ЕГЭ 2012
Информатика и ИКТ
Консультация №4

Задание С1

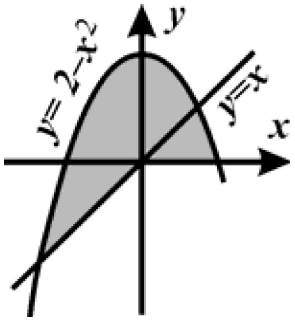
- Задача относится к повышенному уровню сложности
- Программа содержит некоторое количество условий (циклов и массивов нет)
- Необходимо найти ошибку в программе (смысловую, а не синтаксическую)
- Изменить программу для устранения случаев неправильной работы

Пример . Требовалось написать программу, которая вводит с клавиатуры координаты точки на плоскости (**x**, **y** – действительные числа) и определяет принадлежность точки заштрихованной области, включая ее границы.

Программист торопился и написал программу неправильно.

Бейсик	Паскаль	Си

INPUT x, y
IF y>=x THEN
IF y>=0 THEN
IF y<=2-x*x THEN
PRINT "принадлежит"
ELSE
PRINT "не принадлежит"
END IF
END IF
END IF
END
	var x,y: real;
	begin
	readln(x,y);
	if y>=x then
	  if y>=0 then
	  if y<=2-x*x then
	  write('принадлежит')
	  else
	  write('не принадлежит')
	end.
		void main(void){
		float x,y;
		scanf("%f %f",&x,&y);
		if (y>=x)
		if (y>=0)
		if (y<=2-x*x)
		printf("принадлежит");
		else
		printf("не принадлежит");
		}



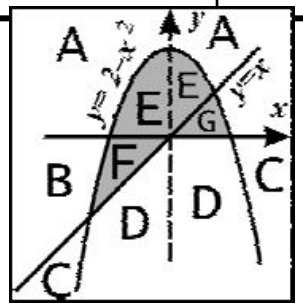
Последовательно выполните следующее.

1. Перерисуйте и заполните таблицу, которая показывает, как работает программа при аргументах, принадлежащих различным областям (A, B, C, D, E, F и G). Точки, лежащие на границах областей, отдельно не рассматривать.

В столбцах условий укажите "да", если условие выполнится, "нет" если условие не выполнится, "—" (прочерк), если условие не будет проверяться, «не изв.», если программа ведет себя по-разному для разных значений, принадлежащих данной области. В столбце "Программа выведет" укажите, что программа выведет на экран. Если программа ничего не выводит, напишите "—" (прочерк). Если для разных значений, принадлежащих области, будут выведены разные тексты, напишите «не изв». В последнем столбце укажите "да" или "нет".

2. Укажите, как нужно доработать программу, чтобы не было случаев ее неправильной работы. (Это можно сделать несколькими способами, достаточно указать любой способ доработки исходной программы.)

Область	**y>=x?**	**y>=0?**	**y<=2-x*x?**	**вывод**	**верно?**
A					
B					
C					
D					
E					
F					
G					



Бейсик	Паскаль	Си

IF y>=x THEN
IF y>=0 THEN
IF y<=2-x*x THEN
PRINT "принадлежит"
ELSE
PRINT "не принадлежит"
END IF
END IF
END IF
END
	if y>=x then
	  if y>=0 then
	  if y<=2-x*x then
	  write('принадлежит')
	  else
	  write('не принадлежит')
	end.
		if (y>=x)
		if (y>=0)
		if (y<=2-x*x)
		printf("принадлежит");
		else
		printf("не принадлежит");
		}

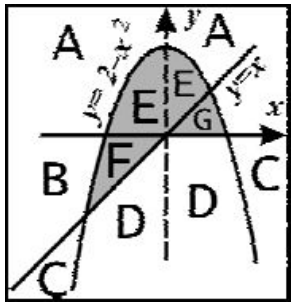
```
var x,y: real;
begin
readln(x,y);
if y>=x then
  if y>=0 then
   if y<=2-x*x then
    write('принадлежит')
   else
    write('не принадлежит')
end.
```

истина

ложь

В столбцах условий укажите "да", если условие выполнится, "нет" если условие не выполнится, "—“ (прочерк), если условие не будет проверяться, «не изв.», если программа ведет себя по-разному для разных значений, принадлежащих данной области. В столбце "Программа выведет" укажите, что программа выведет на экран. Если программа ничего не выводит, напишите "—" (прочерк). Если для разных значений, принадлежащих области, будут выведены разные тексты, напишите «не изв». В последнем столбце укажите "да" или "нет".

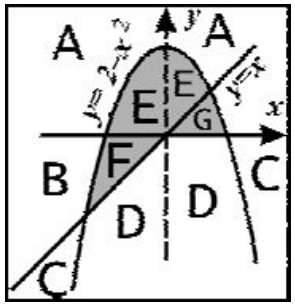
```
if y>=x then
  if y>=0 then
    if y<=2-x*x then
      write('принадлежит')
    else
      write('не принадлежит')
```



Область	y>=x?	y>=0?	y<=2-x*x?	вывод	верно?
A	Да	Да	Нет	Не принадлежит	да
B					
C					
D					
E					
F					
G					

В столбцах условий укажите "да", если условие выполнится, "нет" если условие не выполнится, "—“ (прочерк), если условие не будет проверяться, «не изв.», если программа ведет себя по-разному для разных значений, принадлежащих данной области. В столбце "Программа выведет" укажите, что программа выведет на экран. Если программа ничего не выводит, напишите "—" (прочерк). Если для разных значений, принадлежащих области, будут выведены разные тексты, напишите «не изв». В последнем столбце укажите "да" или "нет".

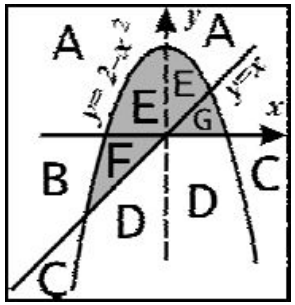
```
if y>=x then
  if y>=0 then
    if y<=2-x*x then
      write('принадлежит')
    else
      write('не принадлежит')
```



Область	y>=x?	y>=0?	y<=2-x*x?	вывод	верно?
A	Да	Да	Нет	Не принадлежит	да
B	Да	Нет	-	-	нет
C					
D					
E					
F					
G					

В столбцах условий укажите "да", если условие выполнится, "нет" если условие не выполнится, "—“ (прочерк), если условие не будет проверяться, «не изв.», если программа ведет себя по-разному для разных значений, принадлежащих данной области. В столбце "Программа выведет" укажите, что программа выведет на экран. Если программа ничего не выводит, напишите "—" (прочерк). Если для разных значений, принадлежащих области, будут выведены разные тексты, напишите «не изв». В последнем столбце укажите "да" или "нет".

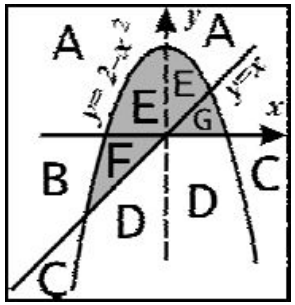
```
if y>=x then
  if y>=0 then
    if y<=2-x*x then
      write('принадлежит')
    else
      write('не принадлежит')
```



Область	y>=x?	y>=0?	y<=2-x*x?	вывод	верно?
A	Да	Да	Нет	Не принадлежит	да
B	Да	Нет	-	-	нет
C	Нет	-	-	-	нет
D	Нет	-	-	-	нет
E					
F					
G					

В столбцах условий укажите "да", если условие выполнится, "нет" если условие не выполнится, "—“ (прочерк), если условие не будет проверяться, «не изв.», если программа ведет себя по-разному для разных значений, принадлежащих данной области. В столбце "Программа выведет" укажите, что программа выведет на экран. Если программа ничего не выводит, напишите "—" (прочерк). Если для разных значений, принадлежащих области, будут выведены разные тексты, напишите «не изв». В последнем столбце укажите "да" или "нет".

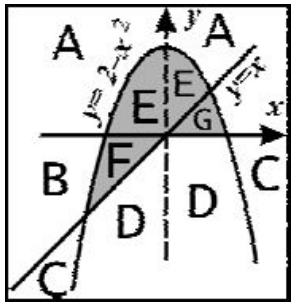
```
if y>=x then
  if y>=0 then
    if y<=2-x*x then
      write('принадлежит')
    else
      write('не принадлежит')
```



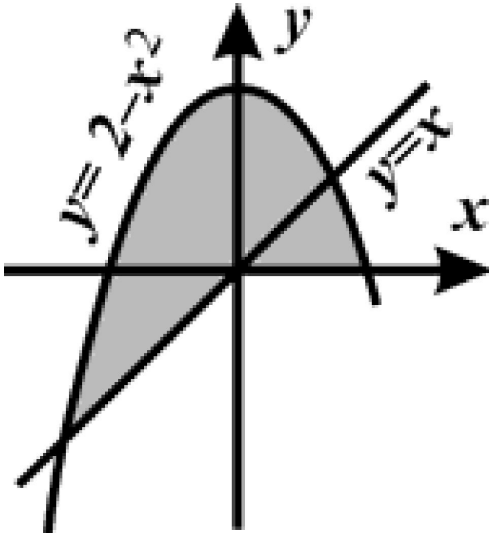
Область	y>=x?	y>=0?	y<=2-x*x?	вывод	верно?
A	Да	Да	Нет	Не принадлежит	да
B	Да	Нет	-	-	нет
C	Нет	-	-	-	нет
D	Нет	-	-	-	нет
E	Да	Да	Да	Принадлежит	да
F					
G					

В столбцах условий укажите "да", если условие выполнится, "нет" если условие не выполнится, "—“ (прочерк), если условие не будет проверяться, «не изв.», если программа ведет себя по-разному для разных значений, принадлежащих данной области. В столбце "Программа выведет" укажите, что программа выведет на экран. Если программа ничего не выводит, напишите "—" (прочерк). Если для разных значений, принадлежащих области, будут выведены разные тексты, напишите «не изв». В последнем столбце укажите "да" или "нет".

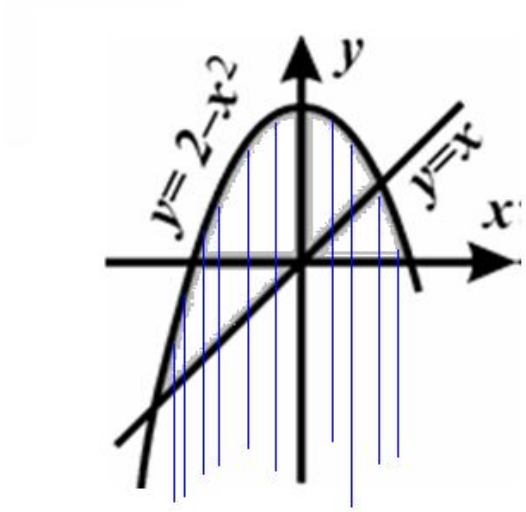
```
if y>=x then
  if y>=0 then
    if y<=2-x*x then
      write('принадлежит')
    else
      write('не принадлежит')
```

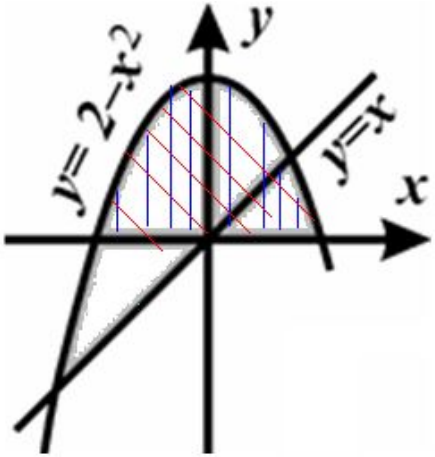


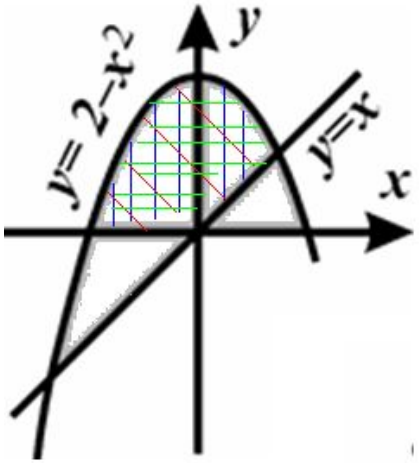
Область	y>=x?	y>=0?	y<=2-x*x?	вывод	верно?
A	Да	Да	Нет	Не принадлежит	да
B	Да	Нет	-	-	нет
C	Нет	-	-	-	нет
D	Нет	-	-	-	нет
E	Да	Да	Да	Принадлежит	да
F	Да	Нет	-	-	нет
G	Нет	-	-	-	нет

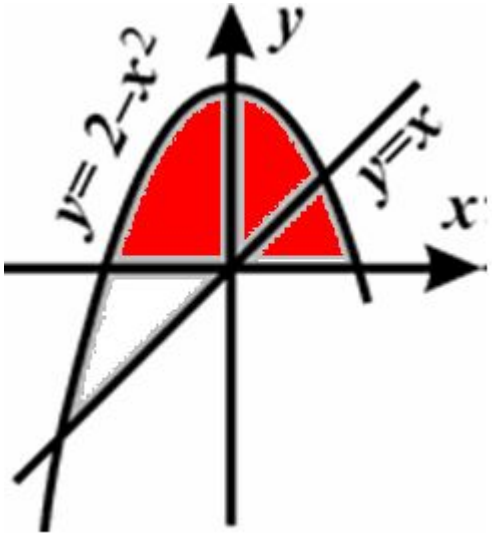


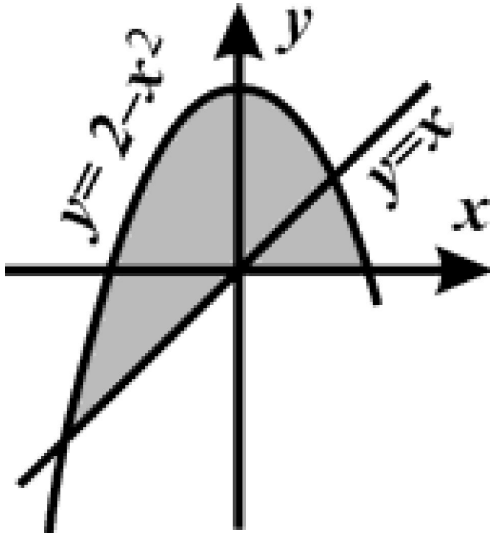
(y<=2-x*x) и (y>=0) и (y>=x)

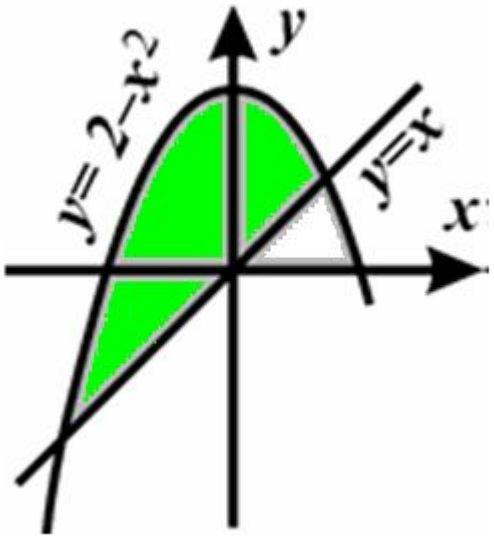






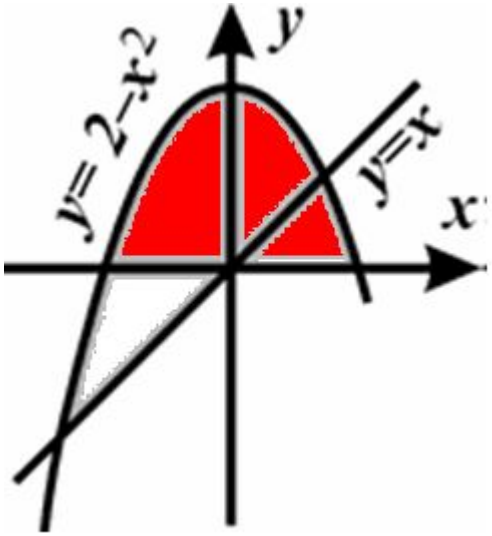


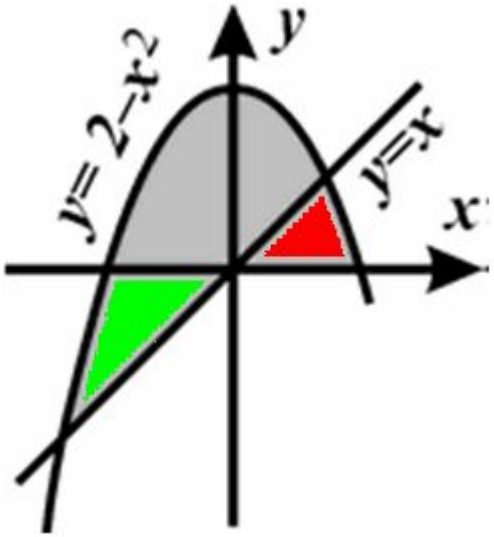


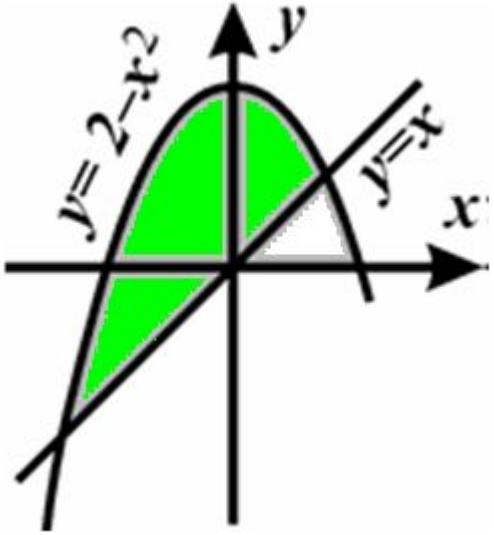


(y<=2-x*x) и (y>=0)

(y<=2-x*x) и y>=x







(y<=2-x*x) и (y>=0)

(y<=2-x*x) и (y>=x)

(y<=2-x*x) и (y>=0) или (y<=2-x*x) и (y>=x)

(y<=2-x*x) и ((y>=0) или (y>=x))

Бейсик	Паскаль	Си

INPUT x, y
IF ((y>=x)or(y>=0))and(y<=2-x*x)
 THEN
  PRINT "принадлежит"
ELSE
  PRINT "не принадлежит"
END IF
END
	var x,y: real;
	begin
	readln(x,y);
	if ((y>=x)or(y>=0))and(y<=2-x*x)
	  then
	  write('принадлежит')
	  else
	  write('не принадлежит')
	end.
		void main(void){
		float x,y;
		scanf("%f %f",&x,&y);
		if (((y>=x) II(y>=0)) && (y<=2-x*x))
		  printf("принадлежит");
		 else
		  printf("не принадлежит");
		}

Задание С2

- Написать короткую (≈15 строк) простую программу (например, обработка массива)
- Программа линейной обработки входной последовательности

Пример . **(C2)** Дан целочисленный массив из 30 элементов. Элементы массива могут принимать целые значения от –1000 до 1000. Опишите на русском языке или на одном из языков программирования алгоритм, позволяющий найти и вывести минимальное значение среди элементов массива, которые имеют чётное значение и не делятся на три. Гарантируется, что в исходном массиве есть хотя бы один элемент, значение которого чётно и не кратно трем.

Исходные данные объявлены так, как показано ниже. Запрещается использовать переменные, не описанные ниже, но разрешается не использовать часть из них.

Паскаль	Бейсик	Си

const
N=30;
var
a: array [1..N] of integer;
i, j, min: integer;
begin
for i:=1 to N do readln(a[i]);
	N=30
	DIM A(N) AS INTEGER
	DIM I, Y, MIN AS INTEGER
	FOR I = 1 TO N
	INPUT A(I)
	NEXT I
		#include <stdio.h>
		#define N 30
		void main(void)
		{int a[N];
		int i, y, min;
		for (i=0; i<N; i++)
		scanf("%d", &a[i]);

Паскаль

```
 const
N=30;
var
a: array [1..N] of integer;
i, j, min: integer;
begin
for i:=1 to N do readln(a[i]);
min:=1000;
for i:=1 to N do
if (a[i] mod 2=0) and (a[i] mod 3<>0)and(a[i]<min)
  then min:=a[i];
writeln(min);
end.
```

Бейсик

```
N=30
DIM A(N) AS INTEGER
DIM I, Y, MIN AS INTEGER
FOR I = 1 TO N
INPUT A(I)
NEXT I
MIN = 1000
FOR I = 1 TO N
IF (A(I) MOD 2 = 0) AND (A(I) MOD 3<>= 0) AND (A(I) < MIN) THEN
MIN = A(I)
ENDIF
NEXT I
PRINT MIN
```

Си

```
#include <stdio.h>
#define N 30
void main(void)
{int a[N];
int i, y, min;
for (i=0; i<N; i++)
scanf("%d", &a[i]);
min=1000;
for (i=0; i<N; i++)
if (a[i]%2==0 && a[i]%2!=0 && a[i]<min)
        min=a[i];
printf("%d", min);
}
```

Пример. (С2) Опишите на русском языке или одном из языков программирования алгоритм вычисления количества наибольших элементов в заданном целочисленном массиве из 30 элементов.

Паскаль	Бейсик	Си

var
a:array[1..30] of integer;
i, k, max: integer;
begin
for i:=1 to 30 do
read(a[i]);
	DIM I, K, max, A(30) AS INTEGER
	FOR I=1 to 30
	INPUT A(I)
	NEXT I
		void main(void)
		{ int i, k, max, a[30];
		for(i=0;i<30;i++)
		scanf(“%d",&a[i]);

Паскаль	Бейсик	Си

var
a:array[1..30] of integer;
i, k, max: integer;
begin
for i:=1 to 30 do
read(a[i]);
max:=a[1]; k:= 1;
	DIM I, K,max,A(30) AS INTEGER
	FOR I=1 to 30
	INPUT A(I)
	NEXT I
	max=A(1): k= 1
		void main(void)
		{ int i, k, max, a[30];
		for(i=0;i<30;i++)
		scanf(“%d",&a[i]);
		max=a[0]; k= 1;

Паскаль	Бейсик	Си

var
a:array[1..30] of integer;
i, k, max: integer;
begin
for i:=1 to 30 do
read(a[i]);
max:=a[1]; k:= 1;
for i:=2 to 30 do
if a [i] > max
then
	DIM I, K,max,A(30) AS INTEGER
	FOR I=1 to 30
	INPUT A(I)
	NEXT I
	max=A(1): k= 1
	FOR I = 2 TO 30
	IF A(I) >max THEN
		void main(void)
		{ int i, k, max, a[30];
		for(i=0;i<30;i++)
		scanf(“%d",&a[i]);
		max=a[0]; k= 1;
		for(i=1;i<30;i++)
		if (a [i] > max)

Паскаль	Бейсик	Си

var
a:array[1..30] of integer;
i, k, max: integer;
begin
for i:=1 to 30 do
read(a[i]);
max:=a[1]; k:= 1;
for i:=2 to 30 do
if a [i] > max
then
begin
max:= a[i] ; k:=1
end
	DIM I, K,max,A(30) AS INTEGER
	FOR I=1 to 30
	INPUT A(I)
	NEXT I
	max=A(1): k= 1
	FOR I = 2 TO 30
	IF A(I) >max THEN
	max = A(I): K=1
		void main(void)
		{ int i, k, max, a[30];
		for(i=0;i<30;i++)
		scanf(“%d",&a[i]);
		max=a[0]; k= 1;
		for(i=1;i<30;i++)
		if (a [i] > max)
		{max= a[i] ; k=1;}

Паскаль

```
 var
a:array[1..30] of integer;
 i, k, max: integer;
begin
  for i:=1 to 30 do
read(a[i]);
  max:=a[1]; k:= 1;
  for i:=2 to 30 do
     if a [i] > max
        then
          begin
            max:=  a[i] ; k:=1
          end
         else
             if a[i]=max
                  then  k:=k+1;
```

Бейсик

```
DIM I, K,max,A(30) AS INTEGER
 FOR I=1 to 30
 INPUT A(I)
NEXT I
max=A(1): k= 1
FOR I = 2 TO 30
IF A(I) >max THEN
        max =  A(I): K=1
     ELSE
        IF A(I)=max THEN K=K+1
ENDIF
```

Си

```
void main(void)
{ int i, k, max, a[30];
  for( i=0;i<30;i++)
scanf("%d",&a[i]);
max=a[0]; k= 1;
 for( i=1;i< 30;i++)
   if (a [i] > max)
    {max= a[i] ; k=1;}
      else
          if (a[i]==max)
                k++;
```

Паскаль

```
 var
a:array[1..30] of integer;
 i, k, max: integer;
begin
  for i:=1 to 30 do
read(a[i]);
  max:=a[1]; k:= 1;
  for i:=2 to 30 do
     if a [i] > max
        then
          begin
            max:=  a[i] ; k:=1
          end
         else
             if a[i]=max
                  then  k:=k+1;
write(k)
end.
```

Бейсик

```
DIM I, K, MAX,A(30) AS INTEGER
 FOR I=1 to 30
 INPUT A(I)
NEXT I
MAX=A(1): k= 1
FOR I = 2 TO 30
IF A(I) >MAX THEN
        MAX =  A(I): K=1
     ELSE
        IF A(I)=MAX THEN K=K+1
ENDIF
NEXT I
PRINT K
END
```

Си

```
void main(void)
{ int i, k, max, a[30];
  for( i=0;i< 30;i++)
scanf(“%d",&a[i]);
max=a[0]; k= 1;
 for( i=1;i< 30;i++)
   if (a [i] > max)
    {max= a[i] ; k=1;}
      else
          if (a[i]==max)
                k++;
  printf(“%d”,k);
}
```

Паскаль

```
 var
a:array[1..30] of integer;
 i, k, max: integer;
begin
  for i:=1 to 30 do
read(a[i]);
  max:=a[1]; k:= 1;
  for i:=2 to 30 do
     if a [i] > max
        then
          begin
            max:=  a[i] ; k:=1
          end
         else
             if a[i]=max
                  then  k:=k+1;
write(k)
end.
```

Бейсик

```
DIM I, K, MAX,A(30) AS INTEGER
 FOR I=1 to 30
 INPUT A(I)
NEXT I
MAX=A(1): k= 1
FOR I = 2 TO 30
IF A(I) >MAX THEN
        MAX =  A(I): K=1
     ELSE
        IF A(I)=MAX THEN K=K+1
ENDIF
NEXT I
PRINT K
END
```

Си

```
void main(void)
{ int i, k, max, a[30];
  for( i=0;i< 30;i++)
scanf("%d",&a[i]);
max=a[0]; k= 1;
 for( i=1;i< 30;i++)
   if (a [i] > max)
    {max= a[i] ; k=1;}
      else
          if (a[i]==max)
                k++;
  printf("%d",k);
}
```

Пример. (С2) С клавиатуры вводят целые числа. Признак конца ввода - ввод нуля. Опишите на русском языке или одном из языков программирования алгоритм вычисления суммы тех чисел, которые кратны трем. Считаем, что хотя бы одно такое число в последовательности есть.

Паскаль	Бейсик	Си
var a, S: integer; begin S:=0;	DIM A,S AS INTEGER S=0	void main() { int a,S; S=0;

Паскаль	Бейсик	Си

var
a, S: integer;
begin
S:=0;
readln(a);
While (a<>0) do
Begin
	DIM A,S AS INTEGER
	S=0
	INPUT A
	WHILE A<> 0
		void main()
		{ int a,S;
		S=0;
		scanf(“%d",&a);
		while(a <>0)
		{

Паскаль	Бейсик	Си

var
a, S: integer;
begin
S:=0;
readln(a);
While (a<>0) do
Begin
  if (a mod 3 = 0) then
  S:=S+a;
	DIM A,S AS INTEGER
	S=0
	INPUT A
	WHILE A<> 0
	  IF A MOD 3 = 0 THEN
	  S = S+A
	  ENDIF
		void main()
		{ int a,S;
		S=0;
		scanf(“%d",&a);
		while(a <>0)
		{
		  if (a % 3 == 0)
		  S+=a;

Паскаль	Бейсик	Си

var
a, S: integer;
begin
S:=0;
readln(a);
While (a<>0) do
Begin
if (a mod 3 = 0) then
S:=S+a;
Readln(a)
End;
	DIM A,S AS INTEGER
	S=0
	INPUT A
	WHILE A<> 0
	IF A MOD 3 = 0 THEN
	S = S+A
	ENDIF
	INPUT A
	WEND
		void main()
		{ int a,S;
		S=0;
		scanf(“%d",&a);
		while(a <>0)
		{
		if (a % 3 == 0)
		S+=a;
		scanf(“%d",&a);
		}

Паскаль	Бейсик	Си

var
a, S: integer;
begin
S:=0;
readln(a);
While (a<>0) do
Begin
if (a mod 3 = 0) then
S:=S+a;
Readln(a)
End;
writeln(S)
end.
	DIM A,S AS INTEGER
	S=0
	INPUT A
	WHILE A<> 0
	IF A MOD 3 = 0 THEN
	S = S+A
	ENDIF
	INPUT A
	WEND
	PRINT S
	END
		void main()
		{ int a,S;
		S=0;
		scanf(“%d",&a);
		while(a <>0)
		{
		if (a % 3 == 0)
		S+=a;
		scanf(“%d",&a);
		}
		printf(“%d”,k);
		}

Пример С3.

У исполнителя Утроитель две команды, которым присвоены номера:

1. прибавь 1,

2. умножь на 3.

Первая из них увеличивает число на экране на 1, вторая – утраивает его. Программа для Утроителя – это последовательность команд. Сколько есть программ, которые число 1 преобразуют в число 17? Ответ обоснуйте.

Пример С3. У исполнителя Утроитель две команды, которым присвоены номера:

1. прибавь 1,

2. умножь на 3.

Решение.

1. Количество программ, которые преобразуют число 1 в 17 равно количеству программ, которые преобразуют число 17 в 1
2. Для каждого следующего числа рассмотрим, из какого числа оно может быть получено за одну команду исполнителя.

– Если число N не делится на три, то оно может быть получено
  - только из предыдущего N-1 с помощью команды **прибавь 1**.

– Если число N делится на три, то оно может быть получено
  - из предыдущего N-1 с помощью команды **прибавь 1**
  - из числа N/3 с помощью команды **умножь на 3**

Пример С3.

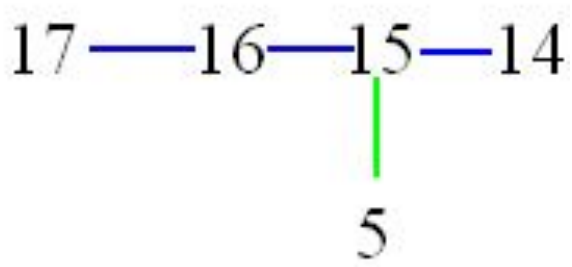
1. прибавь 1,

2. умножь на 3.

Решение.

- Если число N не делится на три, то оно может быть получено только из предыдущего N-1 с помощью команды **прибавь 1**.

- Если число N делится на три, то оно может быть получено из предыдущего N-1 с помощью команды **прибавь 1** из числа N/3 с помощью команды **умножь на 3**



Пример С3.

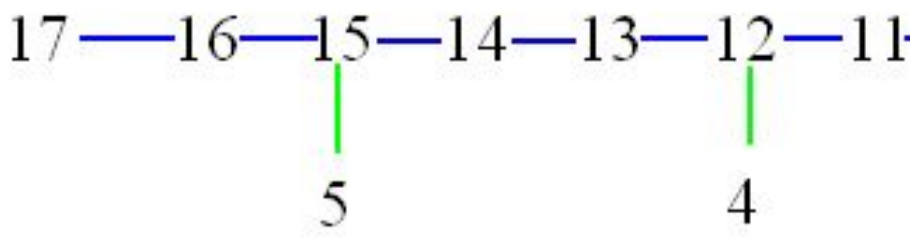
1. прибавь 1,

2. умножь на 3.

Решение.

Если число N не делится на три, то оно может быть получено только из предыдущего N-1 с помощью команды **прибавь 1**.

Если число N делится на три, то оно может быть получено
из предыдущего N-1 с помощью команды **прибавь 1**
из числа N/3 с помощью команды **умножь на 3**



Пример С3.

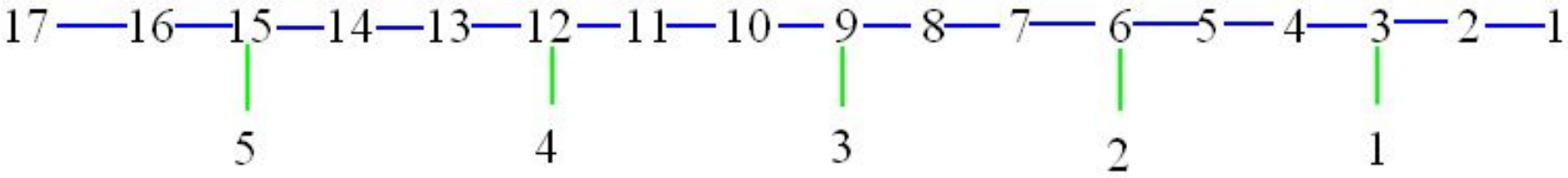
1. прибавь 1,

2. умножь на 3.

Решение.

Если число N не делится на три, то оно может быть получено только из предыдущего N-1 с помощью команды **прибавь 1**.

Если число N делится на три, то оно может быть получено
из предыдущего N-1 с помощью команды **прибавь 1**
из числа N/3 с помощью команды **умножь на 3**



Пример С3.

1. прибавь 1,

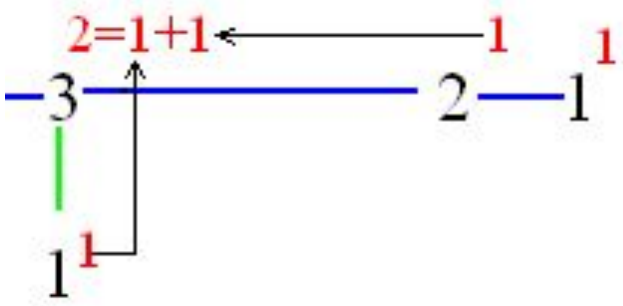
2. умножь на 3.

Решение.

Для каждого из чисел определим, сколько программ исполнителя существует для его получения.

- Если число N не делится на три, то количество искомых программ для такого числа равно количеству программ для числа N-1.

- Если число N делится на три, то количество искомых программ равно сумме количеств программ для числа N-1 и количества программ для числа N/3



Пример С3.

1. прибавь 1,

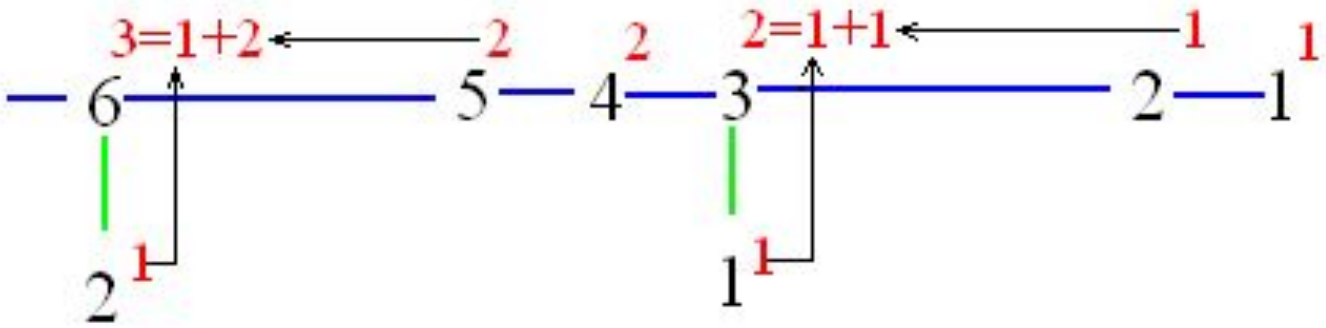
2. умножь на 3.

Решение.

Для каждого из чисел определим, сколько программ исполнителя существует для его получения.

– Если число N не делится на три, то количество искомых программ для такого числа равно количеству программ для числа N-1.

– Если число N делится на три, то количество искомых программ равно сумме количеств программ для числа N-1 и количества программ для числа N/3



Пример С3.

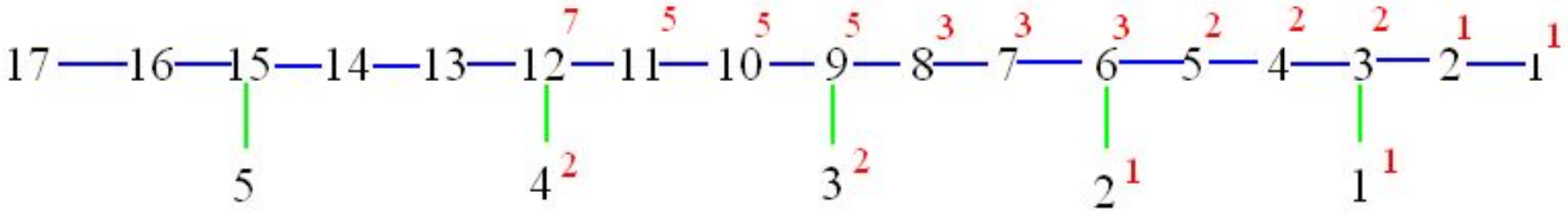
1. прибавь 1,

2. умножь на 3.

Решение.

Для каждого из чисел определим, сколько программ исполнителя существует для его получения.

- Если число N не делится на три, то количество искомых программ для такого числа равно количеству программ для числа N-1.
- Если число N делится на три, то количество искомых программ равно сумме количеств программ для числа N-1 и количества программ для числа N/3



Пример С3.

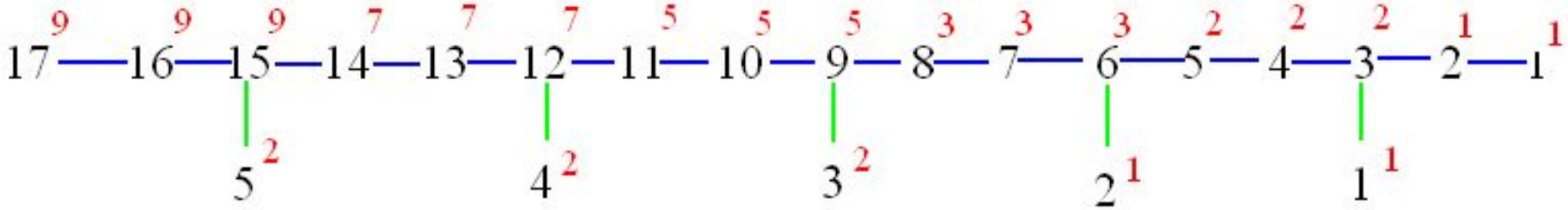
1. прибавь 1,

2. умножь на 3.

Решение.

Для каждого из чисел определим, сколько программ исполнителя существует для его получения.

- Если число N не делится на три, то количество искомых программ для такого числа равно количеству программ для числа N-1.
- Если число N делится на три, то количество искомых программ равно сумме количеств программ для числа N-1 и количества программ для числа N/3



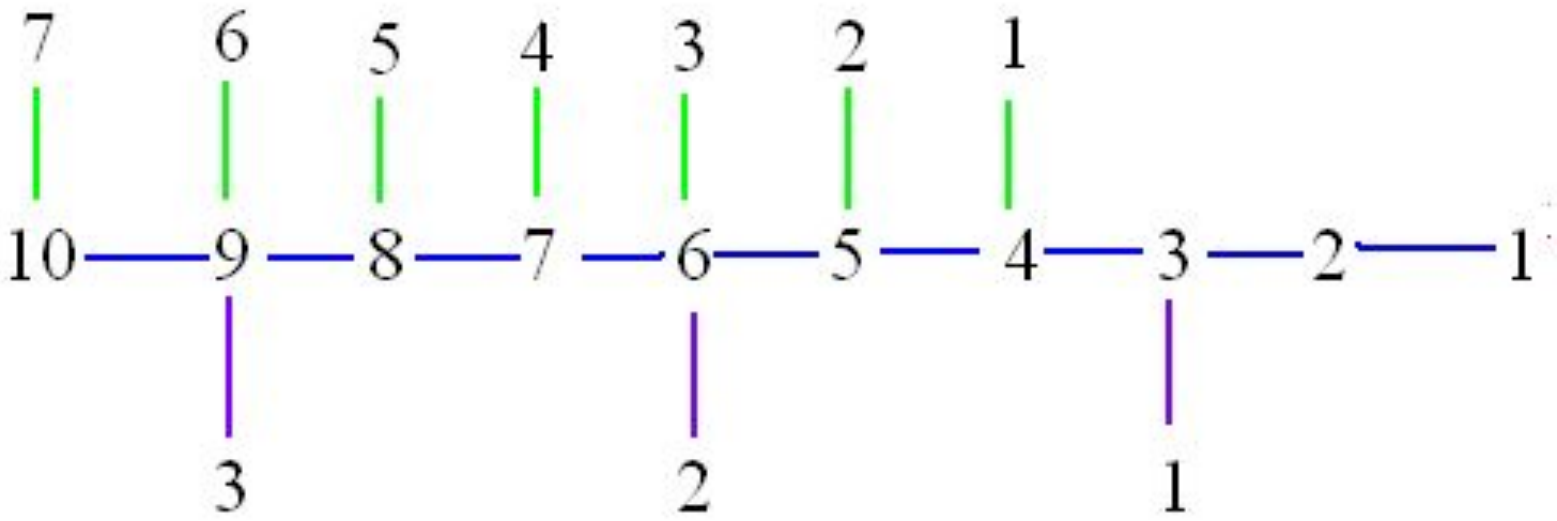
Ответ: 9

Пример С3.

У исполнителя Утроитель три команды, которым присвоены номера:

1. **прибавь 1,**
2. **прибавь 3,**
3. **умножь на 3.**

Сколько есть программ, которые число 1 преобразуют в число 10? Ответ обоснуйте.

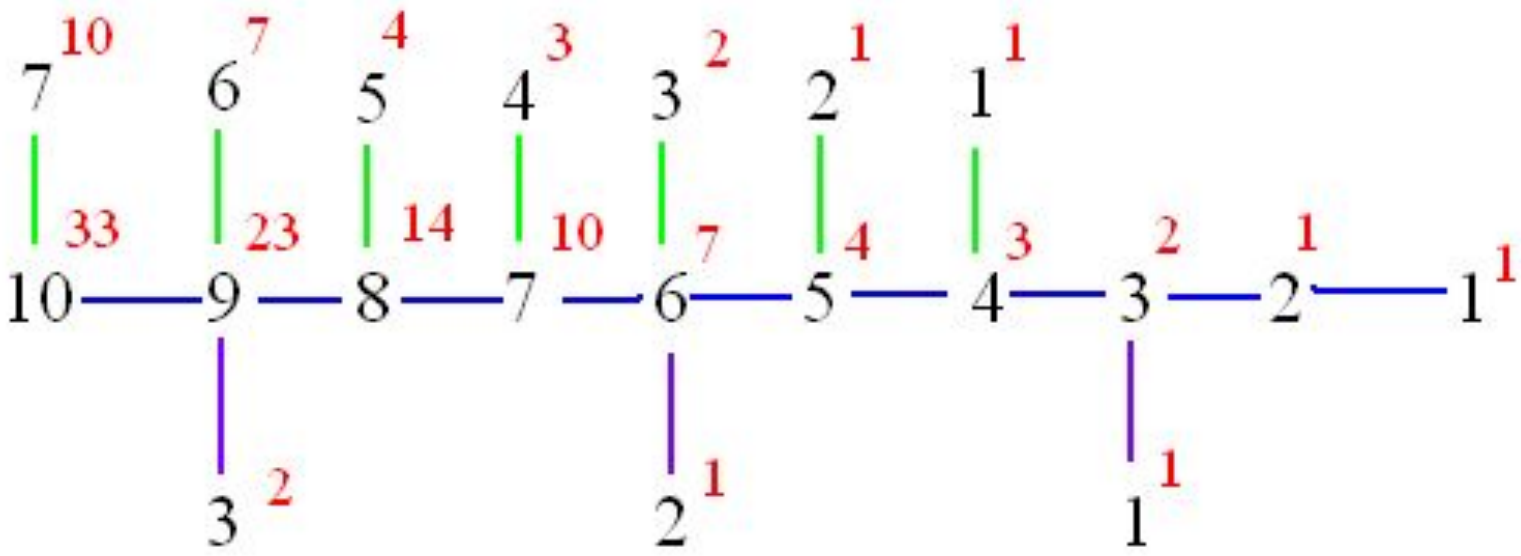


Пример С3.

У исполнителя Утроитель три команды, которым присвоены номера:

1. **прибавь 1,**
2. **прибавь 3,**
3. **умножь на 3.**

Сколько есть программ, которые число 1 преобразуют в число 10? Ответ обоснуйте.



Характеристика задания С4

- Нацелено на проверку умения создавать собственные программы для решения задач средней сложности.
- Проверяется умение составить алгоритм и написать законченную программу на одном из зыков программирования
- Оценивается в 4 балла

Типичная постановка задачи С4 содержит:

- Формат входных данных.
- Назначение программы, т.е. какую информацию программа должна извлечь из исходных данных и как их преобразовать.
- Формат выходных данных.
- Дополнительные условия и рекомендации.

Пример задачи

В командных олимпиадах по программированию для решения предлагается не больше 11 задач. Команда может решать предложенные задачи в любом порядке. Подготовленные решения команда посылает в единую проверяющую систему соревнований. Написать эффективную, в том числе по используемой памяти, программу, которая будет статистически обрабатывать пришедшие запросы, чтобы определить наиболее популярные задачи. Следует учитывать, что количество запросов в списке может быть очень велико. На вход программе в первой строке подаётся количество пришедших запросов N. В каждой из последующих N строк записано название задачи в виде текстовой строки. Длина строки не превосходит 100 символов, название может содержать буквы, цифры, пробелы и знаки препинания.

Пример входных данных:

6
A+B
Крестики-Нолики
Прямоугольник
Простой делитель
A+B
Простой делитель

Программа должна вывести список из трёх наиболее популярных задач с указанием количества запросов по ним. Если в запросах упоминаются менее трех задач, то выведите информацию об имеющихся задачах. Если несколько задач имеют ту же частоту встречаемости, что и третья по частоте встречаемости задача, их тоже нужно вывести.

Пример выходных данных для приведённого выше примера входных данных:

A+B 2
Простой делитель 2
Крестики-Нолики 1
Прямоугольник 1

Формат входных данных

На вход программе в первой строке подаётся количество пришедших запросов N.
В каждой из последующих N строк записано название задачи в виде текстовой строки. Длина строки не превосходит 100 символов, название может содержать буквы, цифры, пробелы и знаки препинания.

Пример входных данных:

6 **количество пришедших запросов**
A+B **название задачи**
Крестики-Нолики **название задачи**
Прямоугольник
Простой делитель
A+B
Простой делитель

Назначение программы:

Написать эффективную, в том числе по используемой памяти, программу, которая будет статистически обрабатывать пришедшие запросы, чтобы определить наиболее популярные задачи.

Формат выходных данных:

Программа должна вывести список из трёх наиболее популярных задач с указанием количества запросов по ним. Если в запросах упоминаются менее трех задач, то выведите информацию об имеющихся задачах. Если несколько задач имеют ту же частоту встречаемости, что и третья по частоте встречаемости задача, их тоже нужно вывести.

Дополнительные условия и рекомендации :

Написать эффективную, в том числе по используемой памяти, программу, которая будет статистически обрабатывать пришедшие запросы, чтобы определить наиболее популярные задачи. Следует учитывать, что количество запросов в списке может быть очень велико.

```
Var n, Num, i, j, t: integer;
Count: array[1..11] of integer;
s: string;
Names: array[1..11] of string;
Begin
Num:=0; {Число различных задач}
ReadLn(N);
for i:=1 to N do
  begin
     readLn(S); {считали задачу}
     j:=1;
     while (j<=Num) and (s<>Names[j])
            do j:=j+1;
     if j<=Num
          then Count[j]:=Count[j]+1
          else begin
                 Names[j]:=s;
                 Count[j]:=1;
                 Num:=Num+1
               end
  end;
```

```
for i:=Num downto 2 do
for j:=2 to i do if Count[j-1]<Count[j] then
begin
    t:=Count[j];
    Count[j]:=Count[j-1]; Count[j-1]:=t;
    s:=Names[j];
    Names[j]:=Names[j-1];
    Names[j-1]:=s;
end;

if Num >= 3 then j := 3 else j := Num;

i := 1;
while (i <= Num) and (Count[i] >= Count[j])
do
  begin
       WriteLn(Names[i], ' ', Count[i]);
        i := I + 1;
  end
end.
```

Критерии оценивания

- 4 балла – правильная и эффективная программа (допускается одна синтаксическая ошибка)
- 3 балла – правильная, но неэффективная программа или правильная в целом и эффективная программа, но неверно работающая в одном из частных случаев (допускается три синтаксических ошибки)
- 2 балла – программа работает в целом верно (допускается две логические, пять синтаксических ошибок)
- 1 балл –программа не удовлетворяет критериям 2 баллов (допускается четыре логические, семь синтаксических ошибок)