Аверьянов Андрей Александрович

Балл: 57

Задача №1

Условие

Однажды между Платоном и Ксенофонтом, учениками Сократа, разгорелся нешуточный спор, кто из них умнее. Они никак не могли его разрешить и обратились за помощью к своему учителю. Мудрый философ предложил им решить одну математическую задачу. Он сказал: «Кто из вас быстрее догадается о том, как связаны числа между собой числа в парах, тот и может считать себя самым умным». Он предложил им следующие пары чисел

1	1
2	2
3	3
10	11
100	121
100000	162151

Но ни один из них не справился и не смог найти закономерность. Помогите им понять, что же здесь задумал великий Сократ. Напишите программу, которая получает на вход одно целое число N ((0 <= N <= 10^{18}), и в качестве ответа выводит одно целое число -парное ему.

Примеры

Входные данные	Выходные данные
1	1
2	2
3	3
10	11
100	121
100000	162151

Исходный код

Задача №2

Условие

В школьном театре решили поставить спектакль по сценарию выпускника лицея Клевцова Саши. На кастинг пришла половина учеников школы, но однофамильцев среди них не оказалось. На каждую роль было отобрано не более восьми учащиеся, которые могут эту роль сыграть. Сколькими способами можно распределить роли между учащимися?

В первой строке на вход программы подаётся количество ролей. В каждой последующей строке вводится информация о какой-то одной роли: сначала название роли без пробелов далее через пробел фамилии отобранных на неё учащихся. Фамилии отделены пробелом, сами фамилии пробелов не содержат, претендентов на каждую роль не меньше одного.

Максимальное количество ролей - **8**, учащихся на каждую роль не более **8**.

Выведите одно целое число – сколькими способами можно выстроить очередь.

Пример

Входные данные	Выходные данные
2 Король Петров Иванов Принц Иванов Сидоров	3

Комментарий к примеру. Если короля играет Петров, тогда для принца 2 варианта актёра. Если короля играет Иванов, то для принца один вариант актёра. Итого 3 варианта.

Исходный код

```
def factorial(ch):
    output = 1
    for i in range(1,ch+1):
        output*=i
    return output
roles = []
names = []
count = 0
n = int(input())
for i in range(n):
    buffer = str(input()).split()
    roles.append(buffer[0])
    names.append(buffer[1::])
variants = 0
buff = ""
for i in range(len(names)):
    for k in range(len(names[i])):
        if(names[i][k] not in buff):
            buff+=names[i][k]
            count+=1
            print(buff)
print(count)
print(int(factorial(count)/(factorial(n)*factorial(count-n))))
```

Задача №3

Условие

Исполнитель получает на вход натуральное число X (не превышающее 10^6). По этому числу, точнее по его представлению в четырнадцатеричной системе счисления строится новое число Y по следующим правилам.

В четырнадцатеричном представлении числа X предпоследняя цифра увеличивается на 1 (гарантируется, что в четырнадцатеричном представлении X числа больше 2-х цифр). Например, 4885_{10} = 1A**C**D$_{14}$ -> 1A**D**D$_{14}$ = 4899_{10}.

Если предпоследняя цифра D, тогда предпоследняя цифра становится 0, а последняя изменяется по следующему принципу: четная увеличивается н 1, а нечетная уменьшается на 1. Например, последняя цифра нечетная 4887_{10} = 1A**D*1***$_{14}$ -> 1A**0*0***$_{14}$ = 4704_{10}, последняя цифра четная 4888_{10} = 1A**D*2***$_{14}$ -> 1A**0*3***$_{14}$ = 4707_{10}.

Введем понятие расстояния

Ah = | Полученное_число – Исходное_число |

Напишите программу, которая будет считать наибольшее расстояние Ah для чисел из заданного интервала [A,B] и наименьшее исходное число, для которого оно было вычислено.

На вход программы подаётся два целых числа A и B (20 ≤ A ≤ B ≤1 000 000), записанных через пробел.

Программа должна вывести два числа наибольшее расстояние Ah и через пробел исходное число, для которого оно было посчитано.

Входные данные	Вывод	Примечание
4884 4888	**183 4887**	|4898 - 4884| =14 (1A**D**C$_{14}$ – 1A**C**C$_{14}$) |4899 - 4885| =14 (1A**D**D$_{14}$ – 1A**C**D$_{14}$) |4705 - 4886| =181 (1A**01**$_{14}$ – 1A**D0**$_{14}$) |4704 - 4887| =<u>183 (1A**00**$_{14}$ – 1A**D**1$_{14}$)</u> |4707 - 4888| =181 (1A**03**$_{14}$ – 1A**D**2$_{14}$)
4883 4885	**14 4883**	|4897 - 4883|=14 (1A**D**B$_{14}$ – 1A**C**B$_{14}$) |4898 - 4884|=14 (1A**D**C$_{14}$ – 1A**C**C$_{14}$) |4899 - 4885|=14 (1A**D**D$_{14}$ – 1A**C**D$_{14}$)

Исходный код

```
def convertDecTo14(ch1):
    outpCh1 = ""
    outpCh2 = ""
    ost = {10:"A", 11:"B", 12:"C", 13:"D"}
    while ch1 > 0:
        buffer = ch1%14
        if(buffer > 9):
            outpCh1 += ost[buffer]
        else:
            outpCh1 += str(buffer)
        ch1 = ch1//14
    return outpCh1[::-1]
def preobr(ch:str):
    ost = {"0":0, "1":1, "2":2, "3":3, "4":4, "5":5, "6":6, "7":7, "8":8, "9":9, "A":10,"B":11,"C":12,"D":13}
    ost2 = {10: "A", 11: "B", 12: "C", 13: "D"}
    output = ""
    if(ch[-2] != 'D'):
        number = -3
        buffer = int(ost[ch[-2]])
        buffer += 1
        isDop = False
        if(buffer > 13):
            isDop = True
            buffer -= 14
            if (buffer > 9):
                ch = ch[:-2] + str(ost2[buffer]) + ch[-1::]
            else:
                ch = ch[:-2] + str(buffer) + ch[-1::]
        else:
            if (buffer > 9):
                ch = ch[:-2] + str(ost2[buffer]) + ch[-1::]
            else:
                ch = ch[:-2] + str(buffer) + ch[-1::]
        while isDop == True:
            if((number*-1) <= len(ch)):
                buffer = int(ost[ch[number]])
                buffer += 1
                if (buffer > 13):
                    isDop = True
                    buffer -= 14
                    if (buffer > 9):
                        ch = ch[:number] + str(ost2[buffer]) + ch[(number+1)::]
                    else:
                        ch = ch[:number] + str(buffer) + ch[(number+1)::]
                else:
                    isDop = False
                    if (buffer > 9):
                        ch = ch[:number] + str(ost2[buffer]) + ch[(number+1)::]
                    else:
                        ch = ch[:number] + str(buffer) + ch[(number+1)::]
                number -= 1
            else:
                buffer = int(0)
                buffer += 1
                if (buffer > 13):
                    isDop = True
                    buffer -= 14
                    if (buffer > 9):
                        ch = str(ost2[buffer]) + ch
                    else:
                        ch = str(buffer) + ch
                else:
                    isDop = False
                    if (buffer > 9):
                        ch = str(ost2[buffer]) + ch
                    else:
                        ch = str(buffer) + ch
                number -= 1
    else:
        if(int(ost[ch[-1]])%2 == 0):
            ch = ch[:-2:] + '0' + ch[-1]
            buffer = int(ost[ch[-1]])
            buffer += 1
            if (buffer > 9):
                ch = ch[:-1:] + str(ost2[buffer])
            else:
                ch = ch[:-1:] + str(buffer)
        else:
            ch = ch[:-2:] + '0' + ch[-1]
            buffer = int(ost[ch[-1]])
            buffer -= 1
            if (buffer > 9):
                ch = ch[:-1:] + str(ost2[buffer])
            else:
                ch = ch[:-1:] + str(buffer)
    return ch

def convert14toDec(ch):
    ost = {"0": 0, "1": 1, "2": 2, "3": 3, "4": 4, "5": 5, "6": 6, "7": 7, "8": 8, "9": 9, "A": 10, "B": 11, "C": 12, "D": 13}
    outp = 0
    ch = str(ch)
    for i in range(len(str(ch))):
        outp += int(ost[ch[i*-1-1]])*(14**(i))
```

```
    return outp
start, fin = list(map(str, input().split()))
start = int(start)
fin = int(fin)
maxim = 0
delta = 0
numb = 0
for i in range(start, fin+1):
    delta = abs(convert14toDec(preobr(convertDecTo14(i)))-i)
    if(delta >maxim):
        maxim=delta
        numb = i
print(maxim, numb)
```

Задача №4

Условие

Несколько датчиков отслеживают давление внутри капсулы и передают показания на центральный пост. Показание каждого датчика строится следующим образом: сначала пишется двузначный номер датчика, потом к нему приписывается измеренное давление. Ни один номер датчика не начинается с нуля. Найдите наибольшее значение давления, зафиксированное датчиками. Все числа записаны в десятичной системе счисления.

Формат ввода

В строке вводится сначала целое число ***n*** – количество показаний датчиков ($n \leq 1000$), затем ***n*** натуральных чисел в диапазоне от 101 до 99999 включительно, все числа отделены друг от друга одним или несколькими пробелами.

Формат вывода

Вывести одно целое число – наибольшее значение давления.

Пример

Входные данные	Выходные данные
4 1001 1120 1130 1240	40

Исходный код

```
data = []
sch = str(input()).split()
for i in range(int(sch[0])):
    data.append(int(sch[i+1][2::]))
print(max(data))
```

Задача №5

Условие

Спутник ведет наблюдение на орбите Сатурна. Раз в минуту он отсылает закодированное сообщение на Землю. Сообщение представляет из себя набор латинских букв. Если спутник фиксирует полярное сияние в атмосфере планеты, в текст сообщения вставляется набор знаков **pls**. Если полярного сияния не наблюдается, набор знаков **pls** отсутствует в сообщении. Найдите наибольшую продолжительность полярного сияния на основании сообщений спутника.

Формат ввода

В первой строке вводится сначала целое число ***n*** – количество слов ($n \leq 1000$), затем в ***n*** следующих строках записано по слову. Слова состоят только из строчных латинских букв.

Формат вывода

Вывести одно целое число – наибольшую продолжительность полярного сияния.

Примеры

Входные данные	Выходные данные

3 adsaplsds plsdf abvf	2
4 asdpls sdfg sdfr pls	1

Исходный код

```
data = []
for i in range(int(input())):
    data.append(input())
maxim = 0
maxData = []
for i in range(len(data)):
    if("pls" in data[i]):
        maxim += 1
    else:
        maxData.append(maxim)
        maxim = 0
maxData.append((maxim))
print(max(maxData))
```