Отборочный (заочный) онлайн-этап Олимпиады школьников «Шаг в будущее» по профилю «Инженерное дело» специализации «Техника и технологии (программирование)» (общеобразовательный предмет информатика), осень 2020 год

8 класс

Вариант 1

Задача 1

Пётр Васильевич работает учителем информатики и каждый рабочий день ездит в школу. До школы он может добраться на трамвае, который ходит с одинаковым интервалом в 10 минут. Первый трамвай идёт в *5 часов* ***k*** *минут*. Сегодня Пётр Васильевич подошёл к остановке в *8 часов* ***m*** *минут* и хочет уехать как можно скорее. Если бы в ту же минуту подошёл его трамвай, он бы уехал сразу. Напишите программу, которая определит, во сколько сегодня уедет Пётр Васильевич.

На вход программе подаются два целых числа: ***k*** и ***m*** ($0 \le k, m < 60$).

В одной строке выходных данных через пробел программа должна вывести искомое время: час и минуту отправления.

Пример

Входные данные	**Выходные данные**
20 22	8 30

Трамвай отправляется в 5:20, 5:30, 5:40, …, 8:10, 8:20, 8:30, ...

Придя на остановку в 8:22, Пётр Васильевич уедет на трамвае в 8:30.

Тесты

Входные данные	Выходные данные
20 22	8 30
23 22	8 23
43 22	8 23
23 23	8 23
43 23	8 23
23 24	8 33
43 24	8 33
11 51	8 51
11 52	9 1
12 53	9 2

Решение

```
var
   a, m, m1: longint;
Begin
   read(a, m);
   a := a mod 10;

   m1 := m mod 10;
   inc(m, a - m1);
   if a < m1 then
      inc(m, 10);
   writeLn(8 + m div 60, ' ', m mod 60)
End.
```

Задача 2
Дано неотрицательное целое число N (0 <= N <= 109). Напишите программу, которая подсчитывает количество пар одинаковых соседних цифр в двоичном представлении данного числа.
На вход программы подаётся число N.
Программа должна вывести одно целое число - вычисленное значение.

Пример

Ввод	Вывод	Примечание
25	2	$25_{10} = 11001_2$ 1ая пара **11***001* 2ая пара *11***00***1*
24	3	$24_{10} = 11000_2$ 1ая пара **11***000* 2ая пара *11***00***0* 3яя пара *110***00**

Тесты

Ввод	Вывод
0	0
51	3
64	5
127	6
213	1
255	7
32768	14
54613	1
65535	15
1000000000	16

Решение

```
program tur1_2020_2021;
var n: longint;
count, digit, preDigit: integer;
begin
  count := 0;
  read(n);
  digit := n mod 2;
  n := n div 2;
  while n>0 do
  begin
    preDigit := digit;
    digit := n mod 2;
    n := n div 2;
    if digit = preDigit then
      count := count + 1;
  end;
  write(count)
end.
```

Задача 3

Для того чтобы компьютеры поддерживали актуальное время, они могут обращаться к серверам точного времени SNTP (Simple Network Time Protocol). К сожалению, компьютер не может просто получить время у сервера, потому что информация по сети передаётся не мгновенно: пока сообщение с текущим временем дойдёт до компьютера, оно потеряет свою актуальность. Протокол взаимодействия клиента (компьютера, запрашивающего точное время) и сервера (компьютера, выдающего точное время) выглядит следующим образом:

1. Клиент отправляет запрос на сервер и запоминает время отправления *A* (по клиентскому времени).
2. Сервер получает запрос в момент времени *B* (по точному серверному времени) и отправляет клиенту сообщение, содержащее время *B*.
3. Клиент получает ответ на свой запрос в момент времени *C* (по клиентскому времени) и запоминает его. Теперь клиент, из предположения, что сетевые задержки при передаче сообщений от клиента серверу и от сервера клиенту одинаковы, может определить и установить себе точное время, используя известные значения *A, B, C*.

Вам предстоит реализовать алгоритм, с точностью до секунды определяющий точное время для установки на клиенте по известным *A, B* и *C*. При необходимости округлите результат до целого числа секунд по правилам арифметики (в меньшую сторону, если дробная часть числа меньше ½, иначе в большую сторону).

Возможно, что, пока клиент ожидал ответа, по клиентскому времени успели наступить новые сутки, однако известно, что между отправкой клиентом запроса и получением ответа от сервера прошло менее 24 часов.

Программа получает на вход три временные метки *A, B, C*, по одной в каждой строке.

Все временные метки представлены в формате «hh:mm:ss», где «hh» – это часы, «mm» – минуты, «ss» – секунды. Часы, минуты и секунды записываются ровно двумя цифрами каждое (возможно, с дополнительными нулями в начале числа).

Программа должна вывести одну временную метку в формате, описанном во входных данных, – вычисленное точное время для установки на клиенте. В выводе не должно быть пробелов, пустых строк в начале вывода.

Пример

Входные данные	Выходные данные
15:01:00 18:09:45 15:01:40	18:10:05

Ввод	**Вывод**	**Примечание**
15:01:00 18:09:45 15:01:40	18:10:05	Клиент отправил запрос в 15:01:00 по своим часам, сервер получил запрос в 18:09:45 по своим часам. Клиент получил ответ в 15:01:40, в этот момент точное время будет 18:10:05.

Тесты

Ввод	**Ожидаемый вывод**
15:01:00 18:09:45 15:01:40	18:10:05
23:58:10	00:02:13

00:01:17 00:00:01	
00:00:00 23:59:01 00:00:50	23:59:26
15:01:00 18:09:45 14:01:40	05:40:05
01:00:00 23:59:01 23:00:50	10:59:26

Решение

```
#!/usr/bin/env python3
from collections import namedtuple

Timestamp = namedtuple('Timestamp', ['h', 'm', 's'])
Timestamp.__str__ = lambda self: "{h:02}:{m:02}:{s:02}".format(h=self.h, m=self.m, s=self.s)

def get_timestamp(s):
	slices = ((0, 2), (3, 5), (6, 9),)
	return Timestamp(*(int(s[l:r]) for l, r in slices))

SECS_IN_MINUTE = 60
SECS_IN_HOUR = SECS_IN_MINUTE * 60
SECS_IN_DAY = SECS_IN_HOUR * 24

def get_seconds(t):
	ret = t.h * SECS_IN_HOUR
	ret += t.m * SECS_IN_MINUTE
	ret += t.s
	return ret

Timestamp.__int__ = get_seconds

def get_timestamp_from_seconds(secs):
	h = secs // SECS_IN_HOUR
	h %= 24
	secs %= SECS_IN_HOUR
	m = secs // SECS_IN_MINUTE
	secs %= SECS_IN_MINUTE
	s = secs
	return Timestamp(h, m, s)

A, B, C = [int(get_timestamp(input())) for _ in range(3)]

tm2 = 2 * B + (C - A) % SECS_IN_DAY
tm = tm2 // 2 + tm2 % 2
tm = get_timestamp_from_seconds(tm)
print(tm)
```

Задача 4

Шарики объемом 1 см^3 бросают в воду. Зная массу каждого шарика, найти самый тяжелый из них, который при этом не утонет в воде. Два любых шарика отличаются по массе.

Формат ввода

В первой строке программы вводится натуральное число **N** – количество шариков. Далее в **N** строчках вводится по одному положительному вещественному числу $\mathbf{m_i}$ – масса шарика под номером **i**. Масса указывается в граммах. Плотность воды – 1 г/см^3.

Формат вывода

Вывести одно целое число – номер самого тяжелого шарика, который при этом не утонет в воде. Если такого шарика нет, вывести 0.

Пример

Входные данные	Выходные данные
3 1.2 0.99 7.8	2

Тесты

Входные данные	Выходные данные
3 1.2 0.99 7.8	2
7 0.99 1.0 0.67 0.98 0.78 0.43 0.57	2
7 2.99 3.67 4.98 1.78 1.0 5.43 4.57	5
6 0.99 0.67 0.98 0.78 0.43 0.57	1
7 0.99	7

0.67 0.98 0.78 0.43 0.57 1.0	
7 0.97 0.67 0.98 0.78 0.43 0.57 0.99	7

Решение

```
program pzv2;
var
 i,n,num:integer;
 x,max:real;
begin
 readln(n);
 max:=-1;
 num:=0;
 for i:=1 to n do
 begin
   readln(x);
   if (x>max) and (x<=1)  then
   begin
     num:=i;
     max:=x;
   end
 end;
 writeln(num);
end.
```

Задача 5

Система «Легенда» отслеживает перемещения эскадры в северной Атлантике. Она отправляет в штаб сообщение, в котором перечисляются сигнатуры кораблей. Сообщение состоит из слов, каждое слово состоит только из строчных латинских букв. Если в слове букв **a** больше, чем остальных знаков, то считается, что это слово является сигнатурой авианосца. Если в сообщении больше половины сигнатур авианосца, то считается, что в эскадре есть авианосец.

Формат ввода

В первой строке вводится сначала натуральное число ***n*** – количество слов в сообщении (***n ≤ 1000***), затем в ***n*** следующих строках записано по одному слову.

Формат вывода

Вывести через пробел два целых числа. Первое число – количество сигнатур авианосца в сообщении. Второе число должно быть *0*, если в эскадре нет авианосца, и *1*, если авианосец есть.

Примеры

Входные данные	Выходные данные
5 baa sdaaaaf hgaafdaa bcd asdf	2 0

Тесты

Входные данные	Выходные данные
5 baa sdaaaaf hgaafdaa bcd asdf	2 0
7 aaaa aaaa b bm aaaa aaaa aaaa	5 1
3 aaaa aaaa aaaa	3 1
3 ab ab ab	0 0
10 aaa aaaaa baaaaavc nbaaaavc bv uibcs aaaa bc aaaa aaaa	6 1

Решение

```
program szv4;

function calc(s,s1:string):integer;
var
 i,k:integer;
```

```
begin
 k:=0;
 for i:=1 to length(s) do
 begin
   if copy(s,i,1)=s1 then
     k:=k+1;
 end;
 calc:=k;
end;

var
 n,i,c:integer;
 s:string;
begin
 readln(n);
 c:=0;
 for i:=1 to n do
   begin
     readln(s);
     if 2*calc(s,'a')>length(s) then
      c:=c+1;
   end;
 if (2*c>n) then
   writeln(c,' ',1)
   else
   writeln(c,' ',0)
end.
```

Задача 6 *Ситуационная задача*

В браузерной игре, посвященной единоборствам, герой каждый ход может нанести удар по одной из четырех частей тела: голове, груди, животу и ногам. Во время боя случайные события могут помешать ударить какую-то одну часть тела.

Сколько существует последовательностей ударов длиной n, если известна цепочка случайных событий, произошедших за эти ходы?

На вход программе подается строка длиной n (N≤20) символов, состоящая из цифр 0,1,2,3,4.

Символ 0 означает, что случайных событий на этом ходу нет.

Символы с 1 по 4 означают, что нельзя ударить по голове, груди, животу и ногам соответственно.

Вывести целое число – количество последовательностей ударов.

Пример

Ввод	Вывод
11	9

Тесты

Ввод	Вывод
1	3
000000	4096
11231	243
00123100	20736
0101020304	248832

Решение

```
program z10081;

function sum(p:integer; s1,sk:string):integer;
begin
 if s1=sk then
   sum := 0
   else
   sum:=p;
end;

var
 s,s1:string;
 i,a,b,c,d,psum:integer;
begin
 readln(s);
 psum:=1;
 s1:=copy(s,1,1);
 a:=sum(psum,s1,'1');
 b:=sum(psum,s1,'2');
 c:=sum(psum,s1,'3');
 d:=sum(psum,s1,'4');
 for i:=2 to length(s) do
  begin
   s1:=copy(s,i,1);
   psum:=a+b+c+d;
   a:=sum(psum,s1,'1');
   b:=sum(psum,s1,'2');
   c:=sum(psum,s1,'3');
   d:=sum(psum,s1,'4');
  end;
 writeln(a+b+c+d);
end.
```