|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | А1 | А2 | А3 | А4 | А5 | А6 | А7 | А8 | А9 | А10 | А11 | А12 | А13 |
| 701 | 2 | 2 | 3 | 2 | 1 | 3 | 1 | 3 | 2 | 4 | 4 | 1 | 2 |
| 702 | 3 | 3 | 4 | 3 | 1 | 4 | 3 | 4 | 3 | 2 | 2 | 3 | 3 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | B1 | B2 | B3 | B4 | B5 | B6 | B7 | B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
| 701 | 120 | 12121 | 410 | ККБФБФ | 3 | 324 | 27 | 4 | 8 | 256000 | 1488 | 2341 | 15 | -64 | 7 |
| 702 | 32 | 11221 | 163 | БЦФЦФ | 1 | 9,17 | 72 | 7 | 23 | 2560 | 126 | 4123 | 8 | -40 | 10 |

***Часть С.***

***---------------------701----------------------***

***С1***



2. Возможная доработка (Паскаль):

if (y<=x+1) and (y>=x\*x-1) and (y>=0) then

write('принадлежит')

else

write('не принадлежит')

Возможны и другие способы доработки.

Пример:

if (y<=x+1) then

if (y>=x\*x-1) then

if (y>=0) then

write('принадлежит')

else

write('не принадлежит')

else

write('не принадлежит')

else

write('не принадлежит')

**С2**

Пояснение.

Решение на естественном языке:

Объявим константу N, равную числу элементов массива (30), целочисленные переменные р для хранения текущего произведения отрицательных элементов и i для хранения индекса просматриваемого элемента. Ввод значений в массив описывать не требуется, так как он задан. Присвоим р значение единица. В цикле от первого (нулевого) до последнего (N или N - 1) значения индекса совершим следующие действия.

Сравним с нулем значение текущего элемента массива. В случае, если значение текущего элемента меньше нуля, умножим текущее значение переменной р на значение рассматриваемого элемента массива.

По завершении цикла выводим значение переменной р.

|  |  |  |
| --- | --- | --- |
| ПРОГРАММА НА ПАСКАЛЕ | ПРОГРАММАНА БЕЙСИКЕ | ПРОГРАММА НА СИ |
| const N=30;  var a: array [1..N] of integer;  p, i: integer;  begin  p := 1;  for i:= 1 to N do  if a [i] < 0  then  p : = p \* a [ i ] ;  writeln (p);  end. | N=30  DIM A(N), Р, I AS INTEGER  P = i  FOR 1=1 TO N  IF A(I) < 0 THEN  P = P \* A (I)  ENDIF  NEXT I  PRINT P  END | void main(void)  { const N=30;  int array [N];  int p;  p = 1;  for (int i=0; i { if (array [i]<0)  { p= p \* array[i];  }  }  printf ("%d", p);  } |

**С3**

**Пояснение.**

Обозначим R(n) — количество программ, которые преобразуют число 1 в число n. Обозначим t(n) наибольшее нечетное число, кратное четырем, не превосходящее n.   
Обе команды исполнителя увеличивают исходное число, поэтому общее количество команд в программе не может превосходить 32.   
Верны следующие соотношения:   
1. Если n не делится на 4, то тогда R (n) = R(T(n)), так как существует   
единственный способ получения n из T(n) - прибавлением единиц.   
2. Пусть n делится на 4.   
Тогда R (n) = R(n/4)+R(n-1)= R(n/4)+R(n-4) (если n>4).   
При n = 4 выполнено: R(n) = 2 (два способа: прибавлением трех единиц или   
однократным умножением на 4).   
Поэтому достаточно по индукции вычислить значения R(n) для всех нечётных   
чисел, кратных четырем и не превосходящих 32.   
Имеем:   
R(1)= R(2) = R(3) = 1   
R(4) = 2 = R(5) = R(6) = R(7)   
R(8) = R(2) + R(7) = 1 + 2 = 3 = R(9) = R(10) = R(11)   
R(12) = R(3)+R(11) = 1 + 3 = 4 = R(13)= R(14) = R(15)   
R(16) = R(4) + R(15) = 2 + 4 = 6 = R(17) = R(18) = R(19)   
R(20) = R(5) + R(19) = 2 + 6 = 8 = R(21) = R(22) = R(23)   
R(24) = R(6) + R(23) = 2 + 8 = 10 = R(25) = R(26) = R(27)   
R(28) = R(7) + R(27) = 2 + 10 = 12 = R(29) = R(30) = R(31)   
R(32) = R(8) + R(31) = 3 + 12 = 15

Ответ: 15.  
Другой способ решения   
Будем решать поставленную задачу последовательно для чисел 1, 2, 3,..., 32 (то есть для каждого из 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 4, то оно может быть получено только из предыдущего числа с помощью команды **прибавь 1**. Значит, количество искомых программ для такого числа равно количеству программ для предыдущего числа: http://reshuege.ru:89/formula/63/63dc70cfaab066cbfb2fdce67777b95d.png.   
Если число на 4 делится, то вариантов последней команды два: **прибавь 1 и умножь на 4**, тогда http://reshuege.ru:89/formula/09/092f38bf6347381406717c45ec477fbd.png. Заполним соответствующую таблицу по приведёным формулам слева направо:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| 1 | 1 | 1 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 4 | 4 | 4 | 4 | 6 |
| 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 |
| 6 | 6 | 6 | 8 | 8 | 8 | 8 | 10 | 10 | 10 | 10 | 12 | 12 | 12 | 12 | 15 |

При этом ячейки, относящиеся к числам, которые не делятся на 4, можно в решении и опустить (за исключением первого и последнего чисел):

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 4 | 8 | 12 | 16 | 20 | 24 | 28 | 32 |
| 1 | 2 | 3 | 4 | 6 | 8 | 10 | 12 | 15 |

Ответ: 15.

**С4**

**Пояснение.**

program C4;   
uses crt;   
var a:array['a'..'z'] of integer;   
i:integer;   
ch:char;   
Begin   
repeat   
      read(ch);   
      if ch in ['a'..'z'] then a[ch]:=a[ch]+1;   
Until ch='.';   
for i:=1 to 255 do   
begin   
     if (chr(i) in ['a'..'z']) and (a[chr(i)]>0) then   
     writeln(chr(i),a[chr(i)]);   
end;   
End.

**--------------702--------------**

**С1.**



2. Возможная доработка (Паскаль):

if (y<=x\*x\*x) and (y\*y+x\*x<=4) and (x>=0) then

write('принадлежит')

else

write('не принадлежит')

Возможны и другие способы доработки.

Пример:

if (y<=x\*x\*x) then

if (y\*y+x\*x<=4) then

if (x>=0) then

write('принадлежит')

else

write('не принадлежит')

else

write('не принадлежит')

else

write('не принадлежит')

**С2**

uses crt;

const n=30;

type massiv=array[1..n] of integer;

var a:massiv;

min,max,i,sum:integer;

sred1,sred2:real;

Begin

for i:=1 to n do

begin

write('a[',i,']= ');readln(a[i]);

end;

sum:=0;

min:=a[1];

max:=a[1];

for i:=1 to n do

begin

if a[i]<=min then min:=a[i];

if a[i]>=max then max:=a[i];

sum:=sum+a[i];

end;

sred1:=(min+max)/2;

sred2:=sum/n;

writeln('Разность: ',sred1─sred2);

End.

**С3**

Обозначим R(n) — количество программ, которые преобразуют число 4 в число n. Обозначим t(n) наибольшее кратное 3, не превосходящее n.   
Обе команды исполнителя увеличивают исходное число, поэтому общее количество команд в программе не может превосходить 30.   
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 единиц.   
2. Пусть n делится на 3.   
Тогда R(n) = R(n / 3) + R(n - 1)= R(n / 3) + R(n - 3) (если n > 3).   
При n = 9 R(n)) = 1 (один способ: прибавлением тройки).   
Поэтому достаточно постепенно вычислить значения R(n) для всех чисел, кратных 3 и не превосходящих 34: сначала вычисляем R(4), затем R(6), R(9) и т. д.   
Имеем:   
R(4)=1   
R(6) = R(9)=1 = R(5) = R(10)= R(11),   
R(12) = R(4)+R(9)=1+1=2= R(13)=R(14),   
R(15) = R(5) + R(12) =1 + 2 = 3= R(16) = R(17),   
R(18) = R(6) + R(15) =1 + 3 = 4= R(19) = R(20),   
R(21) = R(7) + R(18) =1 + 4 = 5= R(22) = R(23),   
R(24) = R(8) + R(21) =1 + 5 =6= R(25) = R(26),   
R(27) = R(9) + R(24) =1 + 6 =7= R(28) = R(29),   
R(30) = R(9) + R(27) =1 + 7 =8= R(31) = R(32),   
R(33) = R(10) + R(30) =1 + 8 =9= R(34).

Ответ: 9.   
Другая форма решения.   
Будем решать поставленную задачу последовательно для чисел 4, 5, 6,..., 34 (то есть для каждого из 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 пять, то оно может быть получено только из предыдущего с помощью команды **прибавь 1**. Значит, количество искомых программ для такого числа равно количеству программ для предыдущего возможного числа: http://reshuege.ru:89/formula/50/50157e00b89afe00920b192df0158f58.png.   
Если число на 3 делится, то вариантов последней команды два: **прибавь 1 и умножь на 3**, тогда http://reshuege.ru:89/formula/28/2838ee947f49935d395b495234ed9d1a.png. Заполним соответствующую таблицу по приведёным формулам слева направо:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 3 | 3 | 3 | 4 |
| 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 |
| 4 | 4 | 5 | 5 | 5 | 6 | 6 | 6 | 7 | 7 | 7 | 8 | 8 | 8 | 9 |
| 34 |  | | | | | | | | | | | | | |
| 9 |

При этом ячейки, относящиеся к числам, которые не делятся на 3, можно в решении и опустить (за исключением первого и последнего чисел):

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 4 | 6 | 9 | 12 | 15 | 18 | 21 | 24 | 27 | 30 | 33 | 34 |
| 1 | 1 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 9 |

Ответ: 9.

**С4**

Программа читает входные данные, сразу подсчитывая минимальную длину встречающихся слов. За второй проход исходных данных производится замена букв латинского алфавита и печать расшифрованного сообщения. Баллы начисляются только за программу, которая решает задачу хотя бы для частного случая (например, для строчных английских букв и без циклического сдвига).   
  
Пример правильной и эффективной программы на языке Паскаль:   
var f:boolean;   
i, k, min: integer;   
c,cnew:char;   
s:string;   
begin   
s : = " ;   
min:=250;   
k:=0;   
f:=false;   
repeat read(c);   
s:=s+c;   
if f then {слово началось}   
if с in ['a'..'z','A'..'Z']   
then k:=k+l   
else begin   
if kcmin then   
min:=k;   
f:=false   
end   
else {f=false}   
if с in [ 'a' ..'z', 'A'..'Z']   
then begin f:=true; k:=l end   
until c='. ' ;   
for i:=l to length(s) do   
begin   
cnew: = chr (ord (s [i] ) -min) ;   
case s[i] of   
' a'..'z1:i f cnew<1 a 1 then   
write(chr(ord(cnew)+2 6))   
else write(cnew);   
1A'..' Z 1:if cnewc'A' then   
write(chr(ord(cnew)+26))   
else write(cnew);   
else write(s[i])   
end;   
end;   
readln end.   
  
Пример правильной программы на языке Бейсик:   
DIM i, j, min, к, f, a (26) AS. INTEGER   
DIM s AS STRING   
INPUT s   
i = 1   
k = 0   
min = 250   
f = 0   
WHILE NOT (MID$(s, i, 1) = ".")   
c$ = MID$ (s, i, 1)   
IF f = 1 THEN   
IF (c$ >= "A") AND (с $ <= "Z") OR   
(C$ >= "a") AND (c$ <= "z") THEN   
k = k + 1   
ELSE IF k < min THEN min = k   
f = 0   
ENDIF   
ELSE   
IF (с $ >= "A") AND (c$ <= "Z") OR   
(c$ >= "a") AND (c$ <= "z") THEN   
f = 1: k = 1   
ENDIF   
ENDIF   
i = i + I   
WEND   
IF к < min THEN min = k   
FOR j = 1 TO i   
cnew$ = CHR$(ASC(MID$(s, j, 1)) - min)   
IF (MID$(s, j, 1) >= "a") AND (MID$(s, j, 1)   
<= "z") THEN   
IF cnew$ < "a" THEN   
PRINT (CHR$(ASC(cnew$) + 26));   
ELSE PRINT cnew$;   
ENDIF   
ELSE   
IF (MIDS (s, j, 1) >= "A") AND (MID$ (s, j, 1)   
<= "Z") THEN   
IF cnew$ < "A" THEN   
PRINT (CHR$(ASC(cnew$) + 26));   
ELSE PRINT cnew$;   
ENDIF   
ELSE PRINT MID$(s, j, 1);   
ENDIF   
ENDIF   
NEXT j   
END