# Assignment 1

## Problem 1

1. What are the differences between procedural programming and object oriented programming?

Figure 1: Problem 1 Task Requirement

**Core Concepts**

- **Procedural Programming**: Focuses on functions and procedures to execute a sequence of tasks.
- **OOP**: Centers on objects, which encapsulate data and behaviors (methods).

**Structure**

- **Procedural**: Organized in a linear flow of functions.
- **OOP**: Organized around classes and objects.

**Data Handling**

- **Procedural**: Data and functions are separate; data is passed to functions.
- **OOP**: Data and functions are bundled; methods operate on object data.

**Key Features**

- **Encapsulation**: OOP promotes data hiding and access control; procedural programming has minimal encapsulation.
- **Inheritance**: OOP supports inheritance for code reuse; procedural does not.
- **Polymorphism**: OOP allows method overloading and overriding; procedural has limited support.

**Use Cases**

- **Procedural**: Best for simple scripts and tasks.
- **OOP**: Ideal for complex applications requiring modularity and reusability.

## Problem 2

Place image of task requirement here

Figure 2: Problem 2 Task Requirement

- **Banking Systems**: Many banking applications use Java for secure, scalable, and maintainable systems, managing accounts, transactions, and user interactions.
- **E-commerce Platforms**: Platforms like eBay and Amazon utilize Java to handle extensive product catalogs, user data, and transactions effectively.
- **Healthcare Systems**: Java is used in healthcare applications for managing patient records, scheduling, and billing, ensuring data integrity and security.

## Problem 3

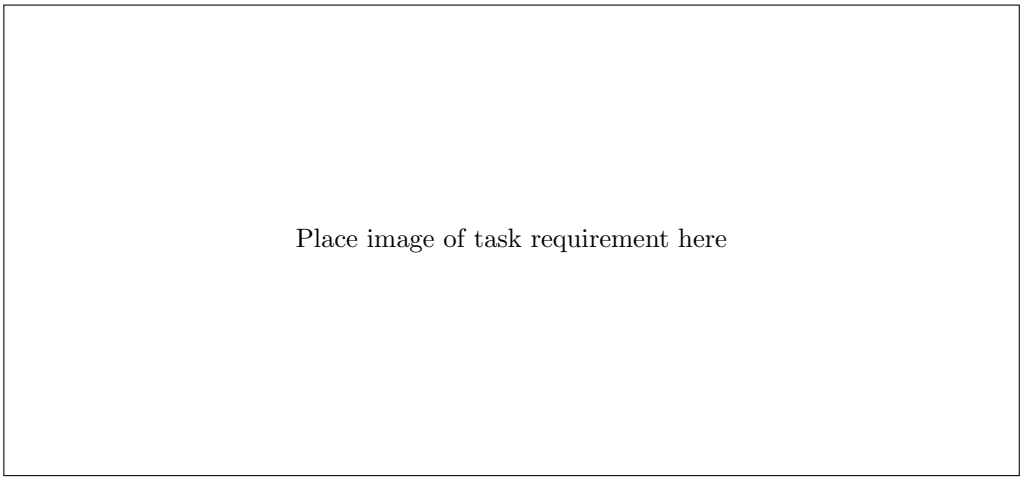


Figure 3: Problem 3 Task Requirement

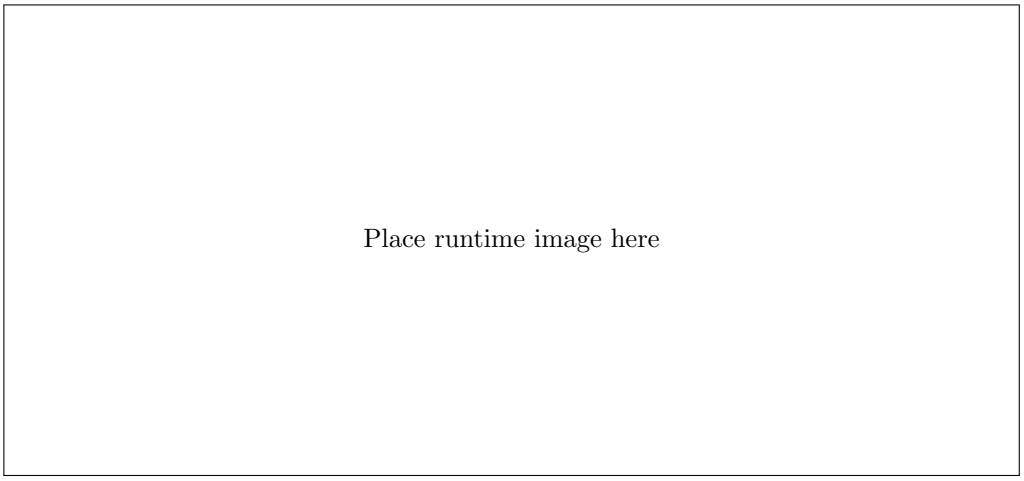


Figure 4: Problem 3 Runtime Output

# Assignment 2

## 2.1 DisplayInitials.java

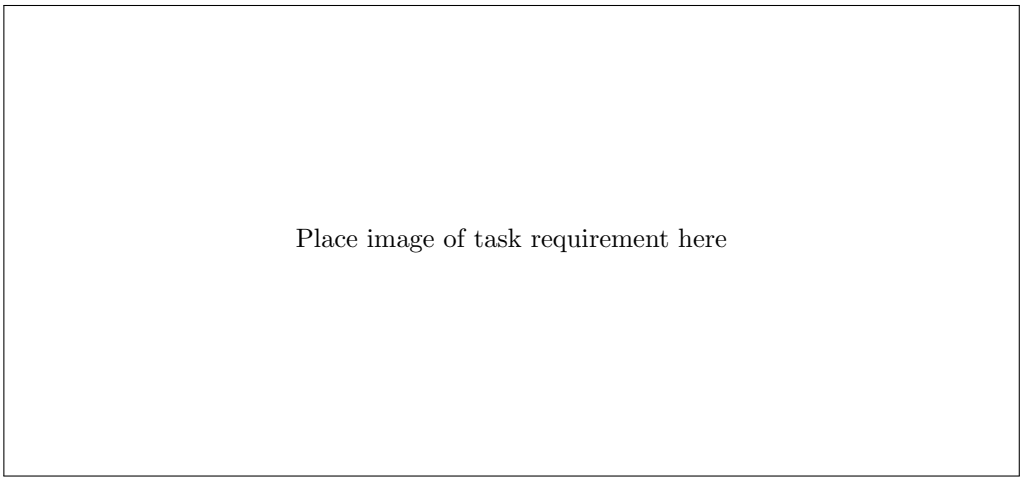


Figure 5: 2.1 Task Requirement

```java
import javax.swing.JOptionPane;

public class DisplayIntergers {
    public static void main(String[] args) {
        String firstInput = JOptionPane.showInputDialog("Hay nhap so thu 1:");
        String secondInput = JOptionPane.showInputDialog("Hay nhap so thu 2:");

        int firstNumber = Integer.parseInt(firstInput);
        int secondNumber = Integer.parseInt(secondInput);

        JOptionPane.showMessageDialog(null, "So thu 1: " + firstNumber);
        JOptionPane.showMessageDialog(null, "So thu 2: " + secondNumber);
    }
}
```

1: DisplayInitials.java

## 2.2 (Ex17 - Book) IngredientAdjuster.java

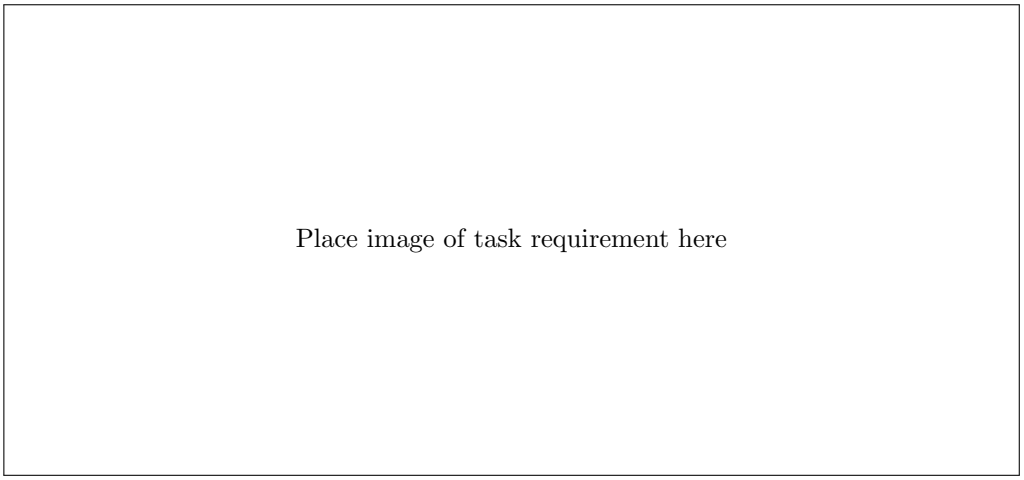


Figure 6: IngredientAdjuster Task Requirement

```java
import java.util.Scanner;

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

        final double sugarPerCookie = 1.5 / 48;
        final double butterPerCookie = 1.0 / 48;
        final double flourPerCookie = 2.75 / 48;

        System.out.print("Enter the number of cookies you want to make: ");
        int cookiesWanted = scanner.nextInt();

        double sugarNeeded = sugarPerCookie * cookiesWanted;
        double butterNeeded = butterPerCookie * cookiesWanted;
        double flourNeeded = flourPerCookie * cookiesWanted;

        System.out.printf("For %d cookies, you need:\n", cookiesWanted);
        System.out.printf("%.2f cups of sugar\n", sugarNeeded);
        System.out.printf("%.2f cups of butter\n", butterNeeded);
        System.out.printf("%.2f cups of flour\n", flourNeeded);

        scanner.close();
    }
}
```

2: IngredientAdjuster.java

## (Ex18 - Book) WordGame.java

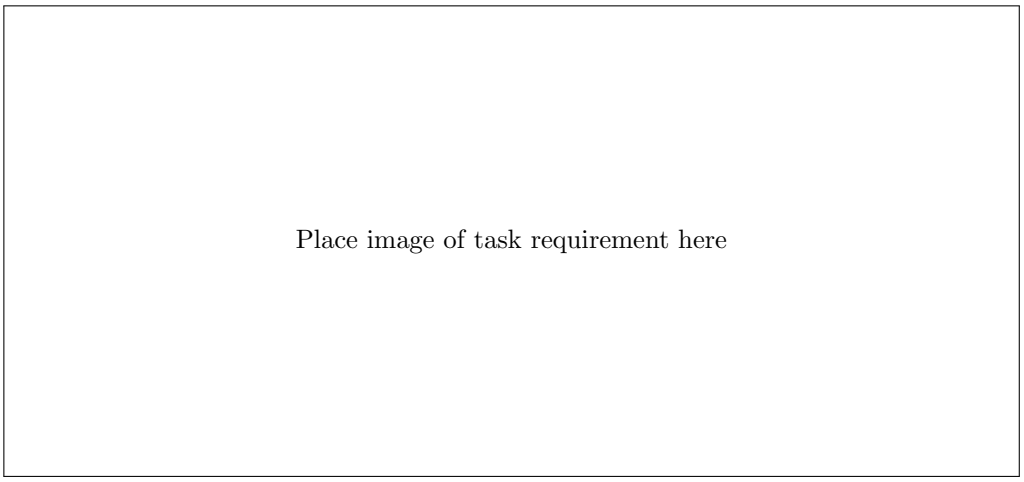


Figure 7: WordGame Task Requirement

```java
import java.util.Scanner;

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

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

        System.out.print("Enter your age: ");
        int age = scanner.nextInt();
        scanner.nextLine();  // Consume the newline

        System.out.print("Enter the name of a city: ");
        String city = scanner.nextLine();

        System.out.print("Enter the name of a college: ");
        String college = scanner.nextLine();

        System.out.print("Enter a profession: ");
        String profession = scanner.nextLine();

        System.out.print("Enter a type of animal: ");
        String animal = scanner.nextLine();

        System.out.print("Enter a pet's name: ");
        String petName = scanner.nextLine();

        System.out.println("\nHere is your story:");
        System.out.println("There once was a person named " + name + " who lived
            in " + city + ".");
        System.out.println("At the age of " + age + ", " + name + " went to
            college at " + college + ".");
        System.out.println(name + " graduated and went to work as a(n) " +
            profession + ".");
        System.out.println("Then, " + name + " adopted a(n) " + animal + " named "
             + petName + ".");
```

```
34          System.out.println("They both lived happily ever after!");

36          scanner.close();
37      }
38  }
```

3: WordGame.java

## (Ex19 - Book) StockTransaction.java

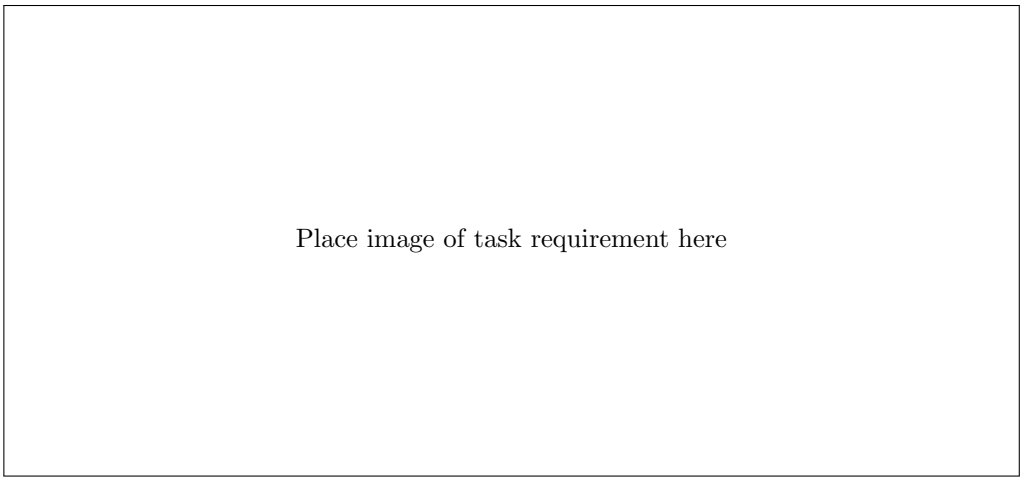


Figure 8: StockTransaction Task Requirement

```
1  public class StockTransaction {
2      public static void main(String[] args) {
3          int sharesBought = 1000;
4          double purchasePricePerShare = 32.87;
5          double commissionRate = 0.02;

7          int sharesSold = 1000;
8          double salePricePerShare = 33.92;

10         double purchaseAmount = sharesBought * purchasePricePerShare;
11         double purchaseCommission = purchaseAmount * commissionRate;

13         double saleAmount = sharesSold * salePricePerShare;
14         double saleCommission = saleAmount * commissionRate;

16         double profit = (saleAmount - saleCommission) - (purchaseAmount +
               purchaseCommission);

18         System.out.printf("Amount paid for the stock: $%.2f%n", purchaseAmount);
19         System.out.printf("Commission paid on the purchase: $%.2f%n",
               purchaseCommission);
20         System.out.printf("Amount the stock sold for: $%.2f%n", saleAmount);
21         System.out.printf("Commission paid on the sale: $%.2f%n", saleCommission);
22         System.out.printf("Profit after commissions: $%.2f%n", profit);
23     }
24 }
```

4: StockTransaction.java

### 2.3 DisplayIntergers.java

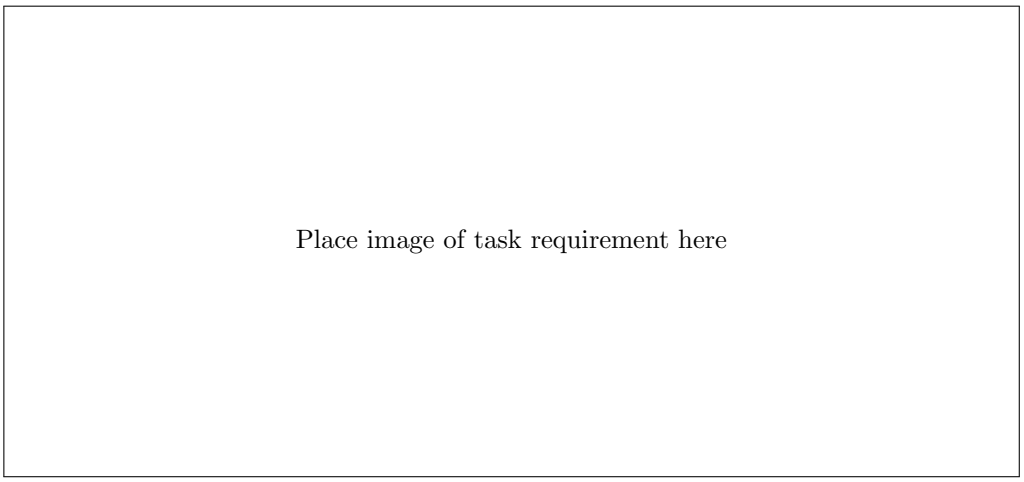


Figure 9: 2.3 Task Requirement

```java
import javax.swing.JOptionPane;

public class DisplayIntergers {
    public static void main(String[] args) {
        String firstInput = JOptionPane.showInputDialog("Hay nhap so thu 1:");
        String secondInput = JOptionPane.showInputDialog("Hay nhap so thu 2:");

        int firstNumber = Integer.parseInt(firstInput);
        int secondNumber = Integer.parseInt(secondInput);

        JOptionPane.showMessageDialog(null, "So thu 1: " + firstNumber);
        JOptionPane.showMessageDialog(null, "So thu 2: " + secondNumber);
    }
}
```

5: DisplayIntergers.java

## Assignment 3

### 1. BMI Measure

Figure 10: BMI Measure Task Requirement

```java
import java.util.Scanner;

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

        final double POUNDS_TO_KG = 0.45359237;
        final double INCHES_TO_METERS = 0.0254;
        Scanner input = new Scanner(System.in);
        System.out.print("Enter your weight in pounds: ");
        double weightPounds = input.nextDouble();
```

```
        System.out.print("Enter your height in inches: ");
        double heightInches = input.nextDouble();
        double weightKg = weightPounds * POUNDS_TO_KG;
        double heightMeters = heightInches * INCHES_TO_METERS;
        double bmi = weightKg / (heightMeters * heightMeters);
        System.out.printf("Your BMI is: %.2f\n", bmi);

        if (bmi < 18.5) {
            System.out.println("Category: Underweight");
        } else if (bmi < 25.0) {
            System.out.println("Category: Normal");
        } else if (bmi < 30.0) {
            System.out.println("Category: Overweight");
        } else {
            System.out.println("Category: Obese");
        }

        input.close();
    }
}
```

6: BMICalculator.java

## 2. Programming Challenges

### 3.7 SortedNames.java

Figure 11: Programming Challenge 3.7 Task Requirement

```
import java.util.Arrays;
import java.util.Scanner;

public class SortedNames {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter the first name: ");
        String name1 = input.nextLine();
        System.out.print("Enter the second name: ");
        String name2 = input.nextLine();
        System.out.print("Enter the third name: ");
        String name3 = input.nextLine();
        String[] names = { name1, name2, name3 };
        Arrays.sort(names);
        System.out.println("\nNames in ascending order:");
        for (String name : names) {
            System.out.println(name);
        }
        input.close();
    }
}
```

7: SortedNames.java

### 3.8 PackageDiscountCalculator.java

Figure 12: Programming Challenge 3.8 Task Requirement

```java
import java.util.Scanner;

public class PackageDiscountCalculator {
    public static void main(String[] args) {
        final double PACKAGE_PRICE = 99.0;
        double discountRate = 0;
        Scanner input = new Scanner(System.in);
        System.out.print("Enter the number of packages purchased: ");
        int quantity = input.nextInt();
        if (quantity >= 10 && quantity <= 19) {
            discountRate = 0.20;
        } else if (quantity >= 20 && quantity <= 49) {
            discountRate = 0.30;
        } else if (quantity >= 50 && quantity <= 99) {
            discountRate = 0.40;
        } else if (quantity >= 100) {
            discountRate = 0.50;
        }
        double discountAmount = quantity * PACKAGE_PRICE * discountRate;
        double totalAmount = (quantity * PACKAGE_PRICE) - discountAmount;
        System.out.println("Discount amount: $" + discountAmount);
        System.out.println("Total amount after discount: $" + totalAmount);
        input.close();
    }
}
```

8: PackageDiscountCalculator.java

### 3.9 ShippingCharges.java

Figure 13: Programming Challenge 3.9 Task Requirement

```java
import java.util.Scanner;

public class ShippingCharges {
    public static void main(String[] args) {
        final double RATE_2_POUNDS = 1.10;
        final double RATE_2_TO_6_POUNDS = 2.20;
        final double RATE_6_TO_10_POUNDS = 3.70;
        final double RATE_10_POUNDS = 3.80;
        Scanner input = new Scanner(System.in);
        System.out.print("Enter the weight of the package (in pounds): ");
        double weight = input.nextDouble();
        System.out.print("Enter the distance to be shipped (in miles): ");
        int distance = input.nextInt();
        int increments = (int) Math.ceil(distance / 500.0);
        double rate;
        if (weight <= 2) {
            rate = RATE_2_POUNDS;
        } else if (weight <= 6) {
            rate = RATE_2_TO_6_POUNDS;
        } else if (weight <= 10) {
```

```
            rate = RATE_6_TO_10_POUNDS;
        } else {
            rate = RATE_10_POUNDS;
        }
        double totalCharges = rate * increments;
        System.out.printf("The shipping charges are: $%.2f\n", totalCharges);
        input.close();
    }
}
```

9: ShippingCharges.java

# Assignment 5

## 1. Occurrence of max numbers

Figure 14: Task Requirement

```
package OccurenceOfMaxNum;

import java.util.Scanner;

public class MaxNumCounter {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("(The program will read integers input til the first 0)
            ");
        System.out.println("Enter numbers:");
        findLargestAndCount(scanner);
        scanner.close();
    }

    public static void findLargestAndCount(Scanner scanner) {
        int max = 0;
        int count = 1;

        while (true) {
            if (scanner.hasNextInt()) {
                int number = scanner.nextInt();
                if (number == 0) {
                    break;
                }
                if (number > max) {
                    max = number;
                    count = 1;
                } else if (number == max) {
                    count++;
                }
            } else {
                System.out.println("Invalid input (not integers) ignored.");
                scanner.next(); // Clear the invalid input
            }
        }

        if (max != 0) {
```

```
            System.out.println("The largest number is " + max);
            System.out.println("The occurrence count of the largest number is " +
                count);
        } else {
            System.out.println("The largest number is 0");
            System.out.println("The occurrence count of the largest number is 1");
        }
    }
}
```

10: MaxNumCounter.java

## 2. Programming Challenges

### Challenge 1

Figure 15: Task Requirement

```
package ProgrammingChallenges;
import java.util.Scanner;

public class challenge1 {
    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        //Input string
        System.out.print("Enter a string: ");
        String inputString = scanner.nextLine();

        //Input position
        int position;
        while (true) {
            System.out.print("Enter a position: ");
            position = scanner.nextInt();

            // Check if the position is valid
            if (position >= 0 && position < inputString.length()) {
                break;
            } else {
                System.out.println("Invalid position. Please enter a position
                    between 0 and " + (inputString.length() - 1));
            }
        }

        // Call the showChar method with the user inputs
        showChar(inputString, position);

        scanner.close();
    }

    public static void showChar(String str, int position) {
        if (position >= 0 && position < str.length()) {
            char ch = str.charAt(position);
            System.out.println("The character at position " + position + " is " +
                ch);
```

```
        } else {
            System.out.println("Invalid position. Please enter a position between
                0 and " + (str.length() - 1));
        }
    }
}
```

11: Challenge1.java

**Challlenge 2**

Figure 16: Task Requirement

```
package ProgrammingChallenges;

import java.util.Scanner;

public class challenge2 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input the wholesale cost
        System.out.print("Enter the item's wholesale cost: ");
        double wholesaleCost = scanner.nextDouble();

        //Input the markup percentage
        System.out.print("Enter the item's markup percentage: ");
        double markupPercentage = scanner.nextDouble();

        // Calculate the retail price
        double retailPrice = calculateRetail(wholesaleCost, markupPercentage);

        // Display the retail price
        System.out.printf("The item's retail price is: %.2f%n", retailPrice);

        scanner.close();
    }

    public static double calculateRetail(double wholesaleCost, double
        markupPercentage) {
        return wholesaleCost + (wholesaleCost * markupPercentage / 100);
    }
}
```

12: Challenge2.java

**Challenge 4**

Figure 17: Task Requirement

```
package ProgrammingChallenges;

import java.util.Scanner;
```

```java
public class challenge4 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input the number of rooms
        int numberOfRooms = getValidIntInput(scanner, "Enter the number of rooms to be painted: ");

        // Input the price of the paint per gallon
        double pricePerGallon = getValidDoubleInput(scanner, "Enter the price of the paint per gallon: ");

        // Initialize total square feet
        double totalSquareFeet = 0;

        // Input the square feet of wall space for each room
        for (int i = 1; i <= numberOfRooms; i++) {
            totalSquareFeet += getValidDoubleInput(scanner, "Enter the square feet of wall space for room " + i + ": ");
        }

        // Calculate the required data
        double gallonsOfPaintRequired = calculateGallonsOfPaintRequired(totalSquareFeet);
        double hoursOfLaborRequired = calculateHoursOfLaborRequired(totalSquareFeet);
        double costOfPaint = calculateCostOfPaint(gallonsOfPaintRequired, pricePerGallon);
        double laborCharges = calculateLaborCharges(hoursOfLaborRequired);
        double totalCost = calculateTotalCost(costOfPaint, laborCharges);

        // Display the results
        System.out.printf("The numbers of gallons of paint required: %.2f%n", gallonsOfPaintRequired);
        System.out.printf("The hours of labor required: %.2f%n", hoursOfLaborRequired);
        System.out.printf("The cost of the paint: $%.2f%n", costOfPaint);
        System.out.printf("The labor charges: $%.2f%n", laborCharges);
        System.out.printf("The total cost of the paint job: $%.2f%n", totalCost);

        scanner.close();
    }

    public static int getValidIntInput(Scanner scanner, String prompt) {
        int input;
        while (true) {
            System.out.print(prompt);
            if (scanner.hasNextInt()) {
                input = scanner.nextInt();
                scanner.nextLine(); // Consume the newline character
                break;
            } else {
                System.out.println("Invalid input. Please enter a valid integer.");
                scanner.next(); // Clear the invalid input
            }
        }
        return input;
```

```
    }

    public static double getValidDoubleInput(Scanner scanner, String prompt) {
        double input;
        while (true) {
            System.out.print(prompt);
            if (scanner.hasNextDouble()) {
                input = scanner.nextDouble();
                scanner.nextLine(); // Consume the newline character
                break;
            } else {
                System.out.println("Invalid input. Please enter a valid number.");
                scanner.next(); // Clear the invalid input
            }
        }
        return input;
    }

    public static double calculateGallonsOfPaintRequired(double totalSquareFeet) {
        return totalSquareFeet / 115;
    }

    public static double calculateHoursOfLaborRequired(double totalSquareFeet) {
        return (totalSquareFeet / 115) * 8;
    }

    public static double calculateCostOfPaint(double gallonsOfPaintRequired,
        double pricePerGallon) {
        return gallonsOfPaintRequired * pricePerGallon;
    }

    public static double calculateLaborCharges(double hoursOfLaborRequired) {
        return hoursOfLaborRequired * 18.00;
    }

    public static double calculateTotalCost(double costOfPaint, double
        laborCharges) {
        return costOfPaint + laborCharges;
    }
}
```

13: Challenge4.java

## 3. Multiplication Table

Figure 18: Task Requirement

```
package MultiplicationTable;

import javax.swing.JOptionPane;

public class MultiplicationTable {
    public static void main(String[] args) {
        StringBuilder table = new StringBuilder();

        // Add the header and underline
```

```
        table.append("Multiplication Table\n");
        table.append("-------------------------------------------------------\n");

        // Add the top row of numbers
        table.append("    |");
        for (int i = 1; i <= 9; i++) {
            table.append(String.format("%6d", i));
        }
        table.append("\n");

        // Create the multiplication table using nested for loops
        for (int i = 1; i <= 9; i++) {
            table.append(String.format("%2d |", i));
            for (int j = 1; j <= 9; j++) {
                if (i*j >= 10){
                table.append(String.format("%5d", i * j));
                } else table.append(String.format("%6d", i * j));
            }
            table.append("\n");
        }

        // Display the multiplication table in a dialog box with the specified
            header
        JOptionPane.showMessageDialog(null, table.toString(), "Example 3.4 Output"
            , JOptionPane.INFORMATION_MESSAGE);
    }
}
```

14: MultiplicationTable.java

# Assignment 6

## 1. Implementation of Student Class

Figure 19: Student Class Task Requirement

```
import java.util.Arrays;

public class Student {
    private String id;
    private String name;
    private double accumulatedCPA;
    private String[] preferCourse;

    public String getId() {
        return id;
    }

    public void setId(String id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }
```

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

    public double getAccumulatedCPA() {
        return accumulatedCPA;
    }

    public void setAccumulatedCPA(double accumulatedCPA) {
        this.accumulatedCPA = accumulatedCPA;
    }

    public String[] getPreferCourse() {
        return preferCourse;
    }

    public void setPreferCourse(String[] preferCourse) {
        this.preferCourse = preferCourse;
    }

    public String studentGetInfo(){
        return this.getId() + " | Student name: " + this.getName() + " |
            Accumulated CPA:  " + this.getAccumulatedCPA() + " | Student prefer
            course: " + Arrays.toString(this.getPreferCourse());
    }
}
```

15: Student.java

```
public class Main {
    public static void main(String[] args) {
        Student Bob = new Student();
        Bob.setId("TROY1231");
        Bob.setName("Bob Walker");
        Bob.setAccumulatedCPA(3.21);
        Bob.setPreferCourse(new String[] {"Math", "Philosophy", "Art"});

        Student Trung = new Student();
        Trung.setId("TROY1217");
        Trung.setName("Bao Trung");
        Trung.setAccumulatedCPA(2.67);
        Trung.setPreferCourse(new String[] {"Computer Science I", "Math"});

        Student Nina = new Student();
        Nina.setId("TROY1362");
        Nina.setName("Nina Gigi");
        Nina.setAccumulatedCPA(2.99);
        Nina.setPreferCourse(new String[] {"Sing", "Dance", "Physics"});

        System.out.println(Bob.studentGetInfo());
        System.out.println(Trung.studentGetInfo());
        System.out.println(Nina.studentGetInfo());
    }
}
```

16: Main.java

## 2. Implementation of TV Class

Figure 20: TV Class Task Requirement

```java
public class TV {
    private int channel;
    private int volumeLevel;
    private boolean on;

        private static final int MAX_CHANNEL = 1000;
    private static final int MIN_CHANNEL = 0;
    private static final int MAX_VOLUME = 100;
    private static final int MIN_VOLUME = 0;

    TV() {
        this.channel = MIN_CHANNEL;
        this.volumeLevel = 50;
        this.on = false;
    }

    public void turnOn() {
        this.on = true;
        System.out.println("TV turned on");
    }

    public void turnOff() {
        System.out.println("TV turning off");
        this.on = false;
    }

    private void currentVolume() {
        System.out.println("Current volume: " + this.volumeLevel);
    }

    private void currentChannel() {
        System.out.println("Channel: " + this.channel);
    }

    public void setChannel(int newChannel) {
        if(this.on && newChannel >= MIN_CHANNEL && newChannel <= MAX_CHANNEL) {
            this.channel = newChannel;
            currentChannel();
        }
    }

    public void setVolume(int newVolumeLevel) {
        if(this.on && newVolumeLevel >= MIN_VOLUME && newVolumeLevel <= MAX_VOLUME
            && newVolumeLevel % 5 == 0) {
            this.volumeLevel = newVolumeLevel;
            currentVolume();
        }
    }

    public void channelUp() {
        if(this.on && this.channel + 1 <= MAX_CHANNEL) {
            this.channel++;
            currentChannel();
```

```
        }
    }

    public void channelDown() {
        if(this.on && this.channel - 1 >= MIN_CHANNEL) {
                this.channel--;
                currentChannel();
        }
    }

    public void volumeUp() {
        if(this.on && volumeLevel + 5 <= MAX_VOLUME) {
                this.volumeLevel += 5;
                currentVolume();
        }
    }

    public void volumeDown() {
        if (this.on && volumeLevel - 5 >= MIN_VOLUME) {
                this.volumeLevel -= 5;
                currentVolume();
        }
    }
}
```

17: TV.java

```
public class Main {
    public static void main(String[] args) {
        TV SamsungTV = new TV();
        SamsungTV.setChannel(45);
        SamsungTV.setVolume(25);
        SamsungTV.volumeUp();

        SamsungTV.turnOn();
        SamsungTV.setVolume(25);
        SamsungTV.volumeUp();
        SamsungTV.volumeDown();
        SamsungTV.channelDown();
        SamsungTV.channelUp();
        SamsungTV.setChannel(123);
        SamsungTV.setChannel(1230);
        SamsungTV.turnOff();
    }
}
```

18: Main.java

## 6.1 Employee Class

Figure 21: Employee Class Task Requirement

```
package employeeClass;

public class Employee {
```

```
    private String name;
    private int idNumber;
    private String department;
    private String position;

    public Employee(String name, int idNumber, String department, String position)
        {
        this.name = name;
        this.idNumber = idNumber;
        this.department = department;
        this.position = position;
    }

    public Employee(String name, int idNumber) {
        this(name, idNumber, "", "");
    }

    public Employee() {
        this("", 0, "", "");
    }

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

    public void setIdNumber(int idNumber) {
        this.idNumber = idNumber;
    }

    public void setDepartment(String department) {
        this.department = department;
    }

    public void setPosition(String position) {
        this.position = position;
    }

    public String getName() {
        return name;
    }

    public int getIdNumber() {
        return idNumber;
    }

    public String getDepartment() {
        return department;
    }

    public String getPosition() {
        return position;
    }
}
```

19: Employee.java

```
package employeeClass;
```

```
public class Main {
    public static void main(String[] args) {
                Employee employee1 = new Employee("Susan Meyers", 47899, "
                    Accounting", "Vice President");
                Employee employee2 = new Employee("Mark Jones", 39119, "IT", "
                    Programmer");
                Employee employee3 = new Employee("Joy Rogers", 81774, "
                    Manufacturing", "Engineer");

                displayEmployeeInfo(employee1);
                displayEmployeeInfo(employee2);
                displayEmployeeInfo(employee3);
            }

            public static void displayEmployeeInfo(Employee employee) {
                System.out.printf(" Name: "    + employee.getName());
                System.out.printf(" | ID Number: " + employee.getIdNumber());
                System.out.printf(" | Department: " + employee.getDepartment());
                System.out.printf(" | Position: " + employee.getPosition());
                System.out.println();
        }
    }
```

20: Main.java

## 6.2 Car Class

Figure 22: Car Class Task Requirement

```
package carClass;

public class Car {
    private int yearModel;
    private String make;
    private int speed;

    public Car(int yearModel, String make) {
        this.yearModel = yearModel;
        this.make = make;
        this.speed = 0;
    }

    public int getYearModel() {
        return yearModel;
    }

    public String getMake() {
        return make;
    }

    public int getSpeed() {
        return speed;
    }

    public void accelerate() {
        speed += 10;
```

```
    }

    public void brake() {
        if (speed >= 10) {
            speed -= 10;
        } else {
            speed = 0;
        }
    }
}
```

21: Car.java

```
package carClass;

public class Main {
    public static void main(String[] args) {
        // Create a Car object
        Car car = new Car(2023, "Toyota");

        // Accelerate the car five times and display the speed after each
           acceleration
        System.out.println("Accelerating...");
        for (int i = 1; i <= 5; i++) {
            car.accelerate();
            System.out.println("Current speed after acceleration " + i + ": " +
               car.getSpeed() + " mph");
        }

        // Brake the car five times and display the speed after each brake
        System.out.println("\nBraking...");
        for (int i = 1; i <= 5; i++) {
            car.brake();
            System.out.println("Current speed after brake " + i + ": " + car.
               getSpeed() + " mph");
        }
    }

}
```

22: Main.java

## 6.3 Person Information Class

Figure 23: Person Information Class Task Requirement

```
package personInformationClass;

public class Person {
    private String name;
    private String address;
    private int age;
    private String phoneNumber;

    public Person(String name, String address, int age, String phoneNumber) {
        this.name = name;
```

```
        this.address = address;
        this.age = age;
        this.phoneNumber = phoneNumber;
    }

    public String getName() {
        return name;
    }

    public String getAddress() {
        return address;
    }

    public int getAge() {
        return age;
    }

    public String getPhoneNumber() {
        return phoneNumber;
    }

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

    public void setAddress(String address) {
        this.address = address;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public void setPhoneNumber(String phoneNumber) {
        this.phoneNumber = phoneNumber;
    }
}
```

23: Person.java

```
package personInformationClass;

public class Main {
    public static void main(String[] args) {
        Person myInfo = new Person("Your Name", "123 Main St, Hometown", 30, "
            555-1234");
        Person friendInfo = new Person("Friend Name", "456 Maple Ave, Nearby Town"
            , 25, "555-5678");
        Person familyMemberInfo = new Person("Family Member Name", "789 Oak Dr,
            Family Town", 55, "555-8765");

        displayPersonInfo(myInfo);
        displayPersonInfo(friendInfo);
        displayPersonInfo(familyMemberInfo);
    }

    public static void displayPersonInfo(Person person) {
        System.out.println("Name: " + person.getName());
        System.out.println("Address: " + person.getAddress());
```

```
17          System.out.println("Age: " + person.getAge());
18          System.out.println("Phone Number: " + person.getPhoneNumber());
19          System.out.println();
20      }
21  }
```

24: Main.java

### 6.4 RetailItem Class

Figure 24: RetailItem Class Task Requirement

```
1   package retailItem;
2
3   public class RetailItem {
4       private String description;
5       private int unitsOnHand;
6       private double price;
7
8       public RetailItem(String description, int unitsOnHand, double price) {
9           this.description = description;
10          this.unitsOnHand = unitsOnHand;
11          this.price = price;
12      }
13
14      public String getDescription() {
15          return description;
16      }
17
18      public int getUnitsOnHand() {
19          return unitsOnHand;
20      }
21
22      public double getPrice() {
23          return price;
24      }
25
26      public void setDescription(String description) {
27          this.description = description;
28      }
29
30      public void setUnitsOnHand(int unitsOnHand) {
31          this.unitsOnHand = unitsOnHand;
32      }
33
34      public void setPrice(double price) {
35          this.price = price;
36      }
37  }
```

25: RetailItem.java

```
1   package retailItem;
2
3   public class Main {
4           public static void main(String[] args) {
```

```
5               RetailItem item1 = new RetailItem("Jacket", 12, 59.95);
6               RetailItem item2 = new RetailItem("Designer Jeans", 40, 34.95);
7               RetailItem item3 = new RetailItem("Shirt", 20, 24.95);

9               displayItemInfo(item1);
10              displayItemInfo(item2);
11              displayItemInfo(item3);
12          }

14          public static void displayItemInfo(RetailItem item) {
15              System.out.println("Description: " + item.getDescription());
16              System.out.println("Units on Hand: " + item.getUnitsOnHand());
17              System.out.println("Price: $" + item.getPrice());
18              System.out.println();
19          }

21  }
```

26: Main.java