Инженерное дело - программирование, заключительный этап, 10 класс

Задача 1 (5 б.)

Химики смешивают несколько добавок к топливу и проверяют, при какой температуре смесь превысит заранее заданное давление. Для этого смесь нагревают в химическом реакторе. Лаборант, которого оставляют следить за реактором, пишет в текстовый файл температуру смеси, которую измеряет раз в минуту. Когда давление превышает заданное значение, процесс прекращается, реактор охлаждают и загружают новую смесь. Определите, сколько длился самый долгий нагрев смеси. При нагреве, что очевидно, температура смеси не уменьшается.

Формат ввода

На вход программе в первой строке подается натуральное число ***N***, не превышающее ***10000*** – количество замеров температуры.

Во второй строке подается натуральное число X, не превышающее ***1000*** – пороговое значение температуры.

Далее в ***N*** строках подается по одному натуральному числу t_i, не превышающему 1000 – температура смеси при измерении номер ***i***.

Формат вывода

Вывести одно целое число – сколько минут длился самый длительный нагрев смеси.

Пример

Входные данные	Выходные данные
7 800 540 690 801 560 805 540 900	3

Решение

```
program pzv1;

var
 s,i,n:integer;
 x,px:real;
 flag:boolean;
begin
 readln(n);
 s:=0;
```

```
 px:=-1;
 flag:=false;
 for i:=1 to n do
 begin
   readln(x);
   if x=100 then
   begin
    flag:=true;
   end
   else
   begin
     if flag and (x<>100) then
      begin
        s:=s+1;
        flag:=false;
      end;
   end;
   px:=x;
 end;
 if x = 100 then
   s:=s+1;
 writeln(s);
end.
```

Задача 2 (8 б.)

Прибор передает результаты измерений блоками информации. Каждый блок представляет собой набор цифр в шестнадцатеричной системе счисления (0123456789ABCDEF). Последняя цифра пятеричной записи суммы цифр блока показывает код ошибки, где 0 означает корректную передачу данных. Предпоследняя цифра пятеричной записи суммы цифр блока показывает тип результата:

0: «Излучение»

1: «Вспышка»

2: «Нагрев»

3: «Охлаждение»

4: «Перегрузка»

Определите количество результатов «Перегрузка», которые были переданы без ошибок после приема n блоков данных.

Формат ввода

В первой строке программе подается на вход число натуральное число ***n***, не превышающее ***1000.***

Далее в каждой из ***n*** строк идет блок данных – набор цифр в шестнадцатеричной системе счисления (***0123456789ABCDEF***), длина блока не превышает ***100*** знаков.

Формат вывода

Вывести одно число – количество результатов «Перегрузка», которые были переданы без ошибок после приема n блоков данных.

Примеры

Входные данные	Выходные данные
4 FFF F3CA5 100 32	2

Решение

```
program z9pzv;

const
  digits = '123456789ABCDEF';

function count(s:string):integer;
var
  i,k:integer;
begin
  k:=0;
  for i:=1 to length(s) do
  begin
   k:=k+pos(copy(s,i,1),digits);
  end;
  count:=k;
end;

var
 n,i,k,m:integer;
 s:string;
begin
 readln(n);
 k:=0;
 for i:=1 to n do
   begin
     readln(s);
     m:=count(s);
     if (m mod 5 = 0) and ((m div 5) mod 5 = 4) then
       begin
         k:=k+1;
       end
   end;
 writeln(k);
end.
```

Задача 3 (10 б.)

В ходе игры «Зарница» Саша и Женя пересылают друг другу важные сообщения. Но для того, чтобы противник не смог их понять, сообщения кодируются. Для кодирования информации ребята используют латинский алфавит из 26 букв, все буквы заглавные. Слова кодируются следующим образом. Каждая буква в слове заменяется ее порядковым номером в алфавите, записанном в системе счисления с основанием ***Sys*** (***2 <= Sys <= 36***). Все полученные числа записываются подряд без пробелов. Если числа (порядковые номера букв) в заданной системе счисления могут иметь разную длину, то более короткие числа дополняются слева нулями до требуемой длины. Например, в десятичной системе счисления порядковый номер буквы ***A*** будет равен ***1***, а буквы ***Z*** – ***26***. Соответственно, при шифровании, к единице слева будет дописан ноль. То есть код буквы ***A*** будет ***01***, а код буквы ***Z*** – ***26***. Для усложнения возможной расшифровки сообщения противником, для кодирования букв, стоящих на разных местах в слове, используются различные системы счисления. Основание использованной системы счисления выбирается исходя из порядкового номера буквы в кодируемом сообщении. Для кодирования первой буквы сообщения используется двоичная система счисления, для второй – троичная, для третьей – четверичная и т.д., до системы счисления с основанием 36 включительно. Далее основания систем счисления повторяются циклически – 2, 3, 4, …36, 2, 3, … Например, слово ***AZ***, будет закодировано как ***00001222***.

Напишите программу, которая будет расшифровывать закодированные сообщения.

На вход программе подается одно закодированное сообщение. Длина сообщения не более 200 символов. Программа должна вывести исходное слово.

Примеры

Входные данные	**Выходные данные**
011010011200142130 15	MAXIMUM
00101112010	END

Латинский алфавит
ABCDEFGHIJKLMNOPQRSTUVWXYZ

Решение

```
alf = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
cifr = '0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ'
```

```
def syslen(sys):
    if sys > 26:
        lenb = 1
    elif sys >= 6:
        lenb = 2
    elif sys >= 3:
        lenb = 3
    else:
        lenb = 5
    return lenb

s = input()
sys = 2
res = ''
while s != '':
    lenb = syslen(sys)
    num = s[:lenb]
    s = s[lenb:]
    p = int(num, sys)
    res += alf[p - 1]
    sys += 1
    if sys == 37:
        sys = 2
print(res)
```

Задача 4 (12 б.)

На кафедре работает N (0<N <= 100) человек. Необходимо составить расписание дежурств по кафедре. Для этого необходимо определить в первую очередь проблемные рабочие дни (рабочими днями считаются все дни недели, кроме воскресенья), когда запланированных занятий ни у кого из сотрудников нет. Кроме того, важно знать дни, когда сотрудников на кафедре очень мало, поэтому помещение может быть закрыто (например, преподаватель ушел в аудиторию читать лекцию).

Для каждого сотрудника есть рабочее расписание. Расписание двух типов. Каждое кодируется особым образом.

1) Преподаватели работают по дням недели. Обозначение: строка начиняется с буквы “P”; далее идет цепочка цифр без пробелов: 1 – понедельник, 2 – вторник, 3 - среда, 4 – четверг, 5 – пятница, 6 – суббота. Например, в строке “P134” – закодировано, что преподаватель ведет занятие по понедельникам (1), средам (3) и четвергам (4).
2) Учебный персонал работает в зависимости от чётности/нечётности даты. Обозначение: строка начиняется с буквы “U”; далее идет одна цифра: 1 – нечётные даты, 0 – чётные даты. Например, в строке “U1” – закодировано, что лаборант работает по нечетным датам (1, 3, 5, 7, …, 29, 31).

Необходимо определить, сколько проблемных дней в период, на который составляется расписание.

Гарантируется, что дата начала меньше даты окончания периода, что дежурства начинаются не ранее 1 января 2023 года. Период не превышает двух календарных лет. 01.01.2023 – воскресенье (остальные дни недели должна определять программа). **Воскресенье – выходной для всех!**

Входные данные:

Построчно.

Дата начала периода в формате dd.mm.yyyy

Дата окончания периода (включая этот день) в формате dd.mm.yyyy

N – количество сотрудников.

В каждой последующей строчке закодированы персональные расписания преподавателей и учебного персонала (формат смотри выше).

Выходные данные:

Число - количество дней, когда на кафедре никого нет.

На следующей строке – два числа через пробел: сколько дней на кафедре присутствует минимальное количество сотрудников и само минимальное количество сотрудников, когда кто-то есть на кафедре.

Входные данные	Вывод	Примечание
09.01.2023 23.01.2023 5 P135 U0 P123 P15 U0	2 1 1	На кафедре никого не планируется 19 и 21 января 2023 года (два дня). Обратите внимание, что воскресенья не считаются! А 17 января будет только один человек (один день).
09.01.2023 23.01.2023 5 P135 U0 P123 P15 U1	0 4 1	Каждый рабочий день на кафедре кто-то есть. Минимальное количество людей на кафедре 1 человек. Таких присутственных дня четыре: 12, 14, 18 и 21 января.
10.01.2023 13.01.2023 5 P135 U1	2 2 4	На кафедре никого не планируется 10 и 12 января 2023 года (два дня). А 11 и 13 января будет минимальное количество человек - 4 (два дня).

P13 P15 U1		

Решение

```
from datetime import date, timedelta

d1, m1, y1 = map(int, input().split('.'))
d2, m2, y2 = map(int, input().split('.'))
cnt = [[0] * 7 for _ in range(2)]
a = date(y1, m1, d1)
b = date(y2, m2, d2)
one = timedelta(days=1)

while a<=b:
  tt = a.timetuple()
  y1,m1,d1 = tt[0],tt[1],tt[2]
  w = (a.weekday() + 1) % 7
  cnt[d1 % 2][w] += 1
  a += one

x = [[0] * 7 for _ in range(2)]
n =int(input())
for _ in range(n):
  s = input()
  if s[0] == 'P':
    for c in s[1:]:
      x[0][int(c) % 7] += 1
      x[1][int(c) % 7] += 1
  else:
    for j in range(7):
      x[int(s[1])][j] += 1



lx = [x[0][i] for i in range(1, 7) if x[0][i] != 0 and cnt[0][i] != 0]
rx = [x[1][i] for i in range(1, 7) if x[1][i] != 0 and cnt[1][i] != 0]

mx = min(lx + rx)
ans0 = 0
ans = 0
for i in range(2):
  for j in range(1, 7):
    if x[i][j] == mx:
      ans += cnt[i][j]
    elif x[i][j] == 0:
       ans0 += cnt[i][j]
print(ans0)
print(ans, mx)
```

Задача 5 (15 б.)

Город имеет форму круга радиуса ***R*** с центром в точке (0,0).

Сеть метро состоит из ***N*** *линий метро* (часть линий или все проходят через город).

Линия метро - ломаная из отрезков прямых, вершины которых имеют целочисленные координаты. *Линия метро* не имеет самопересечений и может быть замкнутой. Во всех точках с целочисленными координатами, через которые проходят *линии метро* расположены *станции метро* .

Для каждой точки с целочисленными координатами определим параметр ***вес вершины***. ***Вес вершины*** — это количество *станций метро,* расстояние до которых не более 1 (длины клетки).

Город разбит на *кварталы*. *Квартал* — это единичная клетка с целочисленными координатами вершин, хотя бы одна из которых находящаяся ***строго*** внутри города.

Для каждого *квартала* определим параметр ***доступность***. *Доступность квартала* равна сумме *весов вершин* квартала (вершина *квартала* может быть вне города)

Найдите значение "***доступности***" для каждого квартала. Для каждой полученной "***доступности***" определите *число* кварталов, имеющих эту ***доступность***.

Входные данные

В первой строке заданы значения ***R, N*** ($4<R<201$, $0<N<1001$)

В следующих ***N*** строках заданы описания *линий метро*.

Каждая линия описывается следующим образом:

первое число в строке ***M*** равно количеству вершин ломаной, далее даны координаты вершин (по два числа на вершину).

Замкнутые ломаные определяются тем, что координаты начальной и конечной вершины совпадают.

Выходные данные

В первой строке выведите число ***K*** - количество различных значений "***доступности***" (включая нулевую).

В следующих ***K*** строках выведите по *два числа* - значение "***доступности***" и *число* кварталов, имеющих такое значение "***доступности***"

Примеры:

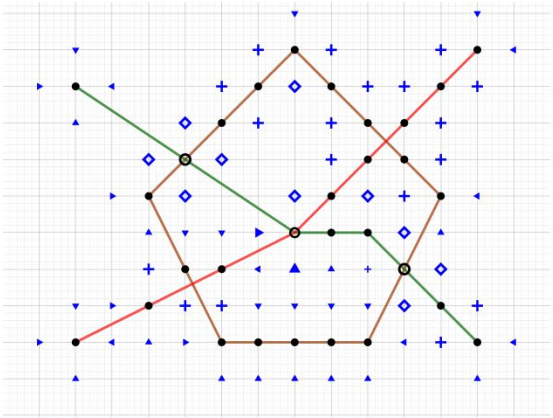

Рис. 1

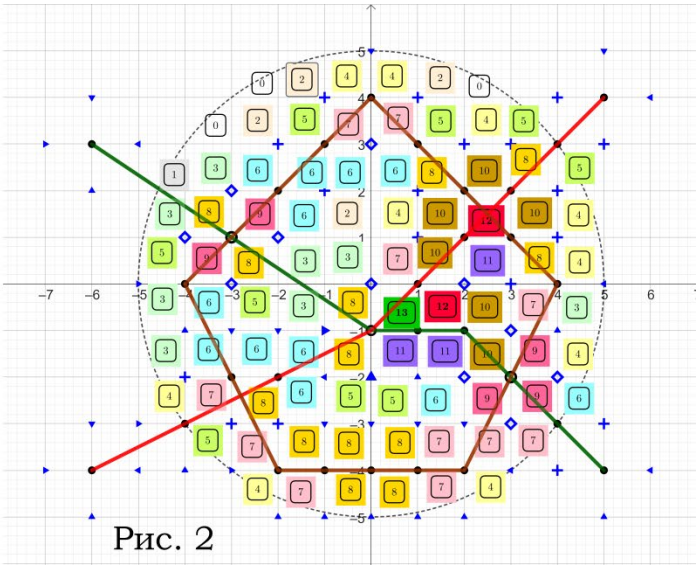

Рис. 2

Входные данные	Выходные данные	Пояснение
5 3 6 2 -4 -2 -4 -4 0 0 4 4 0 2 -4 4 -6 3 0 -1 2 -1 5 -4 3 5 4 0 -1 -6 -4	14 0 3 1 1 2 4 3 8 4 10 5 9 6 12 7 11 8 13 9 5 10 6 11 3 12 2 13 1	Город (рис. 1,2) расположен в круге радиуса 5 с центром в точке (0,0). Сеть метро состоит из 6 *линий метро* (2 радиальных, 1 кольцевая): **6 2 -4 -2 -4 -4 0 0 4 4 0 2 -4** - кольцевая линия из 5 звеньев, 16 станций **4 -6 3 0 -1 2 -1 5 -4** - радиальная линия из 3 звеньев, 8 станций **3 5 4 0 -1 -6 -4** - радиальная линия из 2 звеньев, 9 станций Есть три пересадки (в вершинах (-3,1), (0,-1), (3,-2)) На рис.1 отмечены вершины, которые не являются станциями и имеют не нулевой вес (треугольник - вес 1, крестик - вес 2, ромб - вес 3) На рис.2 для всех городских кварталов указано значение параметра доступности квартала.

Решение

```
def nod (a,b):
  if a*b==0 : return max(a,b)
  while b>0:
    a,b=b,a%b
  return a
def not_in_circle(x,y,r): # проверка принадлежности клетки (x,y) центральному кругу радиуса R
    if x*x+y*y<r*r  :  return True # True если вершина принадлежит городу
    if x*x+(y+1)*(y+1)<r*r  :  return True
    if (x+1)*(x+1)+y*y<r*r  :  return True
```

```
    if (x+1)*(x+1)+(y+1)*(y+1)<r*r  :  return True
    return False # ВСЕ вершины вне города или на его границе
# Ввод данных
R, N = map(int,input().split())
W,H=2*R+2,2*R+2 # длина и ширина покрывающего прямоугольника
AA=[] # список для описания станций
YX=[] # массив для станций
for _ in range (2*R+4) : # заполнение массива YX
  X=[0]*(2*R+4) # список уровня с
  YX.append(X)
for jj in range(N): # ОСНОВНОЙ ЦИКЛ по ЛИНИЯМ МЕТРО
  A=list(map(int,input().split())) # описание линии метро
  k=A[0] #количество станций на линии
  j=1 #  указатель на первую станцию линии
  x0,y0=1+A[j]+W//2,1+A[j+1]+H//2 # начальная точка линии со сдвигом
  x1,y1=1+A[-2]+W//2,1+A[-1]+H//2 # последняя вершина ломаной со сдвигом
  if x0 !=x1 or y0!=y1 :  # линия не закольцована => пробуем занести в таблицу станций
    if  (x0*(W-x0)>=0)  and  (y0*(H-y0)>=0)  :  # вершина принадлежит покрывающему
прямоугольнику
      YX[y0][x0]+=1 # маркировка вершины
  for ii in range(k-1) : # ЦИКЛ по звеньям ломаной ЛИНИИ МЕТРО
    j+=2
    x1,y1=1+A[j]+W//2,1+A[j+1]+H//2 # следующая вершина со сдвигом
    vx,vy=x1-x0,y1-y0 # вектор звена ломаной линии метро
    d=nod(abs(vx),abs(vy)) # кол-во станций, которое надо добавить
    vx,vy=vx//d,vy//d # базовый вектрор
    for i in range(1,d+1) :
         x,y =x0+vx*i,y0+vy*i # очередная станция
         if (x*(W-x)>=0) and (y*(H-y)>=0) : # станция принадлежит квадрату города
           YX[y][x]+=1 # маркировка станции
    x0,y0=x1,y1 # сдвигаем указатель на начало следующего отрезка линии метро
# Обработка результатов маркировки станций, подсчет ДОСТУПНОСТИ кварталов.
rez=[0]*(6*N+1) # максимальная ЗНАЧИМОСТЬ не более 4* число линий метро
for y in range(1,H): # перебор левых нижних углов кварталов по y
  #print(*YX[H-y+1])
  for x in range (1,W) : # перебор левых нижних углов кварталов по x
    if not_in_circle(x-1-W//2,y-1-H//2,R) : # проверка квартала на принадлежность городу
      k=YX[y][x-1]+YX[y+1][x-1] # сумма уровней вершин квартала
      k+=YX[y][x+2]+YX[y+1][x+2] # сумма уровней вершин квартала
      k+=YX[y+2][x]+YX[y+2][x+1] # сумма уровней вершин квартала
      k+=YX[y-1][x]+YX[y-1][x+1] # сумма уровней вершин квартала
      k+=3*(YX[y][x]+YX[y+1][x]+YX[y+1][x+1]+YX[y][x+1]) # сумма уровней вершин квартала
      rez[k]+=1 # маркируем ЗНАЧИМОСТЬ квартала
      #if y-H//2>4 : print('k=',k,(x-W//2-1,y-H//2-1))
rz=0 # количество разных ЗНАЧИМОСТЕЙ
for r in rez :
  if r>0 : rz+=1
print(rz) # вывод результата
for i in range(len(rez)) : # просмотр массива результатов
  if rez[i]>0 : # результат отличен от нуля
    print(i,rez[i]) # вывод результата
```