# Lab: Inheritance

Problems for in-class lab for the [Python OOP Course @SoftUni](https://softuni.bg/courses/python-oop).

Submit your solutions in the SoftUni judge system at <https://judge.softuni.bg/Contests/1940>.

# Part I: Inheritance

## Food

In a folder called **project** create two files: **food.py** and **fruit.py**:

* In the **food.py** file create a class called Food which will receive expiration\_date (str) upon initialization.
* In the **fruit.py** file create a class called Fruit with will receive name (str) and expiration\_date (str) upon initialization.

Fruit should inherit from Food.

Submit in Judge a **zip file** of thefolder **project**.

## Single Inheritance

In a folder called **project** create two files: **animal.py** and **dog.py**:

* In the **animal.py** file create a class called Animal with a single method eat() that returns: **"eating…"**.
* In the **dog.py** file create a class called Dog with a single method bark() that returns: **"barking…"**.

Dog should inherit from Animal.

Submit in Judge a **zip file** of thefolder **project**.

## Multiple Inheritance

In a folder called **project** create three files: **person.py** and **employee.py** and **teacher.py**.

In each file create its corresponding class - Person, Employee and Teacher:

* Person with a single method sleep() that returns: **"sleeping..."**
* Employee with a single method get\_fired() that returns: **"fired..."**
* Teacher with a single method teach() that returns: **"teaching..."**.

Teacher should inherit from Person and Employee.

Submit in Judge a **zip file** of thefolder **project**.

## Multilevel Inheritance

In a folder called **project** create three files: **vehicle.py** and **car.py** and **sports\_car.py**.

In each file create its corresponding class - Vehicle, Car and SportsCar:

* Vehicle with a single method move() that returns: **"moving..."**
* Car with a single method drive() that returns: **"driving..."**
* SportsCar with a single method race() that returns: **"racing..."**.

SportsCar should inherit from Car and Car should inherit from Vehicle.

Submit in Judge a **zip file** of thefolder **project**.

## Hierarchical Inheritance

In a folder called **project** create three files: **animal.py** and **dog.py** and **cat.py**.

In each file create its corresponding class - Animal, Dog and Cat:

* Animal with a single method eat() that returns: **"eating..."**
* Dog with a single method bark() that returns: **"barking..."**
* Cat with a single method meow() that returns: **"meowing..."**

Both Dog and Cat should inherit from Animal.

Submit in Judge a **zip file** of thefolder **project**.

# Part II: Reusing Classes

## Stack of Strings

Create a class Stack which can store **only strings** and has the following functionality:

* Instance attribute: **data: list**
* Method: push(element) – adds an element at the end of the stack
* Method: pop() – removes and returns the last element in the stack
* Method: top() - returns a reference to the top most element of the stack
* Method: is\_empty() - returns boolean True/False
* Override the **string method** to return the stack data in the format:

**"[{element(N)}, {element(N-1)} ... {element(0)}]"**