Name: Sariya Mazhar

Enrollment Number: mern02

Batch / Class:

Assignment: (Bridge Course Day 4)

Date of Submission:27/06/2025

Problem Solving Activity 1

1. Program Statement: Identify Repetition

- Write a Java program that calculates the area of three rectangles using repeated code
- Then, identify the lines that are repeated.
- Create a function calculateArea(int length, int width) and call it for each rectangle.

---

2. Algorithm

Step1: Declare and initialise variables

Step2: Add method with parameter

Step3: Call the static method

Step4: Perform calculations

Step5: Result is seen.

---

3. Pseudocode

```
Start
Define method Area(w, h)
  Set res = w * h
  Print "The area of rectangle is " + res
End method:
  Call Area(13, 2)
  Call Area( 4,54)
  Call Area(2,33)
Stop
```

4. Program Code

```
public class Prog1{
   static void Area(int w,int h){
      int res=(w*h);
      System.err.println("The area of rectangle is "+res);
   }
   public static void main(String[] args) {
      Area(13,2);
      Area(4,54);
      Area(2,33);
   }
}
```

---

5. Test Cases

Test Case No.	Input	Expected Output	Actual Output	Status (Pass/Fail)
1	(13,2)	26	26	Pass
2	(4,54)	216	216	Pass
3	(2,33)	66	66	Pass

---

6. Output

```
ines/jdk-24.jdk/Contents/Home/bin/java --enable-preview
de/User/workspaceStorage/470f142dc3020e2428b4528a3fc774
The area of rectangle is 26
The area of rectangle is 216
The area of rectangle is 66
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %
```

7. Observation / Reflection

- I faced challenge in understanding normal program and with methods how it can be overcomed.I was able to identify repetition and make lines of code using method.

Problem Solving Activity 2

1. Program Statement:ATM Program

- Think of a simple ATM program with tasks like:
- Checking balance
- Depositing money
- Withdrawing money
- Define at least three functions to modularize this program.

---

2. Algorithm

Step1: Start

Step2: Set initial balance = 1000.0

Step3: Show ATM menu using a loop until user chooses to exit

Step4: Ask for user's choice

If choice is:

1: Show current balance

2: Ask for deposit amount → add to balance

3: Ask for withdrawal amount → if balance is enough, deduct it

4: Exit

Else: Show invalid choice message

Step5: End

---

3. Pseudocode

```
begin
  set balance ← 1000.0
  function checkbalance()
    print "current balance: ₹" + balance
```

```
end function
function depositmoney(amount)
  balance ← balance + amount
  print "deposited: ₹" + amount
end function
function withdrawmoney(amount)
  if amount ≤ balance then
    balance ← balance - amount
    print "withdrawn: ₹" + amount
  else
    print "insufficient balance!"
  end if
end function
do
  display atm menu
  prompt user for choice
  switch choice
    case 1: call checkbalance()
    case 2: prompt deposit amount → call depositmoney(amount)
    case 3: prompt withdraw amount → call withdrawmoney(amount)
    case 4: print "thank you for using the atm"
    default: print "invalid choice"
  end switch
while choice ≠ 4
End
```

---

4. Code:

```
import java.util.Scanner;
public class Prog2 {
   static double balance = 250000.0;
   public static void checkBalance() {
```

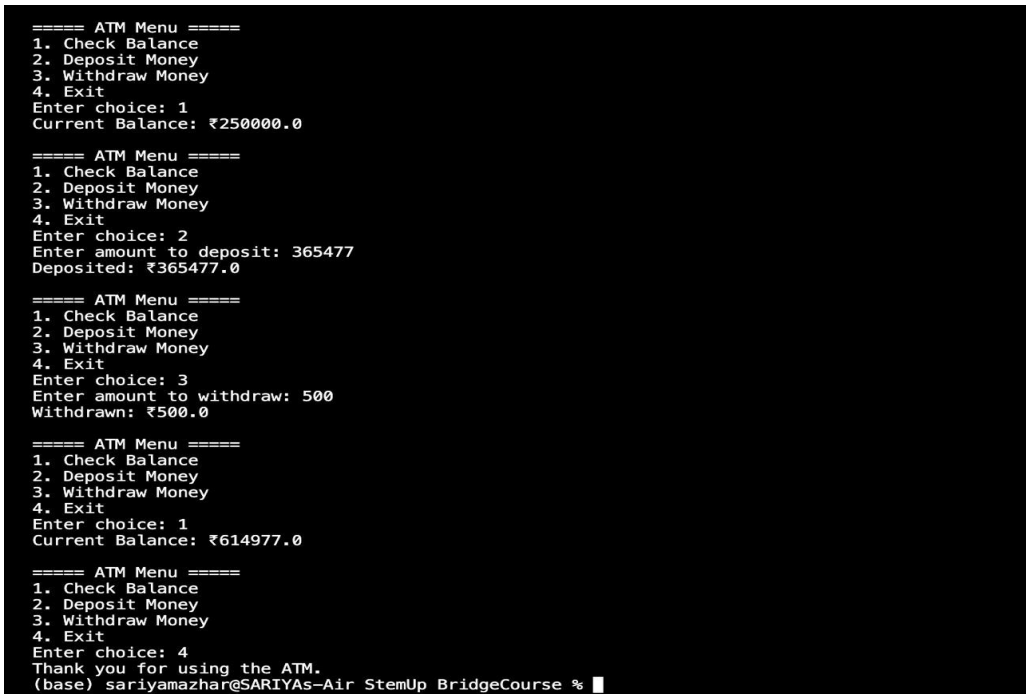
```
    System.out.println("Current Balance: ₹" + balance);
}
public static void depositMoney(double amount) {
    balance += amount;
    System.out.println("Deposited: ₹" + amount);
}
public static void withdrawMoney(double amount) {
    if (amount <= balance) {
        balance -= amount;
        System.out.println("Withdrawn: ₹" + amount);
    } else {
        System.out.println("Insufficient balance!");
    }
}
public static void main(String[] args) {
    Scanner sc = 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 choice: ");
        choice = sc.nextInt();

        switch (choice) {
            case 1:
                checkBalance();
                break;
            case 2:
```

```
                System.out.print("Enter amount to deposit: ");
                double deposit = sc.nextDouble();
                depositMoney(deposit);
                break;
            case 3:
                System.out.print("Enter amount to withdraw: ");
                double withdraw = sc.nextDouble();
                withdrawMoney(withdraw);
                break;
            case 4:
                System.out.println("Thank you for using the ATM.");
                break;
            default:
                System.out.println("Invalid choice.");
        }
    } while (choice != 4);
    sc.close();
  }
}
```

5. Test Cases

Test Case No.	Input	Expected Output	Actual Output	Status (Pass/Fail)
1	(1)	Check Balance	CB:250000	Pass
2	(2)	Deposit Money	DM:365477	Pass
3	(3)	Withdraw Money	WM:500	Pass

6. Output



---

7. Observation:

- I understood the difference in normal bank ATM using switchcase and using method with switchcase.It helped to iterate and calculate values inbuilt.

Problem Solving Activity 3

1. Program Statement: Greeting Function

- Create greetUser (String name) to greet the user.
- Call it three times with different names.

---

2. Algorithm

Step 1: Define method greetUser(name) that prints a greeting message using the given name

Step 2: In main() method, call greetUser() with name "Sariya"

Step 3: Call greetUser() with name "Sameer"

Step 4: Call greetUser() with name "Afreen"

3. Pseudocode

```
Start
Define method greetUser(name)
  Print "Hello!!! How are you " + name + " ?"
End method
  Call greetUser("Sariya")
  Call greetUser("Sameer")
  Call greetUser("Afreen")
Stop
```

---

4. Code

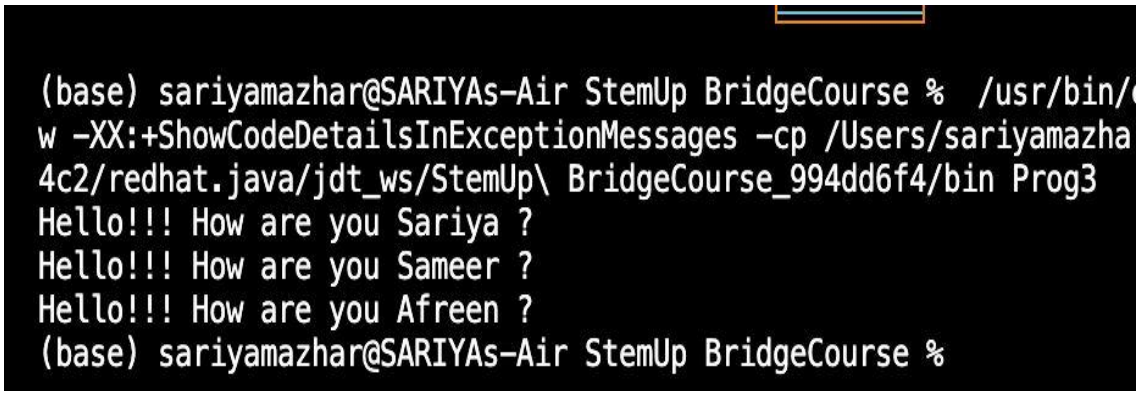
```
public class Prog3 {
   static void greetUser(String name){
      System.out.println("Hello!!! How are you "+name+" ?");
   }
   public static void main(String[] args) {
      greetUser("Sariya");
      greetUser("Sameer");
      greetUser("Afreen");
   }
}
```

---

5. Testcases

Test Case No.	Input	Expected Output	Actual Output	Status (Pass/Fail)
1	(-)	Hello!!! Sariya	Hello!!! Sariya	Pass
2	(-)	Hello!!! Sameer	Hello!!! Sameer	Pass
3	(-)	Hello!!! Afreen	Hello!!! Afreen	Pass

6. Output:



---

7. Observation

- These are the simple coding solution calling static method using Greet Method.

Problem Solving Activity 4

1. Program Statement: Calculate Square

- Create int calculateSquare(int number).
- Call it and store result in a variable, print it.
- Use return value directly in a print statement.

---

2. Algorithm

Step 1: Define method calculateSquare(num)

Step 2: Inside the method, compute res = num * num

Step 3: Print the result

Step 4: In main(), call calculateSquare(14)

Step 5: Call calculateSquare(25)

Step 6: Call calculateSquare(124)

Step 7: Stop

3. Pseudocode

Start

Define method calculateSquare(num)

Set res = num * num

Print "the square of number is: " + res

In main:

Call calculateSquare(14)

Call calculateSquare(25)

Call calculateSquare(124)

Stop

---

4. Code

```
public class Prog4 {
   public static void calculateSquare(int num){
      int res=num*num;
      System.out.println("the square of number is: "+res);
   }
   public static void main(String[] args) {
      calculateSquare(14);
      calculateSquare(25);
      calculateSquare(124);
   } }
```

---

5. Testcases

Test Case No.	Input	Expected Output	Actual Output	Status (Pass/Fail)
1	14	196	196	Pass
2	25	625	625	Pass
3	124	15376	15376	Pass

6. Output

```
de/User/workspaceStorage/470f142dc3020e2428b4528a3fc77
the square of number is: 196
the square of number is: 625
the square of number is: 15376
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %
```

---

7. Observation

- Another coding problem that uses method to call the parameters n number of time. It saves time and memory

---

Problem Solving Activity 5

1. Program Statement: Sum Two Numbers

- Create double add Numbers (double numl, double num2).
- Call it and print the sum.

---

2. Algorithm

Step 1: Start

Step 2: Declare a method addNumbers that takes two double values

Step 3: Return the sum of the two values

Step 4: In main(), declare and initialize two double variables

Step 5: Call the addNumbers() method with the two values

Step 6: Print the returned sum

Step 7: Stop

3. Pseudocode:

```
Start
  Define method addNumbers(num1, num2)
    Return num1 + num2
    Set number1 = 12.5
    Set number2 = 7.3
    result = call addNumbers(number1, number2)
    Print "The sum is: " + result
Stop
```

---

4. Code:

```
public class Prog5 {
    public static double addNumbers(double n1, double n2) {
        return n1 + n2;
    }
    public static void main(String[] args) {
        double n1 = 12.5;
        double n2 = 7.3;
        double sum = addNumbers(n1, n2);
        System.out.println("The sum is: " + sum);
    }
}
```

---

5. Test Case:

Test Case No.	Input	Expected Output	Actual Output	Status (Pass/Fail)
1	(5.3, 3.5)	8.8	8.8	Pass
2	(-32.2, 5)	-27.2	-27.2	Pass
3	(0, 0.3)	0.3	0.3	Pass

6. Output

```
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %  /usr/bi
w -XX:+ShowCodeDetailsInExceptionMessages -cp /Users/sariyamaz
4c2/redhat.java/jdt_ws/StemUp\ BridgeCourse_994dd6f4/bin Prog5
The sum is: 8.8
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %  cd /Use
ines/jdk-24.jdk/Contents/Home/bin/java --enable-preview -XX:+S
de/User/workspaceStorage/470f142dc3020e2428b4528a3fc774c2/redh
The sum is: -27.2
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %  cd /Use
ines/jdk-24.jdk/Contents/Home/bin/java --enable-preview -XX:+S
de/User/workspaceStorage/470f142dc3020e2428b4528a3fc774c2/redh
The sum is: 0.3
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %
```

---

7. Observation

- If the input is either positive or negative it displays the answer. I have given all the testcases, positive, negative and also with 0. It works seamless.

Problem Solving Activity 6

1. Program Statement:Temperature Converter

- Create double celsius To Fahrenheit (double celsius).
- Create double fahrenheit To Celsius (double fahrenheit).
- Test with sample values.

---

2. Algorithm

Step 1: Start

Step 2: Define method celsiusToFahrenheit(c)

Step 3: Inside it, compute Fahrenheit as (c × 1.8) + 32 and return it

Step 4: Define method fahrenheitToCelsius(F)

Step 5: Inside it, compute Celsius as (F − 32) / 1.8 and return it

Step 6: In main(), call celsiusToFahrenheit(32) and store result in variable a

Step 7: Call fahrenheitToCelsius(193.0) and store result in variable b

Step 8: Print both a and b

---

3. Pseudocode

```
Start
Define method celsiusToFahrenheit(c)
  Set Fahrenheit = (c * 1.8) + 32
  Return Fahrenheit
End method
Define method fahrenheitToCelsius(F)
  Set Celsius = (F - 32) / 1.8
  Return Celsius
End method
In main:
  Set a = call celsiusToFahrenheit(32)
  Set b = call fahrenheitToCelsius(193.0)
  Print "The Fahrenheit is " + a
  Print "The Celsius is " + b
Stop
```

---

4. Code:

```
public class Prog6 {
   static double celsiusToFahrenheit(double c){
      double Fahrenheit=(c*1.8)+32;//F = (°C × 1.8) + 32
      return Fahrenheit;
   }
   static double fahrenheitToCelsius(double F){
       double Celsius=(F - 32) / 1.8;//C = (°F - 32) / 1.8
       return Celsius;
   }
   public static void main(String[]args){
```

```
        double a=celsiusToFahrenheit(32);
        double b=fahrenheitToCelsius(193.0);
        System.out.println("The Fahrenheit is "+a);
         System.out.println("The Celsius is "+b);
    }
}
```

5. Test Cases

Test Case No.	Input	Expected Output	Actual Output	Status (Pass/Fail)
1	(32, 193)	(89.6,89.44)	(89.6,89.44)	Pass
2	(-32.2, 153)	(-25.96, 67.22)	(-25.96, 67.22)	Pass
3	(0, 0.3)	(32.0,-17.66)	(32.0,-17.66)	Pass

6. Output

```
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %  /usr/b
w -XX:+ShowCodeDetailsInExceptionMessages -cp /Users/sariyama
4c2/redhat.java/jdt_ws/StemUp\ BridgeCourse_994dd6f4/bin Prog
The Fahrenheit is 89.6
The Celsius is 89.44444444444444
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %  cd /Us
ines/jdk-24.jdk/Contents/Home/bin/java --enable-preview -XX:-
de/User/workspaceStorage/470f142dc3020e2428b4528a3fc774c2/red
The Fahrenheit is -25.960000000000008
The Celsius is 67.22222222222221
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %  cd /Us
ines/jdk-24.jdk/Contents/Home/bin/java --enable-preview -XX:-
de/User/workspaceStorage/470f142dc3020e2428b4528a3fc774c2/red
The Fahrenheit is 32.0
The Celsius is -17.61111111111111
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %
```

7. Observation

- This coding problem is tested with all the possible temperatures.I have checked with -degrees as well

Problem Solving Activity 7

1. Program Statement: Scope Experiment

---

2. Issue in code:

```
public class ScopeTest {
static String globalMessage = "I am global!";
static void displayMessages() {
}
String localMessage = "I am local!";
System.out.println(globalMessage);
public static void main(String[] args) {
displayMessages();
// Try to print localMessage here and observe the error.
}
}
```

---

3. Fixed Code:

```
public class ScopeTest {
    static String globalMessage = "I am global!";
    static void displayMessages() {
        String localMessage = "I am local!";
        System.out.println(globalMessage);     //  global variable is valid
        System.out.println(localMessage);      //  local variable is valid
    }
    public static void main(String[] args) {
        displayMessages();
        // Trying to access localMessage here will cause a compilation error:
        // System.out.println(localMessage);   ✕ Error
    } }
```

4. Observation:

- We cannot print the localMessage as it is a variable and you can't print variables directly. localMessage in main() will result in a compilation error, because localMessage is not in scope there**.**

Problem Solving Activity 8

1. Program Statement: Price Calculator (Function Composition)

- double calculateDiscount (double originalPrice, double discountPercentage);
- double calculate Tax (double amount, double taxRate);
- double calculate Final Price (double itemPrice, double discountPerc, double taxRate);
- Call and print the result.

---

2. Algorithm

Step 1: Start

Step 2: Set itemPrice = 1000

Step 3: Set discountPerc = 10

Step 4: Set taxRate = 5

Step 5: Call calculateDiscount(itemPrice, discountPerc)

Step 6: Get discount amount

Step 7: Subtract discount from itemPrice to get priceAfterDiscount

Call calculateTax(priceAfterDiscount, taxRate)

Get tax amount

Add tax to priceAfterDiscount → finalPrice

Step 8: Display finalPrice

---

3. Pseudocode:

```
begin
  set itemprice ← 1000
```

```
set discountperc ← 10
set taxrate ← 5
function calculatediscount(originalprice, discountpercentage)
  return originalprice × (discountpercentage / 100)
end function
function calculatetax(amount, taxrate)
  return amount × (taxrate / 100)
end function
function calculatefinalprice(itemprice, discountperc, taxrate)
  set discount ← calculatediscount(itemprice, discountperc)
  set priceafterdiscount ← itemprice − discount
  set tax ← calculatetax(priceafterdiscount, taxrate)
  return priceafterdiscount + tax
end function
call calculatefinalprice(itemprice, discountperc, taxrate)
display "final price: rs ", finalprice
End
```

---

4. Code:

```
public class Prog8{
   public static double calculateDiscount(double originalPrice, double discountPercentage) {
      return originalPrice * (discountPercentage / 100);
   }
   public static double calculateTax(double amount, double taxRate) {
      return amount * (taxRate / 100);
   }
   public static double calculateFinalPrice(double itemPrice, double discountPerc, double taxRate) {
      double discount = calculateDiscount(itemPrice, discountPerc);
      double priceAfterDiscount = itemPrice - discount;
      double tax = calculateTax(priceAfterDiscount, taxRate);
      return priceAfterDiscount + tax;
```

```
    }
    public static void main(String[] args) {

        double finalPrice = calculateFinalPrice(1000.0, 10.0, 5.0);
        System.out.println("Final Price: Rs " + finalPrice);
    }
}
```

5. Test Cases

Test Case No.	Input	Expected Output	Actual Output	Status (Pass/Fail)
1	(1000,5% D, 5% tax)	(945)	(945)	Pass
2	(5999,5% D, 5%tax)	(5885.25)	(5885.25)	Pass
3	(7690,12%D,8%tax)	(7308.576)	(7308.576)	Pass

6. Output

```
 /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-24.jdk/Cont
ariyamazhar/Library/Application\ Support/Code/User/workspaceSto
in Prog8
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %  /usr/bin
w -XX:+ShowCodeDetailsInExceptionMessages -cp /Users/sariyamazh
4c2/redhat.java/jdt_ws/StemUp\ BridgeCourse_994dd6f4/bin Prog8
Final Price: Rs 945.0
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %  cd /User
ines/jdk-24.jdk/Contents/Home/bin/java --enable-preview -XX:+Sh
de/User/workspaceStorage/470f142dc3020e2428b4528a3fc774c2/redha
Final Price: Rs 5885.25
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %  cd /User
ines/jdk-24.jdk/Contents/Home/bin/java --enable-preview -XX:+Sh
de/User/workspaceStorage/470f142dc3020e2428b4528a3fc774c2/redha
Final Price: Rs 7308.576
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %
```

7. Observation

- Code Readability,Clearly structured and menu-driven interface Static Variablebalance is static.

Problem Solving Activity 9

1. Program Statement: Refactor Repetitive Code

- Refactor my Day 2 calculator to use add, subtract, multiply,divide seamlessly:

---

2. Algorithm

Step 1: Start

Step 2: Define add(num1, num2) to return sum of num1 and num2

Step 3: Define subtract(num1, num2) to return difference of num1 and num2

Step 4: Define multiply(num1, num2) to return product of num1 and num2

Step 5: Define divide(num1, num2) to return division result if num2 is not zero

Step 6: In main(), read two numbers from the user

Step 7: Read the arithmetic operator from the user

Step 8: Use switch-case to call the corresponding function based on the operator

Step 9: Display the result

Step 10: End

---

3. Pseudocode

```
Start
Define function add(num1, num2)
  return num1 + num2
End function
Define function subtract(num1, num2)
  return num1 - num2
End function
Define function multiply(num1, num2)
  return num1 * num2
End function
Define function divide(num1, num2)
  if num2 == 0
```

```
    print "Error: Cannot divide by zero"
    return 0
  else
    return num1 / num2
End function
In main:
  Read num1
  Read num2
  Read operator
  switch(operator)
    case '+':
      result = add(num1, num2)
    case '-':
      result = subtract(num1, num2)
    case '*':
      result = multiply(num1, num2)
    case '/':
      result = divide(num1, num2)
    default:
      print "Invalid operator"
  Print result if operator was valid
Stop
```

---

4. Code:

```
import java.util.Scanner;
public class Prog9 {
   public static double add(double num1, double num2) {
      return num1 + num2;
   }
   public static double subtract(double num1, double num2) {
      return num1 - num2;
```

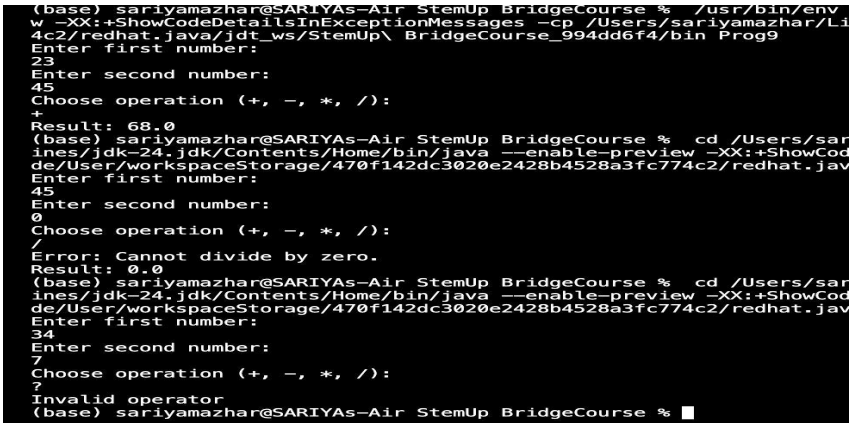
```
}
public static double multiply(double num1, double num2) {
    return num1 * num2;
}
public static double divide(double num1, double num2) {
    if (num2 == 0) {
        System.out.println("Error: Cannot divide by zero.");
        return 0;
    }
    return num1 / num2;
}
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter first number:");
    double num1 = sc.nextDouble();
    System.out.println("Enter second number:");
    double num2 = sc.nextDouble();
    System.out.println("Choose operation (+, -, *, /):");
    char op = sc.next().charAt(0);
    double result = 0;
    boolean valid = true;

    switch (op) {
        case '+':
            result = add(num1, num2);
            break;
        case '-':
            result = subtract(num1, num2);
            break;
        case '*':
            result = multiply(num1, num2);
```

```
            break;
        case '/':
            result = divide(num1, num2);
            break;
        default:
            System.out.println("Invalid operator");
            valid = false;
    }
    if (valid) {
        System.out.println("Result: " + result);
    }
  }
}
```

5. Testcase

Test Case No.	Input	Expected Output	Actual Output	Status (Pass/Fail)
1	(23,45,+)	(68.0)	(68.0)	Pass
2	(45,0,/)	(Error!)	(Error!)	Pass
3	(34,7,?)	(Invalid operator)	(Invalid operator)	Pass

6. Output



7. Observation

- Each operation is handled in separate functions, The program became easier to read, maintain, and scale with better error handling.

Problem Solving Activity 10

1. **Program Statement:Customizable Greeting (Overloading)**.

- I want to create void customGreet(String name, String greeting), void customGreet(String name) and call functions individually.

---

2. Algorithm

Step 1: Start

Step 2: Define method customGreet(String name, String greeting) to print greeting with name

Step 3: Define method customGreet(String name) to print "Hello" with name

Step 4: Define method customGreet() to print "Hello, Guest!"

Step 5: In main(), call customGreet("Sariya", "Good Morning")

Step 6: Call customGreet("Sariya")

Step 7: Call customGreet()

Step 8: Stop

---

3. **Pseudocode**

```
Define method customGreet(name, greeting)
  Print greeting + ", " + name + "!"
Define method customGreet(name)
  Print "Hello, " + name + "!"
Define method customGreet()
  Print "Hello, Guest!":
  Call customGreet("Sariya", "Good Morning")
  Call customGreet("Sariya")
  Call customGreet()
Stop
```

4. Code

```
public class Prog10 {
    public static void customGreet(String name, String greeting) {
        System.out.println(greeting + ", " + name + "!");
    }
    public static void customGreet(String name) {
        System.out.println("Hello, " + name + "!");
    }
    public static void customGreet() {
        System.out.println("Hello, Guest!");
    }
    public static void main(String[] args) {
        customGreet("Sariya", "Good Morning");
        customGreet("Sariya");
        customGreet();
    }
}
```

---

5. Testcase

Test Case No.	Input	Expected Output	Actual Output	Status (Pass/Fail)
1	Sariya	GM, Sariya	GM, Sariya	Pass
2	Bruno	GM,Bruno	GM,Bruno	Pass
3	Virat	GM,Virat	GM,Virat	Pass

6. Output

```
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %  /usr/bin/env
w -XX:+ShowCodeDetailsInExceptionMessages -cp /Users/sariyamazhar/Li
4c2/redhat.java/jdt_ws/StemUp\ BridgeCourse_994dd6f4/bin Prog10
Good Morning, Sariya!
Hello, Sariya!
Hello, Guest!
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %  cd /Users/sar
ines/jdk-24.jdk/Contents/Home/bin/java --enable-preview -XX:+ShowCod
de/User/workspaceStorage/470f142dc3020e2428b4528a3fc774c2/redhat.jav
Good Morning, Bruno!
Hello, Bruno!
Hello, Guest!
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %  cd /Users/sar
ines/jdk-24.jdk/Contents/Home/bin/java --enable-preview -XX:+ShowCod
de/User/workspaceStorage/470f142dc3020e2428b4528a3fc774c2/redhat.jav
Good Morning, Virat!
Hello, Virat!
Hello, Guest!
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %
```

---

7. Observation

- This is a very easy program where we need to call a function greeting the individual.

Problem Solving Activity 11

1. Program Statement: Power Calculator

- To Write myPower(int base, int exponent) using a loop and Compare with Math.pow(base, exponent).

---

2. Algorithm

Step 1: Start

Step 2: Define method myPower(base, exponent)

Step 3: Initialize result as 1

Step 4: Repeat loop from i = 1 to exponent

Step 5: Multiply result by base in each iteration

Step 6: Return result

Step 7: In main(), call myPower() and Math.pow() with same inputs

Step 8: Print both results

3. Pseudocode

```
start
define method mypower(base, exponent)
  set result = 1
  repeat i = 1 to exponent
    result = result * base
  end repeat
  return result
in main:
  set base = 2
  set exponent = 5
  customresult = call mypower(base, exponent)
  mathresult = call math.pow(base, exponent)
  print "result using mypower(): " + customresult
  print "result using math.pow(): " + mathresult
Stop
```

---

4. Code:

```
public class Prog11 {
   public static int myPower(int base, int exponent) {
      int result = 1;
      for (int i = 1; i <= exponent; i++) {
         result *= base;
      }
      return result;
   }
   public static void main(String[] args) {
      int base = 2;
```

```
        int exponent = 5;
        int customResult = myPower(base, exponent);
        double mathResult = Math.pow(base, exponent);
        System.out.println("Result using myPower(): " + customResult);
        System.out.println("Result using Math.pow(): " + mathResult);
    }
}
```

5. Testcase

Test Case No.	Input	Expected Output	Actual Output	Status (Pass/Fail)
1	(2,5)	32.0	32.0	Pass
2	(5,2)	25.0	25.0	Pass
3	(67,2)	4489	4489	Pass

6. Output

```
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %  /usr/bin/env /Libra
w -XX:+ShowCodeDetailsInExceptionMessages -cp /Users/sariyamazhar/Library/
4c2/redhat.java/jdt_ws/StemUp\ BridgeCourse_994dd6f4/bin Prog11
Result using myPower(): 32
Result using Math.pow(): 32.0
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %  cd /Users/sariyamaz
ines/jdk-24.jdk/Contents/Home/bin/java --enable-preview -XX:+ShowCodeDetai
de/User/workspaceStorage/470f142dc3020e2428b4528a3fc774c2/redhat.java/jdt_
Result using myPower(): 25
Result using Math.pow(): 25.0
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %  cd /Users/sariyamaz
ines/jdk-24.jdk/Contents/Home/bin/java --enable-preview -XX:+ShowCodeDetai
de/User/workspaceStorage/470f142dc3020e2428b4528a3fc774c2/redhat.java/jdt_
Result using myPower(): 4489
Result using Math.pow(): 4489.0
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %
```

7. Observation

- This program helped me to calculate exponential values easily using math.pow function.

Problem Solving Activity 12

1. Program Statement: Trace the Flow

- To trace the flow how the calling function work upon one another.

---

2. Algorithm

Step 1: Start

Step 2: Define function B() to compute and return a value

Step 3: Define function C(value) to print the value

Step 4: Define function A()

Step 5: In A(), call B() and store its return value

Step 6: Call C() using the result from B()

Step 7: In main(), call A() to begin flow

Step 8: Stop

---

3. Pseudocode

```
Start
Define function C(value)
  Print "Final Result: " + value
End function
Define function B()
  Set result = 10 + 5
  Return result
End function
Define function A()
  Set bResult = call B()
  Call C(bResult)
End function
  Call A()
Stop
```

4. Code:

```
public class Prog12 {
    public static void C(int value) {
        System.out.println("Final Result: " + value);
    }
    public static int B() {
        int result = 10 + 5;
        return result;
    }
    public static void A() {
        int bResult = B();
        C(bResult);
    }
    public static void main(String[] args) {
        A();
    }
}
```

---

5. Testcase

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

6. Output:

```
 /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-24.jdk/Conte
ariyamazhar/Library/Application\ Support/Code/User/workspaceStor
in Prog12
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %  /usr/bin/
w -XX:+ShowCodeDetailsInExceptionMessages -cp /Users/sariyamazha
4c2/redhat.java/jdt_ws/StemUp\ BridgeCourse_994dd6f4/bin Prog12
Final Result: 15
(base) sariyamazhar@SARIYAs-Air StemUp BridgeCourse %
```

---

7. Observation

- It explains how functions interact in a call sequence. Function A coordinates the flow by calling B and passing its result to C.