



LAB REPORT

Course Code: CSE-222

Course Title: Object Oriented Programming Lab

Experiment Name: Various Types of Problem & Solution On Java

Submitted to -

Name: MD. ASHAF UDDAULA

Designation: Lecturer

Department of Computer Science & Engineering

Daffodil International University

Submitted by -

Name: NAHIDUL ISLAM PRANTO

ID: 0242310005101939

Section: 64_F_2

Department of Computer Science & Engineering

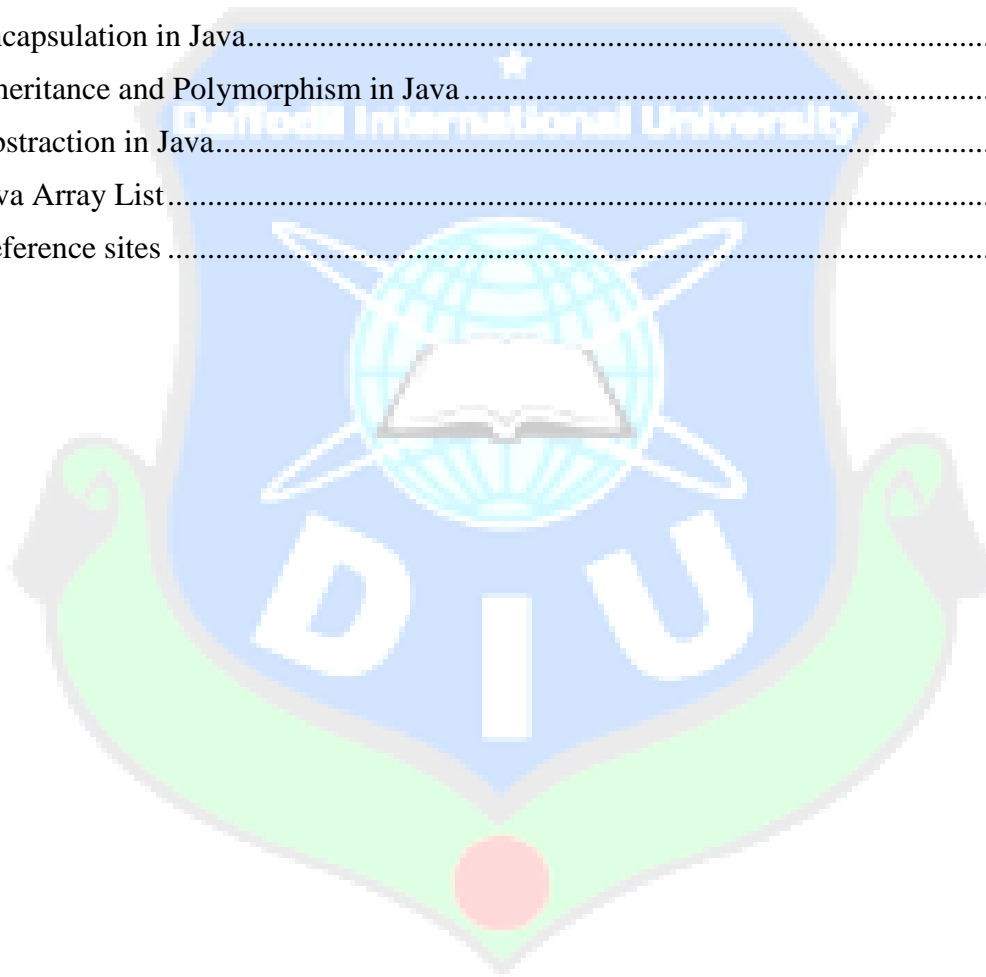
Daffodil International University

| |
|-------------------------------------|
| Submission Date: 27-Nov-2024 |
|-------------------------------------|

Table of Contents

| | |
|---|----|
| 1. Java Output / Print | 4 |
| 2. Java Variables: String, int, float, char, Boolean | 4 |
| 3. Problem: Calculate the Area of a Rectangle | 5 |
| 4. Arithmetic Operators | 6 |
| 4.1 Arithmetic Operators | 6 |
| 4.2 Assignment operators: | 7 |
| 4.3 Comparison operators: | 8 |
| 4.4 Logical operators: | 9 |
| 4.5 Bitwise operators: | 10 |
| 5. Java String Methods: | 11 |
| 6. Java Math | 14 |
| 7. Java Conditions Statements | 16 |
| 7.1 if, else, else if | 16 |
| 7.2 switch-case | 16 |
| 8. Java Loop: | 18 |
| 8.1 For-Loop: | 18 |
| 8.2 While Loop: | 18 |
| 8.3 Do-While | 19 |
| 9. Java Break and Continue | 20 |
| 10. Java Array: 1D Array, 2D Array | 21 |
| 11. Java User Input (Scanner): nextLine(), nextBoolean(), nextByte(), nextDouble(), nextFloat(), nextInt(), nextLong(), nextShort() | 23 |
| 12. Java Methods | 25 |
| 12.1 Method with no arguments and no return value | 25 |
| 12.2 Method with arguments but no return value | 25 |
| 12.3 Method with no arguments but with return value | 26 |
| 12.4 Method with arguments and return value | 27 |
| 13. Java Method Overloading | 28 |
| 14. Problem Solving with User Input | 29 |
| 14.1 Check Leap Year | 29 |
| 14.2 BMI Calculator | 30 |
| 14.3 Calculate the Area of Triangle | 31 |

| | |
|--|----|
| 14.4 Calculate Factorial..... | 32 |
| 14.5 Counting Vowels in a String | 32 |
| 14.6 Prime Factorization | 33 |
| 14.7 Reverse a Number | 34 |
| 14.8 Fibonacci Series: | 35 |
| 14.9 Sum of Natural Numbers (1 to 100)..... | 36 |
| 15. Java Constructors & Constructor Overloading..... | 37 |
| 16. Encapsulation in Java..... | 39 |
| 17. Inheritance and Polymorphism in Java..... | 41 |
| 18. Abstraction in Java..... | 43 |
| 19. Java Array List..... | 45 |
| 20. Reference sites | 47 |

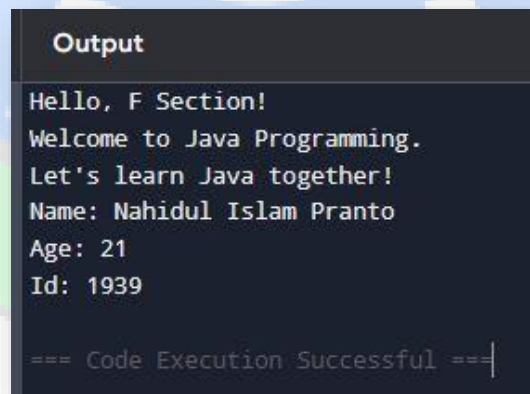


1. Java Output / Print

Here is a Java code example where we can find and print the output -

```
1. public class Main {  
2.     public static void main(String[] args) {  
3.         System.out.print("Hello, ");  
4.         System.out.print("F Section!\n");  
5.         System.out.println("Welcome to Java Programming.");  
6.         System.out.println("Let's learn Java together!");  
7.         String name = "Nahidul Islam Pranto";  
8.         int id = 1939;  
9.         int age = 21;  
10.        System.out.println("Name: " + name);  
11.        System.out.println("Age: " + age);  
12.        System.out.println("Id: " + id);  
13.    }  
14. }
```

Here is the output of this code:

A screenshot of a terminal window showing the output of the Java code. The output is displayed in a dark-themed window with a title bar that says "Output". The text inside the window is: "Hello, F Section!", "Welcome to Java Programming.", "Let's learn Java together!", "Name: Nahidul Islam Pranto", "Age: 21", "Id: 1939", and "=== Code Execution Successful ===".

```
Output  
Hello, F Section!  
Welcome to Java Programming.  
Let's learn Java together!  
Name: Nahidul Islam Pranto  
Age: 21  
Id: 1939  
=== Code Execution Successful ===
```

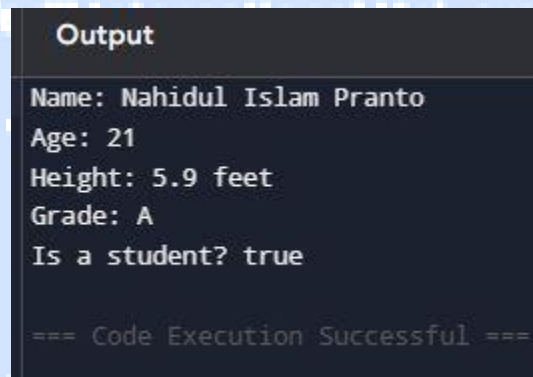
2. Java Variables: String, int, float, char, Boolean

Here is a Java code example where we can find various types of variables –

```
1. public class Main {  
2.     public static void main(String[] args) {  
3.         String name = "Nahidul Islam Pranto";  
4.         int age = 21;  
5.         float height = 5.9f;
```

6. char grade = 'A';
7. boolean isStudent = true;
8. System.out.println("Name: " + name);
9. System.out.println("Age: " + age);
10. System.out.println("Height: " + height + " feet");
11. System.out.println("Grade: " + grade);
12. System.out.println("Is a student? " + isStudent);
13. }
14. }

Here is the output of this code:



```

Output
Name: Nahidul Islam Pranto
Age: 21
Height: 5.9 feet
Grade: A
Is a student? true

=== Code Execution Successful ===

```

3.Problem: Calculate the Area of a Rectangle

Here's a Java program to calculate the area of a rectangle. The formula for the area of a rectangle is:

$$\text{Area} = \text{Length} \times \text{Width}$$

Here is the code:

1. import java.util.Scanner;
2. public class Main {
3. public static void main(String[] args) {
4. Scanner input = new Scanner(System.in);
5. System.out.print("Enter the length of the rectangle: ");
6. double length = input.nextDouble();
7. System.out.print("Enter the width of the rectangle: ");
8. double width = input.nextDouble();
9. double area = length * width;
10. System.out.println("The area of the rectangle is: " + area);
11. input.close();
12. }

13. }

Here is the output of this code:

```
Output
Enter the length of the rectangle: 10
Enter the width of the rectangle: 45
The area of the rectangle is: 450.0

=== Code Execution Successful ===
```

4. Arithmetic Operators

4.1 Arithmetic Operators

Here's a simple Java program that uses only arithmetic operators (addition, subtraction, multiplication, division, and modulus) to calculate the area of a rectangle and perform a few arithmetic operations:

```
1. public class Main {
2.     public static void main(String[] args) {
3.         double length = 5.0;
4.         double width = 3.0;
5.         double area = length * width;
6.         double sum = length + width;
7.         double difference = length - width;
8.         double product = length * width;
9.         double quotient = length / width;
10.        double remainder = length % width;
11.        System.out.println("Length: " + length);
12.        System.out.println("Width: " + width);
13.        System.out.println("Area (Length * Width): " + area);
14.        System.out.println("Sum (Length + Width): " + sum);
15.        System.out.println("Difference (Length - Width): " + difference);
16.        System.out.println("Product (Length * Width): " + product);
17.        System.out.println("Quotient (Length / Width): " + quotient);
18.        System.out.println("Remainder (Length % Width): " + remainder);
19.    }
20. }
```

Here is the output of this code:

```
Output
Length: 5.0
Width: 3.0
Area (Length * Width): 15.0
Sum (Length + Width): 8.0
Difference (Length - Width): 2.0
Product (Length * Width): 15.0
Quotient (Length / Width): 1.6666666666666667
Remainder (Length % Width): 2.0

=== Code Execution Successful ===
```

4.2 Assignment operators:

Here's a Java program that demonstrates the use of assignment operators (=, +=, -=, *=, /=, and %=) to perform calculations.

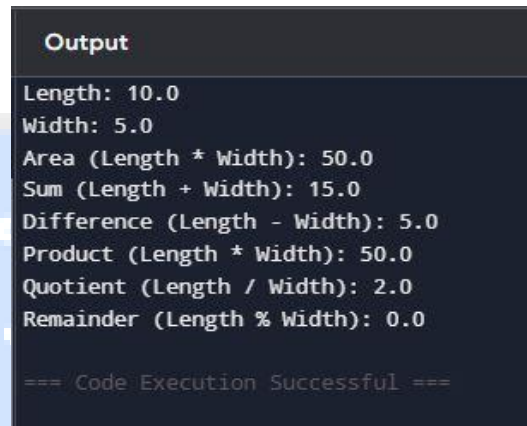
```
1. public class Main {
2.     public static void main(String[] args) {
3.         double length = 10.0;
4.         double width = 5.0;
5.         double area = 0.0;
6.         area = length;
7.         area *= width;
8.         double sum = 0.0;
9.         sum += length;
10.        sum += width;
11.        double difference = length;
12.        difference -= width;
13.        double product = 1.0;
14.        product *= length;
15.        product *= width;
16.        double quotient = length;
17.        quotient /= width;
18.        double remainder = length;
19.        remainder %= width;
20.        System.out.println("Length: " + length);
21.        System.out.println("Width: " + width);
22.        System.out.println("Area (Length * Width): " + area);
23.        System.out.println("Sum (Length + Width): " + sum);
24.        System.out.println("Difference (Length - Width): " + difference);
25.        System.out.println("Product (Length * Width): " + product);
```

```

26. System.out.println("Quotient (Length / Width): " + quotient);
27. System.out.println("Remainder (Length % Width): " + remainder);
28. }
29. }

```

Here is the output of this code:



```

Output
Length: 10.0
Width: 5.0
Area (Length * Width): 50.0
Sum (Length + Width): 15.0
Difference (Length - Width): 5.0
Product (Length * Width): 50.0
Quotient (Length / Width): 2.0
Remainder (Length % Width): 0.0

=== Code Execution Successful ===

```

4.3 Comparison operators:

Here's a Java program that uses comparison operators (==, !=, >, <, >=, and <=) to compare two numbers provided by the user:

```

1. import java.util.Scanner;
2. public class Main {
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         System.out.print("Enter the first number: ");
6.         double num1 = input.nextDouble();
7.         System.out.print("Enter the second number: ");
8.         double num2 = input.nextDouble();
9.         System.out.println("Comparing " + num1 + " and " + num2 + ":");
10.        System.out.println("Are the numbers equal? " + (num1 == num2));
11.        System.out.println("Are the numbers not equal? " + (num1 != num2));
12.        System.out.println("Is the first number greater than the second? " + (num1 > num2));
13.        System.out.println("Is the first number less than the second? " + (num1 < num2));
14.        System.out.println("Is the first number greater than or equal to the second? " + (num1 >=
            num2));
15.        System.out.println("Is the first number less than or equal to the second? " + (num1 <=
            num2));
16.        input.close();
17.    }
18. }

```


Here is the output of this code:

```
Output
Enter the first number: 21
Enter the second number: 52
Comparing 21.0 and 52.0:
Are the numbers equal? false
Are the numbers not equal? true
Is the first number greater than the second? false
Is the first number less than the second? true
Is the first number greater than or equal to the second? false
Is the first number less than or equal to the second? true

=== Code Execution Successful ===
```

4.4 Logical operators:

Here's a Java program that uses logical operators (&&, ||, !) to evaluate user inputs and display results based on logical conditions:

```
1. import java.util.Scanner;
2. public class Main {
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         System.out.print("Enter the first boolean value (true/false): ");
6.         boolean value1 = input.nextBoolean();
7.         System.out.print("Enter the second boolean value (true/false): ");
8.         boolean value2 = input.nextBoolean();
9.         System.out.println("Logical Operations Results:");
10.        System.out.println("Logical AND (value1 && value2): " + (value1 && value2));
11.        System.out.println("Logical OR (value1 || value2): " + (value1 || value2));
12.        System.out.println("Logical NOT (!value1): " + (!value1));
13.        System.out.println("Logical NOT (!value2): " + (!value2));
14.        input.close(); // Close the scanner
15.    }
16. }
```

Here is the output of this code:

```
Output
Enter the first boolean value (true/false): true
Enter the second boolean value (true/false): true
Logical Operations Results:
Logical AND (value1 && value2): true
Logical OR (value1 || value2): true
Logical NOT (!value1): false
Logical NOT (!value2): false

=== Code Execution Successful ===
```

4.5 Bitwise operators:

Here's a Java program that uses bitwise operators (&, |, ^, ~, <<, >>, >>>) to perform operations on integers provided by the user.

```
1. import java.util.Scanner;
2. public class Main {
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         System.out.print("Enter the first integer: ");
6.         int num1 = input.nextInt();
7.         System.out.print("Enter the second integer: ");
8.         int num2 = input.nextInt();
9.         System.out.println("Bitwise Operations Results:");
10.        System.out.println("Bitwise AND (num1 & num2): " + (num1 & num2));
11.        System.out.println("Bitwise OR (num1 | num2): " + (num1 | num2));
12.        System.out.println("Bitwise XOR (num1 ^ num2): " + (num1 ^ num2));
13.        System.out.println("Bitwise Complement (~num1): " + (~num1));
14.        System.out.println("Left Shift (num1 << 2): " + (num1 << 2));
15.        System.out.println("Right Shift (num1 >> 2): " + (num1 >> 2));
16.        System.out.println("Unsigned Right Shift (num1 >>> 2): " + (num1 >>> 2));
17.        input.close();
18.    }
19. }
```

Here is the output of this code:

```
Output
Enter the first integer: 5
Enter the second integer: 3
Bitwise Operations Results:
Bitwise AND (num1 & num2): 1
Bitwise OR (num1 | num2): 7
Bitwise XOR (num1 ^ num2): 6
Bitwise Complement (~num1): -6
Left Shift (num1 << 2): 20
Right Shift (num1 >> 2): 1
Unsigned Right Shift (num1 > 2): 1

=== Code Execution Successful ===
```

5. Java String Methods:

Here's a comprehensive Java program demonstrating all the specified methods with user input. Each method is applied to showcase its functionality.

```
1. import java.util.Scanner;
2. public class Main {
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         // Take user inputs
6.         System.out.print("Enter a string: ");
7.         String str1 = input.nextLine();
8.         System.out.print("Enter another string: ");
9.         String str2 = input.nextLine();
10.        System.out.print("Enter a character to check its index: ");
11.        char ch = input.next().charAt(0);
12.        // Demonstrate all string methods
13.        System.out.println("\n--- String Methods Demonstration ---");
14.        // charAt()
15.        System.out.println("Character at index 2: " + str1.charAt(2));
16.        // codePointAt()
17.        System.out.println("Code point at index 1: " + str1.codePointAt(1));
18.        // codePointBefore()
19.        System.out.println("Code point before index 2: " + str1.codePointBefore(2));
20.        // codePointCount()
21.        System.out.println("Code point count: " + str1.codePointCount(0, str1.length()));
22.        // compareTo()
23.        System.out.println("Comparison (str1 vs str2): " + str1.compareTo(str2));
24.        // compareToIgnoreCase()
25.        System.out.println("Comparison ignoring case: " + str1.compareToIgnoreCase(str2));
```

```
26. // concat()
27. System.out.println("Concatenation of strings: " + str1.concat(str2));
28. // contains()
29. System.out.println("Does str1 contain 'a'? " + str1.contains("a"));
30. // contentEquals()
31. System.out.println("Does str1 content equal to str2? " + str1.contentEquals(str2));
32. // copyValueOf()
33. char[] charArray = {'H', 'e', 'l', 'l', 'o'};
34. System.out.println("Copy value of char array: " + String.copyValueOf(charArray));
35. // endsWith()
36. System.out.println("Does str1 end with 'z'? " + str1.endsWith("z"));
37. // equals()
38. System.out.println("Are str1 and str2 equal? " + str1.equals(str2));
39. // equalsIgnoreCase()
40. System.out.println("Are str1 and str2 equal (ignore case)? " + str1.equalsIgnoreCase(str2));
41. // format()
42. System.out.println("Formatted string: " + String.format("Hello %s!", str1));
43. // getBytes()
44. byte[] byteArray = str1.getBytes();
45. System.out.println("Byte array of str1: " + new String(byteArray));
46. // getChars()
47. char[] chars = new char[5];
48. str1.getChars(0, 5, chars, 0);
49. System.out.println("Characters extracted from str1: " + String.valueOf(chars));
50. // hashCode()
51. System.out.println("Hash code of str1: " + str1.hashCode());
52. // indexOf()
53. System.out.println("Index of " + ch + " in str1: " + str1.indexOf(ch));
54. // intern()
55. System.out.println("Interned string of str1: " + str1.intern());
56. // isEmpty()
57. System.out.println("Is str1 empty? " + str1.isEmpty());
58. // lastIndexOf()
59. System.out.println("Last index of " + ch + " in str1: " + str1.lastIndexOf(ch));
60. // length()
61. System.out.println("Length of str1: " + str1.length());
62. // matches()
63. System.out.println("Does str1 match '[a-zA-Z]*'? " + str1.matches("[a-zA-Z]*"));
64. // offsetByCodePoints()
65. System.out.println("Offset by code points: " + str1.offsetByCodePoints(0, 2));
```

```

66. // regionMatches()
67. System.out.println("Does region match? " + str1.regionMatches(0, str2, 0, 3));
68. // replace()
69. System.out.println("Replace 'a' with 'e': " + str1.replace('a', 'e'));
70. // replaceFirst()
71. System.out.println("Replace first 'a' with 'e': " + str1.replaceFirst("a", "e"));
72. // replaceAll()
73. System.out.println("Replace all 'a' with 'e': " + str1.replaceAll("a", "e"));
74. // split()
75. String[] parts = str1.split(" ");
76. System.out.println("Split str1 by spaces:");
77. for (String part : parts) {
78. System.out.println(part);
79. }
80. // startsWith()
81. System.out.println("Does str1 start with 'H'? " + str1.startsWith("H"));
82. // subSequence()
83. System.out.println("Subsequence (1 to 4): " + str1.subSequence(1, 4));
84. // substring()
85. System.out.println("Substring (1 to 4): " + str1.substring(1, 4));
86. // toCharArray()
87. char[] toChars = str1.toCharArray();
88. System.out.println("To char array:");
89. for (char c : toChars) {
90. System.out.print(c + " ");
91. }
92. System.out.println();
93. // toLowerCase()
94. System.out.println("Lowercase str1: " + str1.toLowerCase());
95. // toString()
96. System.out.println("String representation of str1: " + str1.toString());
97. // toUpperCase()
98. System.out.println("Uppercase str1: " + str1.toUpperCase());
99. // trim()
100. System.out.println("Trimmed str1: " + str1.trim());
101. // valueOf()
102. int number = 12345;
103. System.out.println("Value of number as string: " + String.valueOf(number));
104. input.close();
105. }

```

106. }

Here is the output of this code:

```
Output
Enter a string: Nahid
Enter another string: Pranto
Enter a character to check its index: 2

--- String Methods Demonstration ---
Character at index 2: h
Code point at index 1: 97
Code point before index 2: 97
Code point count: 6
Comparison (str1 vs str2): -2
Comparison ignoring case: -2
Concatenation of strings: Nahid Pranto
Does str1 contain 'a'? true
Does str1 content equal to str2? false
Copy value of char array: Hello
Does str1 end with 'z'? false
Are str1 and str2 equal? false
Are str1 and str2 equal (ignore case)? false
Formatted string: Hello Nahid !
Byte array of str1: Nahid
Characters extracted from str1: Nahid
Hash code of str1: -1969109680
Index of '2' in str1: -1
Interned string of str1: Nahid
Is str1 empty? false
Last index of '2' in str1: -1
Length of str1: 6
Does str1 match '[a-zA-Z]*'? false
Offset by code points: 2
Does region match? false
Replace 'a' with 'e': Nehid
Replace first 'a' with 'e': Nehid
Replace all 'a' with 'e': Nehid
Split str1 by spaces:
Nahid

--- Code Exited With Errors ---
```

6. Java Math

Here is a simple Java program demonstrating the usage of the Math class, which provides methods for performing basic numeric operations such as exponentiation, logarithms, trigonometry, and more.

1. import java.util.Scanner;
2. public class Main {
3. public static void main(String[] args) {
4. Scanner input = new Scanner(System.in);
5. // Prompt user for numbers
6. System.out.print("Enter the first number: ");
7. double num1 = input.nextDouble();
8. System.out.print("Enter the second number: ");
9. double num2 = input.nextDouble();
10. // Demonstrating Math methods
11. System.out.println("\n--- Java Math Class Methods ---");

```

12. // Absolute value
13. System.out.println("Absolute value of num1: " + Math.abs(num1));
14. // Maximum and Minimum
15. System.out.println("Maximum of num1 and num2: " + Math.max(num1, num2));
16. System.out.println("Minimum of num1 and num2: " + Math.min(num1, num2));
17. // Square root
18. System.out.println("Square root of num1: " + Math.sqrt(num1));
19. // Power
20. System.out.println("num1 raised to the power of num2: " + Math.pow(num1, num2));
21. // Trigonometric functions (sin, cos, tan)System.out.println("Sine of num1 (radians): " +
    Math.sin(num1));
22. System.out.println("Cosine of num1 (radians): " + Math.cos(num1));
23. System.out.println("Tangent of num1 (radians): " + Math.tan(num1));
24. // Logarithm
25. System.out.println("Natural log of num1: " + Math.log(num1));
26. System.out.println("Base-10 log of num1: " + Math.log10(num1));
27. // Rounding methods
28. System.out.println("Ceiling of num1: " + Math.ceil(num1));
29. System.out.println("Floor of num1: " + Math.floor(num1));
30. System.out.println("Round of num1: " + Math.round(num1));
31. // Random number generation
32. System.out.println("Random number (0.0 to 1.0): " + Math.random());
33. input.close(); // Close the scanner
34. }
35. }

```

Here is code output:

```

Output
Enter the first number: 10
Enter the second number: 30

--- Java Math Class Methods ---
Absolute value of num1: 10.0
Maximum of num1 and num2: 30.0
Minimum of num1 and num2: 10.0
Square root of num1: 3.1622776601683795
num1 raised to the power of num2: 1.0E30
Cosine of num1 (radians): -0.8390715290764524
Tangent of num1 (radians): 0.6483608274590866
Natural log of num1: 2.302585092994046
Base-10 log of num1: 1.0
Ceiling of num1: 10.0
Floor of num1: 10.0
Round of num1: 10
Random number (0.0 to 1.0): 0.7779344860553017

=== Code Execution Successful ===

```


7. Java Conditions Statements

7.1 if, else, else if

Here's a simple Java program demonstrating the use of **if**, **else**, and **else if** conditional statements:

```
1. import java.util.Scanner;
2. public class Main {
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         // Prompt user for a number
6.         System.out.print("Enter a number: ");
7.         int number = input.nextInt();
8.         // Check conditions using if, else if, and else
9.         if (number > 0) {
10.            System.out.println("The number is positive.");
11.        } else if (number < 0) {
12.            System.out.println("The number is negative.");
13.        } else {
14.            System.out.println("The number is zero.");
15.        }
16.        input.close(); // Close the scanner
17.    }
18. }
```

Output

```
Enter a number: 5
The number is positive.
```

```
=== Code Execution Successful ===
```

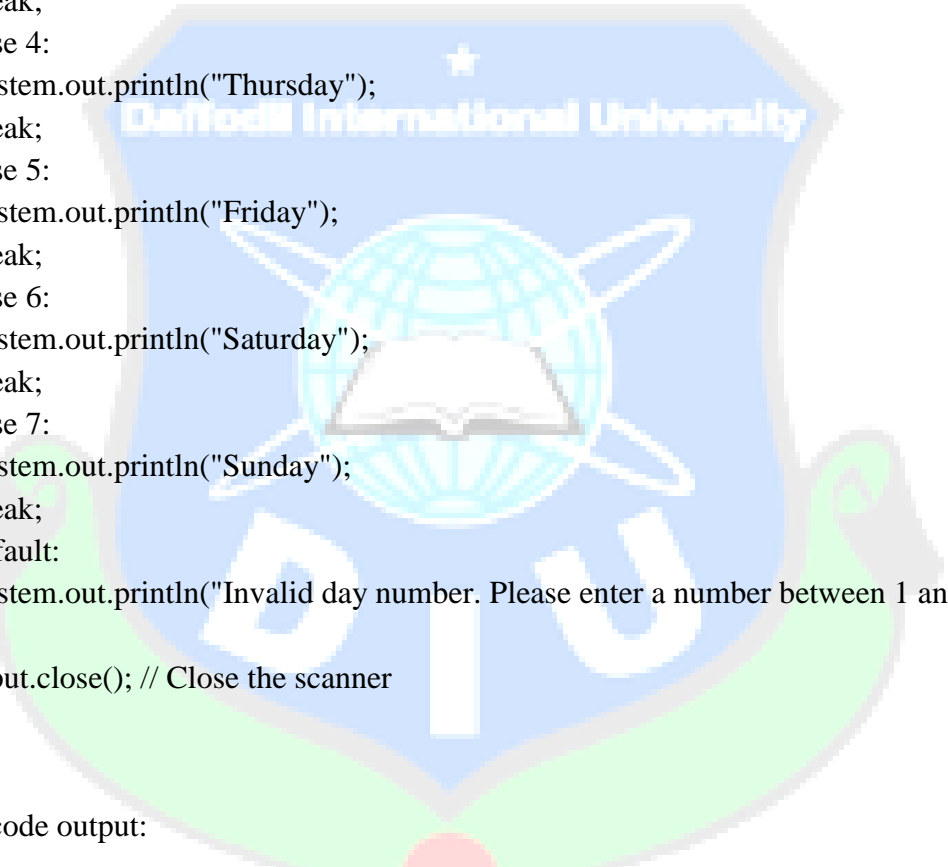
7.2 switch-case

Here's a simple Java program demonstrating the use of the switch-case conditional statement:

```
1. import java.util.Scanner;
2. public class Main {
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         // Prompt user for a day number
6.         System.out.print("Enter a day number (1-7): ");
7.         int day = input.nextInt();
8.         // Determine the day of the week using switch-case
9.         switch (day) {
```



```
10. case 1:
11. System.out.println("Monday");
12. break;
13. case 2:
14. System.out.println("Tuesday");
15. break;
16. case 3:
17. System.out.println("Wednesday");
18. break;
19. case 4:
20. System.out.println("Thursday");
21. break;
22. case 5:
23. System.out.println("Friday");
24. break;
25. case 6:
26. System.out.println("Saturday");
27. break;
28. case 7:
29. System.out.println("Sunday");
30. break;
31. default:
32. System.out.println("Invalid day number. Please enter a number between 1 and 7.");
33. }
34. input.close(); // Close the scanner
35. }
36. }
```

The logo of Daffodil International University is a shield-shaped emblem. It features a blue background with a white star at the top. Below the star, the text "Daffodil International University" is written in white. In the center, there is a white globe with a white book open in front of it. The shield is flanked by green leaves and a red flower at the bottom.

Here is a code output:

```
Output
Enter a day number (1-7): 5
Friday

=== Code Execution Successful ===
```

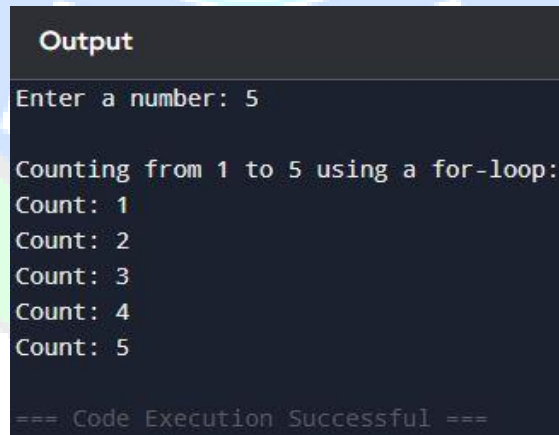
8. Java Loop:

8.1 For-Loop:

Here's a simple Java program demonstrating the for-loop:

```
1. import java.util.Scanner;
2. public class Main {
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         // Prompt user for the number of iterations
6.         System.out.print("Enter a number: ");
7.         int n = input.nextInt();
8.         System.out.println("\nCounting from 1 to " + n + " using a for-loop:");
9.         // Using a for-loop to iterate and print numbers
10.        for (int i = 1; i <= n; i++) {
11.            System.out.println("Count: " + i);
12.        }
13.        input.close(); // Close the scanner
14.    }
15. }
```

Here is a code output:



```
Output
Enter a number: 5

Counting from 1 to 5 using a for-loop:
Count: 1
Count: 2
Count: 3
Count: 4
Count: 5

=== Code Execution Successful ===
```

8.2 While Loop:

Here's a simple Java program demonstrating the while-loop:

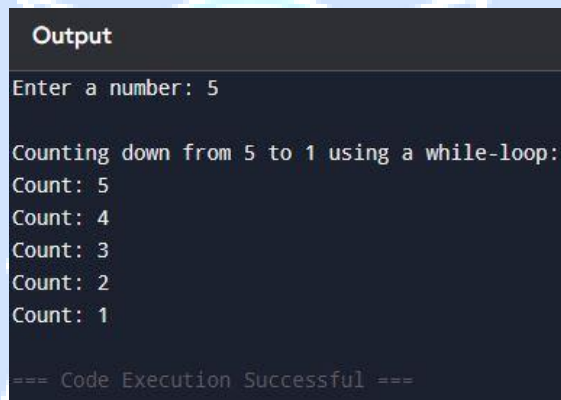
```
1. import java.util.Scanner;
2. public class Main {
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         // Prompt user for a number
6.         System.out.print("Enter a number: ");
```

```

7. int number = input.nextInt();
8. System.out.println("\nCounting down from " + number + " to 1 using a while-loop:");
9. // Initialize the loop variable
10. int i = number;
11. // Loop until i reaches 1
12. while (i >= 1) {
13. System.out.println("Count: " + i);
14. i--; // Decrease the loop variable
15. }
16. input.close(); // Close the scanner
17. }
18. }

```

Here is a code output:



```

Output
Enter a number: 5

Counting down from 5 to 1 using a while-loop:
Count: 5
Count: 4
Count: 3
Count: 2
Count: 1

=== Code Execution Successful ===

```

8.3 Do-While

Here's a simple Java program demonstrating the do-while loop:

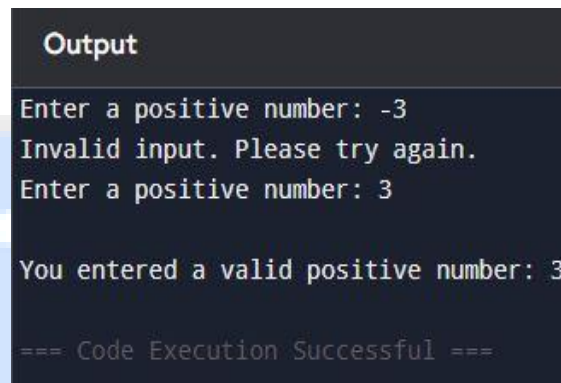
```

1. import java.util.Scanner;
2. public class Main {
3. public static void main(String[] args) {
4. Scanner input = new Scanner(System.in);
5. // Initialize a variable
6. int number;
7. // Do-while loop to ensure the user enters a positive number
8. do {
9. System.out.print("Enter a positive number: ");
10. number = input.nextInt();
11. if (number <= 0) {
12. System.out.println("Invalid input. Please try again.");
13. }
14. } while (number <= 0);

```

```
15. System.out.println("\nYou entered a valid positive number: " + number);
16. input.close(); // Close the scanner
17. }
18. }
```

Here is code output:



```
Output
Enter a positive number: -3
Invalid input. Please try again.
Enter a positive number: 3
You entered a valid positive number: 3
=== Code Execution Successful ===
```

9. Java Break and Continue

Here's a simple Java program demonstrating the use of break and continue statements in loops:

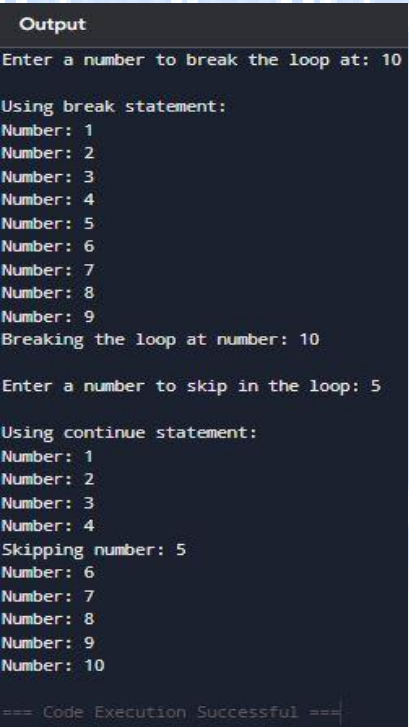
```
1. import java.util.Scanner;
2. public class Main {
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         // Taking user input for break example
6.         System.out.print("Enter a number to break the loop at: ");
7.         int breakAt = input.nextInt();
8.         System.out.println("\nUsing break statement:");
9.         // Using break in a for-loop to stop when the user input number is encountered
10.        for (int i = 1; i <= 10; i++) {
11.            if (i == breakAt) {
12.                System.out.println("Breaking the loop at number: " + i);
13.                break; // Exit the loop when i reaches the breakAt number
14.            }
15.            System.out.println("Number: " + i);
16.        }
17.        // Taking user input for continue example
18.        System.out.print("\nEnter a number to skip in the loop: ");
19.        int skipAt = input.nextInt();
20.        System.out.println("\nUsing continue statement:");
21.        // Using continue in a for-loop to skip the user input number
22.        for (int i = 1; i <= 10; i++) {
```

```

23. if (i == skipAt) {
24. System.out.println("Skipping number: " + i);
25. continue; // Skip the rest of the loop body for i == skipAt
26. }
27. System.out.println("Number: " + i);
28. }
29. input.close(); // Close the scanner
30. }
31. }

```

Here is an output of this code:



```

Output
Enter a number to break the loop at: 10

Using break statement:
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5
Number: 6
Number: 7
Number: 8
Number: 9
Breaking the loop at number: 10

Enter a number to skip in the loop: 5

Using continue statement:
Number: 1
Number: 2
Number: 3
Number: 4
Skipping number: 5
Number: 6
Number: 7
Number: 8
Number: 9
Number: 10

=== Code Execution Successful ===

```

10. Java Array: 1D Array, 2D Array

Here's a simple Java program demonstrating both 1D Array and 2D Array where the user provides input for both:

```

1. import java.util.Scanner;
2. public class Main {
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         // 1D Array Example
6.         System.out.print("Enter the size of the 1D array: ");
7.         int size1D = input.nextInt();
8.         int[] array1D = new int[size1D];

```

```

9. // Taking input for the 1D array
10. System.out.println("\nEnter the elements for the 1D array:");
11. for (int i = 0; i < size1D; i++) {
12. System.out.print("Element " + (i + 1) + ": ");
13. array1D[i] = input.nextInt();
14. }
15. // Displaying the 1D array
16. System.out.println("\nThe 1D array is:");
17. for (int i = 0; i < size1D; i++) {
18. System.out.println("Element " + (i + 1) + ": " + array1D[i]);
19. }
20. // 2D Array Example
21. System.out.print("\nEnter the number of rows for the 2D array: ");
22. int rows = input.nextInt();
23. System.out.print("Enter the number of columns for the 2D array: ");
24. int columns = input.nextInt();
25. int[][] array2D = new int[rows][columns];
26. // Taking input for the 2D array
27. System.out.println("\nEnter the elements for the 2D array:");
28. for (int i = 0; i < rows; i++) {
29. for (int j = 0; j < columns; j++) {
30. System.out.print("Element [" + (i + 1) + "][" + (j + 1) + "]: ");
31. array2D[i][j] = input.nextInt();
32. }
33. }
34. // Displaying the 2D array
35. System.out.println("\nThe 2D array is:");
36. for (int i = 0; i < rows; i++) {
37. for (int j = 0; j < columns; j++) {
38. System.out.print(array2D[i][j] + " ");
39. }
40. System.out.println();
41. }
42. input.close(); // Close the scanner
43. }
44. }

```

Here is code input and output:

```

Output
Enter the size of the 1D array: 2

Enter the elements for the 1D array:
Element 1: 5
Element 2: 6

The 1D array is:
Element 1: 5
Element 2: 6

Enter the number of rows for the 2D array: 2
Enter the number of columns for the 2D array: 2

Enter the elements for the 2D array:
Element [1][1]: 5
Element [1][2]: 6
Element [2][1]: 7
Element [2][2]: 8

The 2D array is:
5 6
7 8

=== Code Execution Successful ===

```

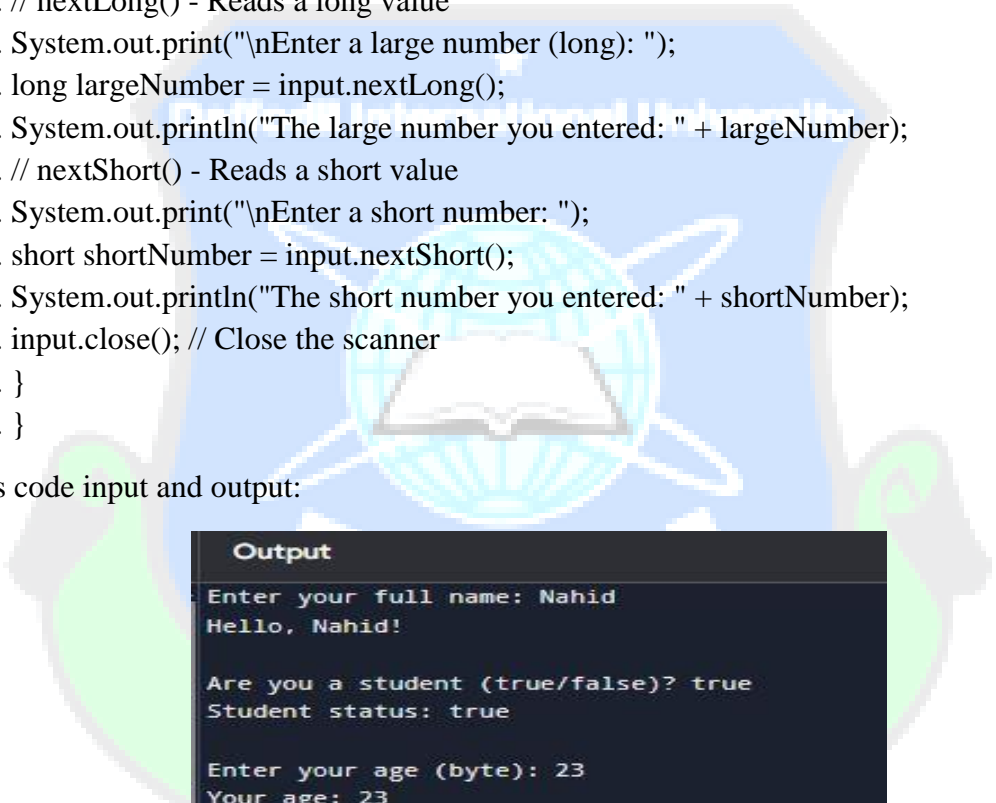
11. Java User Input (Scanner): `nextLine()`, `nextBoolean()`, `nextByte()`, `nextDouble()`, `nextFloat()`, `nextInt()`, `nextLong()`, `nextShort()`

Here's a simple Java program that demonstrates the use of different Scanner methods to read various types of user input:

1. `import java.util.Scanner;`
2. `public class Main {`
3. `public static void main(String[] args) {`
4. `Scanner input = new Scanner(System.in);`
5. `// nextLine() - Reads a full line of text`
6. `System.out.print("Enter your full name: ");`
7. `String name = input.nextLine();`
8. `System.out.println("Hello, " + name + "!");`
9. `// nextBoolean() - Reads a boolean value (true/false)`
10. `System.out.print("\nAre you a student (true/false)? ");`
11. `boolean isStudent = input.nextBoolean();`
12. `System.out.println("Student status: " + isStudent);`
13. `// nextByte() - Reads a byte value`
14. `System.out.print("\nEnter your age (byte): ");`
15. `byte age = input.nextByte();`
16. `System.out.println("Your age: " + age);`
17. `// nextDouble() - Reads a double value`
18. `System.out.print("\nEnter your height in meters (double): ");`
19. `double height = input.nextDouble();`
20. `System.out.println("Your height: " + height + " meters");`

```
21. // nextFloat() - Reads a float value
22. System.out.print("\nEnter your weight in kilograms (float): ");
23. float weight = input.nextFloat();
24. System.out.println("Your weight: " + weight + " kg");
25. // nextInt() - Reads an integer value
26. System.out.print("\nEnter your year of birth (int): ");
27. int birthYear = input.nextInt();
28. System.out.println("Your year of birth: " + birthYear);
29. // nextLong() - Reads a long value
30. System.out.print("\nEnter a large number (long): ");
31. long largeNumber = input.nextLong();
32. System.out.println("The large number you entered: " + largeNumber);
33. // nextShort() - Reads a short value
34. System.out.print("\nEnter a short number: ");
35. short shortNumber = input.nextShort();
36. System.out.println("The short number you entered: " + shortNumber);
37. input.close(); // Close the scanner
38. }
39. }
```

Here is code input and output:



```
Output
Enter your full name: Nahid
Hello, Nahid!

Are you a student (true/false)? true
Student status: true

Enter your age (byte): 23
Your age: 23

Enter your height in meters (double): 120
Your height: 120.0 meters

Enter your weight in kilograms (float): 80
Your weight: 80.0 kg

Enter your year of birth (int): 2002
Your year of birth: 2002

Enter a large number (long): 0242310005101939
The large number you entered: 242310005101939

Enter a short number: 02423
The short number you entered: 2423

=== Code Execution Successful ===
```

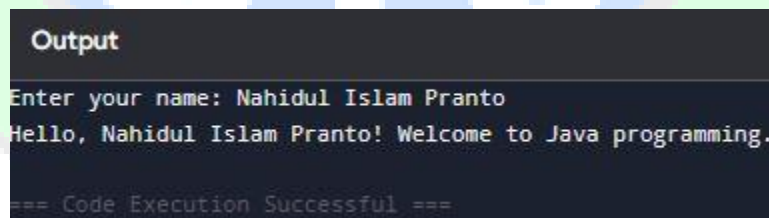

12. Java Methods

12.1 Method with no arguments and no return value

Here's a simple Java program demonstrating a method with no arguments and no return value. The user is prompted to provide input inside the method.

```
1. import java.util.Scanner;
2. public class Main {
3.     // Method with no arguments and no return value
4.     public static void displayMessage() {
5.         Scanner input = new Scanner(System.in);
6.         // Prompt the user to enter their name
7.         System.out.print("Enter your name: ");
8.         String name = input.nextLine();
9.         // Display a personalized message
10.        System.out.println("Hello, " + name + "! Welcome to Java programming.");
11.        input.close(); // Close the scanner
12.    }
13.    public static void main(String[] args) {
14.        // Call the method
15.        displayMessage();
16.    }
17. }
```

Here is the code output and input:



```
Output
Enter your name: Nahidul Islam Pranto
Hello, Nahidul Islam Pranto! Welcome to Java programming.
=== Code Execution Successful ===
```

12.2 Method with arguments but no return value

Here's a simple Java program demonstrating a method with arguments but no return value. The user is prompted to provide input, which is then passed to the method.

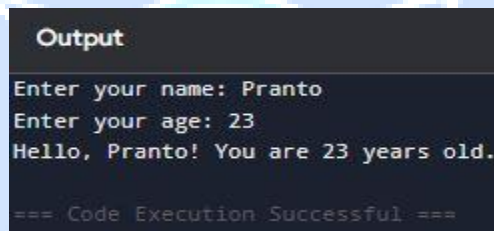
```
1. import java.util.Scanner;
2. public class Main {
3.     // Method with arguments and no return value
4.     public static void displayMessage(String name, int age) {
5.         // Display a personalized message
6.         System.out.println("Hello, " + name + "! You are " + age + " years old.");
7.     }
8.     public static void main(String[] args) {
```

```

9. Scanner input = new Scanner(System.in);
10. // Prompt the user to enter their name
11. System.out.print("Enter your name: ");
12. String name = input.nextLine();
13. // Prompt the user to enter their age
14. System.out.print("Enter your age: ");
15. int age = input.nextInt();
16. // Call the method and pass user inputs as arguments
17. displayMessage(name, age);
18. input.close(); // Close the scanner
19. }
20. }

```

Here is the code output and input:



```

Output
Enter your name: Pranto
Enter your age: 23
Hello, Pranto! You are 23 years old.

=== Code Execution Successful ===

```

12.3 Method with no arguments but with return value

Here's a simple Java program demonstrating a method with no arguments but a return value. The user is prompted to provide input inside the method, and the method returns the value to the calling function.

```

1. import java.util.Scanner;
2. public class Main {
3.     // Method with no arguments but with a return value
4.     public static String getName() {
5.         Scanner input = new Scanner(System.in);
6.         // Prompt the user to enter their name
7.         System.out.print("Enter your name: ");
8.         String name = input.nextLine();
9.         // Return the inputted name
10.        return name;
11.    }
12.    public static void main(String[] args) {
13.        // Call the method and store the return value
14.        String userName = getName();
15.        // Display the returned value
16.        System.out.println("Hello, " + userName + "! Welcome to Java programming.");

```

```
17. }  
18. }
```

Here is the code output and input:



```
Output  
Enter your name: Nahid  
Hello, Nahid! Welcome to Java programming.  
=== Code Execution Successful ===
```

12.4 Method with arguments and return value

Here's a simple Java program demonstrating a method with arguments and a return value. The user is prompted to provide input, and the method performs a calculation based on the input and returns the result.

```
1. import java.util.Scanner;  
2. public class Main {  
3.     // Method with arguments and a return value  
4.     public static int calculateSum(int num1, int num2) {  
5.         return num1 + num2; // Returns the sum of the two numbers  
6.     }  
7.     public static void main(String[] args) {  
8.         Scanner input = new Scanner(System.in);  
9.         // Prompt the user to enter the first number  
10.        System.out.print("Enter the first number: ");  
11.        int number1 = input.nextInt();  
12.        // Prompt the user to enter the second number  
13.        System.out.print("Enter the second number: ");  
14.        int number2 = input.nextInt();  
15.        // Call the method and store the returned value  
16.        int sum = calculateSum(number1, number2);  
17.        // Display the result  
18.        System.out.println("The sum of " + number1 + " and " + number2 + " is: " + sum);  
19.        input.close(); // Close the scanner  
20.    }  
21. }
```

Here is the code output and input:

```
Output
Enter the first number: 25
Enter the second number: 5
The sum of 25 and 5 is: 30

=== Code Execution Successful ===
```

13. Java Method Overloading

Here's a simple Java program demonstrating method overloading, where multiple methods have the same name but differ in the type or number of parameters. The user is prompted to input data, and the overloaded methods handle the inputs accordingly.

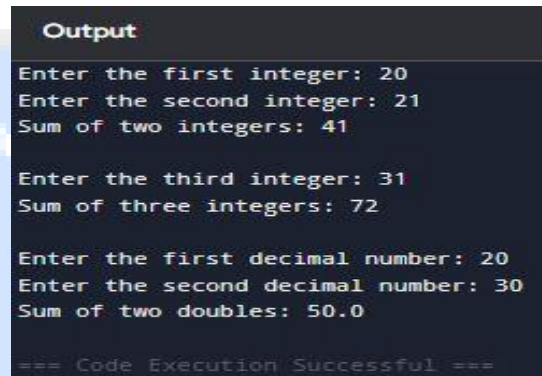
```
1. import java.util.Scanner;
2. public class Main {
3.     // Overloaded method to calculate the sum of two integers
4.     public static int calculateSum(int a, int b) {
5.         return a + b;
6.     }
7.     // Overloaded method to calculate the sum of three integers
8.     public static int calculateSum(int a, int b, int c) {
9.         return a + b + c;
10.    }
11.    // Overloaded method to calculate the sum of two double values
12.    public static double calculateSum(double a, double b) {
13.        return a + b;
14.    }
15.    public static void main(String[] args) {
16.        Scanner input = new Scanner(System.in);
17.        // Option 1: Sum of two integers
18.        System.out.print("Enter the first integer: ");
19.        int num1 = input.nextInt();
20.        System.out.print("Enter the second integer: ");
21.        int num2 = input.nextInt();
22.        System.out.println("Sum of two integers: " + calculateSum(num1, num2));
23.        // Option 2: Sum of three integers
24.        System.out.print("\nEnter the third integer: ");
25.        int num3 = input.nextInt();
26.        System.out.println("Sum of three integers: " + calculateSum(num1, num2, num3));
27.        // Option 3: Sum of two double values
28.        System.out.print("\nEnter the first decimal number: ");
29.        double double1 = input.nextDouble();
```

```

30. System.out.print("Enter the second decimal number: ");
31. double double2 = input.nextDouble();
32. System.out.println("Sum of two doubles: " + calculateSum(double1, double2));
33. input.close(); // Close the scanner
34. }
35. }

```

Here is the code output and input:



```

Output
Enter the first integer: 20
Enter the second integer: 21
Sum of two integers: 41

Enter the third integer: 31
Sum of three integers: 72

Enter the first decimal number: 20
Enter the second decimal number: 30
Sum of two doubles: 50.0

=== Code Execution Successful ===

```

14. Problem Solving with User Input

14.1 Check Leap Year

Here's a simple Java program to check whether a given year is a leap year or not, with the user providing the input.

```

1. import java.util.Scanner;
2. public class Main {
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         // Prompt the user to enter a year
6.         System.out.print("Enter a year to check if it's a leap year: ");
7.         int year = input.nextInt();
8.         // Check if the year is a leap year
9.         if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {
10.            System.out.println(year + " is a leap year.");
11.        } else {
12.            System.out.println(year + " is not a leap year.");
13.        }
14.        input.close(); // Close the scanner
15.    }
16. }

```

Here is the code output and input:

```
Output
Enter a year to check if it's a leap year: 2024
2024 is a leap year.

=== Code Execution Successful ===
```

```
Output
Enter a year to check if it's a leap year: 2025
2025 is not a leap year.

=== Code Execution Successful ===
```

14.2 BMI Calculator

Here's a simple Java program to calculate the Body Mass Index (BMI), where the user provides their weight and height as inputs.

```
1. import java.util.Scanner;
2. public class Main {
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         // Prompt the user to enter their weight in kilograms
6.         System.out.print("Enter your weight in kilograms: ");
7.         double weight = input.nextDouble();
8.         // Prompt the user to enter their height in meters
9.         System.out.print("Enter your height in meters: ");
10.        double height = input.nextDouble();
11.        // Calculate BMI
12.        double bmi = weight / (height * height);
13.        // Display the BMI result
14.        System.out.printf("Your BMI is: %.2f\n", bmi);
15.        // Provide a BMI category
16.        if (bmi < 18.5) {
17.            System.out.println("Category: Underweight");
18.        } else if (bmi >= 18.5 && bmi < 24.9) {
19.            System.out.println("Category: Normal weight");
20.        } else if (bmi >= 25 && bmi < 29.9) {
21.            System.out.println("Category: Overweight");
22.        } else {
23.            System.out.println("Category: Obesity");
24.        }
25.        input.close(); // Close the scanner
26.    }
27. }
```

Here is the code output and input:

```
Output
Enter your weight in kilograms: 90
Enter your height in meters: 1.75
Your BMI is: 29.39
Category: Overweight

=== Code Execution Successful ===
```

14.3 Calculate the Area of Triangle

Here's a simple Java program to calculate the area of a triangle. The user is prompted to provide the base and height as input.

1. import java.util.Scanner;
2. public class Main {
3. public static void main(String[] args) {
4. Scanner input = new Scanner(System.in);
5. // Prompt the user to enter the base of the triangle
6. System.out.print("Enter the base of the triangle: ");
7. double base = input.nextDouble();
8. // Prompt the user to enter the height of the triangle
9. System.out.print("Enter the height of the triangle: ");
10. double height = input.nextDouble();
11. // Calculate the area of the triangle
12. double area = 0.5 * base * height;
13. // Display the result
14. System.out.printf("The area of the triangle is: %.2f\n", area);
15. input.close(); // Close the scanner
16. }
17. }

Here is the code output and input:

```
Output
Enter the base of the triangle: 20
Enter the height of the triangle: 40
The area of the triangle is: 400.00

=== Code Execution Successful ===
```

14.4 Calculate Factorial

Here's a simple Java program to calculate the factorial of a given number. The user provides the input, and the program computes the factorial.

```
1. import java.util.Scanner;
2. public class Main {
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         // Prompt the user to enter a number
6.         System.out.print("Enter a non-negative integer to calculate its factorial: ");
7.         int number = input.nextInt();
8.         // Validate input
9.         if (number < 0) {
10.            System.out.println("Factorial is not defined for negative numbers.");
11.        } else {
12.            // Calculate factorial
13.            long factorial = 1;
14.            for (int i = 1; i <= number; i++) {
15.                factorial *= i; // Multiply current value
16.            }
17.            // Display the result
18.            System.out.println("The factorial of " + number + " is: " + factorial);
19.        }
20.        input.close(); // Close the scanner
21.    }
22. }
```

Here is the code output and input:

```
Output
Enter a non-negative integer to calculate its factorial: 5
The factorial of 5 is: 120
=== Code Execution Successful ===
```

```
Output
Enter a non-negative integer to calculate its factorial: -5
Factorial is not defined for negative numbers.
=== Code Execution Successful ===
```

14.5 Counting Vowels in a String

Here's a simple Java program to count the number of vowels in a user-provided string.

```
1. import java.util.Scanner;
2. public class Main{
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         // Prompt the user to enter a string
6.         System.out.print("Enter a string: ");
```

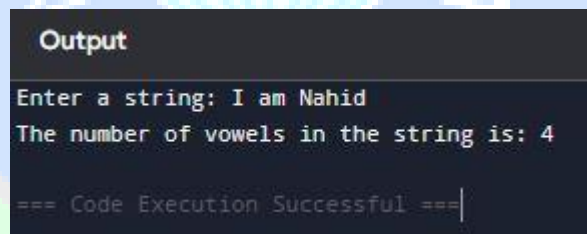


```

7. String userInput = input.nextLine();
8. // Convert the string to lowercase for easier comparison
9. String lowerCaseInput = userInput.toLowerCase();
10. // Initialize a counter for vowels
11. int vowelCount = 0;
12. // Loop through each character in the string
13. for (int i = 0; i < lowerCaseInput.length(); i++) {
14. char ch = lowerCaseInput.charAt(i);
15. // Check if the character is a vowel
16. if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
17. vowelCount++;
18. }
19. }
20. // Display the result
21. System.out.println("The number of vowels in the string is: " + vowelCount);
22. input.close(); // Close the scanner
23. }
24. }

```

Here is the code output and input



```

Output
Enter a string: I am Nahid
The number of vowels in the string is: 4
=== Code Execution Successful ===

```

14.6 Prime Factorization

Here's a simple Java program for prime factorization, where the user inputs a number, and the program calculates its prime factors.

```

1. import java.util.Scanner;
2. public class Main{
3. public static void main(String[] args) {
4. Scanner input = new Scanner(System.in);
5. // Prompt the user to enter a positive integer
6. System.out.print("Enter a positive integer: ");
7. int number = input.nextInt();
8. if (number <= 1) {
9. System.out.println("Prime factorization is not defined for numbers less than or equal to
1. ");
10. } else {

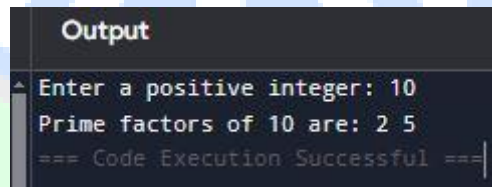
```

```

11. System.out.print("Prime factors of " + number + " are: ");
12. // Divide by 2 until the number is odd
13. while (number % 2 == 0) {
14. System.out.print(2 + " ");
15. number /= 2;
16. }
17. // Check for odd factors starting from 3
18. for (int i = 3; i <= Math.sqrt(number); i += 2) {
19. while (number % i == 0) {
20. System.out.print(i + " ");
21. number /= i;
22. }
23. }
24. // If the remaining number is greater than 2, it is a prime factor
25. if (number > 2) {
26. System.out.print(number);
27. }
28. }
29. input.close(); // Close the scanner
30. }
31. }

```

Here is the code output and input:



```

Output
Enter a positive integer: 10
Prime factors of 10 are: 2 5
=== Code Execution Successful ===

```

14.7 Reverse a Number

Here's a simple Java program to reverse a number input by the user:

```

1. import java.util.Scanner;
2. public class Main {
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         // Prompt the user to enter a number
6.         System.out.print("Enter a number to reverse: ");
7.         int number = input.nextInt();
8.         int reverse = 0; // Variable to store the reversed number
9.         // Reverse the number
10.        while (number != 0) {

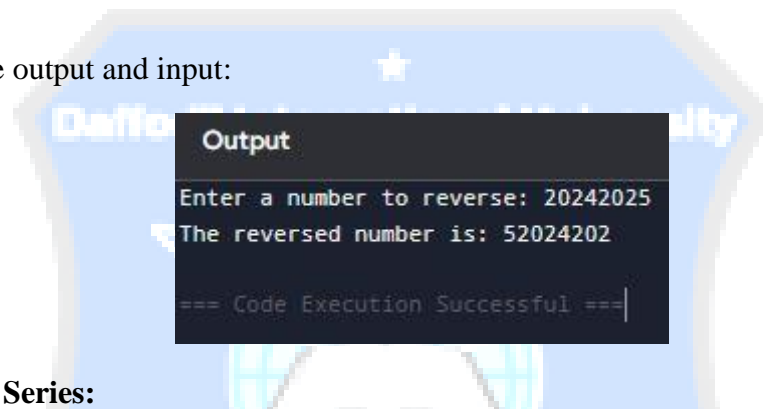
```

```

11. int digit = number % 10; // Extract the last digit
12. reverse = reverse * 10 + digit; // Add it to the reversed number
13. number /= 10; // Remove the last digit
14. }
15. // Display the reversed number
16. System.out.println("The reversed number is: " + reverse);
17. input.close(); // Close the scanner
18. }
19. }

```

Here is the code output and input:



14.8 Fibonacci Series:

Here's a simple Java program to generate the Fibonacci series up to a user-specified number of terms.

```

1. import java.util.Scanner;
2. public class Main {
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         // Prompt the user to enter the number of terms
6.         System.out.print("Enter the number of terms for the Fibonacci series: ");
7.         int n = input.nextInt();
8.         if (n <= 0) {
9.             System.out.println("Please enter a positive integer.");
10.        } else {
11.            System.out.println("The Fibonacci series is: ");
12.            int first = 0, second = 1;
13.            for (int i = 1; i <= n; i++) {
14.                System.out.print(first + " ");
15.                // Calculate the next term
16.                int next = first + second;
17.                first = second;
18.                second = next;
19.            }
20.        }

```

```
21. input.close(); // Close the scanner
22. }
23. }
```

Here is the code output and input:

```
Output
Enter the number of terms for the Fibonacci series: 7
The Fibonacci series is:
0 1 1 2 3 5 8
=== Code Execution Successful ===
```

14.9 Sum of Natural Numbers (1 to 100)

Here's a simple Java program to calculate the sum of natural numbers:

```
1. import java.util.Scanner;
2. public class Main {
3.     public static void main(String[] args) {
4.         Scanner input = new Scanner(System.in);
5.         // Prompt the user to enter the upper limit
6.         System.out.print("Enter a positive integer (upper limit): ");
7.         int n = input.nextInt();
8.         if (n <= 0) {
9.             System.out.println("Please enter a positive integer.");
10.        } else {
11.            int sum = 0;
12.            // Calculate the sum of natural numbers
13.            for (int i = 1; i <= n; i++) {
14.                sum += i; // Add the current number to the sum
15.            }
16.            // Display the result
17.            System.out.println("The sum of natural numbers from 1 to " + n + " is: " + sum);
18.        }
19.        input.close(); // Close the scanner
20.    }
21. }
```

Here is the code output and input:

```
Output
Enter a positive integer (upper limit): 99
The sum of natural numbers from 1 to 99 is: 4950

=== Code Execution Successful ===
```

15. Java Constructors & Constructor Overloading

Here's a Java program demonstrating Constructors and Constructor Overloading. The user will input details, and the constructors will initialize object properties accordingly.

```
1. import java.util.Scanner;
2. class Person {
3.     String name;
4.     int age;
5.     String city;
6.     // Default constructor
7.     public Person() {
8.         this.name = "Unknown";
9.         this.age = 0;
10.        this.city = "Unknown";
11.    }
12.    // Constructor with one parameter
13.    public Person(String name) {
14.        this.name = name;
15.        this.age = 0; // Default age
16.        this.city = "Unknown"; // Default city
17.    }
18.    // Constructor with two parameters
19.    public Person(String name, int age) {
20.        this.name = name;
21.        this.age = age;
22.        this.city = "Unknown"; // Default city
23.    }
24.    // Constructor with three parameters
25.    public Person(String name, int age, String city) {
26.        this.name = name;
27.        this.age = age;
28.        this.city = city;
29.    }
30.    // Method to display information
31.    public void displayInfo() {
```

```
32. System.out.println("Name: " + name);
33. System.out.println("Age: " + age);
34. System.out.println("City: " + city);
35. }
36. }
37. public class Main {
38. public static void main(String[] args) {
39. Scanner input = new Scanner(System.in);
40. // Using default constructor
41. Person person1 = new Person();
42. System.out.println("Default Constructor:");
43. person1.displayInfo();
44. // Using constructor with one parameter
45. System.out.print("\nEnter name: ");
46. String name = input.nextLine();
47. Person person2 = new Person(name);
48. System.out.println("\nConstructor with one parameter:");
49. person2.displayInfo();
50. // Using constructor with two parameters
51. System.out.print("\nEnter name: ");
52. name = input.nextLine();
53. System.out.print("Enter age: ");
54. int age = input.nextInt();
55. Person person3 = new Person(name, age);
56. System.out.println("\nConstructor with two parameters:");
57. person3.displayInfo();
58. // Using constructor with three parameters
59. input.nextLine(); // Consume leftover newline
60. System.out.print("\nEnter name: ");
61. name = input.nextLine();
62. System.out.print("Enter age: ");
63. age = input.nextInt();
64. input.nextLine(); // Consume leftover newline
65. System.out.print("Enter city: ");
66. String city = input.nextLine();
67. Person person4 = new Person(name, age, city);
68. System.out.println("\nConstructor with three parameters:");
69. person4.displayInfo();
70. input.close(); // Close the scanner
71. }
```

72. }

Here is the code output and input:

```
Output
Default Constructor:
Name: Unknown
Age: 0
City: Unknown

Enter name: Nahid

Constructor with one parameter:
Name: Nahid
Age: 0
City: Unknown

Enter name: Pranto
Enter age: 20

Constructor with two parameters:
Name: Pranto
Age: 20
City: Unknown

Enter name: Nurul islam
Enter age: 45
Enter city: Narsingdi

Constructor with three parameters:
Name: Nurul islam
Age: 45
City: Narsingdi

=== Code Execution Successful ===
```

16. Encapsulation in Java

Here's a Java program demonstrating Encapsulation.

```
1. import java.util.Scanner;
2. class Student {
3.     // Private fields (Encapsulation)
4.     private String name;
5.     private int age;
6.     private double grade;
7.     // Getter for name
8.     public String getName() {
9.         return name;
10.    }
11.    // Setter for name
12.    public void setName(String name) {
13.        this.name = name;
14.    }
15.    // Getter for age
16.    public int getAge() {
17.        return age;
18.    }
```

```

19. // Setter for age
20. public void setAge(int age) {
21. if (age > 0) { // Validation
22. this.age = age;
23. } else {
24. System.out.println("Age must be positive. Setting default age to 18.");
25. this.age = 18; // Default age
26. }
27. }
28. // Getter for grade
29. public double getGrade() {
30. return grade;
31. }
32. // Setter for grade
33. public void setGrade(double grade) {
34. if (grade >= 0 && grade <= 100) { // Validation
35. this.grade = grade;
36. } else {
37. System.out.println("Grade must be between 0 and 100. Setting default grade to 0.");
38. this.grade = 0; // Default grade
39. }
40. }
41. // Method to display student details
42. public void displayStudentInfo() {
43. System.out.println("Student Details:");
44. System.out.println("Name: " + name);
45. System.out.println("Age: " + age);
46. System.out.println("Grade: " + grade);
47. }
48. }
49. public class Main {
50. public static void main(String[] args) {
51. Scanner input = new Scanner(System.in);
52. // Create a Student object
53. Student student = new Student();
54. // Set student details using user input
55. System.out.print("Enter student's name: ");
56. String name = input.nextLine();
57. student.setName(name);
58. System.out.print("Enter student's age: ");

```

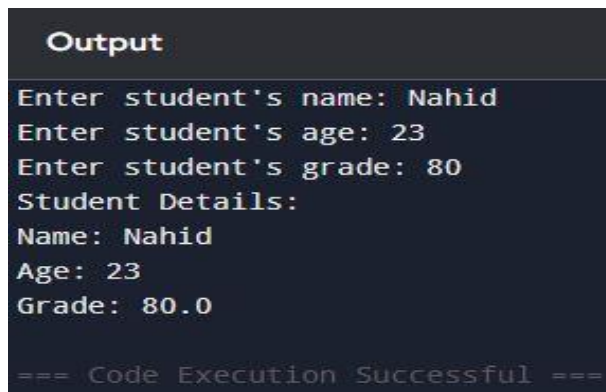


```

59. int age = input.nextInt();
60. student.setAge(age);
61. System.out.print("Enter student's grade: ");
62. double grade = input.nextDouble();
63. student.setGrade(grade);
64. // Display student details
65. student.displayStudentInfo();
66. input.close();
67. // Close the scanner
68. }
69. }

```

Here is the code output and input:



```

Output
Enter student's name: Nahid
Enter student's age: 23
Enter student's grade: 80
Student Details:
Name: Nahid
Age: 23
Grade: 80.0

=== Code Execution Successful ===

```

17. Inheritance and Polymorphism in Java

Here's a Java program that demonstrates Inheritance and Polymorphism.

```

1. import java.util.Scanner;
2. // Parent class
3. class Animal {
4.     private String name;
5.     // Constructor
6.     public Animal(String name) {
7.         this.name = name;
8.     }
9.     // Getter for name
10.    public String getName() {
11.        return name;
12.    }
13.    // Method to display sound (to be overridden)
14.    public void makeSound() {
15.        System.out.println(name + " makes a generic sound.");

```

```

16. }
17. }
18. // Child class 1
19. class Dog extends Animal {
20. public Dog(String name) {
21. super(name);
22. }
23. // Overridden method
24. @Override
25. public void makeSound() {
26. System.out.println(getName() + " barks: Woof Woof!");
27. }
28. }
29. // Child class 2
30. class Cat extends Animal {
31. public Cat(String name) {
32. super(name);
33. }
34. // Overridden method
35. @Override
36. public void makeSound() {
37. System.out.println(getName() + " meows: Meow Meow!");
38. }
39. }
40. public class Main {
41. public static void main(String[] args) {
42. Scanner input = new Scanner(System.in);
43. // Get animal type from the user
44. System.out.print("Enter the type of animal (Dog/Cat): ");
45. String animalType = input.nextLine();
46. System.out.print("Enter the name of the animal: ");
47. String animalName = input.nextLine();
48. // Polymorphism: Create Animal object dynamically
49. Animal animal;
50. if (animalType.equalsIgnoreCase("Dog")) {
51. animal = new Dog(animalName);
52. } else if (animalType.equalsIgnoreCase("Cat")) {
53. animal = new Cat(animalName);
54. } else {
55. animal = new Animal(animalName);

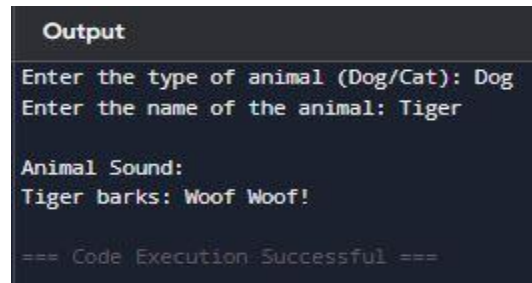
```

```

56. }
57. // Display the sound the animal makes
58. System.out.println("\nAnimal Sound:");
59. animal.makeSound();
60. input.close(); // Close the scanner
61. }
62. }

```

Here is the code output and input:



```

Output
Enter the type of animal (Dog/Cat): Dog
Enter the name of the animal: Tiger

Animal Sound:
Tiger barks: Woof Woof!

=== Code Execution Successful ===

```

18. Abstraction in Java

Here's a Java program that demonstrates Abstraction using an abstract class.

```

1. import java.util.Scanner;
2. // Abstract class
3. abstract class Shape {
4. // Abstract method to calculate the area
5. abstract double calculateArea();
6. // Abstract method to display the shape's name
7. abstract void displayShapeName();
8. }
9. // Subclass 1: Circle
10. class Circle extends Shape {
11. private double radius;
12. // Constructor
13. public Circle(double radius) {
14. this.radius = radius;
15. }
16. // Implement calculateArea
17. @Override
18. double calculateArea() {
19. return Math.PI * radius * radius;
20. }
21. // Implement displayShapeName

```

```

22. @Override
23. void displayShapeName() {
24. System.out.println("Shape: Circle");
25. }
26. }
27. // Subclass 2: Rectangle
28. class Rectangle extends Shape {
29. private double length;
30. private double width;
31. // Constructor
32. public Rectangle(double length, double width) {
33. this.length = length;
34. this.width = width;
35. }
36. // Implement calculateArea
37. @Override
38. double calculateArea() {
39. return length * width;
40. }
41. // Implement displayShapeName
42. @Override
43. void displayShapeName() {
44. System.out.println("Shape: Rectangle");
45. }
46. }
47. public class Main {
48. public static void main(String[] args) {
49. Scanner input = new Scanner(System.in);
50. // Ask the user to choose a shape
51. System.out.print("Enter the shape (Circle/Rectangle): ");
52. String shapeType = input.nextLine();
53. Shape shape; // Declare a reference to the abstract class
54. if (shapeType.equalsIgnoreCase("Circle")) {
55. // Take radius input for Circle
56. System.out.print("Enter the radius of the circle: ");
57. double radius = input.nextDouble();
58. shape = new Circle(radius); // Instantiate Circle
59. } else if (shapeType.equalsIgnoreCase("Rectangle")) {
60. // Take length and width input for Rectangle
61. System.out.print("Enter the length of the rectangle: ");

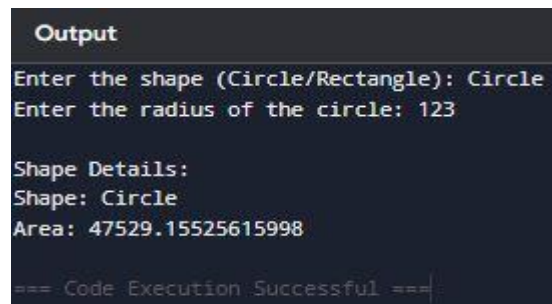
```

```

62. double length = input.nextDouble();
63. System.out.print("Enter the width of the rectangle: ");
64. double width = input.nextDouble();
65. shape = new Rectangle(length, width); // Instantiate Rectangle
66. } else {
67. System.out.println("Invalid shape!");
68. input.close();
69. return;
70. }
71. // Use the abstract class reference to call methods
72. System.out.println("\nShape Details:");
73. shape.displayShapeName();
74. System.out.println("Area: " + shape.calculateArea());
75. input.close(); // Close the scanner
76. }
77. }

```

Here is the code output and input:



```

Output
Enter the shape (Circle/Rectangle): Circle
Enter the radius of the circle: 123

Shape Details:
Shape: Circle
Area: 47529.15525615998

=== Code Execution Successful ===

```

19. Java Array List

Here's a Java program that demonstrates the use of ArrayList.

```

1. import java.util.ArrayList;
2. import java.util.Scanner;
3. public class Main {
4. public static void main(String[] args) {
5. Scanner input = new Scanner(System.in);
6. ArrayList<String> items = new ArrayList<>(); // Create an ArrayList of Strings
7. // Taking input to add items to the ArrayList
8. System.out.print("How many items do you want to add? ");
9. int count = input.nextInt();
10. input.nextLine(); // Consume the newline character
11. for (int i = 0; i < count; i++) {
12. System.out.print("Enter item " + (i + 1) + ": ");

```

```

13. String item = input.nextLine();
14. items.add(item); // Add the item to the ArrayList
15. }
16. // Display the ArrayList
17. System.out.println("\nThe items in the list are:");
18. for (String item : items) {
19. System.out.println(item);
20. }
21. // Modify an item in the list
22. System.out.print("\nDo you want to modify an item? (yes/no): ");
23. String modify = input.nextLine();
24. if (modify.equalsIgnoreCase("yes")) {
25. System.out.print("Enter the position (1 to " + items.size() + ") of the item to modify: ");
26. int position = input.nextInt() - 1;
27. input.nextLine(); // Consume the newline character
28. if (position >= 0 && position < items.size()) {
29. System.out.print("Enter the new value: ");
30. String newValue = input.nextLine();
31. items.set(position, newValue); // Modify the item
32. System.out.println("Item updated successfully!");
33. } else {
34. System.out.println("Invalid position!");
35. }
36. }
37. // Remove an item from the list
38. System.out.print("\nDo you want to remove an item? (yes/no): ");
39. String remove = input.nextLine();
40. if (remove.equalsIgnoreCase("yes")) {
41. System.out.print("Enter the position (1 to " + items.size() + ") of the item to remove: ");
42. int position = input.nextInt() - 1;
43. if (position >= 0 && position < items.size()) {
44. items.remove(position); // Remove the item
45. System.out.println("Item removed successfully!");
46. } else {
47. System.out.println("Invalid position!");
48. }
49. }
50. // Display the final ArrayList
51. System.out.println("\nThe final list is:");
52. for (String item : items) {

```

```
53. System.out.println(item);
54. }
55. input.close(); // Close the scanner
56. }
57. }
```

Here is the code output and input:

```
Output
How many items do you want to add? 5
Enter item 1: Apple
Enter item 2: lemon
Enter item 3: drinks
Enter item 4: water
Enter item 5: tea

The items in the list are:
Apple
lemon
drinks
water
tea

Do you want to modify an item? (yes/no): no

Do you want to remove an item? (yes/no): yes
Enter the position (1 to 5) of the item to remove: 2
Item removed successfully!

The final list is:
Apple
drinks
water
tea

=== Code Execution Successful ===
```

20. Reference sites

- <https://www.w3schools.com/java/>
- <https://www.programiz.com/java-programming>
- <https://www.programiz.com/java-programming/online-compiler/>
- <https://github.com/in28minutes/java-tutorial-for-beginners>