9-11 класс

1. «Страусы и жирафы» (100 баллов)

```
const FNameIn = 'input.txt';
      FNameOut = 'output.txt';
var a, b, s, g:int64;
    fin, fout:text;
begin
assign(fin, FNameIn);
reset(fin);
readln(fin, a, b);
close(fin);

assign(fout, FNameOut);
rewrite(fout);
if (a mod 2 <> 0) or (b mod 2 <> 0) then
begin
    writeln(fout, -1, ' ', -1);
end
else
begin
    s := a - b div 2;
    g := (b - a) div 2;
    if (s < 0) or (g < 0) then
    begin
        writeln(fout, -1, ' ', -1);
    end
    else
    begin
        writeln(fout, s, ' ', g);
    end;
end;
close(fout);
end.
```

Обозначим количество страусов s, а количество жирафов g. Так как у каждого животного два глаза, то $2s+2g=a$. Из того, что у страуса две ноги, а у жирафа 4, следует, что $2s+4g=b.$

Решим полученную систему уравнений. Умножим первое уравнение на 2:

$$\begin{cases} 4s+4g=2a \\ 2s+4g=b \end{cases}$$

Вычтем из первого уравнения второе: $2s=2a-b$. Откуда $s=a-\frac{b}{2}$.

Теперь из первоначального второго уравнения вычтем первое: $2g=b-a.$ Откуда

$$g=\frac{b-a}{2}.$$

Найти количество животных нельзя, если a или b нечетное, или найденные значения s или g получились отрицательные.

Разбиение тестов по группам

Номер группы	Номера тестов	Баллы за группу тестов	Баллы за тест
0 (тесты из условия)	01-04	0	0
1	05-18	70	5
2	19-24	30	5

2. «Катание на лифте» (100 баллов)

```
const FNameIn = 'input.txt';
      FNameOut = 'output.txt';
```

```
var fin, fout: text;
   n, m, i, j, x1, x2: integer;
   name: string;
   names_all: array of string;
   is_up: boolean;
begin
assign(fin, FNameIn); reset(fin);
readln(fin, n);
for i := 1 to n do
begin
    readln(fin, name);
    readln(fin, m);
    readln(fin, x1);
    is_up := true;
    for j := 1 to m - 1 do
    begin
        readln(fin, x2);
        if x1 > x2 then
            is_up := false;
        x1 := x2;
    end;
    if is_up then
    begin
        setlength(names_all, length(names_all) + 1);
        names_all[length(names_all) - 1] := name;
    end;
end;
assign(fout, FNameOut);
rewrite(fout);
for i := 0 to length(names_all) - 1 do
begin
    writeln(fout, names_all[i]);
end;
close(fout);
end.
```

В задаче требуется среди всех последовательностей чисел (этажи, на которых останавливался каждый из детей) выбрать те, которые являются возрастающими. При решении можно записывать каждую последовательность в отдельный массив, а можно это делать по ходу считывания данных, как приведено в авторском решении.

Разбиение тестов по группам

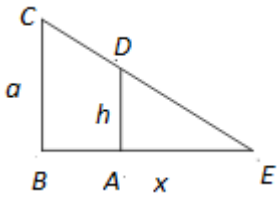
Номер группы	Номера тестов	Баллы за группу тестов	Баллы за тест
0 (тесты из условия)	01	0	0
1	02-11	50	5
2	12-21	50	5

3. «Тень на плетень» (100 баллов)

```
const FNameIn = 'input.txt';
      FNameOut = 'output.txt';
var fin, fout:text;
   x1, h, x2, a, x: real;
begin
assign(fin, FNameIn); reset(fin);
read(fin,x1,h,x2,a);
close(fin);
x:=h/(a-h)*abs(x2-x1);
assign(fout, FNameOut);rewrite(fout);
writeln(fout,x:7:3);
```

```
      close(fout);
      end.
```

Это модификация задачи из банка задач ОГЭ по математике.
Введем обозначения:



$BC = a$ м – высота фонаря, $AD = h$ м – рост человека. Расстояние между точками *A* и *B* равно $|x2 - x1|$.
Пусть длина тени равна *x* м.

Треугольники *BCE* и *ADE* подобны по двум углам. Откуда $\frac{BC}{AD} = \frac{BE}{AE}$, в наших обозначениях $\frac{a}{h} = \frac{x + AB}{x}$

. Следовательно, $x = \frac{h}{a - h} \cdot AB$

Разбиение тестов по группам

Номер группы	Номера тестов	Баллы за группу тестов	Баллы за тест
0 (тесты из условия)	01, 02	0	0
1	03-05	30	10
2	06-12	70	10

4. «Перекодирование черно-белой картинки» (100 баллов)

```
const FNameIn = 'input.txt';
      FNameOut = 'output.txt';
      Nmax = 100;

var fin, fout: text;
    s: string; // данная строка
    let: char; // буква в кодировке
    dig: integer; // цифра в кодировке
    ans: string; // ответ в виде строки
    n: integer; // количество точек в изображении
    w, h: integer; // ширина и высота картинки
    i, j :integer;

begin
    assign(fin, FNameIn);
    reset(fin);
    readln(fin, s);
    close(fin);
    // Декодирование картинки в строку
    ans := '';
    for i := 1 to length(s) do
        if (s[i] = 'B') or (s[i] = 'W') then
        begin
          if dig  = 1 then
```

```
            ans := ans + let;
          if s[i] = 'B' then
            let := '0';
          if s[i] = 'W' then
            let := '1';
          dig := 1;
        end
        else
        begin
          dig := ord(s[i]) - ord('0');
          for j := 1 to dig do
            ans := ans + let;
          dig := 0;
        end;
        if dig = 1 then
           ans := ans + let;

    // размеры картинки
    n := length(ans);
    w := trunc(sqrt(n));
    while n mod w <> 0 do
      w := w - 1;
    h := n div w;
       // запись ответа в файл
    assign(fout, FNameOut);
    rewrite(fout);
    writeln(fout, w, ' ', h);
    for i := 1 to w do
    begin
        s := copy(ans, 1, h);
        delete(ans, 1, h);
        writeln(fout, s);
    end;
    close(fout);
end.
```

Данная задача является технической. Необходимо декодировать полученную строку. Каждую букву «B» необходимо заменить «0» и повторить его такое количество раз, которое указано следом за буквой. Аналогично, каждую букву «W» необходимо заменить «1» и повторить ее такое количество раз, которое указано следом за буквой.

После декодирования нужно определить размеры изображения. Длина полученной строки будет равна количеству точек в изображении. Так как в задаче сказано, что размеры должны быть максимально приближены к квадрату, то нужно найти первый делитель, который будет равен или меньше корня из длины полученной строки. Это будет высота изображения *h*. А частное от деления длины полученной строки на высоту – ширина изображения *w*.

Останется только аккуратно вывести элементы изображения в формате *h* строк по *w* символов в каждой.

Разбиение тестов по группам

Номер группы	Номера тестов	Баллы за группу тестов	Баллы за тест
0 (тесты из условия)	01-03	0	0
1	04-13	30	3
2	14-27	70	5

5. «Скачки за руку и сердце принцессы» (100 баллов)

```
const FNameIn = 'input.txt';
      FNameOut = 'output.txt';
type mas1 = array of integer;
```

```
var fin, fout: text;
    n, m:integer; // количество вершин и ребер
    k: integer; // длина цикла
    g: array[1..Nmax] of mas1; // списки
                               // смежных вершин
    mark: array[1..Nmax] of integer; // массив
                               // меток для обхода
    put: array of integer; // путь
    is_cikl: boolean; // существует ли цикл
    is_beg_cikl: boolean; // дошли до начала цикла
    nach_cikl: integer; // начало цикла
    a, b: integer; // ребро
    i, j:integer;


    procedure dfs(v, prev:integer);
    // поиск цикла
        var i, u: integer;
    begin
    mark[v] := 1;
    for i := 0 to length(g[v]) - 1 do
      if not is_cikl then
      begin
          u := g[v][i];
          if mark[u] = 0 then
              dfs(u, v)
          else
            if (mark[u] = 1) and (u <> prev) then
            begin
               is_cikl := true;
               nach_cikl := u;
            end;
      end;

if is_cikl then
       begin
           if not is_beg_cikl then
           begin
               setlength(put, length(put) + 1);
               put[length(put) - 1] := v;
               if v = nach_cikl then
                   is_beg_cikl := true;
           end;
       end;
       mark[v] := 2;
    end;


begin
    assign(fin, FNameIn);
    reset(fin);
    readln(fin, n, m);
    for i :=1 to m do
    begin
        readln(fin, a, b);
        setlength(g[a], length(g[a]) + 1);
        g[a][length(g[a]) - 1] := b;
        setlength(g[b], length(g[b]) + 1);
        g[b][length(g[b]) - 1] := a;
    end;
    close(fin);

    // Поиск цикла
    for i := 1 to n do
```

```
    mark[i] := 0;
is_cikl := false;
is_beg_cikl := false;
for i := 1 to n do
    if (mark[i] = 0) and not is_cikl then
        dfs(i, 0);


// Запись ответа в файл
assign(fout, FNameOut);
rewrite(fout);
if not is_cikl then
begin
    writeln(fout, -1);
end
else
begin
    k := length(put);
    writeln(fout, k);
    writeln(k);
    for i := 0 to k - 1 do
    begin
        write(fout, ' ', put[i]);
    end;
    writeln(fout);
end;
close(fout);
end.
```

Задача на поиск цикла в графе.
Площади в столице являются вершинами графа. Дороги, которые соединяют площади, являются ребрами графа. Граф неориентированный.
Если в графе есть обратные ребра, то в графе есть цикл.
Разобьем все ребра на группы:

- ребра дерева – ребра, по которым проходил обход в глубину;
- прямые ребра – ребра, ведущие в вершины, которые располагаются в дереве поиска ниже;
- обратные ребра – ребра, ведущие в вершины, которые располагаются в дереве поиска выше. Обратное ребро ведет в посещенную вершину, из которой мы еще не вышли;
- перекрестные ребра – ребра, которые пересекают разные ветки дерева. Перекрестное ребро ведет в посещенную вершину, из которой мы уже вышли.

Прямое и перекрестное ребра циклов дать не могут. Ребра дерева сами по себе цикла образовывать не могут тоже.
Запустим обход в глубину. У каждой вершины должно быть три состояния: белая (0) – вершина не была посещена, серая (1) – вершина посещена, но ее обработка еще не закончена, черная (2) – обработка вершины завершена.
Вначале все вершины белые. При вызове функции обхода вершина становится серой. Далее мы ищем смежные с ней вершины, которые отличаются от той, из которой мы пришли. Если мы нашли белую с ней смежную вершину, то вызываем обход в глубину от нее. Если мы нашли серую сменную с ней вершину, то значит, в графе есть цикл. При этом все вершины, от которых в данный момент вызвана функция обхода в глубину и не завершена, и образуют цикл. Необходимо просто все их завершить и получить ответ. При выходе из функции обхода вершину нужно покрасить в черный цвет.
Так как граф может быть несвязным, функцию обхода нужно вызывать от всех непомеченных вершин, пока не будет найден цикл или пока не будет закончен перебор вершин.
Программа участника должна найти цикл в графе. Если в данном графе существует несколько циклов, то достаточно указать любой из них (не обязательно совпадающий с тем, который находит авторское решение задачи).

Разбиение тестов по группам

Номер группы	Номера тестов	Баллы за группу тестов	Баллы за тест
0 (тесты из условия)	01-02	0	0

1	02-17	60	4
2	18-22	20	4
3	23-27	20	4