# LAB RECORD

23CSE111- Object Oriented Programming

***Submitted by***

CH.SC.U4CSE24134 -P JEEVAN SANDEEP

**BACHELOR OF TECHNOLOGY**

IN

COMPUTER SCIENCE AND ENGINEERING

AMRITA VISHWA VIDYAPEETHAM

AMRITA SCHOOL OF COMPUTING

CHENNAI

March - 2025

AMRITA VISHWA VIDYAPEETHAM
VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY UNDER SECTION 3 OF UGC ACT, 1956
SCHOOL OF COMPUTING

**AMRITA VISHWA VIDYAPEETHAM**

**AMRITA SCHOOL OF COMPUTING, CHENNAI**

# BONAFIDE CERTIFICATE

This is to certify that the Lab Record work for 23CSE111-Object Oriented Programming Subject submitted by ***CH.SC.U4CSE24134 – P JEEVAN SANDEEP*** in **"Computer Science and Engineering"** is a Bonafide record of the work carried out under my guidance and supervision at Amrita School of Computing, Chennai.

This Lab examination held on / /2025

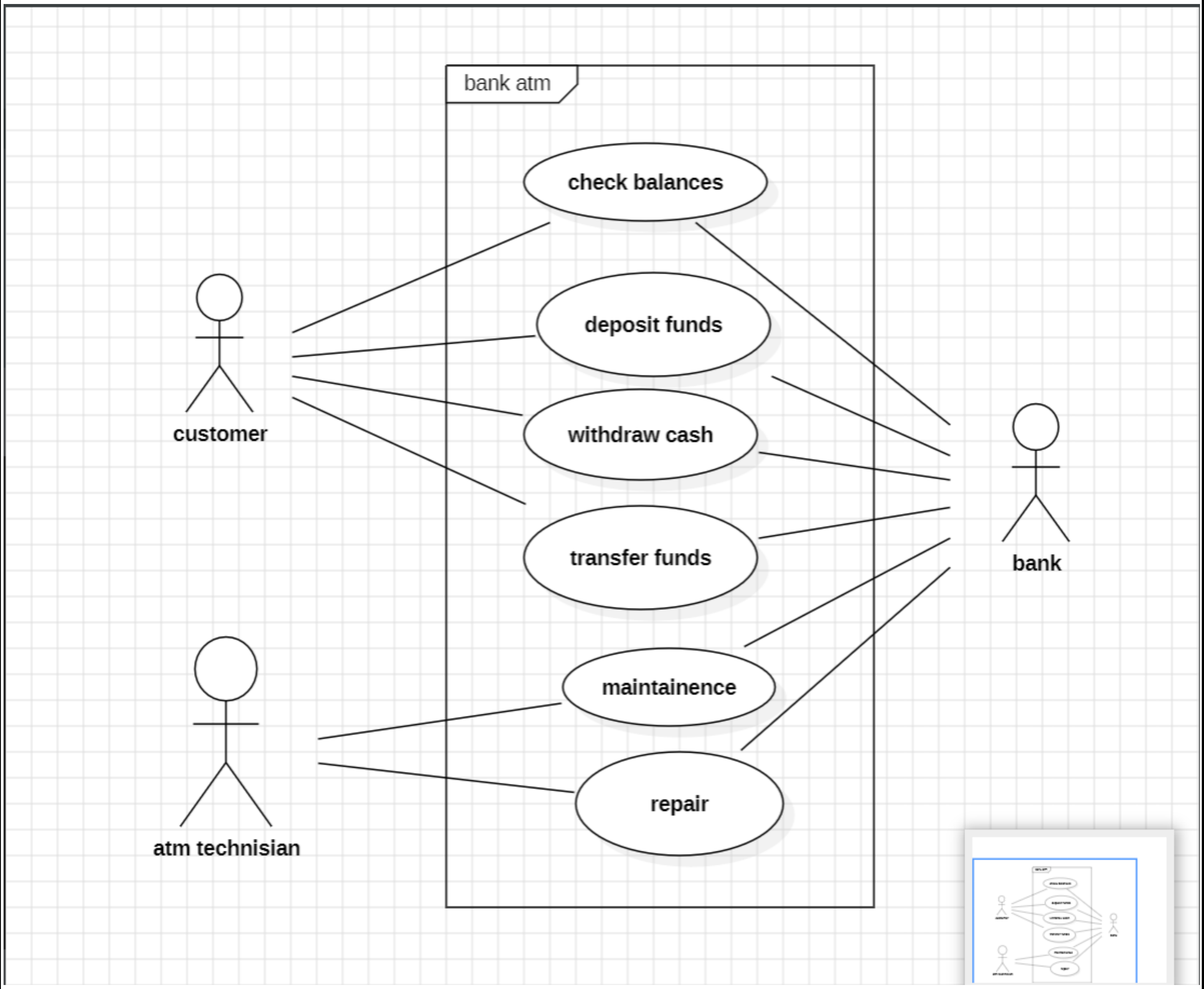
Internal Examiner 1 Internal Examiner 2

# INDEX

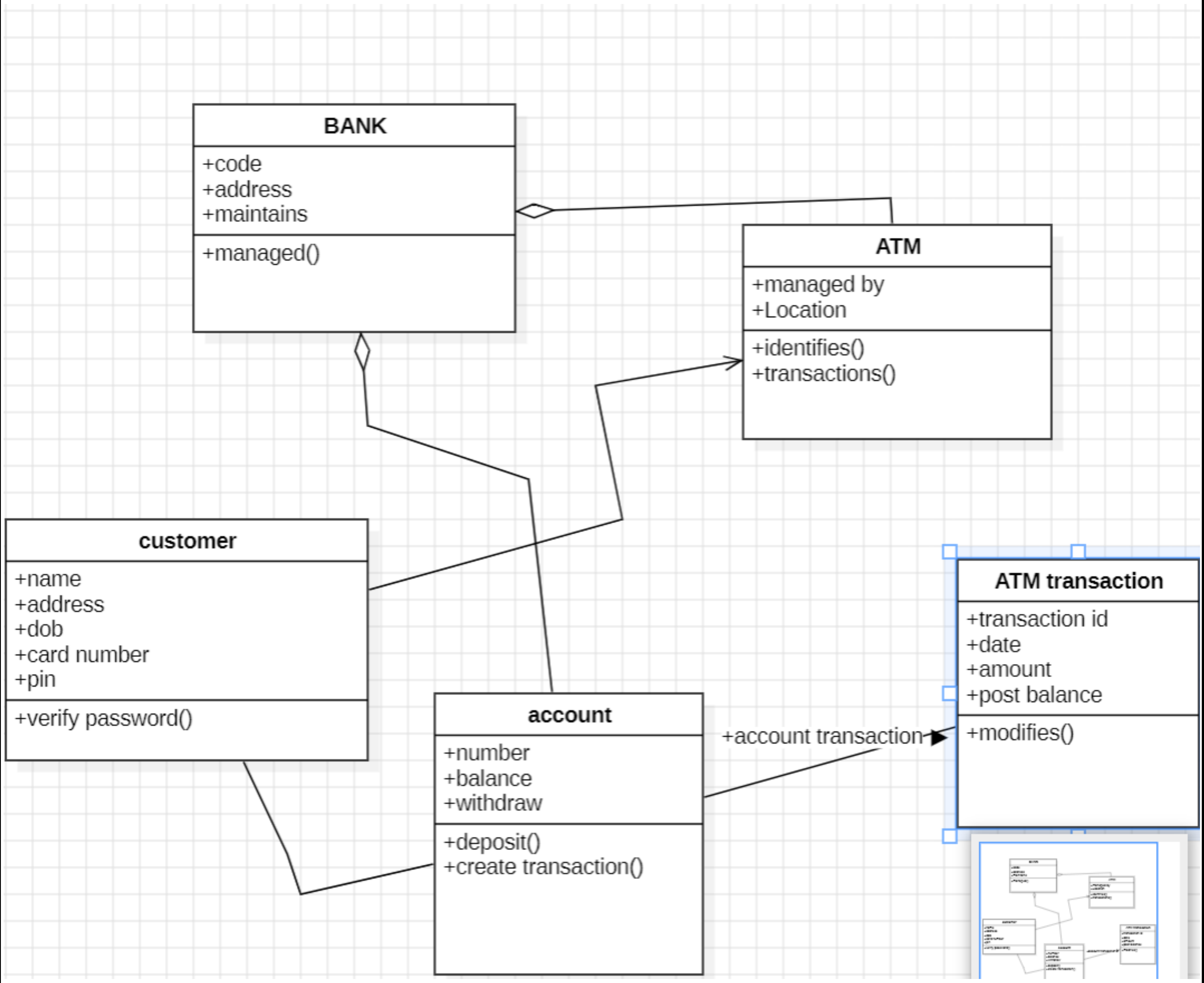
# UML DIAGRAMS

## 1. ATM MACHINE

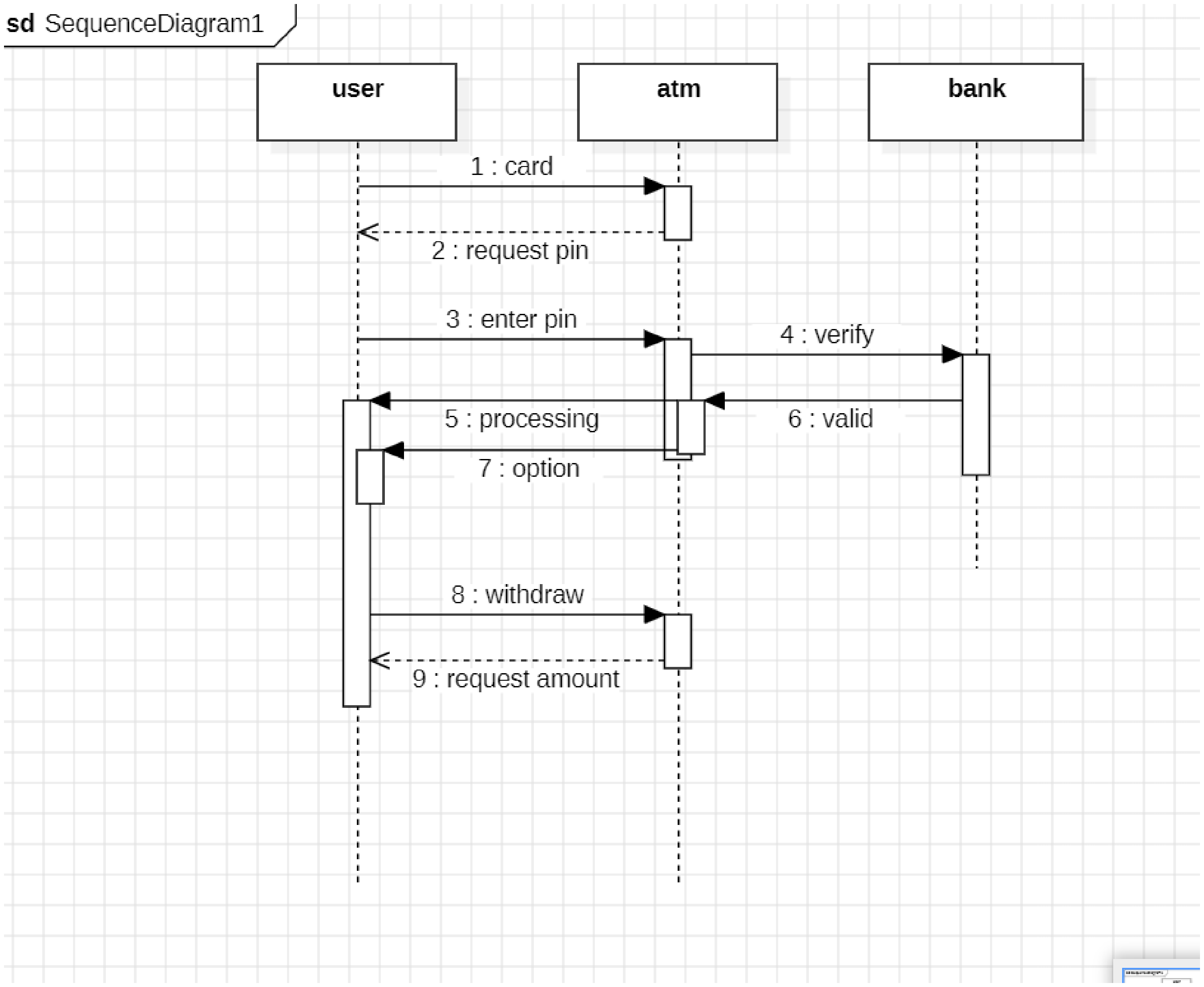
### 1.a) Use Case Diagram:



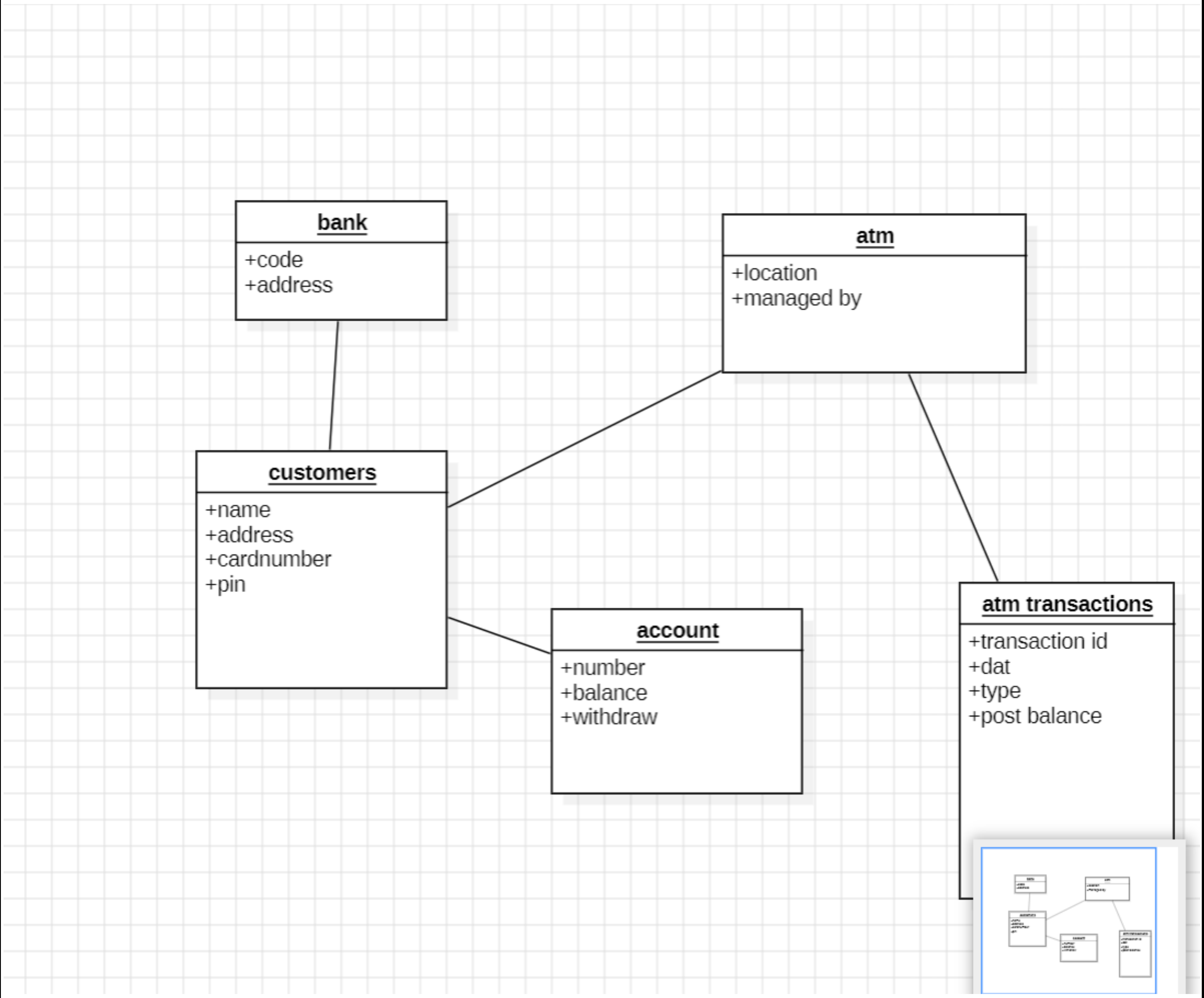
## 1.b) Class Diagram:



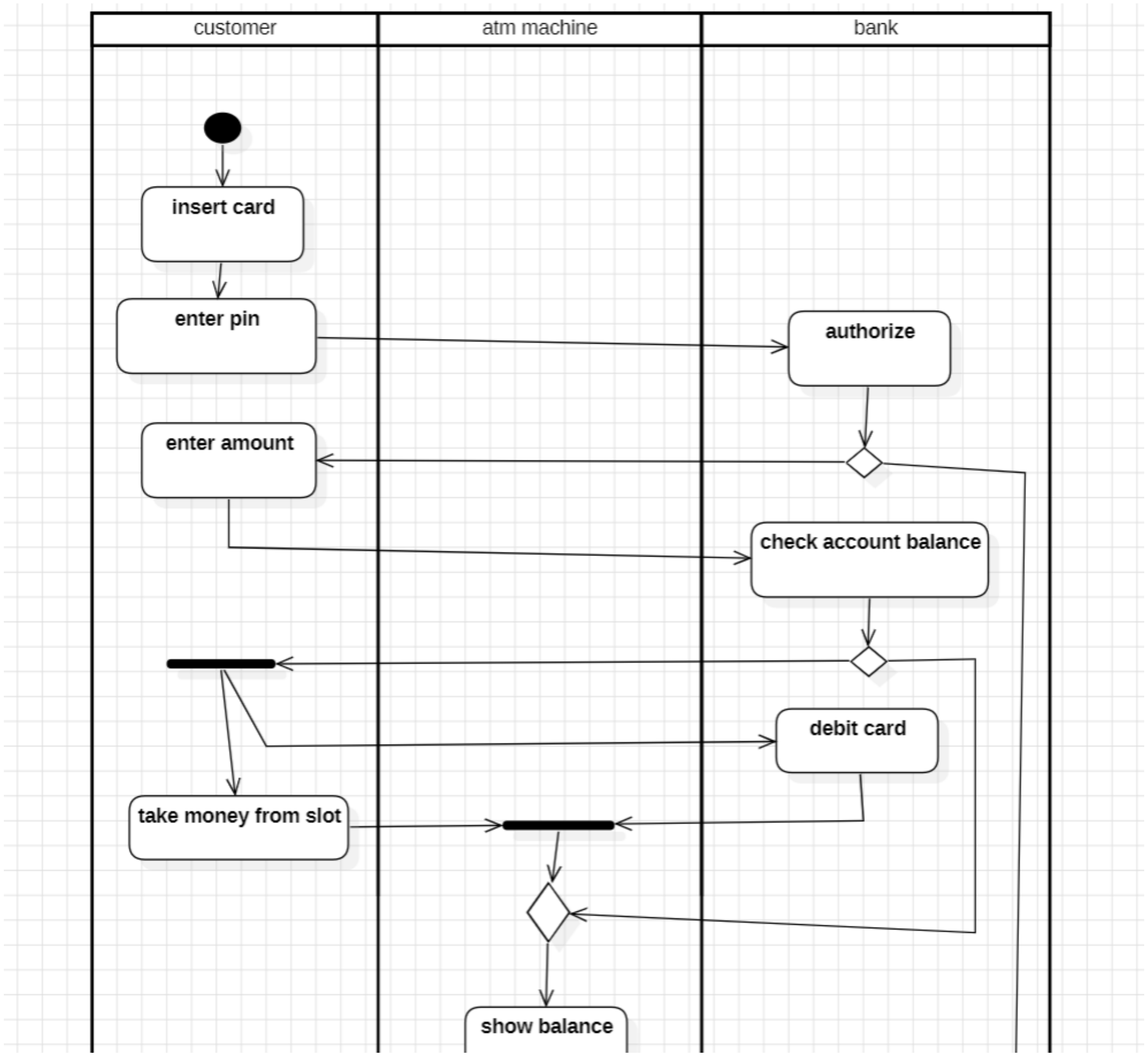
## 1.c) Sequence Diagram:



## 1.d) Object Diagram:

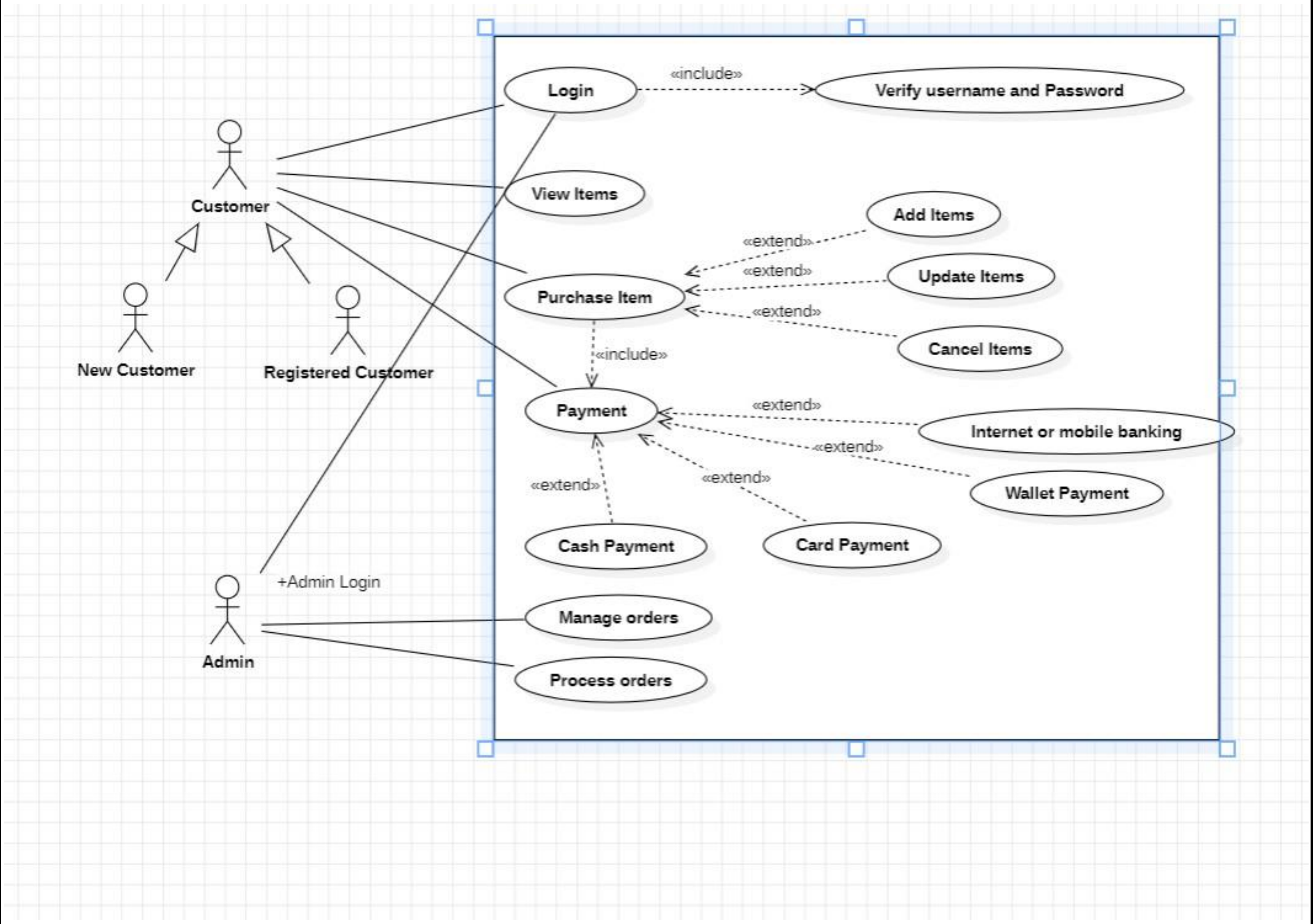


## 1.e) State-Activity Diagram:

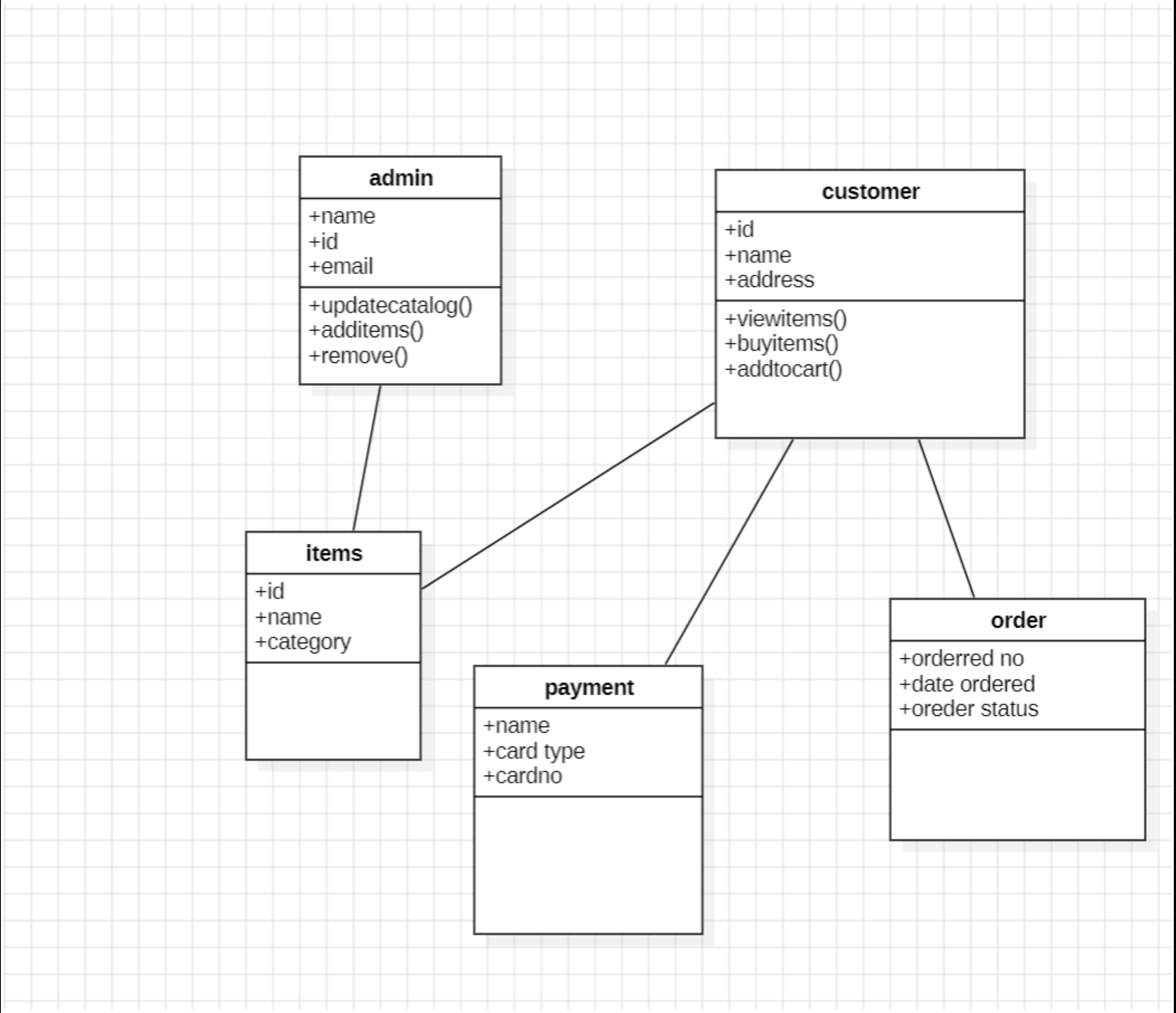


# 2. ONLINE SHOPPING

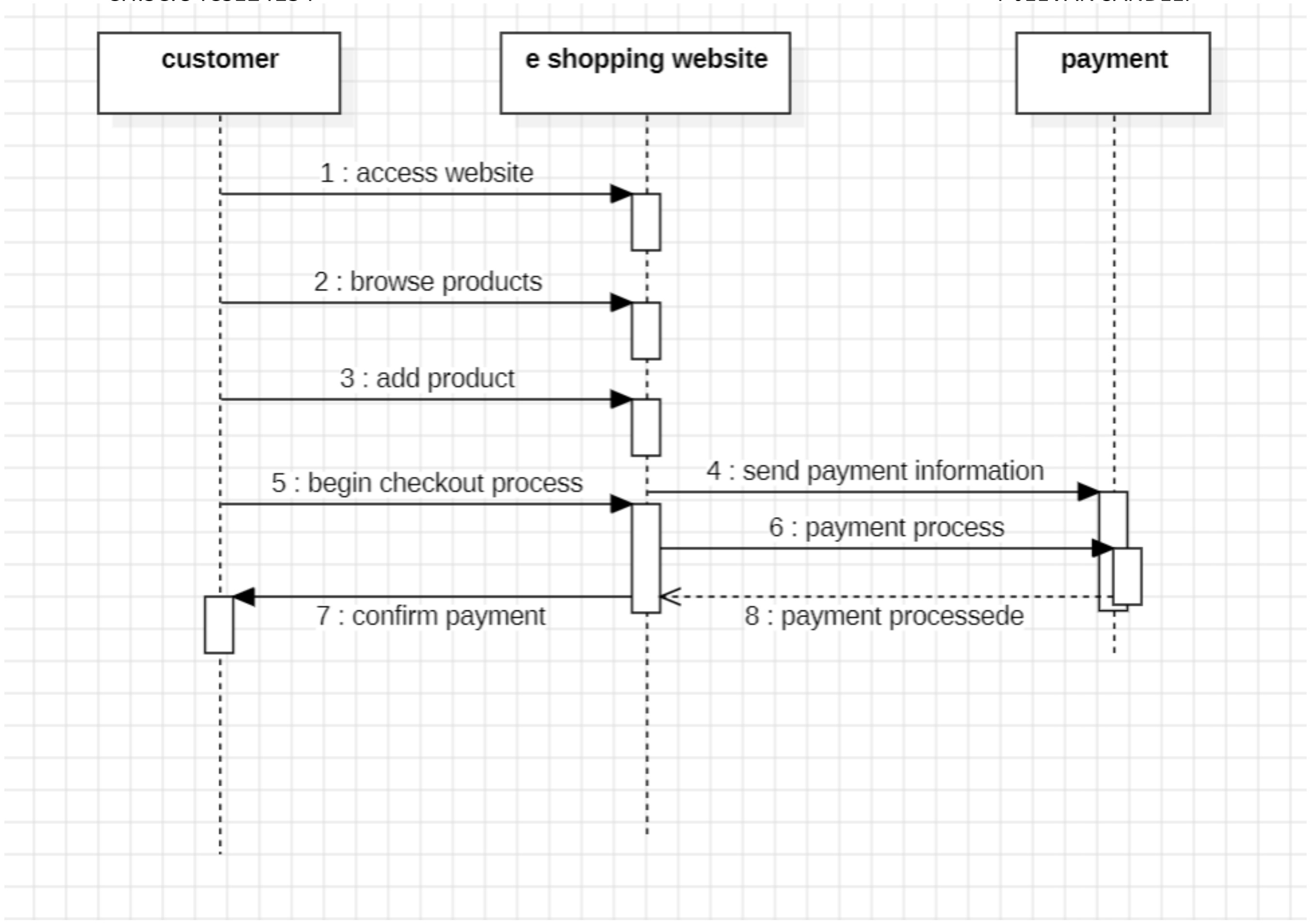
## 2.a) Use Case Diagram:



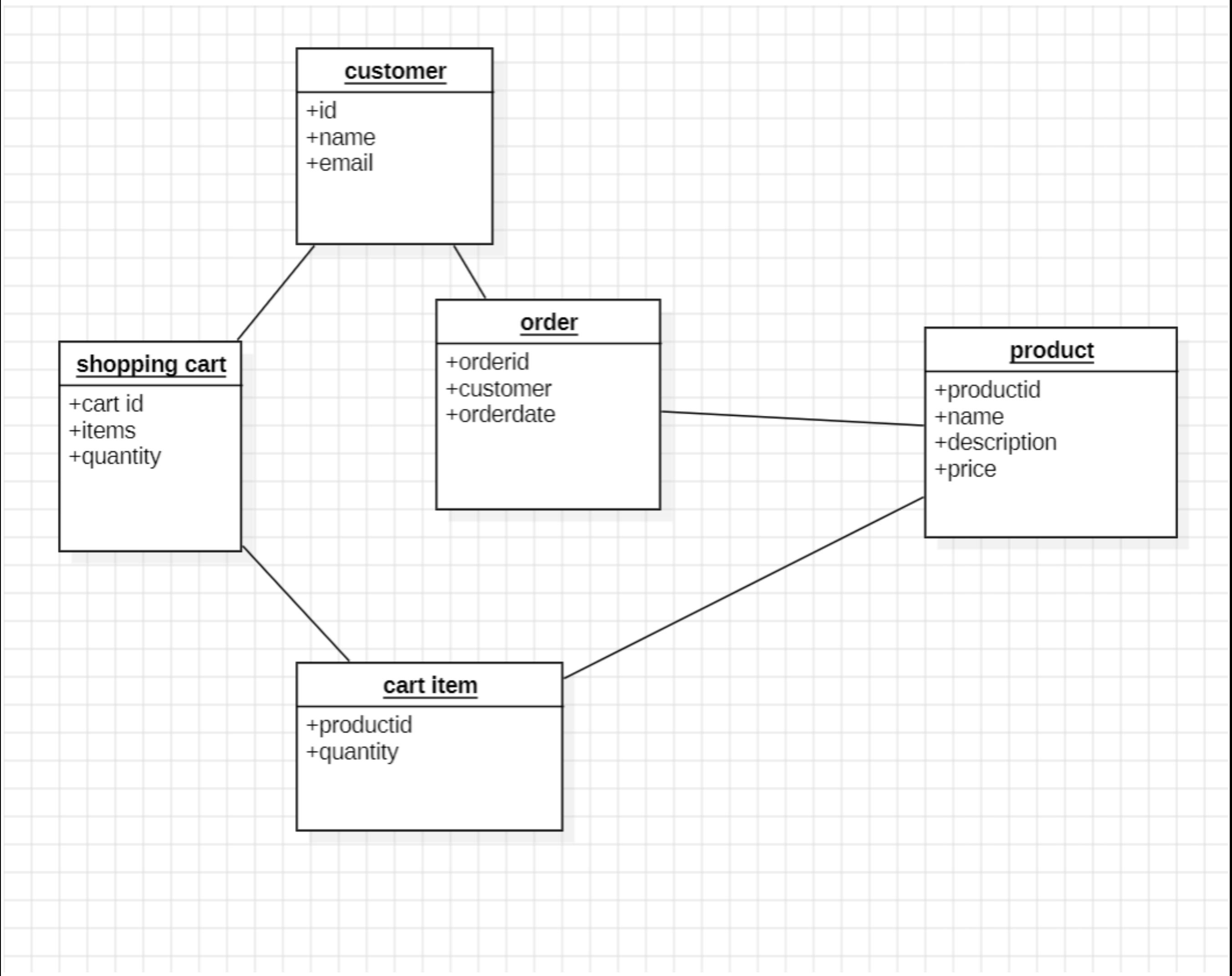
## 2.b) Class Diagram:



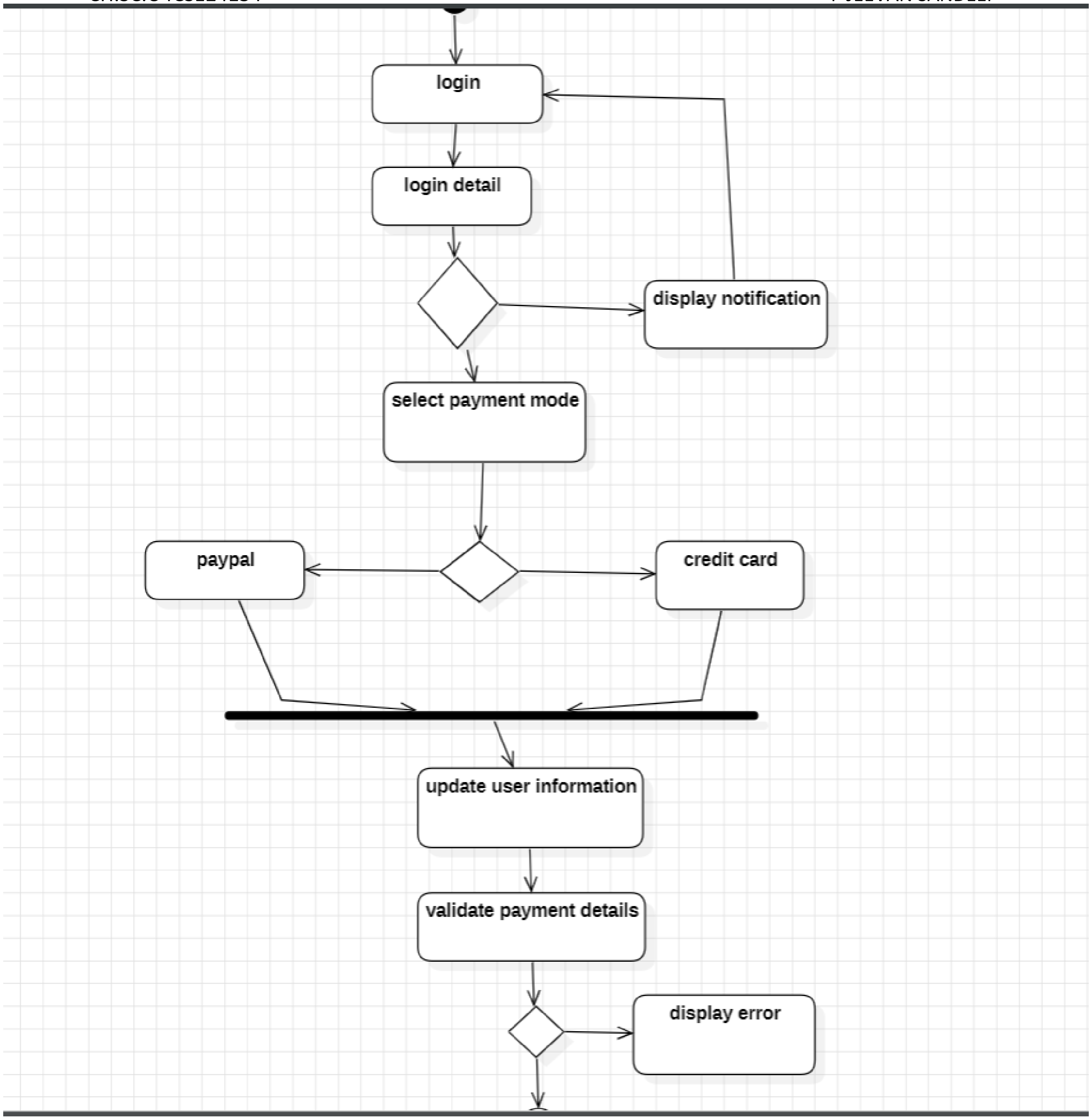
## 2.c) Sequence Diagram:


customer
e shopping website
payment
1 : access website
2 : browse products
3 : add product
4 : send payment information
5 : begin checkout process
6 : payment process
7 : confirm payment
8 : payment processede

## 2.d) Object Diagram:



## 2.e) State-Activity Diagram:


login
login detail
display notification
select payment mode
paypal
credit card
update user information
validate payment details
display error

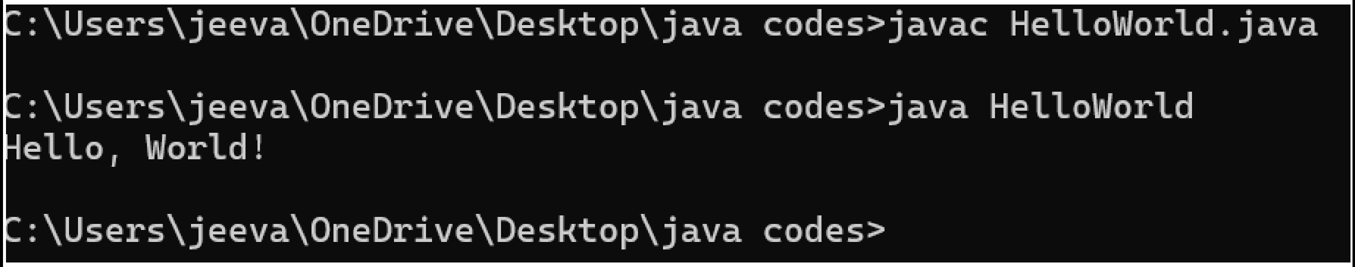
# 3. Basic Java Programs

## 3.a) Hello world

**Code:**

```
public class HelloWorld {
  public static void main(String[] args) {
         System.out.println("Hello, World!");
             }
           }
```

**Output:**



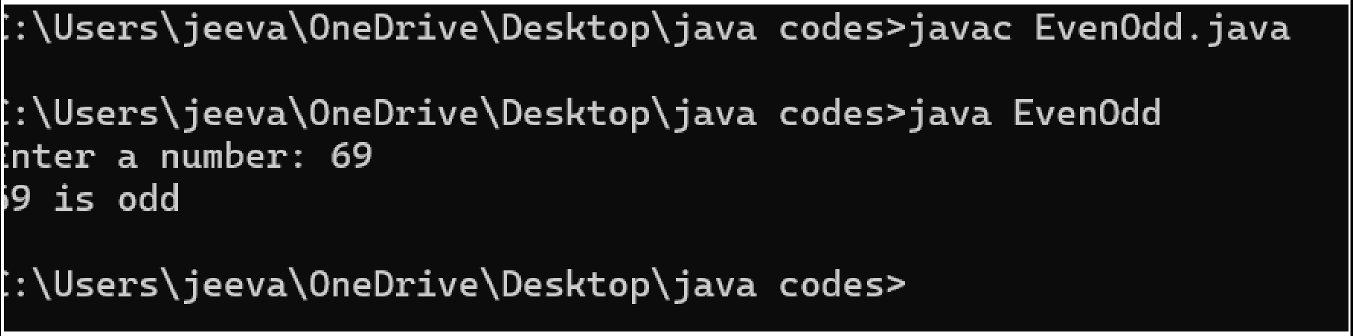
```
C:\Users\jeeva\OneDrive\Desktop\java codes>javac HelloWorld.java

C:\Users\jeeva\OneDrive\Desktop\java codes>java HelloWorld
Hello, World!

C:\Users\jeeva\OneDrive\Desktop\java codes>
```

## 3.b) Even Odd

**Code:**

```
                import java.util.Scanner;

public class EvenOdd {

    public static void main(String[] args) {

        Scanner reader = new Scanner(System.in);

        System.out.print("Enter a number: ");
        int num = reader.nextInt();

        if(num % 2 == 0)
            System.out.println(num + " is even");
        else
            System.out.println(num + " is odd");
    }
}
```

**Output:**



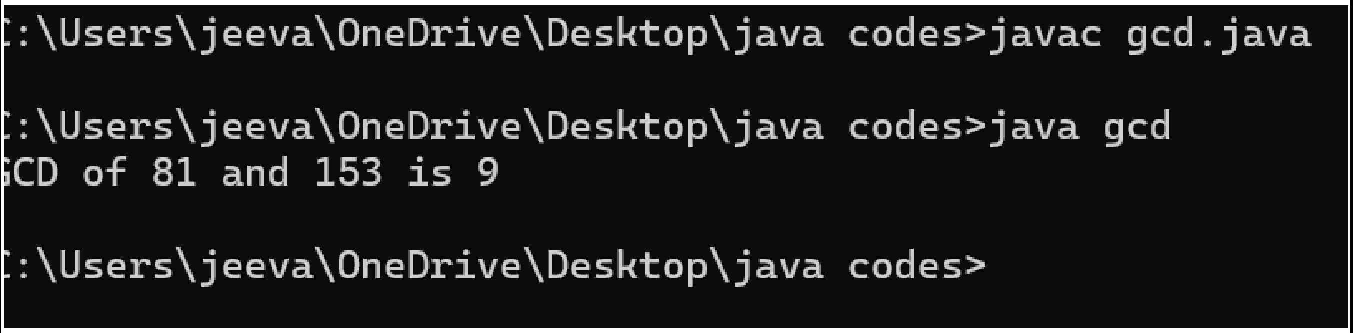
```
:\Users\jeeva\OneDrive\Desktop\java codes>javac EvenOdd.java

:\Users\jeeva\OneDrive\Desktop\java codes>java EvenOdd
nter a number: 69
9 is odd

:\Users\jeeva\OneDrive\Desktop\java codes>
```

## 3.c) Gcd:

**Code:**

```
class gcd {
  public static void main(String[] args) {
    int n1 = 81, n2 = 153;
    int gcd = 1;

    for (int i = 1; i <= n1 && i <= n2; ++i) {
      if (n1 % i == 0 && n2 % i == 0)
        gcd = i;
    }

    System.out.println("GCD of " + n1 +" and " + n2 + " is " + gcd);
  }
}
```

**Output:**



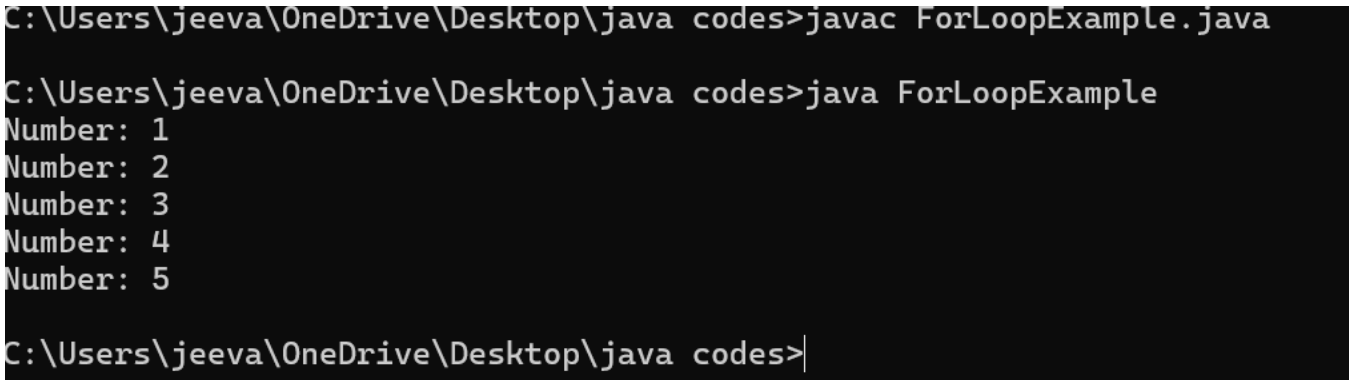
```
:\Users\jeeva\OneDrive\Desktop\java codes>javac gcd.java

:\Users\jeeva\OneDrive\Desktop\java codes>java gcd
CD of 81 and 153 is 9

:\Users\jeeva\OneDrive\Desktop\java codes>
```

## 3.d) For loop

**Code:**

```
        public class ForLoopExample {
    public static void main(String[] args) {
        for (int i = 1; i <= 5; i++) {
            System.out.println("Number: " + i);
        }
    }
}
```

**Output;**



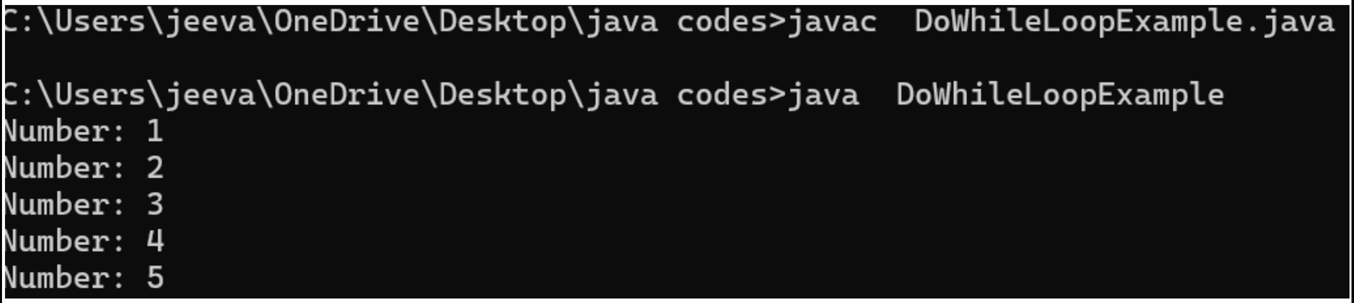
```
C:\Users\jeeva\OneDrive\Desktop\java codes>javac ForLoopExample.java

C:\Users\jeeva\OneDrive\Desktop\java codes>java ForLoopExample
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5

C:\Users\jeeva\OneDrive\Desktop\java codes>
```

## 3.e) : DoWhileLoopExample

**Code:**

```
 public class DoWhileLoopExample {
    public static void main(String[] args) {
     int i = 1;
       do {
          System.out.println("Number: " + i);
        i++;
         } while (i <= 5);
               }
          }
```

**Output:**



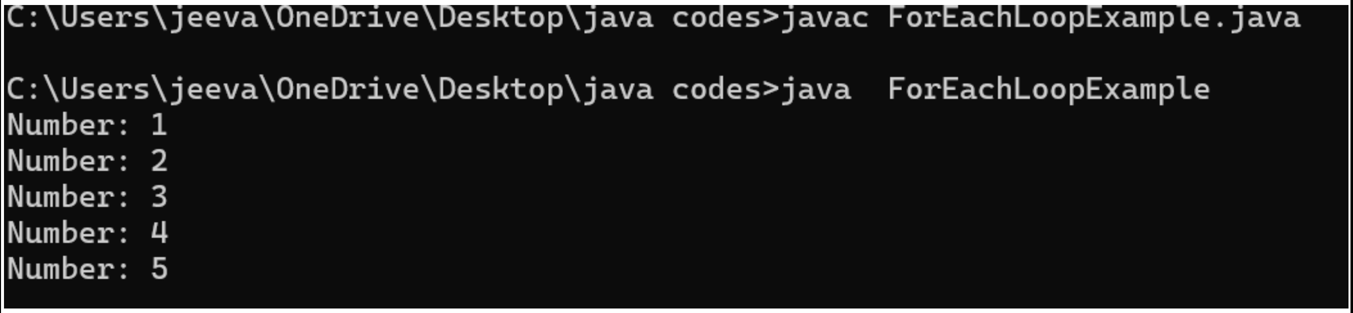
```
C:\Users\jeeva\OneDrive\Desktop\java codes>javac  DoWhileLoopExample.java

C:\Users\jeeva\OneDrive\Desktop\java codes>java  DoWhileLoopExample
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5
```

## 3.f) ForEachLoopExample

**Code:**

```
    public class ForEachLoopExample {
         public static void main(String[] args) {
               int[] numbers = {1, 2, 3, 4, 5};
                  for (int num : numbers) {
          System.out.println("Number: " + num);
      }
 }
}
```

**Output:**



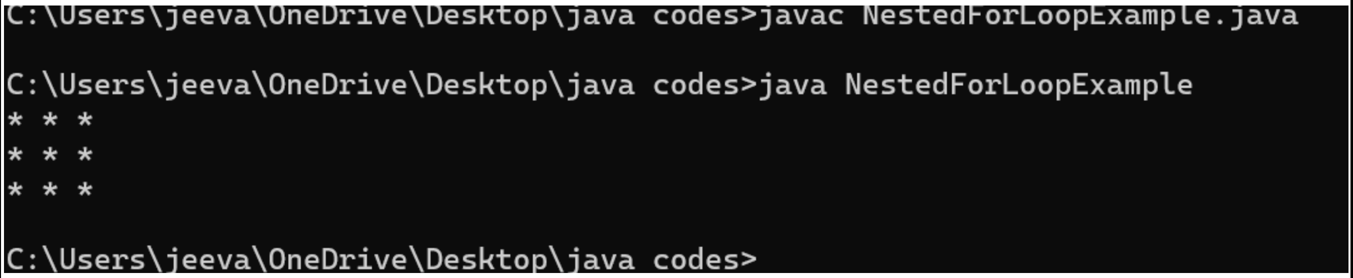
```
C:\Users\jeeva\OneDrive\Desktop\java codes>javac ForEachLoopExample.java

C:\Users\jeeva\OneDrive\Desktop\java codes>java  ForEachLoopExample
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5
```

## 3.g) NestedForLoopExample

**Code:**

```
 public class NestedForLoopExample {
    public static void main(String[] args) {
        for (int i = 1; i <= 3; i++) {
            for (int j = 1; j <= 3; j++) {
                System.out.print("* ");
            }
            System.out.println();
        }
    }
}
```

**Output:**



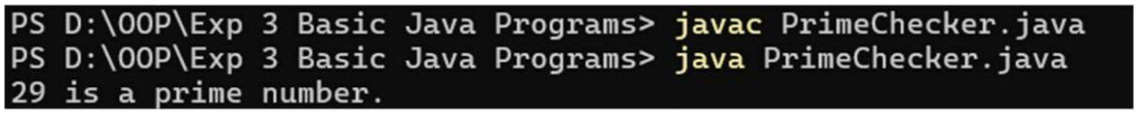
```
C:\Users\jeeva\OneDrive\Desktop\java codes>javac NestedForLoopExample.java

C:\Users\jeeva\OneDrive\Desktop\java codes>java NestedForLoopExample
* * *
* * *
* * *

C:\Users\jeeva\OneDrive\Desktop\java codes>
```

## 3.f) Prime Checker:

**Code:**

```
public class PrimeChecker {
    public static void main(String[] args) { int num =
        29;
        boolean isPrime = true; if
        (num <= 1) {
            isPrime = false;
        } else {
            for (int i = 2; i * i <= num; i++) { // Removed
Math.sqrt()
                if (num % i == 0) {
                    isPrime = false;
                    break;
                }
            }
        }
        if (isPrime) {
            System.out.println(num + " is a prime number.");
        } else {
            System.out.println(num + " is not a prime number.");
        }
    }
}
```

**Output:**



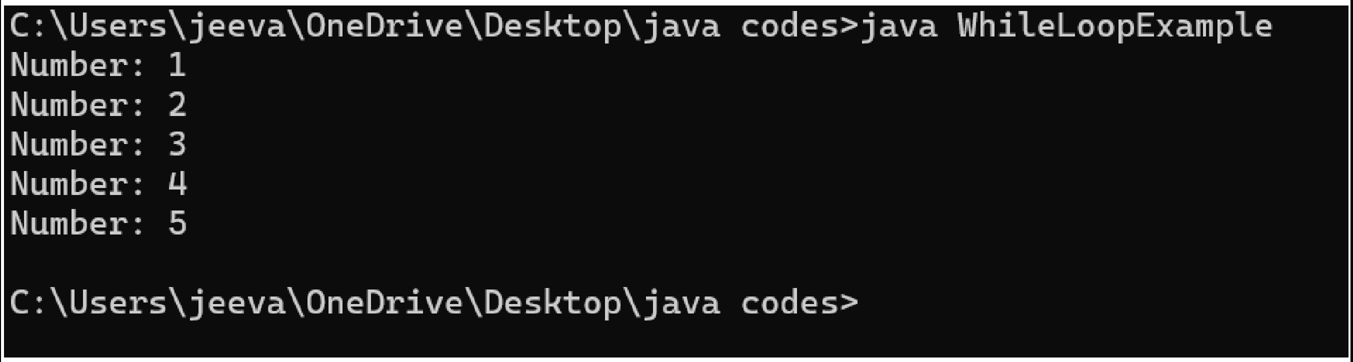
```
PS D:\OOP\Exp 3 Basic Java Programs> javac PrimeChecker.java
PS D:\OOP\Exp 3 Basic Java Programs> java PrimeChecker.java
29 is a prime number.
```

## 3.g) WhileLoopExample:

**Code:**

```
public class WhileLoopExample {
    public static void main(String[] args) {
        int i = 1;
        while (i <= 5) {
            System.out.println("Number: " + i);
            i++;
        }
    }
}
```

**Output:**



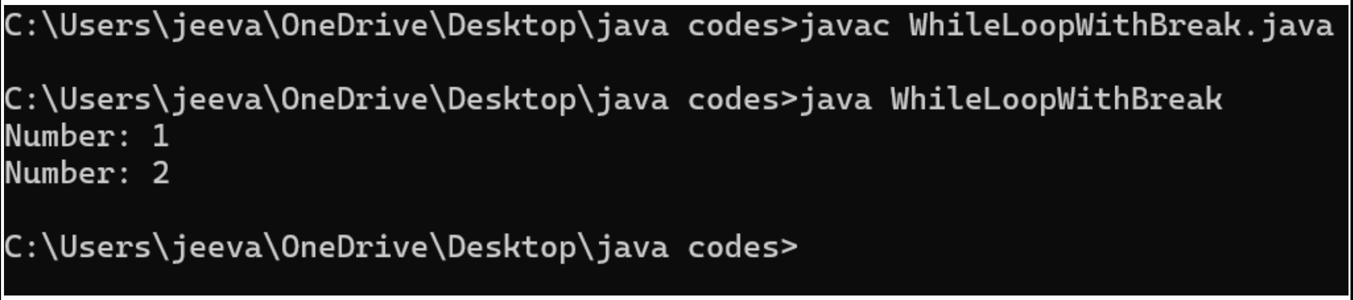
```
C:\Users\jeeva\OneDrive\Desktop\java codes>java WhileLoopExample
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5

C:\Users\jeeva\OneDrive\Desktop\java codes>
```

## 3.h) WhileLoopWithBreak:

**Code:**

```
 public class WhileLoopWithBreak {
    public static void main(String[] args) {
        int i = 1;
        while (i <= 5) {
            if (i == 3) {
                break;
            }
            System.out.println("Number: " + i);
            i++;
        }
    }
}
```

**Output:**



```
C:\Users\jeeva\OneDrive\Desktop\java codes>javac WhileLoopWithBreak.java

C:\Users\jeeva\OneDrive\Desktop\java codes>java WhileLoopWithBreak
Number: 1
Number: 2

C:\Users\jeeva\OneDrive\Desktop\java codes>
```

## INHERITANCE

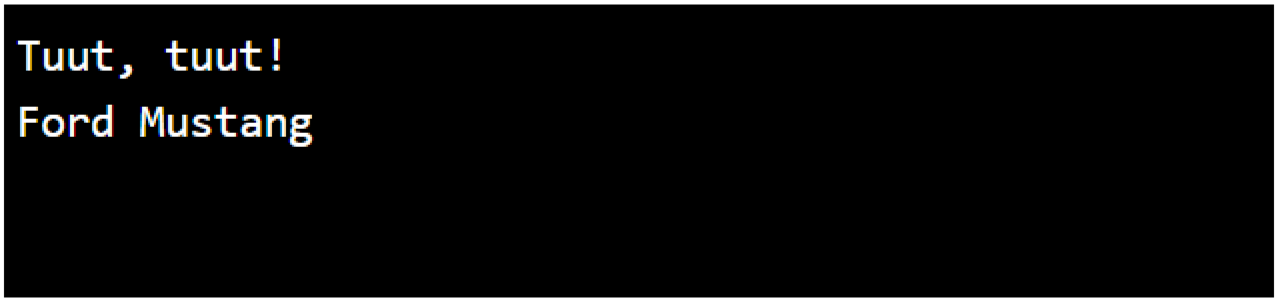
### 4)SINGLE INHERITANCE PROGRAMS

4a) cars

Code:

```
class Vehicle {
  protected String brand = "Ford";
  public void honk() {
    System.out.println("Tuut, tuut!");
  }
}

class Car extends Vehicle {
  private String modelName = "Mustang";
  public static void main(String[] args) {

    Car myCar = new Car();

    myCar.honk();
    System.out.println(myCar.brand + " " + myCar.modelName);
  }
}
```

**Output:**



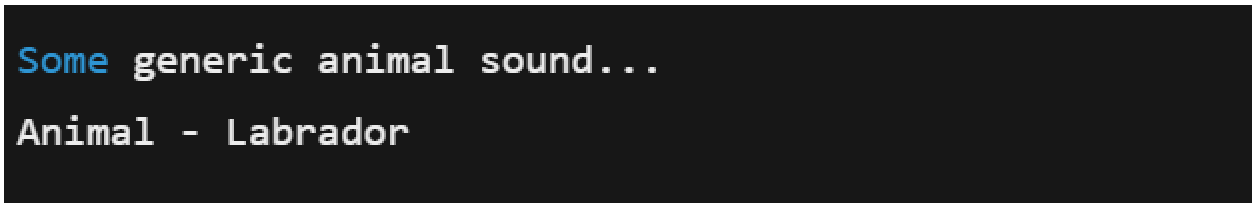
```
Tuut, tuut!
Ford Mustang
```

**4b) animal sound**

**Code:**

```
class Animal {
  protected String name = "Animal";

  public void makeSound() {
    System.out.println("Some generic animal sound...");
  }
}

class Dog extends Animal {
  private String breed = "Labrador";

  public static void main(String[] args) {
    Dog myDog = new Dog();

    myDog.makeSound();
    System.out.println(myDog.name + " - " + myDog.breed);
  }
}
```

Output:



```
Some generic animal sound...
Animal - Labrador
```

**5) MULTILEVEL INHERITANCE PROGRAMS**

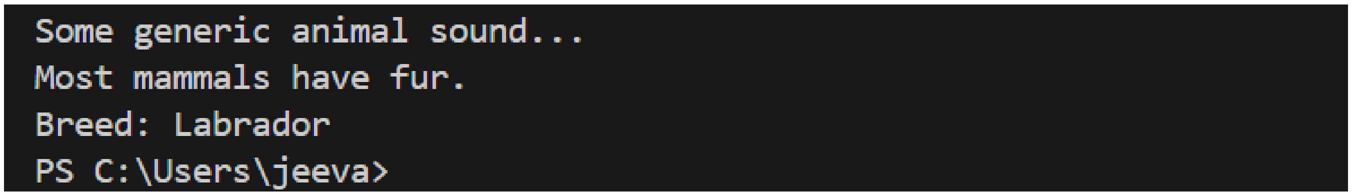
**5a) animals**

**Code:**

```
class Animal {
    protected String name = "Animal";

    public void makeSound() {
      System.out.println("Some generic animal sound...");
    }
  }

  class Mammal extends Animal {
    public void hasFur() {
      System.out.println("Most mammals have fur.");
    }
  }

  class Dog extends Mammal {
```

```
private String breed = "Labrador";

    public static void main(String[] args) {
      Dog myDog = new Dog();

      myDog.makeSound();  // Inherited from Animal
      myDog.hasFur();     // Inherited from Mammal
      System.out.println("Breed: " + myDog.breed);
    }
  }
```

**Output:**



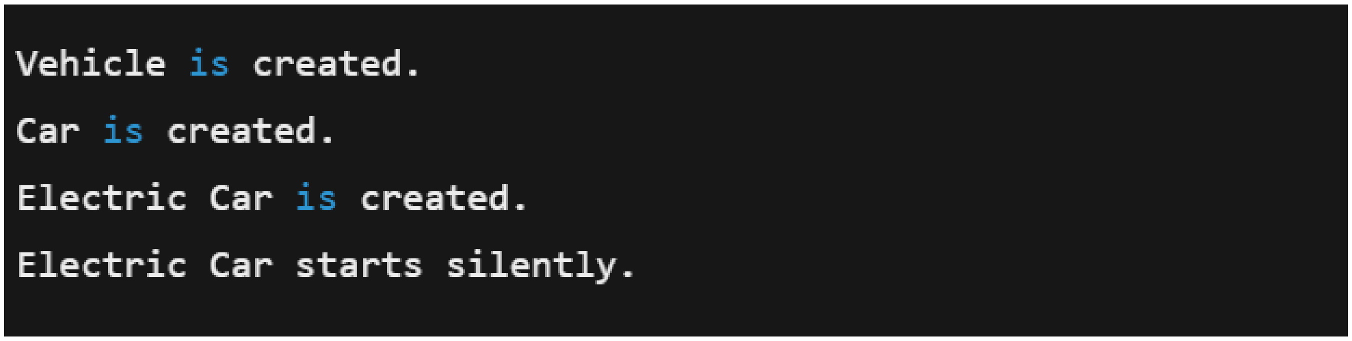
```
Some generic animal sound...
Most mammals have fur.
Breed: Labrador
PS C:\Users\jeeva>
```

**5b)** vehicleHierarchy

**Code:**

```
class Vehicle {
   Vehicle() {
      System.out.println("Vehicle is created.");
   }

   void start() {
        System.out.println("Vehicle is starting...");
    }
}


class Car extends Vehicle {
    Car() {
        System.out.println("Car is created.");
    }

    @Override
    void start() {
        System.out.println("Car engine starts with a key.");
    }
}


class ElectricCar extends Car {
    ElectricCar() {
        System.out.println("Electric Car is created.");
    }

    @Override
    void start() {
        System.out.println("Electric Car starts silently.");
    }
}


public class MultilevelInheritance1 {
    public static void main(String[] args) {
        ElectricCar myCar = new ElectricCar();

        myCar.start();
    }
}
```

**Output:**



```
Vehicle is created.

Car is created.

Electric Car is created.

Electric Car starts silently.
```

**6) HIERARCHICAL INHERITANCE PROGRAMS**

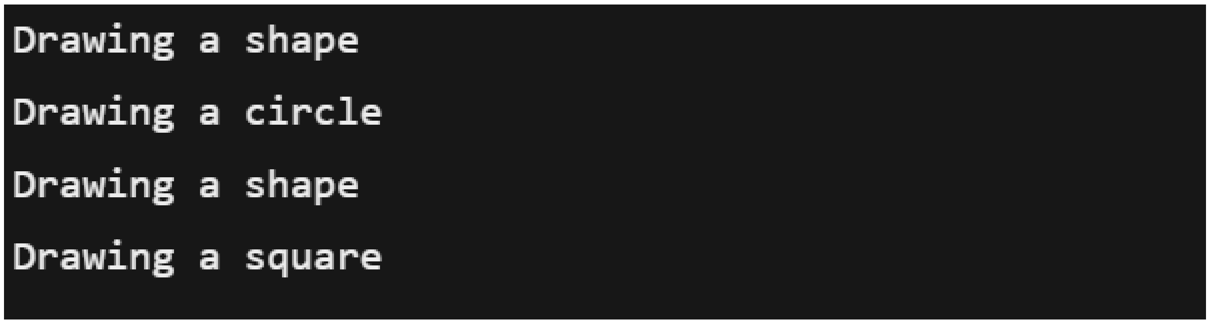
**6a) SHAPES**

**Code:**

```
class Shape {
   void draw() {
      System.out.println("Drawing a shape");
   }
}
class Circle extends Shape {
   void drawCircle() {
      System.out.println("Drawing a circle");
   }
}
class Square extends Shape {
   void drawSquare() {
      System.out.println("Drawing a square");
   }
}
public class Main {
   public static void main(String[] args) {
      Circle circle = new Circle();
```

```
        Square square = new Square();
        circle.draw();
        circle.drawCircle();
        square.draw();
        square.drawSquare();
    }
}
```

**Output:**



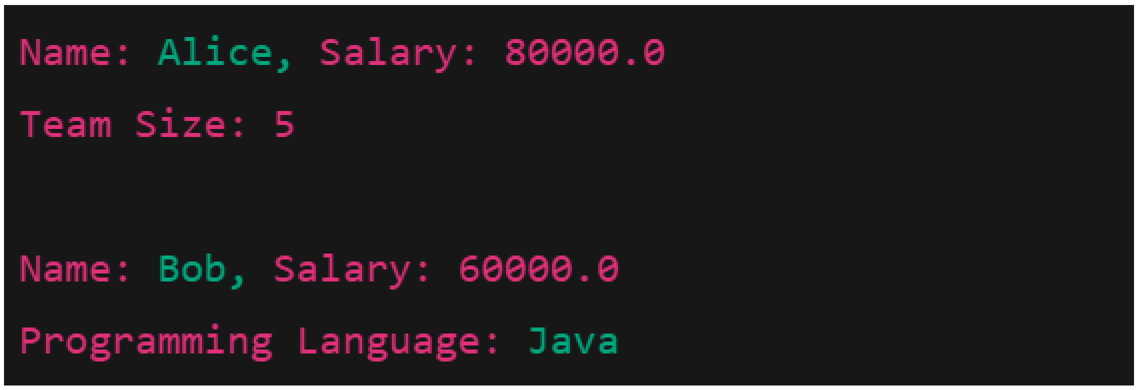
```
Drawing a shape
Drawing a circle
Drawing a shape
Drawing a square
```

## 6b) EMPLOYEE

**Code:**

```
class Employee {
    String name;
    double salary;
    Employee(String name, double salary) {
        this.name = name;
        this.salary = salary;
    }
    void displayInfo() {
        System.out.println("Name: " + name + ", Salary: " + salary);
    }
}
class Manager extends Employee {
   int teamSize;
   Manager(String name, double salary, int teamSize) {
      super(name, salary);
      this.teamSize = teamSize;
   }
   void showManagerDetails() {
      displayInfo();
      System.out.println("Team Size: " + teamSize);
   }
}
class Developer extends Employee {
   String programmingLanguage;
   Developer(String name, double salary, String programmingLanguage) {
      super(name, salary); // Call parent constructor
      this.programmingLanguage = programmingLanguage;
   }
   void showDeveloperDetails() {
      displayInfo();
      System.out.println("Programming Language: " + programmingLanguage);
   }
}
public class Main1 {
   public static void main(String[] args) {
      Manager manager = new Manager("Alice", 80000, 5);
      Developer developer = new Developer("Bob", 60000, "Java");
      manager.showManagerDetails();
      System.out.println();
      developer.showDeveloperDetails();
   }
}
```

**Output:**


```
Name: Alice, Salary: 80000.0
Team Size: 5

Name: Bob, Salary: 60000.0
Programming Language: Java
```

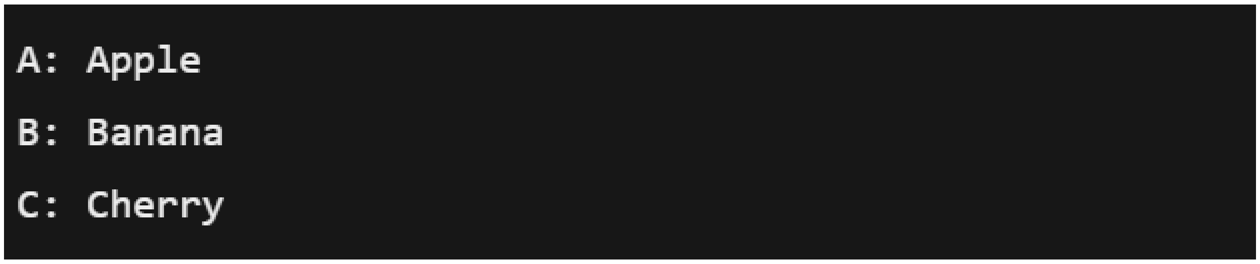
**7) HYBRID INHERITANCE PROGRAMS**

**7a) FRUIT**

**Code:**

```
import java.util.HashMap;
import java.util.LinkedList;
import java.util.Map;

public class HybridDataStructure {
  private Map<String, String> map;
  private LinkedList<String> order;

  public HybridDataStructure() {
    map = new HashMap<>();
    order = new LinkedList<>();
  }
  public void add(String key, String value) {
    if (!map.containsKey(key)) {
            order.add(key);
        }
        map.put(key, value);
    }
    public String get(String key) {
        return map.get(key);
    }
    public void printInsertionOrder() {
        for (String key : order) {
            System.out.println(key + ": " + map.get(key));
        }
    }
    public static void main(String[] args) {
        HybridDataStructure hybrid = new HybridDataStructure();
        hybrid.add("A", "Apple");
        hybrid.add("B", "Banana");
        hybrid.add("C", "Cherry");
        hybrid.printInsertionOrder();
    }
}
```

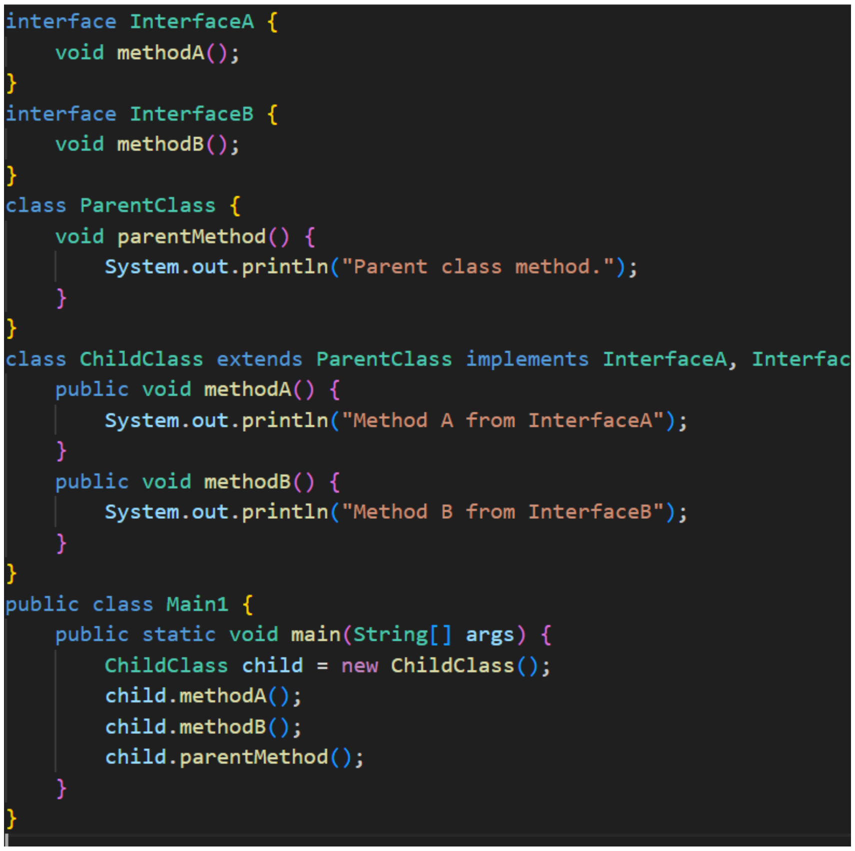
**Output:**



```
A: Apple
B: Banana
C: Cherry
```

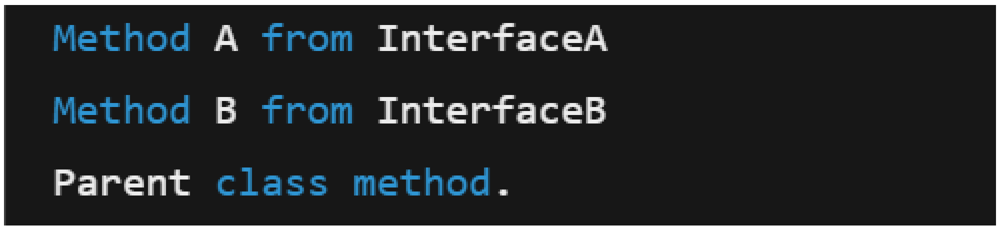
## 7b) CHILD

**Code:**



```java
interface InterfaceA {
    void methodA();
}
interface InterfaceB {
    void methodB();
}
class ParentClass {
    void parentMethod() {
        System.out.println("Parent class method.");
    }
}
class ChildClass extends ParentClass implements InterfaceA, Interfac
    public void methodA() {
        System.out.println("Method A from InterfaceA");
    }
    public void methodB() {
        System.out.println("Method B from InterfaceB");
    }
}
public class Main1 {
    public static void main(String[] args) {
        ChildClass child = new ChildClass();
        child.methodA();
        child.methodB();
        child.parentMethod();
    }
}
```

**Output:**



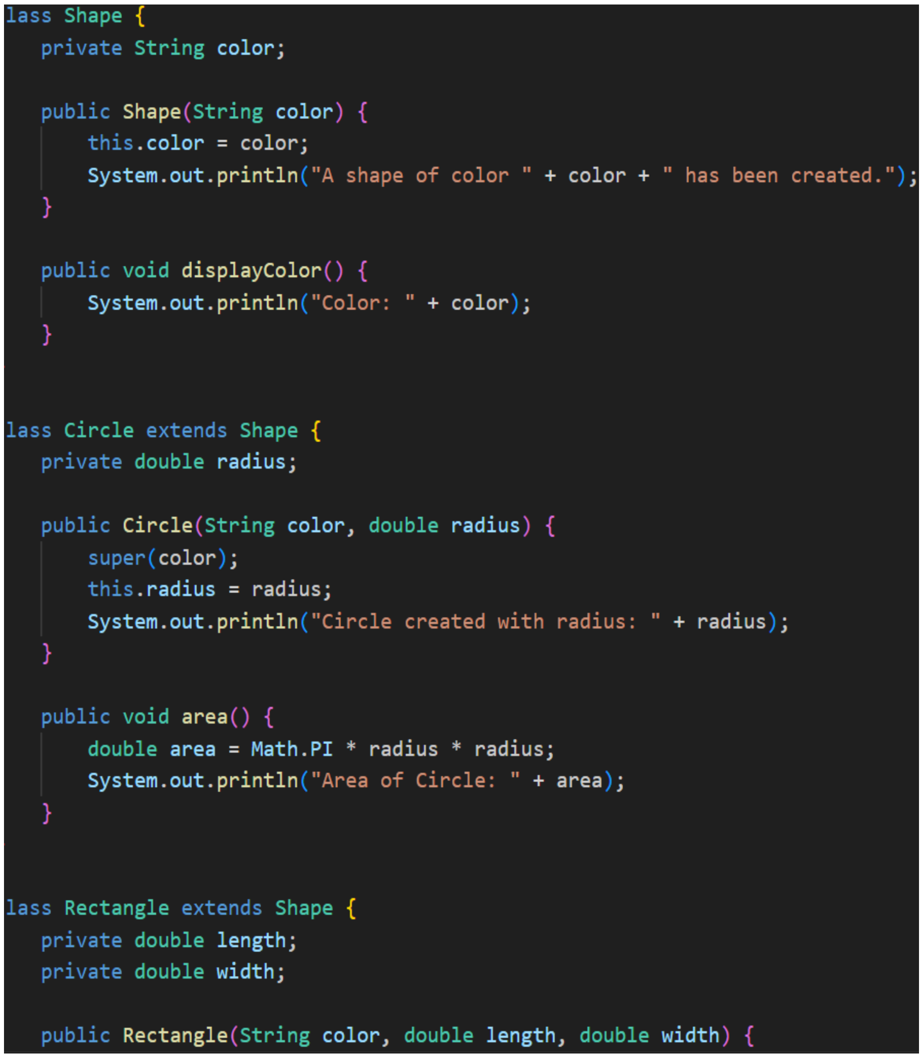
```
Method A from InterfaceA
Method B from InterfaceB
Parent class method.
```

**POLYMORPHISM**

## 8) CONSTRUCTOR PROGRAMS

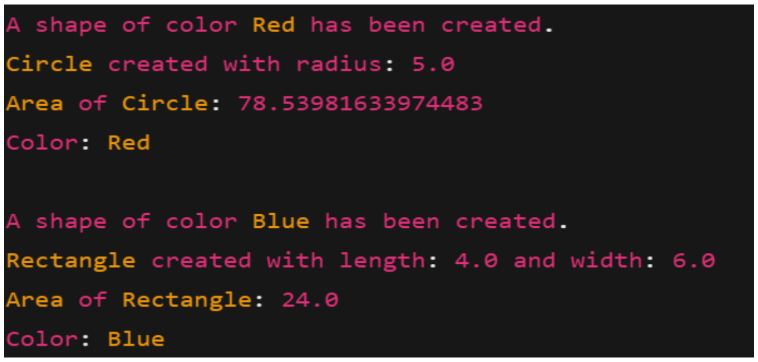
### a) BookDemo

### Code:



```
lass Shape {
   private String color;

   public Shape(String color) {
       this.color = color;
       System.out.println("A shape of color " + color + " has been created.");
   }

   public void displayColor() {
       System.out.println("Color: " + color);
   }


lass Circle extends Shape {
   private double radius;

   public Circle(String color, double radius) {
       super(color);
       this.radius = radius;
       System.out.println("Circle created with radius: " + radius);
   }

   public void area() {
       double area = Math.PI * radius * radius;
       System.out.println("Area of Circle: " + area);
   }


lass Rectangle extends Shape {
   private double length;
   private double width;

   public Rectangle(String color, double length, double width) {
```

## Output:

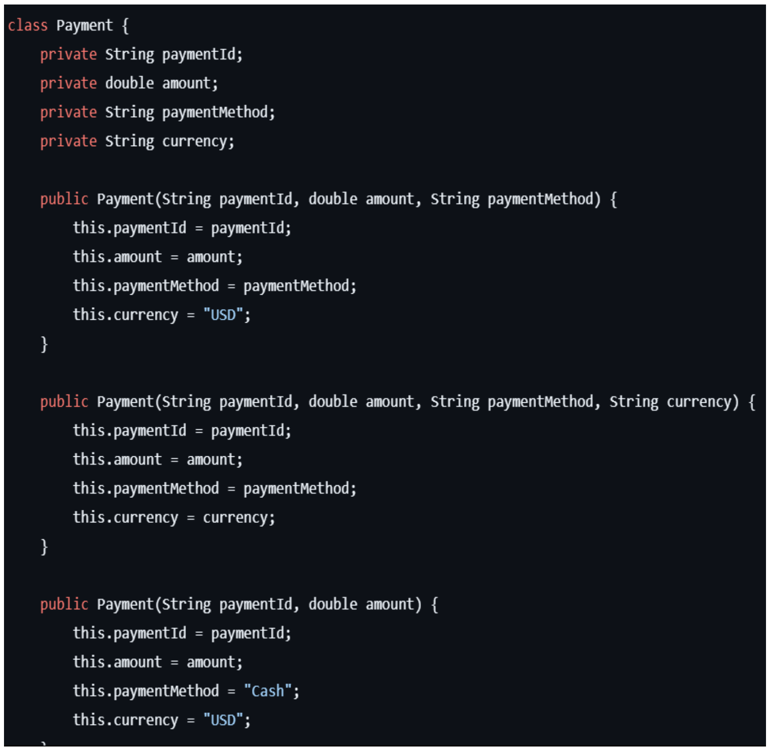


```
A shape of color Red has been created.
Circle created with radius: 5.0
Area of Circle: 78.53981633974483
Color: Red

A shape of color Blue has been created.
Rectangle created with length: 4.0 and width: 6.0
Area of Rectangle: 24.0
Color: Blue
```

## 9)CONSTRUCTOR OVERLOADING PROGRAMS

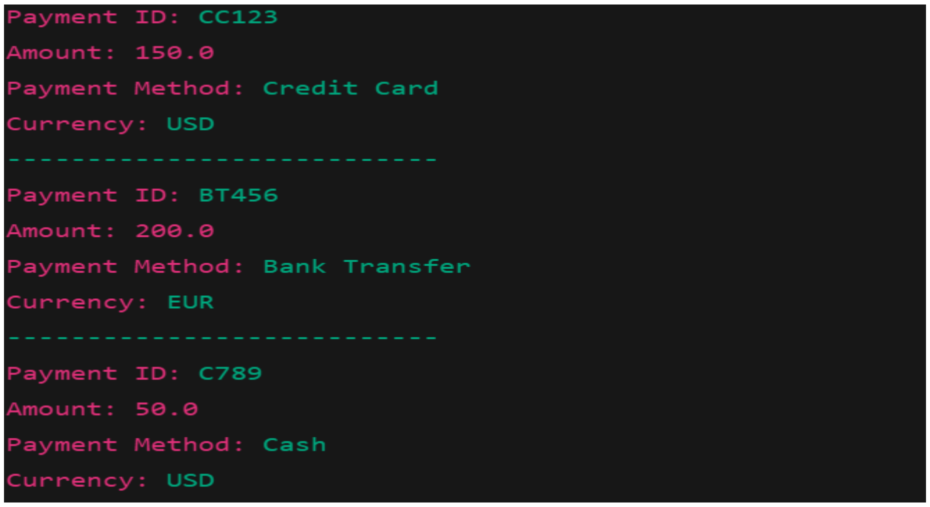
### 9.a)MovieTicket

### Code:



```java
class Payment {
    private String paymentId;
    private double amount;
    private String paymentMethod;
    private String currency;

    public Payment(String paymentId, double amount, String paymentMethod) {
        this.paymentId = paymentId;
        this.amount = amount;
        this.paymentMethod = paymentMethod;
        this.currency = "USD";
    }

    public Payment(String paymentId, double amount, String paymentMethod, String currency) {
        this.paymentId = paymentId;
        this.amount = amount;
        this.paymentMethod = paymentMethod;
        this.currency = currency;
    }

    public Payment(String paymentId, double amount) {
        this.paymentId = paymentId;
        this.amount = amount;
        this.paymentMethod = "Cash";
        this.currency = "USD";
    }
```

## Output:

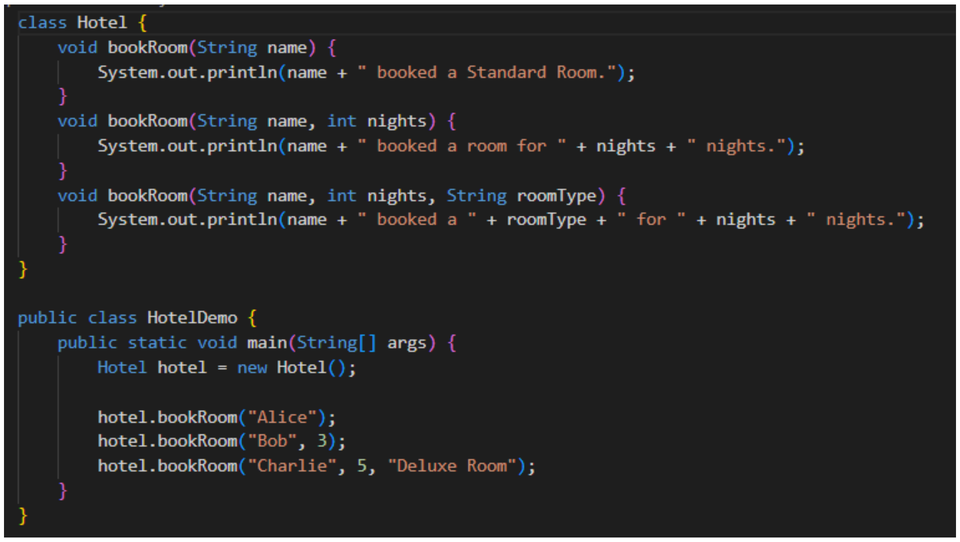


```
Payment ID: CC123
Amount: 150.0
Payment Method: Credit Card
Currency: USD
---------------------------
Payment ID: BT456
Amount: 200.0
Payment Method: Bank Transfer
Currency: EUR
---------------------------
Payment ID: C789
Amount: 50.0
Payment Method: Cash
Currency: USD
```

## 10)METHOD OVERLOADING PROGRAMS

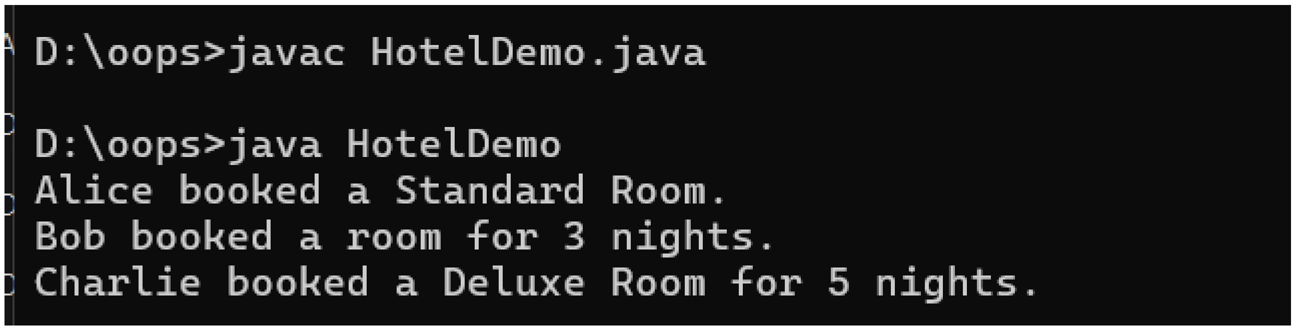
### 10.a) HotelDemo

## Code:



```java
class Hotel {
    void bookRoom(String name) {
        System.out.println(name + " booked a Standard Room.");
    }
    void bookRoom(String name, int nights) {
        System.out.println(name + " booked a room for " + nights + " nights.");
    }
    void bookRoom(String name, int nights, String roomType) {
        System.out.println(name + " booked a " + roomType + " for " + nights + " nights.");
    }
}

public class HotelDemo {
    public static void main(String[] args) {
        Hotel hotel = new Hotel();

        hotel.bookRoom("Alice");
        hotel.bookRoom("Bob", 3);
        hotel.bookRoom("Charlie", 5, "Deluxe Room");
    }
}
```

## Output:



```
D:\oops>javac HotelDemo.java

D:\oops>java HotelDemo
Alice booked a Standard Room.
Bob booked a room for 3 nights.
Charlie booked a Deluxe Room for 5 nights.
```

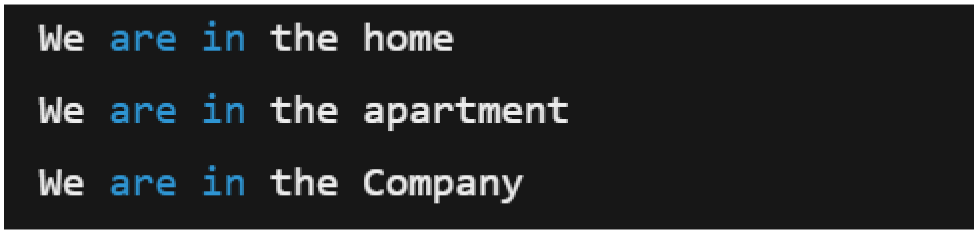
**10.b) HOME**

**Code:**

```
class Home {
   public void display() {
      System.out.println("We are in the home");
   }
}

class Apartment extends Home {
   public void display() {
      System.out.println("We are in the apartment");
   }
}

class Company extends Home {
   public void display() {
      System.out.println("We are in the Company");
```

```
    }
}

public class Room {
    public static void main(String[] args) {
        Home myHome = new Home();
        Home myApartment = new Apartment();
        Company myObj = new Company();

        myHome.display();
        myApartment.display();
        myObj.display();
    }
}
```

**Output:**



```
We are in the home
We are in the apartment
We are in the Company
```

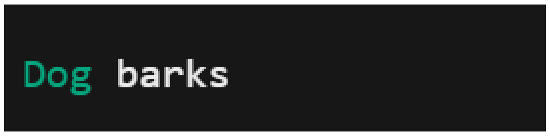
## 11) METHOD OVERRIDING PROGRAMS

### 11.a) ANIMAL

**Code:**

```
class Animal {
   public void makeSound() {
      System.out.println("Animal makes a sound");
   }
}
class Dog extends Animal {
   @Override
   public void makeSound() {
      System.out.println("Dog barks");
   }
}

public class Main {
   public static void main(String[] args) {
      Animal animal = new Dog();
      animal.makeSound();
   }
}
```

**Output:**



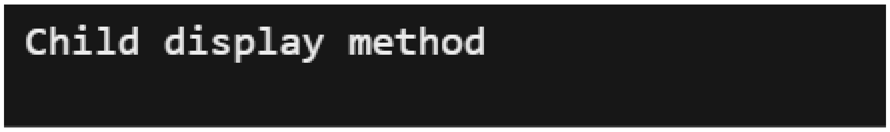
```
Dog barks
```

**11.b) PARENT**

**Code:**

```
class Parent {
   protected void display() {
      System.out.println("Parent display method");
   }
}
class Child extends Parent {
   @Override
   public void display() {
      System.out.println("Child display method");
   }
}
public class Main3 {
   public static void main(String[] args) {
      Parent obj = new Child();
      obj.display();
   }
}
```

**Output:**



```
Child display method
```

## ABSTRACTION

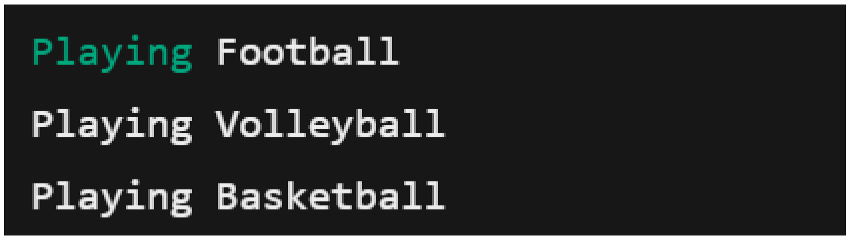
### 12) INTERFACE PROGRAMS

### 12a)GAMES

**Code:**

```
interface Playable {
   void play();
}

class Football implements Playable {
   public void play() {
      System.out.println("Playing Football");
   }
}

class Volleyball implements Playable {
   public void play() {
      System.out.println("Playing Volleyball");
   }
}

class Basketball implements Playable {
   public void play() {
      System.out.println("Playing Basketball");
   }
}

public class Main {
   public static void main(String[] args) {
      Playable football = new Football();
      Playable volleyball = new Volleyball();
      Playable basketball = new Basketball();
      football.play();
      volleyball.play();
      basketball.play();
   }
}
```

**Output:**



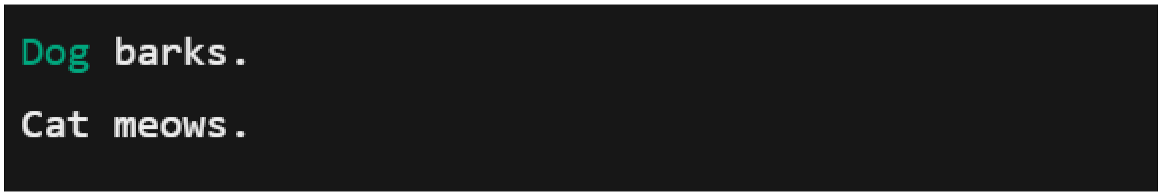
```
Playing Football
Playing Volleyball
Playing Basketball
```

**12b)ANIMAL**

**Code:**

```
interface Animal {
void makeSound();
}
class Dog implements Animal {
public void makeSound() {
System.out.println("Dog barks.");
}
}
class Cat implements Animal {
public void makeSound() {
System.out.println("Cat meows.");
}
}
public class Main {
public static void main(String[] args) {
Animal a1 = new Dog();
Animal a2 = new Cat();
a1.makeSound();
a2.makeSound();
}
}
```

**Output:**



```
Dog barks.
Cat meows.
```

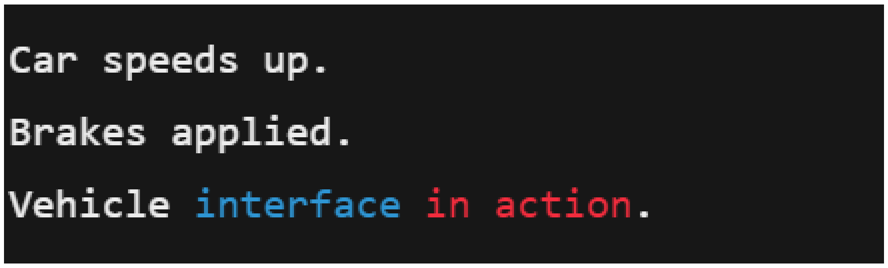
## 12c) VEHICLE

**Code:**

```
interface Vehicle {
void speedUp();
default void applyBrakes() {
System.out.println("Brakes applied.");
}
static void showMessage() {
System.out.println("Vehicle interface in action.");
}
}
class Car implements Vehicle {
public void speedUp() {
System.out.println("Car speeds up.");
}
}
public class Main {
```

```
public static void main(String[] args) {
Car c = new Car();
c.speedUp();
c.applyBrakes();
Vehicle.showMessage();
}
}
```

**Output:**



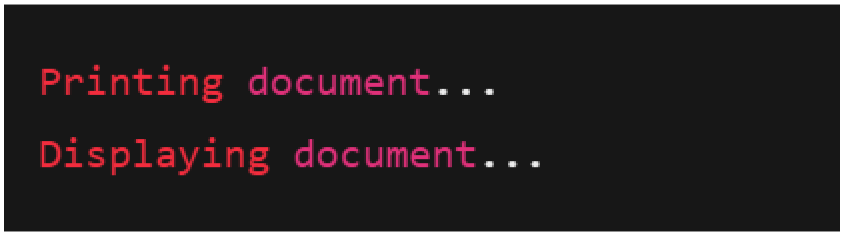
```
Car speeds up.
Brakes applied.
Vehicle interface in action.
```

**12d) PRINTING**

**Code:**

```
interface Printable {
void print();
}
interface Showable {
void show();
}
class Document implements Printable, Showable {
public void print() {
System.out.println("Printing document...");
}
public void show() {
System.out.println("Displaying document...");
}
}
public class Main {
public static void main(String[] args) {
Document doc = new Document();
doc.print();
doc.show();
}
}
```

**Output:**



```
Printing document...
Displaying document...
```

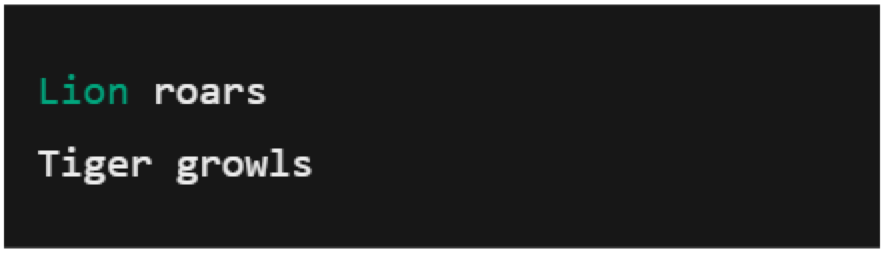
## 13) ABSTRACT CLASS PROGRAMS

### 13 a) Animal  SOUND

**Code:**

```
abstract class Animal {
  abstract void sound();
}

class Lion extends Animal {
  public void sound() {
    System.out.println("Lion roars");
  }
}

class Tiger extends Animal {
  public void sound() {
    System.out.println("Tiger growls");
  }
}

public class AnimalSoundTest {
  public static void main(String[] args) {
    Animal lion = new Lion();
    Animal tiger = new Tiger();
    lion.sound();
    tiger.sound();
  }
}
```

## Output:



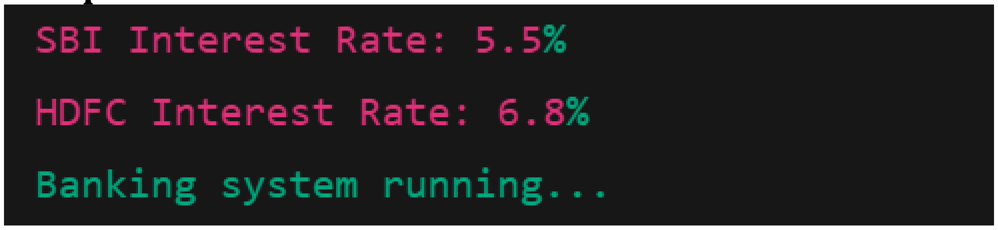
```
Lion roars
Tiger growls
```

## 13 b) BankAccount

## Code:

```
.abstract class Bank {
abstract double getInterestRate();
void display() {
System.out.println("Banking system running...");
}
}
class SBI extends Bank {
double getInterestRate() {
return 5.5;
}
}
class HDFC extends Bank {
double getInterestRate() {
return 6.8;
}
}
public class Main {
public static void main(String[] args) {
Bank b1 = new SBI();
Bank b2 = new HDFC();
System.out.println("SBI Interest Rate: " + b1.getInterestRate() + "%");
System.out.println("HDFC Interest Rate: " + b2.getInterestRate() + "%");
b1.display();
}
}
```

**Output:**



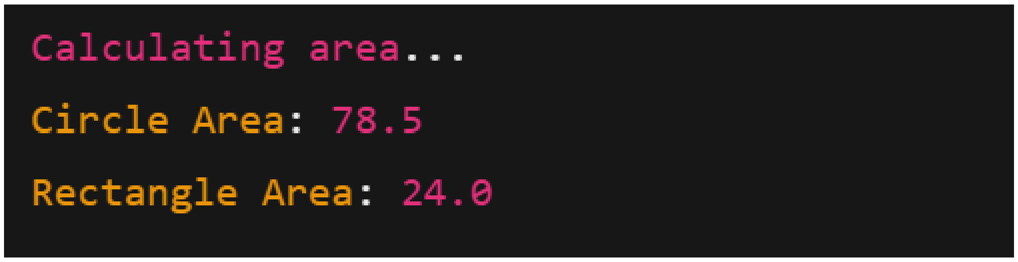
```
SBI Interest Rate: 5.5%

HDFC Interest Rate: 6.8%

Banking system running...
```

## 13 c) Payment

**Code:**

```
abstract class Shape {
abstract double calculateArea();
void display() {
System.out.println("Calculating area...");
}
}
class Circle extends Shape {
double radius;
Circle(double radius) {
this.radius = radius;
}
double calculateArea() {
return 3.14 * radius * radius;
}
}
class Rectangle extends Shape {
double length, width;
Rectangle(double length, double width) {
this.length = length;
this.width = width;
}
double calculateArea() {
return length * width;
}
}
public class Main {
public static void main(String[] args) {
Shape s1 = new Circle(5);
Shape s2 = new Rectangle(4, 6);
s1.display();
System.out.println("Circle Area: " + s1.calculateArea());
System.out.println("Rectangle Area: " + s2.calculateArea());
}
}
```

## Output:



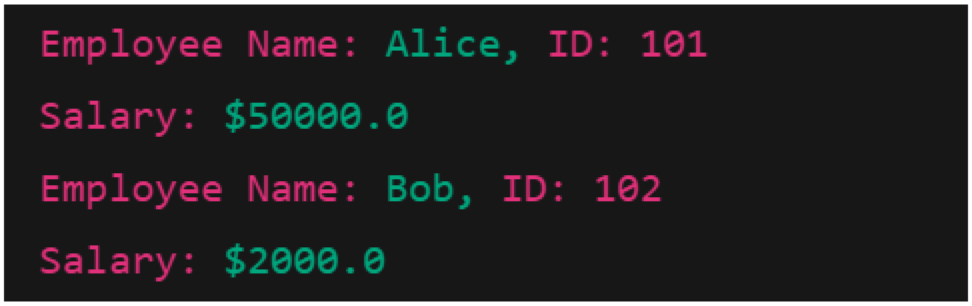
```
Calculating area...
Circle Area: 78.5
Rectangle Area: 24.0
```

## 13 d) employee salary

### Code:

```
abstract class Employee {
String name;
int id;
Employee(String name, int id) {
this.name = name;
this.id = id;
}
abstract double calculateSalary();
void display() {
System.out.println("Employee Name: " + name + ", ID: " + id);
}
}
class FullTimeEmployee extends Employee {
double salary;
FullTimeEmployee(String name, int id, double salary) {
super(name, id);
this.salary = salary;
}
double calculateSalary() {
return salary;
}
}
class PartTimeEmployee extends Employee {
double hourlyRate;
int hoursWorked;
PartTimeEmployee(String name, int id, double hourlyRate, int hoursWorked) {
super(name, id);
this.hourlyRate = hourlyRate;
this.hoursWorked = hoursWorked;
}
double calculateSalary() {
return hourlyRate * hoursWorked;
}
}
public class Main {
public static void main(String[] args) {
Employee e1 = new FullTimeEmployee("Alice", 101, 50000);
Employee e2 = new PartTimeEmployee("Bob", 102, 20, 100);
e1.display();
System.out.println("Salary: $" + e1.calculateSalary());
e2.display();
System.out.println("Salary: $" + e2.calculateSalary());
}
}
```

**Output:**



```
Employee Name: Alice, ID: 101
Salary: $50000.0
Employee Name: Bob, ID: 102
Salary: $2000.0
```

# 14)ENCAPSULATION

## ENCAPSULATION PROGRAMS

## 14a) student

## Code:

```
class Student {
   private String name;  // Private variable (Encapsulation)

   public void setName(String name) {
      this.name = name;
   }

   public String getName() {
      return name;
   }
}

public class Encapsulation {
   public static void main(String[] args) {
      Student s = new Student();

      s.setName("Alice");
      System.out.println("Student Name: " + s.getName());
   }
}
```

## Output:

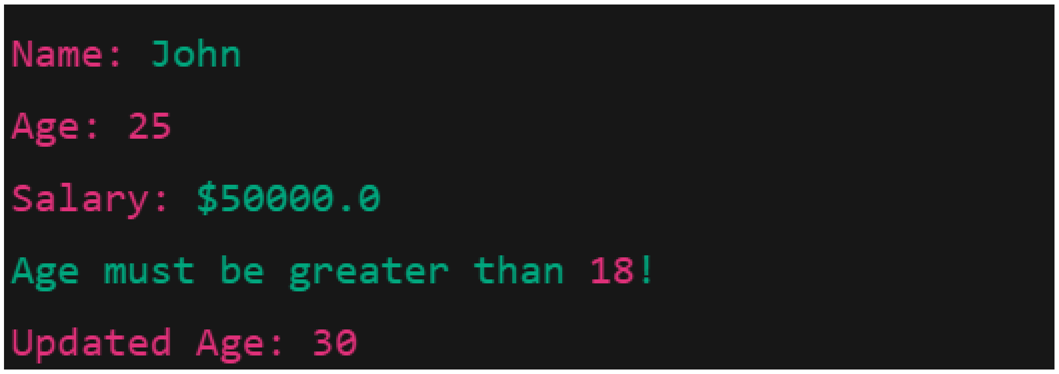
```
Student Name: Alice
```

## 14b) EMPLOYEE

## Code:

```
class Employee {
    private String name;
    private int age;
    private double salary;


    public Employee(String name, int age, double salary) {
        this.name = name;
        this.age = age;
        this.salary = salary;
    }


    public String getName() {
        return name;
    }


    public void setName(String name) {
        this.name = name;
    }


    public int getAge() {
        return age;
    }


    public void setAge(int age) {
        if (age > 18) { // Ensuring valid age
            this.age = age;
        } else {
            System.out.println("Age must be greater than 18!");
        }
    }


    public double getSalary() {
        return salary;
    }
```

**Output:**



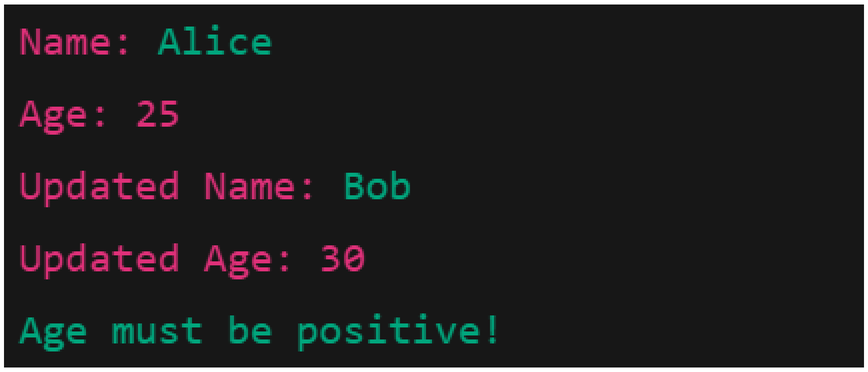
```
Name: John
Age: 25
Salary: $50000.0
Age must be greater than 18!
Updated Age: 30
```

## 14c )PERSON

## Code:

```
class Person {
     private String name;
    private int age;
    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        if (age > 0) {
            this.age = age;
        } else {
            System.out.println("Age must be positive!");
        }
    }
}
public class EncapsulationExample {
    public static void main(String[] args) {
        Person p = new Person("Alice", 25);

        System.out.println("Name: " + p.getName());
        System.out.println("Age: " + p.getAge());

        p.setName("Bob");
        p.setAge(30);

        System.out.println("Updated Name: " + p.getName());
        System.out.println("Updated Age: " + p.getAge());

        p.setAge(-5);
    }
}
```

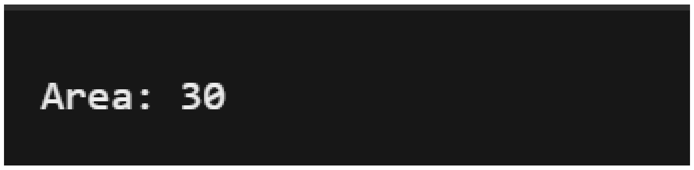
## Output:


```
Name: Alice
Age: 25
Updated Name: Bob
Updated Age: 30
Age must be positive!
```

## 14d) AREA
## Code:

```
class Area {
  int length;
  int breadth;

  Area(int length, int breadth) {
    this.length = length;
    this.breadth = breadth;
  }

  public void getArea() {
    int area = length * breadth;
    System.out.println("Area: " + area);
  }
}

class Main {
  public static void main(String[] args) {

    Area rectangle = new Area(5, 6);
    rectangle.getArea();
  }
}
```

## Output:


```
Area: 30
```

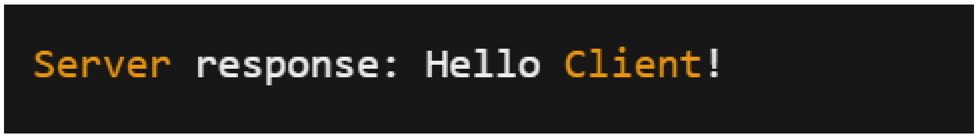
## 15)PACKAGES PROGRAMS

### 15.a) USER DEFINED

### Code:

```
import java.io.*;
import java.net.*;

public class ClientSocketDemo {
   public static void main(String[] args) {
      String hostname = "localhost";
      int port = 12345;

      try (
         Socket socket = new Socket(hostname, port);
         PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
         BufferedReader in = new BufferedReader(new InputStreamReader(socket.getInputStream()))
      ) {
         out.println("Hello Server!");
         String response = in.readLine();
         System.out.println("Server response: " + response);
      } catch (IOException e) {
         e.printStackTrace();
      }
   }
}
```

### Output:



```
Server response: Hello Client!
```

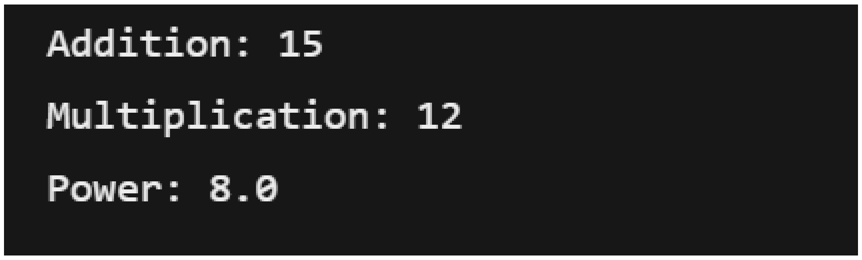
### 15.b) User Defined Packages

### Code:

```
package mathoperations;
public class Calculator {
public int add(int a, int b) {
return a + b;
}
public int multiply(int a, int b) {
```

```
return a * b;
}
}
package mathoperations;
public class AdvancedCalculator {
public double power(double base, double exponent) {
return Math.pow(base, exponent);
}
}
import mathoperations.Calculator;
import mathoperations.AdvancedCalculator;
public class Main {
public static void main(String[] args) {
Calculator calc = new Calculator();
AdvancedCalculator advCalc = new AdvancedCalculator();
System.out.println("Addition: " + calc.add(5, 10));
System.out.println("Multiplication: " + calc.multiply(3, 4));
System.out.println("Power: " + advCalc.power(2, 3));
}
}
```

## Output:



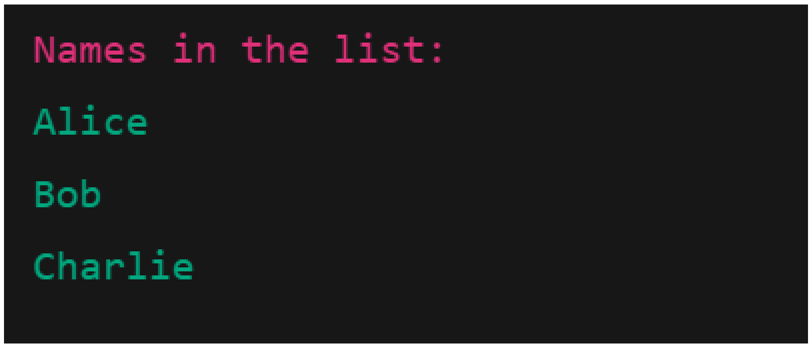
```
Addition: 15

Multiplication: 12

Power: 8.0
```

## 15c) Built – in Package(3 Packages)

## Code:

```
import java.util.ArrayList;

public class NameListDemo {
   public static void main(String[] args) {
      ArrayList<String> names = new ArrayList<>();
      names.add("Alice");
      names.add("Bob");
      names.add("Charlie");

      System.out.println("Names in the list:");
      for (String name : names) {
         System.out.println(name);
      }
   }
}
```

**Output:**



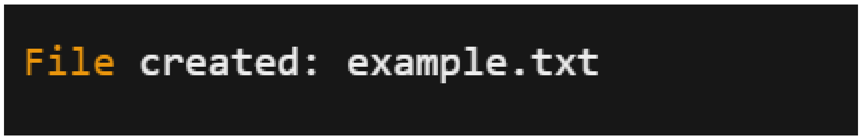
```
Names in the list:
Alice
Bob
Charlie
```

## 15d) Built – in Package(3 Packages)

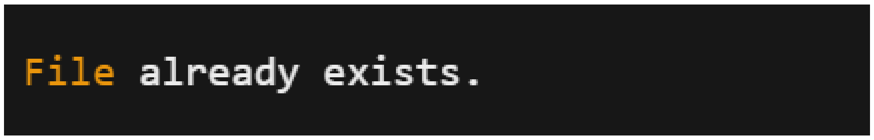
**Code:**

```
import java.io.File;
import java.io.IOException;
public class Main {
public static void main(String[] args) {
File file = new File("example.txt");
try {
if (file.createNewFile()) {
System.out.println("File created: " + file.getName());
} else {
System.out.println("File already exists.");
}
} catch (IOException e) {
System.out.println("An error occurred.");
}
}
}
```

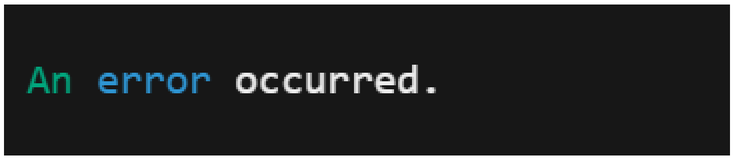
**Output:**



```
File created: example.txt
```



```
File already exists.
```



```
An error occurred.
```

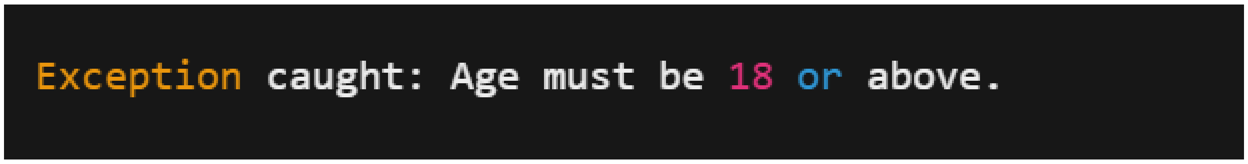
## 16)EXCEPTION HANDLING PROGRAMS

### 16a) FileExceptionExample

### Code:

```
class AgeException extends Exception {
   public AgeException(String message) {
      super(message);
   }
}

public class ExceptionExample4 {
   public static void validateAge(int age) throws AgeException {
      if (age < 18) {
         throw new AgeException("Age must be 18 or above.");
      } else {
         System.out.println("Valid age: " + age);
      }
   }

   public static void main(String[] args) {
      try {
         validateAge(15);
      } catch (AgeException e) {
         System.out.println("Exception caught: " + e.getMessage());
      }
   }
}
```

### Output:



```
Exception caught: Age must be 18 or above.
```

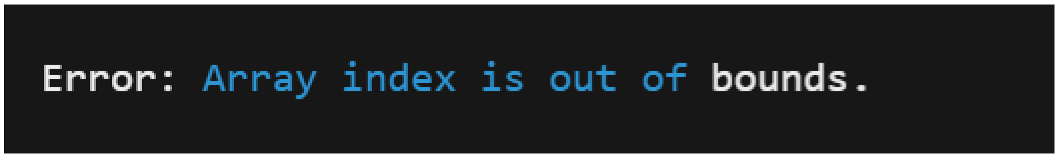
### 16b) Division Example

### Code:

```
public class ExceptionExample2 {
public static void main(String[] args) {
try {
int[] arr = {1, 2, 3};
System.out.println(arr[5]);
} catch (ArrayIndexOutOfBoundsException e) {
System.out.println("Error: Array index is out of bounds.");
}
}
```

}

## Output:



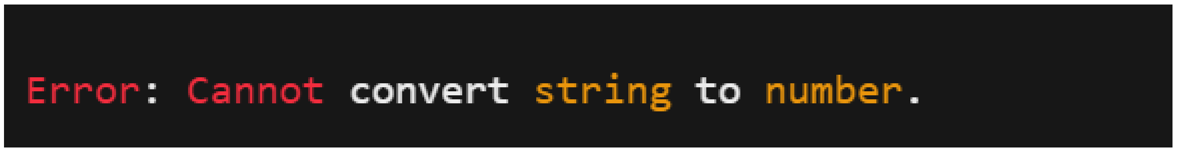
```
Error: Array index is out of bounds.
```

## 16c) StringIndexOutOfBoundsExample

## Code:

```
public class ExceptionExample3 {
public static void main(String[] args) {
try {
int num = Integer.parseInt("abc");
} catch (NumberFormatException e) {
System.out.println("Error: Cannot convert string to number.");
} catch (Exception e) {
System.out.println("General Exception Caught.");
}
}
}
```

## Output:



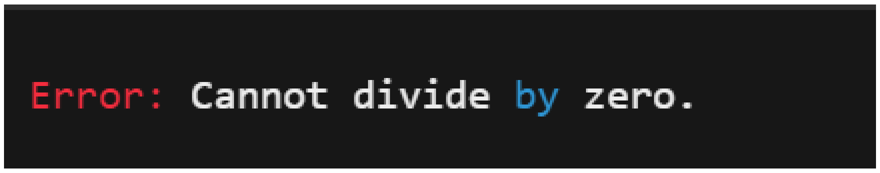
```
Error: Cannot convert string to number.
```

## 16d)NumberFormatExample

**Code:**

```
public class ExceptionExample1 {
    public static void main(String[] args) {
        try {
            int num1 = 10, num2 = 0;
            int result = num1 / num2; // This will throw an exception
            System.out.println("Result: " + result);
        } catch (ArithmeticException e) {
            System.out.println("Error: Cannot divide by zero.");
        }
    }
}
```

**Output:**



```
Error: Cannot divide by zero.
```

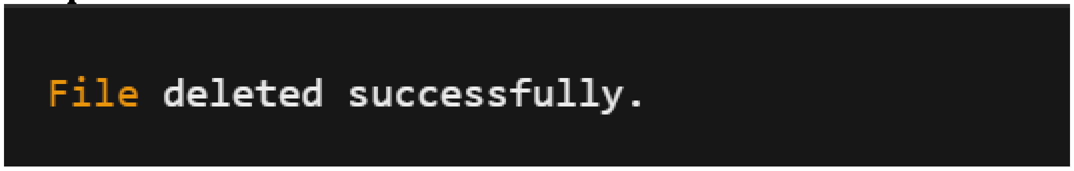
## 17)FILE HANDLING PROGRAMS

### 17a) DELETING FILE

**Code:**

```
import java.io.File;

public class FileHandlingExample4 {
    public static void main(String[] args) {
        File file = new File("sample.txt");
        if (file.delete()) {
            System.out.println("File deleted successfully.");
        } else {
            System.out.println("Failed to delete the file.");
        }
    }
}
```

**Output:**



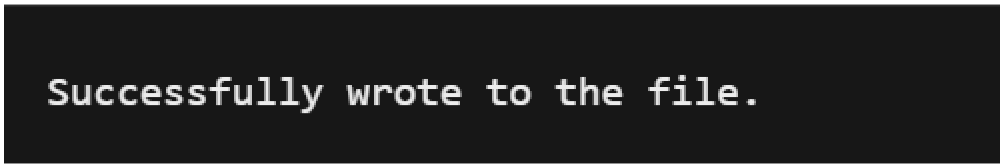
```
File deleted successfully.
```

## 17b) WRITING FILE

**Code:**

```
import java.io.FileWriter;
import java.io.IOException;
public class WriteFileExample {
public static void main(String[] args) {
try {
FileWriter writer = new FileWriter("example.txt");
writer.write("Hello, this is a test file.");
writer.close();
System.out.println("Successfully wrote to the file.");
} catch (IOException e) {
System.out.println("An error occurred.");
}
}
}
```

**Output:**



```
Successfully wrote to the file.
```

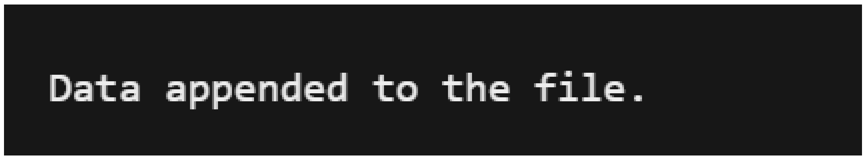
## 17c) APPEND FILE

**Code:**

```
import java.io.FileWriter;
import java.io.IOException;
```

```
public class FileHandlingExample3 {
   public static void main(String[] args) {
      try {
         FileWriter writer = new FileWriter("sample.txt", true);
         writer.append("\nAppending new text.");
         writer.close();
         System.out.println("Data appended to the file.");
      } catch (IOException e) {
         System.out.println("Error appending to the file.");
      }
   }
}
```

## Output:



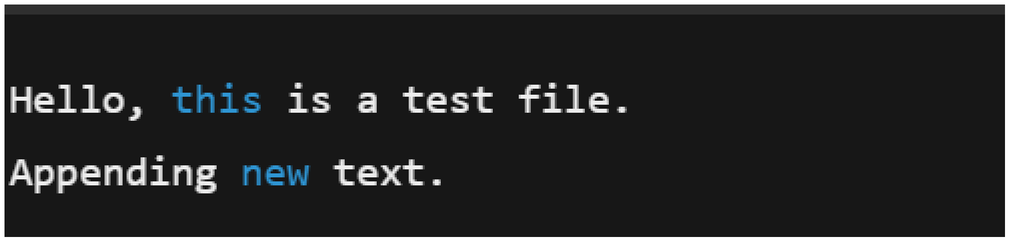
```
Data appended to the file.
```

## 17d) READING FILE

## Code:

```
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;

public class FileHandlingExample2 {
   public static void main(String[] args) {
      try {
         File file = new File("sample.txt");
         Scanner reader = new Scanner(file);
         while (reader.hasNextLine()) {
            String data = reader.nextLine();
            System.out.println("File Content: " + data);
         }
         reader.close();
      } catch (FileNotFoundException e) {
            System.out.println("File not found.");
        }
    }
}
```

**Output:**



```
Hello, this is a test file.
Appending new text.
```