Задания заключительного тура

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

секция информатики, 11 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: $0 < a,b,c \le 1000$

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 C. «Эврика!»

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

Вот оно! И как только Петя не догадался об этом раньше?! Все дело – в прямоугольных треугольниках!

Эта гениальная мысль пришла в голову Пете совсем внезапно, когда уже казалось, что его план по взлому Самого Главного Сервера снова зашел в тупик. И вдруг – вот оно, решение! Действительно, попробуйте вспомнить – где в дикой природе вы встречали ровные прямые углы? Нет, отдельные частные случаи в расчет брать не будем – наверняка, если измерить очень внимательно, то окажется, что углы там тоже не совсем прямые. А теперь посмотрите вокруг, на то, что человек создал искусственно. Видите? Сплошные прямые углы! Один за другим! Везде, все состоит из прямых углов!

Петя уверен, что ему удалось обнаружить Фундаментальный Критерий того, что произвольно взятый объект является созданным руками человека. Петя пока не знает точно, как именно применить этот Критерий для взлома Самого Главного Сервера – но он уверен, что после такого Фундаментального Открытия решение всех остальных вопросов будет уже делом техники. Поэтому сейчас Петя решил взять небольшой отдых, чтобы сходить проведать Васю и выяснить, как продвигается его план, а заодно и невзначай сообщить Васе о своих собственных успехах. Вас же, чтобы не терять время впустую, Петя попросил написать программу, реализующую на практике проверку его Фундаментального Критерия. Программа должна по заданным координатам трех точек на плоскости определить, являются ли данные точки вершинами прямоугольного треугольника, и вычислить длину его гипотенузы.

Формат входных данных

В трех строках входных данных содержатся по два целых числа: координаты X,Y трех вершин соответственно: $-100 \le Xi, Yi \le 100$. Гарантируется, что заданные точки попарно различны и не лежат на одной прямой.

Формат результата

В единственную строку результата вывести единственное вещественное число – длину гипотенузы, с точностью не менее трех значащих цифр после запятой, в случае если заданные вершины образуют прямоугольный треугольник. Если вершины не образуют прямоугольный треугольник, вывести в качестве результата число -1.

Пример входных данных и результата:

standard input	standard output
0 4 3 0 3 4	5.000
2 -1 -1 1 -1 -2	-1

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	-14

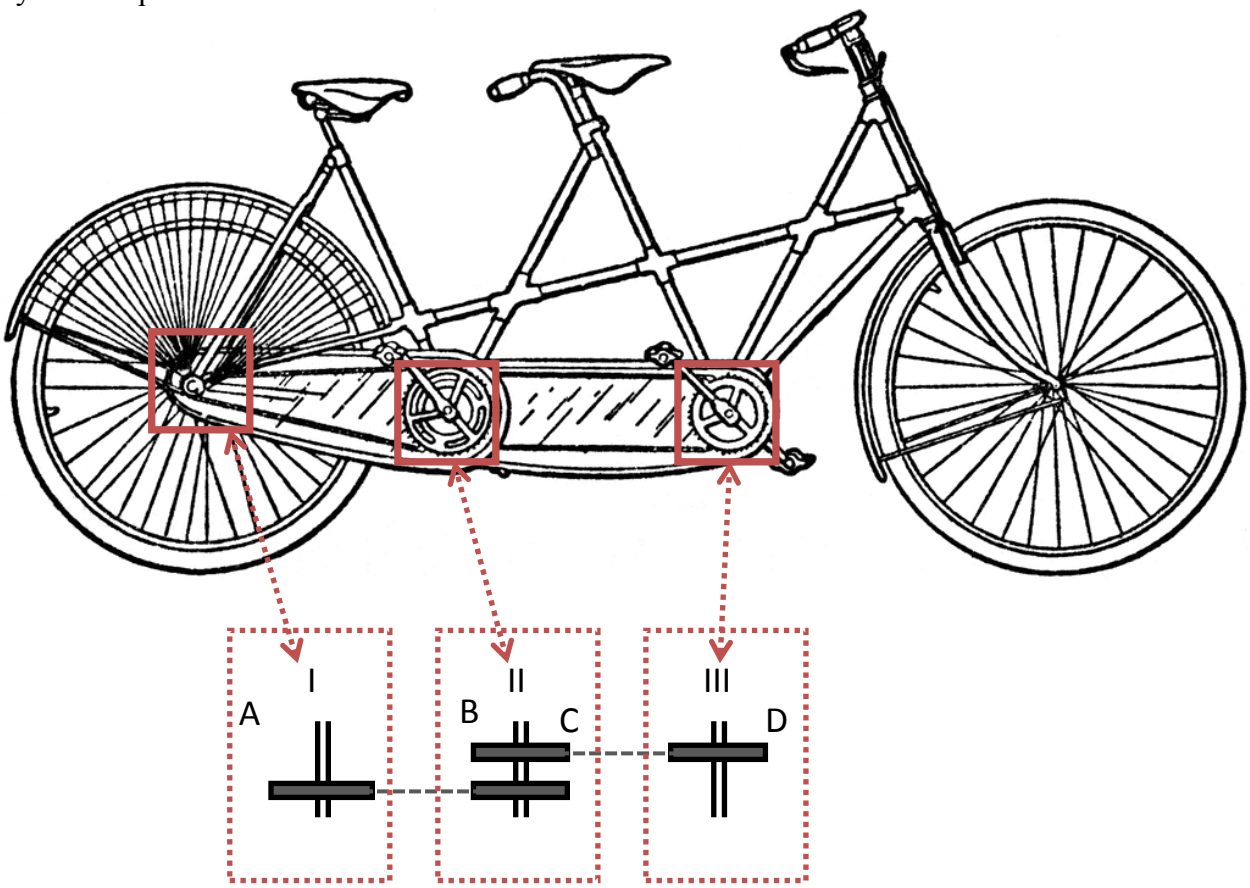
2 2 3 2 5 1 5 3 3 3 4 4	

Задача F. «Тандем»

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

Альберт и Альфред снова изобретают велосипед. На этот раз, правда, речь идет о велосипеде-тандеме. Их не устраивает классическая конструкция такого велосипеда, когда оба велосипедиста должны крутить педали с одинаковой скоростью. Альфред любит очень точную настройку передаточных чисел, такую, чтобы заднее колесо велосипеда крутилось ровно вдвое чаще, чем частота вращения педалей. Также Альфред любит сидеть за рулем, на переднем сиденье тандема. Альберт же предпочитает более спокойную езду, когда можно спокойно любоваться окрестными пейзажами, и не нужно крутить педали очень часто.

Чтобы реализовать эту идею, братья хотят использовать звездочки с разным количеством зубьев на разных осях:



Теперь братья хотят выяснить, могут ли они собрать нужную им конфигурацию, используя имеющиеся у них в наличии шестеренки.

Формат входных данных

В первой строке входных данных содержится единственное целое число N: $4 \le N \le 50$ – количество шестеренок, которые есть у братьев. В следующих N строках входного файла указывается по одному целому числу T_i: $3 \le T_i \le 1000000$ – количество зубчиков на соответствующей шестеренке.

Формат результата

Если возможно из имеющихся шестеренок выбрать четыре таких, что выполняются следующие условия:

1) Угловая скорость вращения оси I ровно вдвое выше, чем угловая скорость вращения оси III

2) Угловая скорость вращения оси II меньше, чем угловая скорость вращения оси III

то вывести в качестве в качестве результата в единственной строке четыре целых числа, разделенных пробелами: номера четырех шестеренок в порядке A, B, C, D (согласно маркировке на рисунке). Если вариантов решения задачи несколько, вывести любой из них, удовлетворяющий условию.

Если решения, удовлетворяющего условию, не существует, вывести в качестве результата единственное число -1.

Пример входных данных и результата:

standard input	**standard output**
5 4 11 77 44 8	2 4 5 1
4 999999 500000 899999 900000	-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 123456789012345678 9	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 C. «Эврика!»

Поскольку по условию задачи входные точки являются попарно различными и не лежат на одной прямой, они гарантированно образуют невырожденный треугольник. Поэтому для решения задачи достаточно вычислить по известным координатам вершин на плоскости длины всех сторон треугольника, а затем проверить, выполняется ли для этих длин сторон теорема Пифагора: квадрат наибольшей по длине стороны должен быть равен сумме квадратов двух других сторон.

При решении этой задачи могут возникнуть две сложности:

1) Вещественное сравнение.

   Поскольку длины сторон a, b, c в общем случае будут вещественны, равенство $a^2 = b^2+c^2$ для предварительно вычисленных вещественных значений может не выполняться точно. Причина в том, что вещественные числа в памяти компьютера хранятся с некоторым приближением (величина его зависит от конкретного языка программирования и используемого типа) – и из-за округлений в процессе вычисления значения a^2 и b^2+c^2 будут немного, но различаться. То есть, выражение

   if $(a^2 = b^2+c^2)$ then …

   окажется ложным, хотя при вычислении с абсолютной точностью оно должно было бы быть истинным.

   Возможных решений данной проблемы два. Первое - при сравнении вещественных чисел использовать сравнение с заданной точностью. Выражение

   if $(a^2 = b^2+c^2)$ then …

   нужно в этом случае переписать в виде:

   if $\mathrm{abs}(a^2 - b^2 - c^2) < \varepsilon$ then …

   где ε – предварительно заданная константа, равная, например, 10^{-9}.

   Сравнение с заданной точностью зачастую является допустимым решением, но иногда также может приводить к ошибкам (когда два близких, но отличающихся вещественных

числа будут при таком сравнении приняты равными). Поэтому более корректным является второй способ - решение задачи в целых числах. В данном случае для проверки теоремы Пифагора нам нужны не сами длины сторон треугольника, а их квадраты: a^2, b^2, c^2 – которые при целочисленных координатах вершин всегда являются целыми числами. Поэтому достаточно вычислить, используя целочисленные типы данных, квадраты длин сторон треугольника, проверить для них выполнение теоремы Пифагора, и, если она выполняется, вычислить и вывести в ответ единственное вещественное число саму длину гипотенузы.

2) Вывод вещественного числа с заданной точностью.
   В каждом языке программирования существуют свои инструменты для вывода вещественных чисел в различных представлениях.

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 F. «Тандем»

Если T[i] – количество зубьев соответствующей шестеренки, ω_I, ω_{II}, ω_{III} – угловые скорости трех осей соответственно (согласно рисунку в условии), то связаны они будут следующими соотношениями:

$\omega_I * T[A] = \omega_{II} * T[B]$

$\omega_{II} * T[C] = \omega_{III} * T[D]$

Согласно условию задачи, должно выполняться:

1) $\omega_I = 2 * \omega_{III}$
2) $\omega_{II} < \omega_{III}$

Выражая эти соотношения через количество зубьев шестеренок, получаем условия:

1) $2 * T[A] * T[C] = T[B] * T[D]$
2) $T[C] > T[D]$

Как и в случае задачи «C», выполнение этих условий следует проверять в целых числах, избегая перехода к вещественным числам.

Малые количество неизвестных (4) и максимально возможное число шестеренок ($N \le 50$) позволяет искать решение задачи полным перебором всех возможных комбинаций шестеренок - то есть, 50*49*48*47= 5527200 комбинаций. Для еще большего ускорения работы алгоритма в перебор можно было внести отсечение, подбирая сначала пару шестеренок C и D и проверяя для неё выполнение условия T[C] > T[D], и лишь для подходящих пар продолжая поиск шестеренок A и 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.