Заключительный этап Олимпиады школьников «Шаг в будущее» по профилю «Инженерное дело» специализации «Техника и технологии (программирование)» (общеобразовательный предмет информатика), весна 2021 год

11 класс

Вариант 1

Задача 1 (5 баллов)

Условие
Определить, сколько цифр заданного натурального числа являются его делителями.

Входные данные: натуральное число, не превышающее 10^8

Выходные данные: количество цифр числа, которые одновременно являются его делителями.

Пример

Входные данные	Результат
1050	2
333	3

Проверочные тесты

Входные данные	Ожидаемый результат
1050	2
333	3
1	1
200	1
23	0

Пример решения

```
n = int(input())
ans = 0
k = n;
while k:
    t = k % 10
    if t != 0 and n % t == 0:
        ans += 1
    k //= 10

print(ans)
```

Задача 2 (8 баллов)

Условие

К 16-разрядному числу применили операцию циклического побитового сдвига вправо, в результате которой была допущена ошибка в одном разряде.

Заданы исходное число и число после сдвига, записанные в 16-ричной системе счисления. Требуется определить, в каком разряде нового числа допущена ошибка, и указать его номер и правильное значение.

Разряды нумеруются справа налево (от младшего к старшему) от 1 до 16.

Входные данные: два 16-разрядных числа, записанных в 16-ричной системе счисления и состоящих ровно из 4-х цифр 0..F, и количество разрядов, на которые осуществлён сдвиг, от 1 до 15. Цифры A..F всегда задаются заглавными буквами. Исходные значения записаны в одну строку через пробел.

Выходные данные: номер ошибочного разряда в числе после сдвига и его верное значение.

Пример

Входные данные	Результат
1ABC C0AB 4	9 1
FFFF FFF7 15	4 1

Проверочные тесты

Входные данные	Ожидаемый результат
1ABC C0AB 4	9 1
FFFF FFF7 15	4 1
1111 1110 4	1 1
1000 3000 15	13 0
AAAA 5545 1	5 1

Пример решения

```
s = input()
s = s.split()
x = list(s[0])
y = list(s[1])
stepp = int(s[2])
hexs = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D',
'E', 'F']
bins = ['0000', '0001', '0010', '0011', '0100', '0101', '0110', '0111',
'1000', '1001', '1010', '1011', '1100', '1101', '1110', '1111']
xbin = list()
ybin = list()
for elem in x:
    xbin.append(bins[hexs.index(elem)])
xbin = ''.join(xbin)
# print(xbin)
for elem in y:
```

```
    ybin.append(bins[hexs.index(elem)])
ybin = ''.join(ybin)
# print(ybin)

xbin = xbin[16-stepp:] + xbin[:16-stepp]
# print(xbin)
for i in range(len(xbin)):
    if xbin[i] != ybin[i]:
        print(16-i, xbin[i])
```

Задача 3 (10 баллов)

Условие
Задана строка, состоящая из цифр и знаков "+" и "-". Требуется определить, можно ли расставить в ней скобки так, чтобы получившееся выражение было равно нулю.

Входные данные: строка символов, длина которой не превышает 20.

Выходные данные: выражение, равное нулю, если его возможно получить, и слово NO, если невозможно.

Пример

Входные данные	Результат
3-1+2	3-(1+2)
4+3+2	NO

Проверочные тесты

Входные данные	Ожидаемый результат
3-1+2	3-(1+2)
4+3+2	NO
1-1	1-1
4-2+3+1	4-(2+3)+1
7+9-8+2+1+5	7+9-(8+2+1+5)

Пример решения

```
def do_marks(x, a, b):
    if x[a][0] == '-':
        for i in range(a + 1, b + 1):
            x[i][0] = '-' if x[i][0] == '+' else '+'
    return x


def do_sum(x):
    sum = 0
    for it in x:
        if it[0] == '-':
```

```
            sum -= it[-1]
        else:
            sum += it[-1]
    return sum

def find_bracket(x):
    n = len(x)
    for i in range(n):
        for j in range(i + 1, n):
            tmp = do_marks(x, i, j)
            s = do_sum(tmp)
            do_marks(x, i, j)
            if s == 0:
                return i, j
    return None, None

def print_brackets(x, brackets):
    s = ''
    for it in brackets:
        x[it[0]] = list((x[it[0]][0], '(', x[it[0]][1]))
        x[it[1]].append(')')
    for it in x:
        for jt in it:
            s += str(jt)
    if s[0] == '+':
        s = s[1:]
    print(s)

s = input()
if s[0] != '-':
    s = '+' + s
data = []
for i in range(0, len(s), 2):
    data.append(s[i:i + 2])

n = len(data)
for i in range(n):
    data[i] = list(data[i])
    data[i][-1] = int(data[i][-1])

if do_sum(data) == 0:
    print(s if s[0] == '-' else s[1:])
else:
    a, b = find_bracket(data)
    if a == None:
        print('NO')
    else:
        print_brackets(data, [[a, b]])
```

Задача 4 (12 баллов)

Условие

Дан прямоугольник размерами NxM (N и M - целые в диапазоне от 1 до 50). Требуется определить, можно ли заполнить его площадь прямоугольными блоками размером KxL (K и L - целые от 1 до 5) так, чтобы блоки не накладывались друг на друга, целиком попадали в прямоугольник и их не требовалось обрезать.

Блоки разрешается поворачивать на 90 градусов.

Входные данные: 4 числа N, M, K, L, записанные через пробел.

Выходные данные: необходимое количество блоков, если ими можно заполнить прямоугольник, или 0, если заполнить прямоугольник с соблюдением указанных ограничений невозможно.

Пример

Входные данные	Результат
2 4 1 3	0
4 3 2 1	6

Проверочные тесты

Входные данные	Ожидаемый результат
2 4 1 3	0
4 3 2 1	6
1 1 1 1	1
4 7 3 3	0
12 7 4 3	7

Пример решения

```
def solve(n, m, k, l):
    if (n * m) % (k * l) != 0:
        return 0
    else:
        if max(n, m) < max(k, l):
            return 0
        else:
            return (n * m) // (k * l)



n, m, k, l = map(int, input().split())
ans = solve(n, m, k, l)
print(ans)
```

Задача 5 (15 баллов)

Условие

Пятиклассник Вася отправился на прогулку, взяв с собой сотовый телефон. Перед этим его родители для Васиной безопасности и собственного спокойствия нашли и установили специальную программу, которая передаёт им сведения о ближайших к Васе антеннах сотовой связи.

Помогите Васиным родителям написать программу, которая по истории Васиных перемещений определит, какое расстояние прошёл Вася.

Входные данные: в первой строке записано целое число N (от 3 до 10) - количество вышек сотовой связи на Васином маршруте. Далее идёт N строк, в которых через пробел записаны координаты X и Y соответствующей антенны (в метрах, целые, от -1000 до 1000 каждая).

В следующей строке записано целое число K (от 2 до 100) - количество точек Васиной прогулки, в которых передавались сведения о его местоположении. Далее идёт K строк, в которых через пробел указаны 7 чисел: время в минутах с начала прогулки (целое от 0 до 300), номера и мощности сигнала с трёх ближайших к Васе антенн. Антенны нумеруются по порядку ввода с 1, мощность задаётся в виде процентов от 1 до 100, целым числом. Первая точка имеет время 0, последняя соответствует концу прогулки.

В рамках задачи считать, что:

- Вася гуляет по ровной местности;
- между зафиксированными точками идёт только по прямой;
- все антенны находятся на уровне земли, имеют одинаковую мощность и Васин телефон теряет с ними связь на расстоянии в 1000 метров.

Выходные данные: расстояние, пройденное Васей за прогулку, округлённое со сотен метров.

Пример

Входные данные	Результат
4 0 0 500 0 0 1000 500 1000 3 0 1 90 3 10 2 49 10 1 10 3 90 4 49 30 3 49 4 90 2 10	1300

Проверочные тесты

Входные данные	Ожидаемый результат
4 0 0 500 0 0 1000 500 1000 3	1300

0 1 90 3 10 2 49 10 1 10 3 90 4 49 30 3 49 4 90 2 10	
3 100 0 200 100 300 0 2 0 1 90 2 90 3 78 5 1 78 2 90 3 90	200
3 100 0 200 100 300 0 2 0 1 90 2 90 3 78 5 1 78 2 78 3 90	300
7 100 100 200 100 300 100 100 500 200 500 300 500 100 900 3 0 1 86 2 78 3 68 10 7 86 5 46 6 49 30 3 90 2 86 3 78	2000
7 100 100 200 100 300 100 100 500 200 500 300 500 100 900 5 0 1 86 2 78 3 68 5 1 80 4 80 2 78 10 6 80 3 80 5 78 15 6 90 5 86 4 78 25 3 90 2 86 1 78	1400

Пример решения

```
from math import sqrt
N = int(input())
cells = []
for i in range(N):
    X, Y = map(int, input().split())
    cells.append([X, Y])

K = int(input())
data_dict = {}
for i in range(K):
    inp = list(map(int, input().split()))

    data_dict[inp[0]] = [[inp[j], inp[j+1]] for j in range(1,len(inp),2)]

db = []
for key in sorted(data_dict.keys()):
    db.append(data_dict[key])

points = []
for i in range(K):
    data = db[i]
    y1b = (100-data[0][1])/100*1000
    y2b = (100-data[1][1])/100*1000
    y3b = (100-data[2][1])/100*1000
    i1 = data[0][0]
    i2 = data[1][0]
    i3 = data[2][0]
    x1 = cells[i1-1][0]
    x2 = cells[i2-1][0]
    x3 = cells[i3-1][0]
    y1 = cells[i1-1][1]
    y2 = cells[i2-1][1]
    y3 = cells[i3-1][1]

    if x2-x1 == 0:
        a = (x3*x3-x2*x2) + (y3*y3-y2*y2) + y2b*y2b - y3b*y3b
        b = 2*(x3-x2)
        z = 2*(y3-y1)-4*(y3-y2)*(x3-x1)/b
    else:
        a = (x2*x2-x1*x1) + (y2*y2-y1*y1) + y1b*y1b - y2b*y2b
        b = 2*(x2-x1)
        z = 2*(y3-y1)-4*(y2-y1)*(x3-x1)/b


    c = -(x3-x1)*2*a/b+ y3*y3 - y1*y1 + x3*x3 -x1*x1 + y1b*y1b -y3b*y3b

    y = c/z

    if x2-x1 == 0:
        x = (a-2*y*(y3-y2))/b
    else:
        x = (a-2*y*(y2-y1))/b

    points.append([x, y])

answer = 0
```

```
for i in range(len(points)-1):
    x1 = points[i][0]
    y1 = points[i][1]
    x2 = points[i+1][0]
    y2 = points[i+1][1]
    answer += sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1))

print(int(round(answer, -2)))
```