***Упражнение 5 – Обектно-ориентирано програмиране с Python***

1. Класове и обекти:

class Dog:

def \_\_init\_\_(self, name, age):

self.name = name

self.age = age

init функцията има 4 отстъпа

а тялото на функцията – 8 отстъпа

Тези отстъпи показват, че init принадлежи към класа Dog.

self.name = name създава атрибут, наречен име, и му присвоява стойността на параметъра name.

self.age = age създава атрибут, наречен възраст, и му присвоява стойността на параметъра age.

Атрибутите на класа са атрибути, които имат една и съща стойност за всички екземпляри на клас. Можете да дефинирате атрибут на клас, като присвоите стойност на име на променлива извън .\_\_init\_\_().

class Dog:

# Class attribute

species = "Canis familiaris"

def \_\_init\_\_(self, name, age):

self.name = name

self.age = age

Атрибутите на класа се дефинират директно под първия ред на името на класа и са с отстъп четири интервала. Винаги трябва да им се присвоява първоначална стойност. Когато се създаде екземпляр на класа, атрибутите на класа се създават автоматично и се присвояват на първоначалните им стойности.

Използвайте атрибути на клас, за да дефинирате свойства, които трябва да имат една и съща стойност за всеки екземпляр на клас. Използвайте атрибути на екземпляр за свойства, които се различават за всеки обект.

# Instance method

def description(self):

return f"{self.name} is {self.age} years old"

# Another instance method

def speak(self, sound):

return f"{self.name} says {sound}"

* description() връща низ, показващ името и възрастта на кучето.
* speak() има един параметър, наречен звук, и връща низ, съдържащ името на кучето и звука, който кучето издава.

1. Наследяване

class Person:  
  def \_\_init\_\_(self, fname, lname):  
    self.firstname = fname  
    self.lastname = lname  
  
  def printname(self):  
    print(self.firstname, self.lastname)  
  
#Use the Person class to create an object, and then execute the printname method:  
  
x = Person("John", "Doe")  
x.printname()

class Student(Person):  
  pass //ако не искаме да добавяме член-данни

class Student(Person):  
  def \_\_init\_\_(self, fname, lname): //вече не наследява на базовия клас init функцията

class Student(Person):  
  def \_\_init\_\_(self, fname, lname):  
    Person.\_\_init\_\_(self, fname, lname)

class Student(Person):  
  def \_\_init\_\_(self, fname, lname):  
    super().\_\_init\_\_(fname, lname)//super функцията позволява производния клас да наследява всички данни и методи на базовия

1. Задачи:
2. Напишете клас Rectangle с атрибути за дължина и ширина.
   1. Създайте метод Perimeter() за изчисляване на периметъра на правоъгълника и метод Area() за изчисляване на площта на правоъгълника.
   2. Създайте метод display(), който показва дължината, ширината, периметъра и площта на обект, създаден с помощта на инстанция върху клас правоъгълник.
3. Създайте клас BankAccount, който представлява банкова сметка, имаща като атрибути: accountNumber, name (име на собственика на акаунта като низов тип), баланс.
   1. Създайте конструктор с параметри
   2. Създайте метод deposit(), който управлява действията за депозит.
   3. Създайте метод withdrawal(), който управлява действията за теглене.
   4. Създайте метод bankFees(), за да приложите банковите такси с процент от 5% от балансовата сметка.
   5. Създайте метод display() за показване на подробности за акаунта.
4. Създайте клас Person с атрибути: име и възраст.
   1. Създайте метод display(), който показва името и възрастта на обект, създаден чрез класа Person.
   2. Създайте производен клас Student, който има атрибут faculty.
   3. Създайте метод displayStudent(), който показва името, възрастта и факултета на обект, създаден чрез класа Student.
   4. Създайте обект на ученик чрез инстанция в класа Student и след това тествайте метода displayStudent.