**Функции, операторы и методы Python**

**Математические действия:**

\* умножение

\*\* возведение в степень

/ деление

// целочисленное деление

% остаток от целочисленного деления

a += b a = a + b

**Операторы сравнения:**

> , >= больше, больше равно

< , <= меньше, меньше равно

== , != равно, не равно

**Строки и переменные: *строки неизменны***

input() ввести строку (тип str)

print() вывести строку, переменную

[0:2:2] срез строки [с какого : до какого : с шагом]

[-1] последний / в обратном порядке

type() вывести тип данных

str строчные

int целые числа

float дробные числа

bool логиче ское значение True/False

dict словарь

int() преобразовать строку в число

str() преобразовать число в строку

len() кол-во символов в строке

ord() код символа строки

\n символ переноса строки

.upper() перевести строку в верхний регистр

.lower() перевести строку в нижний регистр

.title() перевести первый символ каждого слова в верхний регистр

.count(‘…’) кол-во повторов символов ‘…’ в строке

.find() найти первое вхождение подстроки в строке (-1 не найдено)

.rfind() найти первое вхождение подстроки в строке справа налево

.index() возвращает индекс первого вхождения элемента в списке

.replace() заменить в строке, символ (старый, новый)

.isalpha() проварить что в строке только буквы

.isdigit() проверить что в стоке только цифры

.isnumeric() проверить что в стоке только цифры

.islower() проверить что в стоке строчные буквы

.isupper() проверить что в стоке прописные буквы

.isalnum() проверить что в стоке только цифры и буквы

.isspace() проверить что в стоке только пробелы

.endswith() проверить окончание строки

.rjust(n, ‘…’) дополнить строку до длины N символом ‘…’ слева

.ljust(n, ‘…’) дополнить строку до длины N символом ‘…’ справа

.strip() удаляет пробелы/символы переноса в начале и конце строки

.strip() удаляет пробелы/символы переноса в начале строки

.rstrip() удаляет пробелы/символы переноса в конце строки

.split(‘ ‘) разделить строку символом ‘ ‘ ,получить список из строк

(‘, ‘).join() получить строку из списка, удалить символы ‘, ’ в строке

.join(reversed()) перевернуть строку в обратную сторону

**Спецсимволы:**

\n перенос строки

\t табуляция

\a звуковой сигнал

\b эмуляция backspace

\\ обратный слеш и «экранирование» спецсимволов

r’…pro\tex\next…' raw строка (сырая строка) - спецсимволы не работают

f’…{} … {} …' f строка - можно вставлять в текст переменные)

**Списки и кортежи: *списки можно редактировать***

[…] списки, очередность списка [] начинается с 0

[0:2:2] срез списка, [с какого : до какого : с шагом}

[::-1] срез списка, в обратном порядке

len() кол-во элементов в списке

max() нахождение максимального значения элементов

min() нахождение минимального значения элементов

sum() сумма элементов списка

sorted() сортирует список (возвращает новый список)

sorted(список, reversed=True) сортирует в обратном порядке

.append() добавить элемент в конец писка

.insert(n, ‘…’) добавить (вставить) элемент в указанную позицию

.extend() добавить посимвольно элемент в конец писка

.remove() удалить из списка элемент по значению

.pop() удалить последний или указанный элемент по индексу

.clear() очистить список

in … проверить наличие элемента в списке

not in … проверить отсутствие элемента в списке

.find() найти первое вхождение подстроки в строке (-1 не найдено)

.rfind() найти первое вхождение подстроки в строке справа налево

.index() возвращает индекс первого вхождения элемента в списке

.count() возвращает кол-во вхождений элемента в списке

.sort() сортирует список по алфавиту (не создает новый))

.reverse() вывести список в обратном порядке

список […] […] многомерный список с вложениями (основной/вложенный)

enumerate() возвращает индекс и значение элемента списка

tuple ([…]) кортеж – неизменяемый список

**Словари и множества:**

{‘a’: 111, ‘b’: 222…} словарь {ключ: значение}

print (‘ ’) вывести значение ключа ‘ '

del (словарь) [ключ] удалить пару из словаря

.update({}) добавить в словарь

.get() возвращает значение по ключу

.pop() удаляет ключ и возвращает значение

.keys() возвращает коллекцию ключей из словаря

.values() возвращает коллекцию значений из словаря

.items() возвращает пары ключ: значение

{1, 2, 3, ‘a’, ‘b’, True, False, 0} множество (уникальный список }

print() не выведет True=1, Fales=отмена все что дальше

set() возвращает множество (уникальные значения )

.remove() удалить элемент из множества

.discard() удалить без сообщения об ошибке

.add() добавить 1 элемент во множество

.update() добавить несколько элемент во множество

zip(a, b) собрать словарь из двух списков {a1:b1, a2:b2…}  
dict(zip()) преобразовать словарь для вывода на экран

\*[] распаковка списка

\*\*[] распаковка словаря

*a, b = map(str.strip, input().split())* ввод данных в строку

*X, Y = map(int, input().split())* ввод данных в строку

*print(float(a)+float(b))* вывод результат вычислений

**Условия и циклы:**

if проверяет первое условие

elif следующее условие, если первой ложно

else если все предыдущие условия ложны

while … : выполнять пока условие верно  
break остановить выполнение цикла

continue пропустить выполнение оставшихся команд и повтор цикла

for i in … выполнять до достижения конца последовательности

range(…, …, …) последовательность числе от … до … с шагом …

**Функции:**

def …() : обычная функция

def …(name) : принимающая функция

return … возврат значения и завершает функцию

import random библиотека random

random.choice() случайный элемент из списка

def …(\*args) : распаковка позиционных параметров

def …(\*\*kwargs) : распаковка именованные параметров

def …(a=, b=…) параметры по умолчанию b, с …

**Встроенные функции:**

int() целое число

float() число с плавающей запятой

bool() логические значения

str() строки

list() список

tuple() кортеж

dict() словарь

set() множество

len() количество элементов в списке

sun() сумма всех элементов списка

max() максимальное значение в списке  
min() минимальное значение в списке

round(a, n) округленное значение, n цифр после запятой

zip(a, b) собрать список из элементов списков a и b

any() True если хотя бы один элемент списка True / не 0 / не пусто

all() True если все элементы списка True / не 0 / не пусто

dir() получить информацию об атрибутах / методах

isinstance(a, str) проверяет относиться ли а к классу строчных

id() получить уникальный ID объекта

a is b сравнить объекты a и b (True/False)

help() получить информацию о функциях

math.ceil() округление в большую сторону

math.floor() округление в меньшую сторону

math.frunc() отбрасывание дробной части

**Модули и пакеты:**

import Модуль1 подключить Модуль1 (целиком)

import Модуль1 as M1 подключить Модуль1 как М1

print(dir(M1)) посмотреть атрибуты Модуля1

М1.Фукция1() запустить Функию1 из Модуля1

from М1 import a, b подключить переменные a, b из Модуля1

from М1 import func1 подключить функцию func1 из Модуля1

from М1 import func1 as f1 подключить функцию func1 из Модуля1 как f1

from М1 import \* подключить все элементы из Модуля1

if \_ \_name\_ \_ == ’\_ \_main:\_ \_’не выполнять Main при вызове по Имени

main() (ограничительный скрипт)

**Пространства имен:**

Порядок поиска имен - изнутри наружу  
(сначала локальное, потом глобальное пространство, потом внутреннее)

global ‘…’ использовать глобальную переменную

nonlocal ‘…’ на использовать локальную переменную

**Классы и объекты:**

class () создание класса

self.name = name … переменная Name созданная на основе класса

def age(self): метод Age созданный на основе класса

def \_\_init\_\_(self, name): инициализация переменной с атрибутом Name

\_\_init\_\_(self) конструктор (создает объект класса)

\_\_del\_\_(self) деструктор (удаляет объект класса)

\_\_len\_\_(self) размер объекта

**Магические методы:**

\_\_lt\_\_(self, other) сравнение на Меньше (self < other)

\_\_le\_\_(self, other) сравнение на Меньше Равно (self <= other)

\_\_gt\_\_(self, other) сравнение на Больше (self > other)

\_\_gу\_\_(self, other) сравнение на Больше Равно (self >= other)

\_\_eq\_\_(self, other) сравнение на Равно (self == other)

\_\_ne\_\_(self, other) сравнение на Не Равно (self != other)

\_\_bool\_\_() проверка Истины

\_\_add\_\_(self, other) сложение (self + other)

\_\_sub\_\_(self, other) вычитание (self - other)

\_\_mul\_\_(self, other) умножение (self \* other)

\_\_truediv\_\_( self, other) деление (self / other)

\_\_str\_\_() информация об объекта класса для пользователей

\_\_repr\_\_() информация об объекта класса для разработчиков

\_\_contains\_\_() проверка вхождения объекта в класс

**Класс object и метод \_\_new\_\_:**

class object базовый класс, всей иерархии классов

включает в себя множество методов

…(self) указатель на объект класса

…(cls) указатель на класс

*\_\_instans = None* не дает создать новые объекты класса

*def \_\_new\_\_(cls, \*args, \*\*kwargs):*

*if cls.\_\_instance is None:* (если объекта нет, то создаст новый

*cls.\_\_instance = super().\_\_new\_\_(cls)* и вернет ссылку на него

*return cls.\_\_instance* если есть, то вернет ранее созданный)

**Переменное количество параметров в классах:**

*def \_\_init\_\_(self, \*args, \*\*kwargs)* переменное количество параметров

*self.args = args.get()* достать значение из списка и

*for i in args ()* пробежаться по списку и создавать

*setattr(self, i)* характеристику объекта класса

*self.kwargs = kwargs.get(‘name’)* достать значение по ключу Name

*for key.values in kwargs.items()* пробежаться по ключам и создавать

*setattr(self, key, values)* характеристику объекта класса

**Наследование классов:**

class aaa(bbb) дочерний класс ааа, родительский bbb

class \_aaa защищенный класс \_aaa, функции и переменные   
 доступны только локально

class \_\_aaa приватный класс, не наследуется

*def get\_a(self) геттер, возвращает приватный атрибут \_\_A  
 return self.\_\_a*

*def set\_a(self, b) сеттер, изменяет приватный атрибут \_\_A  
 self.\_\_a += b*

class aaa(bbb, ccc) множественное наследование

super().Метод() обращение к методу родительского класса без имени

например *super().\_\_init\_\_()*

.mro() вывод цепочки наследования

**Работа с файлами и форматированный вывод:**

ord(‘…’) перевести символ ‘…’ в код

chr(…) перевести код ‘…’ в символ

hex(ord(‘…’)) перевести символ ‘…’ в 16-ричный код

b’…’ байт – представление символа в 2-ом или 16-ом коде

.decode() перевод 16-ричного кода в символ

*from pprint impotr pprint выводить информацию в терминал в* *более удобочитаемом виде*

file = open(name, ‘r’) открыть файл Name для чтения

pprint(file.read()) считать содержимое файла

file.close() закрыть файл

file = open(name, ‘w’) открыть файл Name для записи (перезаписи)

file.write(‘…text…’) перезаписать содержимое файла

file.close() закрыть файл

file = open(name, ‘а’) открыть файл Name для append (добавления)

file.write(‘…text…\n’) добавить содержимое файла (\n перенос строки)

file.close() закрыть файл

rb / wb / ab режим чтения/записи/добавления бинарных файлов

print(file.tell()) возвращает положение курсора в файле

print(file.seek(n)) устанавливает курсор в позицию N

file = open(name, ‘r’, encoding=’utf-8’) открыть файл Name в кодировке UTF-8

print(file.writable()) возвращает можно ли записывать файл

print(file.readable()) возвращает можно ли читать файл

print(file.seekable()) возвращает можно ли перемещаться по файлу

print(file.closed) возвращает закрыт ли сейчас файл

print(file.name) возвращает имя файла

*name = example.txt* пример работы с файлом

*with open(name) as file:* открываем текстовый файл

*for line in file:* выводим построчно в консоль

*for char in line:* выводим посимвольно в консоль

*print(char, end=’’)* по окончании файл сам закроется

print(‘text1’ + ‘text2’) сложение двух строк

print(‘text %s, text %d’ % (‘значение1’, число)

print(‘text %(name)s, text %(year)s’ % {‘name’: ‘Значение1’, ‘year’: ’Значение2’)

print(‘text {} text {}’.format(‘Значение1’, ‘Значение2’))

print(‘text {name} text {year}’.format(name=‘Значение1’, year=‘Значение2’))

print(f‘text {переменная/функция} text’)

**Библиотека графического интерфейса:**

import tkinter импорт библиотеки

window = tkinter.Tk () создать окно

windows.title() заголовок окна

.geometry() задать размер окна (350x350)

.resizable(False, False) запретить изменение размера окна

tk.Button(window, text=’…’, 2, 2, …) создать кнопку, ширина, высота, команда

.place (x=100, y=200) разместить кнопку в окне

tk.Enry(window, …) создать текстовое поле шириной …

tk.Label(window. Text=’…’) создать надпись над текстовым полем

.get() извлечь значение из текстового поля

.delete(0, ‘end’) удалить значение из текстового поля

.insert(i, …) вставить значение в текстовое поле (индекс, переменная)

window.mailloop циклическое обновление окна

pip install auto-py-to-exe установка конвертера кода в приложение

Вебинар Вводный курс Python <https://drive.google.com/file/d/12q2vZ_YnR4GXXEgzTuM9ykmC2x9YCAFo/view>

Вебинар 1-й модуль

<https://kinescope.io/oCsS3ExfYmRz2vxBDKRwTo>

Вебинар 2-й модуль

<https://kinescope.io/2tbVKRcBdo726GRg9FuZ3K>

<https://drive.google.com/file/d/1nXYfcNCoVIqreU6V7gj-AZaSTp1feypy/view>