

## Задача 1

Предприятие непрерывного цикла «Гранулит» впервые начинает свою работу в понедельник 27 ноября 2023 года. В ходе производства некоторое количество воды объемом  $V$  литров загрязняется. Каждую среду вся загрязненная вода заливается в очистные сооружения. Цикл очистки заканчивается к утру понедельника. За это время может очиститься  $M$  литров воды. Вся очищенная вода возвращается в производство утром этого же понедельника. В очистных сооружениях может находиться не более  $Z$  литров воды. Необходимо определить на какой по счету день работы предприятия, очистные сооружения переполнятся, то есть количество воды в очистных сооружениях станет больше  $Z$  литров. 27 ноября 2023 года считать первым днем работы предприятия.

### Формат ввода

На вход программе в одной строке подается три целых числа, записанные через пробел  $V, M, Z$  ( $1 \leq V, M, Z \leq 10^9$ ).

### Формат вывода

Вывести одно целое число – номер дня, в который произойдет переполнение очистных сооружений.

Если переполнение не произойдет никогда, вывести число 0.

### Пример

Входные данные	Вывод	Пояснение
100 50 110	10	В среду поступит 100 литров. В понедельник уйдет 50. Останется 50 литров. В среду поступит 100 литров. Получится 150 литров, что больше 110. Произойдет переполнение. Это день номер 10 от момента начала работы предприятия.

### Решение

```
V, M, Z = map(int, input().split())
k = 0
d = V - M
if V > Z:
    print(3)
elif d <= 0:
    print(0)
else:
    print ((Z - V + d) // d) * 7 + 3)
```

## Задача 2

Игра «Космическое домино».

Правила.

1. В игре участвуют только ТРЕХЗНАЧНЫЕ числа.

2. Перед началом игры для каждого игрока случайным образом генерируется некоторый диапазон трехзначных чисел, из которого и ТОЛЬКО из него он может выбирать числа для продолжения игры.
3. Первые два стартовых числа генерируются тоже компьютером.
4. Игроки ходят по очереди, доставляя сопряженное число из своего диапазона (если такое есть) к левому или правому концу цепочки.
5. Если у игрока нет в диапазоне числа, сопрягаемого ни с одним из концов цепочки, то игрок пропускает ход.
6. Если игрок в своем диапазоне может найти число, сопрягаемое с обоими концами цепочки, то игра завершается его выигрышем.

Определение: число  $M$  называется **сопрягаемым** с числом  $N$ , если оно построено по следующим правилам:

1. Если  $N$  нечетное, то число  $M$  начинается с нечетной цифры, если  $N$  – четное, то с четной (но не с 0!)
2. Последние две цифры числа  $M$  есть сумма цифр числа  $N$ .

Заметим, что оба числа являются трехзначными!

Например, если  $N = 213$ , тогда  $M$  может быть 106, 306, 506, 706 или 906. Для числа  $N = 812$ , число  $M$  может быть только 211, 411, 611 или 811.

Примечание: числа в цепочке могут повторяться.

### Входные данные:

$L$  и  $R$  два натуральных трехзначных числа через пробел в одной строке. Левое и правое число в цепочке соответственно.

$A$  и  $B$  два натуральных трехзначных числа через пробел в одной строке. Диапазон, который выпал игроку в начале игры.  $A < B$ . Число  $A$  или  $B$  также может быть выбрано игроком для хода, если является сопрягаемым с концом цепочки.

### Выходные данные:

0, если игрок пропускает ход.

Максимальное из сопрягаемых чисел, которым игрок может завершить игру с указанием перед ним без пробела литеры “V”.

Если игрок может сделать ход, но не завершить игру, то максимальное из сопрягаемых с концами чисел с указанием перед ним без пробела литеры “L”, если его надо поставить к левому концу, и литеры “R”, если к правому. Если число можно добавить в любой конец, то ставим его в **правый** конец.

Входные данные	Вывод	Примечание
812 434 400 800	V611	В диапазон игрока попало два числа, которыми можно завершить игру 411 и 611 (они сопрягаются с обоими концами цепочки). Наибольшее из них 611.
213 812 306 520	L506	На данный момент игровая цепочка выглядит: 213 ... 812

		<p>Множество сопрягаемых с концами цепочки чисел: 106, 306, 506, 706, 906, 211, 411, 611 или 811.</p> <p>Из них в распоряжении игрока только 3 числа: 306, 506, 411.</p> <p>Среди них нет числа, которое бы сопрягалось с обоими концами (игра не завершается).</p> <p>Наибольшее из них 506, и оно сопрягается с левым концом.</p> <p>После хода цепочка будет иметь вид: 506 213 ... 812</p>
213 812 350 400	0	<p>Множество сопрягаемых с концами цепочки чисел: 106, 306, 506, 706, 906, 211, 411, 611 или 811.</p> <p>К сожалению, в диапазон игрока не входит ни одно из чисел, поэтому он пропускает ход.</p>

Решение

```
def linking(n):
```

```
    s = sum([int(x) for x in str(n)])
```

```
    if n % 2 == 0:
```

```
        m = [ y * 100 + s for y in [2,4,6,8]]
```

```
    else:
```

```
        m = [ y * 100 + s for y in [1,3,5,7,9]]
```

```
    return m
```

```
left, right = map(int, input().split())
```

```
a, b = map(int, input().split())
```

```
mL = [x for x in linking(left) if a <= x <= b]
```

```
mR = [x for x in linking(right) if a <= x <= b]
```

```
res = [x for x in mL if x in mR]
```

```
hod = mL + mR
```

```
print(res, hod)
```

```
if not hod:
```

```
    print(0)
```

```
elif res:
```

```
    mn = max(res)
```

```
    print("V" + str(mn))
```

```
else:
```

```
    mn = max(hod)
```

```

if mn in mR:
    print("R" + str(mn))
elif mn in mL:
    print("L" + str(mn))

```

### Задача 3

Исполнитель Говорун действует в трехмерном пространстве с декартовой системой координат.

Исполнитель Говорун умеет исполнять только одну команду **Переместиться вдоль O на N**, (O - x, y или z; N - целое число), которая моментально перемещает исполнителя вдоль оси **O** на **N** единиц.

Примеры команд			
Позиция Говоруна	Команда	Новая позиция	
(1,3,4)	x 100	101,3,4	
(0,0,0)	y -10	0, -10, 0	
5,6, -7)	z 8	5. 6. 1	

Исполнитель Говорун был запущен с корабля "Синяя чайка" (координаты корабля - целые числа) для патрулирования пространства.

В ходе патрулирования Говорун пристальное внимание уделяет "*особым*" точкам.

Точка с координатами (*a, b, c*) является "*особой*", если  $(|a|+|b|+|c|)$  — простое число.

Определите сколько различных "*особых*" точек с целочисленными координатами посетил Говоруна, считая начальную и конечную точки.

#### Входные данные:

1-я строка содержит координаты корабля "Синяя чайка" (все числа целые и по модулю не более  $10^9$ );

2-я строка содержит число *N* — количество команд ( $N \leq 10^5$ ).

В следующих *N* строках записаны параметры для команд исполнителя — буква и целое число (число по модулю не превосходит 1000).

Начальные координаты Говоруна совпадают с координатами корабля "Синяя чайка", позиция корабля "Синяя чайка" не является "*особой*" точкой.

#### Выходные данные

Ответ на задачу - количество различных "*особых*" целочисленных точек, в которых успеет побывать Исполнитель до завершения программы

#### Примеры:

входные данные	Выходные данные	Пояснение
12 -3 3 5 y -1 z -2	3	Говорун стартует из точки (12, -3, 3). После 1-й команды переходит в точку (12, -4, 3) — это " <i>особая</i> " точка ( $ 12 + -4 + 3 = 19$ - простое число)

z 1 z -1 x -24		<p>После 2-й команды переходит в точку (12, -4, 1) — это "особая" точка (<math> 12 + -4 + 1 =17</math> - простое число)</p> <p>После 3-й команды переходит в точку (12, -4, 2)</p> <p>После 4-й команды переходит в точку (12, -4, 1) — это особая точка (<math> 12 + -4 + 1 =17</math> - простое число), но она не учитывается (Говорун был в ней после выполнения 2-й команды)</p> <p>После 5-й команды переходит в точку (-12, -4, 1) — это особая точка (<math> -12 + -4 + 1 =17</math> - простое число)</p> <p>В результате Говорун побывает в 6 точках (учитываем начальную и финальную), из которых 4 являются "особыми".</p> <p>В точке (12, -4, 1) Говорун побывает 2 раза.</p>
286 -285 287 5 x -5 y -1 y 1 z -5 z 2	1	Говорун дважды побывает в "особой точке" (281, -285, 287)

## Решение

```
def min_del(n):
    if n%2==0 and n> 0: return 2
    if n<9 :return n
    p=3
    while n%p !=0 and p*p<n : p+=2
    if p*p>n : p=n
    return p
```

```
def is_prost(n):
    if n<2 : return False
    if n==min_del(n): return True
    return False
```

```
x,y,z=map(int,input().split())
N=int(input())
tt_pr=set()
tt1=set()
tt0=set()
n=abs(x)+abs(y)+abs(z)
if is_prost(n):
    tt_pr.add((x,y,z))
    tt1.add(n)
else : tt0.add(n)
```

```

for _ in range(N):
    a,k =input().split()
    k=int(k)
    if a=='x' : x+=k
    if a=='y' : y+=k
    if a=='z' : z+=k
    t=(x,y,z)
    n=abs(x)+abs(y)+abs(z)
    if n in tt0 : continue
    elif (x,y,z) in tt_pr : continue
    elif n in tt1 : tt_pr.add((x,y,z))
    elif is_prost(n):
        tt_pr.add((x,y,z))
        tt1.add(n)
    else : tt0.add(n)
answer=len(tt_pr)
print(answer)

```

#### Задача 4

Младшеклассники в процессе изучения арифметики играют в игру. Из стопки по одной тянут карточки с цифрами и выкладывают их в цепочку. Процедура повторяется, пока в стопке есть карточки. Какое наибольшее четырёхзначное число можно собрать таким образом, если известна последовательность карточек?

#### Формат ввода

На вход программе в первой строке подается натуральное число  $N$  ( $4 \leq N \leq 10000$ ) – количество карточек. Далее в  $N$  строках подаётся по одному целому числу – от 0 до 9. Гарантируется, что в первой строчке подаётся число больше 0.

#### Формат вывода

Вывести одно целое число – наибольшее возможное, которое можно получить по правилам, описанным в условии задачи.

#### Пример

Ввод	Вывод
5 7 2 3 6 9	9763

#### Решение

```

def count(x):
    s = 0

```

```

while x>0:
    if x%12 in [0,3,6,9]:
        s+=1
        x=x//12
return s

n = int(input())
answer = 0
for i in range(n):
    x = int(input())
    answer+=count(x)

print(answer)

```

### Задача 5

Старшеклассник Миша собирает робота, который должен ездить по лабиринту. Всего робот умеет выполнять 12 различных команд, но для нас представляют интерес четыре из них – «на клетку вперёд», «на клетку назад», «на клетку влево», «на клетку вправо». Миша решил передавать роботу инструкции в виде цифр числа: робот получает число, переводит его в двенадцатеричную систему и выполняет соответствующие цифрам команды. Коды команд, отвечающих за перемещение, кратны трём.

На вход подаётся N чисел с наборами команд. Сколько раз робот изменит положение, если считать, что он не встречает препятствий?

#### Формат ввода

На вход программе в первой строке подается натуральное число N ( $N \leq 10000$ ) – количество наборов команд. Далее в N строках на вход подаётся по одному целому числу в диапазоне от 0 до  $4 \cdot 10^9$  – набор двенадцатеричных команд, записанных в десятичной системе счисления.

#### Формат вывода

Вывести одно целое число – сколько раз робот изменит положение.

#### Пример

Ввод	Вывод
4 33 130 144 12	4

#### Решение

```

def count(x):
    s = 0
    while x>0:
        if x%12 in [0,3,6,9]:
            s+=1
        x=x//12
    return s

n = int(input())
answer = 0
for i in range(n):
    x = int(input())
    answer+=count(x)

print(answer)

```

### Задача 6

Агрокомплекс «Дикое поле» представляет собой **n** полей, вытянувшихся в ряд. Поля пронумерованы, начиная с единицы. Комплекс находится в слабозаселённой зоне, и поэтому наблюдение за полями осуществляется с помощью дронов «Пеларгония». Агрокомплекс располагает двумя такими дронами. Один начинает облёт с первого поля, второй – с поля номер **n**; оба дрона движутся навстречу друг другу, последовательно осматривая поля. Они отличаются по размерам, поэтому на осмотр каждого поля требуется разное время. При этом перелёт с одного поля на другое занимает фиксированное время. Определите, за какое минимальное время дроны осмотрят все поля. Если одна «Пеларгония» уже осматривает поле, вторая на него не влетает.

### Формат ввода

На вход программе в первой строке подаётся натуральное число  $N$  ( $3 \leq N \leq 1000000$ ) – количество полей. Во второй строке подаётся натуральное число  $T$  ( $T \leq 1000$  минут) – время перелёта между соседними полями. Далее в  $N$  строках подаётся по одному натуральному числу  $t_i$  ( $t_i \leq 1000$  минут) – время осмотра поля номер  $i$ .

### Формат вывода

Вывести одно натуральное число – минимально возможное время осмотра всех полей агрокомплекса.

### Пример

Ввод	Вывод
5 1 2 4 3	11



5	
4	

## Решение

```
n = int(input())
d = int(input())
data = n*[0]
for i in range(n):
    data[i] = int(input())

mnm = sum(data)+n*d
pref = [0]
for t in data:
    pref.append(pref[-1]+t)
for i in range(1,n):
    s1,s2=0,0
    s1=pref[i]+s1+(i-1)*d
    s2=pref[n]-pref[i]+(n-i-1)*d
    if max(s1,s2)<mnm:
        mnm = max(s1,s2)

print(mnm)
```