Name: Abdullah Al Mamun Sagor
ID: IT22059

Project Name: OOP Revisiting

1. Classes and Object:

```
Class Student {
    String name;
    int rollNumber;
    void displayDetails() {
        System.out.println("Name: " + name);
        System.out.println("Roll Number: " + rollNumber);
    }
}

public class classesandobject {
    public static void main(String[] args) {
        Student student1 = new student();
        student1.name = "Alice";
        student1.roll number = 101;
        Student student2 = new Student();
        Student2.name = "Bob";
        Studen2.rollNumber = 102;
        System.out.println("Student1:");
        Student1.displayDetails();
        System.Out.println("\n student 2:");
        Student2.displayDetails();
    }
}
```

2. Access Modifier:

```
Class Person

  public String name;
   private int age;
   String city;
  public void setAge (int newAge) {
      age = newAge;
  }
  public void displayInfo () {
      System.out.println ("Name:" + name);
      System.out.println ("Age: " + age);
      System.out.println ("City: " + city);
  }}

Public class accessmodifier {
    public static void main (String[] args) {
        Person P1 = new Person ();
         P1.name = "John";
         P1.city = "New York";
        P1.setAge (25);
        P1.displayInfo ();
    }
}
```

3. Inheritance and Protected Access :

```
Class Shape {
    protected String color;
    protected void displaycolor() {
        System.out.println("color: " + color);
    }
}

Class Circle extends Shape {
    void setcolor(string c) {
        color = c;
        displaycolor();
    }
}

public class inheritanceandprotectedaccess {
    public static void main(String[] args) {
        Circle circle1 = new Circle();
        circle1.setcolor("Red");
    }
}
```

4. Encapsulation:

```
Class Employee {
    private String name;
    private int salary;
    public void setName(String n) {
        name = n;
    }
```

```
Public String getName (){
    return name;
}
public void getSalary (int s){
    Salary = s;
}
    public int getSalary () {
        return Salary;
}}
public class encapsulation{
  public static void main (String [] args){
      Employee emp1 = new Employee ();
    emp1 . SetName ("Alice");
    emp1 . setSalary (50000);
  System . out . println (" Employee Name" + emp.getName());
System.out.println ("Employee Salary : $ " +emp1.getSalary());
}}
}
```

5. Abstract Classes:

```
abstract class Vehicle {
    abstract void ();
}
Class Bike extends vehicle {
    void start () {
System. out. println (" Bike starts with a kick! ");
}
```

```
public class abstractClass {
    public static void main(String[] args) {
        Bike myBike = new Bike();
        myBike.start();
    }
```

6. Interface:

```
Interface Animal {
    void eat();
    void sleep();
}
Class Dog implements Animal {
    public void eat() {
        System.out.println("Dog is eating bones.");
    }
    public void sleep() {
        System.out.println("Dog is sleeping.");
    }
}
public class interface {
    public static void main(String[] args) {
        Dog dg1 = new Dog();
        dog1.eat();
        dog_1.sleep();
    }
}
```

7. Multiple Inheritance Using Interface:

```
Interface Animal {
    void eat();
    void sleep();
}
Interface Bird {
    void fly();
}
Class Eagle implements Animal, Bird {
    public void eat() {
    System.out.println("Eagle is sleeping eating.");
    }
    public void sleep() {
        System.out.println("Eagle is sleeping.");
    }
    public void fly() {
        System.out.println("Eagle is flying.");
    }
}
public class multipleinheritance {
    public static void main(String[] args) {
        Eagle eage = new Eagle();
            eagle.eat();
            eagle.sleep();
            eagle.fly();
    }
}
```

8. ATM Project.

```
public class ATMSimulation {
    Public class ATM {
        private double blance;
    Public ATM (double balance){
        this. balance = balance;
    }
public void deposit (double amount ) {
    if (amount >0) {
        balance +==amount ;
    System.out. println (" Deposited: $ " + amount);
} else {
  System.ad. println (" Invalid amount ");
}
}

public void withdraw (double amount ) {
 if (amount >0  && amount <= balance){
  balance - = amount ;
  System. out. println ( withdraw : $" + amount);
 } else if (amount > balance) {
  system. out. println (" Insufficient balance. ");
 } else {
  System. out. println (" Invalid amount ");
 }
}
```

```
Public void checkBalance() {
    System.out.println("Current balance: $" + balance);
}
}

Public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in)
    ATM atm = new ATM ;
   While (true) {
    System.out.println("\n ATM Menu: ");
    System.Out.println(" 1. Deposit ");
    System.out.println("2. Withdraw ");
    System.out.println("3. Check Balance ");
    System.out.println(" 4. Exit ");
    System.out.print(" Choose an option : ");

  int option = scanner.nextInt();
  Switch (option) {
    case 1:
    System.out.print("Enter deposit amount: $ ");
     double depositAmount = scanner.nextDouble();
     atm.deposit(depositAmount);
      break;
    case 2;
    System.out.print("Enter withdrawal amount :$");
    double withdrawalAmount = scanner.nextDouble();
     atm.withdraw(withdrawalAmount);
    break;
```

```
case 3:
 atm.checkBalance();
 break
 case 4:
 system.out.println("Thank You for using the ATM.Goodbye!");
  scanner.close();
  System.exit;
  break;
default:
   System.out.println("Invalid option.Please try again");
}
}
}
}
```

91. Calculator:

```
import java.util.Scanner;
public class calculator{
 public static void main(String[] args){
    Scanner scanner = new Scanner(System.in);
 double num1, num2, result;
 Char operator;
System.out.println("Simple calculator");
System.out.println("Enter first number:");
 num1 = scanner.nextDouble();
```

IT22059

```
System.out.print("Enter an operator (+,-,*,/) : ");
  Operator = scanner.next().charAt(0);
System.out.Print("Enter Second numbe: ");
  num2 = scannerr.nextDouble();
Switch (operator) {
  case'+';
 result=num1+num2;
 system.out.println("Result : " +result);
 break);
Case '-';
  result = num1 - num2;
   System.out.println("ResuH : " + resuH);
   break;
case '*';
   result = num1 * num2;
 system.out.println("Result: " +result);
 break;
case '/';
   if (num2 != 0) {
     result = num1 / num2;
   System.out.println("Result: " + result);
 } else {
     System.out.println("Error: Cannot divide by zero!");
 }
 break;
```

```
default:
    System.out.println("Invalid Operator! ");
  }
  scanner.close();
  }
}
```