OOP – Lab #7 & #8

Aim: Getting Familiar with **Inheritance** and **Polymorphism**

Topics:

1. Class **Inheritance**
2. **Parent** Class (Superclass) and **Child** Class (Subclass)
3. Class **Polymorphism**

Lab Questions –

Q1 – Write Python code to create the following classes.

- **Shape** class,
- **Circle** class,
- **Rectangle** class.

Requirements:

- Consider class inheritance, so both **Circle** and **Rectangle** are **Shapes**.
- All shapes have a **color** attribute.
- Consider class polymorphism by defining a method named **calculate_area( )** in all classes to calculate and return the area of each shape.

```
import math
class Shape:
    def __init__(self, color):
        self.color = color
    def calculate_area(self):
        return 0

class Circle(Shape):
    def __init__(self, color, radius):
        Shape.__init__(self, color)
        self.radius = radius
    def calculate_area(self):
        area = math.pi * self.radius**2
        return area

class Rectangle(Shape):
    def __init__(self, color, width, length):
        Shape.__init__(self, color)
        self.width = width
        self.length = length
    def calculate_area(self):
        area = self.width * self.length
        return area
```

Q2 – Write Python code to create the following classes.

- **Person** class,
- **Student** class,
- **Teacher** class.

Requirements:

- Consider class inheritance, so both **Student** and **Teacher** are **Persons**.
- All classes' required attributes and methods are as follows:

Attributes	Methods
All Persons have **name**, **email**, and **age** attributes.	**change_domain( )** in ONLY *parent* class to change the domain of all emails to a new domain.
Students have an attribute named **grades**, which is a list of their grades in all courses.	**calculate_average( )** in *Student* class to calculate and return the average of each student's grades.
Teachers have an attribute named **teaching_years** to store each teacher's teaching years.	**calculate_extra_payment(***paymentPerYear***)** in *Teacher* class to calculate and return the extra payment amount based on teacher's teaching years.

```
class Person:
    def __init__(self, name, email, age):
        self.name = name
        self.email = email
        self.age = age
    def change_domain(self, newDomain):
        oldDomain = self.email.split('@')[1]
        self.email = self.email.replace(oldDomain, newDomain)

class Student(Person):
    def __init__(self, name, email, age, grades):
        Person.__init__(self,name, email, age)
        self.grades = grades
    def calculate_average(self):
        if len(self.grades)==0:
            return 0
        else:
            avg = sum(self.grades)/len(self.grades)
            return avg

class Teacher(Person):
    def __init__(self, name, email, age, teaching_years):
        Person.__init__(self, name, email, age)
        self.teaching_years = teaching_years
    def calculate_extra_payment(self, paymentPerYear):
        totalPayment = self.teaching_years * paymentPerYear
        return totalPayment
```