**Технически Университет – София,**

**Факултет „Компютърни системи и технологии“**

**Упражнение 6 – Създаване на програми с класове и обекти. Наследяване**

**Какво представляват класовете и обектите ?**

Python е обектно-ориентиран език. Това означава, че в него се използват класове и обекти. Най-общо казано обектите в Python представляват модели на реални обекти от действителността. Обектите могат да включват свойства и методи. Класовете пък могат да се разглеждат като „матрици” за производство на обекти.

**Как се създават и използват класове и обекти в Python ?**

За да създадем клас в Python използваме ключовата дума class. Например следният код:

**class MyFirstClass:**

**x=5**

създава клас с име MyFirstClass и едно свойство x, инициализирано със стойност 5.

За да създадем обект, базиран на този клас, използваме израза

**MyFirstObject= MyFirstClass()**

Елементите на класа достъпваме чрез използване на името на класа и името на елемена, разделени с точка:

**print(MyFirstObject.x)**

Всеки клас в Python има вградена функция конструктур \_\_init\_\_(), която се изпълнява автоматично винаги, когато класът се инициализира, т. е. когато се създава обект на базата на класа. Тази функция може да се използва, за да задаваме стойности на свойства при инициализация или да извършваме други действия, които са необходими при създаването на обектите.

Например за класа Person може да предефинираме функцията \_\_init\_\_(), така че тя да задава стойности на полетата name и age по следния начин:

**class Person:**

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

**self.name=name**

**self.age=age**

**MyPerson=Person('Ivan',35)**

**print(MyPerson.name)**

**print(MyPerson.age)**

Освен свойства, обектите могат да съдържат и методи. Методите на обектите са функции, които принадлежат на обектите. Например за създадения вече клас може да създадем метод greetings, който отпечатва поздравление от съответното лице:

**class Person:**

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

**self.name=name**

**self.age=age**

**def greetings(self):**

**print('Hello, my name is '+self.name)**

**MyPerson=Person('Ivan',35)**

**MyPerson.greetings()**

Параметърът self е рефереция към текущата инстанция на класа. Използва се за достъп до свойствата и методите, принадлежащи на класа. Не е задължително той да се нарича self. Може да го наричаме с друго име, важното е да първият параметър на всеки метод на класа.

Можем да променяме стойности на свойства на обекти, например

**MyPerson.age=40**

Можем също да изтриваме свойства на обекти с ключовата дума del.

**del MyPerson.age**

Можем да изтриваме и цели обекти, отново с ключовата дума del.

**del MyPerson**

Дефиницията на клас не може да е празна, но ако по някакава причина искаме да оставим клас без съдържание, може да използваме ключовата дума pass.

**class Person:**

**pass**

**Какво представлява наследяването?**

Наследяването е подход, който позволява един клас да наследи всички методи и свойства на друг клас. Наследяваният клас се нарича клас-родител. Наследяващият клас се нарича клас-дете.

Всеки клас може да бъде родителски. Създаването на родителски клас е същото, като на всеки друг клас.

Като пример създаваме клас, наречен Person, който има свойства firstname (малко име) и lastname (фамилно име) и метод printname:

**class Person:**

**def \_\_init\_\_(self,fname,lname):**

**self.firstname=fname**

**self.lastname=lname**

**def printname(self):**

**print (self.firstname, self.lastname)**

Може да използваме този клас, за да създадем обект и да извикаме метода printname.

**MyPerson=Person('Ivan','Petrov')**

**MyPerson.printname()**

За да създадем клас, който наследява даден клас, подаваме родителския клас като параметър при създаването на класа-наследник.

Например, можем да създадем клас Student, който наследява методите и класовете на класа Parent.

**class Student(Person):**

**pass**

Тук е подадена ключовата дума pass, защото за момента не е необходимо да се добавят нови свойства или методи към този клас. Така създаден, класът-наследник има същите свойства и методи като родителския клас. За него например може да създадем обект и да извикаме метода printname.

**MyStudent=Student('Petar','Vasilev')**

**MyStudent.printname()**

В класа-наследник можем да предефинираме функцията \_\_init\_\_(). Тогава класът-наследник вече не наследява тази функция от родителя, а има свой собствен вариант за нея. Наследяващият клас може да има свои добавени свойства и методи. Например към класа Student може да се добави свойство graduationyear. Към наследяващия клас може да добавим нови методи. Например, към класа Student може да добавим метода welcome.

**class Person:**

**def \_\_init\_\_(self,fname,lname):**

**self.firstname=fname**

**self.lastname=lname**

**def printname(self):**

**print (self.firstname, self.lastname)**

**class Student(Person):**

**def \_\_init\_\_(self,fname,lname,year):**

**Person.\_\_init\_\_(self,fname,lname)**

**self.graduationyear=year**

**def welcome(self):**

**print("Welcome ", self.firstname, " ",self.lastname," to the class of ", self.graduationyear)**

**MyStudent=Student('Petar','Vasilev','1992')**

**MyStudent.printname()**

**MyStudent.welcome()**

При добавяне на нови методи към класа-наследник с имена, съвпадащи с имена на методи в родителя, родителският метод се припокрива (override) от метода на наследника.

**Задачи :**

1. Да се напише код на Python, който дефинира клас Person с полета име (name), фамилия (family), възраст (age), националност (nationality). Да се дефинира конструктор, който инициализира полетата на класа. Да се добави метод print, който отпечатва имената и националността на съответното лице. Да се създадат обекти-инстанции на класа. За тях да се извика методът print.
2. Да се добави към кода от Задача 1 клас Student, наследник на класа Person с две нови полета – университет (university) и година на обучение (yearofstudy). Да се предифинира за него методът print, така, че да отпечатва и тези две полета. Да се създадат инстанции на новия клас и за тях да се извика методът print.
3. Да се добави към кода от Задача 1 и Задача 2 клас Lecturer, наследник на класа Person с две нови полета – университет (university) и опит (experience) – брой години преподавателски стаж. Да се предифинира за него методът print, така, че да отпечатва тези две полета. Да се създадат инстанции на новия клас и за тях да се извика методът print.