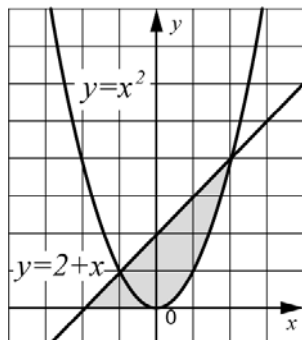


Критерии оценивания заданий с развёрнутым ответом

C1



Требовалось написать программу, при выполнении которой с клавиатуры считываются координаты точки на плоскости (x, y – действительные числа) и определяется принадлежность этой точки заданной закрашенной области (включая границы). Программист торопился и написал программу неправильно.

Бейсик

```
INPUT x, y
IF y<=2+x THEN
  IF y>=0 THEN
    IF y>=x*x THEN
      PRINT "принадлежит"
    ELSE
      PRINT "не принадлежит"
    ENDIF
  ENDIF
ENDIF
END
```

Паскаль

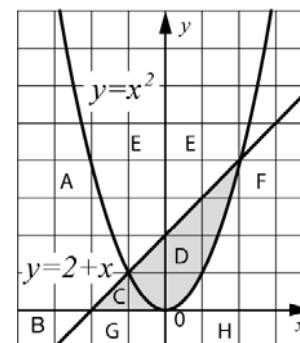
```
var x,y: real;
begin
  readln(x,y);
  if y<=2+x then
    if y>=0 then
      if y>=x*x then
        write('принадлежит')
      else
        write('не принадлежит')
      end
    end
  end.
end.
```

Си

```
void main(void){
  float x,y;
  scanf("%f%f", &x, &y);
  if (y<=2+x)
    if (y>=0)
      if (y>=x*x)
        printf("принадлежит");
      else
        printf("не принадлежит");
}
```

Алгоритмический
язык

```
алг
нач
  вещ x, y
  ввод x, y
  если y<=2+x то
    если y>=0 то
      если y>=x*x то
        вывод 'принадлежит'
      иначе
        вывод 'не принадлежит'
    все
  все
кон
```



Последовательно выполните следующее.

1. Перерисуйте и заполните таблицу, которая показывает, как работает программа при аргументах, принадлежащих различным областям (A, B, C, D, E, F, G и H).

Точки, лежащие на границах областей, отдельно не рассматривать.

Область	Условие 1 ($y \leq 2+x$)	Условие 2 ($y \geq 0$)	Условие 3 ($y \geq x*x$)	Программа выведет	Область обрабатывается верно
A					
B					
C					
D					
E					
F					
G					
H					

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.)

Элементы ответа:

1.

Область	Условие 1 ($y \leq 2+x$)	Условие 2 ($y \geq 0$)	Условие 3 ($y \geq x \cdot x$)	Программа выведет	Область обрабатывается верно
A	нет	—	—	—	нет
B	нет	—	—	—	нет
C	да	да	нет	не принадлежит	нет
D	да	да	да	принадлежит	да
E	нет	—	—	—	нет
F	да	да	нет	не принадлежит	да
G	да	нет	—	—	нет
H	да	нет	—	—	нет

2. Возможная доработка (Паскаль):

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

Возможны и другие способы доработки.

Пример:

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

Указания по оцениванию	Баллы
<p>Обратите внимание! В задаче требовалось выполнить три действия</p> <ol style="list-style-type: none"> 1. Заполнить таблицу. 2. Исправить ошибку в условном операторе. 3. Исправить ошибку, связанную с тем, что выбран неправильный набор условий. <p>Баллы за данное задание начисляются как сумма баллов за верное выполнение каждого действия.</p> <ol style="list-style-type: none"> 1. Действие по заполнению таблицы считается выполненным, если в таблице нет ошибок или ошибки присутствуют только в одной строке. 2. Исправление ошибки в условном операторе. Неправильное использование условного оператора приводит к тому, что при невыполнении первого или второго условия программа не выдает ничего (отсутствуют случаи ELSE). Исправлением этой ошибки может быть либо добавление случая ELSE к каждому условию IF, либо объединение всех условий IF в одно при помощи конъюнкции. В сложных случаях это действие считается выполненным, если программа выдает одно из двух сообщений «принадлежит» или «не принадлежит» для любых чисел x и y, и при этом программа не стала работать хуже, чем раньше, то есть для всех точек, для которых программа ранее выдавала верный ответ, доработанная программа также должна выдавать верный ответ. 3. Ошибка при выборе набора условий приводит к тому, что им не удовлетворяют точки плоскости, у которых $y < x^2$, $y \leq 2-x$, $x > 0$. Исправлением этой ошибки может быть разбиение области на две части и использование дизъюнкции либо отбрасывание от большей области ее части. В сложных случаях это действие считается выполненным, 	

если верно определена закрашенная область, то есть программа выводит сообщение «принадлежит» для всех точек закрашенной области и только для них, для точек вне закрашенной области программа выводит «не принадлежит» или не выводит ничего.	
В работе (во фрагментах программ) допускается наличие отдельных синтаксических ошибок.	
Выполнены все три действия	3
Правильно выполнены два действия из трех (исправлены обе ошибки, но в первом пункте задания не приведена таблица (либо таблица содержит ошибки в двух и более строках), либо приведена таблица (которая содержит ошибки не более чем в одной строке), но исправлена только одна ошибка программы). При написании операций сравнения допускается одно неправильное использование строгих/нестрогих неравенств (считается несущественной ошибкой, погрешностью записи). Например, вместо «y>=0» используется «y>0».	2
Правильно выполнено только одно действие из трех, то есть, либо только приведена таблица, которая содержит ошибки в не более чем одной строке, либо таблица не приведена (или приведена и содержит ошибки более чем в одной строке), но исправлена одна ошибка программы. При оценивании этого задания на 1 балл допускается не учитывать корректность работы программ на точках границ областей (вместо нестрогих неравенств в решении были использованы строгие неравенства или наоборот).	1
Все пункты задания выполнены неверно (таблица анализа правильности алгоритма не приведена, либо содержит ошибки в двух и более строках, программа не приведена, либо ни одна из двух ошибок не исправлена).	0
<i>Максимальный балл</i>	3

C2 Дан целочисленный массив из 30 элементов. Элементы массива могут принимать целые значения от 0 до 1000. Опишите на русском языке или на одном из языков программирования алгоритм, позволяющий найти и вывести сумму элементов массива, не кратных пятнадцати. Гарантируется, что в исходном массиве есть хотя бы один элемент, значение которого не делится на пятнадцать. Исходные данные объявлены так, как показано ниже. Запрещается использовать переменные, не описанные ниже, но разрешается не использовать часть из них.

Паскаль	<pre>const N=30; var a: array [1..N] of integer; i, j, s: integer; begin for i:=1 to N do readln(a[i]); ... end.</pre>
Бейсик	<pre>N=30 DIM A(N) AS INTEGER DIM I, J, S AS INTEGER FOR I = 1 TO N INPUT A(I) NEXT I ... END</pre>
СИ	<pre>#include <stdio.h> #define N 30 void main(void){ int a[N]; int i, j, s; for (i=0; i<N; i++) scanf("%d", &a[i]); ... }</pre>

Естественный язык	Объявляем массив А из 30 элементов. Объявляем целочисленные переменные I, J, S. В цикле от 1 до 30 вводим элементы массива А с 1-го по 30-й. ...
--------------------------	---

В качестве ответа Вам необходимо привести фрагмент программы (или описание алгоритма на естественном языке), который должен находиться на месте многоточия. Вы можете записать решение также на другом языке программирования (укажите название и используемую версию языка программирования, например, Free Pascal 2.4) или в виде блок-схемы. В этом случае вы должны использовать те же самые исходные данные и переменные, какие были предложены в условии (например, в образце, записанном на естественном языке).

Содержание верного ответа и указания по оцениванию
(допускаются иные формулировки ответа, не искажающие его смысла)

На языке Паскаль
<pre>s:=0; for i:=1 to N do if (a[i] mod 15<>0) then s:=s+a[i]; writeln(s);</pre>

На языке Бейсик
<pre>S = 0 FOR I = 1 TO N IF A(I) MOD 15<>0 THEN S = S + A(I) ENDIF NEXT I PRINT S</pre>

На языке СИ
<pre>s=0; for (i=0; i<N; i++) if (a[i]%15!=0) s+=a[i]; printf("% d", s);</pre>

На естественном языке
<p>Записываем в переменную S начальное значение, равное 0. В цикле от первого элемента до тридцатого находим остаток от деления элемента исходного массива на пятнадцать. Если остаток от деления на пятнадцать не равен нулю, то считаем сумму текущего элемента массива и значения переменной S. Результат сложения сохраняем в переменную S. Переходим к следующему элементу массива. После завершения цикла выводим значение переменной S.</p>

Указания по оцениванию	Баллы
<p>Предложен правильный алгоритм, выдающий верное значение.</p> <p>Допускается запись алгоритма на другом языке, использующая аналогичные переменные. В случае если язык программирования использует типизированные переменные, описания переменных должны быть аналогичны описаниям переменных на естественном языке. Использование нетипизированных или необъявленных переменных возможно только в случае, если это допускается языком программирования, при этом количество переменных и их идентификаторы должны соответствовать условию задачи. В алгоритме, записанном на языке программирования, допускается наличие отдельных синтаксических ошибок, не искажающих замысла автора программы</p>	2
<p>В любом варианте решения может присутствовать не более одной ошибки из числа следующих:</p> <ol style="list-style-type: none"> 1) Не инициализируется или неверно инициализируется переменная S. 2) Неверно осуществляется проверка на делимость на пятнадцать. 3) На делимость на пятнадцать проверяется не значение элемента, а его индекс. 4) Неверно осуществляется накопление суммы в цикле (например, <code>s:=a[i];</code>) 5) Отсутствует вывод ответа. 6) Используется переменная, не объявленная в разделе описания переменных. 7) Не указано или неверно указано условие завершения 	1

цикла. 8) Индексная переменная в цикле не меняется (например, в цикле while) или меняется неверно. 9) Неверно расставлены операторные скобки	
Ошибок, перечисленных в п. 1-9, две или больше, или алгоритм сформулирован неверно.	0
Максимальный балл	2

С3 У исполнителя Увеличитель две команды, которым присвоены номера:

1. прибавь 2,
2. умножь на 3.

Первая из них увеличивает число на экране на 2, вторая – умножает его на 3.

Программа для Увеличителя – это последовательность команд.

Сколько есть программ, которые число 1 преобразуют в число 31?

Ответ обоснуйте.

Содержание верного ответа и указания по оцениванию
(допускаются иные формулировки ответа, не искажающие его смысла)

Очевидно, из числа 1 нельзя получить четное число. Пусть n – нечетное число, не превосходящее 33. Обозначим $R(n)$ – количество программ, которые преобразуют число 1 в число n . Обозначим $t(n)$ наибольшее нечетное число, кратное трем, не превосходящее n .

Обе команды исполнителя увеличивают исходное число, поэтому общее количество команд в программе не может превосходить 33.

Верны следующие соотношения:

- 1) Если n не делится на 3, то тогда $R(n) = R(t(n))$, так как существует единственный способ получения n из $t(n)$ – прибавлением двоек.
- 2) Пусть n делится на 3.

Тогда $R(n) = R(n/3) + R(n-2)$ (если $n > 3$).

При $n=3$ $R(n) = 2$ (два способа: прибавлением двойки или однократным умножением на 3).

Поэтому достаточно по индукции вычислить значения $R(n)$ для всех нечетных чисел, кратных трем и не превосходящих 31.

Имеем:

$$R(1) = 1$$

$$R(3) = 2 = R(5) = R(7)$$

$$R(9) = R(3) + R(7) = 2 + 2 = 4 = R(11) = R(13)$$

$$R(15) = R(5) + R(13) = 2 + 4 = 6 = R(17) = R(19)$$

$$R(21) = R(7) + R(19) = 2 + 6 = 8 = R(23) = R(25)$$

$$R(27) = R(9) + R(25) = 4 + 8 = 12 = R(29) = R(31)$$

Ответ: 12

Другой способ решения

Очевидно, из числа 1 нельзя получить четное число. Будем решать поставленную задачу последовательно для чисел 1, 3, 5, ..., 31 (то есть для каждого из таких чисел определим, сколько программ исполнителя существует для его получения). Количество программ, которые преобразуют число 1 в число n , будем обозначать через $R(n)$. Число 1 у нас уже есть, значит, его можно получить с помощью “пустой” программы. Любая непустая программа увеличит исходное число, т.е. даст число, больше 1. Значит, $R(1) = 1$. Для каждого следующего числа рассмотрим, из какого числа оно может быть получено за одну команду исполнителя. Если число не делится на три, то оно может быть получено только из предыдущего нечетного числа с помощью команды **прибавь 2**. Значит, количество искомых программ для такого числа равно количеству программ для предыдущего нечетного числа: $R(i) = R(i-2)$. Если число на 3 делится, то вариантов последней команды два: **прибавь 2** и **умножь на 3**, тогда $R(i) = R(i-2) + R(i/3)$. Заполним соответствующую таблицу по приведенным формулам слева направо:

1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31
1	2	2	2	4	4	4	6	6	6	8	8	8	12	12	12

Указания по оцениванию	Баллы
Правильное указание количества возможных программ со строгим доказательством правильности (одним из приведенных выше способов или любым другим).	3
<p>Два балла ставятся в одном из двух случаев:</p> <p>1. Правильное указание количества возможных программ, основанное на верных рассуждениях, но доказательство правильности неполно. В частности, оценка в 2 балла выставляется в случае, если просто перечислены все правильные программы и не доказано отсутствие других программ, кроме приведенных.</p> <p>Приведены правильные и строгие рассуждения, доведенные до конца, но в вычислениях допущена арифметическая ошибка, в результате чего получен неверный ответ</p>	2
<p>Представленное решение обладает одним из свойств</p> <p>1. Указано, что нужно рассматривать значения n, меньшие, чем 31, и приведены правильные рекуррентные соотношения (см. выше), возможно, неполные.</p> <p>2. Правильно выписаны и обоснованы значения $R(n)$ для небольших n.</p> <p>Правильно написан ответ, но нет его обоснования.</p>	1
Не выполнено ни одно из перечисленных выше условий	0
Максимальный балл	3

C4 По каналу связи передается последовательность положительных целых чисел X_1, X_2, \dots все числа не превышают 1000, их количество заранее неизвестно. Каждое число передается в виде отдельной текстовой строки, содержащей десятичную запись числа. Признаком конца передаваемой последовательности является число 0. Участок последовательности от элемента X_T , до элемента X_{T+N} называется подъемом, если на этом участке каждое следующее число больше предыдущего. Высотой подъема называется разность $X_{T+N} - X_T$. Напишите эффективную программу, которая вычисляет наибольшую высоту среди всех подъемов последовательности. Если в последовательности нет ни одного подъема, программа выдает 0. Программа должна напечатать отчет по следующей форме:

Получено ... чисел

Наибольшая высота подъема: ...

Размер памяти, которую использует Ваша программа, не должен зависеть от длины переданной последовательности чисел.

Перед текстом программы кратко опишите используемый вами алгоритм решения задачи.

Пример входных данных:

144

17

27

3

7

9

11

10

0

Пример выходных данных для приведенного выше примера входных данных:

Получено 8 чисел

Наибольшая высота подъема: 10

**Содержание верного ответа и указания по оцениванию
(допускаются иные формулировки ответа, не искажающие его смысла)**

Программа читает все входные данные один раз, не запоминая все входные данные в массиве. Во время чтения программа помнит число L_{Max} – высоту самого высокого из уже закончившихся подъемов, а также необходимые сведения о текущем подъеме, например, число L – высоту текущего подъема (то есть разность между последним и первым числом участка) и последнее прочитанное число T (это число – наибольшее из чисел текущего подъема). Прочитав очередное число R , программа сравнивает его с числом T . Если $R > T$, то значение L увеличивается на $R - T$. В противном случае фиксируется конец подъема и начало нового участка. То есть, во-первых, значение L сравнивается с L_{Max} и, при необходимости, L_{Max} полагается равным L . Во-вторых, полагаем $L = 0$

Ниже приведены примеры решения задания на языках Паскаль и Бейсик, а также на Алгоритмическом языке. Допускаются решения, записанные на других языках программирования.

Пример правильной и эффективной программы на языке Паскаль:

```
program c4_1;

var
  R, T, N, L, LMax : Integer;

begin
  N:=0;
  L:=0;
  LMax:=0;
  T:=1001;
  repeat
    ReadLn(R);
    if R<>0 then N:=N+1;
    if R>T then L:=L+R-T
    else begin
      if L>LMax then LMax:=L;
      L:=0;
    end;
    T:=R;
  until R=0;
  WriteLn('Получено ', N, ' чисел');
  WriteLn('Наибольшая высота подъема ', LMax);
end.
```

Пример правильной и эффективной программы на Алгоритмическом языке:

```
алг C4_1
нач
  цел R, T, N, L, LMax
  N:=0
  L:=0
  LMax:=0
  T:=1001;
```

```
нц
  ввод R
  если R<>0 то N:=N+1 все
  если R>T то
    L:=L+R-T
  иначе
    если L>LMax то LMax:=L все
    L:=0
  все
  T:=R
кц при R=0
вывод "Получено ", N, " чисел", нс
вывод "Наибольшая высота подъема ", LMax, нс
кон
```

Пример правильной и эффективной программы на языке Бейсик:

```
DIM R, T, N, L, LMax AS INTEGER
N = 0
L = 0
LMax = 0
T = 1001;
DO
  INPUT R
  IF R <> 0 THEN N = N + 1
  IF R > T THEN
    L = L + R-T
  ELSE
    IF L > LMax THEN LMax = L
    L = 0
  END IF
  T = R
LOOP UNTIL R = 0
PRINT "Получено "; N; " чисел"
PRINT "Наибольшая высота подъема "; Lmax
```

Указания по оцениванию	Баллы
Программа правильно работает для любых входных данных произвольного размера, и находит ответ, не сохраняя входные данные в массиве. Допускается наличие в тексте программы одной синтаксической ошибки: пропущен или неверно указан знак пунктуации, неверно написано или пропущено зарезервированное слово языка программирования, не описана или неверно описана переменная, применяется операция, недопустимая для соответствующего типа данных (если одна и та же ошибка встречается несколько раз, то это считается за одну ошибку).	4
Программа работает верно, но размер используемой памяти зависит от длины используемой последовательности. Например, входные данные запоминаются в массиве или другой структуре данных (например, контейнер <code>priority_queue</code> , <code>set</code> или <code>map</code> в C++), размер которого соответствует числу N. Допускается наличие от одной до трех синтаксических ошибок. Возможно, в принципиально верно организованном вводе данных есть ошибка.	3
Программа работает в целом верно, эффективно или нет, но, в реализации алгоритма содержатся ошибки при инициализации цикла анализа массива данных или обработке конца массива. Допускается наличие от одной до пяти синтаксических ошибок, описанных выше.	2
В программе есть блок выделения очередного подъема, однако этот блок написан с ошибками. Допускается наличие от одной до пяти синтаксических ошибок, описанных выше	1
Прочее	0
<i>Максимальный балл</i>	4

Ответы к заданиям с выбором ответа

№ задания	Ответ
A1	4
A2.	2
A3.	1
A4	2
A5	2
A6.	4
A7	2

№ задания	Ответ
A8.	2
A9.	4
A10.	1
A11.	4
A12.	3
A13.	2

Ответы к заданиям с кратким ответом

№ задания	Ответ
B1	10
B2.	12121
B3.	150
B4.	ТТРВ
B5	3
B6	46
B7.	19
B8.	7

№ задания	Ответ
B9	19
B10	2312
B11.	FBGA
B12	4600
B13.	9
B14	8
B15.	2

Ответы к заданиям с выбором ответа

№ задания	Ответ
A1.	1
A2.	1
A3.	1
A4.	1
A5.	2
A6.	1
A7.	4

№ задания	Ответ
A8.	4
A9.	3
A10.	4
A11.	3
A12.	3
A13.	3

Ответы к заданиям с кратким ответом

№ задания	Ответ
B1.	400
B2.	11221
B3.	660
B4.	ТТРМ
B5.	1
B6.	48
B7.	91
B8.	5

№ задания	Ответ
B9.	23
B10.	5128
B11.	FBHA
B12.	5000
B13.	11
B14.	6
B15.	2