## Урок №3. Ввод-вывод и базовые переменные

0 сек.

**Задание №1**

Давай представим, что мы пишем бэкенд для сайта ветеринарной клиники. Нам нужно написать программу, которая будет запрашивать у пользователя вид питомца, его возраст и кличку, а потом выведет все эти данные в одно предложение. Например:

Это желторотый питон по кличке "Каа". Возраст: 34 года.

**Решение задания № 1:**

print('Введите вид питомца')

a = input ()

print('Введите кличку питомца')

b = input()

print('Введите возраст питомца')

c = input()

print('Это', a, 'по кличке', b, '. Возраст:', c, '.')

Введите вид питомца

свиноконь

Введите кличку питомца

Иван

Введите возраст питомца

67

Это свиноконь по кличке Иван . Возраст: 67 .

**Задание №2**

А теперь мы с тобой напишем форму ввода ответа на тест по биологии для студентов. Он должен запрашивать по порядку этапы развития человека (проверим твое умение гуглить, что тоже очень важно для программиста. ) и в конце вывести все стадии, разделенные знаком =>, что будет означать постепенный переход от одного к другому. В следующих уроках мы дополним эту форму до полноценного теста, который будет проверять правильность ответов, а пока - начнем с малого. Напоминаем, что разделить эти данные тебе поможет команда sep внутри команды print, например, чтобы разделить переменные знаком + нужно ввести:

print(a1, a2, a3, sep='+')

Подсказка: последняя стадия развития - Homo sapiens sapiens.

**Решение задания № 2:**

print('введите стадии развития человека начиная от самой древней')

a = ''

for i in range(4):

b = input()

a += b

a += ' => '

print('следующая стадия')

c = input()

a += c

print('Ваш вариант: ', a)

**print('Правильный ответ: Australopithec=> Homo Habilis=> Homo Erectus=> Homo Neanderthalensis=> Homo Sapiens Sapiens')**

введите стадии развития человека начиная от самой древней

tyty

следующая стадия

ujkb

следующая стадия

ujhjvhj

следующая стадия

cghcgh

следующая стадия

cgcghc

Ваш вариант: tyty => ujkb => ujhjvhj => cghcgh => cgcghc

Правильный ответ: Australopithec=> Homo Habilis=> Homo Erectus=> Homo Neanderthalensis=> Homo Sapiens Sapiens

## Урок №4. float, int и арифметические операции

**Задание №1**

Пользователь вводит стороны прямоугольника, выведите его площадь и периметр. На вход программе могут подаваться как целые числа, так и вещественные

**Решение задания № 1:**

print('введите стороны прямоугольника, каждый ввод подтвердите клавишей enter')

a = float(input())

b = float(input())

с = a \* b

d = 2 \* (a + b)

print('Параметры прямоугольника: площадь =', с)

print(' периметр =', d)

введите стороны прямоугольника, каждый ввод подтвердите клавишей enter

5

7

Параметры прямоугольника: площадь = 35.0 периметр = 24.0

**Задание №2**

Дано пятизначное целое число. Напишите алгоритм, который возведёт количество десятков в степень количества единиц. Затем умножит это число на количество сотен. И делит получившееся число на разность количества десятков тысяч и количества тысяч

Например, есть число 46275

Необходимо возвести 7 (десятки) в степень 5 (единицы), умножить получившееся число на 2 (сотни), и разделить на разность между 4 (десятки тысяч) и 6 (тысячи) то есть (4-6)

В результате необходимо получить вещественное число. В нашем примере это будет: -16807.0

**Решение задания № 2:**

print('введите пятизначное число, ввод подтвердите клавишей enter')

a = int(input())

b = a//10000

c = (a - b\*10000)//1000

d = (a-b\*10000-c\*1000)//100

h = (a-b\*10000-c\*1000-d\*100)//10

f = a-b\*10000-c\*1000-d\*100-h\*10

g = ((h\*\*f)\*d)/(b-c)

print(g)

введите пятизначное число, ввод подтвердите клавишей enter

21111

1.0

## Урок №5. Логические и условные операторы

**Задание №1**

Пользователь вводит целое число. Выведите его строку-описание вида "отрицательное четное число", "нулевое число", "положительное нечетное число", например, численным описанием числа 190 является строка "положительное четное число". Если число не является четным - выведите сообщение "число не является четным"

**Решение задания № 1:**

print('введите целое число, ввод подтвердите клавишей enter')

a = 'Это '

b = int(input())

if b < 0:

a += 'отрицательное '

elif b == 0:

a += 'нулевое '

else: a += 'положительное '

c = float(int(b/2)-(float(b)/2))

if b != 0:

if c == 0.0:

a += 'четное'

else: a += 'нечетное '

print(a,'число')

введите целое число, ввод подтвердите клавишей enter

-191

Это отрицательное нечетное число

**Задание №2**

Дано слово из маленьких латинских букв. Сколько там согласных и гласных букв? Гласными называют буквы «a», «e», «i», «o», «u».

Для решения задачи создайте переменную и в неё положите слово с помощью input()

А также определите количество каждой из этих гласных букв Если какой-то из перечисленных букв нет - Выведите False

**Решение задания № 2:**

print('введите слово, ввод подтвердите клавишей enter')

b = input()

c = len(b)

o = 0

a = 0

u = 0

e = 0

i = 0

d = 0

for j in range(c):

g = b[j]

if g == 'a':

a += 1

elif g == 'o':

o += 1

elif g == 'u':

u += 1

elif g == 'e':

e += 1

elif g == 'i':

i += 1

else: d+=1

print('в данном слове гласных букв a -',a,' o-',o,' e', e, ' i-', i, ' u-',u)

введите слово, ввод подтвердите клавишей enter

etrtiueoartr

в данном слове гласных букв a - 1 o- 1 e 2 i- 1 u- 1

**Задание №3**

Два инвестора - Майкл и Иван хотят вложиться в стартап. Фаундеры сказали, что минимальная сумма инвестиций - X долларов, больше инвестировать можно сколько угодно. У Майкла A долларов, у Ивана B долларов. Если оба могут вложиться - выведите 2, если только Майкл - Mike, если только Иван - Ivan, если не могут по отдельности, но вместе им хватает - 1, если никто - 0.

**Решение задания № 3:**

print('введите требуемые инвестиции, капитал майка и ивана, каждый ввод подтвердите клавишей enter')

a = int(input())

b = int(input())

c = int(input())

d = b + c

if (b > a) or (c > a) or (b+c) < a:

if b > a:

print('Майкл')

if c > a:

print ('Иван')

if b+c < a:

print('0')

else:

print('-1')

введите требуемые инвестиции, капитал майка и ивана, каждый ввод подтвердите клавишей enter

10

8

7

-1

## Урок №6. Циклы while и for

**Задание №1**

Cначала вводится число N, затем вводится ровно N целых чисел. Подсчитайте, сколько из них равны нулю, и выведите это количество.

**Решение задания № 1:**

print('введите количество целых чисел и сами числа, каждый ввод подтвердите клавишей enter')

a = int(input())

c = 0

for i in range(a):

b = int(input())

if b == 0:

c +=1

print(c)

введите количество целых чисел и сами числа, каждый ввод подтвердите клавишей enter

3

0

9

0

2

**Задание №2**

Вводится натуральное число X. Подсчитайте количество натуральных делителей числа X (включая 1 и само число). x ≤ 2e9 (2 миллиарда)

**Решение задания № 2:**

print('введите натуральное число и ввод подтвердите клавишей enter')

a = int(input())

c = 0

for i in range(1, a+1):

b = float(float(a)/i-int(int(a)/i))

if b == 0.0:

c += 1

print(c)

введите натуральное число и ввод подтвердите клавишей enter

24

8

**Задание №3**

Вводятся целые числа A и B. Гарантируется, что A ≤ B. Выведите все четные числа на заданном отрезке через пробел.

**Решение задания № 3:**

print('введите число A и B,которое больше А, каждый ввод подтвердите клавишей enter')

a = int(input())

b = int(input())

c = ''

for i in range(a, b+1):

d = float(float(i)/2-int(int(i)/2))

if d == 0.0:

c += str(i)+' '

print(c)

введите число A и B,которое больше А, каждый ввод подтвердите клавишей enter

4

8

4 6 8

## Урок №7. Строки

**Задание №1**

На вход подается 1 строка без пробелов. По данной строке определите, является ли она палиндромом (то есть, можно ли прочесть ее наоборот, как, например, слово "шалаш"). Необходимо вывести ”yes”, если строка является палиндромом, и “no” в противном случае.

**Решение задания № 1:**

print('введите строку, ввод подтвердите клавишей enter')

a = input()

c = ''

d = 0

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

c += a[i]

if c == a:

print('yes')

введите строку, ввод подтвердите клавишей enter

шалаш

yes

**Задание №2**

Дана строка, длина которой не превосходит 1000. Вам требуется преобразовать все идущие подряд пробелы в один. Выведите измененную строку.

**Решение задания № 2:**

print('введите строку, ввод подтвердите клавишей enter')

a = input()

c = ''

d = 0

for i in range(len(a)):

if a[i] == ' ':

d += 1

if d < 2:

c += a[i]

else:

d = 0

c +=a[i]

print(c)

введите строку, ввод подтвердите клавишей enter

кто кот крот рот от

кто кот крот рот от

## Урок №8. Списки

**Задание №1**

В первой строке вводится число N. Далее в N строк вводится N чисел (1 ≤ N ≤ 10000), по одному числу на строке. Все числа по модулю не превышают 10e5. Переверните массив чисел. Выведите N чисел - перевернутый массив.

**Решение задания № 1:**

print('введите число элементов и элементы, каждый ввод подтвердите клавишей enter')

a = int(input())

b = []

d = []

for i in range(a):

c = int(input())

b.append(c)

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

d.append(b[i])

print(d)

введите число элементов и элементы, каждый ввод подтвердите клавишей enter

5

6

7

8

9

0

[0, 9, 8, 7, 6]

**Задание №2**

В первую строчку вводится число N (1 ≤ N ≤ 100 000). В следующую строку через пробел вводятся N чисел (1 ≤ Ai ≤ 10e9). Вам требуется написать метод, который получает на вход массив и изменяет его таким образом, чтобы на первом месте стоял последний элемент, на втором - первый, на третьем - второй и т. д. Выведите N чисел - измененный массив.

**Решение задания № 2:**

print('введите через проблел элементы массива, ввод подтвердите клавишей enter')

a = list(map(int, input().split()))

d = []

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

d.append(a[i])

print(d)

введите через проблел элементы массива, ввод подтвердите клавишей enter

5 6 7 8

[8, 7, 6, 5]

**Задание №3**

На берегу реки стояли n рыбаков, все они хотели перебраться на другой берег. Одна лодка может выдержать не более m килограмм, при этом в лодку помещается не более 2 человек. Определите, какое минимальное число лодок нужно, чтобы перевезти на другой берег всех рыбаков В первую строку вводится число m (1 ≤ m ≤ 10e6) - максимальная масса, которую может выдержать одна лодка. Во вторую строку вводится число n (1 ≤ n ≤ 100) - количество рыбаков. В следующие N строк вводится по одному числу Ai (1 ≤ Ai ≤ m) - вес каждого путешественника. Программа должна вывести одно число - минимальное количество лодок, необходимое для переправки всех рыбаков на противоположный берег.

**Решение задания № 3:**

print('введите грузоподьемность лодки и число рыбаков,массы рыбаков после каждого ввода нажмите enter')

c = int(input())

b = int(input())

a = []

e = 0

for i in range(b):

d=int(input())

if d>c:

print('этот рыбак слишком тяжелый, он никуда не уплывет')

else:

a.append(d)

a.sort()

while len(a)>0 :

if (a[0]+a[len(a)-1]) <= c:

a.pop(0)

a.pop(len(a)-1)

else:

a.pop(0)

e += 1

print(e)

введите грузоподьемность лодки и число рыбаков,массы рыбаков после каждого ввода нажмите enter

100

5

89

140

этот рыбак слишком тяжелый, он никуда не уплывет

54

45

10

2

## Урок №9. Множества

**Задание №1**

В первую строку вводится число N – количество чисел (1 ≤ N ≤ 100000). Во вторую строку вводится через пробел N чисел, каждое не превышает 2\*10e9 по модулю. Требуется выяснить, сколько среди этих чисел различных. Выведите число, равное количеству различных чисел среди данных.

**Решение задания № 1:**

print('введите число эементов нажмите enter, затем через проблел элементы массива, нажмите enter')

c = int(input())

a = list(map(int, input().split()))

b = set(a)

print(len(b))

введите число эементов нажмите enter, затем через проблел элементы массива, нажмите enter

6

6 7 8 9 6 7

4

**Задание №2**

Вводятся два списка чисел, которые могут содержать до 100000 чисел каждый. Все числа каждого списка находятся на отдельной строке. Выведите, сколько чисел содержится одновременно как в первом списке, так и во втором.

**Решение задания № 2:**

# т. к. задача из раздела множетства наверное имеется ввиду не суммарное количство чисел число уникальных на 2 списка

#иначе просто это было бы как сумма длин

print('введите количество элеметов списка 1 и элементы, после каждого ввода нажмите enter')

a1 = []

c = int(input())

for i in range(c):

d=int(input())

a1.append(d)

print('введите количество элеметов списка 2 и элементы, после каждого ввода нажмите enter')

a2 = []

c = int(input())

for i in range(c):

d = int(input())

a2.append(d)

e = len(set(a1+a2))

print(e)

введите количество элеметов списка 1 и элементы, после каждого ввода нажмите enter

2

3

6

введите количество элеметов списка 2 и элементы, после каждого ввода нажмите enter

3

3

5

7

4

**Задание №3**

Во входную строку водится последовательность чисел через пробел. Для каждого числа выведите слово ”YES” (в отдельной строке), если это число ранее встречалось в последовательности или ”NO”, если не встречалось.

**Решение задания № 3:**

print('введите через пробел элеметы списка после последнего ввода нажмите enter')

a = list(map(int, input().split()))

b = set()

e = ''

for i in range(len(a)):

c=len(b)

b.add(a[i])

c2 = len(b)

if c2 >c:

e += str(a[i])+' - no, '

else:

e += str(a[i])+' - yes, '

print(e)

введите через пробел элеметы списка после последнего ввода нажмите

enter 4 5 6 7 5 6 8 9

4 - no, 5 - no, 6 - no, 7 - no, 5 - yes, 6 - yes, 8 - no, 9 - no,

## Урок №10. Словари

**Задание №1**

Ранее вы выполняли задание связанное с ветеринарной клиникой. В той задаче вам предстояло вывести информацию о питомце на экран. Сейчас вам необходимо создать словарь pets = {}

Примерный вид будет следующим:

pets = {

  "Имя питомца": {

    'Вид питомца': # придумайте каким образом сюда внести информацию,

    'Возраст питомца': # придумайте каким образом сюда внести информацию,

    'Имя владельца': # придумайте каким образом сюда внести информацию

  }

}

У вас должен получиться словарь, с ещё одним словарём внутри. То есть, есть словарь pets. Он в себе хранит ещё один словарь, который обозначается именем питомца. Имя питомца также нужно каким-то образом вносить туда.

Задача не будет считаться выполненной, если вы заходите сразу внести информацию, не прибегая в функции input()

Например:

pets = {

  "Мухтар": {

    "Вид питомца": "Собака",

    "Возраст питомца": 9,

    "Имя владельца": "Павел"

  }

}

Так должен будет выглядеть результируюший словарь, но первоначальный его вид - пустой. Его необходимо заполнить пользовательским вводом через консоль с помощью функции input(), а не вписать значения уже в самом коде.

Возраст питомца должен быть типа int Всё остальное - строки

Так как возраст питомца указывается типом int. Необходимо, в соответствии с указанным возрастом выводит год, года или лет. Например:

Его возраст: 24 года  
Его возраст: 21 год  
Его возраст: 19 лет

И теперь осталось только получить всю информацию о питомце в виде строки, как из задания по Урок №3. Ввод-вывод и базовые переменные. Задание №1, но с небольшими изменениями. Для получения информации необходимо воспользоваться методами словаря keys() и values():  
Это желторотый питон по кличке "Каа". Возраст питомца: 19 лет. Имя владельца: Саша

**Решение задания № 1:**

i = 0

d = dict()

k = 0

while i == 0:

print('если нужно добавить питомца введите 1 если вывести данные об имеющимся введите 2 нажмите enter')

a = int(input())

if a == 1:

k += 1

print('введите имя питомца')

b = input()

print('введите вид питомца')

c = input()

print('введите возраст питомца')

e = int(input())

h = str(e)

u1 = (len(h)-1)

u2 = h[u1]

u = int(u2)

if e == 1 or (e > 20 and u == 1):

v = 'год'

if (e == 2 or e == 3) or (e > 20 and (u == 2 or u == 3)):

v = 'года'

else:

v = 'лет'

print('введите имя владельца')

g = input()

d[b] = {'вид питомца -': c,'возраст питомца-': h + v, 'Имя владельца -': g }

if a == 2:

if k < 1:

print('в базе пока нет питомцев')

else:

print('введите имя питомца')

b = input()

f = d[b]

w = f['вид питомца -']

w2 = f['возраст питомца-']

w3 = f['Имя владельца -']

print('Это ', w, 'по кличке', b, '. Возраст: ', w2, '. Владелец -', w3)

print('введите 1 чтобы продолжить и 2 чтобы выйти из программы')

f = int(input())

if f == 1:

i = 0

if f ==2:

i = 1

если нужно добавить питомца введите 1 если вывести данные об имеющимся введите 2 нажмите enter

1 введите имя питомца

Kaa

введите вид питомца

желторотый питон

введите возраст питомца

23

введите имя владельца

Иван

введите 1 чтобы продолжить и 2 чтобы выйти из программы

1

если нужно добавить питомца введите 1 если вывести данные об имеющимся введите 2 нажмите enter

2

введите имя питомца

Kaa

Это желторотый питон по кличке Kaa . Возраст: 23года . Владелец - Иван

введите 1 чтобы продолжить и 2 чтобы выйти из программы

2

**Задание №2**

С помощью цикла создайте словарь, в котором ключи будут, например от числа 10, до -5 (включительно). А значениями этих ключей будут сами эти числа возведённые в степени равных этим числам

Например:

my\_dict = {

  10: 10000000000,

  9: 387420489,

  # и так далее ...

  -5: -0.00032

}

**Решение задания № 2:**

d = dict()

for i in range(10, -5, -1):

d[i] = i\*\*i

print(d)

{10: 10000000000, 9: 387420489, 8: 16777216, 7: 823543, 6: 46656, 5: 3125, 4: 256, 3: 27, 2: 4, 1: 1, 0: 1, -1: -1.0, -2: 0.25, -3: -0.037037037037037035, -4: 0.00390625}

## Урок №11. Функции

**Задание №1**

* Создайте функцию, которая принимает в качестве параметра - натуральное целое число.
* Данная функция находит факториал полученного числа

Например, факториал числа 3 это число 6.

* Теперь создайте список факториалов чисел от получившегося ранее факториала 6, до 1.

В итоге, на вход программа получает например число 3, возвращает число 6 (факториал числа 3) и вам нужно сделать список из факториалов числа 6 в убывающем порядке. Находим факториал числа 6 - это 720, затем от числа 5 - это 120 и так далее вплоть до 1

То есть, результирующий список будет выглядеть в нашем примере так:  
[720, 120, 24, 6, 2, 1]

**Решение задания № 1:**

print('введите число и нажмите enter')

a = int(input())

c = []

def facktorial(b):

g = 1

for i in range(1, b+1, 1):

g = g \* i

return (g)

for j in range(a,0,-1):

b = facktorial(j)

c.append(b)

print(c)

введите число и нажмите enter

6

[720, 120, 24, 6, 2, 1]

**Задание №2**

В Урок №10. Задание №1 вы создавали словарь с информацией о питомце. Теперь нам нужна "настоящая" база данных для ветеринарной клиники.

Подробный требуемый функционал будет ниже. Пока что справка:

* Создайте функцию create
* Создайте функцию read
* Создайте функцию update
* Создайте функцию delete
* Используйте словарь pets, который будет предоставлен ниже, либо создайте свой аналогичный

**Функция**create**:**

Данная функция будет создавать новую запись с информацией о питомце и добавлять эту информацию в наш словарь pets

**Функция**read

Данная функция будет отображать информацию о запрашиваемом питомце в виде:  
Это желторотый питон по кличке "Каа". Возраст питомца: 19 лет. Имя владельца: Саша

**Функция**update

Данная функция будет обновлять информацию об указанном питомце

**Функция**delete

Данная функция будет удалять запись о существующем питомце

Структруа результирующего словаря pets будет как и в Урок №10. Задание №1, но с небольшим видоизменением:

**Словарь pets**

pets = {

    1:

        {

            "Мухтар": {

                "Вид питомца": "Собака",

                "Возраст питомца": 9,

                "Имя владельца": "Павел"

            },

        },

    2:

        {

            "Каа": {

                "Вид питомца": "желторотый питон",

                "Возраст питомца": 19,

                "Имя владельца": "Саша"

            },

        },

    # и так далее

}

Здесь, 1 и 2 - это идентификаторы наших питомцев. Это уникальные ключи, по которым мы сможем обращаться к нашим записям в "базе данных"

Суть будущей программы будет заключаться в следующем:

* Программа будет работать с помощью цикла while с условием command != 'stop', то есть до тех пор, пока на предложение ввести команду, пользователь не введёт слово stop
* Перед взаимодействием с "базой данных" запрашивается одна из команд в качестве пользовательского ввода. Пусть это будет переменная command
* Функция create должна добавлять новую информацию таким образом, чтобы идентификатор увеличивался на единицу. Чтобы у вас была возможность получать последний ключ в словаре воспользуйтесь импортом модуля collections. В начале вашей программы пропишите строчку: import collection, а в функции create в первых строках пропишите следующий код:

def create():

  last = collections.deque(pets, maxlen=1)[0]

last в данном случае и будет число последнего ключа (или в нашей логике - идентификатора записи). Именно его и необходимо будет увеличивать на единицу при добавлении следующей записи.

Как вам уже известно - суть функций заключается в том, чтобы использовать один и тот же код в нескольких местах. В данной задаче вам предстоит получать информацию о питомце несколько раз. Чтобы не повторяться в коде, вам нужно создать такие функции

get\_pet(ID):

def get\_pet(ID):

  # функция, с помощью которой вы получите информацию о питомце в виде словаря

  # сделайте проверку, если питомца с таким ID нету в нашей "базе данных"

  # верните в этом случае False

  # а если питомец всё же есть в "базе данных" - верните информацию о нём

  # выглядеть это может примерно так:

  return pets[ID] if ID in pets.keys() else False

get\_suffix(age):

def get\_suffix(age):

  # функция, с помощью которой можно получить суффикс

  # 'год', 'года', 'лет'

  # реализацию этой функции вам предстоит придумать самостоятельно

  # функция будет возвращать соответствующую строку

  return

pets\_list():

def pets\_list():

  # Эта функция будет создана для удобства отображения всего списка питомцев

  # Информацию по каждому питомцу можно вывести с помощью цикла for

Обратите внимание, если ID не существует в словаре с питомцами - будет возникать ошибка. Вам можно от неё избавиться, если правильно составить проверочное условие. Здесь не потребуется использовать такие конструкции, как try, except, чтобы обработать возникшую ошибку

**Решение задания № 2:**

i = 0

d = dict()

k = 0

def add\_pet(k):# добавляем питомца в словарь

k += 1

print('Ваш ID: ', k)

print('введите имя питомца')

b = input()

print('введите вид питомца')

c = input()

print('введите возраст питомца')

e = int(input())

h = str(e)

u1 = (len(h)-1)

u2 = h[u1]

u = int(u2)

if e == 1 or (e > 20 and u == 1):

v = 'год'

if (e == 2 or e == 3) or (e > 20 and (u == 2 or u == 3)):

v = 'года'

else:

v = 'лет'

print('введите имя владельца')

g = input()

d[k] = {'Кличка питомца': b,'вид питомца -': c,'возраст питомца-': h + v, 'Имя владельца -': g }

return(k,d)

def get\_pet():# запрашиваем данные о питомце

print('введите ID питомца')

b = int(input())

if b in d.keys():

f = d[b]

w0 = f['Кличка питомца']

w = f['вид питомца -']

w2 = f['возраст питомца-']

w3 = f['Имя владельца -']

print('Это ', w, 'по кличке', w0, '. Возраст: ', w2, '. Владелец -', w3)

else:

print('нет такого ID в базе')

return()

def upd\_pet(): # обновляем данные о питомце

print('Ввведите ID питомца')

p =int(input())

f = d[p]

w0 = f['Кличка питомца']

w = f['вид питомца -']

w2 = f['возраст питомца-']

w3 = f['Имя владельца -']

print('Это ', w, 'по кличке', w0, '. Возраст: ', w2, '. Владелец -', w3)

print('введите новое имя питомца')

b = input()

print('введите новый вид питомца')

c = input()

print('введите новый возраст питомца')

e = int(input())

h = str(e)

u1 = (len(h)-1)

u2 = h[u1]

u = int(u2)

if e == 1 or (e > 20 and u == 1):

v = 'год'

if (e == 2 or e == 3) or (e > 20 and (u == 2 or u == 3)):

v = 'года'

else:

v = 'лет'

print('введите новое имя владельца')

g = input()

d[p] = {'Кличка питомца': b,'вид питомца -': c,'возраст питомца-': h + v, 'Имя владельца -': g }

return(d)

def del\_pet():

print('Введите ID питомца')

y = int(input())

f = d[y]

w0 = f['Кличка питомца']

w = f['вид питомца -']

w2 = f['возраст питомца-']

w3 = f['Имя владельца -']

print('Это ', w, 'по кличке', w0, '. Возраст: ', w2, '. Владелец -', w3)

print('чтобы подтвердить удаление введите Y, чтобы отменить введите N, восстановить данные будет невозможно')

dd = input()

del d[y]

print('Pets data deleted')

while i == 0:

print('если нужно добавить питомца введите 1 вывести данные 2, редактировать - 3 удалить -4 и нажмите enter')

a = int(input())

if a == 1:

k=add\_pet(k)

print(k)

if a == 2:

get\_pet()

if a == 3:

upd\_pet()

if a == 4:

del\_pet()

print('введите 1 чтобы продолжить и 2 чтобы выйти из программы')

f = int(input())

if f == 1:

i = 0

if f ==2:

i = 1

если нужно добавить питомца введите 1 вывести данные 2, редактировать - 3 удалить -4 и нажмите enter

1

Ваш ID: 1

введите имя питомца

Bobik

введите вид питомца

dog

введите возраст питомца

15

введите имя владельца

Ivan Pupkin

(1, {1: {'Кличка питомца': 'Bobik', 'вид питомца -': 'dog', 'возраст питомца-': '15лет', 'Имя владельца -': 'Ivan Pupkin'}})

введите 1 чтобы продолжить и 2 чтобы выйти из программы

1

если нужно добавить питомца введите 1 вывести данные 2, редактировать - 3 удалить -4 и нажмите enter

2

введите ID питомца

1

Это dog по кличке Bobik . Возраст: 15лет . Владелец - Ivan Pupkin

введите 1 чтобы продолжить и 2 чтобы выйти из программы

1

если нужно добавить питомца введите 1 вывести данные 2, редактировать - 3 удалить -4 и нажмите enter

3

Ввведите ID питомца

1

Это dog по кличке Bobik . Возраст: 15лет . Владелец - Ivan Pupkin

введите новое имя питомца

Bobik

введите новый вид питомца

dog

введите новый возраст питомца

16

введите новое имя владельца

Ivan Pupkin

введите 1 чтобы продолжить и 2 чтобы выйти из программы

1

если нужно добавить питомца введите 1 вывести данные 2, редактировать - 3 удалить -4 и нажмите enter

4

Введите ID питомца

1

Это dog по кличке Bobik . Возраст: 16лет . Владелец - Ivan Pupkin

чтобы подтвердить удаление введите Y, чтобы отменить введите N, восстановить данные будет невозможно

Y

Pets data deleted

введите 1 чтобы продолжить и 2 чтобы выйти из программы

1

если нужно добавить питомца введите 1 вывести данные 2, редактировать - 3 удалить -4 и нажмите enter

2

введите ID питомца

1

нет такого ID в базе

введите 1 чтобы продолжить и 2 чтобы выйти из программы

2

## Урок №13. Двумерные списки

**Задание №1**

С помощью цикла создайте матрицу вида 10x10 (пример):

[[0, -2, -5, -5, 0, -4, -1, -23, -11, -20],

[-4, -6, 16, 9, -8, 2, 10, -32, -18, -13],

[21, 7, -6, -3, 34, -7, -23, 25, 27, 38],

[7, 29, 38, 19, 55, 48, 46, 49, -8, -46],

[80, -3, 76, 46, 72, -33, 14, 0, 71, -20],

[82, 95, 23, -6, 1, 70, 50, 16, 25, 7],

[52, 36, -25, -35, 18, -36, 93, 11, -44, 9],

[114, 108, 33, 61, -4, 131, 34, -26, 47, -17],

[37, 56, 135, -12, 34, 67, -17, 14, 93, -67],

[-4, 23, 175, -44, 30, -38, 89, 159, 28, 178]]

И ещё одну - такой же размерности. Числа в матрице выше привидены в качестве примера (одна из генераций)

Итого у вас должно получиться сперва две матрица одинаковой размерности

И теперь вам нужно сложить эти две матрицы в третью. Формулу сложения матриц, вы можете найти в интернете, либо посетив этот [ресурс](https://studwork.org/spravochnik/matematika/matricy/slojenie-matric)

Итак. Вы должны создать две матрицы

Чтобы заполнить матрицы различными значениями - воспользуйтесь модулем random

Для подключения модуля random в код используйте строчку, которую нужно поместить в самое начало вашего скрипта

import random

**Для справки:**  
[Random. Официальная документация по модулю (англ.яз)](https://docs.python.org/3/library/random.html)  
[Random. Информация на русском языке](https://pythonworld.ru/moduli/modul-random.html)

Для начала вы можете просто создать переменные matrix\_1 и matrix\_2

matrix\_1 = [[0, -2, -1, -6, -6, 0, -9, -8, -30, -9],

[5, 12, 4, -16, -4, -9, -16, -15, 1, -26],

[13, 39, 14, 23, -4, 40, 32, 6, -8, 23],

[13, -8, 34, 49, 30, 18, 47, 11, -24, 11],

[21, 73, 71, 61, -1, 79, -34, 22, 69, 67],

[75, 25, 25, 39, 100, -12, -21, 81, -10, 87],

[81, 63, 102, 104, 53, -44, 71, -36, -36, -9],

[7, 98, 26, -3, 128, 94, 18, -26, 14, 21],

[65, 128, 80, 124, 27, -32, 73, 59, 19, 34],

[43, 111, 38, 149, 5, 112, 79, 53, 15, 92]]

matrix\_2 = [[0, 4, 6, 11, 15, 6, 9, 26, 15, 21],

[-5, 4, -15, -9, -4, 2, -8, 19, -4, -1],

[-2, -39, -19, 14, 22, 5, -34, 15, 16, -9],

[-22, -52, 11, -11, -3, 16, -11, -6, -32, -2],

[-61, -47, -5, -58, 16, -13, 28, -36, -64, 2],

[-29, 23, 19, 2, -14, -87, 7, -88, 39, 7],

[-6, 18, -97, 26, -64, 0, -72, -34, -68, -92],

[-120, -117, -72, -129, -139, 16, -61, 36, -137, -29],

[-112, -83, 7, -119, -132, -129, -143, -154, -23, -34],

[32, -67, -75, -92, 15, -163, 18, 31, -162, -16]]

И теперь нужно придумать, как их сложить, чтобы получить matrix\_3:

matrix\_3 = [[0, 2, 5, 5, 9, 6, 0, 18, -15, 12],

[0, 16, -11, -25, -8, -7, -24, 4, -3, -27],

[11, 0, -5, 37, 18, 45, -2, 21, 8, 14],

[-9, -60, 45, 38, 27, 34, 36, 5, -56, 9],

[-40, 26, 66, 3, 15, 66, -6, -14, 5, 69],

[46, 48, 44, 41, 86, -99, -14, -7, 29, 94],

[75, 81, 5, 130, -11, -44, -1, -70, -104, -101],

[-113, -19, -46, -132, -11, 110, -43, 10, -123, -8],

[-47, 45, 87, 5, -105, -161, -70, -95, -4, 0],

[75, 44, -37, 57, 20, -51, 97, 84, -147, 76]]

Потренируйтесь сперва в сложении матриц, а затем попробуйте с помощью модуля random сгенерировать матрицы подобных размеров и уже на них применить навыки сложения matrix\_1 и matrix\_2

Задание считается выполненным, если вы напишите алгоритм, который будет уметь как складывать матрицы, так и генерировать матрица различных размерностей. Будь то матрицы 10х10 или 4х3

**Решение задания № 1:**

import random

a = random.randint(1, 10)

b = random.randint(1, 10)

c2 = []

e2 = []

c1 = []

e1 = []

c = []

e = []

for i in range(a):

for j in range(b):

d = random.randint(0, 9)

e.append(d)

c.append(e)

e =[]

for i in range(a):

print(c[i])

print()

for i in range(a):

for j in range(b):

d = random.randint(0, 9)

e1.append(d)

c1.append(e1)

e1 =[]

for i in range(a):

print(c1[i])

print()

for i in range(a):

for j in range(b):

d = 0

e2.append(d)

c2.append(e2)

e2 =[]

for i in range(a):

for j in range(b):

c2[i][j]=c[i][j]+c1[i][j]

for i in range(a):

print(c2[i])

print()

[8, 8, 0]

[4, 1, 6]

[2, 6, 8]

[7, 7, 5]

[0, 4, 1]

[4, 1, 9]

[2, 3, 1]

[9, 8, 7]

[4, 1, 0]

[8, 7, 1]

[8, 2, 6]

[9, 4, 9]

[7, 9, 8]

[7, 4, 6]

[9, 1, 6]

[7, 0, 2]

[5, 0, 6]

[9, 8, 3]

[9, 5, 0]

[2, 0, 7]

[16, 10, 6]

[13, 5, 15]

[9, 15, 16]

[14, 11, 11]

[9, 5, 7]

[11, 1, 11]

[7, 3, 7]

[18, 16, 10]

[13, 6, 0]

[10, 7, 8]

## Урок №14. Рекурсия

**Задание №1**

Есть последовательность

my\_list = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]

Напишите рекурсивную функцию, которая выведет все элементы от первого до последнего и в конце отобразит сообщение Конец списка, если выводить больше нечего. Циклы использовать запрещено

**Решение задания № 1:**

my\_list = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]

i=0

def rec(i):

print(my\_list[i])

i += 1

if i<len(my\_list):

rec(i)

else:

print('конец')

rec(i)

0

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

конец

## Урок №15. ООП

**Задание №1**

Есть родительский класс:

[2]

Создайте объект **Autobus**, который унаследует все переменные и методы родительского класса **Transport** и выведете его.

Ожидаемый результат вывода:

**Название автомобиля: Renaul Logan Скорость: 180 Пробег: 12**

**Решение:**

class Transport:

def \_\_init\_\_(self, name, max\_speed, mileage):

self.name = name

self.max\_speed = max\_speed

self.mileage = mileage

Autobus = Transport('Reno Logan',120,12)

print('Наименование автомобиля:',Autobus.name, 'Максимальная скорость:',Autobus.max\_speed,'Пробег:', Autobus.mileage)

Наименование автомобиля: Reno Logan Максимальная скорость: 120 Пробег: 12

**Задание №2**

Создайте класс **Autobus**, который наследуется от класса **Transport**.

Дайте аргументу **Autobus.seating\_capacity()** значение по умолчанию **50**.

Используйте переопределение метода.

Используйте следующий код для родительского класса транспортного средства:

class Transport:

def \_\_init\_\_(self, name, max\_speed, mileage):

self.name = name

self.max\_speed = max\_speed

self.mileage = mileage

def seating\_capacity(self, capacity):

return f"Вместимость одного автобуса {self.name} {capacity} пассажиров"

Ожидаемый результат вывода:

**Вместимость одного автобуса Renaul Logan: 50 пассажиров**

**Решение:**

class Transport:

def \_\_init\_\_(self, name, max\_speed, mileage):

self.name = name

self.max\_speed = max\_speed

self.mileage = mileage

def seating\_capacity(self, capacity):

return f"Вместимость одного автобуса {self.name} {capacity} пассажиров"

class Autobus(Transport):

def seating\_capacity(self, capacity):

return f"Вместимость одного автобуса {self.name} : {capacity} пассажиров"

Logan = Autobus('Reno Logan',120,12)

pas = Logan.seating\_capacity(50)

print(pas)

Вместимость одного автобуса Reno Logan : 50 пассажиров

Урок №16. Классы и объекты

**Задание №1**

Создайте класс Касса, который хранит текущее количество денег в кассе, у него есть методы:

* top\_up(X) - пополнить на X
* count\_1000() - выводит сколько целых тысяч осталось в кассе
* take\_away(X) - забрать X из кассы, либо выкинуть ошибку, что не достаточно денег

class Kassa(object):

x\_money = 1000

def \_\_init\_\_(self, x\_money):

self.x\_money = int(x\_money)

def top\_up(self, x):

y= self.x\_money + x

self.x\_money = y

return(y)

def count\_1000(self):

a = self.x\_money//1000

return(a)

def take\_away(self,x):

y= self.x\_money - x

if y < 0:

return('в кассе не достаточно денег')

else:

self.x\_money = y

return(y)

def inform(self):

return self.x\_money

a1 = Kassa(8900)

print(a1.inform())

print(a1.top\_up(1200))

print(a1.count\_1000())

print(a1.take\_away(10800))

8900

10100

10

в кассе не достаточно денег

**Задание №2**

Создайте класс Черепашка, который хранит позиции x и y черепашки, а также s - количество клеточек, на которое она перемещается за ход

у этого класс есть методы:

* go\_up() - увеличивает y на s
* go\_down() - уменьшает y на s
* go\_left() - уменьшает x на s
* go\_right() - увеличивает y на s
* evolve() - увеличивает s на 1
* degrade() - уменьшает s на 1 или выкидывает ошибку, когда s может стать ≤ 0
* count\_moves(x2, y2) - возвращает минимальное количество действий, за которое черепашка сможет добраться до x2 y2 от текущей позиции

import random

class Turtle(object):

x = 0

y = 0

s = 1

def \_\_init\_\_(self, x, y, s):

self.x = x

self.y = y

self.s = s

def go\_up(self):# - увеличивает y на s

y = self.y + self.s

self.y = y

return(y)

def go\_down(self): # - уменьшает y на s

y = self.y - self.s

self.y = y

return(y)

def go\_left(self): #- уменьшает x на s

x = self.x - self.s

self.x = x

return(x)

def go\_right(self): # - увеличивает x на s

x = self.x + self.s

self.x = x

return(x)

def evolve(self): # - увеличивает s на 1

s = self.s + 1

self.s = s

return (s)

def degrade(self): # - уменьшает s на 1 или выкидывает ошибку, когда s может стать ≤ 0

s = self.s - 1

if self.s == 0:

print('скорость персонажа не может быть равна нулю')

s += 1

self.s = s

return (s)

def count\_moves(self,x2, y2): # - возвращает минимальное количество действий, за которое черепашка сможет добраться до x2 y2 от текущей позиции

way = int((abs(x2-self.x))/self.s +(abs(y2-self.y))/self.s)

return(way)

Rafael = Turtle(0, 0, 1)

n = random.randint(4,15)

for i in range(n):

kind = random.randint(1,6)

if kind == 1:

Rafael.go\_up()

if kind == 2:

Rafael.go\_down()

if kind == 3:

Rafael.go\_left()

if kind == 4:

Rafael.go\_right()

if kind == 5:

Rafael.evolve()

if kind == 6:

Rafael.degrade()

x2 = random.randint(5, 10)

y2 = random.randint(5, 10)

print(Rafael.count\_moves(x2,y2),'ходов')

2 ходов

## Урок №17. О-нотация

**Задание №1** (ответ дается текстом под кодом)

Какая сложность у этого алгоритма?

n = int(input())

m = int(input())

cnt = 0

for i in range(n):

for j in range(m):

cnt += 1

print(cnt)

O(n+m)

**Задание №2** (ответ дается текстом под кодом)

Какая сложность у этого алгоритма?

n = int(input())

ans = n \* n + 100

ans -= 200

ans \*= ans

if (ans < 1000000):

ans \*= 2

O(n)

**Задание №3** (ответ дается текстом под кодом)

Какая сложность у этого алгоритма?

n = int(input())

m = int(input())

k = 1000

for i in range(n):

for j in range(m):

k += 100

for i in range(m):

for j in range(m):

k -= 100

O(n+m+m^2)

Gepshmrjv

import random;

n=10

arr = list()

for i in range(n):

number = random.randint(1,100)

arr.append(number)

print(arr)

count2 = 0

for i1 in range(n):

count = 0;

for i in range(n-1):

left = arr[i]

right = arr[i+1]

if left<right:

count = 1;

arr[i]= right

arr[i+1] = left

count2 += 1

if count == 0:

break

print(arr)

print(count2)

import random;

# ограничение счета - пределы a и b для движения вправо и в лево соответственно;

#при переборе в право самый маленький элемент переместится вправо и далее проверять эту позицию

#безсмысленно, аналогично с наибольшим элементом при сравнении влево. Каждый проход в обе стороны минус 2 позиции проверки

#В начале цикла счетчик всегда 0, при замене элементов=>1; если замен за цикл нет и счетчик в 0, окончание сортировки

n=10

arr = list()

for i in range(n):

number = random.randint(1,100)

arr.append(number)

print(arr)

count2 = 0

a = 0

b = n-1

for i1 in range(n):

count = 0;

for i in range(a, b):

left = arr[i]

right = arr[i+1]

if left<right:

count = 1;

arr[i]= right

arr[i+1] = left

print(arr)

b -= 1;

for k in range(b, a, -1):

left = arr[k-1]

right = arr[k]

if left<right:

count = 1;

arr[k-1]= right

arr[k] = left

count2 += 1

if count == 0:

break

a +=1

print(arr)

print(count2)