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

SCHOOL OF COMPUTING

VISHAL M.D.

CH.SC.U4CSE24150

OBJECT ORIENTED PROGRAMMING (23CSE111)

LAB RECORD

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.U4CSE24150 – V I S H A L M . D*** 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

Internal Examiner 1 Internal Examiner 2

INDEX

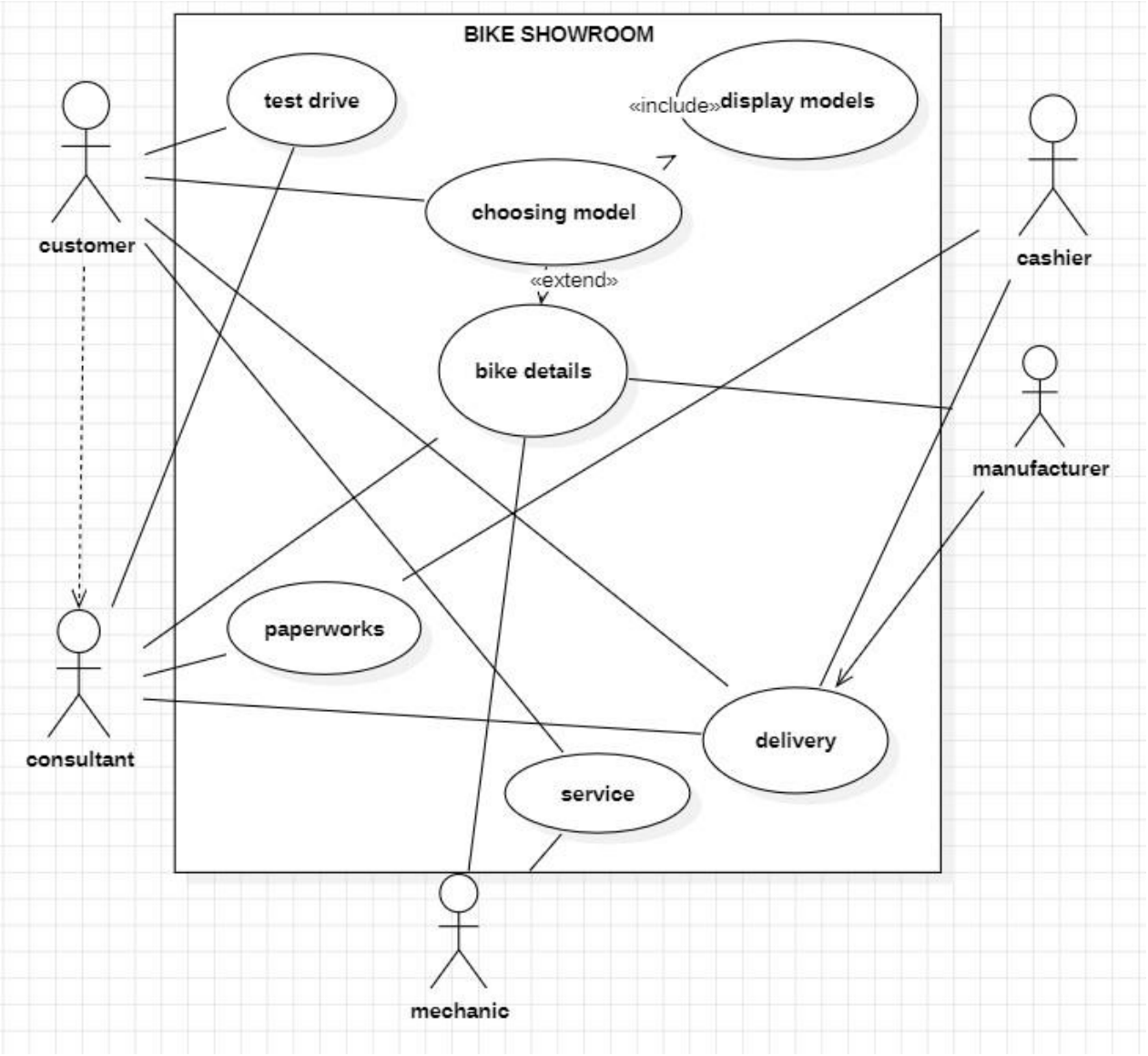
	4.a)EmpManagerSI1	
	4.b) BikeCarSI2	
5.	**MULTILEVEL INHERITANCE PROGRAMS**	
	5.a) AnimalMAMHumanMLI1	
	5.b) DevicesMLI2	
6.	**HIERARCHICAL INHERITANCE PROGRAMS**	
	6.a) ShapesHI1	
	6.b) EmpDevTestHI2	
7.	**HYBRID INHERITANCE PROGRAMS**	
	7.a) TeacherAteacherHYI1	
	7.b) TransportHYI2	
	POLYMORPHISM	
8.	**CONSTRUCTOR PROGRAMS**	
	8.a) DroneSystem	
9.	**CONSTRUCTOR OVERLOADING PROGRAMS**	
	9.a) OnlineExamSystem	
10.	**METHOD OVERLOADING PROGRAMS**	
	10.a) AreaCalculator	
	10.b) StringManipulator	
11.	**METHOD OVERRIDING PROGRAMS**	
	11.a) InterestCal	
	11.b) VehicleSound	
	ABSTRACTION	
12.	**INTERFACE PROGRAMS**	
	12.a) HomeSecuritySystem	
	12.b) MusicInterface	
	12.c) OnlineLibrary	
	12.d) PaymentInterface	
13.	**ABSTRACT CLASS PROGRAMS**	
	13.a) FoodOrderingSystem	
	13.b) OnlinePayment	
	13.c) SmartHome	
	13.d) Vehicle	
	ENCAPSULATION	
14.	**ENCAPSULATION PROGRAMS**	
	14.a) BankEncap	

	14.b) EmpEncap	
	14.c) ShoppingCartApp	
	14.d) StudentMarksApp	
15.	**PACKAGES PROGRAMS**	
	15.a)User Defined Packages-	
	15.b)User Defined Packages	
	15.c)Built – in Package(3 Packages)	
	15.d)Built – in Package(3 Packages)	
16.	**EXCEPTION HANDLING PROGRAMS**	
	16.a) AgeValidation	
	16.b) ArrayExceptionHandling	
	16.c) DivisionHandling	
	16.d) FileExceptionHandling	
17.	**FILE HANDLING PROGRAMS**	
	17.a)logwriter	
	17.b)todolist	
	17.c)studentmanagement	
	17.d)reading file	

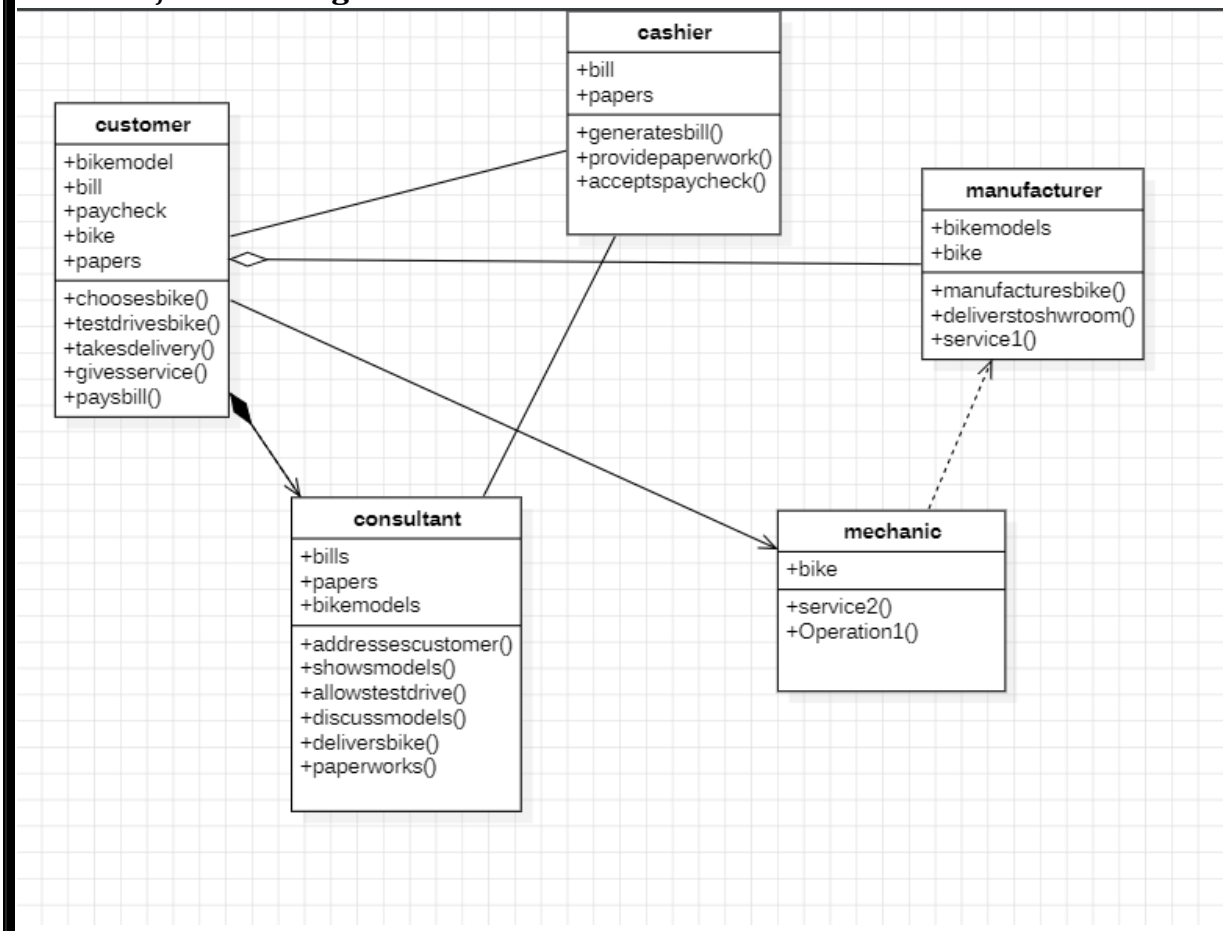
UML DIAGRAMS

1. BIKE SHOWROOM MANAGEMNT

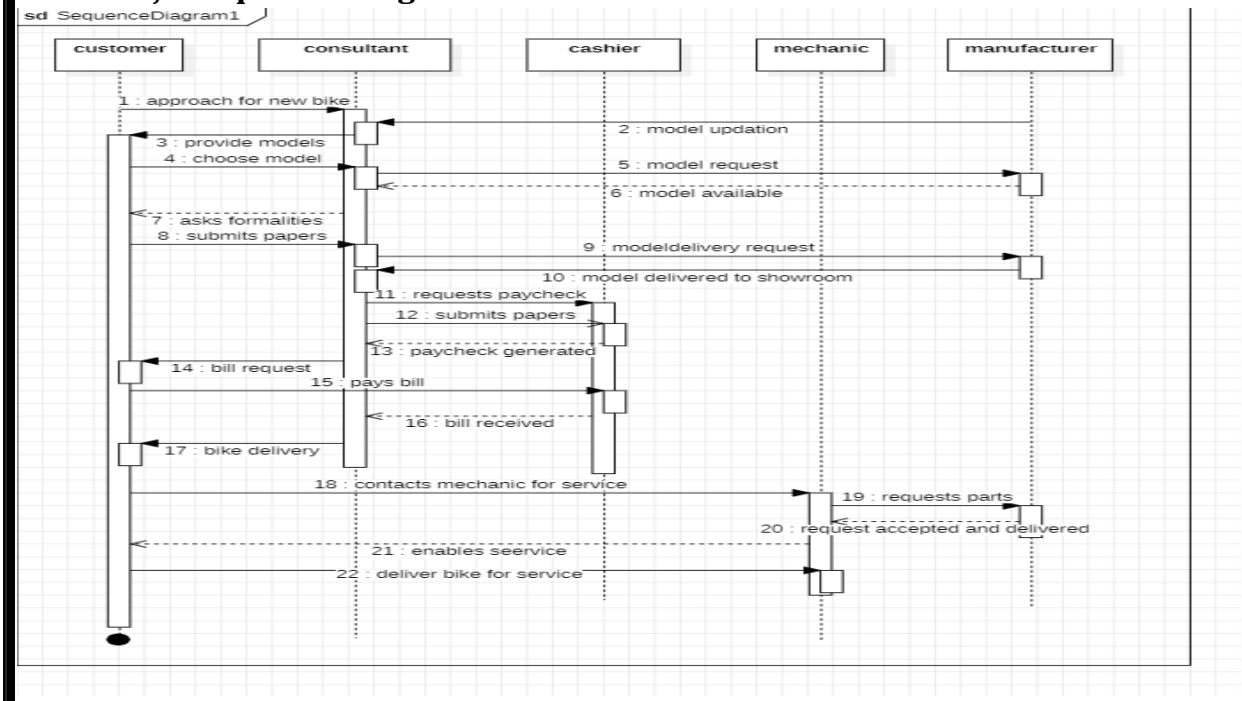
1.a) Use Case Diagram:



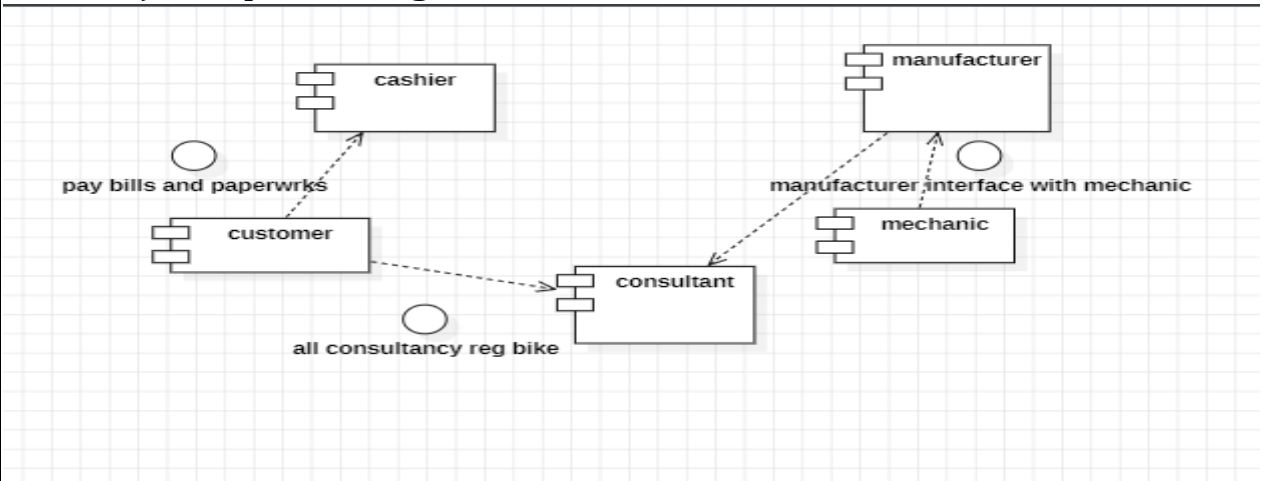
1.b) Class Diagram:



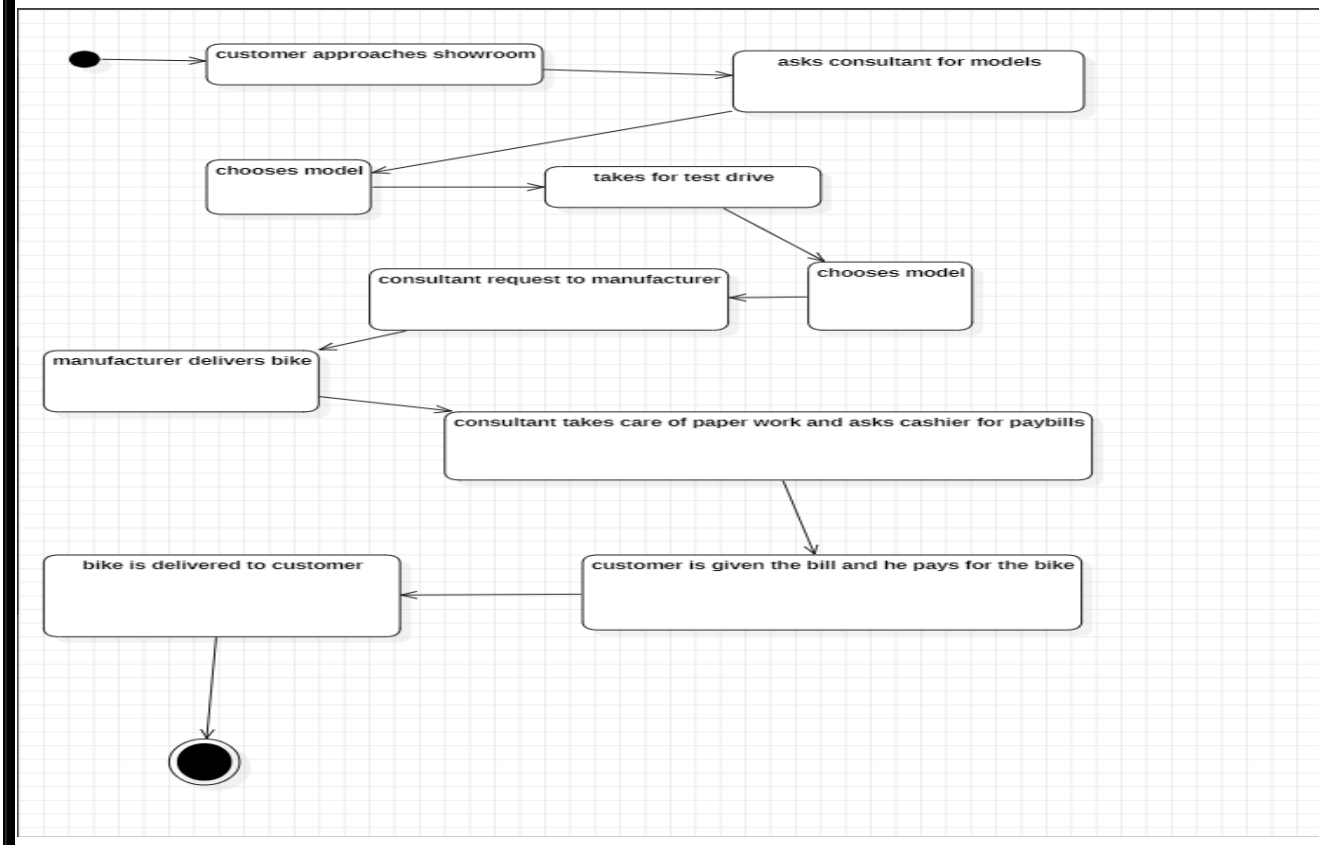
1.c) Sequence Diagram:



1.d) Component Diagram:

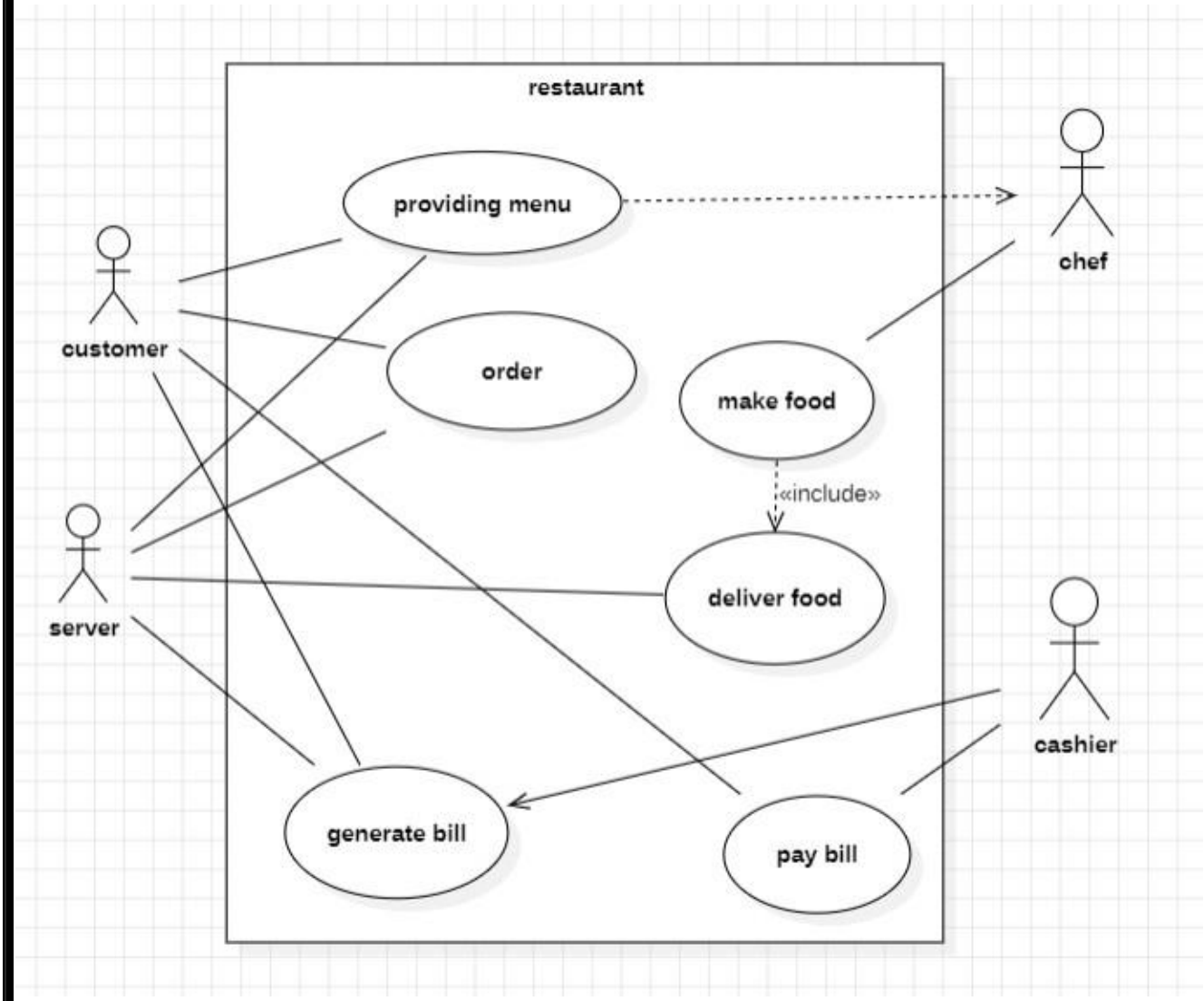


1.e) Activity Diagram:

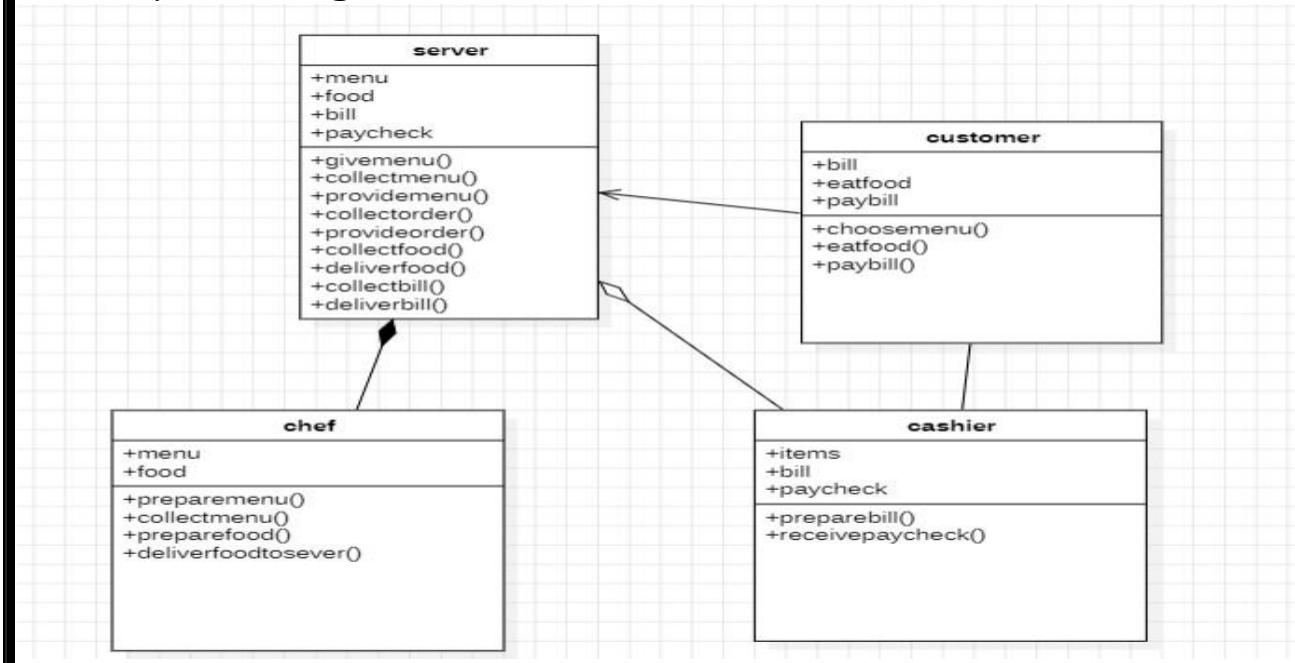


2. RESTAURANT MANAGEMENT

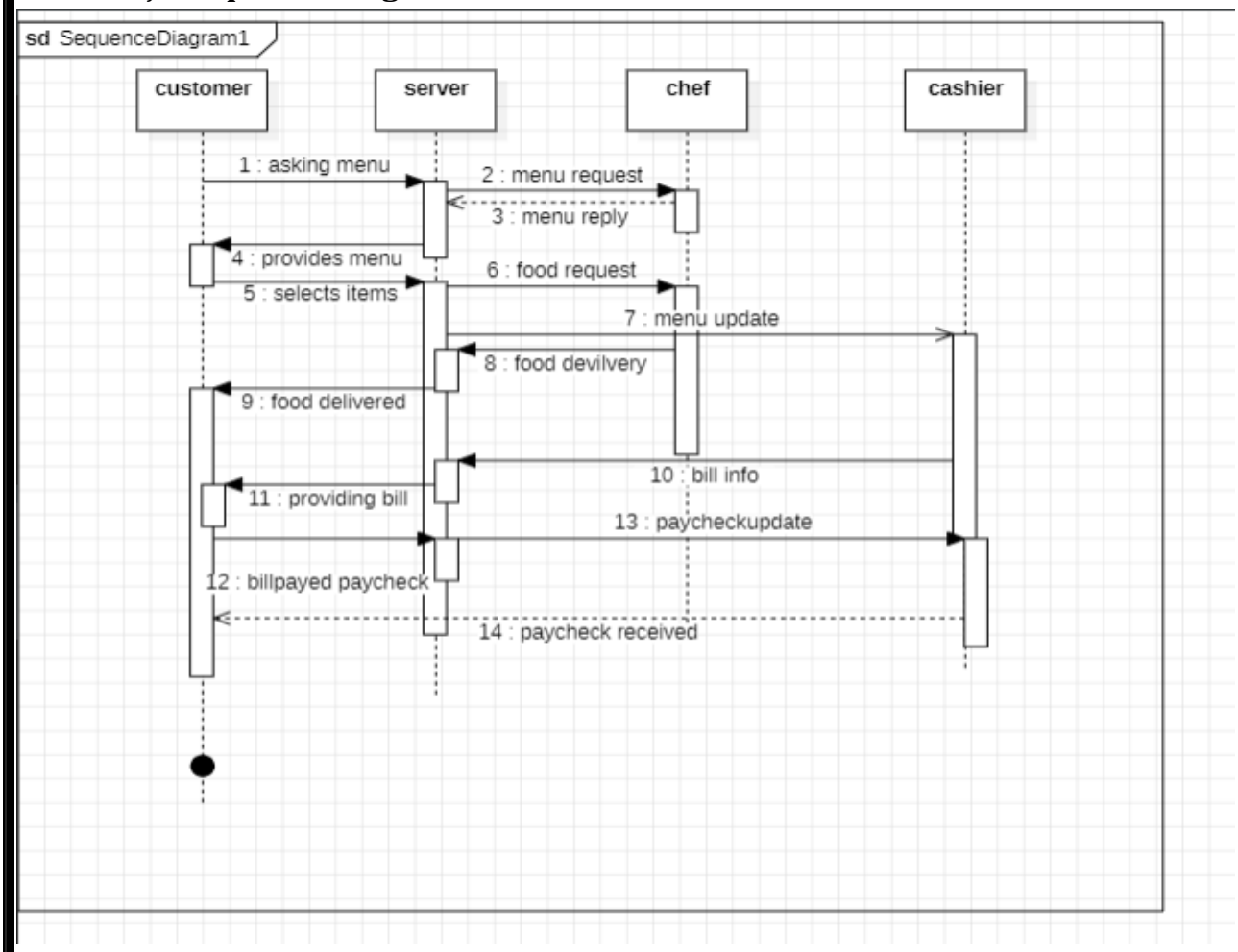
2.a) Use Case Diagram:



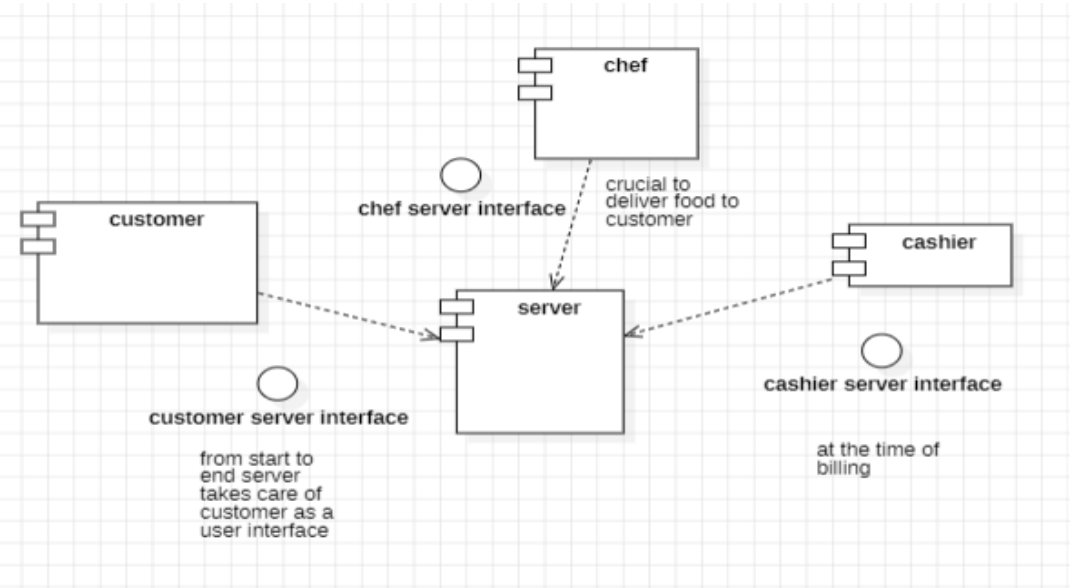
2.b) Class Diagram:



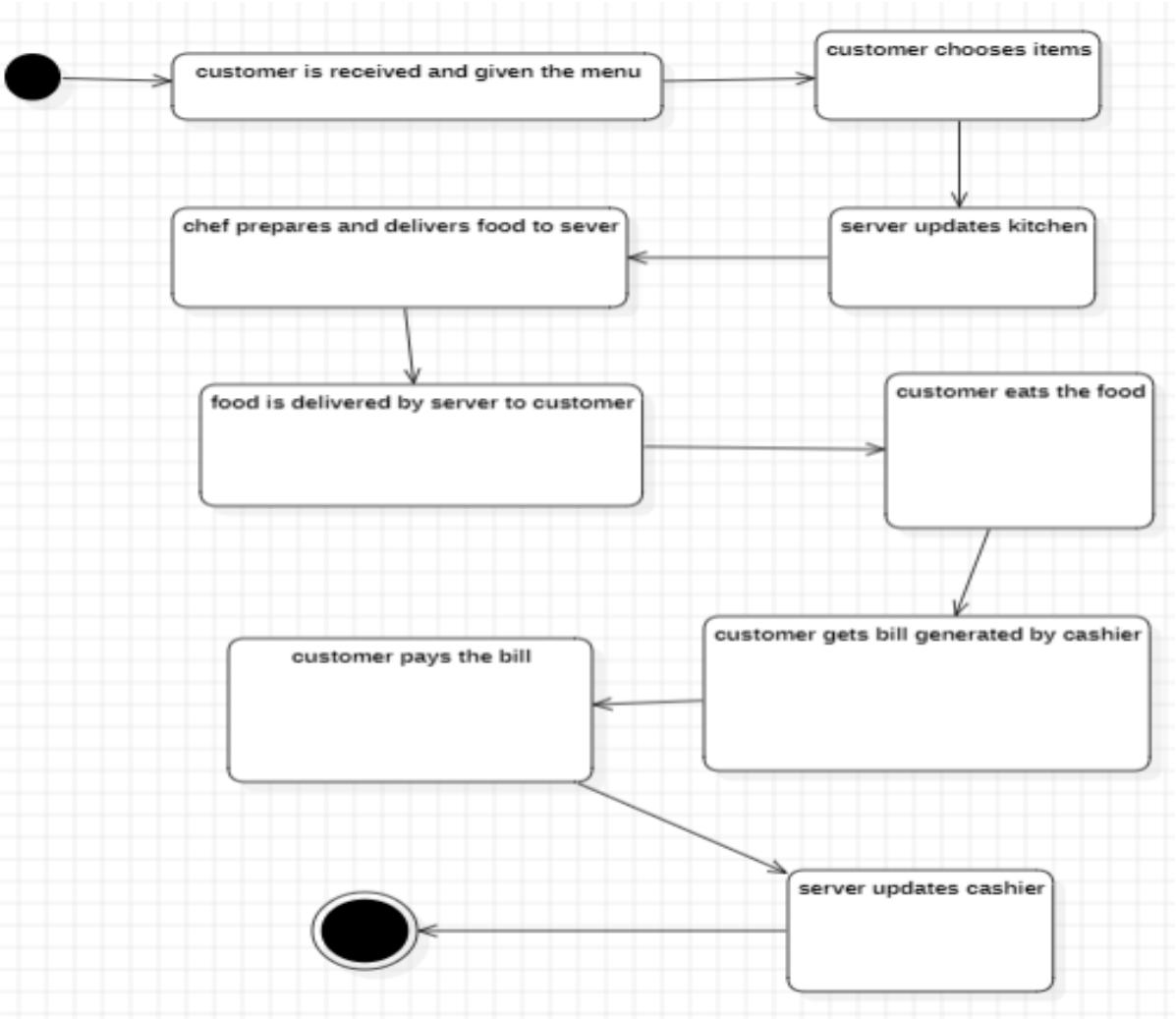
2.c) Sequence Diagram:



2.d) Component Diagram:



2.e) Activity Diagram:



3. Basic Java Programs

3.a) Armstrong Number:

Code:

```
public class ArmstrongNumber {
    public static void main(String[] args) {
        int num = 153; int original = num; int sum = 0;
        while (num != 0) {
            int digit = num % 10;
            sum += digit * digit * digit;
            num /= 10;
        }
        if (sum == original) {
            System.out.println(original + " is an Armstrong
number.");
        } else {
            System.out.println(original + " is not an Armstrong
number.");
        }
    }
```

Output:

```
PS D:\OOP\Exp 3 Basic Java Programs> javac ArmstrongNumber.java
PS D:\OOP\Exp 3 Basic Java Programs> java ArmstrongNumber.java
153 is an Armstrong number.
PS D:\OOP\Exp 3 Basic Java Programs>
```

3.b) Sum of Even, Odd Digits:

Code:

```
public class EvenOddSum {
    public static void main(String[] args) {
        int evenSum = 0; int oddSum = 0; int limit = 10;
        for (int i = 1; i <= limit; i++) {
            if (i % 2 == 0) {
                evenSum += i;
            } else {
                oddSum += i;
            }
        }
        System.out.println("Sum of even numbers from 1 to " + limit
+ ": " + evenSum);
        System.out.println("Sum of odd numbers from 1 to " + limit +
": " + oddSum);
    }
}
```

Output:

```
PS D:\OOP\Exp 3 Basic Java Programs> javac EvenOddSum.java
PS D:\OOP\Exp 3 Basic Java Programs> java EvenOddSum.java
Sum of even numbers from 1 to 10: 30
Sum of odd numbers from 1 to 10: 25
PS D:\OOP\Exp 3 Basic Java Programs>
```

3.c) Factorial:

Code:

```
public class Factorial {
    public static void main(String[] args) {
        int num = 5;
        int factorial = 1;
        for (int i = 1; i <= num; i++) {
            factorial *= i;
        }
        System.out.println("Factorial of " + num + " is " +
factorial);
    }
}
```

Output:

```
PS D:\OOP\Exp 3 Basic Java Programs> javac Factorial.java
PS D:\OOP\Exp 3 Basic Java Programs> java Factorial.java
Factorial of 5 is 120
PS D:\OOP\Exp 3 Basic Java Programs>
```

3.d) Fibonacci Series:

Code:

```
public class FibonacciSeries {
    public static void main(String[] args) {
        int n = 10, first = 0, second = 1;
        System.out.print("Fibonacci Series: " + first + ", " +
second);
        for (int i = 2; i < n; i++) {
            int next = first + second;
            System.out.print(", " + next);
            first = second;
            second = next;
        }
    }
}
```

Output;

```
PS D:\OOP\Exp 3 Basic Java Programs> javac FibonacciSeries.java
PS D:\OOP\Exp 3 Basic Java Programs> java FibonacciSeries.java
Fibonacci Series: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34
PS D:\OOP\Exp 3 Basic Java Programs>
```

3.e) LCM Calculator:

Code:

```
public class LCMCalculator {
    public static void main(String[] args) {
        int a = 12; int b = 18;int lcm;
        int gcd = a;
        int tempB = b;
        while (tempB != 0) {
            int temp = tempB;
            tempB = gcd % tempB;
            gcd = temp;
        }
        lcm = (a * b) / gcd;
        System.out.println("LCM is " + lcm);
    }
}
```

Output:

```
PS D:\OOP\Exp 3 Basic Java Programs> javac LCMCalculator.java
PS D:\OOP\Exp 3 Basic Java Programs> java LCMCalculator.java
LCM is 36
PS D:\OOP\Exp 3 Basic Java Programs>
```

3.f) Number Pattern:

Code:

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

Output:

```
PS D:\OOP\Exp 3 Basic Java Programs> javac NumberPattern.java
PS D:\OOP\Exp 3 Basic Java Programs> java NumberPattern.java
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
PS D:\OOP\Exp 3 Basic Java Programs>
```

3.g) Palindrome Check:

Code:

```
public class PalindromeCheck {
    public static void main(String[] args) {
        int num = 121; int original = num; int reversed = 0;
        while (num != 0) {
            int digit = num % 10;
            reversed = reversed * 10 + digit;
            num /= 10;
        }
        if (original == reversed) {
            System.out.println(original + " is a palindrome.");
        } else {
            System.out.println(original + " is not a palindrome.");
        }
    }
}
```

Output:

```
PS D:\OOP\Exp 3 Basic Java Programs> javac PalindromeCheck.java
PS D:\OOP\Exp 3 Basic Java Programs> java PalindromeCheck.java
121 is a palindrome.
PS D:\OOP\Exp 3 Basic Java Programs>
```

3.h) 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.i) Reverse Number:

Code:

```
public class ReverseNumber {
    public static void main(String[] args) {
        int num = 12345, reversed = 0;
        while (num != 0) {
            int digit = num % 10;
            reversed = reversed * 10 + digit;
            num /= 10;
        }
        System.out.println("Reversed Number: " + reversed);
    }
}
```

Output:

```
PS D:\OOP\Exp 3 Basic Java Programs> javac ReverseNumber.java
PS D:\OOP\Exp 3 Basic Java Programs> java ReverseNumber.java
Reversed Number: 54321
```

3.j) Sum of Digits:

Code:

```
public class SumOfDigits {
    public static void main(String[] args) {
        int num = 9876; int sum = 0;
        while (num != 0) {
            sum += num % 10;
            num /= 10;
        }
        System.out.println("Sum of digits: " + sum);
    }
}
```

Out

```
PS D:\OOP\Exp 3 Basic Java Programs> javac SumOfDigits.java
PS D:\OOP\Exp 3 Basic Java Programs> java SumOfDigits.java
Sum of digits: 30
PS D:\OOP\Exp 3 Basic Java Programs>
```

4) INHERITANCE

4.1) SINGLE INHERITANCE

A: EmpManagerSI1

CODE:

```
class Employee {
   String name;
   double salary;

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

   void display() {
      System.out.println("Name: " + name + ", Salary: $" + salary);
   }
}

// Derived class (inherits from Employee)
class Manager extends Employee {
   String department;

   Manager(String name, double salary, String department) {
      super(name, salary); // Call parent constructor
      this.department = department;
   }

   void displayManager() {
      display(); // Call parent method
      System.out.println("Department: " + department);
   }
}

public class EmpManagerSI1 {
   public static void main(String[] args) {
      Manager manager = new Manager("John Doe", 60000, "IT");
      manager.displayManager();
   }
      }
```

OUTPUT:

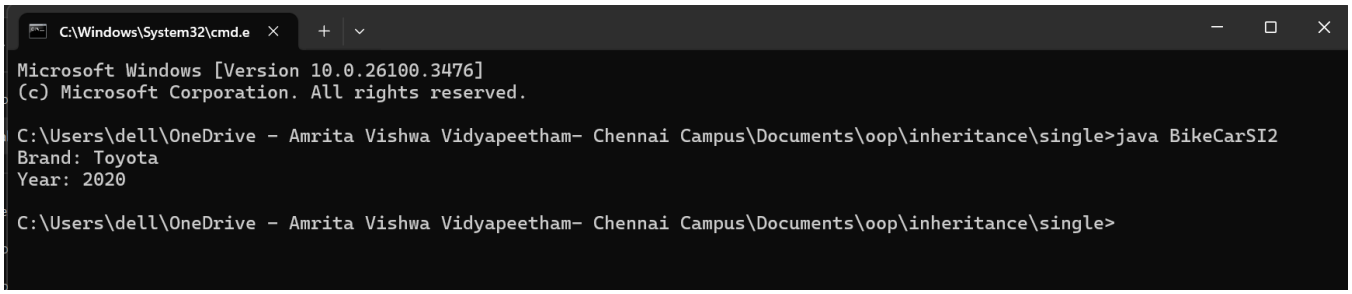
```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\inheritance\single>java EmpManagerSI1
Name: John Doe, Salary: $60000.0
Department: IT

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\inheritance\single>
```

B): BikeCarSI2

CODE):

```
// Base class
class Vehicle {
    String brand;

    Vehicle(String brand) {
        this.brand = brand;
    }

    void showDetails() {
        System.out.println("Brand: " + brand);
    }
}

// Derived class
class Car extends Vehicle {
    int year;

    Car(String brand, int year) {
        super(brand); // Call parent constructor
        this.year = year;
    }

    void displayCar() {
        showDetails(); // Call parent method
        System.out.println("Year: " + year);
    }
}

public class BikeCarSI2 {
    public static void main(String[] args) {
        Car car = new Car("Toyota", 2020);
        car.displayCar();
    }
}
```

OUTPUT):



5): MULTILEVEL INHERITANCE PROGRAMS

5.a) AnimalMAMHumanMLI1

CODE:

```
// Base class
class Animal {
    String type;

    Animal(String type) {
        this.type = type;
    }

    void eat() {
        System.out.println(type + " is eating.");
    }
}

// Intermediate class
class Mammal extends Animal {
    boolean hasFur;

    Mammal(String type, boolean hasFur) {
        super(type);
        this.hasFur = hasFur;
    }

    void breathe() {
        System.out.println(type + " is breathing.");
    }
}

// Derived class
class Human extends Mammal {
    String name;

    Human(String type, boolean hasFur, String name) {
        super(type, hasFur);
```

```
        this.name = name;
    }

    void speak() {
        System.out.println(name + " is speaking.");
    }
}

public class AnimalMAMHumanMLI1 {
    public static void main(String[] args) {
        Human human = new Human("Mammal", true, "Alice");
        human.eat();
        human.breathe();
        human.speak();
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\inheritance\multilevel>java AnimalMAMH
umanMLI1
Mammal is eating.
Mammal is breathing.
Alice is speaking.

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\inheritance\multilevel>
```

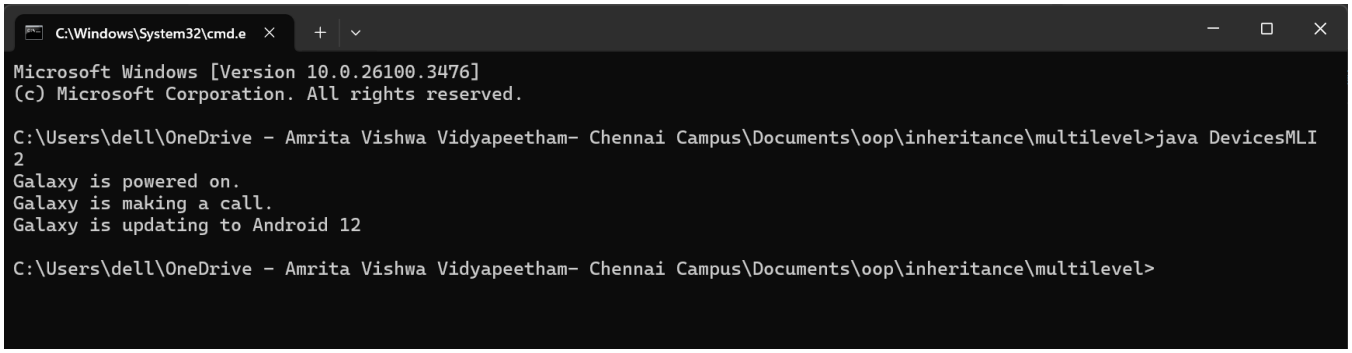
5.b) DevicesMLI2

CODE:

```
// Base class
class Device {
    String model;

    Device(String model) {
        this.model = model;
    }

    void powerOn() {
        System.out.println(model + " is powered on.");
    }
}

// Intermediate class
class SmartPhone extends Device {
    boolean hasCamera;
```

```
    SmartPhone(String model, boolean hasCamera) {
        super(model);
        this.hasCamera = hasCamera;
    }

    void call() {
        System.out.println(model + " is making a call.");
    }
}

// Derived class
class AndroidPhone extends SmartPhone {
    String version;

    AndroidPhone(String model, boolean hasCamera, String version) {
        super(model, hasCamera);
        this.version = version;
    }

    void update() {
        System.out.println(model + " is updating to Android " + version);
    }
}

public class DevicesMLI2 {
    public static void main(String[] args) {
        AndroidPhone phone = new AndroidPhone("Galaxy", true, "12");
        phone.powerOn();
        phone.call();
        phone.update();
    }
}
```

OUTPUT:



6.) HIERARCHICAL INHERITANCE PROGRAMS

6.a) ShapesHI1

CODE:

```
// Base class
class Shape {
    String color;

    Shape(String color) {
        this.color = color;
    }

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

// Derived class 1
class Circle extends Shape {
    double radius;

    Circle(String color, double radius) {
        super(color);
        this.radius = radius;
    }

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

// Derived class 2
class Rectangle extends Shape {
    double length, width;

    Rectangle(String color, double length, double width) {
        super(color);
        this.length = length;
        this.width = width;
    }

    void calculateArea() {
        double area = length * width;
        System.out.println("Rectangle Area: " + area);
    }
}

public class ShapesHI1 {
```

```
    public static void main(String[] args) {
        Circle circle = new Circle("Red", 5);
        Rectangle rectangle = new Rectangle("Blue", 4, 6);

        circle.displayColor();
        circle.calculateArea();

        rectangle.displayColor();
        rectangle.calculateArea();
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\inheritance\heirarchical>java ShapesHI
1
Color: Red
Circle Area: 78.53981633974483
Color: Blue
Rectangle Area: 24.0

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\inheritance\heirarchical>
```

6.b) EmpDevTestHI2

CODE:

```
// Base class
class Employee {
    String name;

    Employee(String name) {
        this.name = name;
    }

    void work() {
        System.out.println(name + " is working.");
    }
}

// Derived class 1
class Developer extends Employee {
    String language;

    Developer(String name, String language) {
        super(name);
        this.language = language;
```

```
    }

    void code() {
        System.out.println(name + " is coding in " + language);
    }
}

// Derived class 2
class Tester extends Employee {
    String tool;

    Tester(String name, String tool) {
        super(name);
        this.tool = tool;
    }

    void test() {
        System.out.println(name + " is testing with " + tool);
    }
}

public class EmpDevTestHI2 {
    public static void main(String[] args) {
        Developer dev = new Developer("Alice", "Java");
        Tester tester = new Tester("Bob", "Selenium");

        dev.work();
        dev.code();

        tester.work();
        tester.test();
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\inheritance\heirarchical>java EmpDevTe
stHI2
Alice is working.
Alice is coding in Java
Bob is working.
Bob is testing with Selenium

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\inheritance\heirarchical>
```

7. HYBRID INHERITANCE PROGRAMS

7.a) TeacherAteacherHYI1

CODE:

```
// Base class
class School {
    String name;

    School(String name) {
        this.name = name;
    }

    void showInfo() {
        System.out.println("School Name: " + name);
    }
}

// Derived class 1
class Teacher extends School {
    String subject;

    Teacher(String name, String subject) {
        super(name);
        this.subject = subject;
    }

    void teach() {
        System.out.println("Teaching " + subject);
    }
}

// Derived class 2
class Student extends School {
    int grade;

    Student(String name, int grade) {
        super(name);
        this.grade = grade;
    }

    void study() {
        System.out.println("Studying in grade " + grade);
    }
}

// Derived class (hybrid: inherits from both Teacher and Student)
class AssistantTeacher extends Teacher {
    int experience;

    AssistantTeacher(String schoolName, String subject, int experience) {
        super(schoolName, subject);
        this.experience = experience;
    }

    void assist() {
        System.out.println("Assisting with " + subject + " for " + experience + "
```

```
years");
    }
}

public class TeacherAteacherHYI1 {
    public static void main(String[] args) {
        AssistantTeacher assistant = new AssistantTeacher("XYZ School", "Math", 2);
        assistant.showInfo();
        assistant.teach();
        assistant.assist();
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\inheritance\hybrid>java TeacherAteache
rHYI1
School Name: XYZ School
Teaching Math
Assisting with Math for 2 years

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\inheritance\hybrid>
```

7.b) TransportHYI2

CODE:

```
// Base class
class Transport {
    String mode;

    Transport(String mode) {
        this.mode = mode;
    }

    void move() {
        System.out.println(mode + " is moving.");
    }
}

// Derived class 1
class LandTransport extends Transport {
    int wheels;

    LandTransport(String mode, int wheels) {
        super(mode);
```

```
        this.wheels = wheels;
    }

    void drive() {
        System.out.println(mode + " is driven on " + wheels + " wheels.");
    }
}

// Derived class 2
class AirTransport extends Transport {
    boolean hasWings;

    AirTransport(String mode, boolean hasWings) {
        super(mode);
        this.hasWings = hasWings;
    }

    void fly() {
        System.out.println(mode + " is flying.");
    }
}

// Derived class (hybrid: inherits from AirTransport)
class Helicopter extends AirTransport {
    int rotors;

    Helicopter(String mode, boolean hasWings, int rotors) {
        super(mode, hasWings);
        this.rotors = rotors;
    }

    void hover() {
        System.out.println(mode + " is hovering with " + rotors + " rotors.");
    }
}

public class TransportHYI2 {
    public static void main(String[] args) {
        Helicopter heli = new Helicopter("Helicopter", false, 2);
        heli.move();
        heli.fly();
        heli.hover();
    }
}
```

OUTPUT:

```
C:\Windows\System32\cmd.e

Microsoft Windows [Version 10.0.26100.3476]
(c) Microsoft Corporation. All rights reserved.

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\inheritance\hybrid>java TransportHYI2
Helicopter is moving.
Helicopter is flying.
Helicopter is hovering with 2 rotors.

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\inheritance\hybrid>
```

POLYMORPHISM

8. CONSTRUCTOR PROGRAMS

8.a) DroneSystem

CODE:

```
// Base class for all drones
class Drone {
    String model;
    int batteryLife;

    // Constructor for base drone
    Drone(String model, int batteryLife) {
        this.model = model;
        this.batteryLife = batteryLife;
    }

    // Polymorphic method
    void performTask() {
        System.out.println(model + " is flying...");
    }
}

// Surveillance Drone
class SurveillanceDrone extends Drone {
    boolean nightVision;

    SurveillanceDrone(String model, int batteryLife, boolean nightVision) {
        super(model, batteryLife);
        this.nightVision = nightVision;
    }
```

```
    @Override
    void performTask() {
        System.out.println(model + " is performing surveillance. Night Vision: " +
(nightVision ? "Enabled" : "Disabled"));
    }
}

// Delivery Drone
class DeliveryDrone extends Drone {
    double maxPayload;

    DeliveryDrone(String model, int batteryLife, double maxPayload) {
        super(model, batteryLife);
        this.maxPayload = maxPayload;
    }

    @Override
    void performTask() {
        System.out.println(model + " is delivering a package. Max Payload: " +
maxPayload + "kg");
    }
}

// Agricultural Drone
class AgriculturalDrone extends Drone {
    int waterCapacity;

    AgriculturalDrone(String model, int batteryLife, int waterCapacity) {
        super(model, batteryLife);
        this.waterCapacity = waterCapacity;
    }

    @Override
    void performTask() {
        System.out.println(model + " is spraying fields. Water Capacity: " +
waterCapacity + " liters");
    }
}

// Main class to test drones
public class DroneSystem {
    public static void main(String[] args) {
        Drone d1 = new SurveillanceDrone("EagleEye-X1", 60, true);
        Drone d2 = new DeliveryDrone("SkyDrop-3000", 45, 5.5);
        Drone d3 = new AgriculturalDrone("AgriBot-A5", 80, 20);

        d1.performTask();
        d2.performTask();
        d3.performTask();
    }
}
```

OUTPUT:

```
C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.26100.3476]
(c) Microsoft Corporation. All rights reserved.

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\polymorphism\CONSTRUCTOR>java DroneSys
tem
EagleEye-X1 is performing surveillance. Night Vision: Enabled
SkyDrop-3000 is delivering a package. Max Payload: 5.5kg
AgriBot-A5 is spraying fields. Water Capacity: 20 liters

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\polymorphism\CONSTRUCTOR>
```

9. CONSTRUCTOR OVERLOADING PROGRAMS

9.a) OnlineExamSystem

CODE:

```
// Base class for Exam
class Exam {
    String subject;
    int duration; // in minutes

    // Default constructor
    Exam(String subject, int duration) {
        this.subject = subject;
        this.duration = duration;
    }

    void startExam() {
        System.out.println("Starting " + subject + " exam. Duration: " + duration +
" minutes.");
    }
}

// Objective Exam (MCQs)
class ObjectiveExam extends Exam {
    int numQuestions;

    ObjectiveExam(String subject, int duration, int numQuestions) {
        super(subject, duration);
        this.numQuestions = numQuestions;
    }

    @Override
    void startExam() {
        System.out.println("Starting Objective Exam: " + subject + ". Duration: " +
duration + " minutes. Number of Questions: " + numQuestions);
```

```
    }
}

// Subjective Exam (Written answers)
class SubjectiveExam extends Exam {
    boolean essayRequired;

    SubjectiveExam(String subject, int duration, boolean essayRequired) {
        super(subject, duration);
        this.essayRequired = essayRequired;
    }

    @Override
    void startExam() {
        System.out.println("Starting Subjective Exam: " + subject + ". Duration: "
+ duration + " minutes. Essay Required: " + (essayRequired ? "Yes" : "No"));
    }
}

// Coding Exam
class CodingExam extends Exam {
    String programmingLanguage;
    int numProblems;

    CodingExam(String subject, int duration, String programmingLanguage, int
numProblems) {
        super(subject, duration);
        this.programmingLanguage = programmingLanguage;
        this.numProblems = numProblems;
    }

    @Override
    void startExam() {
        System.out.println("Starting Coding Exam in " + programmingLanguage + ".
Subject: " + subject + ". Duration: " + duration + " minutes. Number of Problems: "
+ numProblems);
    }
}

// Main class to test the exams
public class OnlineExamSystem {
    public static void main(String[] args) {
        Exam e1 = new ObjectiveExam("Math", 60, 40);
        Exam e2 = new SubjectiveExam("History", 90, true);
        Exam e3 = new CodingExam("Programming", 120, "Java", 5);

        e1.startExam();
        e2.startExam();
        e3.startExam();
    }
}
```

OUTPUT:

```
C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.26100.3476]
(c) Microsoft Corporation. All rights reserved.

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\polymorphism\constructoroverloading>ja
va OnlineExamSystem
Starting Objective Exam: Math. Duration: 60 minutes. Number of Questions: 40
Starting Subjective Exam: History. Duration: 90 minutes. Essay Required: Yes
Starting Coding Exam in Java. Subject: Programming. Duration: 120 minutes. Number of Problems: 5

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\polymorphism\constructoroverloading>
```

10. METHOD OVERLOADING PROGRAMS

10.a) AreaCalculator

CODE:

```
public class AreaCalculator {

    // Calculate area of a square
    public double calculateArea(double side) {
        return side * side;
    }

    // Calculate area of a rectangle
    public double calculateArea(double length, double width) {
        return length * width;
    }

    // Calculate area of a circle
    public double calculateArea(float radius) {
        return Math.PI * radius * radius;
    }

    // Calculate area of a triangle
    public double calculateArea(double base, double height, String shape) {
        if (shape.equalsIgnoreCase("triangle")) {
            return 0.5 * base * height;
        }
        return 0; // For other shapes (extend as needed)
    }

    // Calculate area of a trapezoid
    public double calculateArea(double base1, double base2, double height) {
        return 0.5 * (base1 + base2) * height;
    }

    public static void main(String[] args) {
        AreaCalculator calculator = new AreaCalculator();
```

```
        // Testing all overloaded methods
        System.out.printf("Area of square (side 5): %.2f\n",
calculator.calculateArea(5.0));
        System.out.printf("Area of rectangle (6x4): %.2f\n",
calculator.calculateArea(6.0, 4.0));
        System.out.printf("Area of circle (radius 3): %.2f\n",
calculator.calculateArea(3.0f));
        System.out.printf("Area of triangle (base 8, height 5): %.2f\n",
                          calculator.calculateArea(8.0, 5.0, "triangle"));
        System.out.printf("Area of trapezoid (bases 4 & 6, height 5): %.2f\n",
                          calculator.calculateArea(4.0, 6.0, 5.0));
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\polymorphism\methodoverloading>javac A
reaCalculator.java

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\polymorphism\methodoverloading>java Ar
eaCalculator.java
Area of square (side 5): 25.00
Area of rectangle (6x4): 24.00
Area of circle (radius 3): 28.27
Area of triangle (base 8, height 5): 20.00
Area of trapezoid (bases 4 & 6, height 5): 25.00

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\polymorphism\methodoverloading>
```

10.b) StringManipulator

CODE:

```
public class StringManipulator {

    // Method to reverse a string
    public String manipulate(String str) {
        return new StringBuilder(str).reverse().toString();
    }

    // Method to concatenate two strings
    public String manipulate(String str1, String str2) {
        return str1.concat(str2);
    }

    // Method to repeat a string n times
    public String manipulate(String str, int times) {
        return str.repeat(times);
    }

    // Method to convert string to uppercase or lowercase
    public String manipulate(String str, boolean toUpper) {
        return toUpper ? str.toUpperCase() : str.toLowerCase();
    }
```

```
    // Method to replace all occurrences of a character
    public String manipulate(String str, char oldChar, char newChar) {
        return str.replace(oldChar, newChar);
    }

    // Method to extract substring between two indices
    public String manipulate(String str, int start, int end) {
        return str.substring(start, end);
    }

    public static void main(String[] args) {
        StringManipulator manipulator = new StringManipulator();

        // Testing all overloaded methods
        System.out.println("Reversed string: " + manipulator.manipulate("Hello"));
        System.out.println("Concatenated strings: " +
manipulator.manipulate("Hello", "World"));
        System.out.println("Repeated string: " + manipulator.manipulate("Hi", 3));
        System.out.println("Uppercase: " + manipulator.manipulate("hello", true));
        System.out.println("Lowercase: " + manipulator.manipulate("WORLD", false));
        System.out.println("Character replaced: " +
manipulator.manipulate("banana", 'a', 'o'));
        System.out.println("Substring: " + manipulator.manipulate("Programming", 3,
7));
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\polymorphism\methodoverloading>javac S
tringManipulator.java

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\polymorphism\methodoverloading>java St
ringManipulator.java
Reversed string: olleH
Concatenated strings: HelloWorld
Repeated string: HiHiHi
Uppercase: HELLO
Lowercase: world
Character replaced: bonono
Substring: gram

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\polymorphism\methodoverloading>
```

11. METHOD OVERRIDING PROGRAMS

11.a) InterestCal

CODE:

```
// Base class
class BankAccount {
    double calculateInterest(double amount) {
        return amount * 0.02; // 2% interest
    }

    void displayInterest(double amount) {
        System.out.println("Interest: $" + calculateInterest(amount));
    }
}

// Derived class
class SavingsAccount extends BankAccount {
    @Override
    double calculateInterest(double amount) {
        return amount * 0.05; // 5% interest for savings
    }
}

public class InterestCal {
    public static void main(String[] args) {
        BankAccount regular = new BankAccount();
        SavingsAccount savings = new SavingsAccount();

        System.out.println("Regular Account:");
        regular.displayInterest(1000);

        System.out.println("Savings Account:");
        savings.displayInterest(1000);
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\polymorphism\methodoverriding>java Int
erestCal.java
Regular Account:
Interest: $20.0
Savings Account:
Interest: $50.0

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\polymorphism\methodoverriding>
```

11.b) VehicleSound

CODE:

```
// Base class
class Vehicle {
    void makeSound() {
        System.out.println("Generic vehicle sound");
    }
}

// Derived class
class Car extends Vehicle {
    @Override
    void makeSound() {
        System.out.println("Vroom vroom! (Car sound)");
    }
}

// Derived class
class Motorcycle extends Vehicle {
    @Override
    void makeSound() {
        System.out.println("Brum brum! (Motorcycle sound)");
    }
}

public class VehicleSound {
    public static void main(String[] args) {
        Vehicle generic = new Vehicle();
        Car car = new Car();
        Motorcycle bike = new Motorcycle();

        generic.makeSound();
        car.makeSound();
        bike.makeSound();
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\polymorphism\methodoverriding>javac Ve
hicleSound.java

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\polymorphism\methodoverriding>java Veh
icleSound.java
Generic vehicle sound
Vroom vroom! (Car sound)
Brum brum! (Motorcycle sound)

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\polymorphism\methodoverriding>
```

ABSTRACTION

12. INTERFACE PROGRAMS

12.a) HomeSecuritySystem

CODE:

```
import java.util.Scanner;

// Interface SecurityDevice
interface SecurityDevice {
    void activate();
    void deactivate();
}

// CCTV Camera class implementing SecurityDevice
class CCTV implements SecurityDevice {
    @Override
    public void activate() {
        System.out.println("CCTV Camera is now recording.");
    }

    @Override
    public void deactivate() {
        System.out.println("CCTV Camera is turned off.");
    }
}

// Smart Lock class implementing SecurityDevice
class SmartLock implements SecurityDevice {
    @Override
    public void activate() {
        System.out.println("Smart Lock is now locked.");
    }

    @Override
    public void deactivate() {
        System.out.println("Smart Lock is unlocked.");
    }
}

// Alarm System class implementing SecurityDevice
class AlarmSystem implements SecurityDevice {
    @Override
    public void activate() {
        System.out.println("Alarm System is activated. Intruders will be
detected!");
    }

    @Override
```

```
    public void deactivate() {
        System.out.println("Alarm System is deactivated.");
    }
}

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

        SecurityDevice[] devices = { new CCTV(), new SmartLock(), new AlarmSystem()
};

        System.out.println("Choose an action: \n1. Activate Security \n2.
Deactivate Security");
        int choice = scanner.nextInt();

        switch (choice) {
            case 1:
                for (SecurityDevice device : devices) {
                    device.activate();
                }
                break;
            case 2:
                for (SecurityDevice device : devices) {
                    device.deactivate();
                }
                break;
            default:
                System.out.println("Invalid option.");
        }

        scanner.close();
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\interface>java HomeSecuritySystem.java
Choose an action:
1. Activate Security
2. Deactivate Security
1.
Exception in thread "main" java.util.InputMismatchException
        at java.base/java.util.Scanner.throwFor(Scanner.java:964)
        at java.base/java.util.Scanner.next(Scanner.java:1619)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2284)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2238)
        at HomeSecuritySystem.main(HomeSecuritySystem.java:55)

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\interface>java HomeSecuritySystem.java
Choose an action:
1. Activate Security
2. Deactivate Security
2
CCTV Camera is turned off.
Smart Lock is unlocked.
Alarm System is deactivated.

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\interface>
```

12.b) MusicInterface

CODE:

```
interface MusicPlayer {
    void play();
    void pause();
    String getFormat();
}

class MP3Player implements MusicPlayer {
    private String fileName;

    public MP3Player(String fileName) {
        this.fileName = fileName;
    }

    @Override
    public void play() {
        System.out.println("Playing MP3: " + fileName);
    }

    @Override
    public void pause() {
        System.out.println("Pausing MP3: " + fileName);
    }

    @Override
    public String getFormat() {
        return "MP3";
    }
}

class WAVPlayer implements MusicPlayer {
    private String fileName;

    public WAVPlayer(String fileName) {
        this.fileName = fileName;
    }

    @Override
    public void play() {
        System.out.println("Playing WAV: " + fileName);
    }

    @Override
    public void pause() {
        System.out.println("Pausing WAV: " + fileName);
    }

    @Override
    public String getFormat() {
        return "WAV";
    }
```

```
}

public class MusicInterface{
    public static void main(String[] args) {
        MusicPlayer mp3 = new MP3Player("song.mp3");
        MusicPlayer wav = new WAVPlayer("audio.wav");

        mp3.play();
        mp3.pause();
        System.out.println("Format: " + mp3.getFormat());

        wav.play();
        wav.pause();
        System.out.println("Format: " + wav.getFormat());
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\interface>java MusicInterface.java
Playing MP3: song.mp3
Pausing MP3: song.mp3
Format: MP3
Playing WAV: audio.wav
Pausing WAV: audio.wav
Format: WAV

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\interface>
```

12.c) OnlineLibrary

CODE:

```
import java.util.ArrayList;
import java.util.List;

// Interface LibraryItem
interface LibraryItem {
    void borrowItem();
    void returnItem();
}

// Class implementing LibraryItem for Books
class Book implements LibraryItem {
    private String title;
    private boolean isBorrowed;

    public Book(String title) {
        this.title = title;
        this.isBorrowed = false;
    }

    @Override
    public void borrowItem() {
```

```
        if (!isBorrowed) {
            isBorrowed = true;
            System.out.println(title + " has been borrowed.");
        } else {
            System.out.println(title + " is already borrowed.");
        }
    }

    @Override
    public void returnItem() {
        if (isBorrowed) {
            isBorrowed = false;
            System.out.println(title + " has been returned.");
        } else {
            System.out.println(title + " was not borrowed.");
        }
    }
}

// Class implementing LibraryItem for Magazines
class Magazine implements LibraryItem {
    private String name;

    public Magazine(String name) {
        this.name = name;
    }

    @Override
    public void borrowItem() {
        System.out.println(name + " magazine cannot be borrowed!");
    }

    @Override
    public void returnItem() {
        System.out.println(name + " magazine does not require return.");
    }
}

public class OnlineLibrary {
    public static void main(String[] args) {
        List<LibraryItem> library = new ArrayList<>();
        library.add(new Book("The Java Handbook"));
        library.add(new Magazine("Tech World"));

        // Borrowing items
        for (LibraryItem item : library) {
            item.borrowItem();
        }

        // Returning items
        for (LibraryItem item : library) {
            item.returnItem();
        }
```

```
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\interface>java OnlineLibrary.java
The Java Handbook has been borrowed.
Tech World magazine cannot be borrowed!
The Java Handbook has been returned.
Tech World magazine does not require return.

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\interface>
```

12.d) PaymentInterface

CODE:

```
interface Payment {
    void processPayment(double amount);
    String getPaymentMethod();
}

class CreditCardPayment implements Payment {
    private String cardNumber;

    public CreditCardPayment(String cardNumber) {
        this.cardNumber = cardNumber;
    }

    @Override
    public void processPayment(double amount) {
        System.out.println("Processing $" + amount + " via Credit Card ending in "
+ cardNumber.substring(cardNumber.length() - 4));
    }

    @Override
    public String getPaymentMethod() {
        return "Credit Card";
    }
}

class PayPalPayment implements Payment {
    private String email;

    public PayPalPayment(String email) {
        this.email = email;
    }

    @Override
    public void processPayment(double amount) {
        System.out.println("Processing $" + amount + " via PayPal for " + email);
    }
```

```
    @Override
    public String getPaymentMethod() {
        return "PayPal";
    }
}

public class PaymentInterface {
    public static void main(String[] args) {
        Payment credit = new CreditCardPayment("1234567890123456");
        Payment paypal = new PayPalPayment("user@example.com");

        credit.processPayment(100.0);
        System.out.println("Method: " + credit.getPaymentMethod());

        paypal.processPayment(50.0);
        System.out.println("Method: " + paypal.getPaymentMethod());
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\interface>javac PaymentInt
erface.java

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\interface>java PaymentInte
rface.java
Processing $100.0 via Credit Card ending in 3456
Method: Credit Card
Processing $50.0 via PayPal for user@example.com
Method: PayPal

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\interface>
```

13.ABSTRACT CLASS PROGRAMS

13.a) FoodOrderingSystem

CODE:

```
import java.util.Scanner;

// Abstract class FoodItem
abstract class FoodItem {
    protected String name;
    protected double basePrice;

    public FoodItem(String name, double basePrice) {
        this.name = name;
        this.basePrice = basePrice;
    }

    // Abstract method to calculate final price
    abstract double calculatePrice(int quantity);
```

```
    public void displayInfo() {
        System.out.println(name + " - ₹" + basePrice);
    }
}

// Concrete class Pizza
class Pizza extends FoodItem {
    private String size;

    public Pizza(String size) {
        super("Pizza", size.equals("Large") ? 500 : 300);
        this.size = size;
    }

    @Override
    double calculatePrice(int quantity) {
        return basePrice * quantity;
    }
}

// Concrete class Burger
class Burger extends FoodItem {
    private boolean isCheeseAdded;

    public Burger(boolean isCheeseAdded) {
        super("Burger", isCheeseAdded ? 200 : 150);
        this.isCheeseAdded = isCheeseAdded;
    }

    @Override
    double calculatePrice(int quantity) {
        return basePrice * quantity;
    }
}

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

        // Ordering Pizza
        System.out.print("Choose Pizza Size (Small/Large): ");
        String size = scanner.next();
        System.out.print("Enter quantity: ");
        int pizzaQty = scanner.nextInt();
        FoodItem pizza = new Pizza(size);
        System.out.println("Total Pizza Price: ₹" +
pizza.calculatePrice(pizzaQty));

        // Ordering Burger
        System.out.print("Do you want extra cheese in Burger? (yes/no): ");
        boolean cheese = scanner.next().equalsIgnoreCase("yes");
        System.out.print("Enter quantity: ");
```

```
        int burgerQty = scanner.nextInt();
        FoodItem burger = new Burger(cheese);
        System.out.println("Total Burger Price: ₹" +
burger.calculatePrice(burgerQty));

        scanner.close();
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\abstractclass>java FoodOrd
eringSystem.java
Choose Pizza Size (Small/Large): Large
Enter quantity: 2
Total Pizza Price: ?1000.0
Do you want extra cheese in Burger? (yes/no): yes
Enter quantity: 4
Total Burger Price: ?800.0

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\abstractclass>
```

13.b) OnlinePayment

CODE:

```
import java.util.Scanner;

// Abstract class Payment
abstract class Payment {
    protected double amount;

    // Constructor
    public Payment(double amount) {
        this.amount = amount;
    }

    // Abstract method to process payment
    abstract void processPayment();
}

// Concrete class for Credit Card Payment
class CreditCard extends Payment {
    private String cardNumber;

    public CreditCard(double amount, String cardNumber) {
        super(amount);
        this.cardNumber = cardNumber;
    }

    @Override
    void processPayment() {
```

```
        System.out.println("Processing Credit Card payment of ₹" + amount);
        System.out.println("Payment Successful using Card: " + cardNumber);
    }
}

// Concrete class for UPI Payment
class UPI extends Payment {
    private String upiID;

    public UPI(double amount, String upiID) {
        super(amount);
        this.upiID = upiID;
    }

    @Override
    void processPayment() {
        System.out.println("Processing UPI payment of ₹" + amount);
        System.out.println("Payment Successful using UPI ID: " + upiID);
    }
}

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

        System.out.print("Enter payment amount: ");
        double amount = scanner.nextDouble();

        Payment payment = new CreditCard(amount, "1234-5678-9876-5432");
        payment.processPayment();

        payment = new UPI(amount, "user@upi");
        payment.processPayment();

        scanner.close();
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\abstractclass>java OnlinePayment.java
Enter payment amount: 9
Processing Credit Card payment of ?9.0
Payment Successful using Card: 1234-5678-9876-5432
Processing UPI payment of ?9.0
Payment Successful using UPI ID: user@upi
```

13.c) SmartHome

CODE:

```
// Abstract class SmartDevice
abstract class SmartDevice {
    protected String name;

    public SmartDevice(String name) {
        this.name = name;
    }

    // Abstract method to turn on the device
    abstract void turnOn();
}

// Concrete class SmartLight
class SmartLight extends SmartDevice {
    public SmartLight(String name) {
        super(name);
    }

    @Override
    void turnOn() {
        System.out.println(name + " is now ON with warm white light.");
    }
}

// Concrete class SmartFan
class SmartFan extends SmartDevice {
    public SmartFan(String name) {
        super(name);
    }

    @Override
    void turnOn() {
        System.out.println(name + " is now ON at medium speed.");
    }
}

public class SmartHome {
    public static void main(String[] args) {
        SmartDevice light = new SmartLight("Living Room Light");
        SmartDevice fan = new SmartFan("Ceiling Fan");

        light.turnOn();
        fan.turnOn();
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\abstractclass>java SmartHome.java
Living Room Light is now ON with warm white light.
Ceiling Fan is now ON at medium speed.

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\abstractclass>
```

13.d) Vehicle

CODE:

```
abstract class AbstractVehicle {
    abstract void start();
    abstract void stop();

    void displayInfo() {
        System.out.println("This is a vehicle.");
    }
}

class Car extends AbstractVehicle {
    @Override
    void start() {
        System.out.println("Car engine started.");
    }

    @Override
    void stop() {
        System.out.println("Car engine stopped.");
    }
}

class Motorcycle extends AbstractVehicle {
    @Override
    void start() {
        System.out.println("Motorcycle engine started.");
    }

    @Override
    void stop() {
        System.out.println("Motorcycle engine stopped.");
    }
}

public class Vehicle {
    public static void main(String[] args) {
        AbstractVehicle car = new Car();
        AbstractVehicle bike = new Motorcycle();

        car.displayInfo();
        car.start();
```

```
        car.stop();

        bike.displayInfo();
        bike.start();
        bike.stop();
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\abstractclass>javac Vehicle.java

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\abstractclass>java Vehicle.java
This is a vehicle.
Car engine started.
Car engine stopped.
This is a vehicle.
Motorcycle engine started.
Motorcycle engine stopped.

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\abstraction\abstractclass>
```

ENCAPSULATION

14.ENCAPSULATION PROGRAMS

14.a) BankEncap

CODE:

```
class BankAccount {
    private String accountNumber;
    private double balance;

    public BankAccount(String accountNumber, double initialBalance) {
        this.accountNumber = accountNumber;
        this.balance = initialBalance;
    }

    public String getAccountNumber() {
        return accountNumber;
    }

    public double getBalance() {
        return balance;
    }

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposited $" + amount + ". New balance: $" +
```

```
balance);
        } else {
            System.out.println("Invalid deposit amount!");
        }
    }

    public void withdraw(double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
            System.out.println("Withdrawn $" + amount + ". New balance: $" +
balance);
        } else {
            System.out.println("Invalid withdrawal amount or insufficient funds!");
        }
    }
}

public class BankEncap {
    public static void main(String[] args) {
        BankAccount account = new BankAccount("12345", 1000);
        System.out.println("Account Number: " + account.getAccountNumber());
        System.out.println("Initial Balance: $" + account.getBalance());

        account.deposit(500);
        account.withdraw(200);
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\encapsulation>java BankEncap.java
Account Number: 12345
Initial Balance: $1000.0
Deposited $500.0. New balance: $1500.0
Withdrawn $200.0. New balance: $1300.0

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\encapsulation>
```

14.b) EmpEncap

CODE:

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

    // Constructor
    public Employee(String name, double salary) {
        this.name = name;
        this.salary = salary;
    }

    // Getters and Setters
    public String getName() {
```

```
        return name;
    }

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

    public double getSalary() {
        return salary;
    }

    public void setSalary(double salary) {
        if (salary > 0) {
            this.salary = salary;
        } else {
            System.out.println("Salary cannot be negative!");
        }
    }

    void display() {
        System.out.println("Name: " + name + ", Salary: $" + salary);
    }
}

public class EmpEncap {
    public static void main(String[] args) {
        Employee emp = new Employee("Alice", 50000);
        emp.display();

        emp.setSalary(60000);
        System.out.println("Updated Salary for " + emp.getName() + ": $" +
emp.getSalary());
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\encapsulation>java EmpEncap.java
Name: Alice, Salary: $50000.0
Updated Salary for Alice: $60000.0

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\encapsulation>
```

14.c) ShoppingCartApp

CODE:

```
import java.util.ArrayList;
import java.util.List;
```

```
// Encapsulated Class for Product
class Product {
    private String name;
    private double price;
    private int quantity;

    // Constructor
    public Product(String name, double price, int quantity) {
        this.name = name;
        this.price = price;
        this.quantity = quantity;
    }

    // Getters (Encapsulation)
    public String getName() { return name; }
    public double getPrice() { return price; }
    public int getQuantity() { return quantity; }

    // Setter to update quantity
    public void setQuantity(int quantity) {
        if (quantity > 0) {
            this.quantity = quantity;
        } else {
            System.out.println("Quantity must be positive!");
        }
    }
}

// Encapsulated ShoppingCart Class
class ShoppingCart {
    private List<Product> cart = new ArrayList<>();

    // Add product to cart
    public void addProduct(Product product) {
        cart.add(product);
    }

    // Display cart details
    public void displayCart() {
        double total = 0;
        System.out.println("\nYour Shopping Cart:");
        for (Product p : cart) {
            System.out.println(p.getName() + " - ₹" + p.getPrice() + " x " +
p.getQuantity());
            total += p.getPrice() * p.getQuantity();
        }
        System.out.println("Total Price: ₹" + total);
    }
}

public class ShoppingCartApp {
    public static void main(String[] args) {
        // Creating products
```

```
        Product p1 = new Product("Laptop", 55000, 1);
        Product p2 = new Product("Headphones", 2000, 2);

        // Creating cart and adding products
        ShoppingCart cart = new ShoppingCart();
        cart.addProduct(p1);
        cart.addProduct(p2);

        // Display cart details
        cart.displayCart();
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\encapsulation>java ShoppingCartApp.jav
a

Your Shopping Cart:
Laptop - ?55000.0 x 1
Headphones - ?2000.0 x 2
Total Price: ?59000.0

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\encapsulation>
```

14.d) StudentMarksApp

CODE:

```
import java.util.Scanner;

// Encapsulated Class for Student
class Student {
    private String name;
    private int rollNumber;
    private int marks;

    // Constructor
    public Student(String name, int rollNumber, int marks) {
        this.name = name;
        this.rollNumber = rollNumber;
        setMarks(marks); // Using setter to ensure validation
    }

    // Getters
    public String getName() { return name; }
    public int getRollNumber() { return rollNumber; }
    public int getMarks() { return marks; }

    // Setter with validation
    public void setMarks(int marks) {
        if (marks >= 0 && marks <= 100) {
            this.marks = marks;
        } else {
```

```
            System.out.println("Invalid marks! Marks must be between 0 and 100.");
            this.marks = 0; // Default value
        }
    }

    // Display student details
    public void displayInfo() {
        System.out.println("\nStudent Details:");
        System.out.println("Name: " + name);
        System.out.println("Roll Number: " + rollNumber);
        System.out.println("Marks: " + marks);
    }
}

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

        // Taking user input for student details
        System.out.print("Enter Student Name: ");
        String name = scanner.nextLine();

        System.out.print("Enter Roll Number: ");
        int rollNumber = scanner.nextInt();

        System.out.print("Enter Marks (0-100): ");
        int marks = scanner.nextInt();

        // Creating student object
        Student student = new Student(name, rollNumber, marks);

        // Displaying student details
        student.displayInfo();

        scanner.close();
    }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\encapsulation>javac StudentMarksApp.ja
va

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\encapsulation>java StudentMarksApp.jav
a
Enter Student Name: me
Enter Roll Number: 50
Enter Marks (0-100): 24

Student Details:
Name: me
Roll Number: 50
Marks: 24

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\encapsulation>
```

15. PACKAGES PROGRAMS

15.a)User Defined Packages- Library Management System

CODE:

Library package

```
// Define the package
package library;

public class Book {
   private String title;
   private String author;
   private int bookID;

   // Constructor
   public Book(String title, String author, int bookID) {
      this.title = title;
      this.author = author;
      this.bookID = bookID;
   }

   // Display book details
   public void displayBookInfo() {
      System.out.println("Book ID: " + bookID);
      System.out.println("Title: " + title);
      System.out.println("Author: " + author);
      System.out.println("---------------------------");
   }
}
```

Main program:

```
// Import the package
import library.Book;
import java.util.ArrayList;
import java.util.Scanner;

public class LibraryManagement {
   public static void main(String[] args) {
      // Create an ArrayList to store books
      ArrayList<Book> books = new ArrayList<>();
      Scanner scanner = new Scanner(System.in);

      while (true) {
         System.out.println("\nLibrary Management System");
         System.out.println("1. Add Book");
         System.out.println("2. Display Books");
         System.out.println("3. Exit");
         System.out.print("Enter choice: ");

         int choice = scanner.nextInt();
         scanner.nextLine(); // Consume newline
```

```
        switch (choice) {
            case 1:
                // Add a new book
                System.out.print("Enter Book ID: ");
                int bookID = scanner.nextInt();
                scanner.nextLine(); // Consume newline

                System.out.print("Enter Book Title: ");
                String title = scanner.nextLine();

                System.out.print("Enter Author Name: ");
                String author = scanner.nextLine();

                books.add(new Book(title, author, bookID));
                System.out.println("Book added successfully!");
                break;

            case 2:
                // Display all books
                if (books.isEmpty()) {
                    System.out.println("No books available.");
                } else {
                    System.out.println("\nBook List:");
                    for (Book book : books) {
                        book.displayBookInfo();
                    }
                }
                break;

            case 3:
                // Exit
                System.out.println("Exiting the Library Management System...");
                scanner.close();
                return;

            default:
                System.out.println("Invalid choice! Please try again.");
        }
    }
  }
}
```

OUPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\package>java LibraryManagement

Library Management System
1. Add Book
2. Display Books
3. Exit
Enter choice: 1
Enter Book ID: 54
Enter Book Title: What is life?
Enter Author Name: Schrodinger
Book added successfully!

Library Management System
1. Add Book
2. Display Books
3. Exit
Enter choice: 2

Book List:
Book ID: 54
Title: What is life?
Author: Schrodinger
--------------------------

Library Management System
1. Add Book
2. Display Books
3. Exit
```

15.b)User Defined Packages -- Employee Management System

CODE:

Main program:

```
import employee.*;

public class Main2 {
   public static void main(String[] args) {
      Employee emp = new Employee("Bob", 50000);
      HRManager hr = new HRManager();

      hr.processSalary(emp);
   }
}
```

Employee class:

```
import employee.*;

public class Main2 {
   public static void main(String[] args) {
      Employee emp = new Employee("Bob", 50000);
      HRManager hr = new HRManager();

      hr.processSalary(emp);
   }
}
```

HR manager class:

```
package employee;

public class HRManager {
   public void processSalary(Employee emp) {
      System.out.println("Processing salary for:");
      emp.displayEmployeeInfo();
   }
}
```

OUPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\package>java Main2
Processing salary for:
Employee: Bob, Salary: $50000.0

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\package>
```

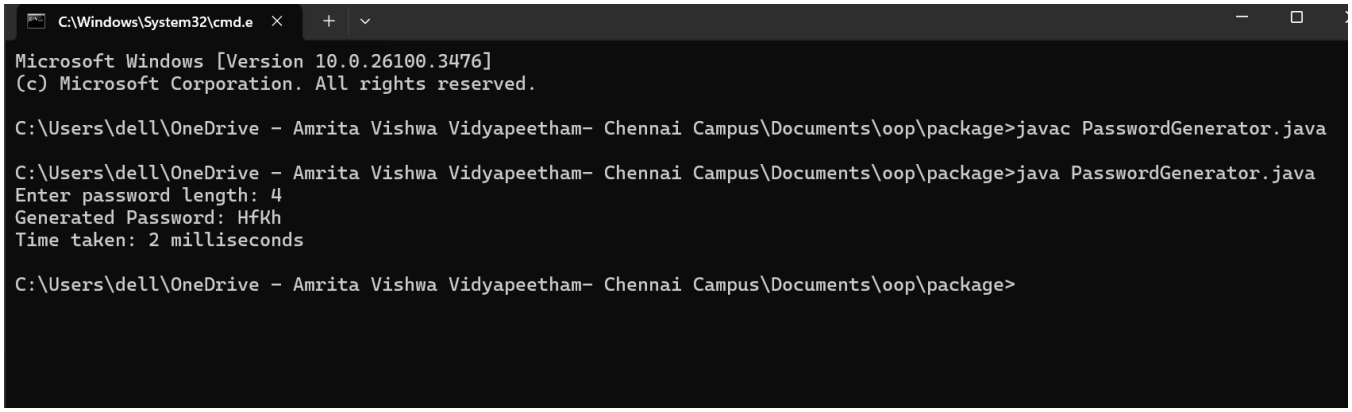
15.c)Built – in Package(3 Packages) -- password generator

CODE:

```
import java.util.Random;    // For random password generation
import java.util.Scanner;   // For user input
import java.time.Instant;   // To measure execution time
import java.time.Duration;  // Calculate time taken

public class PasswordGenerator {
   // Function to generate a random password
   public static String generatePassword(int length) {
      String chars =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789@#$%&*!";
      Random random = new Random();
      StringBuilder password = new StringBuilder();

      for (int i = 0; i < length; i++) {
         int index = random.nextInt(chars.length());
         password.append(chars.charAt(index));
      }

      return password.toString();
   }

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

```
        System.out.print("Enter password length: ");
        int length = scanner.nextInt();

        // Start timing execution
        Instant start = Instant.now();

        // Generate password
        String password = generatePassword(length);

        // End timing execution
        Instant end = Instant.now();
        Duration timeElapsed = Duration.between(start, end);

        System.out.println("Generated Password: " + password);
        System.out.println("Time taken: " + timeElapsed.toMillis() + " milliseconds");

        scanner.close();
    }
}
```

OUPUT:



15.d)Built – in Package(3 Packages)–File Compressor

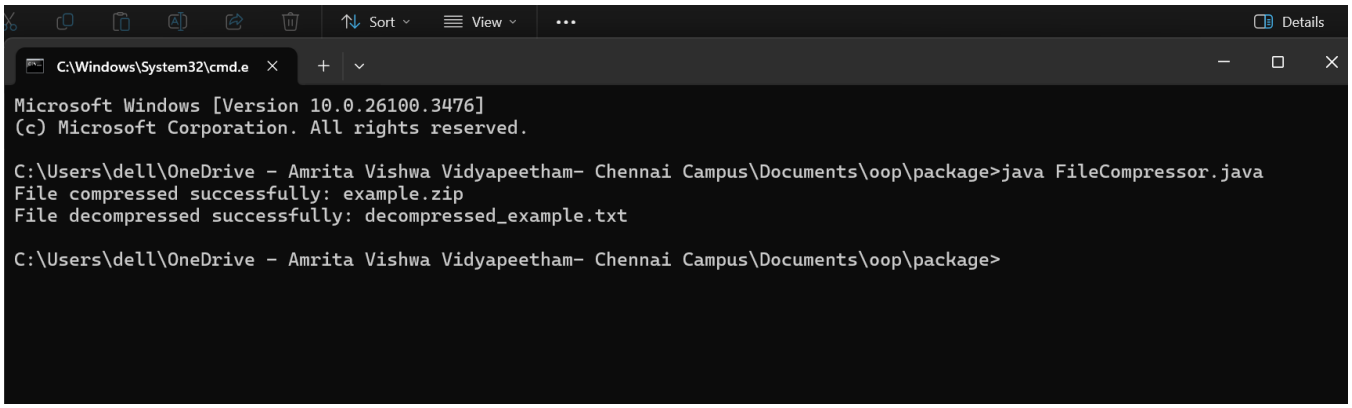
CODE:

```
import java.io.*;
import java.util.zip.*;
import java.nio.file.*;

public class FileCompressor {
    // Function to compress a file
    public static void compressFile(String filePath, String zipFilePath) {
        try (FileInputStream fis = new FileInputStream(filePath);
            FileOutputStream fos = new FileOutputStream(zipFilePath);
            ZipOutputStream zos = new ZipOutputStream(fos)) {

            // Create a zip entry
```

```
        ZipEntry zipEntry = new ZipEntry(new File(filePath).getName());
        zos.putNextEntry(zipEntry);

        byte[] buffer = new byte[1024];
        int bytesRead;
        while ((bytesRead = fis.read(buffer)) != -1) {
            zos.write(buffer, 0, bytesRead);
        }

        zos.closeEntry();
        System.out.println("File compressed successfully: " + zipFilePath);
    } catch (IOException e) {
        System.out.println("Error during compression: " + e.getMessage());
    }
}

// Function to decompress a file
public static void decompressFile(String zipFilePath, String outputFilePath) {
    try (FileInputStream fis = new FileInputStream(zipFilePath);
         ZipInputStream zis = new ZipInputStream(fis);
         FileOutputStream fos = new FileOutputStream(outputFilePath)) {

        ZipEntry entry = zis.getNextEntry();
        if (entry == null) {
            System.out.println("Invalid ZIP file.");
            return;
        }

        byte[] buffer = new byte[1024];
        int bytesRead;
        while ((bytesRead = zis.read(buffer)) != -1) {
            fos.write(buffer, 0, bytesRead);
        }

        System.out.println("File decompressed successfully: " + outputFilePath);
    } catch (IOException e) {
        System.out.println("Error during decompression: " + e.getMessage());
    }
}

public static void main(String[] args) {
    String inputFile = "example.txt";  // File to be compressed
    String zipFile = "example.zip";   // Compressed file
    String outputFile = "decompressed_example.txt"; // Decompressed file

    // Create a sample file
    try {
        Files.writeString(Path.of(inputFile), "Hello, this is a test file for compression!");
    } catch (IOException e) {
        System.out.println("Error creating sample file: " + e.getMessage());
    }

    // Compress and decompress the file
```

```
        compressFile(inputFile, zipFile);
        decompressFile(zipFile, outputFile);
    }
}
```

OUPUT:



16. EXCEPTION HANDLING PROGRAMS

16.a) AgeValidation

CODE:

```
import java.util.Scanner;

// Custom Exception Class
class InvalidAgeException extends Exception {
    public InvalidAgeException(String message) {
        super(message);
    }
}

public class AgeValidation {
    // Function to validate age
    public static void validateAge(int age) throws InvalidAgeException {
        if (age < 18) {
            throw new InvalidAgeException("Age must be 18 or above for registration.");
        } else {
            System.out.println("Registration successful!");
        }
    }

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

        System.out.print("Enter your age: ");
        int age = scanner.nextInt();
```

```
        try {
            validateAge(age);
        } catch (InvalidAgeException e) {
            System.out.println("Error: " + e.getMessage());
        } finally {
            scanner.close();
            System.out.println("Validation process completed.");
        }
    }
}
```

OUPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\exception handling>java AgeValidation.java
Enter your age: 89
Registration successful!
Validation process completed.

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\exception handling>
```

16.b) ArrayExceptionHandling

CODE:

```
public class ArrayExceptionHandling {

    public static void main(String[] args) {
        int[] numbers = {10, 20, 30, 40, 50};

        try {
            System.out.println("Accessing element at index 10: " + numbers[10]); // Invalid index
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Error: Index out of bounds! " + e.getMessage());
        } finally {
            System.out.println("Array operation completed.");
        }
    }
}
```

OUPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\exception handling>java ArrayExceptionHandling.java
Error: Index out of bounds! Index 10 out of bounds for length 5
Array operation completed.

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\exception handling>
```

16.c) DivisionHandling

CODE:

```
import java.util.Scanner; // For user input

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

      try {
         System.out.print("Enter numerator: ");
         int numerator = scanner.nextInt();

         System.out.print("Enter denominator: ");
         int denominator = scanner.nextInt();

         // Performing division
         int result = numerator / denominator;
         System.out.println("Result: " + result);

      } catch (ArithmeticException e) {
         System.out.println("Error: Cannot divide by zero!");
      } finally {
         scanner.close(); // Closing Scanner
         System.out.println("Operation completed.");
      }
   }
}
```

OUTPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\exception handling>java DivisionHandli
ng.java
Enter numerator: 5
Enter denominator: 0
Error: Cannot divide by zero!
Operation completed.

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\exception handling>
```

16.d) FileExceptionHandling

CODE:

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

```
public class FileExceptionHandling {
   public static void main(String[] args) {
      File file = new File("non_existing_file.txt"); // File does not exist

      try {
         Scanner fileReader = new Scanner(file);
         while (fileReader.hasNextLine()) {
            System.out.println(fileReader.nextLine());
         }
         fileReader.close();
      } catch (FileNotFoundException e) {
         System.out.println("Error: File not found!");
      }
   }
}
```

OUPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\exception handling>java FileExceptionH
andling.java
Error: File not found!

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\exception handling>
```

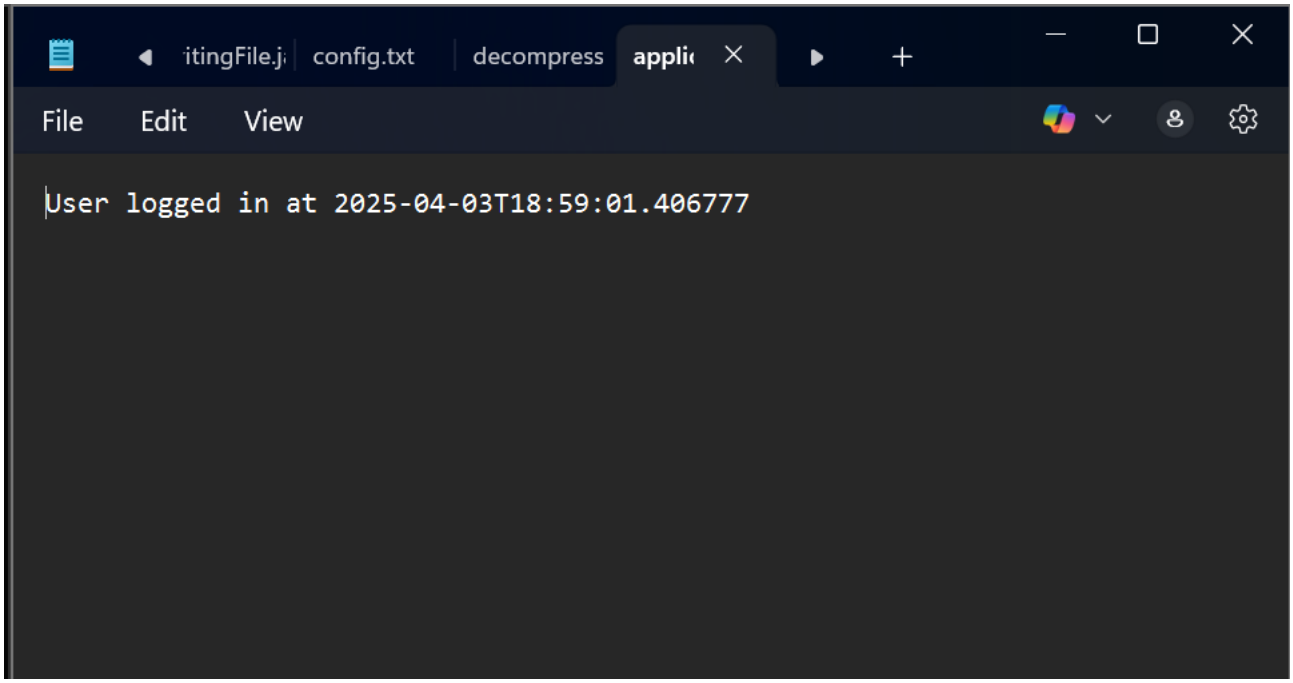
17. FILE HANDLING PROGRAMS

17.a)logwriter

CODE:

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

public class logwriter {
   public static void main(String[] args) {
      try {
         FileWriter writer = new FileWriter("application.log", true);
         String logMessage = "User logged in at " + java.time.LocalDateTime.now() + "\n";
         writer.write(logMessage);
         writer.close();
         System.out.println("Log written successfully.");
      } catch (IOException e) {
         System.out.println("Error writing to file: " + e.getMessage());
      }
   }
}
```

OUPUT:



17.b)todolist

CODE:

```
// Define the package
package todolist;
import java.io.*;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class TaskManager {
   private static final String FILE_NAME = "tasks.txt";
   private static List<String> tasks = new ArrayList<>();

   public static void main(String[] args) {
      loadTasks(); // Load tasks from file
      Scanner scanner = new Scanner(System.in);

      while (true) {
         System.out.println("\nTo-Do List Manager");
         System.out.println("1. Add Task");
         System.out.println("2. View Tasks");
         System.out.println("3. Exit and Save");
         System.out.print("Enter your choice: ");
```

```
        int choice = scanner.nextInt();
        scanner.nextLine(); // Consume newline

        switch (choice) {
            case 1:
                // Add a new task
                System.out.print("Enter task description: ");
                String task = scanner.nextLine();
                tasks.add(task);
                System.out.println("Task added!");
                break;

            case 2:
                // Display tasks
                if (tasks.isEmpty()) {
                    System.out.println("No tasks available.");
                } else {
                    System.out.println("\nYour Tasks:");
                    for (int i = 0; i < tasks.size(); i++) {
                        System.out.println((i + 1) + ". " + tasks.get(i));
                    }
                }
                break;

            case 3:
                // Save tasks and exit
                saveTasks();
                System.out.println("Tasks saved. Exiting...");
                scanner.close();
                return;

            default:
                System.out.println("Invalid choice! Please try again.");
        }
    }
}

// Load tasks from file
private static void loadTasks() {
    try (BufferedReader br = new BufferedReader(new FileReader(FILE_NAME))) {
        String line;
        while ((line = br.readLine()) != null) {
            tasks.add(line);
        }
    } catch (FileNotFoundException e) {
        System.out.println("No previous tasks found.");
    } catch (IOException e) {
        System.out.println("Error reading tasks: " + e.getMessage());
    }
}

// Save tasks to file
```

```
    private static void saveTasks() {
        try (BufferedWriter bw = new BufferedWriter(new FileWriter(FILE_NAME))) {
            for (String task : tasks) {
                bw.write(task);
                bw.newLine();
            }
        } catch (IOException e) {
            System.out.println("Error saving tasks: " + e.getMessage());
        }
    }
}
```

OUPUT:

```
C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\file handling\todolist>java TaskManage
r.java
No previous tasks found.

To-Do List Manager
1. Add Task
2. View Tasks
3. Exit and Save
Enter your choice: 2
No tasks available.

To-Do List Manager
1. Add Task
2. View Tasks
3. Exit and Save
Enter your choice:
```

17.c)studentmanagement

CODE:

MAIN PROGRAM

```
// Define the package
package todolist;
import java.io.*;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class TaskManager {
    private static final String FILE_NAME = "tasks.txt";
    private static List<String> tasks = new ArrayList<>();

    public static void main(String[] args) {
        loadTasks(); // Load tasks from file
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("\nTo-Do List Manager");
```

```
        System.out.println("1. Add Task");
        System.out.println("2. View Tasks");
        System.out.println("3. Exit and Save");
        System.out.print("Enter your choice: ");

        int choice = scanner.nextInt();
        scanner.nextLine(); // Consume newline

        switch (choice) {
            case 1:
                // Add a new task
                System.out.print("Enter task description: ");
                String task = scanner.nextLine();
                tasks.add(task);
                System.out.println("Task added!");
                break;

            case 2:
                // Display tasks
                if (tasks.isEmpty()) {
                    System.out.println("No tasks available.");
                } else {
                    System.out.println("\nYour Tasks:");
                    for (int i = 0; i < tasks.size(); i++) {
                        System.out.println((i + 1) + ". " + tasks.get(i));
                    }
                }
                break;

            case 3:
                // Save tasks and exit
                saveTasks();
                System.out.println("Tasks saved. Exiting...");
                scanner.close();
                return;

            default:
                System.out.println("Invalid choice! Please try again.");
        }
    }
}

// Load tasks from file
private static void loadTasks() {
    try (BufferedReader br = new BufferedReader(new FileReader(FILE_NAME))) {
        String line;
        while ((line = br.readLine()) != null) {
            tasks.add(line);
        }
    } catch (FileNotFoundException e) {
        System.out.println("No previous tasks found.");
    } catch (IOException e) {
        System.out.println("Error reading tasks: " + e.getMessage());
```

```
        }
    }

    // Save tasks to file
    private static void saveTasks() {
        try (BufferedWriter bw = new BufferedWriter(new FileWriter(FILE_NAME))) {
            for (String task : tasks) {
                bw.write(task);
                bw.newLine();
            }
        } catch (IOException e) {
            System.out.println("Error saving tasks: " + e.getMessage());
        }
    }
}
```

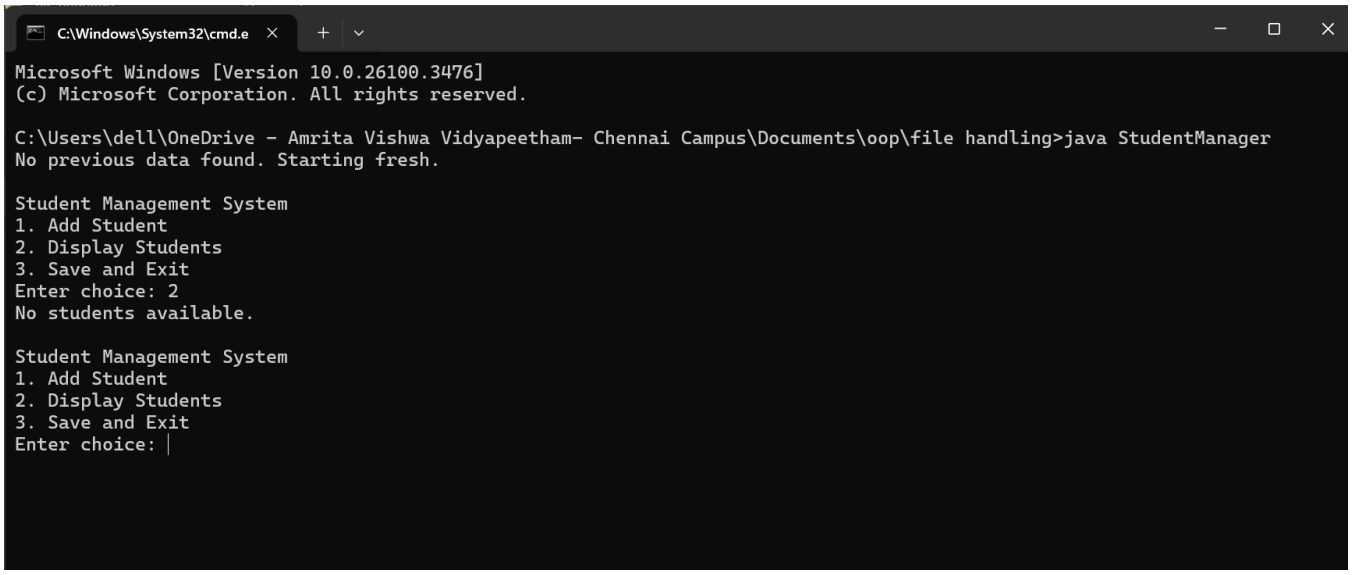
STUDENT MANAGER PROGRAM

```
// Import required packages
import studentmanagement.Student;
import java.io.*;
import java.util.ArrayList;
import java.util.Scanner;

public class StudentManager {
    private static final String FILE_NAME = "students.dat";
    private static ArrayList<Student> students = new ArrayList<>();

    public static void main(String[] args) {
        loadStudents(); // Load existing data
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("\nStudent Management System");
            System.out.println("1. Add Student");
            System.out.println("2. Display Students");
            System.out.println("3. Save and Exit");
            System.out.print("Enter choice: ");

            int choice = scanner.nextInt();
            scanner.nextLine(); // Consume newline

            switch (choice) {
                case 1:
                    // Add a student
                    System.out.print("Enter Roll Number: ");
                    int rollNo = scanner.nextInt();
                    scanner.nextLine();

                    System.out.print("Enter Name: ");
                    String name = scanner.nextLine();
```

```
            System.out.print("Enter Marks: ");
            double marks = scanner.nextDouble();

            students.add(new Student(name, rollNo, marks));
            System.out.println("Student added successfully!");
            break;

        case 2:
            // Display all students
            if (students.isEmpty()) {
                System.out.println("No students available.");
            } else {
                System.out.println("\nStudent List:");
                for (Student s : students) {
                    s.displayStudent();
                }
            }
            break;

        case 3:
            // Save students and exit
            saveStudents();
            System.out.println("Data saved successfully. Exiting...");
            scanner.close();
            return;

        default:
            System.out.println("Invalid choice! Please try again.");
        }
    }
}

// Load students from file
private static void loadStudents() {
    try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(FILE_NAME))) {
        students = (ArrayList<Student>) ois.readObject();
    } catch (FileNotFoundException e) {
        System.out.println("No previous data found. Starting fresh.");
    } catch (IOException | ClassNotFoundException e) {
        System.out.println("Error loading data: " + e.getMessage());
    }
}

// Save students to file
private static void saveStudents() {
    try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(FILE_NAME)))
{
        oos.writeObject(students);
    } catch (IOException e) {
        System.out.println("Error saving data: " + e.getMessage());
    }
}
```

```
}
```

OUPUT:



17.d)reading file

CODE:

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
public class WritingFile {
   public static void main(String[] args) {
      try {
         FileReader fileReader = new FileReader("config.txt");
         BufferedReader bufferedReader = new BufferedReader(fileReader);

         String line;
         System.out.println("Reading configuration:");
         while ((line = bufferedReader.readLine()) != null) {
            System.out.println(line);
         }

         bufferedReader.close();
      } catch (IOException e) {
         System.out.println("Error reading file: " + e.getMessage());
      }
   }
}
```

READING TEXT FILE

Reading comprehension is the ability to understand and interpret written text, which involves integrating the words on the page with existing knowledge and understanding the meaning of the text.
It is a crucial skill that depends on both word reading and language comprehension, and it is essential for success in school and in life.
Effective reading comprehension requires skills such as recognizing literary devices, understanding the situational mood, and determining the writer's purpose.

There are various resources available for improving reading comprehension skills, including free worksheets and interactive exercises. Websites like ReadTheory offer personalized reading comprehension exercises for students from kindergarten to 12th grade, adjusting to each student's reading level and providing instant feedback.
Additionally, sites like K5 Learning provide free reading worksheets and stories with comprehension exercises for children in kindergarten through fifth grade.

OUPUT:

```
C:\Windows\System32\cmd.e

Microsoft Windows [Version 10.0.26100.3476]
(c) Microsoft Corporation. All rights reserved.

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\file handling\ReadingConf>javac Writin
gFile.java

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\file handling\ReadingConf>java Writing
File.java
Reading configuration:
Reading comprehension is the ability to understand and interpret written text, which involves integrating the words on t
he page with existing knowledge and understanding the meaning of the text.
 It is a crucial skill that depends on both word reading and language comprehension, and it is essential for success in
school and in life.
 Effective reading comprehension requires skills such as recognizing literary devices, understanding the situational moo
d, and determining the writer's purpose.

There are various resources available for improving reading comprehension skills, including free worksheets and interact
ive exercises. Websites like ReadTheory offer personalized reading comprehension exercises for students from kindergarte
n to 12th grade, adjusting to each student's reading level and providing instant feedback.
 Additionally, sites like K5 Learning provide free reading worksheets and stories with comprehension exercises for child
ren in kindergarten through fifth grade.

C:\Users\dell\OneDrive - Amrita Vishwa Vidyapeetham- Chennai Campus\Documents\oop\file handling\ReadingConf>
```