

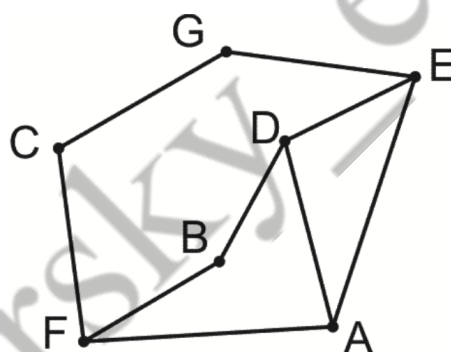
вебиум



Разбор демо-версии 2023 (t.me/kaspersky_ege)

№1 На рисунке схема дорог Н-ского района изображена в виде графа, в таблице содержатся сведения о протяжённости каждой из этих дорог (в километрах).

		Номер пункта						
		1	2	3	4	5	6	7
Номер пункта	1		39	3				
	2	39			8	5		
	3	3					2	
	4		8					53
	5		5				21	30
	6			2		21		13
	7				53	30	13	

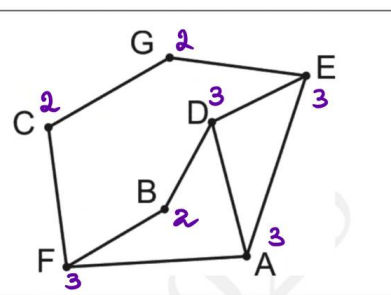


Так как таблицу и схему рисовали независимо друг от друга, то нумерация населённых пунктов в таблице никак не связана с буквенными обозначениями на графе. Определите, какова сумма протяжённости дорог из пункта D в пункт B и из пункта F в пункт A.

Ответ: 58

▼ Пояснения:

		Номер пункта						
		1	2	3	4	5	6	7
Номер пункта	1		39	3				
	2	39			8	5		
	3	3					2	
	4		8					53
	5		5				21	30
	6			2		21		13
	7				53	30	13	



Начнем решение задачи с того, что распишем на графе для каждой вершины ее степень (количество соседних вершин). На картинке они написаны фиолетовым.

Далее будем ставить в соответствие вершины и пункты в таблице:

1. Заметим, что имеется единственная вершина степени 2 – вершина B, которая соединена только с вершинами степени 3. Найдём пункт в таблице, соответствующий этой вершине, это пункт 4.
2. Далее сразу отметим, что два возможных пункта для F/D это 2 и 7, исходя из того, что это соседи вершины B.

3. Заметим, что вершина А степени 3 соединена только с вершинами степени 3. Найдём соответствующий пункт в таблице, это пункт 5.
 4. Вершина А соединена с F, но не соединена с D, соответственно возвращаемся к п.2 и получаем, что пункту 2 соответствует вершина F, а пункту 7 вершина D.
 5. Вершина F соединена с вершиной С степени 2, по таблице получаем, что пункту 1 соответствует вершина С.
 6. Оставшийся пункт 3 соответствует вершине степени 2 — G, и пункт 6 соответствует вершине степени 3 — E.
- Дорога из D в B — 53, а дорога из F в A — 5. Сумма 58

№2 Миша заполнял таблицу истинности логической функции F

$$\neg(y \rightarrow x) \vee (z \rightarrow w) \vee \neg z,$$

но успел заполнить лишь фрагмент из трёх различных её строк, даже не указав, какому столбцу таблицы соответствует каждая из переменных w, x, y, z.

				<i>F</i>
	0			0
0	1			0
1			0	0

Определите, какому столбцу таблицы соответствует каждая из переменных w, x, y, z.

В ответе напишите буквы w, x, y, z в том порядке, в котором идут соответствующие им столбцы (сначала буква, соответствующая первому столбцу; затем буква, соответствующая второму столбцу, и т.д.). Буквы в ответе пишите подряд, никаких разделителей между буквами ставить не нужно.

Пример. Функция F задана выражением $\neg x \vee y$, зависящим от двух переменных, а фрагмент таблицы имеет следующий вид.

		<i>F</i>
0	1	0

В этом случае первому столбцу соответствует переменная y, а второму столбцу – переменная x. В ответе следует написать: yx.

Ответ: uxzw

▼ Пояснения:

```
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):
                if (not(y <= x) or (z <= w) or not z) == 0:
                    print(x, y, z, w, 0)
```

Получаем следующую таблицу истинности:

x y z w F
0 0 1 0 0
1 0 1 0 0
1 1 1 0 0

Строку с тремя единицами можно соотнести только с последней строкой таблицы, поэтому последняя переменная - w. Столбец второй с двумя единицами и одним нулем подходит только под переменную x. Первый столбец тогда будет составлять переменную y, а третий z. Получаем ответ: uxzw.

№3 В файле приведён фрагмент базы данных «Продукты» о поставках товаров в магазины районов города. База данных состоит из трёх таблиц.

Таблица «Движение товаров» содержит записи о поставках товаров в магазины в течение первой декады июня 2021 г., а также информацию о проданных товарах. Поле Тип операции содержит значение Поступление или Продажа, а в соответствующее поле Количество упаковок внесена информация о том, сколько упаковок товара поступило в магазин или было продано в течение дня. Заголовок таблицы имеет следующий вид.

ID операции	Дата	ID магазина	Артикул	Тип операции	Количество упаковок	Цена
-------------	------	-------------	---------	--------------	---------------------	------

Таблица «Товар» содержит информацию об основных характеристиках каждого товара. Заголовок таблицы имеет следующий вид.

Артикул	Отдел	Наименование	Единица измерения	Количество в упаковке	Производитель
---------	-------	--------------	-------------------	-----------------------	---------------

Таблица «Магазин» содержит информацию о местонахождении магазинов. Заголовок таблицы имеет следующий вид.

ID магазина	Район	Адрес
-------------	-------	-------

На рисунке приведена схема указанной базы данных.



Используя информацию из приведённой базы данных, определите общий вес (в кг) крахмала картофельного, поступившего в магазины Заречного района за период с 1 по 8 июня включительно. В ответе запишите только число.

Ответ: 355

▼ Пояснения:

1. Нужно создать фильтр. Кликаем на любую ячейку шапки, далее переходим к созданию фильтра

	A	B	C	D	E	F	G	H	I	J
	ID операции	Дата	ID магазина	Артикул	Тип операции	Количество упаковок	Цена			
1										
2	1	01.06.2021	M1		4 Поступление	180	75			
3	2	01.06.2021	M1		4 Продажа	180	75			
4	3	01.06.2021	M1		5 Поступление	180	70			
5	4	01.06.2021	M1		5 Продажа	170	70			
6	5	01.06.2021	M1		6 Поступление	180	50			
7	6	01.06.2021	M1		6 Продажа	180	50			
8	7	01.06.2021	M1		9 Поступление	180	55			

2. 1) ждем на значок сортировки в ячейке "Тип операции"
- 2) Сбрасываем галочки со всех номеров
- 3) Отмечаем галочкой "Поступление"

Артикул	Тип операции	Количество упаковок	Цена
4	Поступление		
5	Поступление		
6	Поступление		
9	Поступление		
10	Поступление		
13	Поступление		
18	Поступление		
24	Поступление		
25	Поступление		
26	Поступление		
27	Поступление		
28	Поступление		
29	Поступление		
30	Поступление		
33	Поступление		
34	Поступление		
44	Поступление		
45	Поступление		
46	Поступление	180	330
47	Поступление	180	370
48	Поступление	180	180

Тип операции

Сортировка

A-Z По возрастанию Z-A По убыванию

По цвету: Нет

Фильтр

По цвету: Нет

Равно Поступление

И Или

Выберите

Поиск

(Выделить все)

Поступление

Продажа

Автоматическое применение

Применить фильтр Очистить фильтр

- С помощью ctrl+f на листе "Товар" находим "Крахмал картофельный" и запоминаем его артикул 42 и то, что в одной упаковке 0,5 кг
- С помощью ctrl+f на листе "Магазин" находим ID магазинов Заречного района (M3, M9, M11, M14)
- Аналогично 2 пункту делаем фильтр на листе "Движение товаров" по Артикулу и ID Магазина
- Получаем 4 строчки, суммируем количество упаковок(обязательно не диапазоном, а отдельно) и не забываем умножить эту сумму на 0,5, так как в одной упаковке 0,5 кг

СУММ fx =СУММ(F946:F1030;F1142;F1310)*0,5

	A	B	C	D	E	F	G	H	I	J	K	L
	ID операции	Дата	ID магазина	Артикул	Тип операции	Количество упаковок	Цена					
1												
946	945	03.06.2021	M11	42	Поступление	170	90		=СУММ(F946;F1030;F1142;F1310)*0,5			
1030	1029	03.06.2021	M14	42	Поступление	180	90					
1142	1141	03.06.2021	M3	42	Поступление	180	90					
1310	1309	03.06.2021	M9	42	Поступление	180	90					
2274												
2275												
2276												

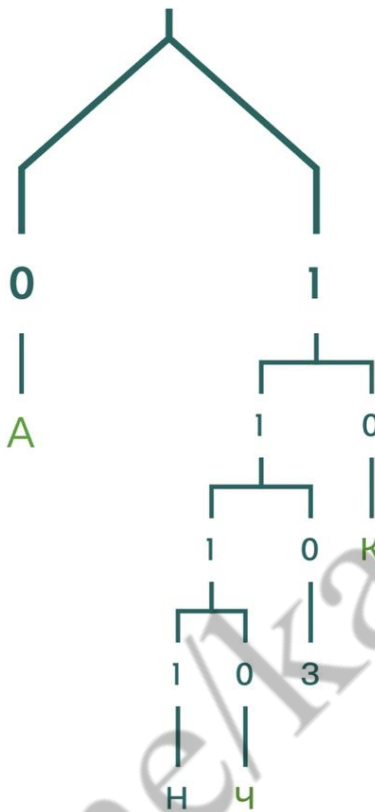
Получаем ответ: 355

№4 По каналу связи передаются сообщения, содержащие только буквы из набора: А, З, К, Н, Ч. Для передачи используется двоичный код, удовлетворяющий прямому условию Фано, согласно которому никакое кодовое слово не является началом другого кодового слова. Это условие обеспечивает возможность однозначной расшифровки закодированных сообщений. Кодовые слова для некоторых букв известны: Н – 1111, З – 110. Для трёх оставшихся букв А, К и Ч кодовые слова неизвестны. Какое количество двоичных знаков потребуется для кодирования слова КАЗАЧКА, если известно, что оно закодировано **минимально** возможным количеством двоичных знаков?

Ответ: 14

▼ Пояснения:

Корень



Подсчитаем количество различных букв в слове, букв А больше всего — их 3, букв К — 2, буква Ч и буква З — по одной. Изобразим в виде дерева коды, соответствующие буквам.

Проанализируем: чтобы длина кода была минимальной, для букв, которые повторяются наибольшее количество раз, код должен быть как можно короче.

Буква А повторяется целых 3 раза, поэтому попробуем для неё код 0 длины 1, для буквы К код 10 длины 2, для оставшейся буквы Ч, которая повторяется всего раз попробуем код 1110 длины 4. Посчитаем сумму длин кодов и получим 14.

Пробуя другие варианты распределения кодов по буквам длина слова получается всегда больше, чем 14.

$$К = 2 * 2 = 4$$

$$А = 3 * 1 = 3$$

$$З = 3$$

$$Ч = 1 * 4 = 4$$

Итого: 14

№5 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:

- если сумма цифр в двоичной записи числа чётная, то к этой записи справа дописывается 0, а затем два левых разряда заменяются на 10;
- если сумма цифр в двоичной записи числа нечётная, то к этой записи справа дописывается 1, а затем два левых разряда заменяются на 11.

Полученная таким образом запись является двоичной записью искомого числа R.

Например, для исходного числа $6_{10} = 110_2$ результатом является число

$1000_2 = 8_{10}$, а для исходного числа $4_{10} = 100_2$ результатом является число $1101_2 = 13_{10}$.

Укажите **минимальное** число N , после обработки которого с помощью этого алгоритма получается число R , большее 40. В ответе запишите это число в десятичной системе счисления.

Ответ: 16

▼ Пояснения:

Найдем изначальное N , т.к. $R > 40$, то изначальное N равняется 10100_2 , нужно рассмотреть вариант с нечетной суммой цифр: $10000_2 \rightarrow 110001_2$ при переводе в 10сс получим $49 > 40$, подходит $10000_2 = 16_{10}$, это ответ, потому что брать меньше мы не можем из за невозможности получить число большее 40, а больше смысла нет, нужно минимальное.

№6 Исполнитель Черепаха действует на плоскости с декартовой системой координат. В начальный момент Черепаха находится в начале координат, её 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:

Вперёд n (где n – целое число), вызывающая передвижение Черепахи на n единиц в том направлении, куда указывает её голова, и **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 7 [Вперёд 10 Направо 120].

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

Ответ: 38

▼ Пояснения:

Как заявил ФИПИ 6 задания направлены на анализ алгоритмов:

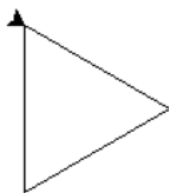
Задание 6 в 2023 году будет посвящено анализу алгоритма для конкретного исполнителя, определению возможных результатов работы простейших алгоритмов управления исполнителями и вычислительных алгоритмов

Так как это графический исполнитель, то достаточно просто можем смоделировать его действие с помощью библиотеки turtle:

```
import turtle
from turtle import *

count = 0 # заводим переменную для подсчёта выполненных циклов
left(90) # поворачиваем черепашку в сторону положительного направления оси ординат (0Y)
while count < 7: # из условия "Повтори 7"
    forward(10) # из условия "Вперед 10"
    right(120) # из условия "Направо 120"
    count += 1 # считаем кол-во пройденных циклов
done()
```

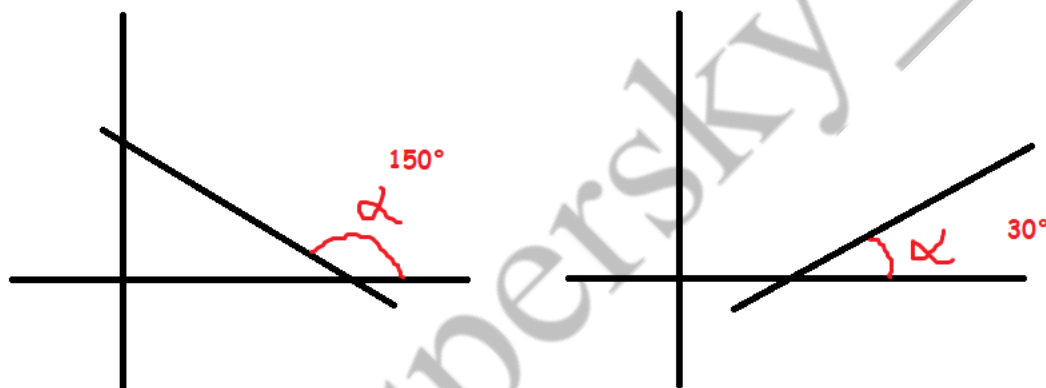
После выполнения кода выше мы получим рисунок траектории черепахи - это равносторонний треугольник



Область внутри данного треугольника ограничена 3 прямыми линиями:

- вертикальная слева задаётся уравнением $x = 0$
- прямые расположенные диагонально можно задать уравнениями вида $y = kx + b$

Коэффициенты k можно найти через $\operatorname{tg}\alpha$, где α - угол, образующийся между горизонтальной прямой и осью абсцисс (OX):

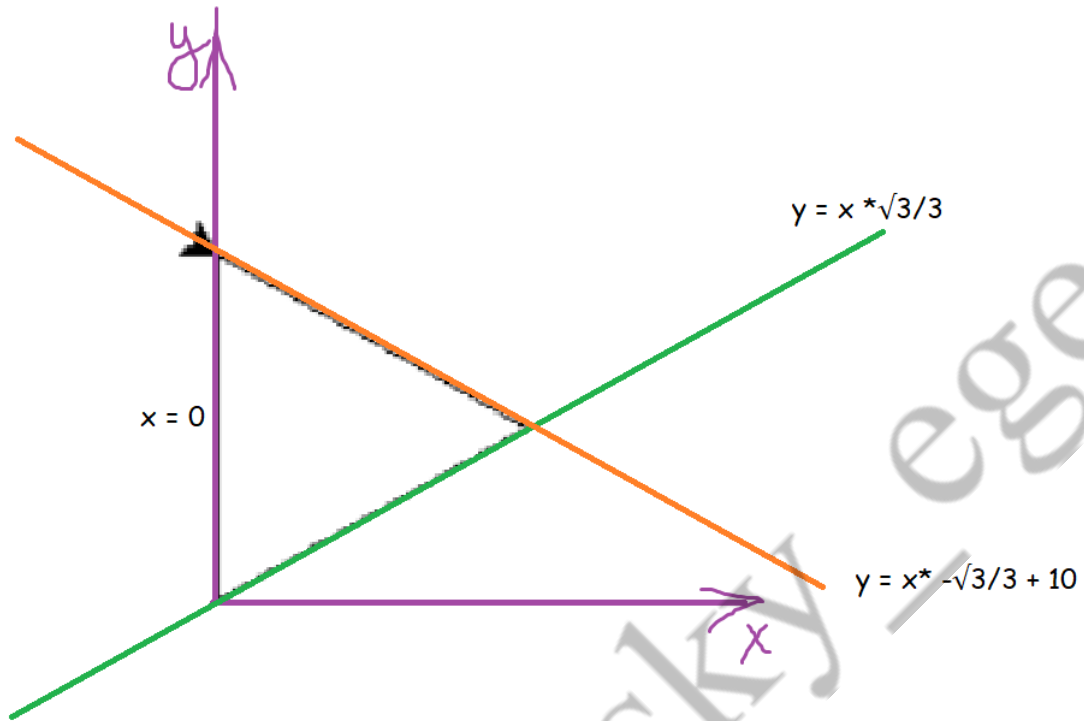


Так как треугольник равносторонний - все его углы равны 60° , значит верхняя и нижняя диагональные прямые образуют с осью абсцисс углы 150° и 30° соответственно, тангенсы этих углов равны $-\sqrt{3}/3$ и $\sqrt{3}/3$ соответственно

Параметр b для нижней диагональной прямой равна 0, для верхней - 10, так как из условия задачи черепаха ходит на 10 единиц вперёд

Получаем следующие формулы прямых:

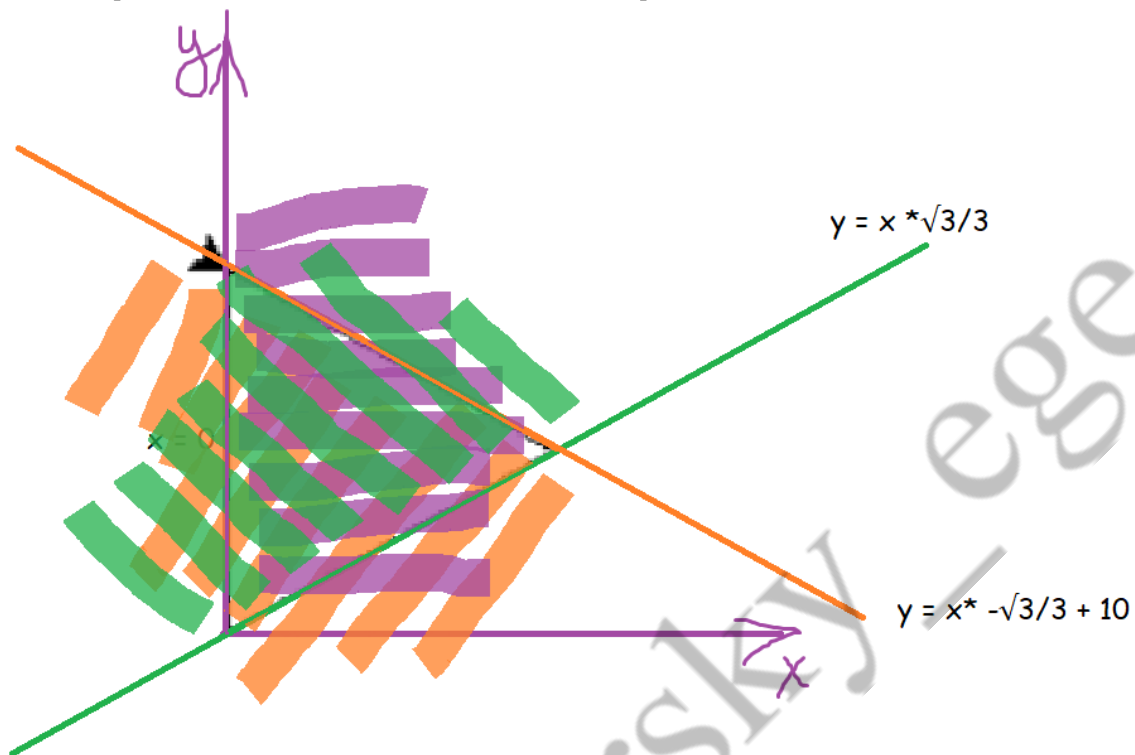
- $x = 0$
- $y = x * \sqrt{3}/3$
- $y = x * -\sqrt{3}/3 + 10$



Идея задачи: проверяем нахождение точки внутри области, ограниченной траекторией движения черепахи, по её координатам относительно 3-х прямых, на которых лежат стороны треугольника

Подходящие нам точки лежат правее вертикальной прямой, ниже верхней диагонали и выше нижней, а значит удовлетворяют следующим условиям:

- $x > 0$
- $y > x * \sqrt{3}/3$
- $y < x * -\sqrt{3}/3 + 10$



Проверим все точки с целочисленными координатами в квадрате 10 на 10, в который помещается треугольник из задачи, для этого напишем программу:

```
import math

count = 0 # заводим переменную для подсчёта точек, удовлетворяющих условию
for x in range(11): # перебираем x
    for y in range(11): # перебираем y
        if x > 0 and x * math.sqrt(3) / 3 < y < (10 + -1 * x * math.sqrt(3) / 3): # записываем ранее сформулированное условие
            count += 1 # если оно выполняется, то увеличиваем значение счётчика
print(count) # выводим найденное значение
```

№7 Музыкальный фрагмент был записан в формате моно, оцифрован и сохранён в виде файла без использования сжатия данных. Размер полученного файла – 28 Мбайт. Затем тот же музыкальный фрагмент был записан повторно в формате стерео (двухканальная запись) и оцифрован с разрешением в 3,5 раза выше и частотой дискретизации в 2 раза меньше, чем в первый раз. Сжатие данных не производилось. Укажите размер полученного при повторной записи файла в Мбайт. В ответе запишите только целое число, единицу измерения писать не нужно.

Ответ: 98

▼ Пояснения:

Пусть в первый раз:

$$1 * a * b * t = 28 \text{ Мб}$$

Где: a - разрешение, b - частота дискретизации, t - время записи

Тогда во второй раз:

$$2 * 3,5 * a * 1/2 b * t = y$$

y - объём который мы ищем

Составим и решим пропорцию:

$$y * 1 * a * b * t = 28 * 2 * 3,5 a * 1/2 b * t$$

Всё лишнее сокращается, остаётся: $28 \cdot 3,5$

Ответ: 98

№8 Определите количество пятизначных чисел, записанных в восьмеричной системе счисления, в записи которых только одна цифра 6, при этом никакая нечётная цифра не стоит рядом с цифрой 6.

Ответ: 2961

▼ Пояснения:

```
#8
#перевод в 8CC
def perevod(n):
    res = ""
    while n>0:
        res = str(n%8) + res
        n = n//8
    return res

nchet = "13579" #список нечетных чисел
cnt = 0 #счетчик для подходящих чисел

#цикл по числам, которые являются 5-значными в 8CC
for x in range(8**4, (8**5)-1):
    chislo = perevod(x)

    #проверяем, что в числе только одна цифра 6, сохраняем ее индекс
    if chislo.count("6") == 1:
        ind = chislo.find("6")
        #если цифра 6 первая в числе
        if ind == 0:
            if not(chislo[ind + 1] in nchet):
                cnt+=1
        #если цифра 6 последняя в числе
        elif ind == len(chislo) - 1:
            if not(chislo[ind - 1] in nchet):
                cnt+=1
        #если цифра 6 является ни первой, ни последней в числе
        else:
            if not((chislo[ind - 1] in nchet) or (chislo[ind + 1] in nchet)):
                cnt+=1

print(cnt)
```

№9 Откройте файл электронной таблицы, содержащей в каждой строке шесть натуральных чисел. Определите количество строк таблицы, содержащих числа, для которых выполнены оба условия:

– в строке только одно число повторяется дважды;

– среднее арифметическое неповторяющихся чисел строки не больше суммы повторяющихся чисел.

В ответе запишите только число.

Ответ: 2241

▼ Пояснения:

Решение экселем

```
lines = open('9.txt').readlines() # после переноса данных из excel в txt
# открываем этот файл и добавляем каждую строку
# в список
cnt = 0 # переменная для ответа

for string in lines: # проходим по строкам
    check = list(map(int, string.split())) # список со ВСЕМИ 6 числами из строки
    line = set(map(int, string.split())) # множество уникальных элементов с уникальными числами из строки
    if len(line) == 5: # если в строке содержится 5 разных чисел (т.е. лишь одно число встречается дважды)
        for x in line: # проходим по множеству с целью узнать, какое число повторяется дважды
            if check.count(x) == 2: # если число повторяется дважды
                if ((sum(check) - 2 * x) / 4) <= x + x: # проверяем второе условие: если ср. арифм неповторяющихся чисел больше суммы
                    # повторяющихся
                    cnt += 1 # увеличиваем количество подходящих строк на 1
```

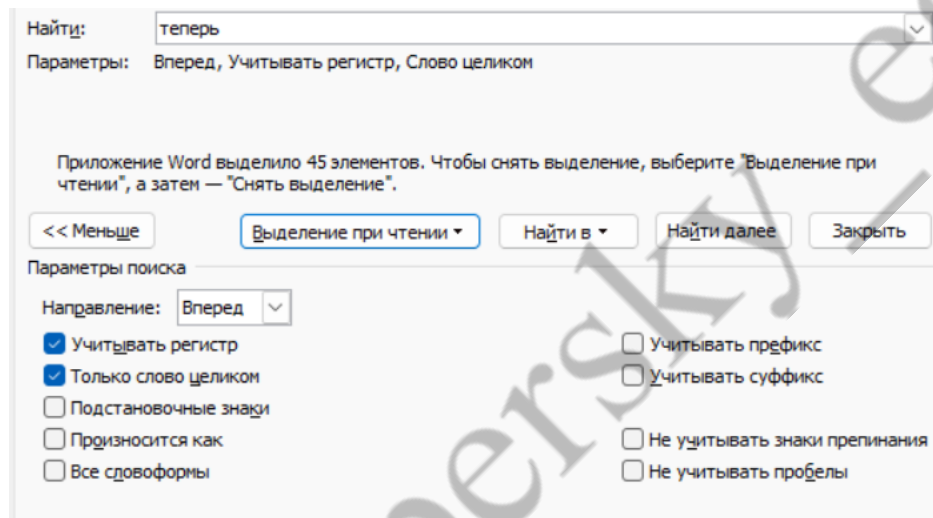
```
print(cnt) # выводим ответ
```

№10 Текст произведения Льва Николаевича Толстого «Севастопольские рассказы» представлен в виде файлов различных форматов. Откройте один из файлов и определите, сколько раз встречается в тексте отдельное слово «теперь» со строчной буквы. Другие формы этого слова учитывать не следует. В ответе запишите только число

Ответ: 45

▼ Пояснения:

Используем расширенный поиск, задаем следующие параметры: “Учитывать слово целиком”, “Учитывать регистр”. Получаем 45 подходящих вариантов слова “теперь”.



№11 При регистрации в компьютерной системе каждому объекту присваивается идентификатор, состоящий из 250 символов и содержащий только десятичные цифры и символы из 1650-символьного специального алфавита. В базе данных для хранения каждого идентификатора отведено одинаковое и минимально возможное целое число байт. При этом используется посимвольное кодирование идентификаторов, все символы кодируются одинаковым и минимально возможным количеством бит.

Определите объём памяти (в Кбайт), необходимый для хранения 65 536 идентификаторов. В ответе запишите только целое число – количество Кбайт

Ответ: 22016

▼ Пояснения:

Всего алфавит $1650+10$ символов, это 2^{11} , т.е. на 1 символ нам нужно 11 бит. Всего 250 символов, $250 \cdot 11 = 2750$ бит, делим на 8 = 344, мы узнали сколько байт нужно на один идентификатор. $344 \cdot 65536 / 1024 = 22016$ (умножаем на количество идентификаторов и делим, чтобы получить килобайты)

№12 Исполнитель Редактор получает на вход строку цифр и преобразовывает её.

Редактор может выполнять две команды, в обеих командах v и w обозначают цепочки цифр.

А) заменить (v, w).

Эта команда заменяет в строке первое слева вхождение цепочки v на цепочку w . Например, выполнение команды заменить (111, 27) преобразует строку 05111150 в строку 0527150.

Если в строке нет вхождений цепочки v , то выполнение команды заменить (v, w) не меняет эту строку.

Б) нашлось (v).

Эта команда проверяет, встречается ли цепочка v в строке исполнителя Редактор. Если она встречается, то команда

возвращает логическое значение «истина», в противном случае возвращает значение «ложь». Строка исполнителя при этом не изменяется.

Цикл

ПОКА условие

последовательность команд

КОНЕЦ ПОКА

выполняется, пока условие истинно.

В конструкции

ЕСЛИ условие

ТО команда1

ИНАЧЕ команда2

КОНЕЦ ЕСЛИ

выполняется команда1 (если условие истинно) или команда2 (если условие ложно).

Дана программа для Редактора:

НАЧАЛО

ПОКА нашлось (>1) ИЛИ нашлось (>2) ИЛИ нашлось (>0)

ЕСЛИ нашлось (>1)

ТО заменить (>1, 22>)

КОНЕЦ ЕСЛИ

ЕСЛИ нашлось (>2)

ТО заменить (>2, 2>)

КОНЕЦ ЕСЛИ

ЕСЛИ нашлось (>0)

ТО заменить (>0, 1>)

КОНЕЦ ЕСЛИ

КОНЕЦ ПОКА

КОНЕЦ

На вход приведённой выше программе поступает строка, начинающаяся с символа «>», а затем содержащая 39 цифр «0», n цифр «1» и 39 цифр «2», расположенных в произвольном порядке.

Определите наименьшее значение n, при котором сумма числовых значений цифр строки, получившейся в результате выполнения программы, является простым числом.

Ответ: 5

▼ Пояснения:

```
def str_process(s): # в функцию записываем из условия обработку строки,
# переписав её под питон
while '>1' in s or '>2' in s or '>0' in s:
    if '>1' in s:
        s = s.replace('>1', '22>', 1)
    elif '>2' in s:
        s = s.replace('>2', '2>', 1)
    elif '>0' in s:
        s = s.replace('>0', '1>', 1)
return s

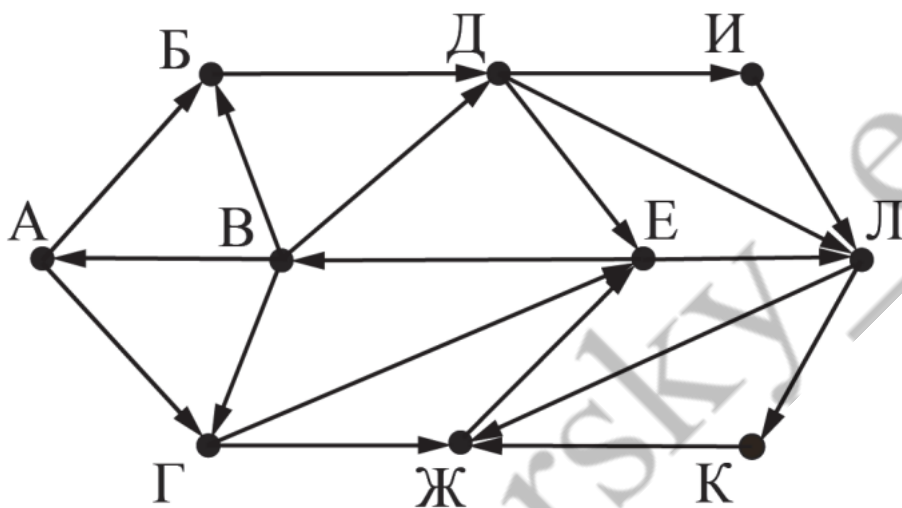
def isPrime(k): #функция проверки на простоту числа
check = True
for i in range(2, int(k/2)+1):
    if k%i == 0: #проверяем делится ли наше число на текущий делитель
        check = False
        break #если мы нашли хоть один делитель - число уже не простое
return check

summ = 0 #создаём переменную, куда будем пихать сумму цифр числа
for n in range(50): #перебираем n в небольшом промежутке, чтоб найти наименьшее
    origs = '>' + 39*'0' + 39*'2' + n*'1' #собираем из кусочков строку с переменным количеством n
    proc = str_process(origs) #с помощью функции преобразуем строку
    proc = proc.replace('>', '', 1) #убираем лишний знак '>'
```

```
summ = sum(map(int, proc)) #считаем сумму цифр получившейся строки
if isPrime(summ): #проверяем на простоту получившееся число
    print(n, summ)
    break #если нам подошло, то дальше нет смысла смотреть
```

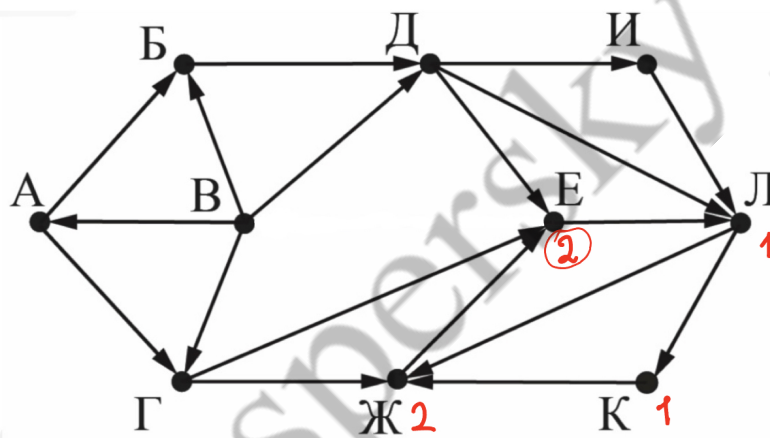
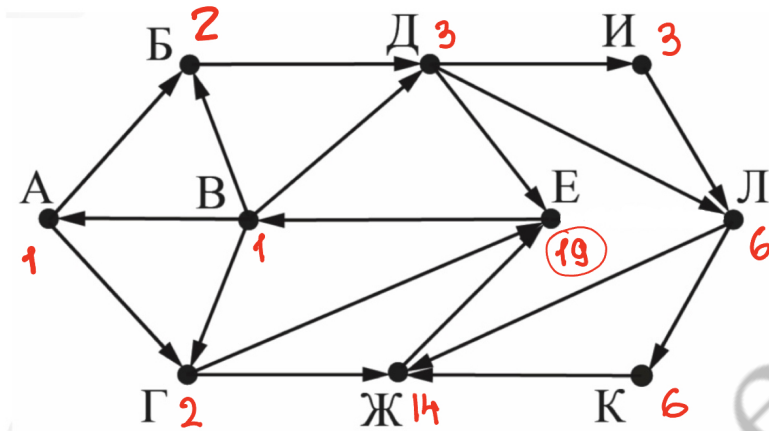
№13 На рисунке представлена схема дорог, связывающих города А, Б, В, Г, Д, Е, Ж, И, К, Л. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой.

Определите количество различных путей ненулевой длины, которые начинаются и заканчиваются в городе Е, не содержат этот город в качестве промежуточного пункта и проходят через промежуточные города не более одного раза.



Ответ: 21

▼ Пояснения:



Ответ: $19 + 2 = 21$

№14 Операнды арифметического выражения записаны в системе счисления с основанием 15.

$$123x_{15} + 1x233_{15}$$

В записи чисел переменной x обозначена неизвестная цифра из алфавита 15-ричной системы счисления. Определите наименьшее значение x , при котором значение данного арифметического выражения кратно 14. Для найденного значения x вычислите частное от деления значения арифметического выражения на 14 и укажите его в ответе в десятичной системе счисления. Основание системы счисления в ответе указывать не нужно.

Ответ: 8767

▼ Пояснения:

```
for x in '0123456789ABCDE': # проходимся переменной x по всем возможным цифрам в 15 СС
    if (int('123' + x + '5', 15) + int('1' + x + '233', 15)) % 14 == 0: # если сумма чисел в выражении кратна 14
        print((int('123' + x + '5', 15) + int('1' + x + '233', 15)) // 14) # выводим ответ
        break # если ответ вывели, останавливаем цикл
```

№15 Обозначим через ДЕЛ(n, m) утверждение «натуральное число n делится без остатка на натуральное число m ». Для какого наименьшего натурального числа A формула:

$$(\text{ДЕЛ}(x, 2) \rightarrow \neg \text{ДЕЛ}(x, 3)) \vee (x + A \geq 100)$$

тождественно истинна (т.е. принимает значение 1) при любом натуральном значении переменной x ?

Ответ: 94

▼ Пояснения:

```
def f(x, a): # функция от x и a
    return (x % 2 != 0 or x % 3 != 0) or (x + a >= 100) # возвращаем рез-тат функции

a = 1 # переменная для a
while True: # пока не найдём ответ
    if all(f(x, a) for x in range(1, 100)): # проверяем, если для какого-то A ф-я тожд. истинна при любом натур. x. Если да, то
        print(a) # выводим A
        break # останавливаем цикл while True, т.к. ответ уже найден
    a += 1 # если ответ не нашли, A увеличиваем на 1
```

№16 Алгоритм вычисления значения функции $F(n)$, где n – натуральное число, задан следующими соотношениями:

$$F(n) = 1 \text{ при } n = 1;$$
$$F(n) = n \times F(n - 1), \text{ если } n > 1.$$

Чему равно значение выражения $F(2023) / F(2020)$?

Ответ: 8266912626

▼ Пояснения:

Начнем раскрывать $F(2023)$ по условию для $n > 1$:

$$F(2023) / F(2020) = 2023 \times F(2022) / F(2020)$$

Видим, что с $F(2022)$ можно проделывать то же упрощение. Начнём последовательно делать это:

$$2023 \times 2022 \times F(2021) / F(2020) = 2023 \times 2022 \times 2021 \times F(2020) / F(2020).$$

Сокращаем $F(2020)$ в числителе и знаменателе и считаем $2023 \times 2022 \times 2021 \times 1 = \mathbf{8266912626}$

№17 В файле содержится последовательность целых чисел. Элементы последовательности могут принимать целые значения от $-10\,000$ до $10\,000$ включительно. Определите количество пар последовательности, в которых только одно число оканчивается на 3, а сумма квадратов элементов пары не меньше квадрата максимального элемента последовательности, оканчивающегося на 3. В ответе запишите два числа: сначала количество найденных пар, затем максимальную из сумм квадратов элементов таких пар. В данной задаче под парой подразумевается два идущих подряд элемента последовательности.

Ответ: 180 190360573

▼ Пояснения:

```
f = open('17.txt') # считываем файл
a = [int(x) for x in f.readlines()] # Заносим все числа из файла в список

maxi = -100000
count = 0
maxi_sum = -20000

for i in a: # проходимся по массиву и ищем максимальное число, оканчивающееся на 3
    if i % 10 == 3 and i > maxi:
        maxi = i

for i in range(len(a) - 1):
    if (abs(a[i]) % 10 == 3 and abs(a[i + 1]) % 10 != 3) or (abs(a[i]) % 10 != 3 and abs(a[i + 1]) % 10 == 3): # проверяем пары на условии
        if a[i] * a[i] + a[i + 1] * a[i + 1] >= maxi * maxi: # проверяем условие на то, что сумма квадратов элементов пар не меньше maxi
            count += 1 # увеличиваем счетчик
            maxi_sum = max(maxi_sum, a[i] * a[i] + a[i + 1] * a[i + 1]) # ищем максимальную сумму квадратов элементов пар
print(count, maxi_sum) # выводим ответ
```

№18 Квадрат разлинован на $N \times N$ клеток ($1 < N < 30$). Исполнитель Робот может перемещаться по клеткам, выполняя за одно перемещение одну из двух команд: вправо или вниз. По команде вправо Робот перемещается в соседнюю правую клетку, по команде вниз – в соседнюю нижнюю. Квадрат ограничен внешними стенами. Между соседними клетками квадрата также могут быть внутренние стены. Сквозь стену Робот пройти не может. Перед каждым запуском Робота в каждой клетке квадрата лежит монета достоинством от 1 до 100. Посетив клетку, Робот забирает монету с собой; это также относится к начальной и конечной клеткам маршрута Робота. Определите максимальную и минимальную денежные суммы, которые может собрать Робот, пройдя из левой верхней клетки в правую нижнюю.

В ответе укажите два числа – сначала максимальную сумму, затем минимальную.

Исходные данные представляют собой электронную таблицу размером $N \times N$, каждая ячейка которой соответствует клетке квадрата. Внутренние и внешние стены обозначены утолщенными линиями.

Пример входных данных

1	8	8	4
10	1	1	3
1	3	12	2
2	3	5	6

Ответ: 1099 1026

▼ Пояснения:

0) Скопируем таблицу в новое место и удалим оттуда все значения, там будем искать ответ:)

1) Расставим формулы рядом со стенками. Когда робот находится справа от вертикальной стены, как на рисунке, мы понимаем, что он не мог туда прийти слева (ходом вправо), а мог только сверху. Поэтому формулы в таких ячейках будут принимать вид: "то, что в соответствующей ячейке исходной таблицы + то, что сверху в текущей таблице". Точно так же снизу от горизонтальных стенок, где в ячейку прийти можно только слева, но не сверху.

2) После того как поставили формулы для всех ячеек рядом со стенками, нужно расставить основную формулу, которая будет выбирать максимальное из верхней и левой ячейки и прибавлять к нему значение из этой же ячейки исходной таблицы. Например: =МАКС(Т40;S41)+Т20

Таким образом нужно заполнить всю оставшуюся таблицу и в правой нижней ячейке получим ответ для максимального, затем можно заменить формулу на МИН и получим ответ для минимального 😊

№19 Два игрока, Петя и Ваня, играют в следующую игру. Перед игроками лежит куча камней. Игроки ходят по очереди, первый ход делает Петя. За один ход игрок может добавить в кучу один камень или увеличить количество камней в куче в два раза. Для того чтобы делать ходы, у каждого игрока есть неограниченное количество камней.

Игра завершается в тот момент, когда количество камней в куче становится не менее 129. Победителем считается игрок, сделавший последний ход, т.е. первым получивший кучу из 129 или больше камней. В начальный момент в куче было S камней, $1 \leq S \leq 128$.

Будем говорить, что игрок имеет выигрышную стратегию, если он может выиграть при любых ходах противника. Укажите

такое значение S, при котором Петя не может выиграть за один ход, но при любом ходе Пети Ваня может выиграть своим первым ходом.

Ответ: 64

▼ Пояснения:

```
# Заполним функцию с возможными ходами
def step(p):
    return p+1, p*2

# И определим значение, при котором игрок побеждает
def win(p):
    return p >= 129

# Создадим словарь, в котором ключом будет номер категории, а значениями в ней - соответствующие S
positions = {}

# Для начала словарь нужно заполнить нулями, чтобы потом можно было делать ходы
for p in range(1,200):
    if not win(p):
        positions[p] = 0

# В первую категорию внесем все позиции, из которых можно победить за 1 ход
for p in positions:
    if any(win(p1) for p1 in step(p)): # хотя бы один ведёт к победе
        positions[p] = 1

# Во вторую категорию внесем позиции, при которых Ваня выигрывает первым ходом при любой игре Пети
for p in positions:
    if positions[p] != 1:
        if all(positions[p1] == 1 for p1 in step(p)): # все ведут в 1 категорию
            positions[p] = 2

# Выведем все позиции, из которых можно победить за 1 ход
print([p for p in positions if positions[p] == 1])

# Выведем полученные во второй категории значения
# Ваня выигрывает первым ходом при любой игре Пети
print([p for p in positions if positions[p] == 2])
```

№20 Для игры, описанной в задании 19, найдите два **наименьших** значения S, при которых у Пети есть выигрышная стратегия, причём одновременно выполняются два условия:

- Петя не может выиграть за один ход;
- Петя может выиграть своим вторым ходом независимо от того, как будет ходить Ваня.

Найденные значения запишите в ответе в порядке возрастания.

Ответ: 32 63

▼ Пояснения:

```
# Заполним и выведем 3-ю категорию, где
# Петя выигрывает своим вторым ходом
for p in positions:
    if positions[p] == 0:
        if any(positions[p1] == 2 for p1 in step(p)): # можно подставить
            positions[p] = 3

print([p for p in positions if positions[p] == 3])
```

№21 Для игры, описанной в задании 19, найдите **минимальное** значение S, при котором одновременно выполняются два условия:

- у Вани есть выигрышная стратегия, позволяющая ему выиграть первым или вторым ходом при любой игре Пети;
- у Вани нет стратегии, которая позволит ему гарантированно выиграть первым ходом.

Если найдено несколько значений S, в ответе запишите минимальное из них

Ответ: 62

▼ Пояснения:

```

# Заполним и выведем 4-ю категорию, где
# Ваня выигрывает своим вторым или первым ходом
for p in positions:
    if positions[p] == 0:
        if all(positions[p1] in (1, 3) for p1 in step(p)): # можно подставить
            positions[p] = 4

print([p for p in positions if positions[p] == 4])

```

№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 . В этом случае процессы могут выполняться только последовательно. Информация о процессах представлена в файле в виде таблицы. В первой строке таблицы указан идентификатор процесса (ID), во второй строке таблицы – время его выполнения в миллисекундах, в третьей строке перечислены с разделителем «;» ID процессов, от которых зависит данный процесс. Если процесс является независимым, то в таблице указано значение 0.

Типовой пример организации данных в файле:

ID процесса B	Время выполнения процесса B (мс)	ID процесса(ов) A
1	4	0
2	3	0
3	1	1; 2
4	7	3

Определите **минимальное** время, через которое завершится выполнение всей совокупности процессов, при условии, что все независимые друг от друга процессы могут выполняться параллельно.

Типовой пример имеет иллюстративный характер. Для выполнения задания используйте данные из прилагаемого файла.

Ответ: 17

▼ Пояснения:

Основной ключ к решению - понимание, как зависят данные в задании процессы друг от друга.

Простыми словами, основываясь на таблице из типового примера: 1 и 2 процесс не зависят от каких-либо других (это показывает 0 в третьем столбце), а значит, могут начинаться с самого начала и идти параллельно. 3 процесс зависит от 1 и 2, то есть, он может начаться только после того, когда окончат свое выполнение и 1, и 2 вместе.

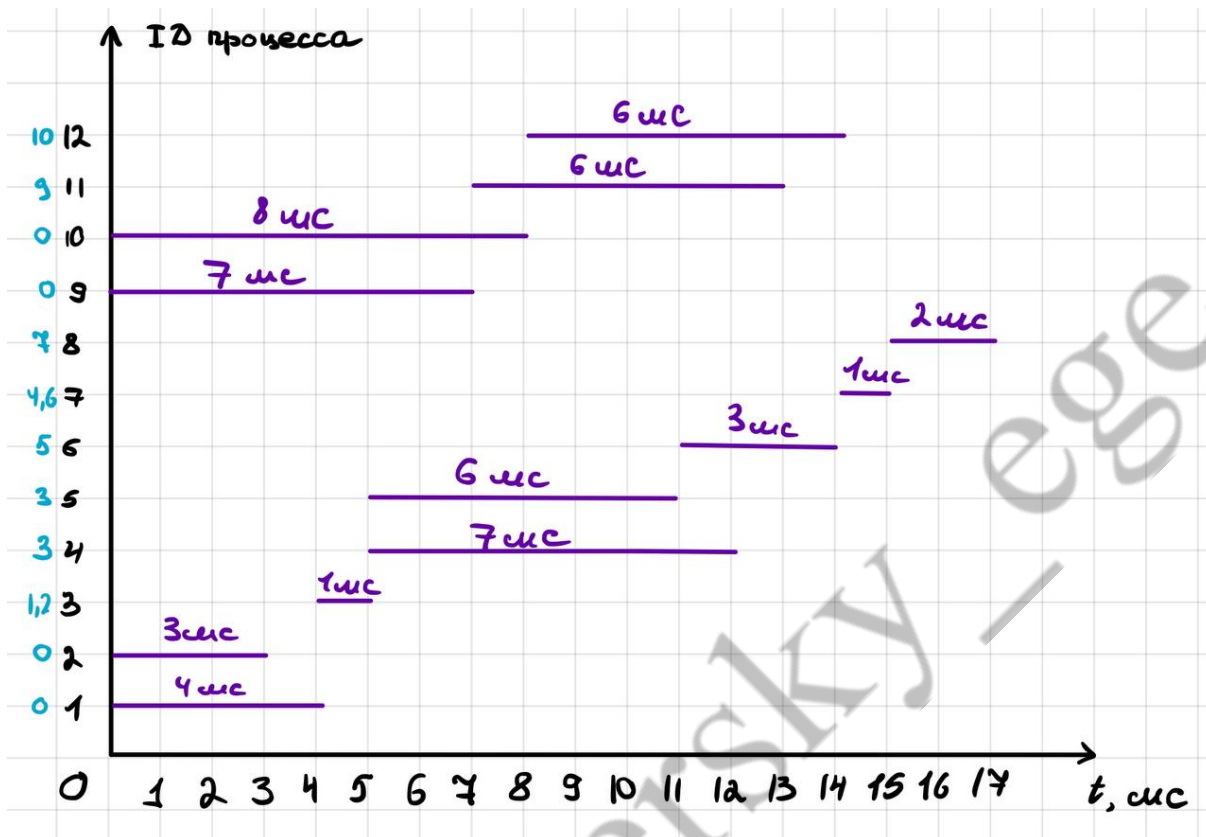
Откроем файл, в нем описано всего 12 процессов, а значит, в данном случае можно найти решение вручную.

ID процесса В	Время выполнения процесса В (мс)	ID процесса (ов) А
1	4	0
2	3	0
3	1	1; 2
4	7	3
5	6	3
6	3	5
7	1	4; 6
8	2	7
9	7	0
10	8	0
11	6	9
12	6	10

Для наглядности решения нанесем значения на график: по оси ОУ - время в мс, оно показывает, в какой момент времени закончит свое выполнение тот или иной процесс; по оси ОХ - ID-номера выполняемых процессов(черным цветом) и ID-номера процессов, от которых они зависят(голубым), взятые в таблице. Фиолетовые отрезки - длительность процессов.

Пример заполнения:

- Первыми на график наносим процессы 1 и 2, они независимые, а значит, могут идти параллельно и с самого начала работы. Их длительность - 4 и 3 мс соответственно
- 3 процесс может начаться только по окончании процессов 1 и 2, то есть, по истечении 4 мс с начала всей работы. Его длительность - 1 мс
- 4 процесс зависит от 3 процесса, а значит, может начаться только в тот момент, когда 3 закончится, то есть, по истечении 5 мс с начала всей работы. Длительность его - 7 мс
- Аналогичным образом наносим на график отрезки, соответствующие каждому процессу, основываясь на информации о том, от каких других процессов они зависят
- Ответ на задачу - это самая правая точка отрезков на графике, в данном задании она соответствует точке 17 по оси ОХ



№23 Исполнитель преобразует число на экране.

У исполнителя есть две команды, которым присвоены номера:

1. Прибавить 1
2. Умножить на 2

Программа для исполнителя – это последовательность команд. Сколько существует программ, для которых при исходном числе 1 результатом является число 35, при этом траектория вычислений содержит число 10 и не содержит 17?

Траектория вычислений программы – это последовательность результатов выполнения всех команд программы. Например, для программы 121 при исходном числе 7 траектория будет состоять из чисел 8, 16, 17.

Ответ: 98

▼ Пояснения:

```
f = {1: 1} #словарь значений
for n in range(2, 36):
    f[n] = 0
    f[n] += f[n - 1] #команда прибавить 1
    if n // 2 in f and n%2==0:#команда умножить на 2
        f[n] += f[n // 2]

    if n == 17:#избегаем 17
        f[n] = 0
    if n == 10:#содержит 10
        for x in range(n):
            f[x] = 0
print(f[35])
```

№24 Текстовый файл состоит из символов A, F, C, D и O.

Определите максимальное количество идущих подряд пар символов вида

СОГЛАСНАЯ + ГЛАСНАЯ

в прилагаемом файле.

Для выполнения этого задания следует написать программу.

Ответ: 95

▼ Пояснения:

```
s = open('24.txt').readline() # открываем файл

# находим и заменяем все пары согласная + гласная на *
for a in 'FCD':
    for b in 'AO':
        s = s.replace(a + b, '*')

cur = 0 # заводим счётчик текущего количества идущих подряд пар
maxi = 0 # заводим переменную - максимальное количество идущих подряд пар
for i in range(len(s)):
    if s[i] == '*': # если найдена звёздочка(то есть пара), то..
        cur += 1 # ..прибавляем 1 к текущему количеству подряд идущих пар
        maxi = max(maxi, cur) # проверяем на максимум
    else:
        cur = 0 # если звёздочка не найдена, то обнуляем текущий счетчик
print(maxi) # выводим ответ - максимум идущих подряд пар
```

№25 Назовём маской числа последовательность цифр, в которой также могут встречаться следующие символы:

- символ «?» означает ровно одну произвольную цифру;
- символ «*» означает любую последовательность цифр произвольной длины; в том числе «*» может задавать и пустую последовательность.

Например, маске 123*4?5 соответствуют числа 123405 и 12300405. Среди натуральных чисел, не превышающих 10^{10} , найдите все числа, соответствующие маске 1?2139*4, делящиеся на 2023 без остатка.

В ответе запишите в первом столбце таблицы все найденные числа в порядке возрастания, а во втором столбце – соответствующие им результаты деления этих чисел на 2023.

Количество строк в таблице для ответа избыточно.

Ответ: 162139404 80148

1321399324 653188

1421396214 702618

1521393104 752048

▼ Пояснения:

```
import itertools

s = '0123456789'
# запишем число из маски 1?2139*4: x = '1'+ a + '2139' + b + '4'
# a - 1 цифра
# b - 0,1,2,3 цифры
res = {} # сюда запишем искомые числа
for a in s:
    for k in range(4): # кол-во цифр в b
        for b_tuple in itertools.product(s, repeat=k):
            b = ''.join(b_tuple)
            x = '1' + a + '2139' + b + '4'
            int_x = int(x)
            if int_x < 10 ** 10 and int_x % 2023 == 0:
                res[int_x] = int_x // 2023

for x in sorted(res.keys()):
    print(x, res[x])
```

№26 В магазине для упаковки подарков есть N кубических коробок. Самой интересной считается упаковка подарка по принципу матрёшки – подарок упаковывается в одну из коробок, та в свою очередь в другую коробку и т.д. Одну коробку

можно поместить в другую, если длина её стороны хотя бы на 3 единицы меньше длины стороны другой коробки. Определите наибольшее количество коробок, которое можно использовать для упаковки одного подарка, и максимально возможную длину стороны самой маленькой коробки, где будет находиться подарок. Размер подарка позволяет поместить его в самую маленькую коробку.

Входные данные

В первой строке входного файла находится число N – количество коробок в магазине (натуральное число, не превышающее 10 000). В следующих N строках находятся значения длин сторон коробок (все числа натуральные, не превышающие 10 000), каждое – в отдельной строке.

Запишите в ответе два целых числа: сначала наибольшее количество коробок, которое можно использовать для упаковки одного подарка, затем максимально возможную длину стороны самой маленькой коробки в таком наборе.

Типовой пример организации данных во входном файле

```
5
43
40
32
40
30
```

Пример входного файла приведён для пяти коробок и случая, когда минимальная допустимая разница между длинами сторон коробок, подходящих для упаковки «матрёшкой», составляет 3 единицы.

При таких исходных данных условию задачи удовлетворяют наборы коробок с длинами сторон 30, 40 и 43 или 32, 40 и 43 соответственно, т.е. количество коробок равно 3, а длина стороны самой маленькой коробки равна 32.

Типовой пример имеет иллюстративный характер. Для выполнения задания используйте данные из прилагаемых файлов.

Ответ: 2767 51

▼ Пояснения:

```
tf = open('26.txt')
n = int(tf.readline()) # n = кол-во коробок
a = [int(tf.readline()) for i in range(n)] # размеры коробок
a.sort(reverse=True) # смотрим с большей коробки

maxi = a[0] # запоминаем размер наибольшей коробки
count = 1 # кол-во коробок
for i in range(1, n):
    if a[i] <= maxi - 3:
        count += 1
        maxi = a[i]
print(count, maxi)
```

№27 У медицинской компании есть N пунктов приёма биоматериалов на анализ. Все пункты расположены вдоль автомагистрали и имеют номера, соответствующие расстоянию от нулевой отметки до конкретного пункта. Известно количество пробирок, которое ежедневно принимают в каждом из пунктов. Пробирки перевозят в специальных транспортировочных контейнерах вместимостью не более 36 штук. Каждый транспортировочный контейнер упаковывается в пункте приёма и вскрывается только в лаборатории.

Стоимость перевозки биоматериалов равна произведению расстояния от пункта до лаборатории на количество контейнеров с пробирками. Общая стоимость перевозки за день равна сумме стоимостей перевозок из каждого пункта в лабораторию. Лабораторию расположили в одном из пунктов приёма биоматериалов таким образом, что общая стоимость доставки биоматериалов из всех пунктов минимальна.

Определите минимальную общую стоимость доставки биоматериалов из всех пунктов приёма в лабораторию.

Входные данные

Дано два входных файла (файл А и файл В), каждый из которых в первой строке содержит число N ($1 \leq N \leq 10\,000\,000$) –

количество пунктов приёма биоматериалов. В каждой из следующих N строк находится два числа: номер пункта и количество пробирок в этом пункте (все числа натуральные, количество пробирок в каждом пункте не превышает 1000). Пункты перечислены в порядке их расположения вдоль дороги, начиная от нулевой отметки.

В ответе укажите два числа: сначала значение искомой величины для файла А, затем – для файла В.

Типовой пример организации данных во входном файле

```
6
1 100
2 200
5 4
7 3
8 2
10 190
```

При таких исходных данных и вместимости транспортировочного контейнера, составляющей 96 пробирок, компании выгодно открыть лабораторию в пункте 2. В этом случае сумма транспортных затрат составит: $1 \cdot 2 + 3 \cdot 1 + 5 \cdot 1 + 6 \cdot 1 + 8 \cdot 2$.

Типовой пример имеет иллюстративный характер. Для выполнения задания используйте данные из прилагаемых файлов.

Предупреждение: для обработки файла В не следует использовать переборный алгоритм, вычисляющий сумму для всех возможных вариантов, поскольку написанная по такому алгоритму программа будет выполняться слишком долго.

Ответ: 51063 5634689219329

▼ Пояснения:

```
# решение на 1 балл

tf = open('27_A.txt')
n = int(tf.readline()) # n = кол-во складов
m = 36 # m = кол-во пробирок в сумке
a = {} # номер пункта: пробирок
for i in range(n):
    x, y = map(int, tf.readline().split())
    if y % m == 0: # считаем кол-во сумок вместо пробирок
        a[x] = y // m
    else:
        a[x] = y // m + 1
min_cena = 10 ** 20
for x in a:
    cena = 0
    for y in a:
        r = abs(x - y)
        cena += a[y] * r
    min_cena = min(cena, min_cena)
print(min_cena)

# 27 на 2 балла
tf = open('27_B.txt')
n = int(tf.readline()) # n = кол-во складов
m = 36 # m = кол-во пробирок в сумке
a = {} # номер пункта: пробирок
forward_sum = 0
cena = 0
for i in range(n):
    x, y = map(int, tf.readline().split())
    if y % m == 0: # считаем кол-во сумок вместо пробирок
        a[x] = y // m
    else:
        a[x] = y // m + 1
    cena += a[x] * x # как-будто везем в начало дороги
    forward_sum += a[x]

# решение на 2 балла
min_cena = 10 ** 20
back_sum = 0
y = 0 # предыдущий пункт
r_b = 0 # расстояние до предыдущего пункта
```

```
r_f = 0 # расстояние до следующего
for x in sorted(a.keys()):
    r_b = x - y
    cena += back_sum * r_b # предыдущие стали дороже
    cena -= r_b * forward_sum # следующие дешевле
    min_cena = min(min_cena, cena)
    back_sum += a[x] # предыдущих больше
    forward_sum -= a[x] # следующих меньше
    y = x
print(min_cena)
```

t.me/kaspersky_ege