

# **Artificial Intelligence**



**Lab 05**

**Name: Hajira Imran**

**Sap ID: 44594**

**Submitted to: Ma'am Ayesha Akram**

**Batch: BSCS-6<sup>th</sup> semester**

## Task 01

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

```
task 01.py x
1  def check_attendance(roll_number): 1 usage
2      present_students = {101, 102, 103, 105}
3
4      return "Present" if roll_number in present_students else "Absent"
5
6  roll_number = int(input("Enter roll number: "))
7  print(check_attendance(roll_number))
8
```

## Output

```
Run task 01 x
C:\Users\lenovo\PycharmProjects\PythonProject3\.venv\Scripts\python.exe "C:\Users\lenovo\Py
Enter roll number: 103
Present
Process finished with exit code 0
```

```
Run task 01 x
C:\Users\lenovo\PycharmProjects\PythonProject3\.venv\Scripts\python.exe "C:\Users\lenovo\PycharmProjects\Pyt
Enter roll number: 99
Absent
Process finished with exit code 0
```

## Task 02

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

```
task 01.py task 02.py x task 03.py task 04.py task 05.py
2 class Car: 3 usages
3     def __init__(self, brand, model, year, price):
4
5         self.model = model
6         self.year = year
7         self.price = price
8
9     def display_info(self): 3 usages
10        print(f"Car: {self.brand} {self.model}, Year: {self.year}, Price: {self.price}")
11
12    # Objects
13    car1 = Car(brand: "Toyota", model: "Corolla", year: 2022, price: 3000000)
14    car2 = Car(brand: "Honda", model: "Civic", year: 2023, price: 3500000)
15    car3 = Car(brand: "Ford", model: "Mustang GT (V8)", year: 2024, price: 10000000)
16
17    # Access attributes
18    print(car1.brand)
19    print(car2.year)
20    print(car3.price)
21
22    # Modify attributes
23    car1.year = 2025
24    car2.model = "Accord"
25    car3.price = 12000000
26
27    car1.display_info()
28    car2.display_info()
29    car3.display_info()
```

## Output

```
Run task 02 x
C:\Users\lenovo\PycharmProjects\PythonProject3\.venv\Scripts\python.exe "C:\Users\lenovo\PycharmProjects\PythonProject3\task 02.py"
Toyota
2023
10000000
Car: Toyota Corolla, Year: 2025, Price: 3000000
Car: Honda Accord, Year: 2023, Price: 3500000
Car: Ford Mustang GT (V8), Year: 2024, Price: 12000000
Process finished with exit code 0
```

### Task 03

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

```
task 01.py task 02.py task 03.py x task 04.py task 05.py
1 class Student: 3 usages
2     def __init__(self, name, age, grades):
3         self.name = name
4         self.age = age
5         self.grades = grades
6
7     def average_grade(self): 1 usage
8         return sum(self.grades) / len(self.grades) if self.grades else 0
9
10    def display_info(self): 3 usages
11        print(f"Student: {self.name}, Age: {self.age}, Average Grade: {self.average_grade():.2f}")
12
13    # Create student objects
14    student1 = Student(name="Ali", age=20, grades=[85, 90, 78, 92])
15    student2 = Student(name="Ayesha", age=22, grades=[88, 76, 95, 89])
16    student3 = Student(name="Hassan", age=19, grades=[90, 85, 87, 80])
17
18    # Access attributes
19    print(student1.name)
20    print(student2.age)
21    print(student3.grades)
22
23    # Display student info
24    student1.display_info()
25    student2.display_info()
26    student3.display_info()
27
```

### Output

```
Ali
22
[90, 85, 87, 80]
Student: Ali, Age: 20, Average Grade: 86.25
Student: Ayesha, Age: 22, Average Grade: 87.00
Student: Hassan, Age: 19, Average Grade: 85.50

Process finished with exit code 0
```

#### **Task 04**

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.

```

task 01.py task 02.py task 03.py task 04.py task 05.py
1 # Parent class
2 class Employee: 3 usages
3     def __init__(self, name, salary):
4         self.name = name
5         self.salary = salary
6
7     def display_details(self): 1 usage
8         print(f"Employee Name: {self.name}, Salary: {self.salary}")
9
10 # Subclass 1: Manager
11 class Manager(Employee): 1 usage
12     def __init__(self, name, salary, department):
13         super().__init__(name, salary)
14         self.department = department
15
16     def display_details(self): 1 usage
17         print(f"Manager Name: {self.name}, Salary: {self.salary}, Department: {self.department}")
18
19 # Subclass 2: Developer
20 class Developer(Employee): 1 usage
21     def __init__(self, name, salary, programming_language):
22         super().__init__(name, salary)
23         self.programming_language = programming_language
24
25     def display_details(self): 1 usage
26         print(f"Developer Name: {self.name}, Salary: {self.salary}, Programming Language: {self.programming_language}")
27
28 #objects for each class
29 emp = Employee( name: "John Doe", salary: 50000)
30 mgr = Manager( name: "Alice Smith", salary: 80000, department: "Sales")
31 dev = Developer( name: "Bob Johnson", salary: 70000, programming_language: "Python")
32 emp.display_details()
33 mgr.display_details()
34 dev.display_details()

```

## Output

```

Run task 04
C:\Users\lenovo\PycharmProjects\PythonProject3\.venv\Scripts\python.exe "C:\Users\lenovo\PycharmProjects\PythonProject3\task 04.py"
Employee Name: John Doe, Salary: 50000
Manager Name: Alice Smith, Salary: 80000, Department: Sales
Developer Name: Bob Johnson, Salary: 70000, Programming Language: Python
Process finished with exit code 0

```

## Task 05

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

Create the following subclasses:

- Circle (takes radius and implements area() as  $\pi * r^2$ )
- Rectangle (takes length and width and implements area() as length  $\times$  width)

- Triangle (takes base and height and implements area() as  $0.5 \times \text{base} \times \text{height}$ )

Use polymorphism to calculate the area of different shapes.

```
task 01.py task 02.py task 03.py task 04.py task 05.py ×
1 import math
2 class Shape: 3 usages
3     def area(self):
4         pass
5 # Subclass 1: Circle
6 class Circle(Shape): 1 usage
7     def __init__(self, radius):
8         self.radius = radius
9
10    def area(self): 1 usage
11        return math.pi * self.radius ** 2
12 # Subclass 2: Rectangle
13 class Rectangle(Shape): 1 usage
14     def __init__(self, length, width):
15         self.length = length
16         self.width = width
17
18    def area(self): 1 usage
19        return self.length * self.width
20 # Subclass 3: Triangle
21 class Triangle(Shape): 1 usage
22     def __init__(self, base, height):
23         self.base = base
24         self.height = height
25    def area(self): 1 usage
26        return 0.5 * self.base * self.height
27 circle = Circle(5)
28 rectangle = Rectangle(length=15, width=4)
29 triangle = Triangle(base=5, height=3)
30
31 # Using polymorphism to calculate areas
32 shapes = [circle, rectangle, triangle]
33 for shape in shapes:
34     print(f"{shape.__class__.__name__} Area: {shape.area():.2f}")
```

## Output



The image shows a PyCharm Run console window for a task named 'task 05'. The console displays the output of a Python script, which includes the area calculations for a circle, a rectangle, and a triangle. The process finished with exit code 0.

```
Run task 05 x
C:\Users\lenovo\PycharmProjects\PythonProject3\.venv\Scripts\python.exe "C:\Users\lenovo\Pyd
Circle Area: 78.54
Rectangle Area: 60.00
Triangle Area: 7.50
Process finished with exit code 0
```