**RIPHAH INTERNATIONAL UNIVERSITY, ISLAMABAD**

****

**Lab 5**

**Bachelors of Computer science – 5th semester**

**Subject:** Artificial Intelligence Lab

**Submitted to:** Ma’am Ayesha

**Submitted by:** Munaza Malik

**Sap Id:** 44956

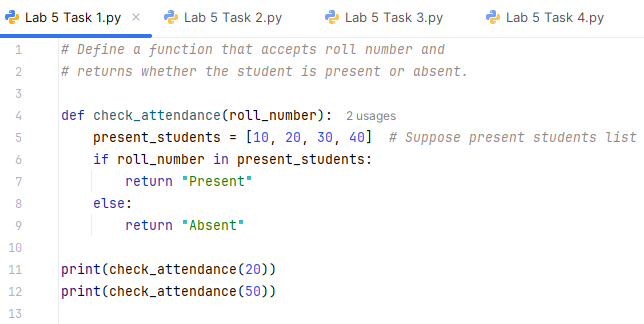
**Date:** 4th March , 2025

**Task 1:**

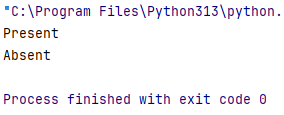
Define a function that accepts roll number and returns whether the student is present or absent.

**Code:**

def check\_attendance(roll\_number):  
 present\_students = [10, 20, 30, 40] *# Suppose present students list* if roll\_number in present\_students:  
 return "Present"  
 else:  
 return "Absent"  
  
print(check\_attendance(20))  
print(check\_attendance(50))



**Output:**

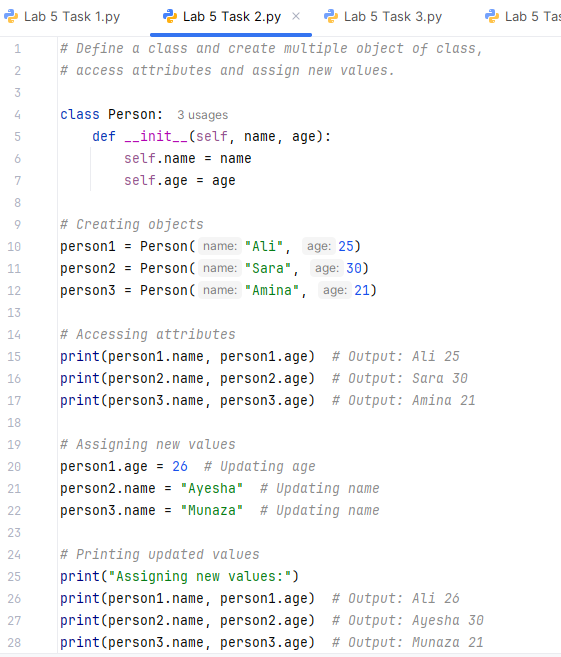


**Task 2:**

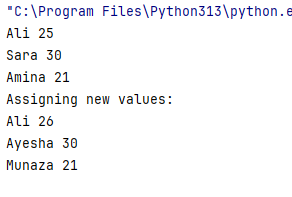
Define a class and create multiple object of class, access attributes and assign new values.

**Code:**

class Person:  
 def \_\_init\_\_(self, name, age):  
 self.name = name  
 self.age = age  
  
*# Creating objects*person1 = Person("Ali", 25)  
person2 = Person("Sara", 30)  
person3 = Person("Amina", 21)  
  
*# Accessing attributes*print(person1.name, person1.age) *# Output: Ali 25*print(person2.name, person2.age) *# Output: Sara 30*print(person3.name, person3.age) *# Output: Amina 21  
  
# Assigning new values*person1.age = 26 *# Updating age*person2.name = "Ayesha" *# Updating name*person3.name = "Munaza" *# Updating name  
  
# Printing updated values*print("Assigning new values:")  
print(person1.name, person1.age) *# Output: Ali 26*print(person2.name, person2.age) *# Output: Ayesha 30*print(person3.name, person3.age) *# Output: Munaza 21*



**Output:**

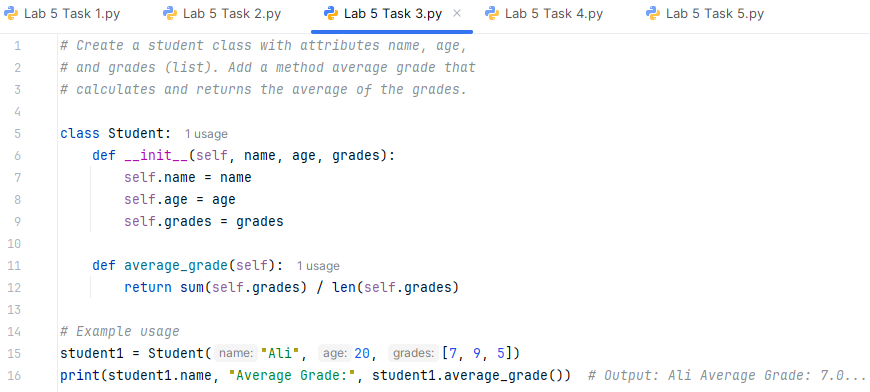


**Task 3:**

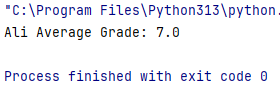
Create a student class with attributes name, age, and grades (list). Add a method average grade that calculates and returns the average of the grades.

**Code:**

class Student:  
 def \_\_init\_\_(self, name, age, grades):  
 self.name = name  
 self.age = age  
 self.grades = grades  
  
 def average\_grade(self):  
 return sum(self.grades) / len(self.grades)  
  
*# Example usage*student1 = Student("Ali", 20, [7, 9, 5])  
print(student1.name, "Average Grade:", student1.average\_grade()) *# Output: Ali Average Grade: 7.0...*



**Output:**



**Task 4:**

Create a base class **Employee** with:

* name
* salary
* Method **display\_details()** to show employee info.

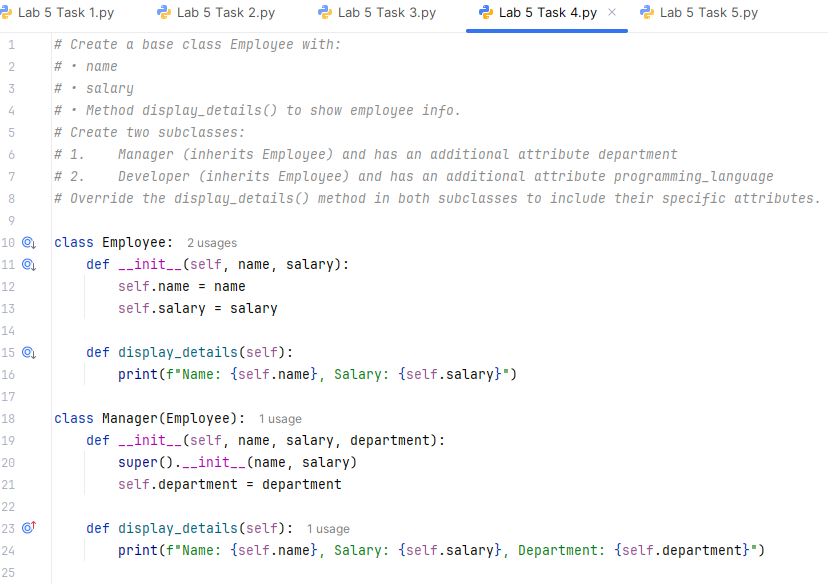
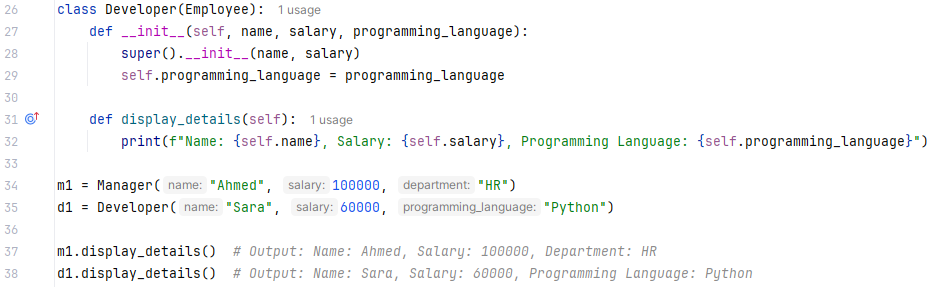
Create two subclasses:

1. **Manager** (inherits Employee) and has an additional attribute **department**
2. **Developer** (inherits Employee) and has an additional attribute **programming\_language**

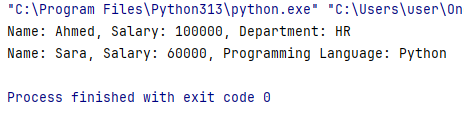
Override the **display\_details()** method in both subclasses to include their specific attributes.

**Code:**

class Employee:  
 def \_\_init\_\_(self, name, salary):  
 self.name = name  
 self.salary = salary  
  
 def display\_details(self):  
 print(f"Name: {self.name}, Salary: {self.salary}")  
  
class Manager(Employee):  
 def \_\_init\_\_(self, name, salary, department):  
 super().\_\_init\_\_(name, salary)  
 self.department = department  
  
 def display\_details(self):  
 print(f"Name: {self.name}, Salary: {self.salary}, Department: {self.department}")  
  
class Developer(Employee):  
 def \_\_init\_\_(self, name, salary, programming\_language):  
 super().\_\_init\_\_(name, salary)  
 self.programming\_language = programming\_language  
  
 def display\_details(self):  
 print(f"Name: {self.name}, Salary: {self.salary}, Programming Language: {self.programming\_language}")  
  
m1 = Manager("Ahmed", 100000, "HR")  
d1 = Developer("Sara", 60000, "Python")  
  
m1.display\_details() *# Output: Name: Ahmed, Salary: 100000, Department: HR*d1.display\_details() *# Output: Name: Sara, Salary: 60000, Programming Language: Python*

**Output:**



**Task 5:**

Create a base class **Shape** with a method **area()**.

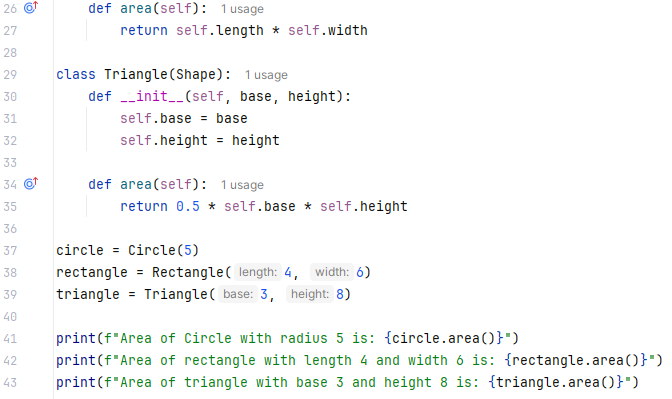
Create the following subclasses:

* **Circle** (takes radius and implements area() as π \* r²)
* **Rectangle** (takes length and width and implements area() as length × width)
* **Triangle** (takes base and height and implements area() as 0.5 × base × height)

Use **polymorphism** to calculate the area of different shapes.

**Code:**

import math  
  
class Shape:  
 def area(self):  
 pass  
  
class Circle(Shape):  
 def \_\_init\_\_(self, radius):  
 self.radius = radius  
  
 def area(self):  
 return math.pi \* self.radius \*\* 2  
  
class Rectangle(Shape):  
 def \_\_init\_\_(self, length, width):  
 self.length = length  
 self.width = width  
  
 def area(self):  
 return self.length \* self.width  
  
class Triangle(Shape):  
 def \_\_init\_\_(self, base, height):  
 self.base = base  
 self.height = height  
  
 def area(self):  
 return 0.5 \* self.base \* self.height  
  
circle = Circle(5)  
rectangle = Rectangle(4, 6)  
triangle = Triangle(3, 8)  
  
print(f"Area of Circle with radius 5 is: {circle.area()}")  
print(f"Area of rectangle with length 4 and width 6 is: {rectangle.area()}")  
print(f"Area of triangle with base 3 and height 8 is: {triangle.area()}")



**Output:**

