Задача 1.

Тесты для проверки

№ теста	Входные данные	Выходные данные	Балл
1	4 3 2 1 1	2 3	1
2	3 20 22 18	1 3	1
3	4 1 2 3 3	2 3	1
4	5 5 5 5 5 5	5 1	1
5	3 1000 1 1000	2 2	1
6	1 1	1 1	1
7	5 1000 1000 1 1000 1 1000	3 2	1
8	5 1000 1000 1 1000 2	3 2	1
9	6 6 5 4 3 2 1	1 6	1
10	6 6 5 6 5 2 1	2 4	1

Решение. Один из вариантов решения может быть таким:

```
var i, n, kb, vt, vm, p, pr:integer;  l:array[1..1001] of integer;
begin
read(n);
for i:=1 to n do read(l[i]);
While p=0 do begin
             p:=1;
             for i:=2 to n do
                      if l[i-1]>l[i] then begin
                                           p:=0;
                                         pr:=l[i-1];
                                         l[i-1]:=l[i];
                                         l[i]:=pr;
                                         end;
             end;
vm:=1;
vt:=1;
for i:=2 to n+1 do
                  if l[i-1]=l[i] then inc(vt)
                                 else begin
                                       inc(kb);
                                      if vt>vm then vm:=vt;
                                      vt:=1;
                                      end;
write(vm, ' ', kb);
end.
```

Автоматическое тестирование данной задачи на сайте **codeforces.com**
по адресу: **http://codeforces.com/problemset/problem/37/A**

Задача 2. НЛО

Решение переменными 80 баллов.
Тесты:
Решение:

Решение переменными:

```
program massiv;
var
  n,a,max,min,i : longint;
begin
  readln(n);
  max:=-maxlongint;
  min:=maxlongint;
  for i:=1 to n do
    begin
      readln(a);
      if a>max then max:=a;
      if a<min then min:=a;
    end;
  writeln(max,' ',min);
end.
```

Решение одномерным массивом:

```
program massiv;
var
  a          : array [1..3000] of  longint;
  n,max,min,i : longint;
begin
  readln(n);
  for i:=1 to n do readln(a[i]);
  max:=a[1];
  min:=a[1];
  for i:=2 to n do
    begin
      if a[i]>max then max:=a[i];
      if a[i]<min then min:=a[i];
    end;
  writeln(max,' ',min);
end.
```

Тесты для проверки

№ теста	Входные данные	Выходные данные	Балл
1	1 2 3 4 5 6 7 8 9 0	9 0	1
2	5 7	7 5	1
3	9 8 7 6 5 4 3	9 3	1
4	1 2 3	3 1	1
5	543 534 542	543 534	1
6	1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18	18 1	1
7	12 21	21 12	1
8	2	2 2	1
9	3 3 3 3 3 3 3 3 3 3	3 3	1
10	100 99	100 99	1

Задача 3. Решение:

Несмотря на то что речь в этой задаче идет о числе, эта задача является задачей на умение обрабатывать строки, поскольку многие языки программирования не имеют поддержки длинных целых чисел. Поэтому входную последовательность чисел нужно считать в строку и работать с ней, как со строкой.

Прежде всего, определим остаток от деления исходного числа на 3. Для этого нужно посчитать сумму цифр исходного числа (то есть сумму отдельных символов строки) и взять остаток от деления суммы на 3. Если остаток от деления на 3 суммы цифр равен 0, то нужно увеличить какую-то цифру числа на 3, 6 или 9, так как в этом случае число уже делится на 3, и нужно изменить одну цифру так, чтобы остаток от деления на 3 не изменился. Если остаток равен 1, то одну цифру числа нужно увеличить на 2, 5 или 8. Если же остаток равен 3, то одну цифру числа нужно увеличить на 1, 4 или 7.

Поскольку нам нужно сделать новое число максимально большим, то нужно максимально увеличить крайнюю левую цифру исходного числа. Будем перебивать все цифры исходного числа слева направо, пока не найдем первую цифру, которую можно увеличить на нужную величину, при этом цифру нужно увеличить как можно больше (после увеличения эта цифра станет равна 7, 8 или 9).

Наконец, возможна ситуация, когда ни одной цифры увеличить нельзя, такое возможно, например, когда число состоит из одних цифр 9, или для числа 888, или для числа 7878. В этом случае нужно уменьшить последнюю цифру числа так, чтобы число стало кратным 3. Например, число 7878 будет преобразовано в число 7875.

Пример решения задачи на Паскале:

```
var N:string;
sum, i, k, d, L:longint;
begin
  readln(N);
  L:=length(N);
  sum := 0;
  for i:=1 to L do
     sum := sum+ ord(N[i])-ord('0');
  k := 3 - sum mod 3;
  for i:=1 to L do
   if ord(N[i])-ord('0') + k <= 9 then
   begin
     d := ord(N[i])-ord('0') + k;
     while d + 3 <= 9 do d := d + 3;
     N[i] := chr(d + ord('0'));
     writeln(N);
     readln;
     exit;
   end;
   d := ord(N[L])-ord('0');
   if k = 1 then
     d := d - 2
   else if k = 2 then
     d := d - 1
     else
       d := d - 3;
   N[L] := chr(d + ord('0'));
   writeln(N);
end.
```

Это решение является наиболее эффективным, его вычислительная сложность составляет O(n), где n — длина данного числа. Но поскольку длина числа не превосходит 100 цифр, можно придумать и менее эффективное решение, которое все равно наберет максимальный балл. Будем перебирать все цифры исходного числа по одной и будем менять каждую цифру числа на все возможные цифры от 0 до 9. В результате мы получим новое число, которое должно быть не равно исходному числу и должно делиться на 3. Делимость на 3 можно проверить, посчитав остаток от деления суммы цифр числа, а в языке Python есть длинная целочисленная арифметика, поэтому можно преобразовать строку к типу int и взять остаток от деления на 3. В результате мы будем получать какие-то новые числа, которые будут храниться, как строки, отличающиеся от исходной строки одним символом и кратные 3. Из этих строк нужно взять максимальную.

Пример подобного решения на языке Python:

```
n = input()
ans = 0
digits = '0123456789'
for i in range(len(n)):
    for digit in digits:
        new = int(n[:i] + digit + n[i + 1:])
        if digit != n[i] and new % 3 == 0:
            ans = max(ans, new)
print(ans)
```

input.txt	output.txt
8	9
9	6
72	78
892	897
962	972
898	897
7878	7875
9305	9705
5123457	8123457
898941694	898991694
99980360	99990360

Задача 4. Решение:

Рассмотрим два способа решения задачи. Давайте повторим в памяти компьютера события, происходившие в легенде.

Способ первый. Будем хранить в массиве имена (то есть номера) всех живых на текущий момент воинов. Причем удобно, чтобы номера людей были записаны в элементах массива с 0 по N-1 (чуть позже станет ясно, почему так). Когда воин будет умирать, будем удалять его из массива, и тех, кто стоял за ним, "сдвигать" на один элемент влево.

Заметим, что если мы уже убили L человек, то в живых осталось M=N-L. Пусть мы только что (на L-ом шаге) убили человека, который был записан в нашем массиве в элементе с номером j (и сдвинули людей, которые были записаны в массиве в элементах с j+1 по M на один элемент влево). Тогда следующим нужно убивать человека, который записан в этом массиве в элементе с номером (j+k-1) mod M. (Запись a mod b обозначает остаток от деления числа a на b).

Взятие остатка от деления на M нужно затем, чтобы как бы "замкнуть" наш массив в круг (то есть как только мы достигаем конца массива, оказываемся в его начале). Вот тут как раз и важно, что люди в массиве у нас расположены, начиная с 0-го элемента - операция взятия остатка от деления на M может дать в качестве результата числа от 0 до M-1. Конечно, можно изначально расположить людей, начиная и с 1-го элемента, однако тогда в этом месте формула окажется несколько сложнее.

Способ второй. Заведем массив, где будем помечать мертвых воинов (т.е. в i-м элементе хранится, жив воин i, или уже нет). Пусть у нас на текущем шаге M живых людей и на предыдущем шаге умер воин j. Чтобы найти следующего, будем бежать по массиву, отсчитывая живых и пропуская мертвых. Тот человек, на котором мы насчитаем k mod M и должен умереть следующим. Почему k mod M, а не k? Считалочка сначала пройдет k div M полных кругов, а затем остановится на человеке k - (k div M) * M = k mod M. Если k mod M оказалось равно 0, то нужно найти ближайшего живого, считая назад, либо (что то же самое) M-го, считая вперед. Через N - 1 шаг останется один человек.

Пример программы на Паскале:

```
const nmax=500;
var a: array[1..nmax] of integer;
    i,j: integer;
    k,n: integer;
    u: integer;
begin
read(n,k);
for i := 1 to n do a[i]:=i;
i := 1;
while n>1 do begin
  i := (i+k-1) mod n;
  if i=0 then i:=n;
  for j := i to n-1 do a[j]:=a[j+1];
  n:=n-1;
end;
writeln(a[1]);
end.
```

input.txt	output.txt
2 1	2
5 2	3
7 5	6
50 25	5
77 28	50
100 99	88
256 2	1
498 68	193
10 3	4
8 3	7