



**SCHOOL OF
COMPUTING**

LAB RECORD

23CSE111- Object Oriented Programming

Submitted by

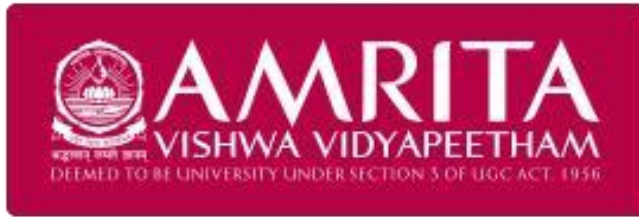
CH.SC.U4CSE24119 - Kavın.JS

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND
ENGINEERING

AMRITA VISHWA VIDYAPEETHAM
AMRITA SCHOOL OF COMPUTING

CHENNAI

March - 2025



**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.U4CSE24119 – Kavin.J.S** 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

S.NO	TITLE	PAGE.NO
	UML DIAGRAM	
1.	ONLINE SHOPPING MANAGEMENT SYSTEM	
	1.a)Use Case Diagram	5
	1.b)State Diagram	5
	1.c) Class Diagram	6
	1.d)Sequence Diagram	7
	1.e)Communication Diagram	8
2.	COURSE MANAGEMENT SYSTEM	
	2.a) Use Case Diagram	9
	2.b) State Diagram	9
	2.c) Class Diagram	10
	2.d)Sequence Diagram	11
	2.e)Communication Diagram	13
3.	BASIC JAVA PROGRAMS	
	3.a)Average Calculator	13
	3.b)Capitalize	14
	3.c)EvenOdd	15
	3.d)Factorial	16
	3.e)Fibonacci	17
	3.f)Multiplication Table	18
	3.g)Palindrome	19
	3.h)PrimeORNot	20
	3.i)Simple Calc	21
	3.j)Vowel Consonant Classifier	23
	INHERITANCE	
4.	SINGLE INHERITANCE PROGRAMS	
	4.a)Animal Barks Program	24
	4.b)Vehicle Constructor Program	25
5.	MULTILEVEL INHERITANCE PROGRAMS	

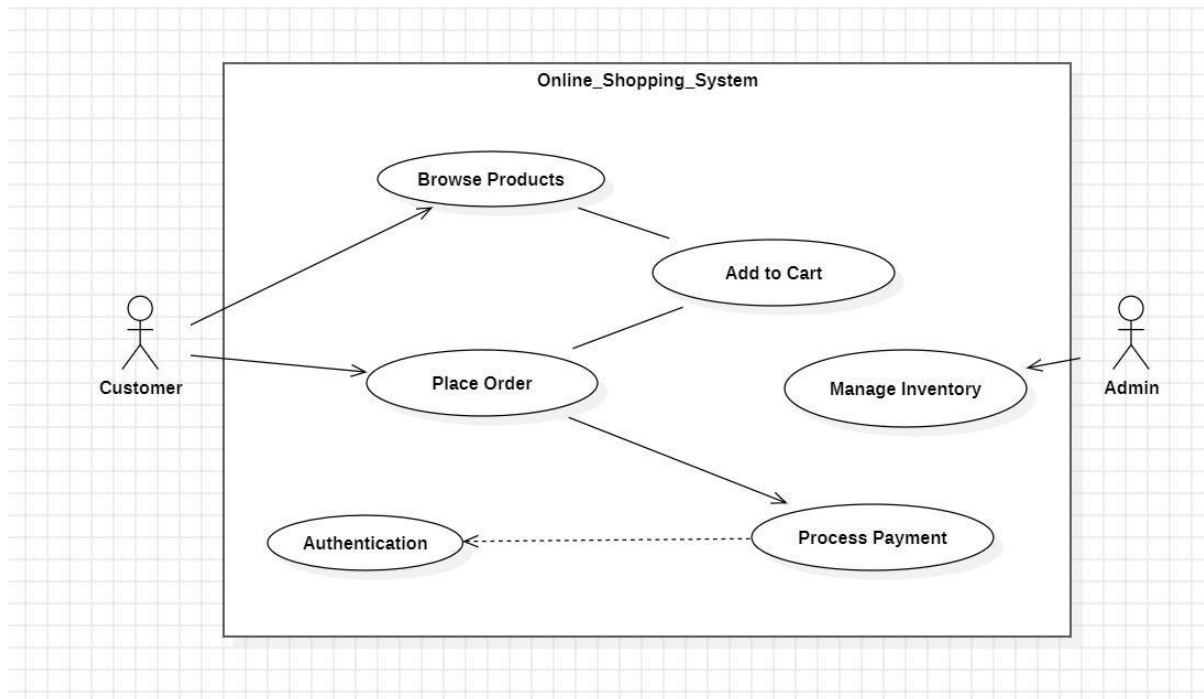
	5.a)Bank Program	26
	5.b)E-Commerce Program	28
6.	HIERARCHICAL INHERITANCE PROGRAMS	
	6.a)University System	29
	6.b)Vehicle System	31
7.	HYBRID INHERITANCE PROGRAMS	
	7.a) Student Performance System	32
	7.b) Smart Home System	33
	POLYMORPHISM	
8.	CONSTRUCTOR PROGRAMS	
	8.a) Student Info	35
9.	CONSTRUCTOR OVERLOADING PROGRAMS	
	9.a)Area of Rectangle CalC	36
10.	METHOD OVERLOADING PROGRAMS	
	10.a) Simple Addition	37
	10.b) Info Display	38
11.	METHOD OVERRIDING PROGRAMS	
	11.a)E-Ticket Booker	39
	11.b)Bank Transaction System	40
	ABSTRACTION	
12.	INTERFACE PROGRAMS	
	12.a))E-Commerce Payment Interface	42
	12.b))Smart Device Control Interface	43
	12.c)Document Export Interface	44
	12.d)Vehicle Interface	45
13.	ABSTRACT CLASS PROGRAMS	
	13.a)Online Payment System	47
	13.b)College Admission System:	48
	13.c)Employee Payroll System	49
	13.d)Shape Area Calculator	50
	ENCAPSULATION	
14.	ENCAPSULATION PROGRAMS	
	14.a)Bank Account System	52
	14.b)Student Report System	53
	14.c)Login System	54
	14.d)Temperature Conversion	55
15.	PACKAGES PROGRAMS	
	15.a)User Defined Packages	56
	15.b)User Defined Packages	56
	15.c)Built – in Package(3 Packages)	57
	15.d)Built – in Package(3 Packages)	58
16.	EXCEPTION HANDLING PROGRAMS	

	16.a)Java program that throws an exception and catch it using a try-catch block	60
	16.b) Java program to create a method that takes an integer as a parameter and throws an exception if the number is odd	61
	16.c)Java program to create a method that takes a string as input and throws an exception if the string does not contain vowel	62
	16.d)Java program that reads a list of integers from the user and throws an exception if any numbers are duplicates	63
17.	FILE HANDLING PROGRAMS	
	17.a)Java - Print File Content, Display File	65
	17.b)Delete file using java program	65
	17.c)Read file line by line using java:	66
	17.d)Java program to append text/string in a file	68

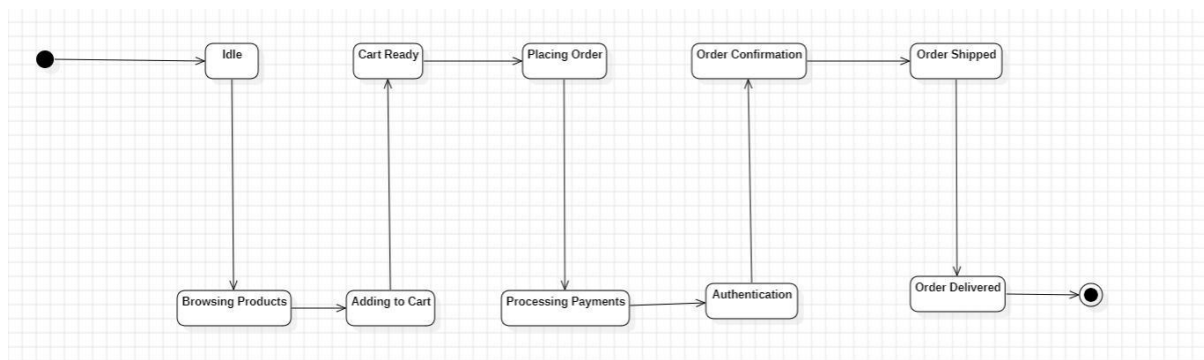
UML DIAGRAMS

1) Online Shopping System:

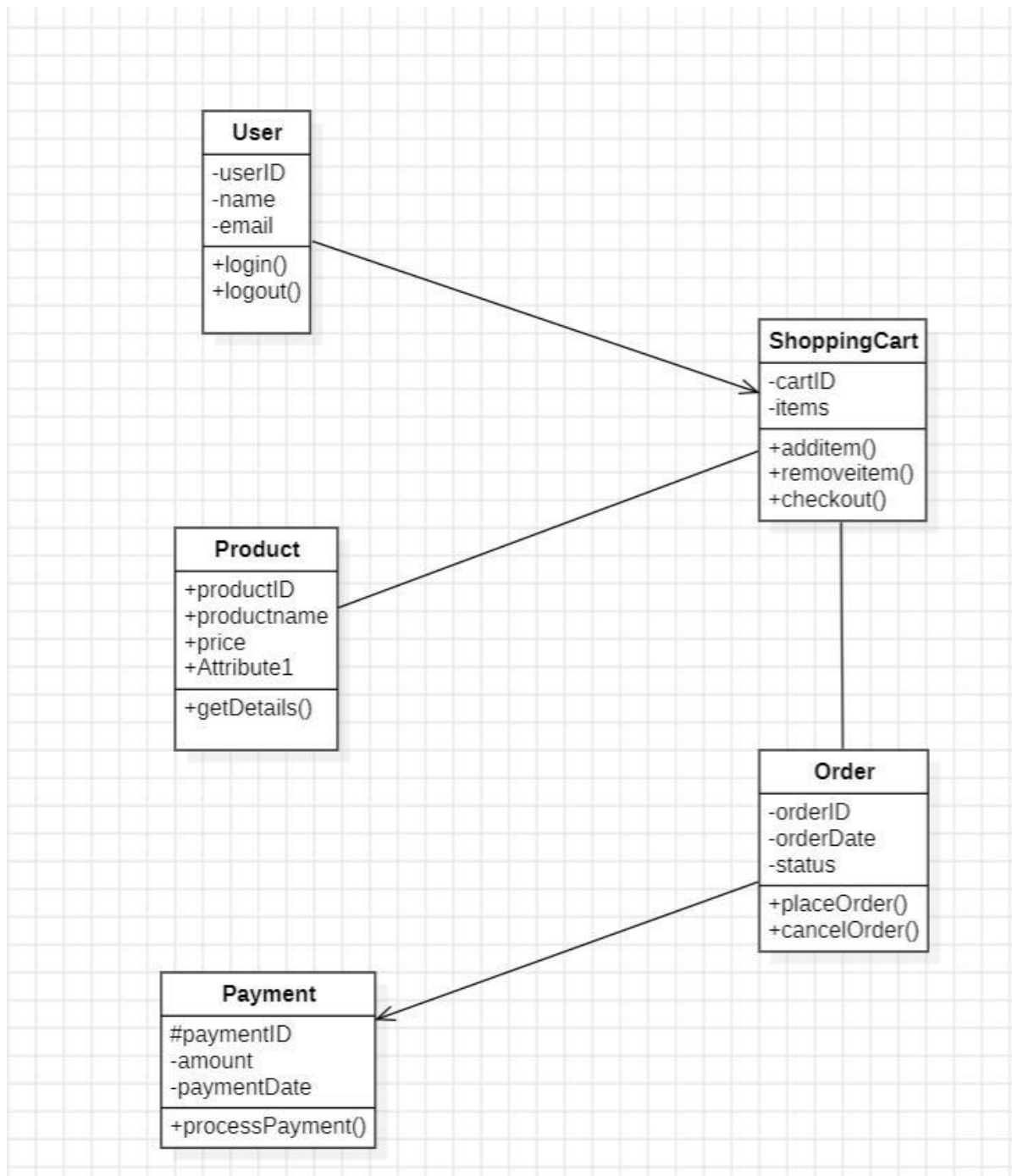
a) Use Case Diagram



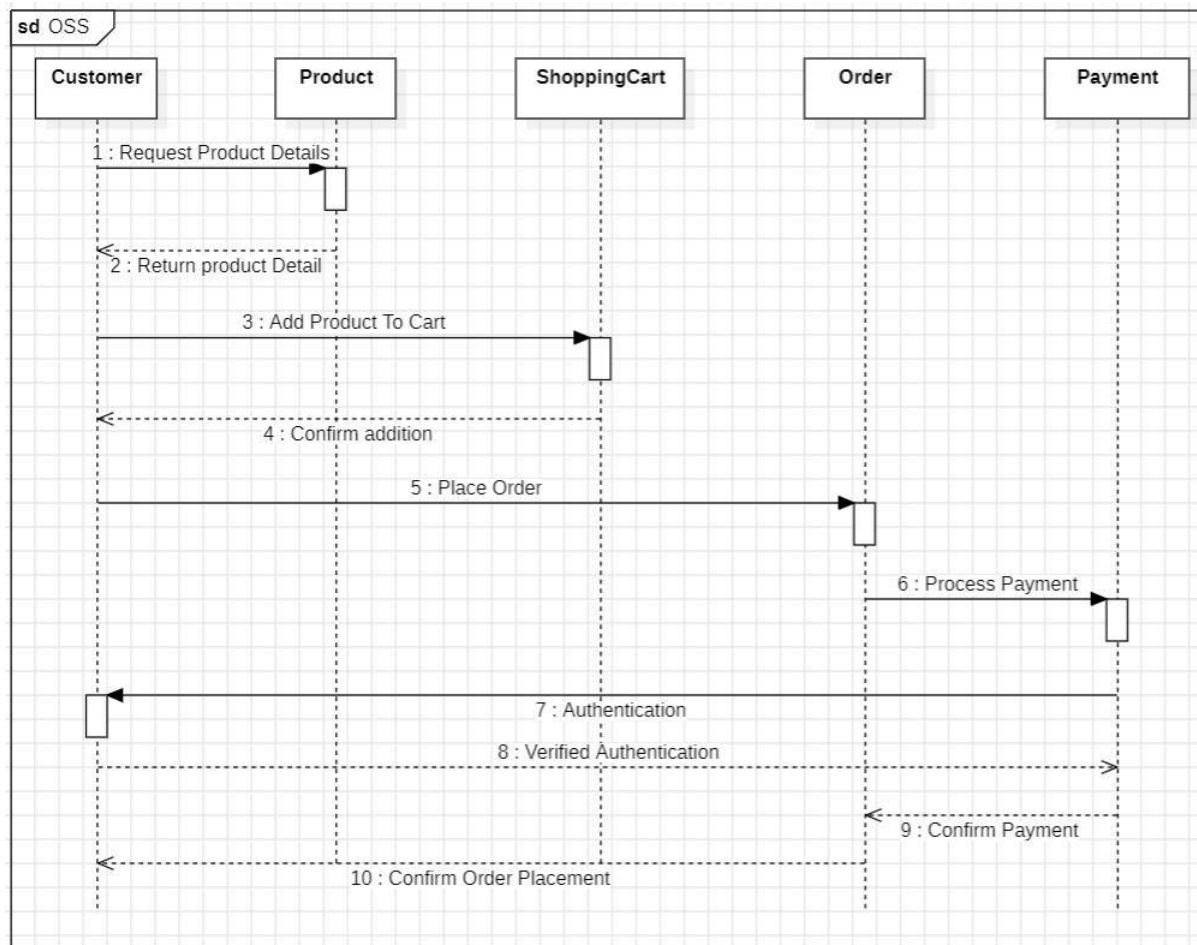
b) State Diagram



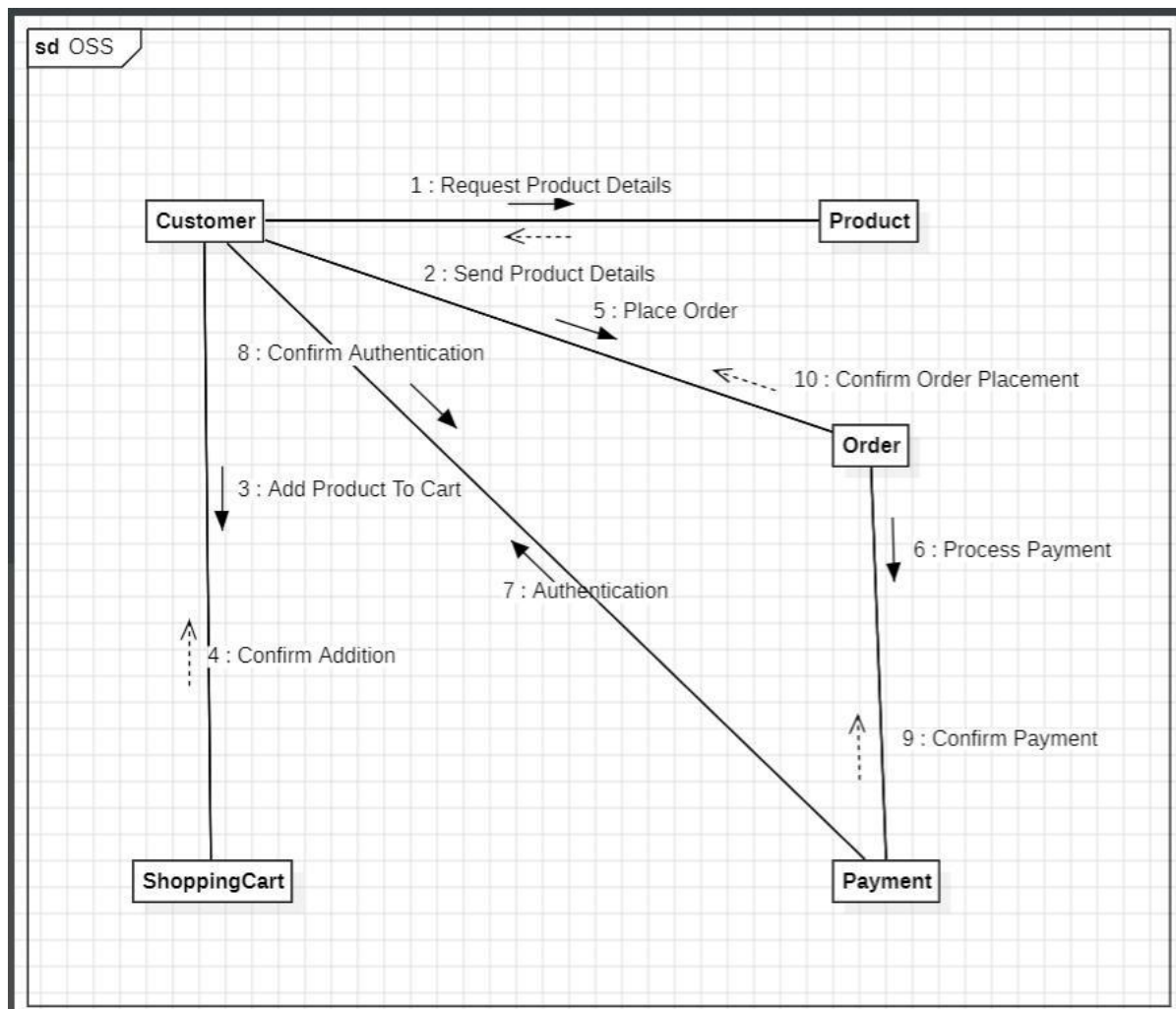
c) Class Diagram



d) Sequence Diagram

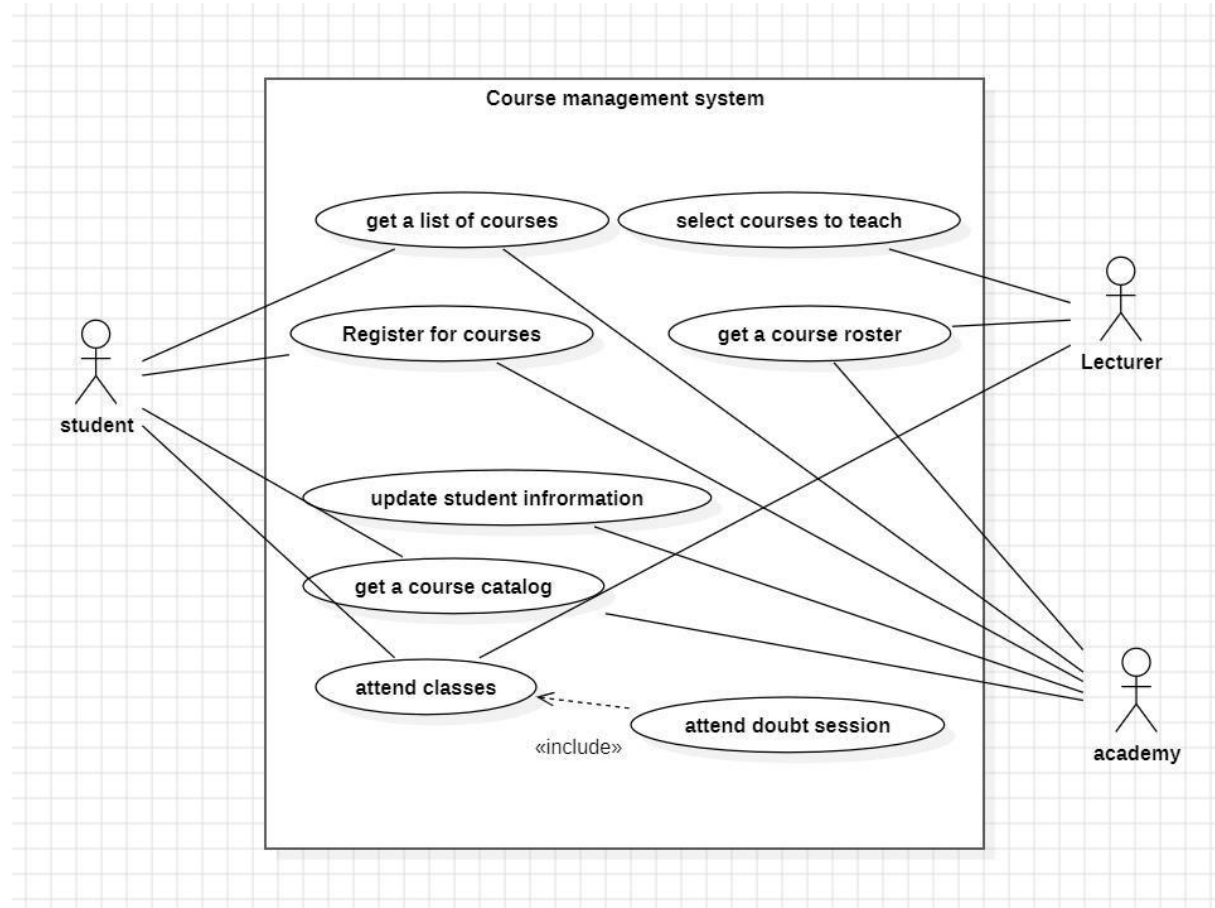


e) Communication Diagram

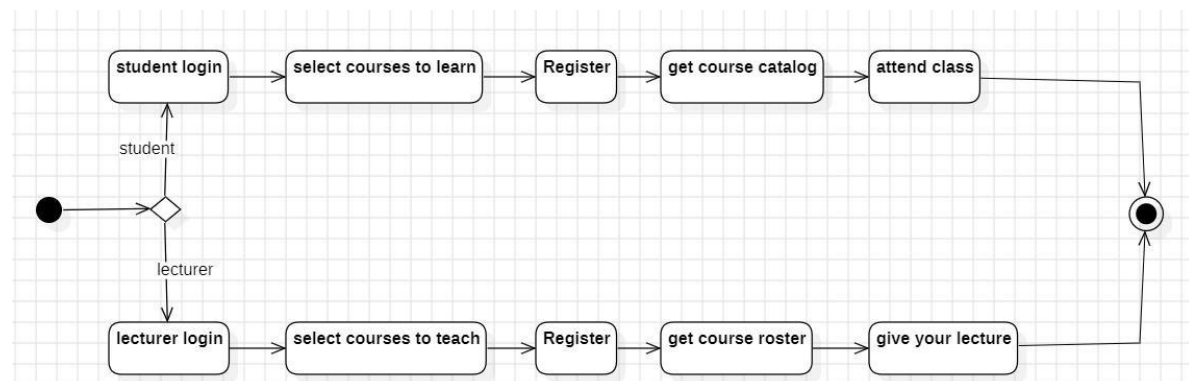


2) Course Management System:

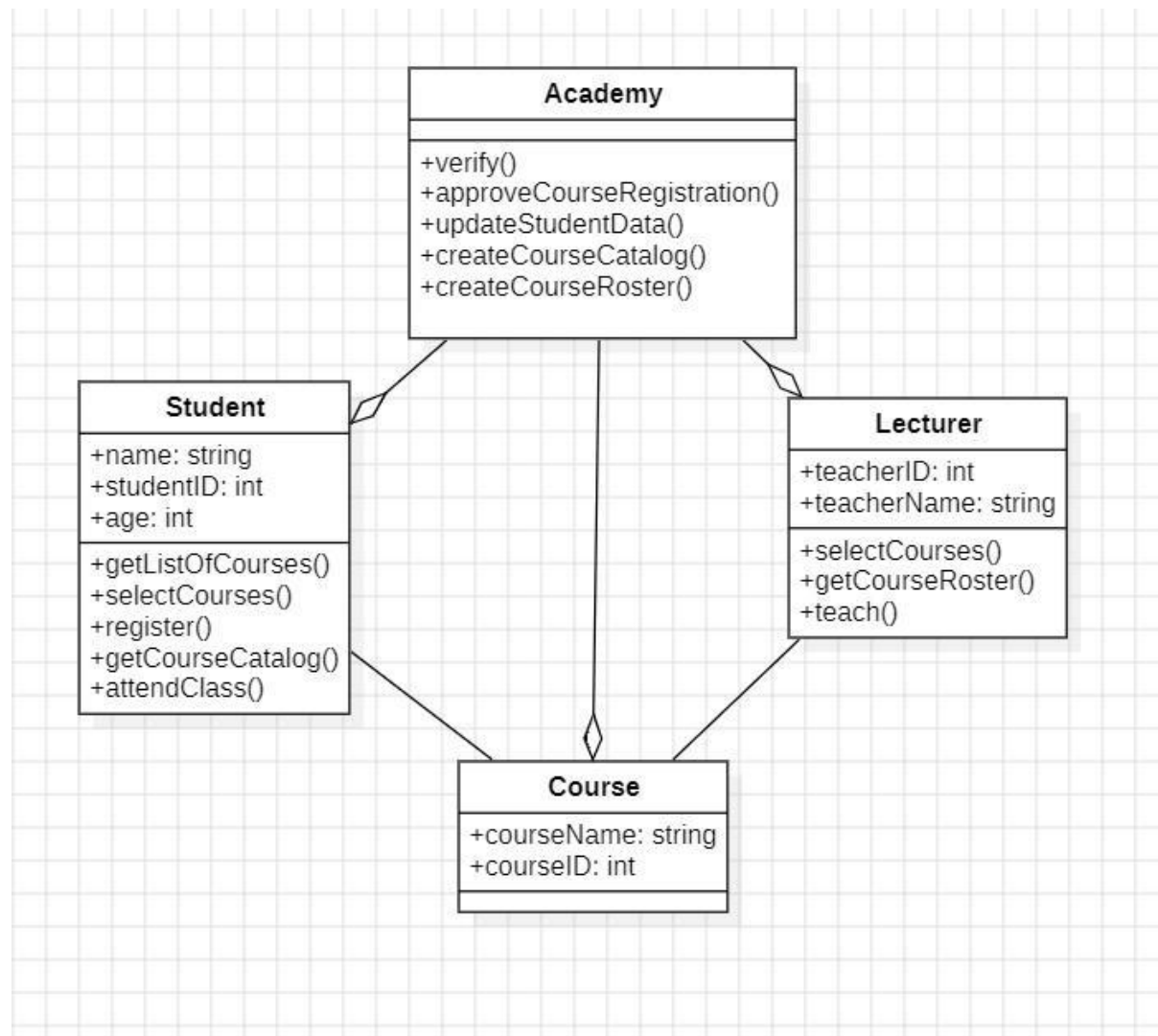
a) Use Case Diagram



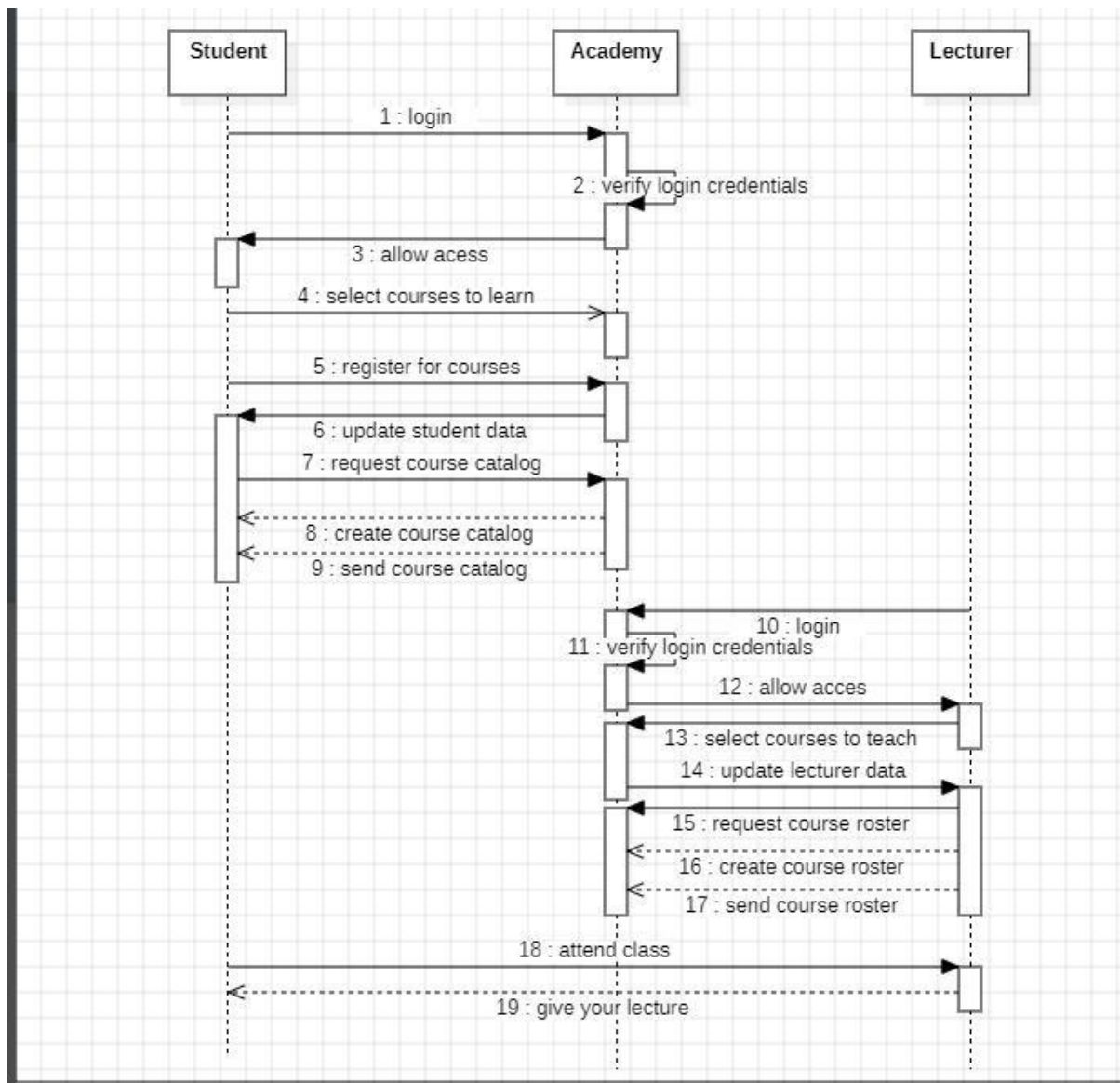
b) State Diagram



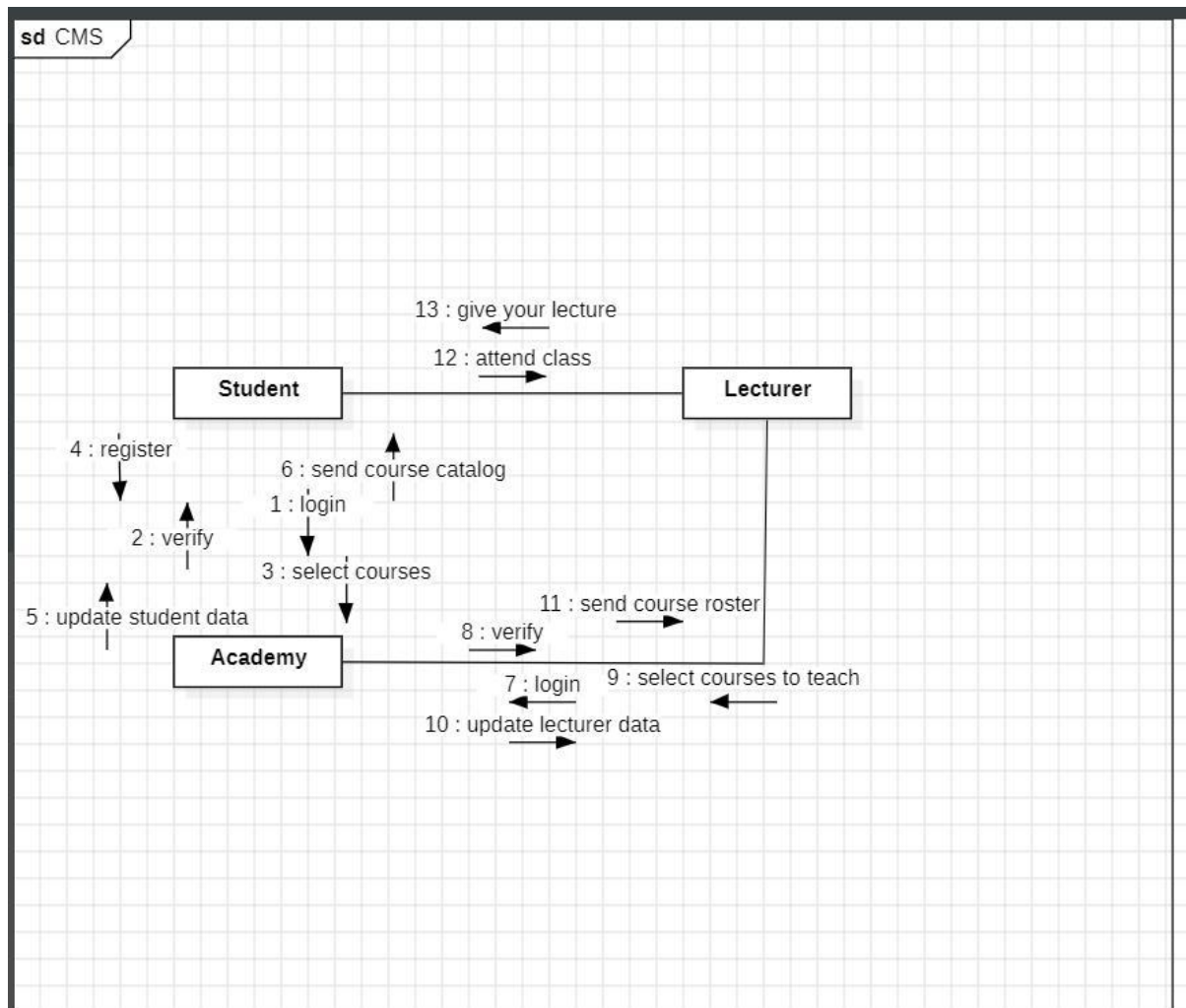
c) Class Diagram



d) Sequence Diagram



e) Communication Diagram



BASIC JAVA PROGRAMS

1) Average Calculator

CODE:

```
J Average.java
1  import java.util.Scanner;
2
3  public class Average {
4      public static void main(String[] args) {
5          Scanner scanner = new Scanner(System.in);
6
7          System.out.print("Enter the number of elements: ");
8          int n = scanner.nextInt();
9
10         double[] numArray = new double[n];
11         double sum = 0.0;
12
13         System.out.println("Enter the numbers:");
14         for (int i = 0; i < n; i++) {
15             numArray[i] = scanner.nextDouble();
16             sum += numArray[i];
17         }
18
19         double average = sum / n;
20         System.out.printf("The average is: %.2f\n", average);
21
22         scanner.close();
23     }
24 }
```

OUTPUT:

```
Microsoft Windows [Version 10.0.26120.3380]
(c) Microsoft Corporation. All rights reserved.

E:\Java\Java Programs>javac Average.java

E:\Java\Java Programs>java Average.java
Enter the number of elements: 4
Enter the numbers:
44
33
22
11
The average is: 27.50

E:\Java\Java Programs>
```

2) Capitalize

CODE:

```
J Capitalize.java > ...
1  import java.util.Scanner;
2
3  public class Capitalize {
4      public static void main(String[] args) {
5          Scanner scanner = new Scanner(System.in);
6
7          System.out.print("Enter a string: ");
8          String name = scanner.nextLine();
9
10         if (!name.isEmpty()) {
11             String firstLetter = name.substring(0, 1).toUpperCase();
12             String remainingLetters = name.substring(1);
13             name = firstLetter + remainingLetters;
14         }
15
16         System.out.println("Capitalized String: " + name);
17         scanner.close();
18     }
19 }
20
```

OUTPUT:

```
E:\Java\Java Programs>javac Capitalize.java

E:\Java\Java Programs>java Capitalize.java
Enter a string: abcdefg
Capitalized String: Abcdefg
```

3) EvenOdd

CODE:

```
J EvenOdd.java > ...
1  import java.util.Scanner;

a\Java Programs\Average.class
3  public class EvenOdd {
4
5      public static void main(String[] args) {
6
7          Scanner reader = new Scanner(System.in);
8
9          System.out.print("Enter a number: ");
10         int num = reader.nextInt();
11
12         if(num % 2 == 0)
13             System.out.println(num + " is even");
14         else
15             System.out.println(num + " is odd");
16     }
17 }
```

OUTPUT:

```
E:\Java\Java Programs>javac EvenOdd.java

E:\Java\Java Programs>java EvenOdd.java
Enter a number: 7
7 is odd

E:\Java\Java Programs>javac EvenOdd.java

E:\Java\Java Programs>java EvenOdd.java
Enter a number: 4
4 is even
```


4) Factorial

CODE:

```
J Factorial.java > ...
1  import java.util.Scanner;
2
3  public class Factorial {
4      public static void main(String[] args) {
5          Scanner scanner = new Scanner(System.in);
6
7          System.out.print("Enter a positive integer: ");
8          int num = scanner.nextInt();
9
10         if (num < 0) {
11             System.out.println("Factorial is not defined for negative numbers.");
12         } else {
13             long factorial = 1;
14
15             for (int i = 1; i <= num; ++i) {
16                 factorial *= i;
17             }
18
19             System.out.printf("Factorial of %d = %d\n", num, factorial);
20         }
21
22         scanner.close();
23     }
24 }
```

OUTPUT:

```
E:\Java\Java Programs>javac Factorial.java

E:\Java\Java Programs>java Factorial.java
Enter a positive integer: 8
Factorial of 8 = 40320
```

5) Fibonacci

CODE:

```
J Fibonacci.java 1 X
J Fibonacci.java > ...
1  import java.util.Scanner;
2
3  public class Fibonacci {
4      Run main | Debug main
      public static void main(String[] args) {
5          Scanner scanner = new Scanner(System.in);
6
7          // Get input from the user
8          System.out.print("Enter the number of terms: ");
9          int n = scanner.nextInt();
10
11         int firstTerm = 0, secondTerm = 1;
12
13         System.out.println("Fibonacci Series till " + n + " terms:");
14
15         for (int i = 1; i <= n; i++) {
16             System.out.print(firstTerm + (i < n ? ", " : "\n")); // Print comma except for the last term
17
18             int nextTerm = firstTerm + secondTerm;
19             firstTerm = secondTerm;
20             secondTerm = nextTerm;
21         }
22
23         scanner.close();
24     }
25 }
```

OUTPUT:

```
E:\Java\Java Programs>javac Fibonacci.java

E:\Java\Java Programs>java Fibonacci.java
Enter the number of terms: 8
Fibonacci Series till 8 terms:
0, 1, 1, 2, 3, 5, 8, 13
```

6) Multiplication Table

CODE:

```
MultiplicationTable.java 1 X
MultiplicationTable.java > ...
1  import java.util.Scanner;
2
3  public class MultiplicationTable {
    Run main | Debug main
4      public static void main(String[] args) {
5          Scanner scanner = new Scanner(System.in);
6
7          System.out.print("Enter a number: ");
8          int num = scanner.nextInt();
9
10         int i = 1;
11         while (i <= 10) {
12             System.out.printf("%d * %d = %d\n", num, i, num * i);
13             i++;
14         }
15
16         scanner.close();
17     }
18 }
```

OUTPUT:

```
E:\Java\Java Programs>javac MultiplicationTable.java

E:\Java\Java Programs>java MultiplicationTable.java
Enter a number: 7
7 * 1 = 7
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28
7 * 5 = 35
7 * 6 = 42
7 * 7 = 49
7 * 8 = 56
7 * 9 = 63
7 * 10 = 70
```

7) Palindrome

Code:

```
J Palindrome.java X
J Palindrome.java > ...
1  import java.util.Scanner;
2
3  public class Palindrome {
4      public static void main(String[] args) {
5          Scanner scanner = new Scanner(System.in);
6
7          System.out.print("Enter a string: ");
8          String str = scanner.nextLine();
9
10         String reverseStr = "";
11         int strLength = str.length();
12
13         for (int i = strLength - 1; i >= 0; --i) {
14             reverseStr += str.charAt(i);
15         }
16
17         if (str.equalsIgnoreCase(reverseStr)) {
18             System.out.println(str + " is a Palindrome String.");
19         } else {
20             System.out.println(str + " is not a Palindrome String.");
21         }
22
23         scanner.close();
24     }
25 }
26
```

Output:

```
E:\Java\Java Programs>javac Palindrome.java

E:\Java\Java Programs>java Palindrome.java
Enter a string: racecar
racecar is a Palindrome String.
```

8) Prime Or Not

Code:

```
imeORNot.java X
rimeORNot.java > ...
import java.util.Scanner;

public class PrimeORNot {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a number: ");
        int num = scanner.nextInt();
        boolean flag = false;

        if (num < 2) {
            flag = true;
        } else {
            for (int i = 2; i <= num / 2; ++i) {
                if (num % i == 0) {
                    flag = true;
                    break;
                }
            }
        }

        if (!flag)
            System.out.println(num + " is a prime number.");
        else
            System.out.println(num + " is not a prime number.");

        scanner.close();
    }
}
```

Output:

```
E:\Java\Java Programs>javac PrimeORNot.java

E:\Java\Java Programs>java PrimeORNot.java
Enter a number: 2362
2362 is not a prime number.

E:\Java\Java Programs>javac PrimeORNot.java

E:\Java\Java Programs>java PrimeORNot.java
Enter a number: 2
2 is a prime number.
```

9) Simple Calculator

Code:

```
J SimpleCalC.java 2 X
J SimpleCalC.java > ...
1  import java.util.Scanner;
2
3  class SimpleCalC {
    Run main | Debug main
4  public static void main(String[] args) {
5      Scanner input = new Scanner(System.in);
6
7      System.out.println("Choose an operator: +, -, *, or /");
8      char operator = input.next().charAt(0);
9
10     System.out.println("Enter first number:");
11     double number1 = input.nextDouble();
12
13     System.out.println("Enter second number:");
14     double number2 = input.nextDouble();
15
16     double result;
17
18     if (operator == '+') {
19         result = number1 + number2;
20         System.out.println(number1 + " + " + number2 + " = " + result);
21     } else if (operator == '-') {
22         result = number1 - number2;
23         System.out.println(number1 + " - " + number2 + " = " + result);
24     } else if (operator == '*') {
25         result = number1 * number2;
26         System.out.println(number1 + " * " + number2 + " = " + result);
27     } else if (operator == '/') {
28         if (number2 != 0) {
29             result = number1 / number2;
30             System.out.println(number1 + " / " + number2 + " = " + result);
31         } else {
32             System.out.println("Error! Division by zero is not allowed.");
33         }
34     } else {
35         System.out.println("Invalid operator!");
36     }
37
38     input.close();
39 }
40
41
```

Output:

```
E:\Java\Java Programs>java SimpleCalC.java
Choose an operator: +, -, *, or /
+
Enter first number:
33
Enter second number:
44
33.0 + 44.0 = 77.0

E:\Java\Java Programs>java SimpleCalC.java
Choose an operator: +, -, *, or /
-
Enter first number:
44
Enter second number:
33
44.0 - 33.0 = 11.0

E:\Java\Java Programs>java SimpleCalC.java
Choose an operator: +, -, *, or /
/
Enter first number:
44
Enter second number:
33
44.0 / 33.0 = 1.3333333333333333
```

10) Vowel Or Consonant

Code:

```
J VowelConsonant.java > ...
1  import java.util.Scanner;
2
3  public class VowelConsonant {
4      public static void main(String[] args) {
5          Scanner scanner = new Scanner(System.in);
6
7          System.out.print("Enter a character: ");
8          char ch = scanner.next().charAt(0); // Read the first character input
9
10         if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' ||
11             ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U') {
12             System.out.println(ch + " is a vowel");
13         } else if (Character.isLetter(ch)) { // Check if it's a Letter
14             System.out.println(ch + " is a consonant");
15         } else {
16             System.out.println("Invalid input! Please enter an alphabetic character.");
17         }
18
19         scanner.close();
20     }
21 }
22
```

Output:

```
E:\Java\Java Programs>java VowelConsonant.java
Enter a character: i
i is a vowel

E:\Java\Java Programs>java VowelConsonant.java
Enter a character: f
f is a consonant
```


INHERITANCE PROGRAMS

4) SINGLE INHERITANCE PROGRAMS:

4A) SIMPLE ANIMAL BARKS PROGRAM:

Code:

```
class Animal
{ void sound() {
    System.out.println("Animal makes a sound");
}
}
```

```
class Dog extends Animal
{ void bark() {
    System.out.println("Dog barks");
}
}
```

```
public class SingleInheritance {
    public static void main(String[] args)
    { Dog d = new Dog();
      d.sound();
      d.bark();
    }
}
```

SCREENSHOT:

```
Microsoft Windows [Version 10.0.26120.3380]
(c) Microsoft Corporation. All rights reserved.

E:\MyPrograms\java>javac SingleInheritance.java

E:\MyPrograms\java>java SingleInheritance.java
Animal makes a sound
Dog barks

E:\MyPrograms\java>
```

4B) Single Inheritance with Constructor:

Code:

```
class Vehicle
```

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

```
class Car extends Vehicle
```

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

```
public class SingleInheritanceConstructor
```

```
    { public static void main(String[] args) {
        Car c = new Car();
    }
}
```

SCREENSHOT:

```
E:\MyPrograms\java>java SingleInheritanceConstructor.java
Vehicle is created
Car is created
```

5) MULTILEVEL INHERITANCE PROGRAMS:**5A) BANK MULTILEVEL INHERITANCE PROGRAM:**

Code:

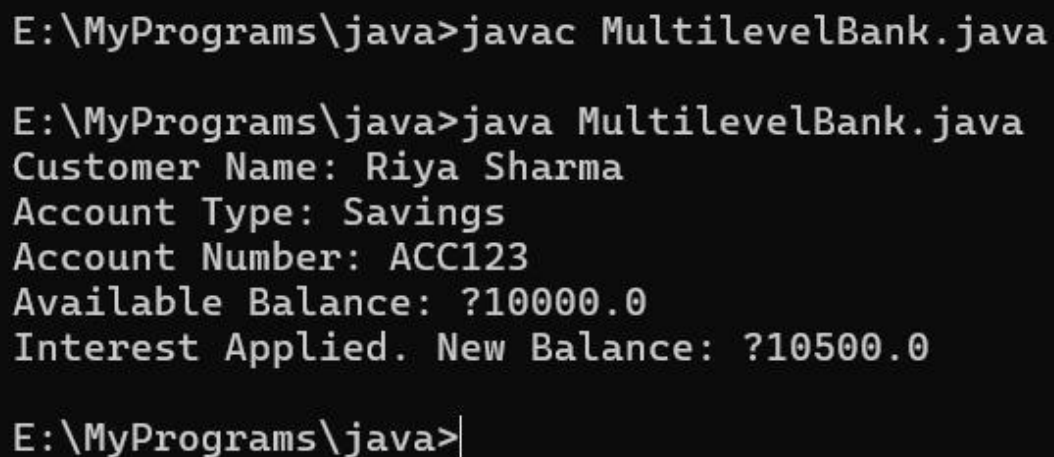
```
class Account {  
    String accNumber = "ACC123";  
    double balance = 10000;  
  
    void displayBalance() {  
        System.out.println("Account Number: " + accNumber);  
        System.out.println("Available Balance: ₹" + balance);  
    }  
}  
  
class SavingsAccount extends Account  
{ double interestRate = 0.05;  
  
    void applyInterest() {  
        balance += balance * interestRate;  
        System.out.println("Interest Applied. New Balance: ₹" + balance);  
    }  
}
```

```
class Customer extends SavingsAccount
```

```
    { void customerDetails() {  
        System.out.println("Customer Name: Riya Sharma");  
        System.out.println("Account Type: Savings");  
    }  
}
```

```
public class MultilevelBank {  
    public static void main(String[] args)  
    { Customer c = new Customer();  
      c.customerDetails();  
      c.displayBalance();  
      c.applyInterest();  
    }  
}
```

SCREENSHOT:



```
E:\MyPrograms\java>javac MultilevelBank.java  
  
E:\MyPrograms\java>java MultilevelBank.java  
Customer Name: Riya Sharma  
Account Type: Savings  
Account Number: ACC123  
Available Balance: ?10000.0  
Interest Applied. New Balance: ?10500.0  
  
E:\MyPrograms\java>|
```

5B) Multilevel Inheritance with Constructor Chaining – E-Commerce:

Code:

```
class Product
{
    Product() {
        System.out.println("Product initialized.");
    }

    void productInfo()
    {
        System.out.println("Product: Smartwatch");
    }
}

class Order extends Product
{
    Order() {
        System.out.println("Order placed successfully.");
    }

    void orderStatus()
    {
        System.out.println("Status: Confirmed");
    }
}

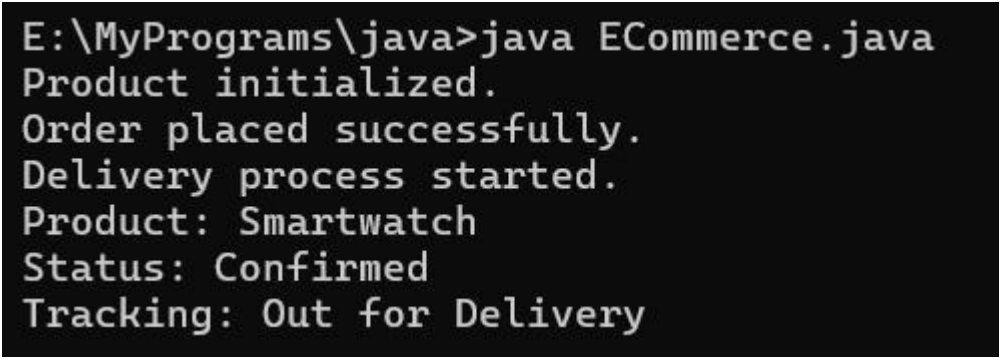
class Delivery extends Order
{
    Delivery() {
        System.out.println("Delivery process started.");
    }

    void trackOrder() {
        System.out.println("Tracking: Out for Delivery");
    }
}
```

```
}

public class ECommerce {
    public static void main(String[] args)
    {
        Delivery d = new Delivery();
        d.productInfo();
        d.orderStatus();
        d.trackOrder();
    }
}
```

SCREENSHOT:



```
E:\MyPrograms\java>java ECommerce.java
Product initialized.
Order placed successfully.
Delivery process started.
Product: Smartwatch
Status: Confirmed
Tracking: Out for Delivery
```

6) HIERARCHICAL INHERITANCE PROGRAMS:

6A) University System:

Code:

```
class University {
    void universityInfo() {
        System.out.println("University: Amrita Vishwa Vidyapeetham");
    }
}
```

```
class Department extends University
```

```
    { void departmentInfo() {  
        System.out.println("Department: Computer Science and Engineering");  
    }  
}
```

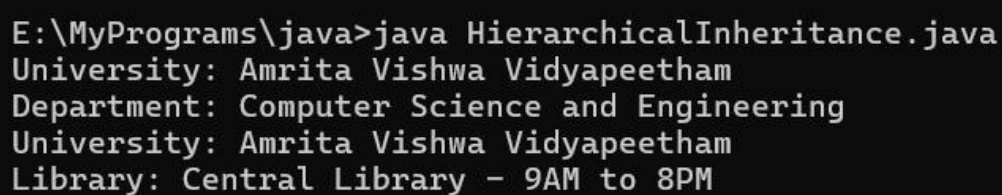
```
class Library extends University
```

```
    { void libraryInfo() {  
        System.out.println("Library: Central Library - 9AM to 8PM");  
    }  
}
```

```
public class HierarchicalInheritance
```

```
    { public static void main(String[] args)  
    {  
        Department dept = new Department();  
        dept.universityInfo();  
        dept.departmentInfo();  
  
        Library lib = new Library();  
        lib.universityInfo();  
        lib.libraryInfo();  
    }  
}
```

SCREENSHOT:

A screenshot of a terminal window showing the execution of a Java program. The command 'E:\MyPrograms\java>java HierarchicalInheritance.java' is entered. The output consists of four lines: 'University: Amrita Vishwa Vidyapeetham', 'Department: Computer Science and Engineering', 'University: Amrita Vishwa Vidyapeetham', and 'Library: Central Library - 9AM to 8PM'.

```
E:\MyPrograms\java>java HierarchicalInheritance.java  
University: Amrita Vishwa Vidyapeetham  
Department: Computer Science and Engineering  
University: Amrita Vishwa Vidyapeetham  
Library: Central Library - 9AM to 8PM
```

6B) Vehicle System:

Code:

```
class Vehicle
{ void type() {
    System.out.println("Vehicle Type: Transport");
}
}
```

```
class Car extends Vehicle
{ void carDetails() {
    System.out.println("Car: Sedan, 5 Seater");
}
}
```

```
class Bike extends Vehicle
{ void bikeDetails() {
    System.out.println("Bike: Sport, 2 Seater");
}
}
```

```
public class VehicleHierarchy {
    public static void main(String[] args)
    { Car c = new Car();
      c.type();
      c.carDetails();
      Bike b = new Bike();
      b.type();
      b.bikeDetails();
    }
}
```


SCREENSHOT:

```
E:\MyPrograms\java>java VehicleHierarchy.java
Vehicle Type: Transport
Car: Sedan, 5 Seater
Vehicle Type: Transport
Bike: Sport, 2 Seater
```

7) HYBRID INHERITANCE PROGRAMS:

7A) Student Performance System:

Code:

```
interface Exam
```

```
    { void
      giveExam();
    }
```

```
class Student {
```

```
    void studentInfo() {
        System.out.println("Student: Kavin, Roll No: CSE102");
    }
}
```

```
class Result extends Student implements Exam
```

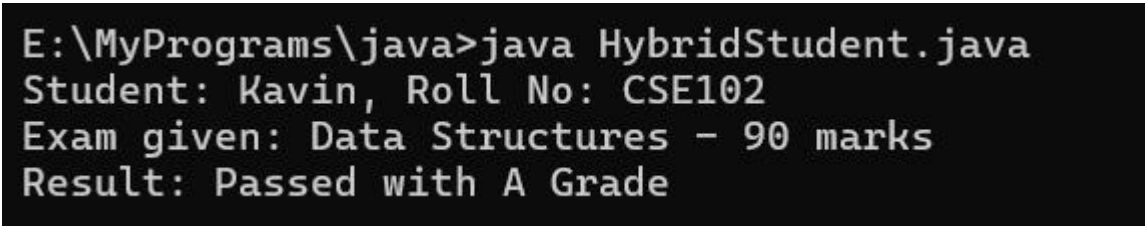
```
    { public void giveExam() {
        System.out.println("Exam given: Data Structures - 90 marks");
    }
```

```
}

void showResult() {
    System.out.println("Result: Passed with A Grade");
}

}

public class HybridStudent {
    public static void main(String[] args)
    { Result r = new Result();
      r.studentInfo();
      r.giveExam();
      r.showResult();
    }
}
```

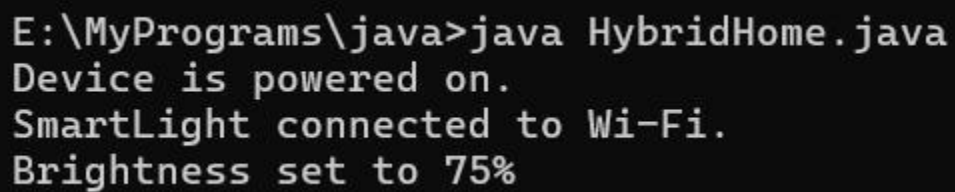
SCREENSHOT:

```
E:\MyPrograms\java>java HybridStudent.java
Student: Kavin, Roll No: CSE102
Exam given: Data Structures - 90 marks
Result: Passed with A Grade
```

7B) Smart Home System:**Code:**

```
interface Internet
{ void connect();
}
```

```
class Device {  
    void powerOn() {  
        System.out.println("Device is powered on.");  
    }  
}  
  
class SmartLight extends Device implements Internet  
{ public void connect() {  
    System.out.println("SmartLight connected to Wi-Fi.");  
}  
  
    void setBrightness()  
    { System.out.println("Brightness set to 75%");  
    }  
}  
  
public class HybridHome {  
    public static void main(String[] args)  
    { SmartLight light = new SmartLight();  
      light.powerOn();  
      light.connect();  
      light.setBrightness();  
    }  
}
```

SCREENSHOT:

```
E:\MyPrograms\java>java HybridHome.java  
Device is powered on.  
SmartLight connected to Wi-Fi.  
Brightness set to 75%
```

8) POLYMORPHISM PROGRAMS

8A) Constructor with Student Info:

Code:

```
class Student
{
    String
    name;    int
    roll;

    Student(String n, int r)
    {
        name = n;
        roll = r;
    }

    void display()
    {
        System.out.println("Name: " + name);
        System.out.println("Roll No: " + roll);
    }
}

public class ConstructorExample
{
    public static void main(String[] args)
    {
        Student s = new Student("Kavin", 119);
        s.display();
    }
}
```

Screenshot:



```
E:\MyPrograms\java>java ConstructorExample.java
Name: Kavin
Roll No: 119
```

9) CONSTRUCTOR OVERLOADING PROGRAMS

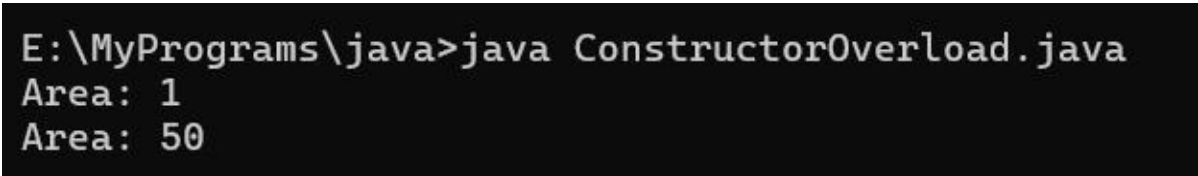
9A) Constructor with Student Info:

Code:

```
class Rectangle
{
    int length,
    width;
    Rectangle() {
        length = 1;
        width = 1;
    }
    Rectangle(int l, int w)
    {
        length = l;
        width = w;
    }
    void area() {
        System.out.println("Area: " + (length * width));
    }
}

public class ConstructorOverload
{
    public static void main(String[] args)
    {
        Rectangle r1 = new Rectangle();
        Rectangle r2 = new Rectangle(10, 5);
        r1.area();
        r2.area();
    }
}
```

Screenshot:



```
E:\MyPrograms\java>java ConstructorOverload.java
Area: 1
Area: 50
```

10) METHOD OVERLOADING PROGRAMS**10A) Calculator with Method Overloading:****Code:**


```
class Calculator {  
    int add(int a, int b)  
        { return a + b;  
    }  
  
    double add(double a, double b)  
        { return a + b;  
    }  
}  
  
public class MethodOverload1 {  
    public static void main(String[] args)  
        { Calculator calc = new Calculator();  
        System.out.println("Sum (int): " + calc.add(5, 10));  
        System.out.println("Sum (double): " + calc.add(5.5, 3.2));  
    }  
}
```

ScreenShot:

```
E:\MyPrograms\java>java MethodOverload1.java  
Sum (int): 15  
Sum (double): 8.7
```

10B) Display Function Overloading:**Code:**

```
class Display {  
    void show(String name)  
        { System.out.println("Name: " + name);  
    }  
  
    void show(int age)  
        { System.out.println("Age: " + age);  
    }  
  
    void show(String name, int age)  
        { System.out.println("Name: " + name + ", Age: " + age);  
    }  
}  
  
public class MethodOverload2 {  
    public static void main(String[] args)  
        { Display d = new Display();  
        d.show("Kavin");  
        d.show(19);  
        d.show("Kavin", 19);  
    }  
}
```

Screenshot:

```
E:\MyPrograms\java>java MethodOverload2.java  
Name: Kavin  
Age: 19  
Name: Kavin, Age: 19
```

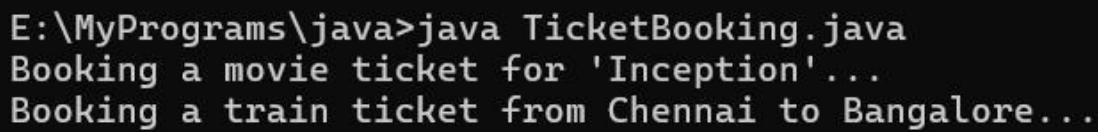
11) METHOD OVERRIDING PROGRAMS**11A) E-Ticket Booking System:****Code:**

```
class Ticket
{
    void book()
    {
        System.out.println("Booking a general ticket...");
    }
}

class MovieTicket extends Ticket
{
    @Override
    void book() {
        System.out.println("Booking a movie ticket for 'Inception'...");
    }
}

class TrainTicket extends Ticket
{
    @Override
    void book() {
        System.out.println("Booking a train ticket from Chennai to Bangalore...");
    }
}

public class TicketBooking {
    public static void main(String[] args)
    {
        Ticket t1 = new MovieTicket();
        Ticket t2 = new TrainTicket();
        t1.book();
        t2.book();
    }
}
```


Screenshot:

```
E:\MyPrograms\java>java TicketBooking.java
Booking a movie ticket for 'Inception'...
Booking a train ticket from Chennai to Bangalore...
```

11B) Banking Transaction System:**Code:**

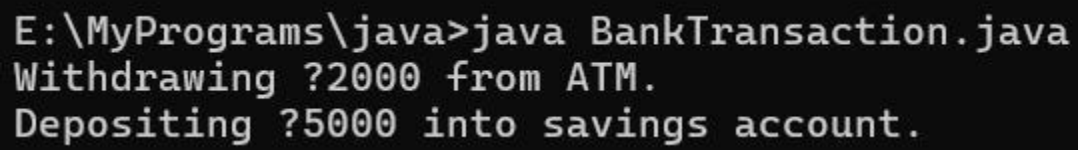
```
class Transaction
{
    void perform()
    {
        System.out.println("Performing a generic transaction...");
    }
}

class Withdrawal extends Transaction
{
    @Override
    void perform() {
        System.out.println("Withdrawing ₹2000 from ATM.");
    }
}

class Deposit extends Transaction
{
    @Override
    void perform() {
        System.out.println("Depositing ₹5000 into savings account.");
    }
}

public class BankTransaction {
    public static void main(String[] args)
    {
        Transaction tx1 = new Withdrawal();
        Transaction tx2 = new Deposit();
    }
}
```

```
        tx1.perform();  
        tx2.perform();  
    }  
}
```

Screenshot:A screenshot of a command prompt window showing the execution of a Java program. The text displayed is: E:\MyPrograms\java>java BankTransaction.java
Withdrawing ?2000 from ATM.
Depositing ?5000 into savings account.

The screenshot shows a black background with white text. The first line is the command prompt path and command. The subsequent lines are the output of the program, showing a withdrawal and a deposit, both using a question mark as a currency symbol.

(Rupees Symbol is changed to “?”)

ABSTRACTION PROGRAMS:**12) INTERFACE PROGRAMS:****12A)E-Commerce Payment Interface:****Code:**

```
interface PaymentGateway {  
    void processPayment(double amount);  
}  
  
class PayPal implements PaymentGateway  
{ public void processPayment(double amount)  
{  
    System.out.println("Processing PayPal payment of ₹" + amount);  
}  
}  
  
class Razorpay implements PaymentGateway  
{ public void processPayment(double amount)  
{  
    System.out.println("Processing Razorpay payment of ₹" + amount);  
}  
}  
  
public class ECommercePayment  
{ public static void main(String[] args)  
{  
    PaymentGateway pg;  
  
    pg = new PayPal();  
    pg.processPayment(1500.00);  
  
    pg = new Razorpay();  
    pg.processPayment(2999.99);  
}  
}
```

Screenshot:

```
E:\MyPrograms\java>java ECommercePayment.java
Processing PayPal payment of ?1500.0
Processing Razorpay payment of ?2999.99
```

12B)Smart Device Control Interface:**Code:**

```
interface SmartDevice
```

```
    { void turnOn();
```

```
    void turnOff();
```

```
}
```

```
class SmartFan implements SmartDevice
```

```
    { public void turnOn() {
```

```
        System.out.println("Smart Fan turned ON.");
```

```
    }
```

```
    public void turnOff()
```

```
    { System.out.println("Smart Fan turned
```

```
    OFF.");
```

```
    }
```

```
}
```

```
class SmartBulb implements SmartDevice
```

```
    { public void turnOn() {
```

```
        System.out.println("Smart Bulb turned ON.");
```

```
    }
```

```
    public void turnOff()
```

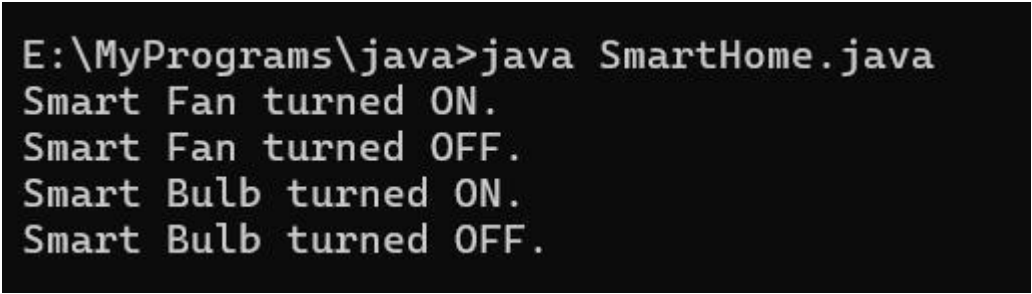
```
    { System.out.println("Smart Bulb turned
```

```
    OFF.");
```

```
    }
```

```
}
```

```
public class SmartHome {  
    public static void main(String[] args)  
    {  
        SmartDevice device1 = new SmartFan();  
        SmartDevice device2 = new SmartBulb();  
  
        device1.turnOn();  
        device1.turnOff();  
  
        device2.turnOn();  
        device2.turnOff();  
    }  
}
```

Screenshot:

```
E:\MyPrograms\java>java SmartHome.java  
Smart Fan turned ON.  
Smart Fan turned OFF.  
Smart Bulb turned ON.  
Smart Bulb turned OFF.
```

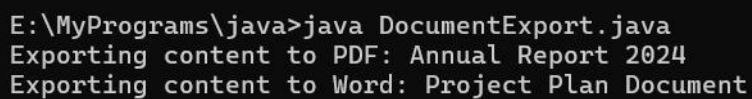
12C)Document Export Interface:**Code:**

```
interface Exportable {  
    void export(String content);  
}  
  
class PDFExporter implements Exportable  
{  
    public void export(String content) {  
        System.out.println("Exporting content to PDF: " + content);  
    }  
}
```

```
class WordExporter implements Exportable
{
    public void export(String content) {
        System.out.println("Exporting content to Word: " + content);
    }
}
```

```
public class DocumentExport {
    public static void main(String[] args)
    {
        Exportable ex1 = new PDFExporter();
        Exportable ex2 = new WordExporter();

        ex1.export("Annual Report 2024");
        ex2.export("Project Plan Document");
    }
}
```

Screenshot:

```
E:\MyPrograms\java>java DocumentExport.java
Exporting content to PDF: Annual Report 2024
Exporting content to Word: Project Plan Document
```

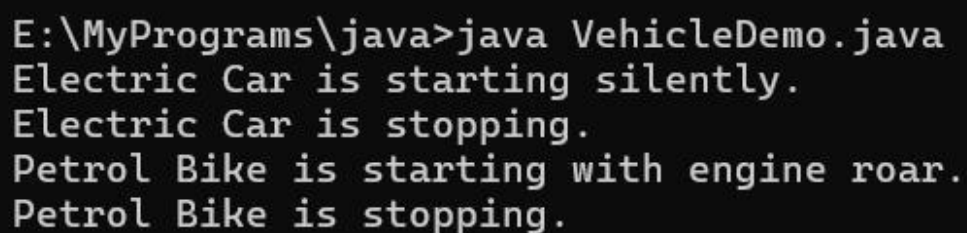
12D)Vehicle Interface:**Code:**

```
interface Vehicle
{
    void start();
    void stop();
}

class ElectricCar implements Vehicle
{
    public void start() {
        System.out.println("Electric Car is starting silently.");
    }
}
```

```
}  
public void stop() {  
    System.out.println("Electric Car is stopping.");  
}  
}  
class PetrolBike implements Vehicle  
{ public void start() {  
    System.out.println("Petrol Bike is starting with engine roar.");  
}  
public void stop() {  
    System.out.println("Petrol Bike is stopping.");  
}  
}  
public class VehicleDemo {  
    public static void main(String[] args)  
    { Vehicle v1 = new ElectricCar();  
      Vehicle v2 = new PetrolBike();  
      v1.start();  
      v1.stop();  
      v2.start();  
      v2.stop();  
    }  
}
```

Screenshot:



```
E:\MyPrograms\java>java VehicleDemo.java  
Electric Car is starting silently.  
Electric Car is stopping.  
Petrol Bike is starting with engine roar.  
Petrol Bike is stopping.
```

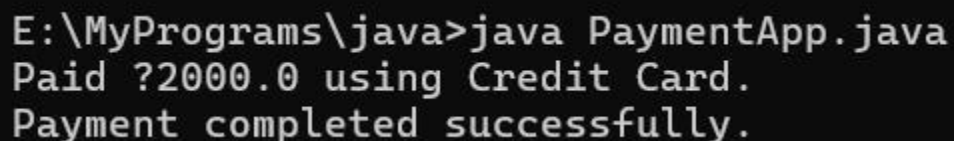
13)ABSTRACT CLASS PROGRAMS

13A)Online Payment System:

Code:

```
abstract class Payment {  
    abstract void makePayment(double amount);  
    void paymentSuccess() {  
        System.out.println("Payment completed successfully.");  
    }  
}  
  
class CreditCardPayment extends Payment  
{ void makePayment(double amount) {  
    System.out.println("Paid ₹" + amount + " using Credit Card.");  
}  
}  
  
public class PaymentApp {  
    public static void main(String[] args)  
    { Payment p = new CreditCardPayment();  
      p.makePayment(2000);  
      p.paymentSuccess();  
    }  
}
```

Screenshot:



```
E:\MyPrograms\java>java PaymentApp.java  
Paid ?2000.0 using Credit Card.  
Payment completed successfully.
```


13B)College Admission System:**Code:**

```
abstract class Admission
{
    abstract void register();
    abstract void payFees();
}

class UGAdmission extends Admission
{
    void register() {
        System.out.println("UG Student Registered");
    }

    void payFees() {
        System.out.println("UG Student paid ₹75,000 fees");
    }
}

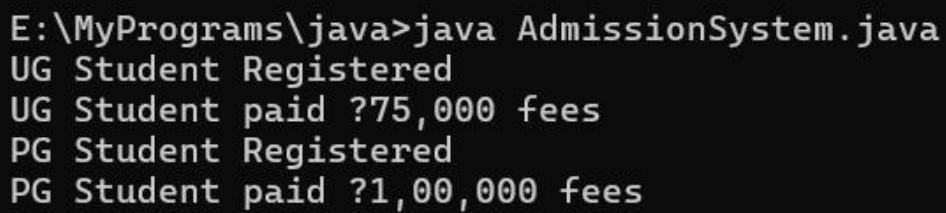
class PGAdmission extends Admission
{
    void register() {
        System.out.println("PG Student Registered");
    }

    void payFees() {
        System.out.println("PG Student paid ₹1,00,000 fees");
    }
}

public class AdmissionSystem {
    public static void main(String[] args)
    {
        Admission student1 = new UGAdmission();
        Admission student2 = new PGAdmission();
    }
}
```

```
student1.register();
student1.payFees();

student2.register();
student2.payFees();
}
}
```

Screenshot:

```
E:\MyPrograms\java>java AdmissionSystem.java
UG Student Registered
UG Student paid ?75,000 fees
PG Student Registered
PG Student paid ?1,00,000 fees
```

13C)Employee Payroll System:**Code:**

```
abstract class Employee
{
    String name;
    int empId;

    Employee(String name, int empId)
    {
        this.name = name;
        this.empId = empId;
    }

    abstract double calculateSalary();
}

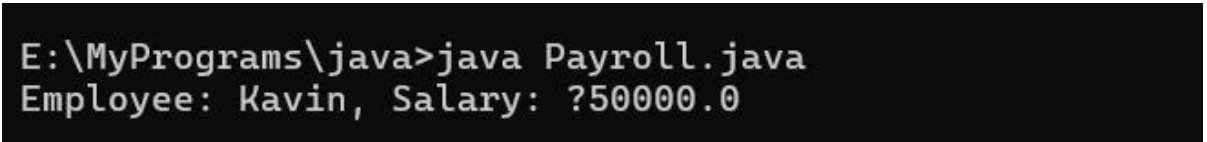
class FullTimeEmployee extends Employee
{
    double salary;
```

```
FullTimeEmployee(String name, int empId, double salary)
{
    super(name, empId);
    this.salary = salary;
}

double calculateSalary()
{
    return salary;
}

}

public class Payroll {
    public static void main(String[] args) {
        FullTimeEmployee emp = new FullTimeEmployee("Kavin", 101, 50000);
        System.out.println("Employee: " + emp.name + ", Salary: ₹" + emp.calculateSalary());
    }
}
```

Screenshot:

```
E:\MyPrograms\java>java Payroll.java
Employee: Kavin, Salary: ₹50000.0
```

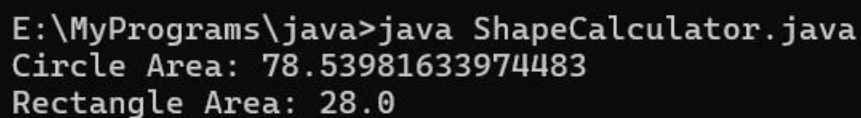
13D)Shape Area Calculator:**Code:**

```
abstract class Shape
{
    abstract double area();
}

class Circle extends Shape
{
    double radius;
    Circle(double r) {
        radius = r;
    }
}
```

```
}  
double area() {  
    return Math.PI * radius * radius;  
}  
}  
  
class Rectangle extends Shape  
{ double length, width;  
Rectangle(double l, double w) {  
    length = l;  
    width = w;  
}  
double area() {  
    return length * width;  
}  
}  
  
public class ShapeCalculator {  
    public static void main(String[] args)  
    { Shape s1 = new Circle(5);  
      Shape s2 = new Rectangle(4, 7);  
      System.out.println("Circle Area: " + s1.area());  
      System.out.println("Rectangle Area: " + s2.area());  
    }  
}
```

Screenshot:



```
E:\MyPrograms\java>java ShapeCalculator.java  
Circle Area: 78.53981633974483  
Rectangle Area: 28.0
```

ENCAPSULATIONPROGRAMS:**14)ENCAPSULATION PROGRAMS:****14A)Bank Account System:****Code:**

```
class BankAccount {  
    private String accountHolder;  
    private double balance;  
  
    public void setAccountHolder(String name)  
    { accountHolder = name;  
    }  
  
    public String getAccountHolder()  
    { return accountHolder;  
    }  
  
    public void deposit(double amount)  
    { if (amount > 0) balance +=  
      amount;  
    }  
  
    public double getBalance()  
    { return balance;  
    }  
}  
  
public class EncapsulationBank {  
    public static void main(String[] args)  
    { BankAccount account = new BankAccount();  
      account.setAccountHolder("Kavin");  
      account.deposit(10000);  
  
      System.out.println("Account Holder: " + account.getAccountHolder());  
    }  
}
```

```
        System.out.println("Balance: ₹" + account.getBalance());
    }
}
```

Screenshot:

```
E:\MyPrograms\java>java EncapsulationBank.java
Account Holder: Kavin
Balance: ₹10000.0
```

14B)Student Report System:**Code:**

```
class Student {
    private String name;
    private int marks;

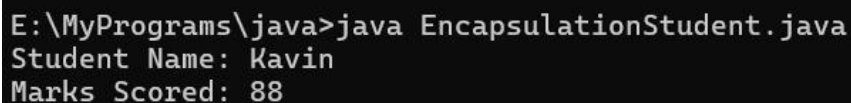
    public void setDetails(String n, int m)
    {
        name = n;
        marks = m;
    }

    public String getName()
    {
        return name;
    }

    public int getMarks()
    {
        return marks;
    }
}

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

```
s.setDetails("Kavin", 88);  
  
System.out.println("Student Name: " + s.getName());  
  
System.out.println("Marks Scored: " + s.getMarks());  
  
}  
  
}
```

Screenshot:

```
E:\MyPrograms\java>java EncapsulationStudent.java  
Student Name: Kavin  
Marks Scored: 88
```

14C)Login System:**Code:**

```
class User {  
    private String username;  
    private String password;  
    public void setCredentials(String u, String p)  
    {  
        username = u;  
        password = p;  
    }  
    public boolean login(String inputUser, String inputPass) {  
        return username.equals(inputUser) && password.equals(inputPass);  
    }  
}  
  
public class EncapsulationLogin {  
    public static void main(String[] args)  
    {  
        User user = new User();  
        user.setCredentials("admin", "1234");  
        System.out.println("Login success: " + user.login("admin", "1234"));  
        System.out.println("Login failed: " + user.login("admin", "wrong"));  
    }  
}
```

Screenshot:

```
E:\MyPrograms\java>java EncapsulationLogin.java
Login success: true
Login failed: false
```

14D)Temperature Conversion:**Code:**

```
class Temperature
{
    private double celsius;
    public void setCelsius(double c)
    {
        if (c >= -273.15) celsius = c;
    }
    public double getFahrenheit()
    {
        return (celsius * 9 / 5) + 32;
    }
}

public class EncapsulationTemperature
{
    public static void main(String[] args)
    {
        Temperature t = new Temperature();
        t.setCelsius(25);
        System.out.println("Temperature in Fahrenheit: " + t.getFahrenheit());
    }
}
```

Screenshot:

```
E:\MyPrograms\java>java EncapsulationTemperature.java
Temperature in Fahrenheit: 77.0
```

15) PACKAGES PROGRAMS:

15A) JAVA PROGRAM TO CONVERT TEMPERATURE**Code:**

```
package converter;

public class TemperatureConverter {

    public double toFahrenheit(double celsius)

        { return (celsius * 9/5) + 32;

        }

    public double toCelsius(double fahrenheit)

        { return (fahrenheit - 32) * 5/9;

        }

}

import converter.TemperatureConverter;

public class Package2 {

    public static void main(String[] args)


        { TemperatureConverter tc = new TemperatureConverter();

        System.out.println("25°C in Fahrenheit: " + tc.toFahrenheit(25));

        System.out.println("77°F in Celsius: " + tc.toCelsius(77));

        }

}
```

Screenshot:

```
25°C in Fahrenheit: 77.0
77°F in Celsius: 25.0
```

15B) JAVA PROGRAM FOR EXPONENTIAL FUNCTION:**Code:**

```
package power;

public class exponents

    { public int square(int x)

        {

            return x * x;

        }

    public int cube(int x)
```

```
{ return x * x * x;

}

}

import power.exponents;
public class Package1 {
    public static void main(String[] args)
    { exponents num = new exponents();
      System.out.println("Square of 3: " + num.square(3));
      System.out.println("Cube of 3: " + num.cube(3));
    }
}
```

Screenshot:



```
Square of 3: 9
Cube of 3: 27
```

15C) JAVA PROGRAM TO GET SQUARE OF 5, IP, DATE&TIME:

Code:

```
import java.text.SimpleDateFormat;
import java.util.Date;
import java.net.InetAddress;
public class BuiltInPackageExample2
{ public static void main(String[] args) {
    double power = Math.pow(2, 5);
    double sqrt = Math.sqrt(64);
    System.out.println("2^5 = " + power);
    System.out.println("Square root of 64 = " + sqrt);
    SimpleDateFormat formatter = new SimpleDateFormat("dd-MM-yyyy HH:mm:ss");
    Date now = new Date();
    System.out.println("Formatted Date: " + formatter.format(now));
}
```

```

    try {

        InetAddress ip = InetAddress.getLocalHost();

        System.out.println("Your System's IP Address: " + ip.getHostAddress());

    } catch (Exception e)

        { System.out.println("Unable to get IP

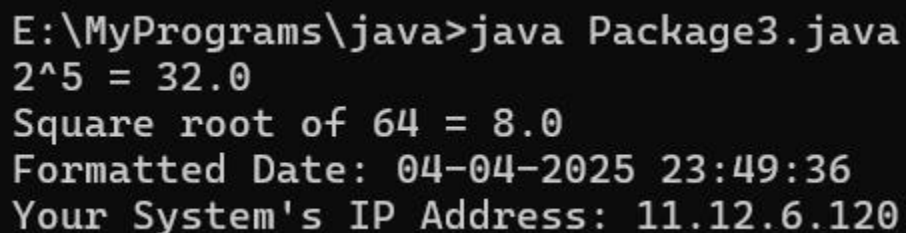
        Address.");

        }

    }

}

```

Screenshot:


```

E:\MyPrograms\java>java Package3.java
2^5 = 32.0
Square root of 64 = 8.0
Formatted Date: 04-04-2025 23:49:36
Your System's IP Address: 11.12.6.120

```

15D) JAVA PROGRAM TO GET DATE&TIME:**Code:**

```

import java.util.Date;

import java.text.DateFormat;

import java.lang.System;

public class Package4 {

    public static void main(String[] args)

        { Date now = new Date();

        DateFormat df = DateFormat.getDateInstance(DateFormat.FULL,

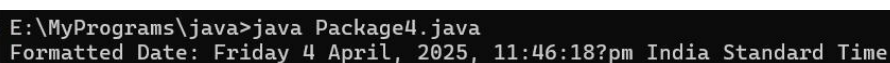
        DateFormat.FULL);

        System.out.println("Formatted Date: " + df.format(now));

        }

}

```

Screenshot:


```

E:\MyPrograms\java>java Package4.java
Formatted Date: Friday 4 April, 2025, 11:46:18pm India Standard Time

```

16)EXCEPTION HANDLING PROGRAMS:

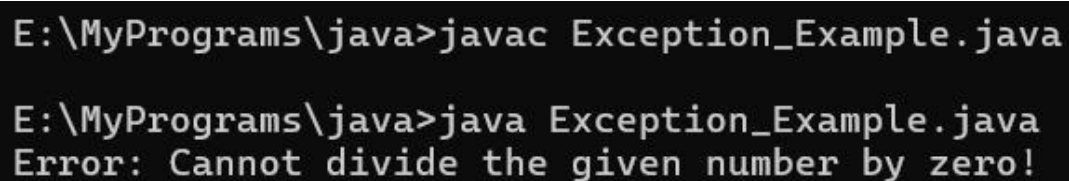
16A)Java program that throws an exception and catch it using a try-catch block.:

Code:

```
public class Exception_Example
{
    public static void main(String[] args)
    {
        try {
            int result = divideNumbers(5, 0);
            System.out.println("Result: " + result);
        } catch (ArithmeticException e)
        {
            System.out.println("Error: " +
                e.getMessage());
        }
    }
}

public static int divideNumbers(int dividend, int divisor)
{
    if (divisor == 0) {
        throw new ArithmeticException("Cannot divide the given number by zero!");
    }
    return dividend / divisor;
}
}
```

Screenshot:



```
E:\MyPrograms\java>javac Exception_Example.java

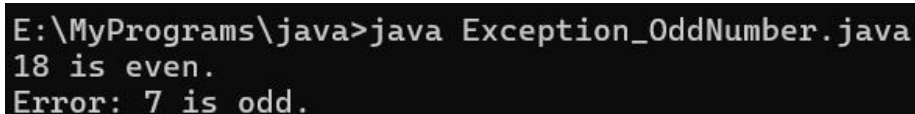
E:\MyPrograms\java>java Exception_Example.java
Error: Cannot divide the given number by zero!
```

16B)Java program to create a method that takes an integer as a parameter and throws an exception if the number is odd.:

Code:

```
public class Exception_OddNumber
{
    public static void main(String[] args)
    {
        int n = 18;
        trynumber(n);
        n = 7;
        trynumber(n);
    }
    public static void trynumber(int n)
    {
        try {
            checkEvenNumber(n);
            System.out.println(n + " is even.");
        } catch (IllegalArgumentException e)
        {
            System.out.println("Error: " +
                e.getMessage());
        }
    }
    public static void checkEvenNumber(int number)
    {
        if (number % 2 != 0) {
            throw new IllegalArgumentException(number + " is odd.");
        }
    }
}
```

Screenshot:



```
E:\MyPrograms\java>java Exception_OddNumber.java
18 is even.
Error: 7 is odd.
```

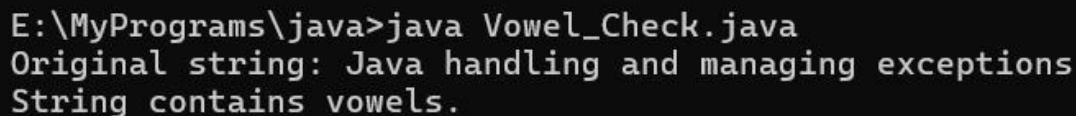
16C) Java program to create a method that takes a string as input and throws an exception if the string does not contain vowels:

Code:

```
public class Vowel_Check {  
    public static void main(String[] args)  
    { try {  
        String text = "Java handling and managing exceptions ";  
        // String text = "Typy gyps fly.";  
        System.out.println("Original string: " + text);  
        checkVowels(text);  
        System.out.println("String contains vowels.");  
    } catch (NoVowelsException e)  
    { System.out.println("Error: " +  
        e.getMessage());  
    }  
}  
  
public static void checkVowels(String text) throws NoVowelsException  
{ boolean containsVowels = false;  
    String vowels = "aeiouAEIOU";  
    for (int i = 0; i < text.length(); i++)  
    { char ch = text.charAt(i);  
        if (vowels.contains(String.valueOf(ch)))  
        { containsVowels = true;  
            break;  
        }  
    }  
    if (!containsVowels) {  
        throw new NoVowelsException("String does not contain any vowels.");  
    }  
}
```

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

Screenshot:



```
E:\MyPrograms\java>java Vowel_Check.java  
Original string: Java handling and managing exceptions  
String contains vowels.
```

16D) Java program that reads a list of integers from the user and throws an exception if any numbers are duplicates:

Code:

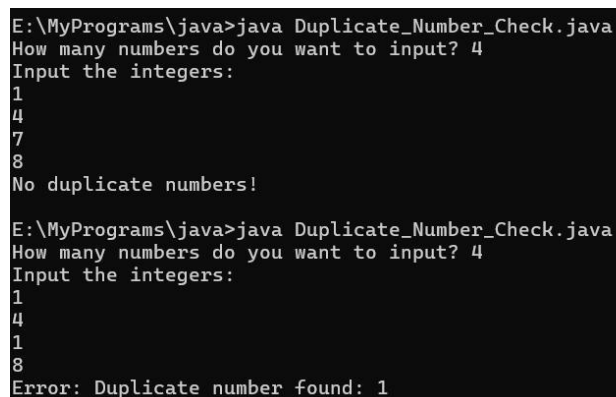
```
import java.util.*;  
  
public class Duplicate_Number_Check  
{ public static void main(String[] args)  
    { try {  
        List < Integer > numbers = readNumbersFromUser();  
        checkDuplicates(numbers);  
        System.out.println("No duplicate numbers!");  
    } catch (Duplicate_Number_Exception e)  
        { System.out.println("Error: " +  
            e.getMessage());  
        }  
    }  
}  
  
public static List < Integer > readNumbersFromUser()  
    { List < Integer > numbers = new ArrayList < > ();  
    Scanner scanner = new Scanner(System.in);  
    System.out.print("How many numbers do you want to input? ");  
    int count = scanner.nextInt();
```

```
System.out.println("Input the integers:");
for (int i = 0; i < count; i++) {

    int num = scanner.nextInt();
    numbers.add(num);
}
scanner.close();
return numbers;
}

public static void checkDuplicates(List < Integer > numbers) throws
Duplicate_Number_Exception {
    Set < Integer > uniqueNumbers = new HashSet < > ();
    for (int num: numbers) {
        if (uniqueNumbers.contains(num)) {
            throw new Duplicate_Number_Exception("Duplicate number found: " + num);
        }
        uniqueNumbers.add(num);
    }
}

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

Screenshot:

```
E:\MyPrograms\java>java Duplicate_Number_Check.java
How many numbers do you want to input? 4
Input the integers:
1
4
7
8
No duplicate numbers!

E:\MyPrograms\java>java Duplicate_Number_Check.java
How many numbers do you want to input? 4
Input the integers:
1
4
1
8
Error: Duplicate number found: 1
```


17)FILE HANDLING PROGRAMS:**17A)Java - Print File Content, Display File:****Code:**

```
import java.io.*;

public class PrintFile {

    public static void main(String args[]) throws IOException

    { File fileName = new File("d:/sample.txt");

    FileInputStream inFile = new FileInputStream("E:\MyPrograms\java\sample.txt");

    int fileLength = (int) fileName.length();

    byte Bytes[] = new byte[fileLength];

    System.out.println("File size is: " + inFile.read(Bytes));

    String file1 = new String(Bytes);

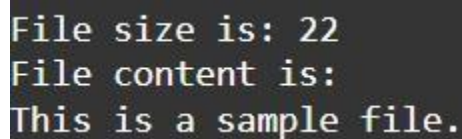
    System.out.println("File content is:\n" + file1);

    //close file

    inFile.close();

    }

}
```

Screenshot:A screenshot of a terminal window with a dark background and light-colored text. It displays the output of the Java program: 'File size is: 22', 'File content is:', and 'This is a sample file.' on three separate lines.

```
File size is: 22
File content is:
This is a sample file.
```

17B)Delete file using java program:**Code:**

```
import java.io.File;

public class ExDeleteFile {

    public static void main(String args[])

    { final String fileName = "file3.txt";

    //File class object
```

```
File objFile = new File(fileName);
//check file is exist or not, if exist delete it
if (objFile.exists() == true) {
    //file exists
    //deleting the file
    if (objFile.delete()) {
        System.out.println(objFile.getName() + " deleted successfully.");
    } else {
        System.out.println("File deletion failed!!!");
    }
} else {
    System.out.println("File does not exist!!!");
}
}
```

Screenshot:

A screenshot of a terminal window with a dark background. The text 'file3.txt deleted successfully.' is displayed in a light-colored, monospaced font.

17C)Read file line by line using

java: Code:

```
import java.io.FileInputStream;
import java.io.IOException;
import java.util.Scanner;

public class ReadLineByLine {
    public static void main(String[] args) {
        // create object of scanner class.
        Scanner Sc = new Scanner(System.in);
```

```
// enter file name.

System.out.print("Enter the file name:");

String sfilename = Sc.next();

Scanner Sc1 = null;

FileInputStream fis = null;

try {

    FileInputStream FI = new FileInputStream(sfilename);

    Sc1 = new Scanner(FI);

    // this will read data till the end of data.

    while (Sc1.hasNext()) {

        String data = Sc1.nextLine();

        // print the data.

        System.out.print("The file data is : " + data);

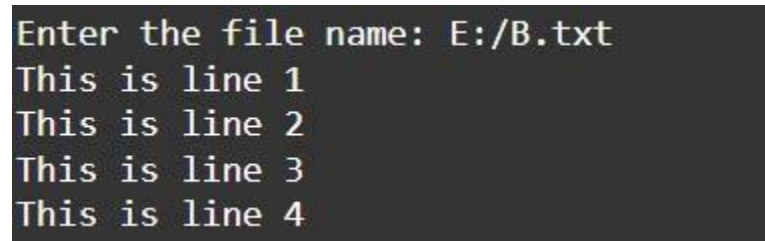
    }

} catch (IOException e)

    { System.out.println(e);

    }

}
```

Screenshot:

```
Enter the file name: E:/B.txt
This is line 1
This is line 2
This is line 3
This is line 4
```

17D) Java program to append text/string in a file:**Code:**

```
import java.io.*;

public class AppendFile {

    public static void main(String[] args) {

        String strFilePath = "E:/J A V A/IncludeHelp.txt";

        try {

            FileOutputStream fos = new FileOutputStream(strFilePath, true);

            String strContent = "Text to be appended.";

            fos.write(strContent.getBytes());

            fos.close();

            System.out.println("Content Successfully Append into File...");

        } catch (FileNotFoundException ex)

            { System.out.println("FileNotFoundException : " + ex.toString());

        } catch (IOException ioe)

            { System.out.println("IOException : " + ioe.toString());


        } catch (Exception e)

            { System.out.println("Exception: " + e.toString());

        }

    }

}
```

Screenshot:A screenshot of a console window with a dark background. The text "Content Successfully Append into File..." is displayed in a light-colored, monospaced font.