Green University of Bangladesh
Department of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering

Semester: (Spring, Year: 2025), B.Sc. in CSE (Day)

LAB REPORT NO 02

Course Title: Data Mining Lab

Course Code: CSE 436 Section: 213 D4

Lab Experiment Name: In-Depth Exploration of Object-Oriented Programming: Class Creation, Attribute Modification, and Instance Verification in Python

Student Details

	Name	ID
1.	Md. Rabby Khan	213902037

Lab Date : **10/02/2025**

Submission Date : **16/02/2025**

Course Teacher's Name : **Md. Jahid Tanvir**

[For Teachers use only: Don't Write Anything inside this box]

Lab Report Status

Marks: …… **Signature: .....**

Comments: ..... **Date: .....**

1. TITLE OF THE LAB EXPERIMENT

Comprehensive Exploration of Object-Oriented Programming Concepts: Class Creation, Attribute Manipulation, and Data Handling in Python

2. OBJECTIVES/AIM

- To verify class relationships using isinstance() and issubclass().
- To define and modify attributes of objects after instantiation.
- To manipulate dictionaries by removing key-value pairs.
- To access and print object attributes effectively.
- To understand object-oriented programming concepts like class creation and manipulation.

3. PROCEDURE / DESIGN

Task 1: Use of isinstance() and issubclass()

- Define two classes: Student and Marks.
- Instantiate objects of both classes.
- Use the isinstance() function to verify if an object is an instance of a specific class.
- Use the issubclass() function to check if a class is a subclass of the object class.

Task 2: Modifying Class Attributes

- Create a Student class with attributes studentName and marks.
- Instantiate an object of the Student class.
- Print the initial values of the attributes.
- Modify the values of studentName and marks attributes.
- Print the modified values.

Task 3: Removing a Dictionary Key

- Create a dictionary representing a student's information with keys like studentId, studentName, and studentClass.
- Print the dictionary before key removal.

- Use the del statement to remove the studentName key.
- Print the updated dictionary after the key removal.

Task 4: Accessing and Printing Object Attributes

- Define a Student class with attributes studentId, studentName, and studentClass.
- Instantiate a Student object with sample values for each attribute.
- Print the attributes of the object using formatted strings.

4. IMPLEMENTATION

LAB REPORT 1 TASK 1

Welcome | Lab_Report 2 Task 1

Lab_Report 2 Task 1

```python
# Task 1: Create two empty classes and check instances and subclasses
class Student:
    pass

class Marks:
    pass

student_instance = Student()
marks_instance = Marks()

print(isinstance(student_instance, Student))
print(isinstance(marks_instance, Marks))

print(issubclass(Student, object))
print(issubclass(Marks, object))

```

OUTPUT:

```
True
True
True
True
```

OUTPUT EXPLANATION:

The code demonstrates class instantiation and inheritance checks using isinstance() and issubclass(). Both classes are instances of their respective classes and subclasses of object, resulting in four True outputs.

LAB REPORT 2 TASK 2

Welcome | Lab_Report 2 Task 1 | Lab Report 2 Task 2

Lab Report 2 Task 2

```python
# Task 2: Student class with studentName and marks
class Student:
    def __init__(self, studentName, marks):
        self.studentName = studentName
        self.marks = marks


student1 = Student("Md Rabby ", 85)

print(f"Original values: Name = {student1.studentName}, Marks = {student1.marks}")


student1.studentName = "Md Rabby Khan Updated"
student1.marks = 95

# Print modified values
print(f"Modified values: Name = {student1.studentName}, Marks = {student1.marks}")

```

OUTPUT:

```
Original values: Name = Md Rabby , Marks = 85
Modified values: Name = Md Rabby Khan Updated, Marks = 95
```

OUTPUT EXPLANATIONS:

A Student class is created with student's name and marks. For instance student1 is initialized with "Md Rabby" and 85. After updating the attributes to "Md Rabby Khan Updated" and 95, the modified values are printed, demonstrating how object attributes can be changed.

LAB REPORT 2 TASK 3

Welcome | Lab_Report 2 Task 1 | Lab Report 2 Task 2 | Lab Report 2 Task 3

Lab Report 2 Task 3

```python
# Task 3: Add a new attribute and remove one
class Student:
    def __init__(self, studentId, studentName):
        self.studentId = studentId
        self.studentName = studentName


student2 = Student(213902037, "Md Rabby Khan")

# Add a new attribute
student2.studentClass = "4th year cse  Student"

print("Before removing studentName:")
print(student2.__dict__)

del student2.studentName

print("After removing studentName:")
print(student2.__dict__)
```

OUTPUT:

```
Before removing studentName:
{'studentId': 213902037, 'studentName': 'Md Rabby Khan', 'studentClass': '4th year cse  Student'}
After removing studentName:
{'studentId': 213902037, 'studentClass': '4th year cse  Student'}
```

OUTPUT EXPLANATIONS:

In this task, a dictionary representing a student's information is created. Initially, the dictionary contains three keys: studentId, studentName, and studentClass. The output shows the dictionary before the removal of the studentName key, displaying the values associated with each key.

After removing the studentName key using the del statement, the dictionary only contains studentId and studentClass. The output confirms the change by printing the updated dictionary, which no longer includes the studentName.

LAB REPORT 2 TASK 4

Welcome | Lab Report 2 Task 4 | Lab_Report 2 Task 1 | Lab Report 2 Task 2 | Lab Report 2 Task 3

Lab Report 2 Task 4

```python
# Task 4: Display attributes using a function
class Student:
    def __init__(self, studentId, studentName):
        self.studentId = studentId
        self.studentName = studentName
        self.studentClass = None

    # Function to display all attributes
    def display_attributes(self):
        for attr, value in self.__dict__.items():
            print(f"{attr}: {value}")

student3 = Student(213902037, "Md Rabby Khan")
student3.studentClass = "4th year cse student "

print("Attributes of Student:")
student3.display_attributes()
```

OUTPUT:

```
Attributes of Student:
studentId: 213902037
studentName: Md Rabby Khan
studentClass: 4th year cse student
```

OUTPUT EXPLANATIONS:

In this task, the attributes of a Student object are printed. The studentId, studentName, and studentClass attributes are displayed with their corresponding values: 213902037, "Md Rabby Khan", and "4th year cse student". The output shows the details of the Student object.

6. ANALYSIS AND DISCUSSION

In **Task 1**, we explored the use of isinstance() and issubclass() functions to check instance-class relationships and inheritance. These functions helped verify that the Student and Marks classes are instances of their respective classes and subclasses of the object class, returning True in both cases.

Task 2 demonstrated how to define a Student class with attributes like studentName and marks, which could be modified after object instantiation. This task showcased Python's ability to update object attributes dynamically.

In **Task 3**, we manipulated a dictionary by removing the studentName key using the del statement, which effectively modified the dictionary to display only the remaining key-value pairs.

Finally, **Task 4** focused on printing object attributes such as studentId, studentName, and studentClass, emphasizing how easily Python allows access to and display of object data. Overall, these tasks reinforced the fundamental concepts of object-oriented programming, including class instantiation, attribute manipulation, dictionary handling, and the use of built-in functions like isinstance() and issubclass().

7. SUMMARY

In this set of tasks, I explored key Python concepts such as object-oriented programming, dictionary manipulation, and built-in functions. In **Task 1**, we used isinstance() and issubclass() to verify object-class relationships and inheritance. **Task 2** demonstrated how class attributes can be modified after object instantiation. **Task 3** focused on removing a key from a dictionary using the del statement, showing how to modify dictionaries in Python. Finally, **Task 4** illustrated how to access and print object attributes. These tasks provided a comprehensive understanding of working with objects, classes, dictionaries, and built-in functions in Python.