**Learner Assignment Submission Format**

**Learner Details**

* **Name: Harshith E**
* **Enrollment Number:**
* **Batch / Class:**
* **Assignment: (Bridge Course Day 4)**
* **Date of Submission: 27-06-2025**

**Problem Solving Activity 4.1**

**Program Statement**

Write a Java program to calculate the area of three rectangles. Initially, compute the areas using repeated code in the main method. Then, identify the repeated lines and refactor the program by creating a reusable function named calculate Area (int length, int width) that prints the area of a rectangle. Use this function to compute the area for all three rectangles.

Requirements:

* Use hardcoded values for the lengths and widths of three rectangles.
* In the refactored version, avoid code duplication by calling the function for each rectangle.
* The function should print the area directly.

**2. Algorithm**

Start

Define a function calculateArea(int length, int width):

* Inside the function, calculate area = length × width.
* Print the area.

In the main method:

* Call calculateArea(5, 3) for Rectangle 1.
* Call calculateArea(7, 4) for Rectangle 2.
* Call calculateArea(6, 2) for Rectangle 3.

End

**3. Pseudocode**

FUNCTION calculateArea(length, width)

SET area = length \* width

PRINT "Area of Rectangle: " + area

END FUNCTION

BEGIN

CALL calculateArea(5, 3)

CALL calculateArea(7, 4)

CALL calculateArea(6, 2)

END

**4. Program Code**

public class RectangleAreaRepeated {

public static void main(String[] args) {

int length1 = 5, width1 = 3;

int area1 = length1 \* width1;

System.out.println("Area of Rectangle 1: " + area1);

int length2 = 7, width2 = 4;

int area2 = length2 \* width2;

System.out.println("Area of Rectangle 2: " + area2);

int length3 = 6, width3 = 2;

int area3 = length3 \* width3;

System.out.println("Area of Rectangle 3: " + area3);

}

}

**Code Refactored with Function**

public class RectangleAreaFunction {

public static void main(String[] args) {

calculateArea(5, 3);

calculateArea(7, 4);

calculateArea(6, 2);

}

public static void calculateArea(int length, int width) {

int area = length \* width;

System.out.println("Area of Rectangle: " + area);

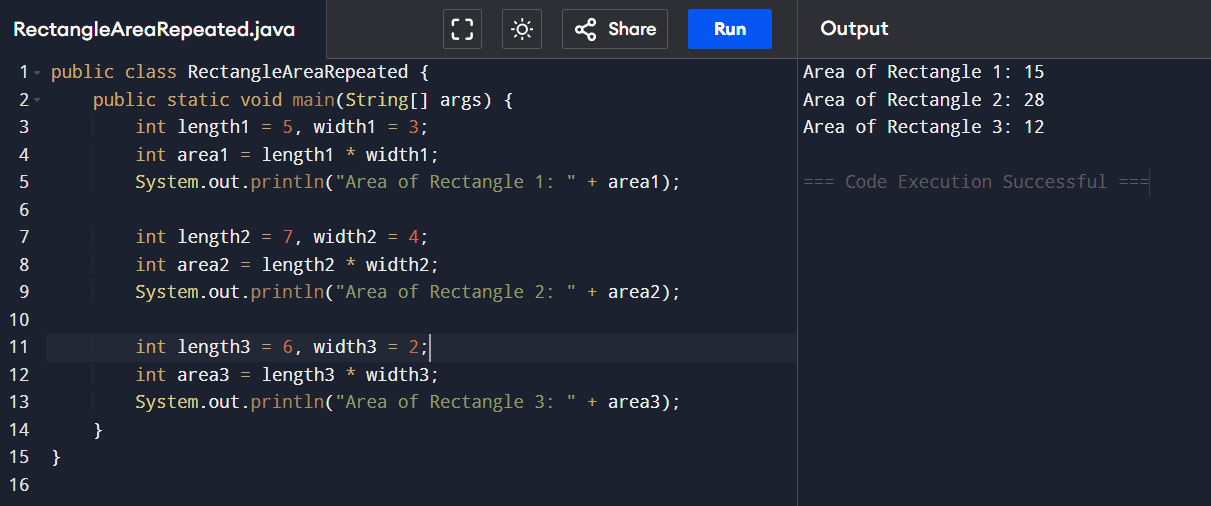
}

}

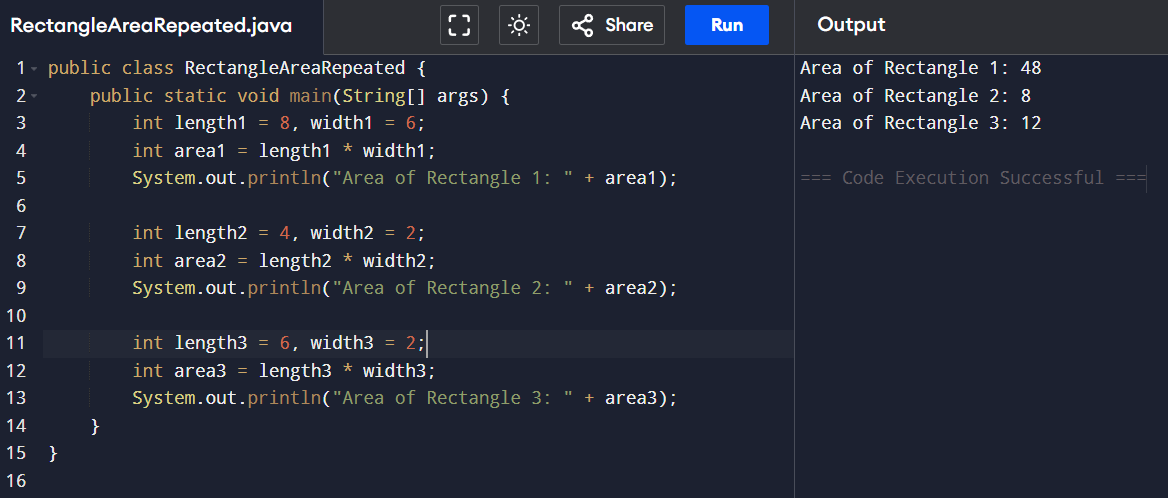
**5. Test Cases:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case No. | Input | Expected Output | Actual Output | Status (Pass/Fail) |
| 1 | 5, 3  7, 4  6, 2 | 15  28  12 | 15  28  12 | Pass |
| 2 | 6, 2  8, 4  9, 5 | 12  32  45 | 12  32  45 | Pass |
| 3 | 8, 6  4, 2  6, 2 | 45  8  12 | 45  8  12 | Pass |

**6. Screenshots of Output**

****

****

****

**7. Observation / Reflection**

* In the initial version, calculating the area of each rectangle involved **repeating the same lines of code**, which made the program longer and harder to maintain.
* The logic used for all three rectangles was **identical**, differing only in the input values (length and width).
* This repetition violates the **DRY (Don't Repeat Yourself)** principle in programming.
* By creating and using a function calculateArea(int length, int width), the code became **cleaner, more readable**, and **reusable**.
* Using functions helps in reducing redundancy and improves maintainability, especially as the number of rectangles grows.
* This exercise reinforced the importance of **modular programming** and highlighted how even a small function can greatly simplify code structure.

**Activity 4.2 Program Breakdown**

**Problem Statement:**

Create a simple Java program that simulates the basic functionalities of an ATM. The system should allow a user to perform common banking operations such as checking the account balance, depositing money, and withdrawing money.

The program should start with an initial balance (e.g., ₹1000).

Display a menu with the following options:

* Check Balance
* Deposit Money
* Withdraw Money
* Exit

**Algorithm**

Start

Initialize the account balance (e.g., balance = 1000)

Display the menu with the following options:

Check Balance

Deposit Money

Withdraw Money

Exit

Read the user's choice

Use a loop to repeat until the user chooses to exit:

* If choice is 1:

Call checkBalance() function

* Else if choice is 2:

Ask user to enter deposit amount

Call depositMoney(amount) function

* Else if choice is 3:

Ask user to enter withdrawal amount

Call withdrawMoney(amount) function

* Else if choice is 4:

Display exit message and terminate loop

* Else:

Display "Invalid choice" message

End loop

End

**Pseudocode**

BEGIN

SET balance = 1000

DO

DISPLAY "=== ATM Menu ==="

DISPLAY "1. Check Balance"

DISPLAY "2. Deposit Money"

DISPLAY "3. Withdraw Money"

DISPLAY "4. Exit"

PROMPT "Enter your choice: "

READ choice

SWITCH (choice)

CASE 1:

CALL checkBalance()

BREAK

CASE 2:

PROMPT "Enter amount to deposit: "

READ depositAmount

CALL depositMoney(depositAmount)

BREAK

CASE 3:

PROMPT "Enter amount to withdraw: "

READ withdrawAmount

CALL withdrawMoney(withdrawAmount)

BREAK

CASE 4:

DISPLAY "Thank you for using the ATM."

BREAK

DEFAULT:

DISPLAY "Invalid choice. Please try again."

WHILE choice != 4

END

**Program Code**

import java.util.Scanner;

public class SimpleATM {

static int balance = 1000; // Initial balance

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int choice;

do {

System.out.println("\n=== ATM Menu ===");

System.out.println("1. Check Balance");

System.out.println("2. Deposit Money");

System.out.println("3. Withdraw Money");

System.out.println("4. Exit");

System.out.print("Enter your choice: ");

choice = scanner.nextInt();

switch (choice) {

case 1:

checkBalance();

break;

case 2:

System.out.print("Enter amount to deposit: ");

int deposit = scanner.nextInt();

depositMoney(deposit);

break;

case 3:

System.out.print("Enter amount to withdraw: ");

int withdraw = scanner.nextInt();

withdrawMoney(withdraw);

break;

case 4:

System.out.println("Thank you for using the ATM.");

break;

default:

System.out.println("Invalid choice. Try again.");

}

} while (choice != 4);

scanner.close();

}

// Function to check balance

public static void checkBalance() {

System.out.println("Current balance: ₹" + balance);

}

// Function to deposit money

public static void depositMoney(int amount) {

if (amount > 0) {

balance += amount;

System.out.println("₹" + amount + " deposited successfully.");

} else {

System.out.println("Invalid amount.");

}

}

// Function to withdraw money

public static void withdrawMoney(int amount) {

if (amount > 0 && amount <= balance) {

balance -= amount;

System.out.println("₹" + amount + " withdrawn successfully.");

} else if (amount > balance) {

System.out.println("Insufficient balance.");

} else {

System.out.println("Invalid amount.");

}

}

}

**Test Cases**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case No. | Input | Expected Output | Actual Output | Status (Pass/Fail) |
| 1 |  |  |  |  |