Разбор задач

31 Районная олимпиада школьников Красноярского края по информатике, 7-8 классы

Основной тур от 4 декабря 2017 г.

ID	Задача	Тема	%
1479	A. Остаток от деления	Задачи для начинающих	7
1483	B. Газонокосильщик	Простая математика	14
1484	C. Предложение	Разбор строк	23
1466	D. Бесконечный поезд	Моделирование	41
1485	E. Ремонт забора	Бинарный поиск	48

Беляев Сергей Николаевич

Задача A. Остаток от деления

(Время: 1 сек. Память: 16 Мб Баллы: 100)

Напомним, как в математике определяется остаток от деления целых чисел.

Для любых целых чисел **a** и **b** (**b** ≠ 0) найдется единственная пара целых чисел **q** и **r** таких, что **a** = **q**×**b** + **r**, где 0 ≤ **r** < |**b**|.

Здесь **a** – делимое, **b** – делитель, **q** – неполное частное, **r** – остаток. Следует заметить, что остаток **r** – это всегда неотрицательное число.

В языках программирования существуют операции для вычисления остатка от деления. Однако эти операции практически всегда в случае отрицательных чисел работают по иным правилам.

Ваша задача – по заданным числам **a** и **b** определить значение остатка от деления **a** на **b**.

Входные данные

Входной файл INPUT.TXT содержит два целых числа **a** и **b** ($-10^{18} \le a, b \le 10^{18}$, $b \ne 0$).

Выходные данные

В выходной файл OUTPUT.TXT выведите ответ на задачу.

Примеры

№	INPUT.TXT	OUTPUT.TXT	Пояснение
1	27 4	3	27 = 6*4 + 3
2	-15 4	1	-15 = -4*4 + 1
3	113 -3	2	113 = -37*(-3) + 2
4	-15 -7	6	-15 = 3*(-7) + 6

Система оценивания

Решения, работающие для положительных чисел, будут оцениваться в 25 баллов.

Решения, работающие для чисел, по модулю не превосходящих 1000, будут оцениваться в 50 баллов.

Решения, работающие для чисел, по модулю не превосходящих 10^9, будут оцениваться в 75 баллов.

Задача A. Остаток от деления

$$a = q \cdot b + r \quad (0 \le r < b, b \ne 0)$$

Заметим, что всюду операция вида **a** mod **b** работает корректно при неотрицательных **a** и **b**. Также очевидно, что изменение знака в числе **b** не влияет на **r** (при этом меняется знак у **q**).

При отрицательном значении **a** и положительном **b** в языках Pascal и C++ результатом операции **a** mod **b** может быть отрицательное значение, а ответом в этом случае будет служить **a** mod **b** + **b**.

В общем случае в любом языке программирования корректным будет следующая цепочка действий:

```
read(a,b)

b = abs(b)

write((a mod b + b) mod b)
```

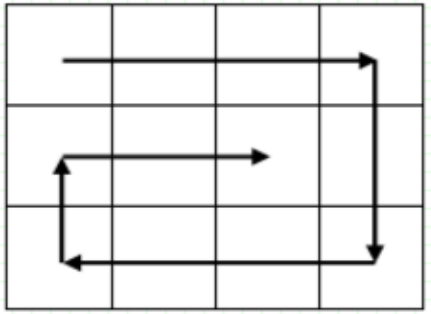
```
//Pascal
var a,b : int64;
begin
  read(a,b);
  b := abs(b);
  write((a mod b + b) mod b)
end.

//C++
#include <iostream>
using namespace std;
int main(){
  long long a,b;
  cin >> a >> b;
  b = b<0?-b:b;
  cout << (a%b+b)%b;
  return 0;
}

#Python
a,b = map(int, input().split())
print(a%abs(b))
```

Задача B. Газонокосильщик

(Время: 1 сек. Память: 16 Мб Баллы: 100)



Газонокосильщику Ивану в очередной раз предстоит подстричь свой газон, который можно представить в виде таблицы высотой N и шириной M. Ширина газонокосилки совпадает с шириной клеток таблицы, поэтому за один проход по прямой можно постричь сразу целую строку или целый столбец. Но постричь газон без поворотов газонокосилки зачастую не представляется возможным.

Иван хочет выполнить свою работу полностью. Для этого он начинает прямолинейное движение с верхнего левого угла в горизонтальном направлении до конца газона, затем он поворачивает направо и далее движется аналогичным образом. Так он продолжает свое движение по спирали до тех пор, пока газон не будет полностью пострижен.

Поскольку повороты направо с газонокосилкой весьма трудоемки, Иван хочет предварительно подсчитать количество поворотов, которые ему предстоит сделать в процессе его работы.

Помогите ему в этом!

Входные данные

Входной файл INPUT.TXT содержит в единственной строке два целых числа N и M – размеры газона ($1 \le M, N \le 2 \cdot 10^9$).

Выходные данные

В выходной файл OUTPUT.TXT выведите целое число – ответ на задачу.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	3 4	4
2	5 3	5

Система оценивания

Решения, работающие для $M \cdot N \le 10^6$, будут оцениваться в 40 баллов.

Решения, работающие для $M+N \le 10^6$, будут оцениваться в 80 баллов.

Задача B. Газонокосильщик

Частичное решение - 40 баллов

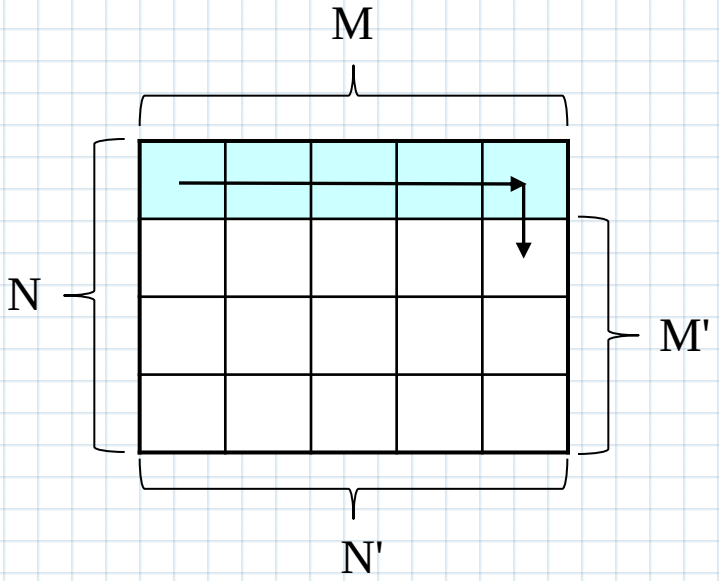
Можно создать матрицу размером N x M и смоделировать движение газонокосильщика (как в известной задаче «Спираль»), при этом считая повороты. Такое решение не может работать на всех тестах, т.к. использует O(N·M) памяти и времени.

Частичное решение - 80 баллов

Мы можем применить метод динамического программирования и на каждом шаге, где (N,M) - размеры газона и N > 1 сводить задачу к меньшей размерности, т.е. двигаться горизонтально из левого верхнего в правый верхний угол, делать поворот, делать одно движение вперед и говорить, что газон теперь имеет размеры (M, N-1). Как только высота станет равна 1, то процесс завершается. При этих действиях несложно подсчитать число поворотов. Заметим, что здесь сложность алгоритма имеет порядок O(N+M), что также не позволяет решить задачу при больших значениях N и M.

Реализация данного алгоритма на языке Python:

```
#Python
n,m = map(int, input().split())
res = 0
while n>1:
    res += 1
    n,m = m,n-1
print(res)
```



Задача B. Газонокосильщик

Полное решение

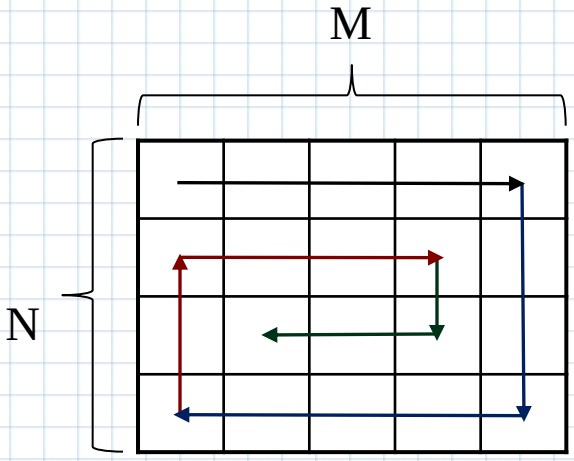
Заметим, что если M ≥ N, то Иван завершит работу на горизонтальной линии. Первую линию он сможет скосить без поворотов. Далее для косьбы каждой последующей горизонтальной линии необходимо будет выполнять ровно 2 поворота. Всего в данном случае потребуется 2×(N-1) поворотов.

При M < N газонокосильщик завершит работу на вертикальной линии. Для прохождения первой вертикальной линии потребуется один поворот, а для косьбы каждой последующей такой линии необходимо будет выполнять 2 поворота. Здесь общее число поворотов составит 2×M-1.

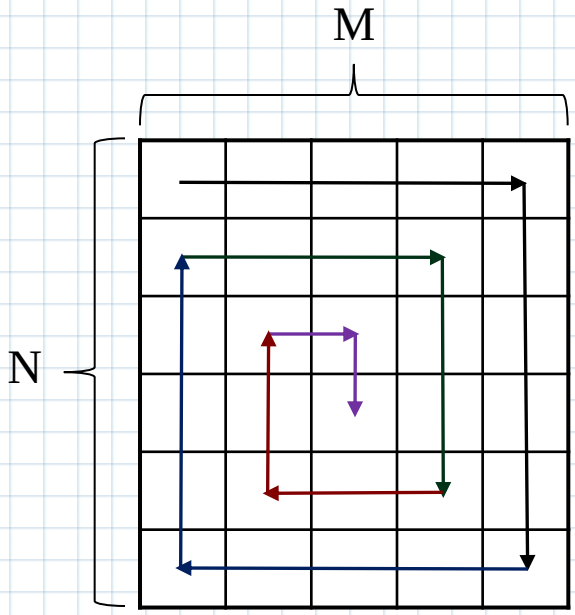
В результате получаем простую алгоритмическую реализацию:

```
int64 n,m
read(n,m)
if(m<n) write(2*m-1)
   else write(2*n-2)
```

Следует не забывать о том, что ответ может не помещаться в знаковый 4-байтный целый тип. Поэтому необходимо использовать 8-байтовый целый тип, либо 4-байтный целый тип без знака.



Случай №1: M ≥ N



Случай №2: M < N

Задача B. Газонокосильщик

```pascal
//Pascal
var n,m : Int64;
begin
   read(n, m);
   if m<n then write(2*m-1)
          else write(2*n-2)
end.
```

```cpp
//C++
#include <iostream>

using namespace std;

int main(){
   unsigned n,m;

   cin >> n >> m;

   if(m<n) cout << 2*m-1;
      else cout << 2*n-2;

   return 0;
}
```

```python
#Python
n, m = map(int, input().split())
print(2*m-1) if m<n else print(2*n-2)
```

Предложение

(Время: 1 сек. Память: 32 Мб Сложность: 23%)

В предложении необходимо найти самое короткое и самое длинное слово.

Входные данные

В первой строке входного файла INPUT.TXT содержится предложение, состоящее из слов, разделенных пробелами. Каждое слово – это последовательность букв английского алфавита. Слова могут отделяться друг от друга одним или несколькими пробелами. Также один или несколько пробелов могут быть как в начале, так и в конце предложения. Гарантируется, что в предложении присутствует хотя бы одно слово. Длина предложения может быть от 1 до 10^6 символов.

Выходные данные

В первой строке выходного файла OUTPUT.TXT выведите слово из заданного предложения, состоящее из наименьшего количества букв. Если таких слов несколько, выведите лексикографически наибольшее из них. Во второй строке следует вывести слово, состоящее из наибольшего числа букв. Если таких слов несколько, выведите лексикографически наименьшее из них.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	one two	two one
2	the pen is mightier than the sword	is mightier
3	No Man IS an IsLand	an IsLand
4	ab Ab AB aB	ab AB

Пояснение

Чтобы лексикографически сравнить слова одинаковой длины, нужно найти и отбросить максимальные совпадающие начала слов, после чего сравнить первые из оставшихся букв. Меньшим будет слово, у которого эта буква встречается раньше в алфавите. При этом любая строчная (маленькая) буква считается лексикографически больше любой заглавной (большой) буквы.

Система оценивания

Решения, работающие только для предложений, состоящих из слов одинаковой длины и не содержащих «лишних» пробелов, будут оцениваться в 60 баллов.

Задача C. Предложение

Рассмотрим сначала ряд частичных решений. Примеры решений, представленные ниже используют встроенные возможности языка для чтения списка слов и функции max и min для поиска максимального и минимального элементов в списке. Данные решения работают на тестах с равной длиной слов. PascalABC.NET использует в функциях min и max кодировку UTF-16, что не позволяет выполнять корректное сравнение строк согласно 8-битной кодировке ASCII.

```
//PascalABC.NET 3.2 - 50 баллов
begin
  var a := ReadString.ToWords;
  Arr(a.Max, a.Min).Print(NewLine)
end.


#Python 3 - 60 баллов
a = input().split()
print(max(a), min(a), sep='\n')
```

Задача C. Предложение

Предположим, что GetWord() - некоторая функция, которая позволяет получить очередное слово предложения. При этом данная функция возвращает логическое значение, соответствующее успешности чтения слова. Тогда читая последовательно слова, мы можем в некоторых строковых переменных Min и Max сохранять на текущий момент самое короткое и самое длинное слово соответственно, а в случае совпадения длины делать соответствующую проверку. После прочтения всех слов из входных данных мы получим в этих переменных ответ на задачу.

Алгоритмическая реализация вышеописанного:

```
string s, Min, Max

Min = Max = GetWord()

while(not eof()){
  s = GetWord()
  if(len(s) < len(Min) or
     len(s) = len(Min) and s > Min) Min = s
  if(len(s) > len(Max) or
     len(s) = len(Max) and s < Max) Max = s
}

writeln(Min)
writeln(Max)
```

Задача C. Предложение

```
//PascalABC.NET 3.2
begin
  var a := ReadString.ToWords;
  var s_min := a[0];
  var s_max := a[0];

  foreach var s in a do begin
    if (length(s)<length(s_min)) or
       (length(s)=length(s_min)) and (s>s_min) then
         s_min := s;
    if (length(s)>length(s_max)) or
       (length(s)=length(s_max)) and (s<s_max) then
         s_max := s
  end;

  writeln(s_min);
  write(s_max)
end.
```

Задача D. Бесконечный поезд

(Время: 2 сек. Память: 16 Мб Баллы: 100)

Это интерактивная задача.

Представьте себе замкнутую по окружности железную дорогу и поезд, последний вагон которого скреплен с первым так, что внутри можно перемещаться между вагонами. Вы оказались в случайном вагоне и желаете определить количество вагонов N ($1 \le N \le 10\,000$). В каждом вагоне можно менять положение переключателя света, но начальное состояние переключателей случайное и заранее неизвестно.

Протокол взаимодействия

Вы можете делать следующие запросы программе жюри:

switch – изменение состояние переключателя света в текущем вагоне;

front – перемещение в следующий вагон;

back – перемещение в предыдущий вагон;

N – вывод ответа, где N – целое число (количество вагонов в поезде).

После каждого запроса (кроме последнего) вашей программе будет сообщено состояние переключателя света в том вагоне, в котором вы оказались: «on» - если свет горит и «off» – иначе. Последним запросом должен быть вывод целого числа N, после чего ваша программа должна немедленно завершиться. На момент подачи любой команды модуль разности между количеством поданных команд front и количеством поданных команд back не должен превышать 3×N.

Пример

№	стандартный ввод	стандартный вывод
1	on off on off on off	switch front front switch front back 3

Пояснение к примеру

В поезде 3 вагона. Предположим, что мы находимся в первом вагоне. Свет горит только в третьем вагоне. Сначала мы включаем свет в первом вагоне, перемещаемся во второй, затем в третий и выключаем в нем свет. После чего движемся вперед, попадаем в первый вагон, движемся назад, попадаем в последний третий вагон и выводим ответ. Заметим, что на основании выполненных в примере запросов нельзя утверждать, что в поезде 3 вагона.

Примечание

Для корректной работы программы после каждой операции вывода данных выводите перевод строки, а также очищайте буфер вывода. Очистка буфера вывода производится следующим образом:

- В языке Pascal: **flush(output)**
- В C/C++: **fflush(stdout)** или **cout.flush()**
- В Java: **System.out.flush()**
- В Python: **sys.stdout.flush()** из библиотеки **sys**
- В C# и Basic: **Console.Out.Flush()**

Система оценивания

Решения, работающие только для $N \le 100$ будут оцениваться в 60 баллов.

Задача D. Бесконечный поезд

Частичное решение: 10 000 + N (20 баллов)

Если бы не было ограничения на количество прохождений по кругу, то решение задачи было бы достаточно простым. Достаточно было бы выключить свет во всех вагонах (для чего можно сделать 10 000 шагов в одном направлении), далее зажечь свет в одном вагоне и продолжать движение вперед до тех пор, пока мы не встретим вагон с включенным светом. Однако, для $N \leq 5000$ это не работает, т.к. мы гарантированно сделаем не менее трех кругов в одном направлении. На этом и основано данное частичное решение, которое работает только при $N > 5000$.

Если сначала двигаться на 5000 шагов вперед, затем на 10 000 шагов назад (вместо предложенного ранее простого движения на 10 000 шагов вперед), то можно получить решение, работающее для $N > 3333$, которое даст 24 балла.

Частичное решение: $O(N^2)$ - 60 баллов

Положим сначала, что $N = 1$ и будем проверять факт, что в поезде ровно N вагонов. В случае подтверждения мы получим ответ, в противном случае будем увеличивать N на единицу. Для того, чтобы проверить, что N - это ответ на задачу, зажжем свет в текущем (первом) вагоне и переместимся на N шагов вперед, выключая свет во всех вагонах, где он горит. Далее сделаем N шагов назад (вернемся тем самым в первый вагон) и посмотрим: горит ли свет в текущем вагоне? Если все еще горит, то ответ больше, чем N. А если нет, то N - ответ на задачу. Данное решение при больших значениях N требует много операций и поэтому не пройдет по времени.

Полное решение: O(N)

Оптимизируем предыдущее частичное решение. А именно будем также зажигать свет в первом вагоне и двигаться вперед на K шагов, выключая свет во всех вагонах, далее также будем возвращаться в первый вагон и смотреть: горит ли в нем свет после наших действий? Единственное здесь отличие в том, что каждый раз значение K будет увеличиваться не на единицу, а вдвое. Это позволит найти такое значение K, что $N \leq K$ за не более, чем 4N шагов, при этом мы сделаем не более 2 кругов в одном направлении. Для дальнейшего определения значения N мы зажжем свет в текущем вагоне и будем продолжать движение в одном направлении, пока не встретим вагон с включенным светом. Количество шагов, которые мы при этом выполним - это и будет N - ответ на задачу. В результате данного алгоритма будет выполнено не более, чем 5N шагов.

Задача D. Бесконечный поезд

```
//Free Pascal
var i,k,n : integer;

function get(cmd :string) : boolean;
var ans : string;
begin
  writeln(cmd);
  flush(output);
  readln(ans);
  get := ans = 'on';
end;

begin
  k := 1; n := 1;
  if not get('switch') then
    get('switch');

  while not get('switch') do begin
    get('switch');
    k := k*2;
    for i:=1 to k do
      if get('front') then
        get('switch');
    for i:=1 to k do get('back')
  end;

  while not get('front') do inc(n);

  writeln(n)
end.
```

Задача E. Ремонт забора

(Время: 2 сек. Память: 16 Мб Баллы: 100)

Илья работает плотником и перед ним стоит задача отремонтировать забор. Забор состоит из N сегментов, i-й из которых имеет высоту A_i. Илья располагает тележкой, на которой лежит стопка из M досок, j-я из которых имеет длину B_j.

Илья идет вдоль забора от первого сегмента к последнему и катит перед собой тележку с досками. Если он хочет увеличить высоту текущего сегмента, он может взять доску с тележки и прибить ее сверху. Тогда новая высота сегмента будет равна сумме изначальной высоты сегмента и длины прибитой доски.

Собираясь увеличить высоту сегмента забора, Илья поступает следующим образом. Он либо использует для увеличения сегмента верхнюю доску с тележки, либо выкидывает одну или несколько верхних досок с тележки и использует следующую доску. Илья никогда не прибивает больше одной доски к сегменту, не возвращается назад вдоль забора и никогда не подбирает ранее выкинутые доски.

Помогите Илье определить максимально возможную высоту забора после ремонта, если под высотой забора понимается высота самого низкого сегмента.

Входные данные

Первая строка входного файла INPUT.TXT содержит целое число N – количество сегментов в заборе ($1 \le N \le 10^5$). Во второй строке содержатся N целых чисел $A_1, A_2, \ldots A_N$ – высоты сегментов забора, перечисленные в том порядке, в котором мимо них пройдет Илья ($1 \le A_i \le 10^8$).

В третьей строке находится целое число M – количество досок на тележке ($1 \le M \le 10^5$). В четвертой строке содержатся M целых чисел $B_1, B_2, \ldots B_M$ – длины досок на тележке, начиная с верхней ($1 \le B_j \le 10^8$).

Выходные данные

В выходной файл OUTPUT.TXT выведите целое число H – максимальную возможную высоту забора после ремонта.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	3 10 5 10 1 5	10
2	6 2 5 4 1 7 5 7 2 3 1 3 2 4 6	5

Система оценивания

Решения, работающие для $H \le 100$, будут оцениваться в 25 баллов.

Задача E. Ремонт забора

N - количество сегментов в заборе

M - количество досок на тележке

A[1..N] - высоты сегментов забора

B[1..M] - высоты досок на тележке

Опишем функцию для проверки возможности отремонтировать забор до высоты h:

```
bool Check(int h){
  j = 1
  for i=1..N{
    if(A[i] ≥ h) continue
    while(j≤M and A[i]+B[j]<h) j++
    if(j > M) return false
    j++
  }
  return true
}
```

Используя функцию Check, несложно реализовать частичное решение, работающее для небольших значений ответа H:

```
ans = 2
while(Check(ans)) ans ++
write(ans-1)
```

Асимптотика приведенного выше алгоритма O((N+M)×H). Для полного решения необходимо использовать бинарный поиск по ответу. Например, следующим образом:

```
l = 1; r = 2·10^8
while(l < r){
   x = (l+r+1) div 2
   if(check(x)) l = x
          else r = x-1
}
write(r)
```

Асимптотика данного решения O((N+M)×log H), что позволяет уложиться во временные рамки.

Задача E. Ремонт забора

```
//PascalABC.NET 3.2
var m,n,l,r,x : integer;
    a,b : array of integer;

function check(x : integer) : boolean;
begin
  result := false;
  var j := 0;
  for var i :=0 to n-1 do begin
    if a[i] >= x then continue;
    while (j<m) and (a[i]+b[j]<x) do inc(j);
    if j = m then exit;
    inc(j)
  end;
  result := true
end;

begin
  read(n);
  a := ReadArrInteger(n);
  read(m);
  b := ReadArrInteger(m);

  (l,r) := (1, 200000000);
  while l<r do begin
    x := (l+r+1) div 2;
    if check(x) then l := x
                else r := x-1
  end;

  write(r)
end.
```