Алгебра логики

¬ A, $\bar{A}$ не A (отрицание, инверсия)

A □ B, $A \cdot B$ A и B (логическое умножение, конъюнкция)

A □ B, $A + B$ A или B (логическое сложение, дизъюнкция)

A → B импликация (следование)

A □ B эквивалентность (равносильность) A → B =

¬ A □ B или A → B = $\bar{A} + B$

формулы де Моргана: $\overline{A \cdot B} = \bar{A} + \bar{B}$ $\overline{A + B} = \bar{A} \cdot \bar{B}$

$1 + A = 1, 0 \cdot A = 0,$ $A + 0 = A,$ $A \cdot 1 = A,$ $A + A = A$

$A \cdot \bar{A} = A, A + \bar{A} = 1, A \cdot \bar{A} = 0, (A + B) \cdot (\bar{A} + B) = B$ $A \cdot (B + C) = A \cdot B + A \cdot C, A + (B \cdot C) = (A + B) \cdot (A + C)$ **Задание №2** пример программы **(СЛЕДИ ЗА СКОБКАМИ)**

```
for x in range(2):    for y in range(2):    for z in range(2):
for w in range(2):          if ((not((x or y) <= (z and w))) and
(x <= w)) == True:                print(x,y,z,w)
```

Задание №4

Условие Фано: для того, чтобы сообщение, записанное с помощью неравномерного по длине кода, однозначно декодировалось, достаточно, чтобы **никакой код не был началом другого (более длинного) кода**. **Обратное условие Фано** также является достаточным условием однозначного декодирования неравномерного кода. В нём требуется, чтобы **никакой код не был окончанием другого (более длинного) кода**.

Для возможности однозначного декодирования достаточно выполнения одного из условий — или прямого, или обратного. ***Внимательно читай, нужно ли использовать весь алфавит! Если да, то оставь одно место!*** **Задание №5**

Внимательно читай, что нужно найти!

В двоичной системе:

- четные числа оканчиваются на 0, нечетные – на 1; - числа, которые делятся на 4, оканчиваются на 00, и т.д.; числа, которые делятся на 2^k, оканчиваются на *k* нулей

bin(x) представление числа x в двоичной системе oct(x) представление числа x в восьмеричной системе hex(x) представление числа x в шестнадцатеричной системе int('xxx',n) перевод из n-ой СС в 10СС (xxx число в n СС)
Срез от x до y(не включительно) с шагом k - a[x:y:k]

```
def f(n):    s=''
while n > 0:        s
= str(n%3)+s
      n //= 3    return
s    c = set()  for n in
range(1,100):
   s = f(n)    if
n%3 == 0:
s = s + s[-3:]
else:
      s = s + f((n%3)*3)
   r = int(s,3)
if r > 150:
      c.add(n)
print(min(c))
```

Пример: На вход алгоритма подается натуральное число *N*. Алгоритм строит по нему новое число *R* следующим образом:

1) Строится троичная запись числа *N*
2) Если *N* кратно 3, то в конец записи дописываются три последние цифры числа.
3) Если *N* не кратно 3, то остаток от деления умножается на 3, переводится в троичную систему и затем дописывается к числу.

Полученная таким образом запись является троичной записью искомого числа *R*.

Укажите минимальное число *N*, после обработки которого автомат получает число большее 150.

Задание №6

Пример: Повтори 21 [Вперёд 10 Направо 60]

Объединение – все, что входит в обе фигуры Пересечение – то, что входит только в пересечение фигур (их общая часть). Обрати внимание: точки на линии учитывать следует или нет

Поставить сетку на 1:1
использовать Черепаха
алг нач
. опустить хвост
. нц 21 раз
. . вперед(7)
. . вправо(60)
. кц
кон

```
from turtle import *
tracer(0) c = 30
lt(90) for i in
range(21):
   fd(7*c)
   rt(60) up() for x in
range(-20, 20):    for y in
range(-20, 20):
      goto(x*c, y*c)
dot(3,'red') update()
#exitonclick()#пайчарм
```

Прога, которая сразу выдает ответ

Повтори 21 [Вперёд 10 Направо 60]

```
from turtle import *
tracer(0)
color("black", "red")
m = 50
begin_fill()
left(90) for i in
range(6):
forward(7*m)
right(60)
end_fill()
update()


canvas = getcanvas()
cnt = 0 for y in range(-110*m, 110*m):
for x in range(-110*m, 110*m):        item
= canvas.find_overlapping(x,y,x,y)
      if len(item) > 0:#сколько на границе
cnt += 1
      if len(item) == 1 and item[0] == 5:#сколько точек внутри
области
         cnt += 1
print(cnt)
done() exit()
```

Задание №7

1 байт = 8 бит = 2^3 бит, **1 Кбайт** = 1024 байта = 2^{10} байта= $2^{10} \cdot 2^3$ бит = 2^{13} бит, **1 Мбайт** = 1024 Кбайта = 2^{10} Кбайта = $2^{10} \cdot 2^{10}$ байта = 2^{20} байта = $2^{20} \cdot 2^3$ бит = 2^{23} бит.

Для хранения растрового изображения нужно выделить в памяти $I = x \cdot y \cdot i$ битов, где x - ширина, y - высота и i – глубина цвета (разрядность кодирования). Количество цветов = 2^i, i -глубина цвета
Для хранения информации о звуке длительностью t секунд, закодированном с частотой дискретизации f Гц и глубиной кодирования Bбит и количестве каналов k требуется $k \cdot B \cdot f \cdot t$ бит памяти; например, при стерео записи ($k = 2$), f = 8кГц, глубине кодирования 16 бит на отсчёт и длительности звука 128 секунд требуется $I = 2 * 8000 \cdot 16 \cdot 128/8/1024/1024 \approx 3{,}9$ Мбайт
Ориг больше сжатого на 30%, значит сжатое = ориг / на 1.3
Сжатое меньше ориг на 30% значит сжатое = ориг * на 0.7

Изображение	Звук
Если разрешение увел в n раз, то объем увел в n^2	Если разрешение увел в n раз, то объем увел в n

Задание №8

Формула для вычисления числа перестановок с повторениями; для двух разных символов она выглядит так: $P(n_A, n_.) = \frac{(n_A+n_b)!}{n_A!\ n_b!}$

Здесь n_a – количество занятых мест, n_b – количество свободных и восклицательный знак обозначает факториал натурального числа.
Число не может начинаться с 0! 0,2,4,6,8 четные 1,3,5,7,9 нечетные
Если задание на СС , то +1 или -1, вспомни про номера слов!

Пример: В качестве кодовых слов Игорь использует трёхбуквенные слова, в которых могут быть только буквы Ш, К, О, Л, А, причём буква К появляется ровно 1 раз. Каждая из других допустимых букв может встречаться в кодовом слове любое количество раз или не встречаться совсем. Сколько различных кодовых слов может использовать Игорь?

```
n=0 s='школа' for a in s:  for
b in s:    for c in s:      if
(a+b+c).count('к')==1:
      n+=1 print(n)
---------------------------------------- from
itertools import product k = 0 for x in
product('ШКОЛА', repeat = 3):    s =
''.join(x)    if s.count('К')==1:
      k += 1 print(k)
```

Задание №11 **Следить за округлением все вместе или по частям!**

Пример: Для регистрации на сайте некоторой страны пользователю требуется придумать пароль. Длина пароля – ровно 11 символов. В качестве символов используются десятичные цифры и 12 различных букв местного алфавита, причём все буквы используются в двух начертаниях: как строчные, так и заглавные (регистр буквы имеет значение!). Под хранение каждого такого пароля на компьютере отводится минимально возможное и одинаковое целое количество байтов, при этом используется посимвольное кодирование и все символы кодируются одинаковым и минимально возможным количеством битов. Определите объём памяти в байтах, который занимает хранение 60 паролей.

Решение:

- согласно условию, в пароле можно использовать 10 цифр (0..9) + 12 заглавных букв местного алфавита + 12 строчных букв, всего 10 + 12 + 12 = 34 символа
- для кодирования номера одного из 34 символов нужно выделить 6 бит памяти (5 не хватает, они закодируют только $2^5 = 32$ варианта) - для хранения всех 11 символов пароля нужно 11 □ 6 = 66 бит - поскольку пароль должен занимать целое число байт, берем ближайшее большее (точнее, не меньшее) значение, которое кратно

8: это 72 = 9 □ 8; то есть один пароль занимает 9 байт

- тогда 60 паролей занимают 9 □ 60 = 540 байт **Ответ: 540.**

Задание №12

Пример: Дана программа для редактора:

```
НАЧАЛО
ПОКА нашлось (25) ИЛИ нашлось (355) ИЛИ нашлось (555)
 ЕСЛИ нашлось (25)
  ТО заменить (25, 5)
 КОНЕЦ ЕСЛИ
 ЕСЛИ нашлось (355)
  ТО заменить (355, 52)
КОНЕЦ ЕСЛИ
 ЕСЛИ нашлось (555)
  ТО заменить (555, 3)
 КОНЕЦ ЕСЛИ
КОНЕЦ ПОКА
КОНЕЦ
```

На вход приведённой выше программе поступает строка, начинающаяся с цифры 2, а затем содержащая n цифр 5 (n > 3). Определите наименьшее значение n, при котором в строке, получившейся в результате выполнения программы, сумма цифр равна 17.

```
for n in range(4, 100):
   s = '2' + n*'5'    while '25' in s or '355'
in s or '555' in s:
      if '25' in s:
         s = s.replace('25','5',1)
if '355' in s:
         s = s.replace('355','52',1)
if '555' in s:
         s = s.replace('555','3',1)
   sm = s.count('2')*2 + s.count('3')*3 +
s.count('5')*5    sm = sum([int(x) for x in s])      if
sm == 17:
      print(n,s)
break
```

Задание №13

IP-адрес состоит из двух частей: адреса сети и адреса узла в этой сети, причём деление адреса на части определяется маской – 32-битным числом, в двоичной записи которого сначала стоят единицы, а потом – нули:

	адрес сети	адрес узла
IP-адрес		
маска	**11.....11**	**00.....00**

Та часть IP-адреса, которая соответствует единичным битам маски, относится к адресу сети, а часть, соответствующая нулевым битам маски – это числовой адрес узла.

Пример: Для узла с IP-адресом 111.81.208.27 адрес сети равен 111.81.192.0. Чему равно наименьшее(наибольшее) возможное значение третьего слева байта маски? Ответ запишите в виде десятичного числа.

Решение:

Адрес сети получается в результате применения поразрядной конъюнкции к заданному IP-адресу узла и маске.

Запишем третий байт IP-адреса и адреса сети в двоичной системе счисления:

IP-адресу	= 11010000$_2$
Маска	= 11?00000$_2$
Адрес сети	= 11000000$_2$

Видим, что два первых слева бита маски **должны быть** единицы, а третий бит может быть как нулем, так и единицей. Для того, чтобы значение было **наименьшим**, этот бит должен быть равен нулю. Получаем, что третий слева байт маски равен 11000000$_2$ = 192$_{10}$

Для того, чтобы значение было **наибольшим**, этот бит должен быть равен единице. Получаем, что третий слева байт маски равен 11100000$_2$ = 224$_{10}$

Подсчет количества адресов в сети

Сеть задана IP-адресом 192.168.32.160 и маской сети 255.255.255.240. Сколько в этой сети IPадресов, для которых сумма единиц в двоичной записи IP-адреса чётна? В ответе укажите только число.

```
k=0 for x1 in '01':
for x2 in '01':
for x3 in '01':
for x4 in '01':
            s = x1 + x2 + x3 + x4
if s.count('1')%2==0:
              k=k+1
print(k)
```

```
from ipaddress import * for ip in
ip_network('192.168.32.160/255.255.255.240'):    if
format(ip).count('1')%2==0:       print(ip)
```

Пример: Сеть задана IP-адресом 255.220.33.160 и маской сети 255.255.**X**.0, где **X** - число, заданное в диапазоне от 0 до 255. Определите минимальное значение **X**, для которого для всех IPадресов этой сети в двоичной записи IP-адреса суммарное количество единиц в левых двух байтах не менее суммарного количества единиц в правых двух байтах. В ответе укажите только число.

Так как первые два байта маски равны 255, то у IP-адреса в левых двух байтах в двоичном представлении будет 13 единиц, в правых двух байтах должно быть <=13. Последний байт маски равен 0, значит в IP-адресе возможно 8 единиц. Поэтому минимальное значение X в маске 240, что позволит получить +4 единицы в IP-адресе и учитывая первую единицу в двоичном представлении числа 33, получим 5 единиц в третьем байте. Что в сумме не превысит 13 единиц в правых двух байтах IP-адреса

Задание №14 НОВЫЙ ТИП

import string

alf = '0123456789' + string.ascii_lowercase[:k](k – сколько букв из алфавита нужно)

Пример: Операнды арифметического выражения записаны в системах счисления с основаниями 15 и 17. 131x115 + 13x317

В записи чисел переменной x обозначена неизвестная цифра, допустимая в данных системах счисления. Определите наибольшее значение x, при котором значение данного арифметического выражения кратно 11. Для найденного значения x вычислите частное от деления значения арифметического выражения на 11 и укажите его в ответе в десятичной системе счисления

```
for x in
'0123456789abcde':
a = int('131'+str(x)+'1',
15)
b = int('13'+str(x)+'3',
17)    s = a + b
if s % 11 == 0:
      print(x, s  // 11)
```

```
for x in range(15):    a =
1*15**4 + 3*15**3 +
1*15**2 + x*15 + 1
   b = 1*17**3 + 3*17**2 + x*17 +
3
   s = a + b    if s
% 11 == 0:
      print(x, s  // 11)
```

Пример: **1 функция**

$F(n) = n$ при $n \le 3$;

$F(n) = n$ // 4 + $F(n–3)$ при 3 < $n \le 32$;

$F(n) = 2 \cdot F(n–5)$ при $n > 32$

Здесь // обозначает деление нацело. В качестве ответа на задание выведите значение $F(100)$

```
import sys
from functools import *
#кэш, чтобы быстрее
работало
@lru_cache(None) def
f(n):    if n <= 3:
return n    elif  n > 3 and
n <= 32:       return n // 4
+ f(n - 3)    elif n > 32:
return 2 * f(n - 5)
#глубина рекурсии 2000
sys.setrecursionlimit(2000)
print(f(100))
```

Пример: **2 функции** Алгоритм вычисления функций $F(n)$ и $G(n)$ задан следующими соотношениями:

$F(1) = G(1) = 1$

$F(n) = 2 \cdot F(n–1) + G(n–1) – 2$, если $n > 1$

$G(n) = F(n–1) + 2 \cdot G(n–1)$, если $n > 1$

Чему равно значение $F(14) + G(14)$?

```
def    f(n):
if n > 1:
      return 2 * f(n-1) + g(n-1)-
2    else: return 1 def g(n):    if
n > 1:       return f(n-1) + 2 *
g(n - 1)    else: return 1
print(f(14) + g(14))
```

Задание №16

```
f = open('17-1.txt')
a = [] for
n in f:
   a.append(int(n))
print(a)
f.close()
```

```
with open('17-1.txt') as f:
   a = list(map(int,
f.readlines()))
```

```
f = open('17.txt')  a
= [int(x) for x in f]
```

Пример:

```
f = open('17-1.txt')
a = list(map(int, f))
max5 = -10001 for i
in range(len(a)):    if
a[i] % 5 == 0:
      max5 = max(max5, a[i])
k = 0 maxs = -20001 for i in
range(len(a)-1):
   if a[i] % max5 == 0 or a[i+1] %
max5 == 0:       k += 1
      maxs = max(maxs, a[i]+a[i+1])
print(k, maxs)
```

Задание №17

Задание №3,9,18

Если есть стенки из чисел, то для макс заменяем на 0, для мин заменяем на 999999.

Перед чем стоит доллар, то и фиксируется. $A14 фикс столбец A, A$14, фикс строка 14.

Формула стандартная	B2+МАКС(B12;A13)
Формула для ладьи (ходит на любое количество клеток по вертикали и горизонтали)	=МАКС($Q14:AC14; AD$1:AD13)+N14
Формула, когда с севера на юг	=МАКС(A12:C12) + B2
Дана последовательность вещественных чисел. Из неё необходимо выбрать несколько подряд идущих чисел так, чтобы каждое следующее число отличалось от предыдущего не более чем на 10. Какую максимальную сумму могут иметь выбранные числа?	=ЕСЛИ(D1>0; ЕСЛИ(ABS(A2-A1) <=10;D1+A2;A2);A2)
2,3.. по величине	Наибольший(промеж, 2)
Копирование столбца с др листа	ВПР(Значение, Таблица, #столбца, 0)
Остаток N при делении на K	Остат(N,K)
Поиск в диапазоне с условием(КОВЫЧКИ)	СЧЁТЕСЛИ(A:A;">0")

Задание №19-21 Читай какие ходы и когда победа!

1 куча 2 хода +1 и *2 Победа >= 25 from functools import * def m(h):
   return h * 2,h + 2

@lru_cache(None) def g(h):   if h >= 25: return "W"   if any(g(i) == "W" for i in m(h)): return "p1" if all(g(i) == "p1" for i in m(h)): return "v1"   if any(g(i) == "v1" for i in m(h)): return "p2"   if all(g(i) == "p1" or g(i) == "p2" for i in m(h)): return "v2"

for i in range(1, 26):
   x = g(i)
print(i, x)

```
from functools import *
def m(h):    a,b = h
   return (a+2,b), (a, b+2), (a*2, b),
(a,b*2)

@lru_cache(None) def g(h):    a, b
= h    if a + b >= 68: return "W"
if any(g(i) == "W" for i in m(h)):
return "p1"      if all(g(i) == "p1"
for i in m(h)):
return "v1"      if any(g(i) == "v1"
for i in m(h)): return "p2"
   if all(g(i) == "p1" or g(i) == "p2" for
i in m(h)): return "v2"


for i in range(1, 60 + 1):
   h = 7, i
x = g(h)
print(i, x)
```

Задание №23

У исполнителя есть две команды, которым присвоены номера: 1. Прибавить 1 2. Умножить на 2

Сколько существует программ, для которых при исходном числе 2 результатом является число 29 и при этом траектория вычислений содержит число 14 и не содержит числа 25?

```
mas = [0] * 30 mas[2] =
1 for i in range(3, 14 +
1):    mas[i] += mas[i -
1]    if i % 2 == 0:
mas[i] += mas[i // 2]
print(mas) for i in
range(15, 29 + 1):    if i
!= 25:
      mas[i] += mas[i - 1]
if i % 2 ==0 and i /2>= 14:
         mas[i] += mas[i // 2]
print(mas)
```

```
def f(x, y):    if x == y:
return 1    elif x > y or
x == 25:
      return 0
else:
      return f(x + 1, y)+ f(x*2,y)
print(f(2,14)*f(14,29))
```

Через Flag

Задание №15 Длина отрезка - конец минус начало Количество точек - конец минус начало + 1 Читай внимательно, ЦЕЛЫЕ НЕОТРИЦАТЕЛЬНЫЕ ИЛИ ПОЛОЖИТЕЛЬНЫЕ ЗНАЧЕНИЯ!

ДЕЛ (Для какого наибольшего натурального числа A формула (¬ДЕЛ(x, A) □ ДЕЛ(x, 21)) → ДЕЛ(x, 14) тождественно истинна (то есть принимает значение 1 при любом натуральном значении переменной x)? **)**

```
for A in range(1,100):    flag = 0    for x in range(1,1000):
if (((x % A != 0) and ( x % 21) == 0) <= (x % 14 == 0)) == 0:
flag = 1    if flag == 0:       print(A)
```

X и Y

```
for A in range(1,200):    flag = 0    for x in range(1,100):       for
y in range(1,100):          if ((y*y <=A)<=(y <= 10)) and (( x <=
9)<=(x * x < A))== 0:             flag = 1    if flag == 0:
      print(A)
```

Побитовая конъюнкция

```
for A in range(0,100):
   flag = 0      for x in
range(0,1000):
      if ((x & 49 == 0) <= ((x & 28 != 0) <= (x & A != 0))) ==
0:         flag = 1         break    if flag == 0:       print(A)
```

Отрезки!

На числовой прямой даны два отрезка: P = [20, 50] и Q = [30,65]. Отрезок A таков, что формула

¬(x ∈ A) → ((x ∈ P) →¬ (x ∈ Q)) истинна при любом значении переменной x. Какова наименьшая возможная длина отрезка A?

```
mind = 10 ** 10 for a1 in
range(180, 660 + 1):
   for a2 in range(a1 + 1, 660+1):
flag = False      for x in
range(180, 660 + 1):
      if ((not(a1<=x<=a2)) <= ((200<=x<=500) (перенос)\
<= (not(300<=x<=650)))) == False:
         flag = True
break      if flag ==
False:       if a2 -a1 <
mind:          mind =
a2 – a1 print(mind/10)
```

Через For else

Задание №15 Длина отрезка - конец минус начало Количество точек - конец минус начало + 1 Читай внимательно, ЦЕЛЫЕ НЕОТРИЦАТЕЛЬНЫЕ ИЛИ ПОЛОЖИТЕЛЬНЫЕ ЗНАЧЕНИЯ!

ДЕЛ (Для какого наибольшего натурального числа A формула (¬ДЕЛ(x, A) □ ДЕЛ(x, 21)) → ДЕЛ(x, 14) тождественно истинна (то есть принимает значение 1 при любом натуральном значении переменной x)? **)**

```
for A in range(1,100):    for x in range(1,1000):       if (((x % A
!= 0) and ( x % 21) == 0) <= (x % 14 == 0)) == 0:
         break    else:
      print(A)
```

X и Y

```
for A in range(1,200):    flag = 0    for x in range(1,100):       for
y in range(1,100):          if ((y*y <=A)<=(y <= 10)) and (( x <=
9)<=(x * x < A))== 0:
         flag = 1    if flag == 0:
      print(A)
```

Побитовая конъюнкция

```
for A in range(0,100):    for x in range(0,1000):       if ((x & 49
== 0) <= ((x & 28 != 0) <= (x & A != 0))) == 0:       break
else:
      print(A)
```

Отрезки!

На числовой прямой даны два отрезка: P = [20, 50] и Q = [30,65]. Отрезок A таков, что формула

¬(x∈ A) → ((x∈ P) →¬ (x ∈Q)) истинна при любом значении переменной x. Какова наименьшая возможная длина отрезка A?

```
mind = 10 ** 10 for a1 in
range(180, 660 + 1):    for a2 in
range(a1 + 1, 660+1):       for x
in range(180, 660 + 1):
        if ((not(a1<=x<=a2)) <= ((200<=x<=500)
<= (not(300<=x<=650)))) == False:            break
else:           if a2 -a1 < mind:               mind = a2 -
a1 print(mind/10)
```

Через all

Задание №15 **Длина отрезка - конец минус начало Количество точек - конец минус начало + 1 Читай внимательно, ЦЕЛЫЕ НЕОТРИЦАТЕЛЬНЫЕ ИЛИ ПОЛОЖИТЕЛЬНЫЕ ЗНАЧЕНИЯ!**

ДЕЛ (Для какого наибольшего натурального числа A формула (¬ДЕЛ(x, A) □ ДЕЛ(x, 21)) → ДЕЛ(x, 14) тождественно истинна (то есть принимает значение 1 при любом натуральном значении переменной x)? **)**

```
def f(x,A):    return (((x % A != 0) and ( x % 21) == 0) <= (x % 14
== 0)) for A in range(1,100):    if all(f(x,A) for x in range(1,1000)):
    print(A)
```

X и Y

```
def f(x,y,A):
  return ((y*y <=A)<=(y <= 10)) and (( x <= 9)<=(x * x < A)) for A
in range(1,200):    if all(f(x,y,A) for x in range(1,100) for y in
range(1,100)):       print(A)
```

Побитовая коньюнкция

```
def f(x,A):
  return ((x & 49 == 0) <= ((x & 28 != 0) <= (x & A != 0)))
for A in range(1,1000):
  if all(f(x,A) for x in range(1,1000)):
    print(A)
break
```

Отрезки!

На числовой прямой даны два отрезка: P = [20, 50] и Q = [30,65]. Отрезок A таков, что формула ¬(x ∈ A) → ((x ∈ P) →¬ (x ∈ Q)) истинна при любом значении переменной x. Какова наименьшая возможная длина отрезка A?

```
def f(x,a1,a2):
  return ((not(a1<=x<=a2)) <= ((200<=x<=500)<=
(not(300<=x<=650)))) mind =
10 ** 10 for a1 in range(180,
660 + 1):    for a2 in range(a1 +
1, 660+1):
    if all(f(x,a1,a2) for x in range(100,700)):
      if a2 -a1 < mind:
mind = a2 - a1
print(mind/10)
```

Задание №24

Следить за краями цикла, если используешь i + 1, i – 1.

Одна строка	Несколько строк
f = open("24.txt") s = f.readline() … f.close()	f = open("24.txt") for s in f: for i in range(len(s)) …. f.close()

Функции в питоне

возвращает позицию первой подстроки subs в строке s	s.find('subs') S.rfind('l') ищет с конца
заменить в строке S все вхождения подстроки old на подстроку new, count раз	S.replace(old, new, count)
Количество A в строке s	s.count('A')
Получить аски код A	Ord('A')
Превратить аски код в символ	Chr(20)
Превращает строку в массив строк, деля по символу, кот в кавычках	s.Split('A')
Ищет длину строки	Len(s)

Пример проги для k =0

В текстовом файле 24.txt находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.	f = open("24.txt") s = f.readline() f.close() k, kmax = 0, 0 for i in range(len(s)): if s[i] == 'C': k += 1 kmax = max(k, kmax) else: k = 0 print(kmax)

Пример проги , когда k = 1

Текстовый файл состоит не более чем из 10^6 символов X, Y и Z. Определите максимальное количество идущих подряд символов, среди которых каждые два соседних различны.	f = open('2.txt') s = f.readline() k, maxS = 1, 1 for i in range(1, len(s)): if s[i]!=s[i-1]: k+=1 maxS = max(k, maxS) else: k = 1 print(maxS) f.close()

Определите символ, который чаще всего встречается в файле сразу после буквы X. В ответе запишите сначала этот символ, а потом сразу (без разделителя) сколько раз он встретился после буквы X.	f = open('1.txt') s = f.readline() a = [0] * 26 nmax, c = 0, 0 for i in range(len(s) - 1): if s[i] == 'X': index = ord (s[i + 1]) - ord ('A') a[index] += 1 for i in range (len(a)): if nmax < a[i]: nmax = a[i] c = i print(chr(c + ord('A')), nmax) f.close()
Текстовый файл состоит из символов A, B, C, D, U. Определите максимальное количество идущих подряд пар символов вида согласная + гласная в прилагаемом файле. Для выполнения этого задания следует написать программу. Ниже приведён файл, который необходимо обработать с помощью данного алгоритма.	f = open('AUBCD.txt') s = f.readline() s = s.replace('U','A') s = s.replace('C','B') s = s.replace('D','B') s = s.replace('BA','*') s = s.replace('B', 'A') s = s.split('A') #print(s) print(len(max(s, key = len))) res = 0 for i in range(len(s)): res = max(res, len(s[i])) print(res)
Текстовый файл состоит из символов N, O и P. Определите максимальное количество идущих подряд последовательностей символов NPO или PNO в прилагаемом файле. Искомая подпоследовательность должна состоять только из троек NPO, или только из троек PNO, или только из троек NPO и PNO в произвольном порядке их следования. Для выполнения этого задания следует написать программу. Ниже приведён файл, который необходимо обработать с помощью данного алгоритма.	f = open('PNO(доп).txt') s = f.readline() s = s.replace('NPO','*') s = s.replace('PNO','*') s = s.replace('N','O') s = s.replace('P','O') s = s.split('O') print(len(max(s, key = len))) res = 0 for i in range(len(s)): res = max(res, len(s[i])) print(res)
Текстовый файл состоит из символов A, B и C. Определите максимальное количество идущих подряд пар символов AB или CB в прилагаемом файле. Искомая подпоследовательность должна состоять только из пар AB, или только из пар CB, или только из пар AB и CB в произвольном порядке следования этих пар.	f = open('24.txt') s = f.readline() s = s.replace('AB','1') s = s.replace('AC','1') s = s.replace('B','A') s = s.replace('C','A') s = s.split('A') maxk = 0 for i in range(len(s)): maxk = max(len(s[i]), maxk) print(maxk)

Пример проги, какая буква встречается чаще всего

Задание №25

Все делители числа	**Проверка на простоту**
del = [] n = int(input()) d = 2 while d * d < n: if n % d == 0: del.append(d) del.append(n // d) d += 1 if d * d == n: del.append(d) print(del)	def isprime(n): d = 2 while d * d <= n: if n % d == 0: return False d += 1 return True for i in range(2, 100): if isprime(i) == True: print(i)

Задание на маску со звездочками и с вопросами

Среди натуральных чисел, не превышающих 10**8, найдите все числа, соответствующие маске 32*823 и не делящиеся на 123 без остатка.	from fnmatch import * for i in range(123, 10**8 + 1, 123): s = str(i) if fnmatch(s, '32*823'): print(i, i // 123) for i in range(123, 10**8 + 1, 123): s = str(i) if s[:2] == '32' and s[-3:] == '823': print(i, i // 123)

Если нечетное количество делителей, то проверяем только числа, которые являются квадратом другого числа! (n ** 0,5 == int(n ** 0,5))

	elif x % 7 == 0:
	res += k14 + k2
	k7 += 1 elif x % 2
	== 0: res +=
	k14 + k7
	k2 += 1
	else:
	res += k14
	k1 += 1 print(res)

Необходимо определить количество пар элементов (ai, aj) этого набора, в которых 1 < i < j < N и сумма элементов кратна 12.	f = open('27.txt') n = int(f.readline()) res, k = 0, 12 mas = [0]*k for i in range(n): x = int(f.readline()) res += mas[(k - x) % k] mas[x % k] += 1 print(res)
Требуется найти наибольшую сумму двух результатов измерений, выполненных с интервалом не менее, чем в 7 минут.	f = open('test.txt') n = int(f.readline()) k = 7 buf = [0] * k max_n, maxSum, elem = 0,0,0 for i in range(k): buf[i] = int(f.readline()) for i in range(k, n): old = buf[i % k] max_n = max(max_n, old) new = int(f.readline()) maxSum = max(masSum, max_n + new) buf[i % k] = new print(maxSum)
Требуется найти наибольшую сумму двух результатов измерений, выполненных с интервалом не менее, чем в 7 минут.	f = open('test.txt') n = int(f.readline()) k = 7 a = [int(x) for x in f] max_n, maxSum = 0,0 for i in range(k, n): max_n = max(max_n, a[i-k]) maxSum = max(masSum, max_n + a[i]) print(maxSum)
МЕТОД ПРЕФИКСНЫХ СУММ Дана последовательность и N натуральных чисел. Рассматриваются все ее непрерывные подпоследовательности, такие что сумма элементов каждой из них кратна 321. Найдите среди них подпоследовательность с минимальной суммой, определите её длину. Если таких подпоследовательности найдено несколько, в ответе укажите количество элементов самой короткой из них.	f = open('27.txt') n = int(f.readline()) smin = 10 ** 11 dlmin = 10 ** 11 s = 0 k = 321 pref = [0] * k dlp = [0] * k for i in range(n): x = int(f.readline()) s += x if s % k == 0: if s < smin: smin, dlmin = s, i + 1 if pref[s % k] != 0: st = s - pref[s % k] dlt = i + 1 - dlp[s % k] if st < smin or (st == smin and dlt < dlmin): smin, dlmin = st, dlt pref[s % k] = s dlp[s % k] = i + 1 print(smin, dlmin)
(ДОСРОК 2022) На каждом 3-м километре кольцевой автодороги с двусторонним движением установлены контейнеры. Центр переработки отходов открыли в одном из пунктов сбора мусора таким образом, чтобы общая стоимость доставки мусора из всех пунктов в этот центр была минимальной. Определите минимальные расходы на доставку мусора в центр переработки отходов. ДЛЯ ЧЕТНОГО КОЛИЧЕСТА	f = open('107_27_B.txt') n = int(f.readline()) a = [int(x) for x in f] mins = 10**20 st = 0 #сумма для 0 элемента for i in range(n): st += a[i]*min(i, n - i) fs = 0#первая фронт сумма for i in range(1, n// 2 + 1): fs += a[i] bs = a[0]#первая бэк сумма for i in range(n // 2 + 1, n): bs += a[i] mins = 10**20 for i in range(1, n): st = st - fs + bs fs = fs - a[i] + a[(i + n // 2) % n] bs = bs + a[i] - a[(i + n // 2) % n] mins = min(mins, st) print(mins*3)

\

(ЕГЭ 2022 резерв) На каждом километре автомагистрали, начиная с первого, расположены пункты питания. Известна суточная потребность каждого пункта питания в количестве готовых обедов. По правилам готовую еду нельзя перевозить на расстояние, превышающее M км.
Для транспортировки используются термоконтейнеры вместимостью не более 6 готовых обедов. Каждый отправляется максимальное количество термоконтейнеров с готовыми обедами. Определите необходимое суммарное количество термоконтейнеров для ежедневной перевозки готовых обедов в пункты питания из двух цехов.
	import math f =
	open('27.txt') n,
	m = map(int,
	f.readline().split()) v = 5
	#ПОМЕНЯЙ!!!!!!!!!
	a = [0] + [math.ceil(int(x) / v)
	for x in f] print(a) pref = [0]
	* (n + 1) for i in range(1,
	n+1):
	pref[i] = pref[i-1] + a[i]
	print(pref)
	
	res = 0 s = 0 for i in
	range(1,n +1): s =
	pref[min(i+m, n)] -
	pref[max(i-m-1,0)] if s
	> res1: res1 = s
	pun2 = i print(res1,
	pun2)

У концерна по производству пастеризованного молока есть N ферм. Все фермы расположены вдоль некоторого прямолинейного пути и имеют номера, соответствующие расстоянию от нулевой отметки до конкретной фермы. Известно количество литров молока, которое ежедневно получают на каждой ферме.
Концерн планирует открыть молокоперерабатывающий завод при одной из ферм. Молоко на завод с ферм перевозят в бидонах вместимостью 20 литров каждый. Стоимость перевозки молока равна произведению расстояния от фермы до завода на количество перевозимых с данной фермы бидонов с молоком. Общая стоимость перевозки за день равна сумме стоимостей перевозок с каждой из ферм до завода. Место для возведения завода выбрано так, чтобы общая стоимость доставки молока со всех ферм была минимальной.
Определите минимальную общую стоимость доставки молока со всех ферм на завод.
	import math
	k =
	20#ПОМЕНЯЙ!!!!!!!!!!
	f = open('27v03_B.txt') n
	= int(f.readline()) a = []
	for s in f:
	x,y = map(int, s.split())
	a.append([x,
	math.ceil(y/k)])
	ts = 0 for i in range(1, n):
	ts +=
	a[i][1]*abs(a[i][0]a[0][0])
	
	fs = 0 for i in
	range(1, n):
	fs += a[i][1]
	
	bs = 0 mins
	= ts
	#print(ts, fs, bs) for i
	in range(1,n): tr =
	a[i][0] -a[i-1][0] bs
	+= a[i-1][1] ts = ts -
	fs*tr + bs*tr
	fs = fs - a[i][1]
	mins = min(mins, ts)
	#print(ts, fs, bs)
	print(mins)

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 V пробирок на расстояние не более M. Каждый транспортировочный контейнер упаковывается в пункте приёма и вскрывается только в лаборатории. Компания планирует открыть лабораторию в одном из пунктов. Лабораторию расположили в одном из пунктов приёма биоматериалов таким образом, чтобы перевезти как можно больше пробирок (ведь ездить можно на расстояние не более M от текущей точки). Найдите количество контейнеров, которые понадобятся для перевозки максимально возможного количества пробирок.	from math import * f = open('27B_1.txt') n,v,m = map(int, f.readline().split()) a = [] for i in range(n): x,y = map(int, f.readline().split()) a.append([x, y]) a.sort() dl = a[-1][0] dor = [0] * (dl+1) for i in range(len(a)): dor[a[i][0]] = a[i][1] pref = [0] * len(dor) for i in range(1, len(dor)): pref[i] = pref[i-1] + dor[i] res = 0 for i in range(0, len(dor)): if dor[i] != 0: ts = pref[min(i+m,len(dor)-1)] - pref[max(i-m-1,0)]
На кольцевой автодороге с двусторонним движением находится N многоэтажных жилых домов (не более одного дома на каждом километре дороги). Длина кольцевой автодороги равна K км. Нулевой километр и K-й километр находятся в одной точке. Жители домов ежедневно получают почту. Которую доставляют роботы-почтальоны. Почта упакована в доставочные пакеты, каждый из которых вмещает не более 9 кг посылок или писем. Каждый доставочный пакет используется для доставки почты только в один жилой дом, при этом в каждый дом может быть доставлено не более одного пакета с неполной загрузкой. Известно, что заряд аккумулятора робота-почтальона позволяет проходить ему не более M км, заряд аккумулятора для возвращения робота в почтовое отделение не учитывается. Почтовое отделение открыли в одном из домов таким образом, чтобы количество доставляемых пакетов с корреспонденцией было максимальным. В каждом доставочном пакете перевозится почта только для одного дома. Определите необходимое количество доставочных пакетов в этом почтовом отделении.	from math import * f = open('27B_3.txt') n, k, m = map(int, f.readline().split()) = 9# ПОМЕНЯЙ на a = [] for i in range(n): x,y = map(int, f.readline().split()) a.append([x, ceil(y/v)]) dor = [0]*k for i in range(len(a)): dor[(a[i][0])%k] = a[i][1] dor = [0] + dor*2 pref = [0]*len(dor) for i in range(1, len(dor)): pref[i] = pref[i-1] + dor[i] #print(pref) res = 0 for i in range(0, len(dor)): if dor[i] != 0: ts = pref[min(i+m,len(dor)-1)] - pref[max(i-m,0)] res = max(ts, res) print(res)

Сортировка по возрастанию	a.sort()
Сортировка по убыванию	a.sort(reverse = True)

В аэропорту есть камера хранения из К ячеек, которые пронумерованы с 1. Принимаемый багаж кладется в свободную ячейку с минимальным номером.

Известно время, когда пассажиры сдают и забирают багаж (в минутах с начала суток). Ячейка доступна для багажа, начиная со следующей минуты, после
окончания срока хранения. Если свободных ячеек не находится, то багаж не принимается в камеру хранения.

Найдите количество багажа, которое будет сдано в камеры за 24 часа и номер ячейки, в которую сдаст багаж последний пассажир.

Входные данные

В первой строке входного файла находится число К — количество ячеек в камере хранения, во второй строке файла число М — количество пассажиров, сдающих
багаж (натуральное число, не превышающее 1000). Каждая из следующих № строк содержит два натуральных числа, не
превышающих 1440: время сдачи багажа и
время выдачи багажа.

Выходные данные

Программа должна вывести два числа: количество сданных в камеру хранения багажа и номер ячейки, в которую примут багаж у последнего пассажира, который сможет сдать багаж

Типовой пример организации данных:

2
5
60
60
1110
1010
1440

Для указанного примера багаж смогут сдать первый, второй, четвёртый и пятый пассажир. Последний пассажир сдаст свой багаж в ячейку один, так как к этому моменту первая и вторая ячейка будут свободны.

```
f = open('26.txt')
n , m = map(int, f.readline().split()) a
= [] for i in range(m):     x , y =
map(int, f.readline().split())     a +=
[[x,y]] a.sort() d = [0] * n res= 0 for i
in range(len(a)):     for j in
range(len(d)):          if a[i][0] > d[j]:
d[j] = a[i][1]            res += 1
res2 = j + 1
        break            print(res,
res2)
```

НЕЭФФЕКТИВНАЯ ПРОГА (СТАРОЕ ЗАДАНИЕ) Все данные – целые числа (возможно, отрицательные). Требуется найти наибольшую сумму двух результатов измерений, выполненных с интервалом не менее, чем в 7 минут.	f = open('27.txt') n = int(f.readline()) a = [0] * n for i in range(n): a[i] = int(f.readline()) maxi = a[0] + a[7] for i in range(n - 7): for j in range(i + 7, n): if a[i] + a[j] > maxi: maxi = a[i] + a[j] print(maxi)
Имеется набор данных, состоящий из пар положительных целых чисел. Необходимо выбрать из каждой пары ровно одно число так, чтобы сумма n всех выбранных чисел не делилась на 3 и при этом была максимально возможной.	f = open('27-B1.txt') n = int(f.readline()) sum1 = 0 div = 100000 for i in range(n): s = f.readline().split() for j in range(len(s)): s[j] = int(s[j]) sum1 += max(s) if max(s) - min(s) < div and (max(s) - min(s)) % 3 != 0: div = max(s) - min(s) if sum1 % 3 != 0: print(sum1) else: print(sum1 - div)
(ОБЫЧНЫЙ МЕТОД ЧАСТИЧНЫХ СУММ) Набор данных состоит из пар натуральных чисел. Необходимо выбрать из каждой пары ровно одно число так, чтобы сумма всех выбранных чисел делилась на 3 и при этом была максимально возможной.	f = open('27-A.txt') n = int(f.readline()) k = 3 mas = list(map(int, f.readline().split())) for i in range(1, n): x = list(map(int, f.readline().split())) gen = [a + b for a in mas for b in x] mas1 = [0] * k for a in gen: mas1[a%k] = max(a, mas1[a%k]) mas = [a for a in mas1 if a != 0] print(mas)
термоконтейнер используется для доставки только в один пункт питания, при этом в каждый пункт питания может быть доставлено не более одного термоконтейнера с неполной загрузкой. Компания-производитель расположила в двух пунктах питания два цеха для производства готовых обедов так, что из этих цехов в пункты питания ежедневно	pref[max(i-m-1,0)] if s > res: res = s pun1 = i print(res, pun1) for i in range(pun1 - m, pun1 + m + 1): a[i] = 0 pref = [0] * (n + 1) for i in range(1, n+1): pref[i] = pref[i-1] + a[i] print(pref)

Задание №26

Дан набор из N натуральных чисел. Необходимо определить количество пар элементов (ai, aj) этого набора, в которых 1 < i < j < N и произведение элементов кратно 14 .	f = open('27_B.txt') n = int(f.readline()) res, k14, k7, k2, k1 = 0,0,0,0,0 for i in range(n): = int(f.readline()) if x % 14 == 0: res += k14 + k7 + k2 + k1 k14 += 1

Как копировать в эксель в несколько столбиков :
1) Данные – текст по столбцам – с разделителями – далее – пробел – далее – готово. Вставить столбики еще раз 2) Файл – открыть – обзор – !!все файлы!! – открыть – тоже самое **Проверяй для второго ответа есть ли такие числа в файле!**

<u>Задание №27</u> Формулы n*m и n*(n-1) // 2