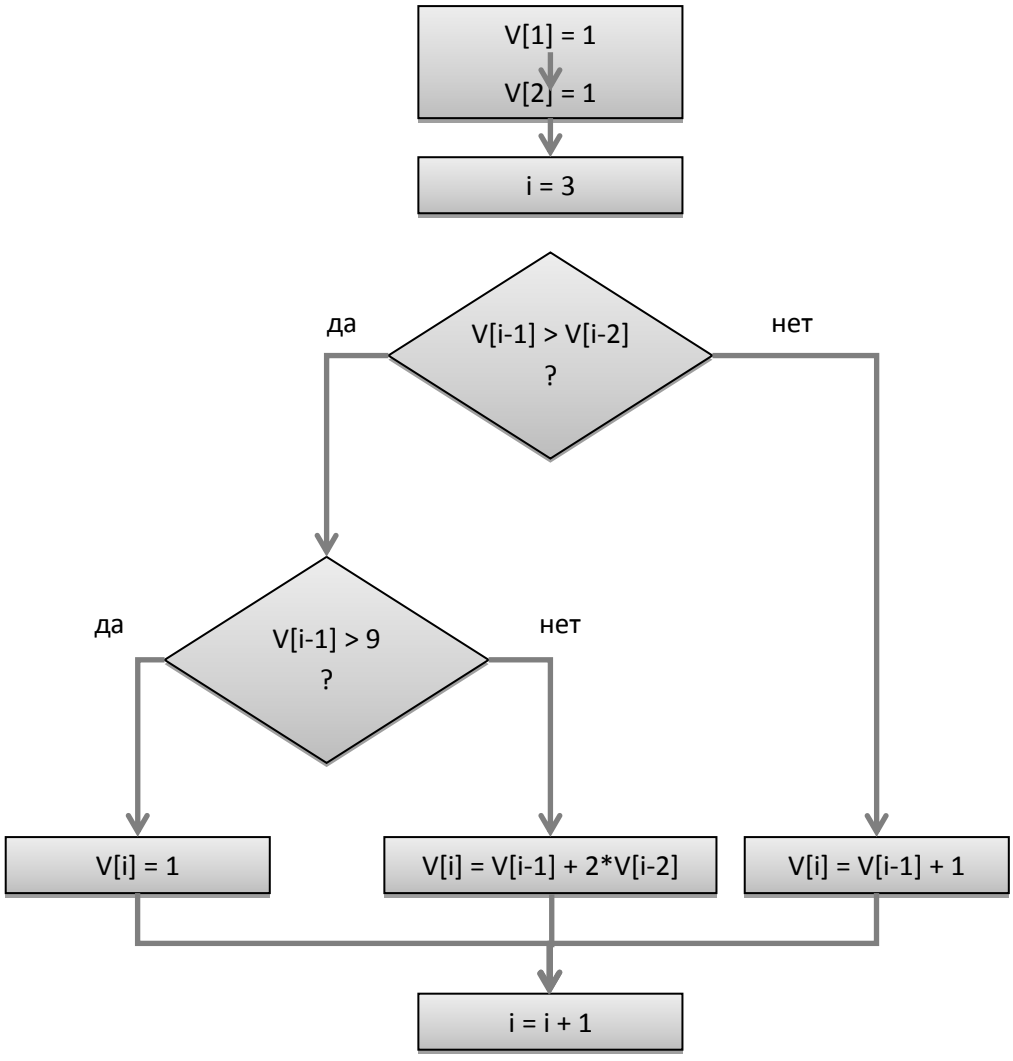
standard input	standard output
3 1 5	NO
4 2 3	YES

Задача B. «Числа Васиаччи»

Максимальное время работы на одном тесте: *2 секунды*
Максимальный объем используемой памяти: *64 мегабайта*

Начинающему хакеру Васе не даёт покоя слава Великого Математика древности Фибоначчи, про которого вчера рассказывал школьный учитель. Вася решил, что стать таким же Великим Математиком– гораздо круче и почётнее, чем Великим Хакером. Тем более, что в последнее время у Васи возникли сложности с разработкой его собственного плана по взлому Самого Главного Сервера (только не рассказывайте об этом Пете!)

Таким образом, Вася твердо решил стать Великим Математиком. Для этого Васе нужна его собственная последовательность чисел, названная в его честь. Чтобы известность и слава Великого Математика пришли как можно быстрее, Вася собирается сделать свою последовательность как можно хитрее. Всю вчерашнюю ночь он не мог заснуть и, в конце концов, нарисовал следующую схему, по которой должны формироваться числа его последовательности:



Теперь Вася хочет научиться узнавать значение произвольного элемента своей последовательности V_n по заданному номеру элемента n. К несчастью, гораздо больше Вася хочет немного поспать. Поэтому, взяв с Вас обещание не раскрывать секрет его последовательности, Вася показал Вам свою схему и попросил написать программу, которая позволит по заданному номеру элемента n вычислять значение элемента V[n] Васиной последовательности.

Не забудьте, что, хотя Вася и нарисовал в своей блок-схеме бесконечный цикл, Ваша программа должна вернуть ответ за ограниченное время.

Формат входных данных

В единственной строке входных данных содержится единственное целое число n, такое что: $1 \leq n \leq 100000$

Формат результата

В единственную строку результата вывести единственное целое число – искомое значение n-го элемента последовательности.

Пример входных данных и результата:

standard input	standard output
1	1
5	8

Задача D. «Рыба»

Максимальное время работы на одном тесте: *2 секунды*
Максимальный объем используемой памяти: *64 мегабайта*

Разбираясь на чердаке, Альберт и Альфред нашли в одной из пыльных коробок набор костяшек для игры в домино. Уборка на чердаке была мгновенно забыта, поскольку выяснить, кто из братьев лучше играет в эту давно забытую игру – несомненно, более важная задача.

В то время как Альберт предпочитает играть, полностью полагаясь на интуицию, Альфред предпочитает строгий математический подход. Однако Альфред не может слишком отвлекаться на математические вычисления, поскольку беспокоится, что его брат будет жульничать. Поэтому Альфред просит Вас помочь ему с оценкой ситуации на поле, пока он сам будет в оба глаза следить за Альбертом.

Программа, которую Вас просит написать Альфред, должна оценивать по текущей ситуации на поле, кто в данный момент времени выигрывает в партии. Согласно правилам, которые используют братья, побеждает тот игрок, у которого сумма очков на руках меньше, чем у его соперника. Сумма очков подсчитывается, как сумма точек всех костей, остающихся на руках. При этом если на руках игрока остается единственная кость «0-0», то она оценивается в 10 очков.

Напомним, что набор костей домино состоит из 28 костей, на каждой из двух половинок которых нанесено от 0 до 6 точек. Полный набор костей содержит по одному разу все возможные сочетания значений на двух половинках.

Формат входных данных

В первой строке входных данных содержатся два целых числа: A, B: количество костей соответственно на руках Альфреда (A) и на столе (B): $0 \le A,B \le 28$. В следующих A строках описываются кости на руках Альфреда. Каждая кость описывается двумя целыми числами в одной строке: C_1, C_2: $0 \le C_1,C_2 \le 6$ соответствующими количеству точек на каждой из половинок кости. Далее в следующих B строках в аналогичном формате описываются кости на столе. Все остальные кости находятся на руках Альберта. Гарантируется, что все кости во входном наборе данных различны.

Формат результата

В единственную строку результата вывести единственное целое число – разницу в количестве очков на руках Альберта и Альфреда. Если выигрывает Альфред, то число должно быть положительно, если Альберт – отрицательно. В случае если количество очков на руках одинаково, в качестве результата должно выводиться число 0.

Пример входных данных и результата:

standard input	standard output
0 0	168
15 3 2 5 0 4 5 5 1 1 5 0 2 4 1 4 4 6 1 2 2 0 6 2 3 0 2 2	-14

3 2 5 1 5 3 3 3 4 4	

Задача E. «Салочки»

Максимальное время работы на одном тесте: *2 секунды*
Максимальный объем используемой памяти: *64 мегабайта*

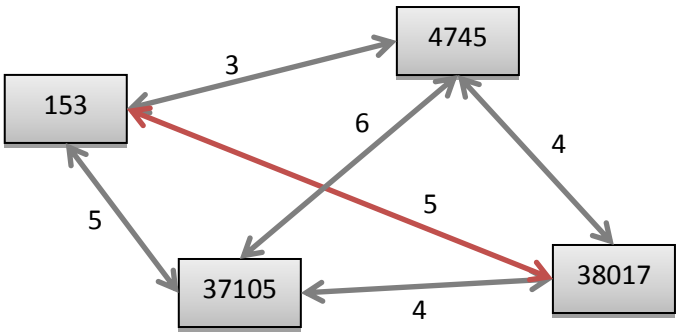
Алиса (ALICE - Artificial Lively Intelligent Creature Entity) и Боб (BOb - Binary Object) играют в салочки. Поле, на котором они играют, представляет собой множество регистров оперативной памяти, соединенных переходами. Каждый регистр имеет номер от 0 до 65535. Длина перехода между двумя регистрами равна количеству различающихся бит в двоичных представлениях номеров этих регистров. Например, расстояние между регистрами #153 и #38017 равно 5:

153_{10}= **0**00**0** 0**0**00 100**1** **1**001$_2$
38017_{10}= **1**00**1** 0**1**00 100**0** **0**001$_2$

Алиса знает, что Боб умеет перемещаться по памяти на долю такта быстрее, чем она сама. Поэтому единственный способ выиграть для неё состоит в том, чтобы заблокировать прямой переход между регистрами после её собственного перемещения. Теперь Алиса хочет знать, существует ли для данной пары регистров на поле какой-либо другой путь, суммарная длина которого не превышает длину прямого перемещения.

Например, пусть поле для игры состоит из 4-х регистров: #153, #4745, #37105, #38017

153_{10}= 0000 0000 1001 1001$_2$
4745_{10}=0001 0010 1000 1001$_2$
37105_{10} = 1001 0000 1111 0001$_2$
38017_{10} = 1001 0100 1000 0001$_2$



Из рисунка видно, что, если заблокировать прямой переход между регистрами #153 и #38017, то длина кратчайшего пути между этими регистрами (#153 => #4745 => #38017) составит 7, то есть, больше, чем длина прямого перемещения (5).

Формат входных данных

В первой строке входных данных содержится единственное целое число N: $3 \le N \le 64$ – количество регистров памяти, из которых состоит поле. В следующих N строках входных данных указывается по одному целому числу R[i] – номер соответствующего регистра памяти, $0 \le R[i] \le 65535$. Гарантируется, что все номера регистров различны. В последней строке входного файла указываются через пробел два различных числа A,B – номера регистров, между которыми перемещается Алиса. Гарантируется, что регистры A и B принадлежат игровому полю.

Формат результата

Если существует еще хотя бы один путь между регистрами A и B, суммарная длина которого равна длине прямого перехода, в единственную строку результата вывести одно целое число – эту длину. Если пути, удовлетворяющего условию, не существует, вывести в качестве результата -1.

Пример входных данных и результата:

standard input	standard output

4 4745 153 37105 38017 153 38017	-1

Задача G. «Устный счет»

Максимальное время работы на одном тесте: *2 секунды*
Максимальный объем используемой памяти: *64 мегабайта*

Алиса (ALICE - Artificial Lively Intelligent Creature Entity) и Боб (BOb - Binary Object) тренируются в устном счете. Алиса придумывает числа, а Боб их складывает или вычитает. Чтобы Бобу не было слишком скучно, Алиса придумывает очень длинные числа…

Однако Алиса настолько занята придумыванием чисел для Боба, что у неё совсем не остаётся времени проверить, правильно ли Боб вычислил результат. Поэтому Алиса просит Вас помочь ей и написать программу, которая будет проверять вычисления Боба.

Боб выполняет вычисления по следующему правилам:

- в начальный момент в памяти Боба в качестве текущего результата записано число 0
- Боб последовательно считывает по одному числу из придуманных Алисой и либо прибавляет его к текущему результату, либо вычитает из текущего результата это число
- Боб прибавляет новое число к текущему результату тогда и только тогда, когда новое число больше, чем текущий результат. В противном случае Боб вычитает новое число из текущего результата.

Формат входных данных

В первой строке входных данных содержится единственное значение N: $1 \le N \le 128$ – количество чисел, придуманных Алисой. В следующих N строках входного файла перечислены эти числа. В каждой строке записано по одному целому неотрицательному числу. Гарантируется, что каждое число содержит не более 255 знаков.

Формат результата

В единственную строку результата вывести единственное число – результат вычислений, который должен получиться у Боба.

Пример входных данных и результата:

standard input	standard output
3 5 7 11	1
2 123456789123456789 123456789123456789	1358024679246913578

Ответы и решения

Задача A. «Опечатка»

Возможность для трех заданных чисел a, b, c построить невырожденный треугольник с соответствующими длинами сторон проверяется при помощи системы неравенств треугольника, выписанных для каждой из трех его сторон. Для заданных чисел должны выполняться следующие условия:

$a < b + c$
$b < c + a$
$c < a + b$

Несложно заметить, что в приведенном листинге программы Пети одно из этих неравенств записано неверно:

```
if ((a-b+c) > 0) and
     ((a-c+b) > 0) and
     ((a-b-c) > 0)
then ...
```

Правильный вариант этого фрагмента кода может выглядеть, например, следующим образом:

```
if ((a-b+c) > 0) and
     ((a-c+b) > 0) and
     ((a-b-c) < 0)
then ...
```

Задача B. «Числа Васиаччи»

По приведенной блок-схеме, согласно которой Вася формирует числа своей последовательности, видно, что значение каждого следующего числа V[i] последовательности зависит не более чем от двух предыдущих: V[i-1], V[i-2]. С учетом того, что формулы определения следующего числа последовательности очень просты, а ограничение на максимальный номер числа, которое нужно посчитать, мало ($n \le 100000$), можно реализовать вычисление нужно числа через обычный цикл, внутри которого аккуратно переписать на соответствующем языке программирования логику, описанную в блок-схеме. Отдельно нужно заметить, что ветка блок-схемы «если V[i-1] > 9 то V[i]=1» ограничивает рост абсолютных значений элементов последовательности, поэтому даже элементы с большими номерами будут умещаться в стандартные целочисленные типы.

Впрочем, решать данную задачу можно и без использования цикла и вычисления всех элементов последовательности. Та же самая ветка блок-схемы, ограничивающая рост элементов: «если V[i-1] > 9 то V[i]=1» обеспечивает, что последовательность является циклической, причем период последовательности значительно меньше, чем порядок максимально возможного значения n: $n \le 100000$.

Для того чтобы выделить период этой последовательности, достаточно построить несколько первых её элементов:

1, **1, 2, 4, 8, 16**, 1, 2, 4, 8, 16, 1, 2, 4, …

Начиная со второго элемента, последовательность является циклической с периодом в 5 элементов, а сам циклический фрагмент состоит из значений 1, 2, 4, 8, 16. Выявив данную закономерность в структуре последовательности, значение любого элемента последовательности становится легко определить аналитически по номеру элемента.

Задача D. «Рыба»

В этой задаче требуется на основе входных данных и правил начисления очков, приведенных в условиях, посчитать разницу в сумме очков на руках двух игроков.

Входные данные полностью описывают игровую ситуацию. Тест №1, приведенный в условии задачи, позволяет сразу узнать или проверить общее количество точек на всех костях домино в наборе. Зная это число, а также посчитав по входным данным количество точек на костях, находящихся на руке одного из игроков, и на столе, можно определить количество точек на костях второго игрока. Аналогично определяется количество костей на руках обоих игроков. Наконец, количество очков на руке каждого игрока равно количеству точек на костях в его руке, за исключением единственного случая – если у игрока на руке только 1 кость, и 0 точек (то есть единственная кость является дублем «0-0») – по условию, эта ситуация оценивается в 10 очков.

В наиболее простом виде, программная реализация данной задачи не предполагает использования массивов.

Задача E. «Салочки»

Согласно введенной в условии задачи метрике, расстояние между двумя регистрами A и B равно количеству различающихся бит в двоичных представлениях номеров этих регистров. Рассмотрим пример из условия (A=153, B=38017). Для некоторого регистра C суммарное расстояние AC+CB будет равно AC тогда и только тогда, когда все значения всех бит, совпадающих для A и B, те же самые и в двоичном представлении C:

153_{10}= 0000 0000 1001 1001$_2$
C = X00X 0X00 100X X001
38017_{10}= 1001 0100 1000 0001$_2$

где на месте X может быть любое значение (0 или 1).
Если же хотя бы один из бит, совпадающих в A и B, для C отличается, то расстояние AC+CB будет всегда больше AC:

153_{10}= 0000 0000 1001 1001$_2$
C′ = X00X 0X00 110X X001
38017_{10}= 1001 0100 1000 0001$_2$

Для решения задачи достаточно проверить для всех чисел во входных данных, не совпадающих с A и B, выполняется ли это условие. Для проверки условия можно явно строить двоичное представление чисел, используя операции деления нацело и остатка от деления (в данном случае на 2). Однако более просто данная задача решается с использованием битовой арифметики. В терминах битовой арифметики, регистр C удовлетворяет требованиям задачи, тогда и только тогда, когда верно следующее равенство:

(A xor B) = ((A xor C) or (C xor B))

Задача G. «Устный счет»

Для решения задачи требуется реализовать несколько простейших операций «длинной» арифметики: чтение длинного числа, сложение и вычитание двух длинных чисел, сравнение двух длинных чисел, печать длинного числа. Основная идея «длинной» арифметики состоит в том, что каждое длинное число сохраняется в массиве, каждый элемент которого содержит один или (чаще всего) несколько разрядов числа в его десятичном представлении (привязка к десятичному представлению сохраняется, прежде всего, для удобства реализации ввода и вывода «длинных» чисел). Более подробно о реализации длинной арифметики можно узнать, например, здесь.