Шаблоны ЕГЭ 2023 программированием на языке Python.

Информатика Умскул (https://vk.com/umsch\_i)

Телега и все другие соцсети @infa\_vikusya - https://t.me/infa\_vikusya

# Задание 2.1.

Логическая функция задана выражением ((y ⟶ z) ⟶ (x ⋀ y)) ⋀ (x ⟶ w), а у нас есть неполная таблица истинности (содержит не все наборы аргументов и значений), которая представлена ниже.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ? | ? | ? | ? | F |
| 0 | 0 | 0 |  | 1 |
| 0 | 0 |  | 1 | 1 |
| 0 |  |  | 1 | 1 |

При условии, что функция во всех случаях тождественно истинна, запишите в ответ имена переменных в том порядке, в котором они идут по таблице.

## Решение:

print("x", "y", "z", "w")

for x in range(2):

for y in range(2):

for z in range(2):

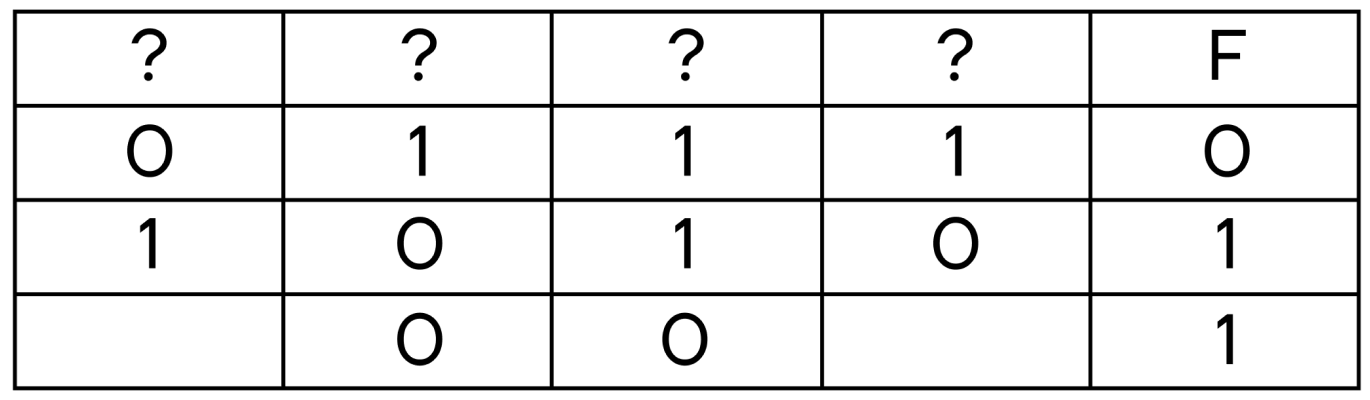
for w in range(2):

if (((y <= z) <= (x and y)) and (x <= w)) == 1:

print(x, y, z, w)

# Задание 2.2.

Логическая функция задана выражением ((y → x) ≡ (w → ￢z)) ∧ (w ∨ x), а у нас есть неполная таблица истинности (содержит не все наборы аргументов и значений), которая приведена ниже.



## Решение:

print("x y z w F")

for x in range(2):

for y in range(2):

for z in range(2):

for w in range(2):

s = ((y <= x) == (w <= (not(z)))) and (w or x)

print(x, y, z, w, int(s))

# Задание 5.1.

На вход алгоритма подаётся натуральное число N. Алгоритм строит по нему новое число R следующим образом.

1) Строится двоичная запись числа N.

2) Если N кратно 5, то в конец двоичной записи числа дописываются три последние цифры числа. Иначе в конец двоичной записи числа дописывается остаток от деления n на 5, умноженный на 4, в двоичной записи.

Полученная таким образом запись является двоичной записью искомого числа R. Укажите минимальное число N, после обработки которого автомат получает число, большее 150.  
В ответе это число запишите в десятичной системе.

## Решение:

def R(n):

n2 = bin(n)[2:]

if n % 5 == 0:

n2 += n2[-3:]

else:

n2 += bin((n % 5) \* 4)[2:]

return int(n2, 2)

for n in range(1, 100):

if R(n) > 150:

print(n)

break

# Задание 5.2.

На вход алгоритма подаётся натуральное число N. Алгоритм строит по нему новое число R следующим образом.

1. Строится двоичная запись числа N.

2. Если сумма цифр десятичной записи числа чётна, то к этой записи справа дописывается 0, если нечётна – 1.

3-4. Пункт 2 повторяется для вновь полученных чисел ещё 2 раза.

Полученная таким образом запись является десятичной записью искомого числа R.

Укажите наименьшее возможное число R, большее 522, которое может получиться в результате работы алгоритма.

## Решение:

def sum\_digit(n):

summ = 0

for i in str(n):

summ += int(i)

return summ

R = []

for N in range(1, 10000):

s = bin(N)[2:]

for j in range(3):

if sum\_digit(int(s, 2)) % 2 == 0:

s += '0'

else:

s += '1'

if int(s, 2) > 522:

R.append(int(s, 2))

print(min(R))

# Задание 5.3.

На вход алгоритма подаётся натуральное число N. Алгоритм строит по нему новое число R следующим образом.

1. Строится двоичная запись числа N.

2. Далее эта запись обрабатывается по следующему правилу:

а) если сумма цифр в двоичной записи числа чётная, то к этой записи справа дописывается 10, а затем два левых разряда заменяются на 10;  
б) если сумма цифр в двоичной записи числа нечётная, то к этой записи справа дописывается 01, а затем два левых разряда заменяются на 11.

Полученная таким образом запись является двоичной записью искомого числа R.

Укажите наибольшее число N, после обработки которого с помощью этого алгоритма получается число R, не большее 390. В ответе это число запишите в десятичной системе счисления.

## Решение:

def R(N):

n2 = bin(N)[2:]

if n2.count("1") % 2 == 0:

n2 = "10" + n2[2:] + "10"

else:

n2 = "11" + n2[2:] + "01"

return int(n2, 2)

for i in range(1000, 1, -1):

if R(i) <= 390:

print(i)

break

# Задание 6.1.

Исполнитель Черепаха действует на плоскости с декартовой системой координат. В начальный момент Черепаха находится в начале координат, ее голова направлена вдоль положительного направления оси ординат, хвост опущен. При опущенном хвосте Черепаха оставляет на поле след в виде линии. В каждый конкретный момент известно положение исполнителя и направление его движения. У исполнителя существует две команды: Вперед n (где n – целое число), вызывающая передвижение Черепахи на n единиц в том направлении, куда указывает ее голова; Направо m (где m – целое число), вызывающая изменение направления движения на m градусов по часовой стрелке.

Запись Повтори k [Команда1 Команда2 … КомандаS] означает, что последовательность из S команд повторится k раз.

Черепахе был дан для исполнения следующий алгоритм:

Повтори 123 [Вперед 137 Направо 120].

Определите, сколько точек с целочисленными координатами будут находится внутри области, ограниченной линией, заданной данным алгоритмом. Точки на линии учитывать не следует.

## Решение:

from math import \*

cnt = 0

for x in range(0, 200):

for y in range(0, 200):

if x > 0 and y > tan(radians(30)) \* x and y < tan(radians(150)) \* x + 137:

cnt += 1

print(cnt)

# Задание 6.2.

Учебный алгоритм Баранка работает на плоскости с декартовой системой координат. В начальный момент Баранка находится в начале координат, а его голова направлена вдоль положительного направления оси абсцисс, хвост опущен. Если у Баранки опущен хвост, то он оставляет следы на поле в виде линий. В каждый конкретный момент известно положение исполнителя и направление его движения.

У исполнителя существует пять команд: Вперёд n (где n – целое число), вызывающая передвижение Баранки на n единиц в том направлении, куда указывает его голова. Направо m (где m –  целое число), вызывающая изменение направления движения на m градусов по часовой стрелке. Поднять хвост, вызывающая поднятие хвоста. Опустить хвост, которая опускает хвост исполнителя. Налево m (где m –  целое число), вызывающая изменение направления движения на m градусов против часовой стрелки.

Запись Повтори k [Команда1 Команда2 … КомандаS] означает, что последовательность из S команд повторится k раз. Баранке был дан для исполнения следующий алгоритм:

Повтори 2 раза [Вперед 6 Направо 90 Вперед 9 Направо 90]

Поднять хвост

Вперед 5 Направо 60

Опустить хвост

Повтори 7 раз [Направо 90 Вперед 4 Повтори 2 раза [Налево 90 Вперед 4]]

Налево 40 Повтори 3 раза [Вперед 9 Направо 120]

Определите количество целых точек, которое находится в области пересечения фигур. Точки на границах фигурах учитывать не нужно.

## Решение:

from turtle import \*

right(90)

pen()

for i in range(2):

forward(6 \* 20)

right(90)

forward(9 \* 20)

right(90)

penup()

forward(5 \* 20)

right(60)

pendown()

for i in range(7):

right(90)

forward(4 \* 20)

for j in range(2):

left(90)

forward(4 \* 20)

left(40)

for i in range(3):

forward(9 \* 20)

right(120)

penup()

for x in range(0,15):

for y in range(0, 15):

setpos(x \* 20,-y \* 20)

dot(2)

done()

# Задание 8.1.

Вам дан набор, состоящий из букв L, O, C, K. Из данного набора составляют кодовые пароли длиной семь символов, самое главное — символы могут повторяться. Ваша задача — определить количество кодов, в которых буква O встречается хотя бы два раза. В ответ запишите целое число – количество таких кодов.

## Решение:

from itertools import \*

cnt = 0

for i in product("LOCK", repeat = 7):

if i.count("O") >= 2:

cnt += 1

print(cnt)

# Задание 8.2.

Какое количество 8-буквенных слов вы можете составить из букв слова «ИНТЕГРАЛ», при условии, что каждую букву можно использовать ровно 1 раз, слово начинается на букву Р и заканчивается буквой А. Словом считается любая последовательность букв вне зависимости имеет или нет данный набор букв смысловое содержание.

## Решение:

from itertools import permutations

cnt = 0

for i in permutations('ИНТЕГРАЛ', r = 8):

if i[0] == 'Р' and i[-1] == 'А':

cnt += 1

print(cnt)

# Задание 8.3.

Вам дан набор, состоящий из букв Ф, А, Й, Л. Из данного набора составляют слова длиной четыре символа, а самое главное — символы могут повторяться. Все слова, которые возможно составить, расположили в алфавитном порядке, а ваша задача — найти номер первого слова, в котором нет букв А и Л. В ответ запишите целое число — позиционный номер данного слова.

Пример списка слов:

1. АААА

2. АААЙ

3. АААЛ

4. АААФ

5. ААЙА

6. ААЙЙ

7. ААЙЛ

…

## Решение:

from itertools import \*

cnt = 1

for i in product("АЙЛФ", repeat = 4):

if "А" not in i and "Л" not in i:

print(cnt)

break

cnt += 1

# Задание 9.1.

В файле электронной таблицы в каждой строке содержатся шесть неотрицательных целых чисел. Определите количество строк таблицы, для которых выполнены следующие условия:

– в строке есть ровно 1 пара совпадающих чисел;

– сумма всех чисел в строке больше произведения повторяющихся чисел.

В ответ запишите одно число — количество подходящих строк.

## Решение:

f = open('92.txt')

cnt = 0

for x in f:

flag = 0

x = list(map(int, x.split()))

if len(list(set(x))) == 5:

for i in range(len(x)):

if x.count(x[i]) == 2:

flag = x[i]

if sum(x) > flag \* flag:

cnt += 1

print(cnt)

# Задание 9.2.

Откройте прикрепленный файл таблицы, которая содержит в каждой из строк по 5 натуральных чисел. Ваша задача состоит в том, чтобы посчитать количество таких строк, в которых все числа различны, а утроенная сумма наибольшего и наименьшего чисел не меньше удвоенной суммы остальных чисел.

В ответ нужно записать только одно число — количество подходящих строк.

## Решение:

f = open("9.txt")

cnt = 0

for i in f:

s = sorted(list(map(int, i.split())))

if len(set(s)) == len(s):

if (s[0] + s[-1]) \* 3 >= (sum(s[1:4])) \* 2:

cnt += 1

print(cnt)

# Задание 12.1.

Исполнитель Редактор получает на вход строку цифр и преобразовывает её. Редактор может выполнять две команды, в обеих командах v и w обозначают цепочки символов.

заменить (v, w)

нашлось (v)

Первая команда заменяет в строке первое слева вхождение цепочки v на цепочку w. Если цепочки v в строке нет, эта команда не изменяет строку. Вторая команда проверяет, встречается ли цепочка v в строке исполнителя Редактор.

Дана программа для исполнителя Редактор:

НАЧАЛО

ПОКА нашлось (>4) ИЛИ нашлось (>7) ИЛИ нашлось (>3)

ЕСЛИ нашлось (>4)

ТО заменить (>4, 37>)

КОНЕЦ ЕСЛИ

ЕСЛИ нашлось (>7)

ТО заменить (>7, 3>)

КОНЕЦ ЕСЛИ

ЕСЛИ нашлось (>3)

ТО заменить (>3, 437>4)

КОНЕЦ ЕСЛИ

КОНЕЦ ПОКА

КОНЕЦ

На вход приведённой выше программе поступает строка, начинающаяся с символа «>», а затем содержащая 97 цифр «4», 30 цифр «7» и n цифр «3», расположенных в произвольном порядке. Определите наименьшее значение n, при котором сумма числовых значений цифр строки, получившейся в результате выполнения программы, будет кратна 103.

## Решение:

for n in range (1, 1000):

s = '>' + '4' \* 97 + '7' \* 30 + '3' \* n

while '>4' in s or '>7' in s or '>3' in s:

if '>4' in s:

s = s.replace('>4', '37>', 1)

if '>7' in s:

s = s.replace('>7', '3>', 1)

if '>3' in s:

s = s.replace('>3', '437>4', 1)

sum\_s = 0

for i in range (len(s)):

if s[i] != '>':

sum\_s += int(s[i])

if sum\_s % 103 == 0:

print(n)

break

# Задание 12.2.

Исполнитель «Редактор» получает на вход строку цифр и преобразовывает её. «Редактор» может выполнять две команды, в обеих командах v и w обозначают цепочки цифр.

заменить (v, w)

нашлось (v)

Первая команда заменяет в строке первое слева вхождение цепочки v на цепочку w. Если цепочки v в строке нет, эта команда не изменяет строку. Вторая команда проверяет, встречается ли цепочка v в строке исполнителя Редактор.

Дана программа для исполнителя «Редактор»:

НАЧАЛО

ПОКА нашлось (4444) ИЛИ нашлось (333)

ПОКА нашлось (4444)

заменить (4444, 3)

КОНЕЦ ПОКА

ПОКА нашлось (333)

заменить (333, 4)

КОНЕЦ ПОКА

КОНЕЦ ПОКА

КОНЕЦ

Дана строка, состоящая из 30 цифр, причем первые одиннадцать цифр – тройки, а остальные – четвёрки. Какая строка получится в результате применения программы к данной строке?

## Решение:

s = '3' \* 11 + '4' \* (30 - 11)

while '4444' in s or '333' in s:

while '4444' in s:

s = s.replace('4444', '3', 1)

while '333' in s:

s = s.replace('333', '4', 1)

print(s)

# Задание 14.1.

Сколько единиц в двоичной записи числа 6413 + 326 - 162?

## Решение:

s = bin(64 \*\* 13 + 32 \*\* 6 - 16 \*\* 2)[2:]

print(s.count("1"))

# Задание 14.2.

Значение арифметического выражения: 1024789 + 256678 - 64567 записали в системе счисления с основанием 5. Сколько цифр «4» содержится в этой записи?

## Решение:

s = 1024 \*\* 789 + 256 \*\* 678 - 64 \*\* 567

cnt = 0

while s > 0:

if s % 5 == 4:

cnt += 1

s //= 5

print(cnt)

# Задание 14.3.

Операнды арифметического выражения записаны в системе счисления с основанием 17:

18xAE17 + 733xC17

В записи чисел переменной x обозначена неизвестная цифра из алфавита 17-ричной системы счисления. Определите наименьшее значение x, при котором значение данного арифметического выражения кратно 13. Для найденного значения x вычислите частное от деления значения арифметического выражения на 13 и укажите его в ответе в десятичной системе счисления. Основание системы счисления в ответе указывать не нужно.

## Решение:

alph = list(range(10))

alph += ["a", "b", "c", "d", "e", "f", "g"]

for x in alph:

s = int(f"18{x}AE", 17) + int(f"733{x}C", 17)

if s % 13 == 0:

print(s // 13)

break

# Задание 15.1.

Выражение ДЕЛ(n, m) обозначает следующее: «натуральное число n делится без остатка на натуральное число m». Для какого наибольшего натурального числа А формула

¬ДЕЛ(x, A) → (¬ДЕЛ(x, 18) ∨ ¬ДЕЛ(x, 42))

тождественно истинна (то есть принимает значение 1 при любом натуральном значении переменной х)?

## Решение:

### Способ 1.

def Del(n, m):

return n % m == 0

def f(x, A):

return (not(Del(x, A))) <= ((not(Del(x, 18))) or (not(Del(x, 42))))

for A in range(1, 1000):

flag = True

for x in range(1, 1000):

if not f(x, A):

flag = False

break

if flag:

print(A)

### Способ 2.

def Del(n, m):

return n % m == 0

def f(x, A):

return (not(Del(x, A))) <= ((not(Del(x, 18))) or (not(Del(x, 42))))

for A in range(1, 1000):

for x in range(1, 1000):

if not f(x, A):

break

else:

print(A)

### Способ 3.

def Del(n, m):

return n % m == 0

def f(x, A):

return (not(Del(x, A))) <= ((not(Del(x, 18))) or (not(Del(x, 42))))

for A in range(1, 1000):

if all(f(x, A) for x in range(1, 1000)):

print(A)

# Задание 15.2.

Выражение n & m означает, что между числами производится побитовая конъюнкция.

Для какого наименьшего натурального значения А формула

(x & 34 ≠ 0) → ((x & 41 = 0) → (x & A ≠ 0))

тождественно истинна (то есть принимает значение 1 при любом натуральном значении переменной х)?

## Решение:

### Способ 1.

def f(x, A):

return (x & 34 != 0) <= ((x & 41 == 0) <= (x & A != 0))

for A in range(1, 1000):

if all(f(x, A) for x in range(1, 1000)):

print(A)

break

### Способ 2.

def f(x, A):

return (x & 34 != 0) <= ((x & 41 == 0) <= (x & A != 0))

for A in range(1, 1000):

for x in range(1, 1000):

if not f(x, A):

break

else:

print(A)

break

### Способ 3.

def f(x, A):

return (x & 34 != 0) <= ((x & 41 == 0) <= (x & A != 0))

for A in range(1, 1000):

flag = True

for x in range(1, 1000):

if not f(x, A):

flag = False

break

if flag:

print(A)

break

# Задание 15.3.

Для какого наименьшего натурального значения A выражение ((y + 5x ≠ 31) ∨ (A > x - 2)) ∧ (A < y + 37) тождественно истинно для любых положительных и целых x и y? В ответ запишите целое число – значение A.

## Решение:

### Способ 1.

def f(x, y, A):

return ((y + 5 \* x != 31) or (A > x - 2)) and (A < y + 37)

for A in range(1, 1000):

flag = True

for x in range(1, 1000):

for y in range(1, 1000):

if f(x, y, A) == False:

flag = False

break

if flag == False:

break

if flag == True:

print(A)

break

### Способ 2.

def f(x, y, A):

return ((y + 5 \* x != 31) or (A > x - 2)) and (A < y + 37)

for A in range(1, 1000):

if all(f(x, y, A) for x in range(1, 1000) for y in range(1, 1000)):

print(A)

break

# Задание 15.4.

Заданы три отрезка R = [12, 31], Q = [6, 15] и P = [17, 23], лежащие на числовой прямой. Также существует отрезок А. Он таков, что формула ((x ∈ А) ∨ (x ∈ P)) ∨ ((x ∈ Q) → (x ∈ R)) истинна, причём переменная x может принимать любые значения. Какую наименьшую длину может принимать отрезок А? Определите и запишите в ответ целое число.

## Решение:

R = list(range(12, 32))

Q = list(range(6, 16))

P = list(range(17, 24))

A = []

for x in range(1000):

if (((x in A) or (x in P)) or ((x in Q) <= (x in R))) == False:

A.append(x)

print(A)

# Задание 15.5.

Заданы два отрезка Q = [8, 17] и P = [3, 11], лежащие на числовой прямой. Также существует отрезок А. Он таков, что формула ((x ∈ А) → (x ∈ P)) ∨ (x ∈ Q) истинна, причём переменная x может принимать любые значения. Какую наибольшую длину может принимать отрезок А? Определите и запишите в ответ целое число.

## Решение:

Q = list(range(8, 18))

P = list(range(3, 12))

A = list(range(1, 1000))

for x in range(1000):

if (((x in A) <= (x in P)) or (x in Q)) == False:

A.remove(x)

print(A)

# Задание 15.6.

Обозначим через Del(n, m) утверждение «натуральное число n делится нацело на натуральное число m». Для какого наибольшего натурального числа A формула:

((x + 40 < A) ∨ (x + A < 40)) → (Del(x, A))

тождественно истинна (то есть принимает значение 1 при любом натуральном значении переменной x)?

## Решение:

### Способ 1.

def Del(n, m):

return n % m == 0

def f(x,A):

return (((x + 40 < A) or (x + A < 40)) <= (Del(x, A)))

for A in range(1, 1000):

if all(f(x, A) for x in range(1, 1000)):

print(A)

### Способ 2.

def Del(n, m):

return n % m == 0

def f(x, A):

return (((x + 40 < A) or (x + A < 40)) <= (Del(x, A)))

for A in range(1, 1000):

for x in range(1, 1000):

if not f(x, A):

break

else:

print(A)

### Способ 3.

def Del(n, m):

return n % m == 0

def f(x, A):

return (((x + 40 < A) or (x + A < 40)) <= (Del(x, A)))

for A in range(1, 1000):

flag = True

for x in range(1, 1000):

if not f(x, A):

flag = False

break

if flag:

print(A)

# Задание 16.1.

Тело функции F(n) задано следующими строчками:

F(n) = n, если n >= 2023,

F(n) = n / 3 + F(n + 2), если n < 2023.

Чему равно значение выражения F(2015) - F(2018)?

Примечание: Оператор “/” означает целочисленное деление.

## Решение:

def F(n):

if n >= 2023:

return n

return n // 3 + F(n + 2)

print(F(2015) - F(2018))

# Задание 16.2.

Функции F(n) и G(n) заданы следующими строчками:

F(1) = G(1) = n,

F(2) = G(2) = n \* n,

F(n) = 5 \* F(n - 1) - G(n - 2) + 8 \* n - 15, при n > 2,

G(n) = 2 \* F(n - 2) + 3 \* G(n - 1) - n \* 2, при n > 2.

В ответ запишите сумму цифр значения, возвращаемого функцией G(n), если в неё передать аргумент n = 53.

## Решение:

from functools import lru\_cache

def sum\_digits(n):

s = 0

for i in str(n):

s += int(i)

return s

@lru\_cache

def F(n):

if n == 1:

return n

if n == 2:

return n \* n

if n > 2:

return 5 \* F(n - 1) - G(n - 2) + 8 \* n - 15

@lru\_cache

def G(n):

if n == 1:

return n

if n == 2:

return n \* n

if n > 2:

return 2 \* F(n - 2) + 3 \* G(n - 1) - n \* 2

print(sum\_digits(G(53)))

# Задание 16.3.

Тело функции F(n) задано следующими строчками:

F(n) = n, при n <= 6,

F(n) = n + F(n // 3 + 1), при n, кратном трём, и если n > 6,

F(n) = n + F(n + 3), при n, не кратном трём, и если n > 6.

Назовите минимальное значение n, для которого F(n) определено и больше 1000.

## Решение:

def F(n):

if n <= 6:

return n

if n % 3 == 0 and n > 6:

return n + F(n // 3 + 1)

if n % 3 != 0 and n > 6:

return -100000000

for n in range(10000):

if F(n) > 1000:

print(n)

break

# Задание 17.1.

В файле содержится последовательность целых чисел из диапазона [-10000, 10000]. Определите количество пар элементов, из которых хотя бы одно число – это квадрат натурального числа, а также максимальную сумму элементов таких пар. В качестве пары рассматривается два элемента, идущие друг за другом. В ответ запишите целые числа без пробела: сначала количество, затем максимальное значение.

## Решение:

f = open("17.1.txt")

a = []

for s in f:

a.append(int(s))

count\_pari = 0

maxValue = -100000

for i in range(len(a)-1):

if (a[i] >= 0 and (a[i]\*\*0.5) == int(a[i]\*\*0.5)) or (a[i+1] >= 0 and (a[i+1]\*\*0.5) == int(a[i+1]\*\*0.5)):

count\_pari += 1

if a[i] + a[i+1] > maxValue:

maxValue = a[i] + a[i+1]

print(count\_pari, maxValue)

# Задание 17.2.

В файле содержится последовательность целых чисел из диапазона [-10000, 10000]. Определите количество троек элементов, в которых все три элемента отрицательны, а также минимальную сумму элементов таких троек. В качестве тройки рассматривается три элемента, идущие друг за другом. В ответ запишите целые числа через пробел: сначала количество, затем минимальное значение суммы.

## Решение:

f = open("17.7.txt")

a = list(map(int, f.read().split()))

c = 0

summa = 30001

for i in range(len(a)-2):

trio = a[i:i+3]

if trio[0] < 0 and trio[1] < 0 and trio[2] < 0:

c += 1

summa = min(summa, sum(trio))

print(c, summa)

# Задание 17.3.

B файле содержится последовательность натуральных чисел. Элементы последовательности могут принимать целые значения от 1 до 10 000 включительно. Определите количество пар последовательности, в которых только одно число двузначное, а сумма пары чисел кратна минимальному двузначному элементу последовательности.

B ответе запишите без пробела сначала количество найденных пар, затем максимальную из сумм элементов таких пар. В данной задаче под парой подразумевается два идущих подряд элемента последовательности.

## Решение:

f = open('17.txt')

a = []

count = 0

twodigit = []

for x in f:

a.append(int(x))

if 10 <= int(x) <= 99:

twodigit.append(int(x))

max\_sum = a[0]

for i in range(len(a)-1):

if ((a[i] in twodigit and not(a[i+1] in twodigit)) or (a[i+1] in twodigit and not(a[i] in twodigit))) and ((a[i] + a[i+1]) % min(twodigit) == 0):

count += 1

max\_sum = max(a[i]+a[i+1], max\_sum)

print(count, max\_sum)

# Задание 17.4.

В файле содержится последовательность натуральных чисел. Элементы последовательности могут принимать целые значения от 1 до 100 000 включительно. Определите количество пар последовательности, в которых остаток от деления хотя бы одного из элементов на 123 равен минимальному элементу последовательности. В ответе запишите количество найденных пар, затем максимальную из сумм элементов таких пар. В данной задаче под парой подразумевается два идущих подряд элемента последовательности. В ответ запишите числа без пробела.

## Решение:

f = open("17.3.txt")

s = [int(x) for x in f]

res = []

min\_elem = min(s)

for i in range(len(s) - 1):

if s[i] % 123 == min\_elem or s[i + 1] % 123 == min\_elem:

res.append(s[i] + s[i + 1])

print(len(res), max(res))

# Задания 19-21.1.

Полина и Вероника играют в игру, правила которой заключаются в следующем:

1) Перед игроками лежит одна куча камней. Игроки ходят по очереди, первый ход делает Полина.

2) За один ход игрок может добавить в кучу 1 камень или увеличить количество камней в куче в 3 раза. Для этого у каждого игрока есть неограниченный запас камней.

3) Игра завершается в тот момент, когда количество камней в куче стало не менее 59.

4) Победителем является тот игрок, который сделал последний ход и получил сумму камней в куче, равную 59 или больше.

Пример: из 10 камней за один ход можно получить кучу из 11 или 30 камней.

В начальный момент в куче было S камней в диапазоне [1, 58].

Примечание. Игрок обладает выигрышной стратегией, если у него есть возможность набрать нужную сумму камней в куче для выигрыша при любых ходах противника.

19. Мы знаем, что Вероника выиграла своим первым ходом после неудачного хода Полины. Определите и запишите в ответ минимальное начальное количество камней S в куче, при котором возможна вышеописанная ситуация.

20. Найдите значение N, при котором у Полины есть выигрышная стратегия, при условии, что параллельно выполняются два условия:

— Полина не может выиграть за один ход;

— Полина может выиграть своим вторым ходом независимо от того, как будет ходить Вероника.

21. Найдите минимальное значение N, при котором параллельно выполняются два условия:

— у Вероники есть стратегия, позволяющая ей выиграть первым или вторым ходом при любой игре Полины;

— у Вероники нет стратегии, которая позволит ей гарантированно выиграть первым ходом.

## Решение:

### Способ 1.

def win1(s):

return (s + 1) >= 59 or (s \* 3) >= 59

def loss1(s):

return (not(win1(s))) and win1(s + 1) and win1(s \* 3)

def win2(s):

return loss1(s + 1) or loss1(s \* 3)

def loss12(s):

return (win1(s \* 3) and win2(s + 1)) or (win1(s + 1) and win2(s \* 3))

s19, s20, s21 = [], [], []

for s in range(1, 59):

if win1(s + 1) or win1(s \* 3):

s19.append(s)

if win2(s):

s20.append(s)

if loss12(s):

s21.append(s)

print(min(s19))

print(s20)

print(min(s21))

### Способ 2.

from functools import lru\_cache

def moves(h):

return h + 1, h \* 3

@lru\_cache

def f(h):

if h >= 59:

return 'Win'

elif any(f(x) == 'Win' for x in moves(h)):

return 'P1'

elif all(f(x) == 'P1' for x in moves(h)):

return 'V1'

elif any(f(x) == 'V1' for x in moves(h)):

return 'P2'

elif all((f(x) == 'P2' or f(x) == 'P1') for x in moves(h)):

return 'V2'

s19, s20, s21 = [], [], []

for i in range(1, 59):

s = i

if any(f(x) == 'P1' for x in moves(s)):

s19.append(s)

if f(s) == 'P2':

s20.append(s)

if f(s) == 'V2':

s21.append(s)

print(min(s19))

print(s20)

print(min(s21))

# Задания 19-21.2.

Полина и Вероника играют в игру, правила которой заключаются в следующем:

1) Перед игроками лежат две кучи камней. Игроки ходят по очереди, первый ход делает Полина.

2) За один ход игрок может добавить в одну кучу 2 камня или увеличить количество камней в куче в 2 раза. Для этого у каждого игрока есть неограниченный запас камней.

3) Игра завершается в тот момент, когда сумма камней двух куч стала не менее 90.

4) Победителем является тот игрок, который сделал последний ход и получил сумму, равную 90 или больше.

Пример: имея кучи (10, 5) камней, за один ход игрок может получить: (12, 5), (20, 5), (10, 7), (10,10).

В начальный момент в кучах было (6, S) камней, где S в диапазоне [1, 83].

19. Мы знаем, что Вероника выиграла своим первым ходом после первого неудачного хода Полины. Определите минимальное начальное количество камней S в куче, при котором возможна такая ситуация.

20. Найдите два значения S, при которых у Полины есть выигрышная стратегия, при условии, что параллельно выполняются два условия:

– Полина не может выиграть за один ход;

– Полина может выиграть своим вторым ходом независимо от того, как будет ходить Вероника.

Найденные значения запишите в ответ в порядке возрастания без разделителей.

Если значений больше двух, в ответ запишите сначала наименьшее значение, за ним наибольшее без разделителей.

21. Найдите значение S, при котором параллельно выполняются два условия:

– у Вероники есть стратегия, позволяющая ей выиграть первым или вторым ходом при любой игре Полины;

– у Вероники нет стратегии, которая позволит ей гарантированно выиграть первым ходом.

## Решение:

### Способ 1.

def Win1(x, S):

return (x + S < 90) and ((x + 2 + S >= 90) or (x \* 2 + S >= 90) or (S \* 2 + x >= 90))

def Loss1(x, S):

return ((not(Win1(x, S))) and Win1(x + 2, S) and Win1(x, S + 2) and Win1(x \* 2, S) and Win1(x, S \* 2))

def Win2(x, S):

return (Loss1(x + 2, S) or Loss1(x, S + 2) or Loss1(x \* 2, S) or Loss1(x, S \* 2))

def Loss12(x, S):

return ((Win1(x + 2, S) or Win2(x + 2, S)) and (Win1(x, S + 2) or Win2(x, S + 2)) and (Win1(x \* 2, S) or Win2(x \* 2, S)) and (Win1(x, S \* 2) or Win2(x, S \* 2)) and (Win2(x + 2, S) or Win2(x, S + 2) or Win2(x \* 2, S) or Win2(x, S \* 2)))

x = 6

s19, s20, s21 = [], [], []

for S in range(1, 84):

if Win1(x + 2, S) or Win1(x, S + 2) or Win1(x \* 2, S) or Win1(x, S \* 2):

s19.append(S)

if Win2(x, S):

s20.append(S)

if Loss12(x, S):

s21.append(S)

print(min(s19))

print(min(s20), max(s20))

print(s21)

### Способ 2.

from functools import lru\_cache

def moves(h):

a, b = h

return (a + 2, b), (a \* 2, b), (a, b + 2), (a, b \* 2)

@lru\_cache(None)

def f(h):

if sum(h) >= 90:

return 'Win'

elif any(f(x) == 'Win' for x in moves(h)):

return 'P1'

elif all(f(x) == 'P1' for x in moves(h)):

return 'V1'

elif any(f(x) == 'V1' for x in moves(h)):

return 'P2'

elif all((f(x) == 'P2' or f(x) == 'P1') for x in moves(h)):

return 'V2'

s19, s20, s21 = [], [], []

for i in range(1, 84):

h = 6, i

if any(f(x) == 'P1' for x in moves(h)):

s19.append(i)

if f(h) == 'P2':

s20.append(i)

if f(h) == 'V2':

s21.append(i)

print(min(s19))

print(min(s20), max(s20))

print(min(s21))

# Задание 22.

В файле содержится информация о совокупности N вычислительных процессов, которые могут выполняться параллельно или последовательно. Будем говорить, что процесс B зависит от процесса A, если для выполнения процесса B необходимы результаты выполнения процесса A. Если процесс B зависит от процесса A, то процесс B может начать выполнение сразу же после завершения процесса A. Количество одновременно выполняемых процессов может быть любым, длительность процесса не зависит от других параллельно выполняемых процессов. В таблице представлены идентификатор (ID) каждого процесса, его длительность и ID поставщиков данных для зависимых процессов. Определите, через какое время после запуска первых процессов будет завершено 50 процессов. В ответе укажите целое число – время в мс.

## Решение:

f = open("22.1.txt")

process = [0] \* 100

f.readline()

for s in f:

data = s.split()

if data[2] == "0":

process[int(data[0])] = int(data[1])

elif len(data[2].split(";")) == 1:

process[int(data[0])] = int(data[1]) + process[int(data[2])]

else:

data[2] = list(map(int, data[2].split(";")))

process[int(data[0])] = int(data[1]) + max(process[data[2][0]], process[data[2][1]])

while 0 in process:

process.remove(0)

process.sort()

print(max(process[:50]))

# Задание 23.1.

У исполнителя есть три команды:

– прибавить 2  
– сделать чётным  
– сделать нечётным

Сделать чётным означает умножить число на два, сделать нечётным – умножить на два и прибавить единицу. Определите количество программ исполнителя, которые преобразуют число 2 в 45. В ответ запишите целое число – количество программ.

## Решение:

def task23(start, end):

if start > end:

return 0

elif start == end:

return 1

else:

return task23(start+2, end) + task23(start\*2, end) + task23((start\*2)+1, end)

print(task23(2, 45))

# Задание 23.2.

У исполнителя есть три команды:

– вычесть 1

– разделить нацело на 2

– разделить нацело на 3

Определите количество программ исполнителя, которые преобразуют число 131 в 3, при условии, что траектория выполнения программы содержит число 41. В ответ запишите целое число – количество программ.

## Решение:

def task23(start, end):

if start < end:

return 0

if start == end:

return 1

else:

return task23(start - 1, end) + task23(start // 2, end) + task23(start // 3, end)

print(task23(131, 41) \* task23(41, 3))

# Задание 23.3.

У исполнителя есть три команды:

– прибавить 1

– умножить на 2

– умножить на 3

Определите количество программ исполнителя, которые преобразуют число 1 в 98, при условии, что траектория выполнения программы содержит число 14 и не содержит число 8. В ответ запишите целое число – количество программ.

## Решение:

def task23(start, end):

if start > end:

return 0

elif start == end:

return 1

elif start == 8:

return 0

else:

return task23(start+1, end) + task23(start\*2, end) + task23(start\*3, end)

print(task23(1, 14) \* task23(14, 98))