Занятие 8

Обработка исключений
Введение в ООП

Стандартные функции

Встроенные функции Python

print	input	str	int	dict	len
float	list	tuple	set	range	type
bool	enumerate	zip	reversed	sum	
min	max	sorted			

Математические функции.

```
import math
```

trunc	floor	ceil	exp	log
sqrt	factorial	fsum	sin	cos
tan	acos	asin	atan	degrees
radians	gamma	lgamma		

Сортировка

Встроенная функция sort()

Name	Best	Average	Worst	Memory	Stable
Quicksort	$n \log n$	$n \log n$	n^2	$\log n$	Depends
Merge sort	$n \log n$	$n \log n$	$n \log n$	Depends	Yes
In-place Merge sort	—	—	$n (\log n)^2$	1	Yes
Heapsort	$n \log n$	$n \log n$	$n \log n$	1	No
Insertion sort	n	n^2	n^2	1	Yes
Introsort	$n \log n$	$n \log n$	$n \log n$	$\log n$	No
Selection sort	n^2	n^2	n^2	1	Depends
Timsort	n	$n \log n$	$n \log n$	n	Yes
Shell sort	n	$n(\log n)^2$	$O(n \log^2 n)$	1	No
Bubble sort	n	n^2	n^2	1	Yes
Binary tree sort	n	$n \log n$	$n \log n$	n	Yes
Cycle sort	—	n^2	n^2	1	No
Library sort	—	$n \log n$	n^2	n	Yes
Patience sorting	—	—	$n \log n$	n	No
Smoothsort	n	$n \log n$	$n \log n$	1	No
Strand sort	n	n^2	n^2	n	Yes
Tournament sort	—	$n \log n$	$n \log n$		
Cocktail sort	n	n^2	n^2	1	Yes
Comb sort	—	—	n^2	1	No
Gnome sort	n	n^2	n^2	1	Yes
Bogosort	n	$n \cdot n!$	$n \cdot n! \to \infty$	1	No

Обработка исключений

ZeroDivisionError

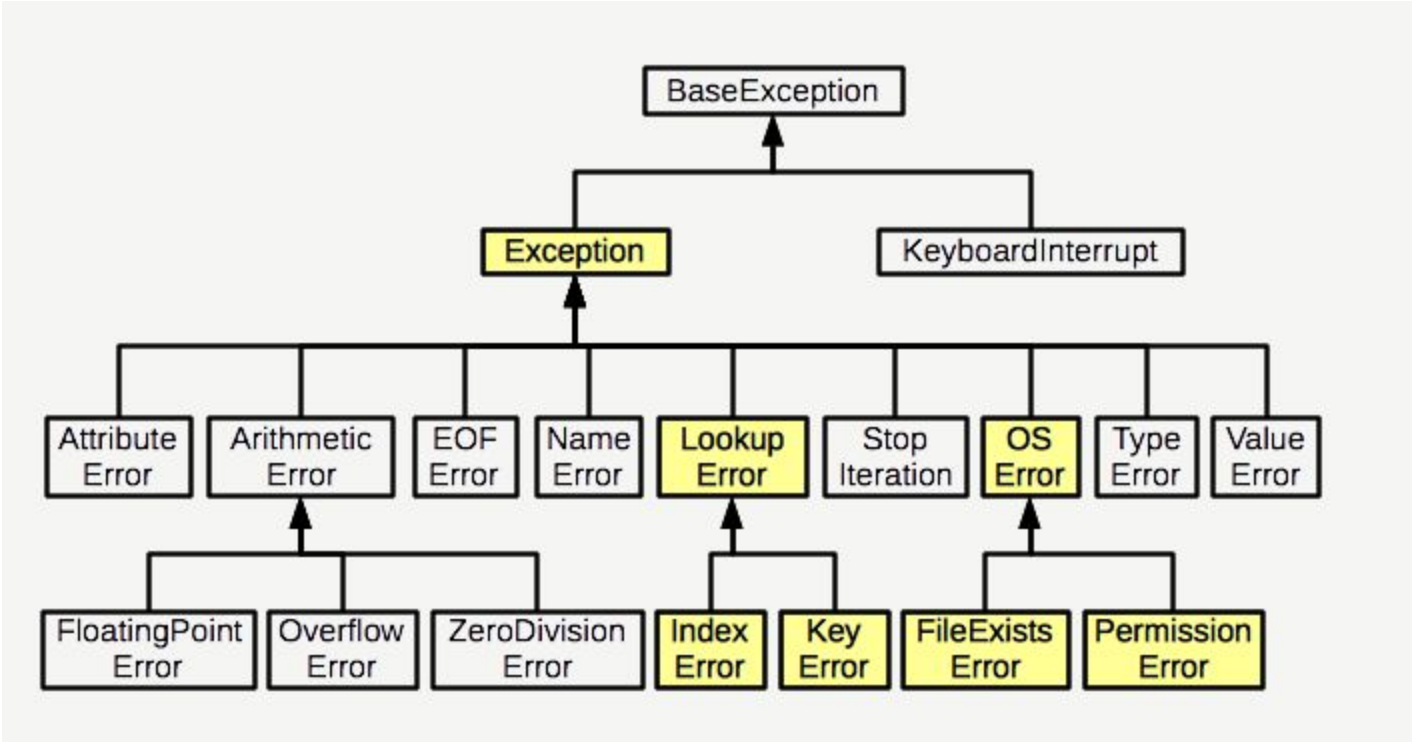
```
>>> 100 / 0
```

TypeError

```
>>> 2 + '1'
```

ValueError

```
>>> int('some')
```


BaseException
Exception
KeyboardInterrupt
Attribute Error
Arithmetic Error
EOF Error
Name Error
Lookup Error
Stop Iteration
OS Error
Type Error
Value Error
FloatingPoint Error
Overflow Error
ZeroDivision Error
Index Error
Key Error
FileExists Error
Permission Error

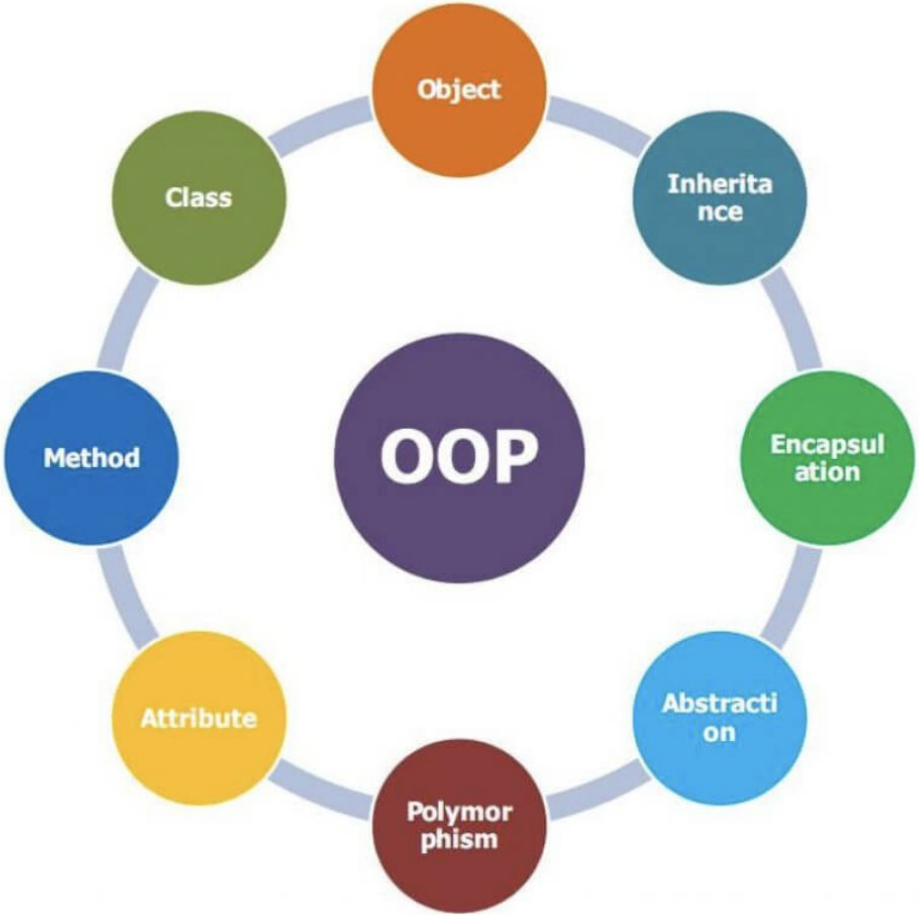
```
>>> try:
...     t = 100 / 0
... except ZeroDivisionError:

...     t = 0
```

```
>>> f = open('sample.txt')
>>> values = []
>>> try:
...     for line in f:
...         values.append(int(line))
... except ValueError:
...     print('This is not a number. Program finished.')
... except Exception:
...     print('Unknown value?')
... else:
...     print('All good.')
... finally:
...     f.close()
>>> print(values)
```

Что такое ООП?

Объектно-ориентированное программирование (ООП) - это модель компьютерного программирования, которая организует разработку программного обеспечения на основе данных или объектов, а не функций и логики. Объект можно определить как поле данных, которое имеет уникальные атрибуты и поведение.

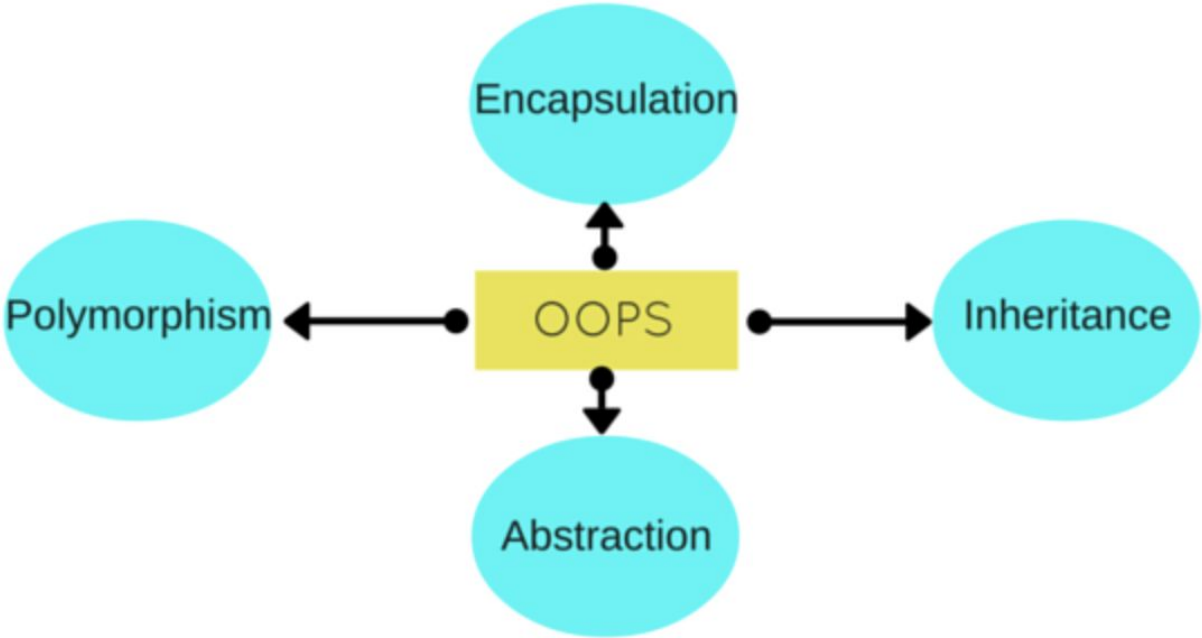

Object
Inherita nce
Encapsul ation
Abstracti on
Polymor phism
Attribute
Method
Class
OOP

Классы - это определяемые пользователем типы данных.

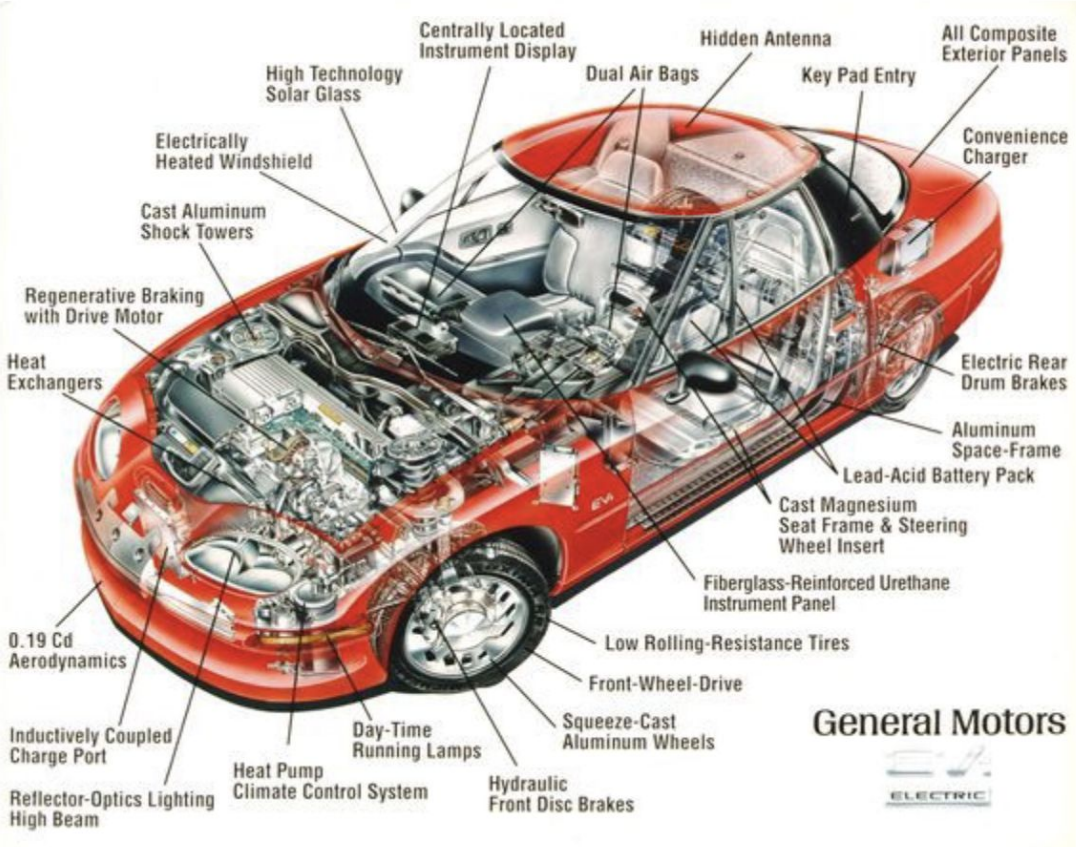
Объекты - это экземпляры класса, созданного со специально определенными данными. Объекты могут соответствовать объектам реального мира или абстрактной сущности.

Методы - это функции, которые определены внутри класса и описывают поведение объекта. Каждый метод, содержащийся в определениях класса, начинается со ссылки на объект-экземпляр.

Атрибуты определены в шаблоне класса и представляют состояние объекта.
Объекты будут иметь данные, хранящиеся в поле атрибутов.
Атрибуты класса принадлежат самому классу.


Encapsulation
Polymorphism
OOPS
Inheritance
Abstraction

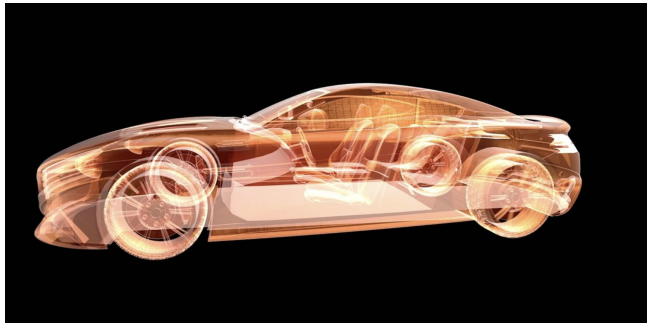
Инкапсуляция и Абстракция


Centrally Located
Instrument Display
Hidden Antenna
All Composite
Exterior Panels
High Technology
Solar Glass
Dual Air Bags
Key Pad Entry
Electrically
Heated Windshield
Convenience
Charger
Cast Aluminum
Shock Towers
Regenerative Braking
with Drive Motor
Heat
Exchangers
Electric Rear
Drum Brakes
Aluminum
Space-Frame
Lead-Acid Battery Pack
Cast Magnesium
Seat Frame & Steering
Wheel Insert
Fiberglass-Reinforced Urethane
Instrument Panel
0.19 Cd
Aerodynamics
Low Rolling-Resistance Tires
Front-Wheel-Drive
Squeeze-Cast
Aluminum Wheels
Inductively Coupled
Charge Port
Day-Time
Running Lamps
Heat Pump
Climate Control System
Hydraulic
Front Disc Brakes
Reflector-Optics Lighting
High Beam
General Motors
ELECTRIC

Наследование


MUSTANG MACH-E

Полиморфизм



Пусть нам нужно описать сотрудников организации с параметрами: имя, возраст, позиция и год начала работы.

Пусть нам нужно описать сотрудников организации с параметрами: имя, возраст, позиция и год начала работы.

```
sidorov = ["Anton Sidorov", 34, "CEO", 2015]
chehov = ["Ivan Chehov", 35, "CTO", 2017]
stepanov = ["Leonard Stepanov", "CPO", 2016]
```

Как объявить класс

```
class Employee:
    pass
```

```
class Employee:
    def __init__(self, name, age, position, year):
        self.name = name
        self.age = age
        self.position = position
        self.year = year
```

position

```
class Employee:
    pass
```

```
>>> a = Employee()
>>> b = Employee()
>>> a == b
```

```
class Employee:
    def __init__(self, name, age, position, year):
        self.name = name
        self.age = age
        self.position = position
        self.year = year


>>> sidorov = Employee("Anton Sidorov", 34, "CEO", 2015)
>>> chehov = Employee("Ivan Chehov", 35, "CTO", 2017)
>>> stepanov = Employee("Leonard Stepanov", "CPO", 2016)


>>> sidorov.name
>>> chehov.age
>>> stepanov.age
```

```
class Employee:
    def __init__(self, name, age, position, year):
          self.name = name
          self.age = age
          self.position = position
          self.year = year


       # Instance method

      def description(self):
          return f"{self.name} is {self.position} in company."


      # Another instance method
      def change_position(self, new_position):
          self.position = new_position


>>> stepanov = Employee("Leonard Stepanov", 25, "CPO", 2016)
>>> stepanov.description()
>>> stepanov.change_position( "CTO")
>>> stepanov.description()
```

Практика

Опишите класс Student со следующей информацией: ФИО, age, grade, subscribed_course. Создайте 5 экземпляров класса с разными значениями.

Добавьте студентов в массив и напишите функцию, которая отсортирует их по возрасту.

Напишите функцию, которая отсортирует студентов по оценкам.

Опишите класс Homework с атрибутами: name, description, complexity, status (passed or not)

Добавьте атрибут homeworks в класс Student

Домашнее задание

1. Напишете интерактивный калькулятор. Предполагается, что пользовательский ввод представляет собой формулу, состоящую из числа, оператора (как минимум + и -) и другого числа, разделенных пробелом (например, 1 + 1). Используйте str.split ()
    a. Если входные данные не состоят из 3 элементов, генерируйте эксепшн FormulaError.
    b. Попробуйте преобразовать первый и третий элемент в float (float_value = float(str_value)). Поймайте любую возникающую ValueError и сгенерируйте вместо него FormulaError
    c. Если второй элемент не является «+» или «-», киньте эксепшн FormulaError
2. Напишите минимум 5 тестов для каждой созданной функции
3. Пришлите ссылку на github с решением

Для генерации исключения используется ключевое слово raise
Пример:

```
raise FormulaError('Input does not consist of three elements')
```

Добавьте следующую строку в начале своего скрипта (main.py) для создания пользовательского исключения FormulaError

```
class FormulaError(Exception): pass
```

Домашнее задание*

4. Добавьте возможность считать более длинные выражения

1 + 2 + 4

1 + 2 - 1

1 + 2 + 3 - 1 + 4 - 2

1 + 2 + 32 - 1 + 4 * 20 + 100 - 200 + 551 / 2

Напишите соответствующие unit тесты

Домашнее задание

5. Изучите код программы с сегодняшнего занятия

6. Добавьте по 2 homeworks для каждого студента

7. Напишите метод для класса Student, который будет изменять status homework для конкретного задания

8. Добавьте класс Table, который должен содержать в себе всех студентов

9. Реализуйте метод в классе Table, для вывода информации о студентах и статусах homeworks в консоль

10. Добавьте 2 студента, проверьте, что метод из задания 5 работает для любого количества студентов

11. Пришлите ссылку на гитхаб на ваше задание