Методические рекомендации по разбору предложенных олимпиадных задач

Муниципального этапа Всероссийской олимпиады школьников по информатике в 2017-2018 учебном году

7- 8 класс

Задача А. Автобусы.

Имя входного файла: input.txt
Имя выходного файла: output.txt
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 мегабайта

Для заезда в детский оздоровительный лагерь организаторы решили заказать автобусы. Известно, что в лагерь собираются поехать *N* детей и *M* взрослых. Каждый автобус вмещает *K* человек. В каждом автобусе, в котором поедут дети, должно быть не менее двух взрослых.

Определите, удастся ли отправить в лагерь всех детей и взрослых, и если да, то какое минимальное количество автобусов требуется для этого заказать.

Формат входного файла

На вход программы поступают 3 натуральных числа, записанных через пробел - *N, M* и *K*, каждое из них не превосходит 10 000.

Формат выходного файла

Выведите количество автобусов, которые нужно заказать. Если же отправить всех в лагерь невозможно, выведите 0 (ноль).

Пример входного и выходного файлов

Входные данные	**Выходные данные**
10 4 7	2
10 4 5	0

Решение.

Алгоритм:

Во первых: нужно учесть, что К может принимать значение меньше или равное 2. В этом случае следует выводить 0, потому, что в каждый автобус мы будем вынуждены посадить взрослых (а дети так и не уедут). Теперь рассмотрим случай когда К больше двух: в том случае нужно будет ровно n/(k-2) автобусов для перевозки детей. Заметим, что если n не

делится нацело на k-2, то автобусов понадобиться на один больше. Так же мы не сможем уехать если количество взрослых делённое на два меньше количества нужных автобусов для перевозки детей. Во всех остальных случаях ответом будет (m+n)/k, но если (m+n) не делится нацело на k то на один автобус больше.

Программа:

```
var a,b,c,k,n,p: integer;
begin
read(a,b,c);
if (b<2) or (c<3) then writeln(0)  else
begin
k:=a div (c-2); //2
if a mod (c-2) <> 0 then inc(k);
if b div k < 2 then writeln(0)  else
begin
k:=(a+b) div c;
if (a+b) mod c <> 0 then inc(k);
writeln(k);
end;
end;
end
```

Задача B. Лавочки

Имя входного файла: input.txt
Имя выходного файла: output.txt
Ограничение по времени: 1 секунда
Ограничение по памяти: 64 мегабайта

Лавочки в парке устроены следующим образом. Несколько одинаковых кубических гранитных блоков ставятся в ряд, а на них кладется гранитная плита (см. рисунок). Архитектор-модернист решил, что будет интереснее, если у всех лавочек расположение гранитных блоков-ножек будет разным (и не обязательно симметричным). При этом они располагаются так, чтобы плита не падала: для этого достаточно, чтобы и слева, и справа от центра плиты был хотя бы один гранитный блок или его часть (в частности, если центр плиты приходится на середину какого-нибудь блока, то и слева, и справа от центра плиты находится часть блока, и плита не падает).

Грабители обнаружили, что можно по одному вытаскивать гранитные блоки, находящиеся с краю (как слева, так и справа). Они хотят вытащить из-под лавочки как можно больше блоков так, чтобы она при этом не упала (передвигать оставшиеся блоки нельзя). Определите, какие блоки они должны оставить.

Формат входного файла

В первой строке входных данных содержатся два числа: L - длина лавочки и K - количество гранитных блоков-ножек. Оба числа натуральные и не превышают 10 000.

Во второй строке следуют K различных целых неотрицательных чисел, задающих положение каждой ножки. Положение ножки определяется расстоянием от левого края плиты до левого края ножки (ножка - это куб размером 1×1×1). Ножки перечислены слева направо (то есть начиная с ножки с меньшим расстоянием до левого края).

Формат выходного файла

Требуется перечислить ножки, которые грабителям нужно оставить. Для каждой ножки нужно выдать ее положение, как оно задано во входных данных. Ножки следует перечислять слева направо, в том порядке, в котором они встречаются во входных данных.

Примеры входных и выходных файлов

Входные данные	Выходные данные
5 2 0 2	2
13 4 1 4 8 11	4 8
14 6 1 6 8 11 12 13	6 8

Второй пример соответствует лавочке на рисунке.

Решение.

Алгоритм:

Обозначим координату (положение) i-й ножки как d_i. Найдём числа $left$ и $right$ — номера самой правой ножки, хотя бы часть которой находится левее середины лавочки, и самой левой ножки, хотя бы часть которой находится правее середины лавочки, соответственно: $left=\max\{i \mid 2d_i<L\}$, $right=\min\{i \mid 2.(d_i+1)>L\}$.

Если в итоге $left=right$ (то есть L нечётно и $\exists i \mid d_i=\lfloor \frac{L}{2} \rfloor$), то вывести нужно одно число d_{left}, в противном случае — сначала d_{left}, затем d_{right}.

Программа:

```
var L,k,n,i: longint;//0..10 000
        a: array [0 .. 9999] of boolean;
begin
        read(L,k);
        for i:=1 to k do begin
                read(n);
                a[n]:=true;
        end;
        if (L mod 2 <>0) and (a[L div 2]) then begin
                write(L div 2);
                halt;
        end ;
        for i:=(L-1) div 2 downto 0 do
                if a[i] then begin
                        write(i,' ');
                        break;
                end;
        for i:=l div 2 to L-1 do
                if a[i] then begin
                        write(i);
                        break;
                end;
end.
```

Задача C. Выборы

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Ограничение по времени:	3 секунды
Ограничение по памяти:	64 мегабайта

На выборах в Государственную думу в избирательные бюллетени внесено N партий. Электронный сканер для считывания информации с бюллетеней передает информацию о каждом бюллетене в следующем формате: если в соответствующей клетке бюллетеня стоит пометка, то сканер передает + (плюс), в противном случае он передает - (минус). Таким образом, он передает последовательность из N символов - плюсов и минусов.

Бюллетень считается действительным, если пометка есть ровно в одной клетке. Недействительные бюллетени в подсчете результатов выборов не участвуют.

Партия проходит в Государственную Думу, только если она набирает не менее 7% от общего числа **действительных** бюллетеней.

Требуется вывести номера (в порядке их перечисления в бюллетене) всех партий, которые проходят в Государственную Думу.

Формат входного файла

В первой строке входных данных содержатся два числа, разделенные пробелом: N - количество партий и M - количество бюллетеней. Оба числа натуральные, $N <= 200$, $M <= 100\,000$.

В следующих M строках записана информация, полученная из бюллетеней. Каждая строка - последовательность из N символов + или - (без пробелов).

Гарантируется, что есть хотя бы один действительный бюллетень.

Формат выходного файла

Выведите через пробел номера партий, прошедших в Думу, в порядке возрастания. Если ни одна из партий не проходит в Думу, выводить ничего не нужно.

Примеры входных и выходных файлов

Входные данные	Выходные данные
3 4 +-- +-- -+- +-+	1 2
1 5 + - - - -	1

Решение.

Алгоритм:

Напишем специальную функцию who, по данной строчке-бюллетеню возвращающую номер выбранного в данном бюллетене кандидата или 0 для недействительного бюллетеня (для этого достаточно один раз пройти циклом по строчке-бюллетеню, помня текущий ответ). Теперь для каждого бюллетеня вызовем функцию who от него и, если результат не равен нулю, увеличим на 1 числа *K* (количество действительных бюллетеней) и *qwho(*количество действительных голосов, отданных за партию с номером, равным результату функции who). Осталось вывести все *i* такие, что $100gi \geq 7K$.

Программа:

```
{$h+}
var
flag:boolean;
a:array [1..10000] of longint;
s:string;
n,m,max,k,i,j:longint;
begin
readln(n,m);
max:=m;
for i:=1 to m do
begin
readln(s);
k:=0;
flag:=false;
for j:=1 to length(s) do
if s[j]='+' then
inc(k);
if k<>1 then
begin
dec(max);
flag:=true;
end;
if not flag then
for j:=1 to length(s) do
if s[j]='+' then
inc(a[j]);
end;
for i:=1 to n do
begin
if a[i]>=max*0.07 then
write(i,' ');
end;
end.
```

Задача D. Билет на электричку

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Ограничение по времени:	3 секунды
Ограничение по памяти:	64 мегабайта

В новых элитных электричках каждому пассажиру положено сидячее место. Естественно, количество сидячих мест ограничено и на всех их может не хватить. Маршрут электрички проходит через N станций, занумерованных от 0 до N-1. Когда человек хочет купить билет, он называет два числа x и y – номера станций, откуда и куда он хочет ехать. При наличии хотя бы одного сидячего места на этом участке на момент покупки ему продается билет, иначе выдается сообщение «билетов нет» и билет не продается. Ваша задача – написать программу, обслуживающую такого рода запросы в порядке их прихода.

Формат входного файла

В первой строке содержаться три числа N – количество станций ($1 \le N \le 10\,000$), K – количество мест в электричке ($1 \le K \le 1000$) и M – количество запросов ($1 \le M \le 50\,000$). В следующих M строках описаны запросы, каждый из которых состоит из двух чисел x и y ($0 \le x < y < N$).

Формат выходного файла

На каждый запрос ваша программа должна выдавать результат в виде числа 0 если билет не продается и 1 если билет был продан. Каждый результат должен быть на отдельной строке

Примеры входных и выходных файлов

Входные данные	**Выходные данные**
5 2 4 0 4 1 2 1 4 2 4	1 1 0 1

Решение.

Алгоритм:

Представим массив длины N в i-той ячейке которого будем хранить количество пассажиров уже купивших билет и едущих от станции i к (i+1).
Будем поддерживать для такого массива RMQ (дерево максимумов) с возможностью быстрой модификации (прибавления) на отрезке.

Теперь при обработке каждого запроса мы сначала узнаём максимум на отрезке [x; y-1], и, если он меньше K (т.е. между каждой парой станций на маршруте существует хотя бы одно свободное место), продаём билет и выполняем update(x, y-1, +1). Иначе отказываем в продаже билета.

Программа:

```
var
mas : array [0..10001] of longint;
n, m ,k, i, a, b, j, z :longint;

begin
readln(n, k, m);
for i := 0 to n-1 do
  mas[i]:=k;
for i:=1 to m do
begin
readln(a, b);
  for j := a to b-1 do
      if mas[j]=0 then z:=5;
  if z=5 then writeln('0')
  else
    begin
     for j:= a to b-1 do dec(mas[j],1);
     writeln('1');
    end;
    z:=0;
end;
end.
```

Задача E. Камни

Имя входного файла:	input.txt
Имя выходного файла:	output.txt
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

На столе лежат *N* камней. За ход игрок может взять

- 1 или 2 камня, если *N* делится на 3;
- 1 или 3, если *N* при делении на 3 дает остаток один;
- 1, 2 или 3, если *N* при делении на 3 дает остаток два.

Каждый ход можно сделать при наличии достаточного количества камней. Проигрывает тот, кто хода сделать не может.

Формат входного файла

Вводится целое число $0 < N <= 100$.

Формат выходного файла

Выведите 1 или 2 – номер игрока, который выиграет при правильной игре.

Примеры входных и выходных файлов

Входные данные	**Выходные данные**
3	2
5	1

Решение.

Алгоритм:

Пусть (F[i] = 1) если выигрывает первый и (F[i] = 2) если второй. Тогда заметим, что F[1]=1,F[2]=1 F[3] = 2. Теперь мы просто заполним наш массив F. Будем считать, что 1 это выигрышная позиция 2 проигрышная. Тогда если мы можем из теперешней позиции попасть в проигрышную, то она выигрышная, а если мы попадаем только в выигрышные, то наша позиция проигрышная. Осталось пробежаться циклом от 4 до n. И выписать условия для разной кратности 3. На самом деле, потом можно заметить, что позиции кратные 3ём проигрышные, а все остальные выигрышные.

Программа:

```
Program kamni;
var
n : integer;
Begin
  readln(n);
```

```
  if(n  mod  3 = 0) then
    writeln('2');
  if(n  mod  3 <> 0) then
    writeln('1');
end.
```