

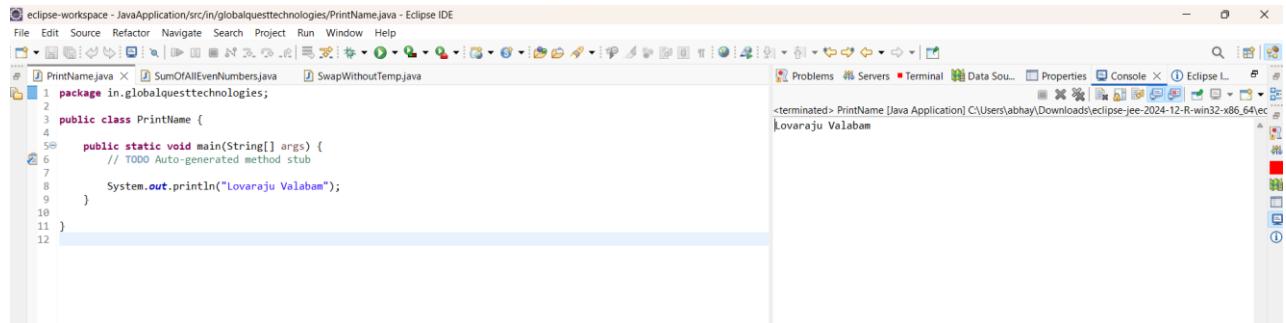
GLOBAL QUEST TECHNOLOGIES

CORE JAVA PROGRAMS

SUBHARANJAN NAYAK

Introduction:

1. Write a Java program to print your name.



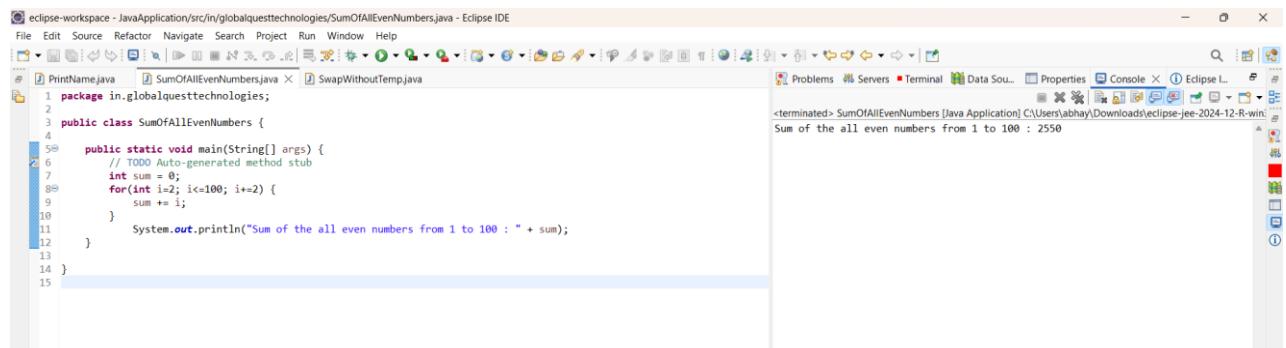
The screenshot shows the Eclipse IDE interface. On the left, the Project Explorer view lists three files: PrintName.java, SumOfAllEvenNumbers.java, and SwapWithoutTemp.java. The PrintName.java file is open in the editor, displaying the following code:

```
1 package in.globalquesttechnologies;
2
3 public class PrintName {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7
8         System.out.println("Lavaraju Valabam");
9     }
10
11 }
12
```

On the right, the Console view shows the output of the application: "Lavaraju Valabam".

```
package in.globalquesttechnologies;
public class PrintName {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println("Lavaraju Valabam");
    }
}
```

2. Write a program to print the sum of all even numbers from 1 to 100.



The screenshot shows the Eclipse IDE interface. On the left, the Project Explorer view lists three files: PrintName.java, SumOfAllEvenNumbers.java, and SwapWithoutTemp.java. The SumOfAllEvenNumbers.java file is open in the editor, displaying the following code:

```
1 package in.globalquesttechnologies;
2
3 public class SumOfAllEvenNumbers {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         int sum = 0;
8         for(int i=2; i<=100; i+=2) {
9             sum += i;
10        }
11    }
12
13 }
14
```

On the right, the Console view shows the output of the application: "Sum of the all even numbers from 1 to 100 : 2550".

```
package in.globalquesttechnologies;
public class SumOfAllEvenNumbers {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int sum = 0;
        for(int i=2; i<=100; i+=2) {
            sum += i;
        }
        System.out.println("Sum of the all even numbers from 1 to 100 : " + sum);
    }
}
```

Write a program to

1. Write a program swap two numbers without using a temporary variable.

The screenshot shows the Eclipse IDE interface with the SwapWithoutTemp.java file open in the editor. The code implements a swap operation using arithmetic assignments. The console tab shows the execution of the program, where it prompts for two numbers (14 and 22), performs the swap using only one temporary variable (a), and then prints the swapped values (22 and 14).

```
1 package in.globalquesttechnologies;
2 import java.util.Scanner;
3 public class SwapWithoutTemp {
4     public static void main(String[] args) {
5         // TODO Auto-generated method stub
6         Scanner scan = new Scanner(System.in);
7         System.out.print("Enter the first number (a) : ");
8         int a = scan.nextInt();
9         System.out.print("Enter the second number (b) : ");
10        int b = scan.nextInt();
11        System.out.println("Before swapping :");
12        System.out.println("a = " + a + ", b = " + b);
13        a = a + b;
14        b = a - b;
15        a = a - b;
16        System.out.println("After swapping :");
17        System.out.println("a = " + a + ", b = " + b);
18        scan.close();
19    }
20}
```

```
<terminated> SwapWithoutTemp [Java Application] C:\Users\abhay\Downloads\eclipse-jee-2024-12-R-win32-x86_64\SwapWithoutTemp\src\in\globalquesttechnologies\SwapWithoutTemp.java
Enter the first number (a) : 14
Enter the second number (b) : 22
Before swapping :
a = 14, b = 22
After swapping :
a = 22, b = 14
```

```
package in.globalquesttechnologies;
import java.util.Scanner;
public class SwapWithoutTemp {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner scan = new Scanner(System.in);
        System.out.print("Enter the first number (a) : ");
        int a = scan.nextInt();
        System.out.print("Enter the second number (b) : ");
        int b = scan.nextInt();
        System.out.println("Before swapping :");
        System.out.println("a = " + a + ", b = " + b);
        a = a + b;
        b = a - b;
        a = a - b;
        System.out.println("After swapping :");
        System.out.println("a = " + a + ", b = " + b);
        scan.close();
    }
}
```

4. Write a program to check whether a given number is prime or not.

The screenshot shows the Eclipse IDE interface with the CheckPrime.java file open in the editor. The code uses a for loop to check if a given number is divisible by any integer from 2 to half of the number. If it finds a divisor, it sets a flag to false and breaks the loop. Finally, it prints whether the number is prime or not based on the value of the flag.

```
1 package in.globalquesttechnologies;
2 import java.util.Scanner;
3 public class CheckPrime {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         System.out.print("Enter a number : ");
7         int num = sc.nextInt();
8         boolean isPrime = true;
9         if (num <= 1) {
10             isPrime = false;
11         } else {
12             for (int i = 2; i <= num / 2; i++) {
13                 if (num % i == 0) {
14                     isPrime = false;
15                     break;
16                 }
17             }
18         }
19         if (isPrime) {
20             System.out.println(num + " is a Prime Number");
21         } else {
22             System.out.println(num + " is NOT a Prime Number");
23         }
24     }
25     sc.close();
26 }
27 }
```

```
<terminated> CheckPrime [Java Application] C:\Users\abhay\Downloads\eclipse-jee-2024-12-R-win32-x86_64\CheckPrime\src\in\globalquesttechnologies\CheckPrime.java
Enter a number : 5
5 is a Prime Number
```

```
package in.globalquesttechnologies;
import java.util.Scanner;
public class CheckPrime {
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter a number : ");
    int num = sc.nextInt();
    boolean isPrime = true;
```

```

if (num <= 1) {
    isPrime = false;
}
else {
    for (int i = 2; i <= num / 2; i++) {
        if (num % i == 0) {
            isPrime = false;
            break;
        }
    }
}
if (isPrime) {
    System.out.println(num + " is a Prime Number");
}
else {
    System.out.println(num + " is NOT a Prime Number");
}
sc.close();
}
}

```

5. Write a program to calculate the factorial of a given number using recursion.

The screenshot shows the Eclipse IDE interface. The left pane displays several Java files in the package explorer, including PrintName.java, SumOfAllEvenNumbers.java, SwapWithoutTemp.java, CheckPrime.java, and factorialJava.java. The factorialJava.java file is currently selected and its code is visible in the editor:

```

File Edit Source Refactor Navigate Search Project Run Window Help
PrintName.java SumOfAllEvenNumbers.java SwapWithoutTemp.java CheckPrime.java factorialJava X
package in.globalquesttechnologies;
import java.util.Scanner;
public class factorial {
    static int factorial(int n) {
        if (n == 0 || n == 1) {
            return 1;
        }
        return n * factorial(n - 1);
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num = sc.nextInt();
        int result = factorial(num);
        System.out.println("Factorial of " + num + " is: " + result);
        sc.close();
    }
}

```

The right pane shows the Eclipse interface with tabs for Problems, Servers, Terminal, Data Sou..., Properties, and Console. The Console tab is active, displaying the output of the application's execution:

```

<terminated> factorial [Java Application] C:\Users\abhay\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ecip
Enter a number: 5
Factorial of 5 is: 120

```

```

package in.globalquesttechnologies;
import java.util.Scanner;
public class factorial {
    static int factorial(int n) {
        if (n == 0 || n == 1) {
            return 1;
        }
        return n * factorial(n - 1);
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num = sc.nextInt();
        int result = factorial(num);
        System.out.println("Factorial of " + num + " is: " + result);

        sc.close();
    }
}

```

6. Write a program to find the roots of a quadratic equation.

The screenshot shows the Eclipse IDE interface with the Roots.java file open in the editor. The code implements a quadratic equation solver. In the terminal view, the user enters values for a, b, and c, and the program outputs the roots. The roots are complex numbers: Root 1 = -0.75 + 1.5612494995995996i and Root 2 = -0.75 - 1.5612494995995996i.

```
1 package in.globalquesttechnologies;
2 import java.util.Scanner;
3 public class Roots {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         System.out.print("Enter value of a: ");
7         double a = sc.nextDouble();
8         System.out.print("Enter value of b: ");
9         double b = sc.nextDouble();
10        System.out.print("Enter value of c: ");
11        double c = sc.nextDouble();
12        // discriminant
13        double d = b * b - 4 * a * c;
14        if (a == 0) {
15            System.out.println("Not a quadratic equation (a cannot be 0)");
16        } else if (d > 0) {
17            // two real and distinct roots
18            double r1 = (-b + Math.sqrt(d)) / (2 * a);
19            double r2 = (-b - Math.sqrt(d)) / (2 * a);
20            System.out.println("Roots are real and different");
21            System.out.println("Root 1 = " + r1);
22            System.out.println("Root 2 = " + r2);
23        } else if (d == 0) {
24            // two real and equal roots
25            double r = -b / (2 * a);
26            System.out.println("Roots are real and equal");
27            System.out.println("Root = " + r);
28        } else {
29            // complex roots
30            double realPart = -b / (2 * a);
31            double imaginaryPart = Math.sqrt(-d) / (2 * a);
32            System.out.println("Roots are complex");
33            System.out.println("Root 1 = " + realPart + " + " + imaginaryPart + "i");
34            System.out.println("Root 2 = " + realPart + " - " + imaginaryPart + "i");
35        }
36        sc.close();
37    }
38 }
```

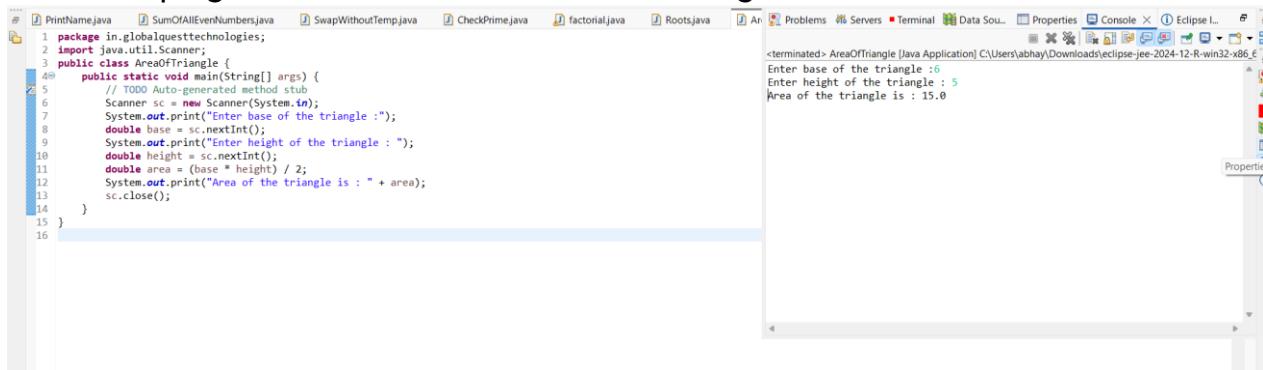
```
package in.globalquesttechnologies;
import java.util.Scanner;
public class Roots {
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter value of a: ");
    double a = sc.nextDouble();
    System.out.print("Enter value of b: ");
    double b = sc.nextDouble();
    System.out.print("Enter value of c: ");
    double c = sc.nextDouble();
    // discriminant
    double d = b * b - 4 * a * c;
    if (a == 0) {
        System.out.println("Not a quadratic equation (a cannot be 0)");
    } else if (d > 0) {
        // two real and distinct roots
        double r1 = (-b + Math.sqrt(d)) / (2 * a);
        double r2 = (-b - Math.sqrt(d)) / (2 * a);
        System.out.println("Roots are real and different");
        System.out.println("Root 1 = " + r1);
        System.out.println("Root 2 = " + r2);
    } else if (d == 0) {
        // two real and equal roots
        double r = -b / (2 * a);
        System.out.println("Roots are real and equal");
        System.out.println("Root = " + r);
    } else {
        // complex roots
        double realPart = -b / (2 * a);
        double imaginaryPart = Math.sqrt(-d) / (2 * a);
        System.out.println("Roots are complex");
        System.out.println("Root 1 = " + realPart + " + " + imaginaryPart + "i");
        System.out.println("Root 2 = " + realPart + " - " + imaginaryPart + "i");
    }
}
```

```

        System.out.println("Root 2 = " + realPart + " - " + imaginaryPart + "i");
    }
    sc.close();
}
}

```

7. Write a program to Calculate the area of a triangle.



```

1 package in.globalquesttechnologies;
2 import java.util.Scanner;
3 public class AreaOfTriangle {
4     public static void main(String[] args) {
5         // TODO Auto-generated method stub
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter base of the triangle : ");
8         double base = sc.nextInt();
9         System.out.print("Enter height of the triangle : ");
10        double height = sc.nextInt();
11        double area = (base * height) / 2;
12        System.out.print("Area of the triangle is : " + area);
13        sc.close();
14    }
15 }

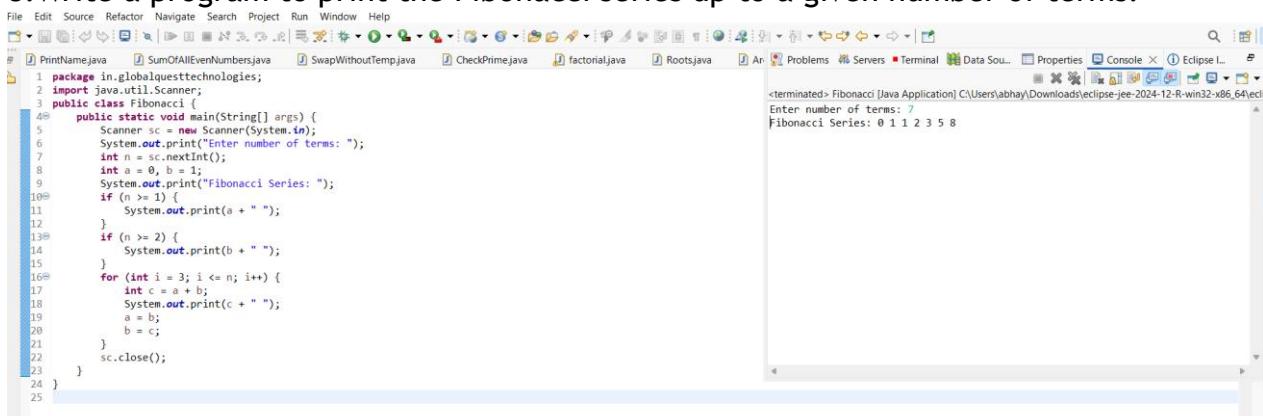
```

```

package in.globalquesttechnologies;
import java.util.Scanner;
public class AreaOfTriangle {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter base of the triangle : ");
        double base = sc.nextInt();
        System.out.print("Enter height of the triangle : ");
        double height = sc.nextInt();
        double area = (base * height) / 2;
        System.out.print("Area of the triangle is : " + area);
        sc.close();
    }
}

```

8. Write a program to print the Fibonacci series up to a given number of terms.



```

1 package in.globalquesttechnologies;
2 import java.util.Scanner;
3 public class Fibonacci {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         System.out.print("Enter number of terms: ");
7         int n = sc.nextInt();
8         int a = 0, b = 1;
9         System.out.print("Fibonacci Series: ");
10        if (n >= 1) {
11            System.out.print(a + " ");
12        }
13        if (n >= 2) {
14            System.out.print(b + " ");
15        }
16        for (int i = 3; i <= n; i++) {
17            int c = a + b;
18            System.out.print(c + " ");
19            a = b;
20            b = c;
21        }
22        sc.close();
23    }
24 }

```

```

package in.globalquesttechnologies;
import java.util.Scanner;
public class Fibonacci {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of terms: ");
        int n = sc.nextInt();
        int a = 0, b = 1;
        System.out.print("Fibonacci Series: ");

```

```

if (n >= 1) {
    System.out.print(a + " ");
}
if (n >= 2) {
    System.out.print(b + " ");
}
for (int i = 3; i <= n; i++) {
    int c = a + b;
    System.out.print(c + " ");
    a = b;
    b = c;
}
sc.close();
}
}

```

9. find the second largest number in an array

The screenshot shows the Eclipse IDE interface with the following details:

- Java Editor:** Displays the source code for a class named `ScndLargestNum`. The code uses a Scanner to read the size of the array and its elements, then iterates through the array to find the second largest number.
- Console Output:** Shows the terminal window with the following interaction:


```
<terminated> ScndLargestNum [Java Application] C:\Users\abhay\Downloads\eclipse-jee-2024-12-R-win32-x86
Enter size of array: 5
Enter array elements:
10
20
30
40
50
Second largest number is: 40
```

```

package in.globalquesttechnologies;
import java.util.Scanner;
public class ScndLargestNum {
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter size of array: ");
    int n = sc.nextInt();
    int[] arr = new int[n];
    System.out.println("Enter array elements:");
    for (int i = 0; i < n; i++) {
        arr[i] = sc.nextInt();
    }
    int largest = arr[0];
    int secondLargest = Integer.MIN_VALUE;
    for (int i = 0; i < n; i++) {
        if (arr[i] > largest) {
            secondLargest = largest;
            largest = arr[i];
        } else if (arr[i] > secondLargest && arr[i] != largest) {
            secondLargest = arr[i];
        }
    }
    if (secondLargest == Integer.MIN_VALUE) {
        System.out.println("No second largest element found");
    } else {

```

```

        System.out.println("Second largest number is: " + secondLargest);
    }
    sc.close();
}
}

```

10. Write a program to reverse a string

The screenshot shows the Eclipse IDE interface with several tabs at the top: PrintName.java, SumOfAllEvenNumbers..., SwapWithoutTemp.java, CheckPrime.java, factorial.java, Roots.java, Problems, Servers, Terminal, Data Sou..., Properties, Console, and Eclipse L... . The code editor contains a Java class ReverseString:

```

1 package in.globalquesttechnologies;
2 //import java.util.Scanner;
3
4 public class ReverseString {
5
6     public static void main(String[] args) {
7         String str = "hello";
8         String reversed = new StringBuilder(str).reverse().toString();
9         System.out.println("Reversed string: " + reversed);
10    }
11
12 }
13
14 }
15

```

In the terminal view, the output is shown as:

```

<terminated> ReverseString [Java Application] C:\Users\abhay\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Reversed string: olleh

```

```

package in.globalquesttechnologies;
//import java.util.Scanner;
public class ReverseString {
public static void main(String[] args) {
    String str = "hello";
    String reversed = new StringBuilder(str).reverse().toString();
    System.out.println("Reversed string: " + reversed);
}
}

```

11. write a program to check if a given string is a palindrome.

The screenshot shows the Eclipse IDE interface with several tabs at the top: PrintName.java, SumOfAllEvenNumbers..., SwapWithoutTemp.java, CheckPrime.java, factorial.java, Roots.java, Problems, Servers, Terminal, Data Sou..., Properties, Console, and Eclipse L... . The code editor contains a Java class Palindrome:

```

1 package in.globalquesttechnologies;
2 import java.util.Scanner;
3
4 public class Palindrome {
5
6     public static void main(String[] args) {
7         Scanner sc = new Scanner(System.in);
8         System.out.print("Enter a string: ");
9         String str = sc.nextLine();
10        String rev = "";
11        for (int i = str.length() - 1; i >= 0; i--) {
12            rev = rev + str.charAt(i);
13        }
14        if (str.equals(rev)) {
15            System.out.println(str + " is a Palindrome");
16        } else {
17            System.out.println(str + " is NOT a Palindrome");
18        }
19    }
20

```

In the terminal view, the output is shown as:

```

<terminated> Palindrome [Java Application] C:\Users\abhay\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter a string: madam
madam is a Palindrome

```

```

package in.globalquesttechnologies;
import java.util.Scanner;
public class Palindrome {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String str = sc.nextLine();
        String rev = "";
        for (int i = str.length() - 1; i >= 0; i--) {
            rev = rev + str.charAt(i);
        }
        if (str.equals(rev)) {
            System.out.println(str + " is a Palindrome");
        } else {
            System.out.println(str + " is NOT a Palindrome");
        }
    }
}

```

```

        sc.close();
    }
}

```

12. Write a program to count the occurrences of a character in a string

The screenshot shows the Eclipse IDE interface. On the left, the code editor displays a Java file named CountCh.java. The code reads a string "good morning", prompts the user to enter a character, and then counts how many times that character appears in the string. The output window on the right shows the application's terminal output: "Enter the character to count: o", followed by "Character 'o' occurs 3 times".

```

1 package in.globalquesttechnologies;
2 import java.util.Scanner;
3 public class CountCh {
4     public static void main(String[] args) {
5         String str = "good morning";
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter the character to count: ");
8         char ch = sc.next().charAt(0);
9         int count = 0;
10        for (int i = 0; i < str.length(); i++) {
11            if (str.charAt(i) == ch) {
12                count++;
13            }
14        }
15        System.out.println("Character '" + ch + "' occurs " + count + " times");
16        sc.close();
17    }
18 }

```

```

package in.globalquesttechnologies;
import java.util.Scanner;
public class CountCh {
    public static void main(String[] args) {
        String str = "good morning";
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the character to count: ");
        char ch = sc.next().charAt(0);
        int count = 0;
        for (int i = 0; i < str.length(); i++) {
            if (str.charAt(i) == ch) {
                count++;
            }
        }
        System.out.println("Character '" + ch + "' occurs " + count + " times");
        sc.close();
    }
}

```

13. Write a program to find the GCD (Greatest Common Divisor) of two numbers

The screenshot shows the Eclipse IDE interface. On the left, the code editor displays a Java file named GcdOfTwoNum.java. The code defines a recursive method to calculate the GCD of two integers using the Euclidean algorithm. The main method calls this function with parameters 36 and 60, and prints the result "GCD is: 12". The output window on the right shows the application's terminal output: "GCD is: 12".

```

1 package in.globalquesttechnologies;
2 public class GcdOfTwoNum {
3     static int gcd(int a, int b) {
4         if (b == 0)
5             return a;
6         return gcd(b, a % b);
7     }
8     public static void main(String[] args) {
9         System.out.println("GCD is: " + gcd(36, 60));
10    }
11 }

```

```

package in.globalquesttechnologies;
public class GcdOfTwoNum {
    static int gcd(int a, int b) {
        if (b == 0)
            return a;
        return gcd(b, a % b);
    }
}

```

```

public static void main(String[] args) {
    System.out.println("GCD is: " + gcd(36, 60));
}
}

```

14. Write a program to convert a decimal number to binary.

The screenshot shows the Eclipse IDE interface with the 'DecimalToBinary.java' file open in the editor. The code converts a decimal number to binary. In the 'Console' view, the output is 'Binary equivalent of 13 is: 1101'.

```

package in.globalquesttechnologies;
public class DecimalToBinary {
    public static void main(String[] args) {
        int num = 13;
        int original = num;
        String binary = "";
        if (num == 0) {
            binary = "0";
        } else {
            while (num > 0) {
                int remainder = num % 2;
                binary = remainder + binary;
                num = num / 2;
            }
        }
        System.out.println("Binary equivalent of " + original + " is: " + binary);
    }
}

```

```

package in.globalquesttechnologies;
public class DecimalToBinary {
    public static void main(String[] args) {
        int num = 13;
        int original = num;
        String binary = "";
        if (num == 0) {
            binary = "0";
        } else {
            while (num > 0) {
                int remainder = num % 2;
                binary = remainder + binary;
                num = num / 2;
            }
        }
        System.out.println("Binary equivalent of " + original + " is: " + binary);
    }
}

```

15. Write a program to calculate the sum of digits of a given number.

The screenshot shows the Eclipse IDE interface with the 'SumOfDigits.java' file open in the editor. The code calculates the sum of digits of a given number. In the 'Console' view, the output is 'Enter a number : 67' and 'Sum of digits of 67 is : 13'.

```

package in.globalquesttechnologies;
import java.util.Scanner;
public class SumOfDigits {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.print("Enter a number : ");
        int num = scan.nextInt();
        int sum = 0;
        int number = num;
        while (num != 0) {
            int digit = num % 10;
            sum = sum + digit;
            num = num / 10;
        }
        System.out.println("Sum of digits of " + number + " is : " + sum);
        scan.close();
    }
}

```

```

package in.globalquesttechnologies;
import java.util.Scanner;
public class SumOfDigits {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.print("Enter a number : ");
        int num = scan.nextInt();

```

```

int sum = 0;
int number = num;
while (num != 0) {
    int digit = num % 10;
    sum = sum + digit;
    num = num / 10;
}
System.out.println("Sum of digits of " + number + " is : " + sum);
scan.close();
}

}

```

16. Write a program to check whether a given year is a leap year or not.

```

1 package in.globalquesttechnologies;
2 import java.util.Scanner;
3 public class LeapYearCheck {
4     public static void main(String[] args) {
5         Scanner scan = new Scanner(System.in);
6         System.out.print("Enter a year : ");
7         int year = scan.nextInt();
8         if((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {
9             System.out.println(year + " is a Leap year");
10        } else {
11            System.out.println(year + " is not a Leap year");
12        }
13        scan.close();
14    }
15 }
16
17 }
18

```

```

package in.globalquesttechnologies;
import java.util.Scanner;
public class LeapYearCheck {
public static void main(String[] args) {
    Scanner scan = new Scanner(System.in);
    System.out.print("Enter a year : ");
    int year = scan.nextInt();
    if((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {
        System.out.println(year + " is a Leap year");
    }
    else {
        System.out.println(year + " is not a Leap year");
    }
    scan.close();
}
}

```

17. Write a program to find the factorial of a number using iteration.

```

1 package in.globalquesttechnologies;
2 import java.util.Scanner;
3 public class FactorialUsingIteration {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         System.out.print("Enter a number : ");
7         int num = sc.nextInt();
8         long factorial = 1;
9
10        //iterative approach
11        for(int i=1; i<=num; i++) {
12            factorial = factorial * i;
13        }
14        System.out.println("Factorial of " + num + " is : " + factorial);
15        sc.close();
16    }
17 }
18

```

```

package in.globalquesttechnologies;
import java.util.Scanner;
public class FactorialUsingIteration {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number : ");
        int num = sc.nextInt();
        long factorial = 1;

        //iterative approach
        for(int i=1; i<=num; i++) {
            factorial = factorial * i;
        }
        System.out.println("Factorial of " + num + " is : " + factorial);
        sc.close();
    }
}

```

18. Write a program to print the Pascal's triangle.

The screenshot shows the Eclipse IDE interface with the code for PascalsTriangle.java in the editor. The code uses nested loops to print a Pascal's triangle of size n=5. The terminal window shows the printed output:

```

1
1 1
1 2 1
1 3 3 1
1 4 6 4 1

```

```

package in.globalquesttechnologies;
public class PascalsTriangle {
    public static void main(String[] args) {
        int n = 5;
        for (int i = 0; i < n; i++) {
            for (int k = 1; k < n - i; k++) {
                System.out.print(" ");
            }
            int number = 1;
            // Print numbers
            for (int j = 0; j <= i; j++) {
                System.out.print(number + " ");
                number = number * (i - j) / (j + 1);
            }
            System.out.println();
        }
    }
}

```

19. Write a program to check if a number is Armstrong or not.

The screenshot shows the Eclipse IDE interface with the code editor open. The code is for an Armstrong number checker. The console output shows the result for the number 153.

```
1 package in.globalquesttechnologies;
2 public class ArmstrongNumberCheck {
3     public static void main(String[] args) {
4         int number = 153;
5         boolean isArmstrong = checkArmstrongNumber(number);
6         if(isArmstrong) {
7             System.out.println(number + " is an Armstrong number");
8         } else {
9             System.out.println(number + " is not an Armstrong number");
10        }
11    }
12 }
13 private static boolean checkArmstrongNumber(int number) {
14     int originalNumber = number;
15     int numberOfDigits = (int)Math.Log10(number) + 1;
16     int sum = 0;
17     while(number > 0) {
18         int digit = number % 10;
19         sum += Math.pow(digit, numberOfDigits);
20         number /= 10;
21     }
22     return sum == originalNumber;
23 }
```

<terminated> ArmstrongNumberCheck [Java Application] C:\Users\abhay\Downloads\eclipse-jee-2024-12-R-wii
153 is an Armstrong number

```
package in.globalquesttechnologies;
public class ArmstrongNumberCheck {
    public static void main(String[] args) {
        int number = 153;
        boolean isArmstrong = checkArmstrongNumber(number);
        if(isArmstrong) {
            System.out.println(number + " is an Armstrong number");
        } else {
            System.out.println(number + " is not an Armstrong number");
        }
    }
    private static boolean checkArmstrongNumber(int number) {
        int originalNumber = number;
        int numberOfDigits = (int)Math.log10(number) + 1;
        int sum = 0;
        while(number > 0) {
            int digit = number % 10;
            sum += Math.pow(digit, numberOfDigits);
            number /= 10;
        }
        return sum == originalNumber;
    }
}
```

20. Write a program to find the area and perimeter of a circle.

The screenshot shows the Eclipse IDE interface with the code editor open. The code is for calculating the area and perimeter of a circle with a radius of 7. The console output shows the results.

```
1 package in.globalquesttechnologies;
2 public class AreaAndPerimeterOfCircle {
3     public static void main(String[] args) {
4         double radius = 7;
5         double area = Math.PI * radius * radius;
6         double perimeter = 2 * Math.PI * radius;
7         System.out.println("Area of the circle = " + area);
8         System.out.println("Perimeter of the circle = " + perimeter);
9     }
10 }
11 
```

<terminated> AreaAndPerimeterOfCircle [Java Application] C:\Users\abhay\Downloads\eclipse-jee-2024-12-R-v
Area of the circle = 153.938040258985
Perimeter of the circle = 43.982297150257104

```
package in.globalquesttechnologies;
public class AreaAndPerimeterOfCircle {
    public static void main(String[] args) {
        double radius = 7;
        double area = Math.PI * radius * radius;
        double perimeter = 2 * Math.PI * radius;
```

```

        System.out.println("Area of the circle = " + area);
        System.out.println("Perimeter of the circle = " + perimeter);
    }
}

```

21. Write a program to print the multiplication table of a given number.

The screenshot shows the Eclipse IDE interface with the code editor containing the following Java code:

```

1 package in.globalquesttechnologies;
2 import java.util.Scanner;
3 public class MultiplicationTable {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         System.out.print("Enter a number : ");
7         int num = sc.nextInt();
8         System.out.println("multiplication Table of " + num + " : ");
9         for(int i=1; i<=10; i++) {
10             System.out.println(num + " x " + i + " = " + (num * i));
11         }
12     }
13 }
14
15

```

The output window shows the multiplication table for the number 5:

```

<terminated> MultiplicationTable [Java Application] C:\Users\abhay\Downloads\eclipse-jee-2024-12-R-win32\x
Enter a number : 5
multiplication Table of 5 :
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50

```

```

package in.globalquesttechnologies;
import java.util.Scanner;
public class MultiplicationTable {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number : ");
        int num = sc.nextInt();
        System.out.println("multiplication Table of " + num + " : ");
        for(int i=1; i<=10; i++) {
            System.out.println(num + " x " + i + " = " + (num * i));
        }
        sc.close();
    }
}

```

22. Write a program to find the sum of all elements in an array.

The screenshot shows the Eclipse IDE interface with the code editor containing the following Java code:

```

1 package in.globalquesttechnologies;
2 public class ArraySum {
3     public static void main(String[] args) {
4         int[] array ={1,2,3,4,5};
5         int sum = 0;
6         for(int i=0; i<array.length; i++) {
7             sum +=array[i];
8         }
9         System.out.println("Sum of elements in the array : " + sum);
10    }
11 }
12

```

The output window shows the sum of the array elements:

```

<terminated> ArraySum [Java Application] C:\Users\abhay\Downloads\eclipse-jee-2024-12-R-win32\x86_64\ec
Sum of elements in the array : 15

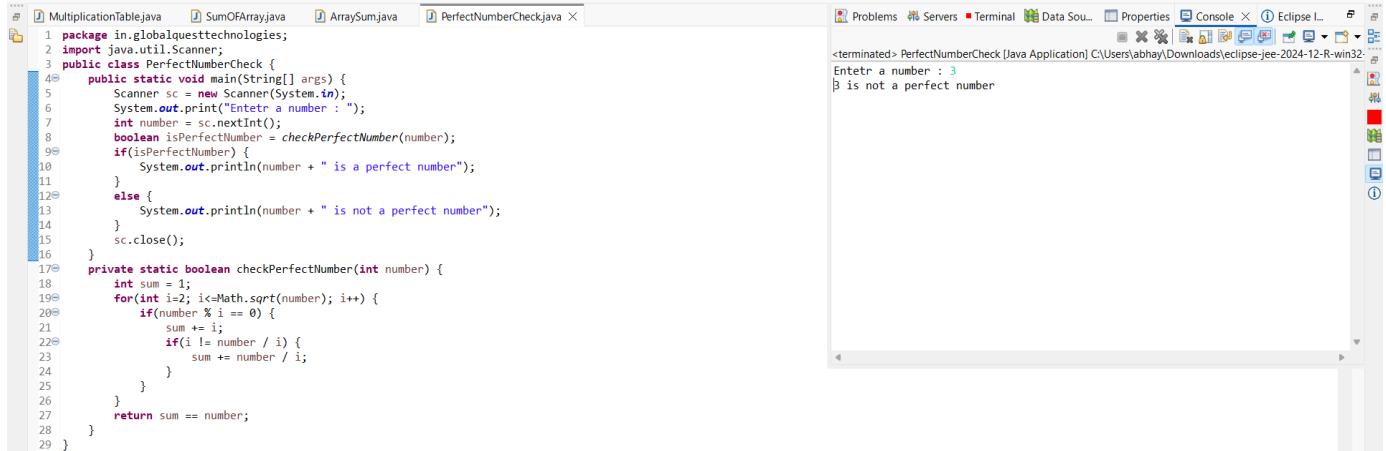
```

```

package in.globalquesttechnologies;
public class ArraySum {
    public static void main(String[] args) {
        int[] array ={1,2,3,4,5};
        int sum = 0;
        for(int i=0; i<array.length; i++) {
            sum +=array[i];
        }
        System.out.println("Sum of elements in the array : " + sum);
    }
}

```

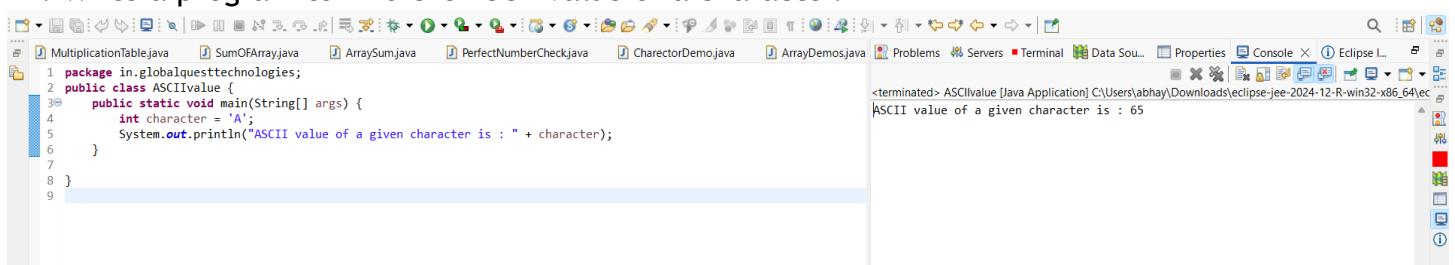
23. Write a program to check if a given number is a perfect number.



```
package in.globalquesttechnologies;
import java.util.Scanner;
public class PerfectNumberCheck {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number : ");
        int number = sc.nextInt();
        boolean isPerfectNumber = checkPerfectNumber(number);
        if(isPerfectNumber) {
            System.out.println(number + " is a perfect number");
        } else {
            System.out.println(number + " is not a perfect number");
        }
        sc.close();
    }
    private static boolean checkPerfectNumber(int number) {
        int sum = 1;
        for(int i=2; i<Math.sqrt(number); i++) {
            if(number % i == 0) {
                sum += i;
                if(i != number / i) {
                    sum += number / i;
                }
            }
        }
        return sum == number;
    }
}
```

The screenshot shows the Eclipse IDE interface with the Java code for checking perfect numbers. The code uses a scanner to read an integer from the user and a private method to calculate the sum of divisors. If the sum equals the original number, it's a perfect number. The console output shows the program running and correctly identifying 3 as not being a perfect number.

24. Write a program to find the ASCII value of a character.



```
package in.globalquesttechnologies;
public class ASCIIValue {
    public static void main(String[] args) {
        int character = 'A';
        System.out.println("ASCII value of a given character is : " + character);
    }
}
```

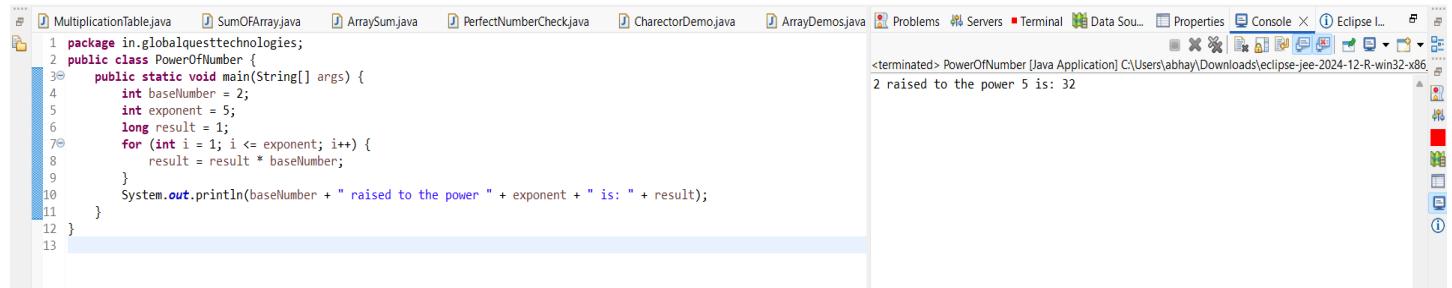
The screenshot shows the Eclipse IDE interface with the Java code for finding ASCII values. The code reads a character ('A') and prints its ASCII value. The console output shows the program running and correctly identifying the ASCII value of 'A' as 65.

```

package in.globalquesttechnologies;
public class ASCIIvalue {
    public static void main(String[] args) {
        int character = 'A';
        System.out.println("ASCII value of a given character is : " + character);
    }
}

```

25. Write a program to calculate the power of a number.



The screenshot shows the Eclipse IDE interface with the PowerOfNumber.java file open in the editor. The code calculates 2 raised to the power of 5, resulting in 32. The output is displayed in the Console view.

```

package in.globalquesttechnologies;
public class PowerOfNumber {
    public static void main(String[] args) {
        int baseNumber = 2;
        int exponent = 5;
        long result = 1;
        for (int i = 1; i <= exponent; i++) {
            result = result * baseNumber;
        }
        System.out.println(baseNumber + " raised to the power " + exponent + " is: " + result);
    }
}

```

<terminated> PowerOfNumber [Java Application] C:\Users\abhay\Downloads\eclipse-jee-2024-12-R-win32-x86_64\PowerOfNumber\src\in\globalquesttechnologies\PowerOfNumber.java
2 raised to the power 5 is: 32

```

package in.globalquesttechnologies;
public class PowerOfNumber {
    public static void main(String[] args) {
        int baseNumber = 2;
        int exponent = 5;
        long result = 1;
        for (int i = 1; i <= exponent; i++) {
            result = result * baseNumber;
        }
        System.out.println(baseNumber + " raised to the power " + exponent + " is: " + result);
    }
}

```

operators

1. Write a program to perform arithmetic operations (+, -, *, /) on two numbers.

The screenshot shows the Eclipse IDE interface with the ArithmeticOperations.java file open in the editor. The code defines a class with a main method that performs arithmetic operations on two integers (a=10, b=20) and prints the results. The output window shows the results of addition, subtraction, multiplication, and division.

```
package in.globalquesttechnologies;
public class ArithmeticOperations {
    public static void main(String[] args) {
        int a = 10;
        int b = 20;
        System.out.println("Arithmetic Operations on two numbers :");
        System.out.println("-----");
        System.out.println(a + " + " + b + " = " + (a+b));
        System.out.println(a + " - " + b + " = " + (a-b));
        System.out.println(a + " * " + b + " = " + (a*b));
        System.out.println(a + " / " + b + " = " + (a/b));
    }
}
```

```
<terminated> ArithmeticOperations [Java Application] C:\Users\abhay\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ArithmeticOperations
Arithmetic Operations on two numbers :
-----
10 + 20 = 30
10 - 20 = -10
10 * 20 =200
10 / 20 = 0
```

```
package in.globalquesttechnologies;
public class ArithmeticOperations {
    public static void main(String[] args) {
        int a = 10;
        int b = 20;
        System.out.println("Arithmetic Operations on two numbers :");
        System.out.println("-----");
        System.out.println(a + " + " + b + " = " + (a+b));
        System.out.println(a + " - " + b + " = " + (a-b));
        System.out.println(a + " * " + b + " = " + (a*b));
        System.out.println(a + " / " + b + " = " + (a/b));
    }
}
```

2. Write a program to perform bitwise AND, OR, and XOR operations on two integers.

The screenshot shows the Eclipse IDE interface with the BitwiseOperations.java file open in the editor. The code defines a class with a main method that performs bitwise operations on two integers (a=1, b=2) and prints the results. The output window shows the results of bitwise AND, OR, and XOR.

```
package in.globalquesttechnologies;
public class BitwiseOperations {
    public static void main(String[] args) {
        int num1 = 1;
        int num2 = 2;
        int andResult = num1 & num2;
        int orResult = num1 | num2;
        int xorResult = num1 ^ num2;
        System.out.println("Bitwise AND (a & b): " + andResult);
        System.out.println("Bitwise OR (a | b): " + orResult);
        System.out.println("Bitwise XOR (a ^ b): " + xorResult);
    }
}
```

```
<terminated> BitwiseOperations [Java Application] C:\Users\abhay\Downloads\eclipse-jee-2024-12-R-win32-x86_64\BitwiseOperations
Bitwise AND (a & b): 0
Bitwise OR (a | b): 3
Bitwise XOR (a ^ b): 3
```

```
package in.globalquesttechnologies;
public class BitwiseOperations {
    public static void main(String[] args) {
        int num1 = 1;
        int num2 = 2;
        int andResult = num1 & num2;
        int orResult = num1 | num2;
        int xorResult = num1 ^ num2;
        System.out.println("Bitwise AND (a & b): " + andResult);
        System.out.println("Bitwise OR (a | b): " + orResult);
        System.out.println("Bitwise XOR (a ^ b): " + xorResult);
    }
}
```

3. Write a program to check whether a given number is positive, negative, or zero.

The screenshot shows the Eclipse IDE interface with the 'CheckNumber.java' file open in the editor. The code uses a Scanner to read an integer from the user and then prints out whether it is Positive, Negative, or Zero. The console output shows the program running and printing 'The number is Positive' for the input value 5.

```
package in.globalquesttechnologies;
import java.util.Scanner;
public class CheckNumber {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num = sc.nextInt();
        if (num > 0) {
            System.out.println("The number is Positive");
        } else if (num < 0) {
            System.out.println("The number is Negative");
        } else {
            System.out.println("The number is Zero");
        }
        sc.close();
    }
}
```

```
package in.globalquesttechnologies;
import java.util.Scanner;
public class CheckNumber {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num = sc.nextInt();
        if (num > 0) {
            System.out.println("The number is Positive");
        } else if (num < 0) {
            System.out.println("The number is Negative");
        } else {
            System.out.println("The number is Zero");
        }
        sc.close();
    }
}
```

4. Write a program to swap two numbers using bitwise XOR operator.

The screenshot shows the Eclipse IDE interface with the 'SwapUsingXOR.java' file open in the editor. The code swaps two integers, a and b, using the XOR operator (^). It prints the values before and after the swap. The console output shows the initial values a=5, b=3, and the swapped values a=3, b=5.

```
package in.globalquesttechnologies;
public class SwapUsingXOR {
    public static void main(String[] args) {
        int a = 5;
        int b = 3;
        System.out.println("Before swapping :");
        System.out.println("a = " + a + ", b = " + b);
        a = a ^ b;
        b = a ^ b;
        a = a ^ b;
        System.out.println("After swapping :");
        System.out.println("a = " + a + ", b = " + b);
    }
}
```

```
package in.globalquesttechnologies;
public class SwapUsingXOR {
    public static void main(String[] args) {
        int a = 5;
        int b = 3;
        System.out.println("Before swapping :");
        System.out.println("a = " + a + ", b = " + b);
        a = a ^ b;
        b = a ^ b;
        a = a ^ b;
        System.out.println("After swapping :");
        System.out.println("a = " + a + ", b = " + b);
    }
}
```

```
}
```

5. Write a program to calculate the area of a circle using the radius entered by the user.

```
1 package in.globalquesttechnologies;
2 import java.util.Scanner;
3 public class CircleArea {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         System.out.print("Enter radius of the circle : ");
7         double radius = sc.nextInt();
8         double area = Math.PI * radius * radius ;
9         System.out.println("Area of the circle : " + area);
10        sc.close();
11    }
12 }
```

```
<terminated> CircleArea [Java Application] C:\Users\abhay\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Enter radius of the circle : 2
Area of the circle : 12.566370614359172
```

```
package in.globalquesttechnologies;
import java.util.Scanner;
public class CirlceArea {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter radius of the circle : ");
        double radius = sc.nextInt();
        double area = Math.PI * radius * radius ;
        System.out.println("Area of the circle : " + area);
        sc.close();
    }
}
```

6. Write a program to convert temperature from Fahrenheit to Celsius.

```
1 package in.globalquesttechnologies;
2 public class FahrenheitToCelciusConversion {
3     public static void main(String[] args) {
4         double fahrenheit = 98.6;
5         double celcius = (fahrenheit - 32) * 5 / 9;
6         System.out.println(fahrenheit + " degrees fahrenheit is equal to " + celcius + " degrees Celcius");
7     }
8 }
```

```
<terminated> FahrenheitToCelciusConversion [Java Application] C:\Users\abhay\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
98.6 degrees fahrenheit is equal to 37.0 degrees Celcius
```

```
package in.globalquesttechnologies;
public class FahrenheitToCelciusConversion {
    public static void main(String[] args) {
        double fahrenheit = 98.6;
        double celcius = (fahrenheit - 32) * 5 / 9;
        System.out.println(fahrenheit + " degrees fahrenheit is equal to " + celcius + " degrees Celcius");
    }
}
```

7. Write a program to check if a given number is divisible by both 5 and 7.

```
1 package in.globalquesttechnologies;
2 import java.util.Scanner;
3 public class DivisibleBy5And7 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         System.out.print("Enter a number: ");
7         int num = sc.nextInt();
8         if (num % 5 == 0 && num % 7 == 0) {
9             System.out.println(num + " is divisible by both 5 and 7");
10        } else {
11            System.out.println(num + " is not divisible by both 5 and 7");
12        }
13        sc.close();
14    }
15 }
16 }
```

```
<terminated> DivisibleBy5And7 [Java Application] C:\Users\abhay\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Enter a number: 35
35 is divisible by both 5 and 7
```

```

package in.globalquesttechnologies;
import java.util.Scanner;
public class DivisibleBy5And7 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num = sc.nextInt();
        if (num % 5 == 0 && num % 7 == 0) {
            System.out.println(num + " is divisible by both 5 and 7");
        } else {
            System.out.println(num + " is not divisible by both 5 and 7");
        }
        sc.close();
    }
}

```

8. Write a program to calculate the compound interest.

The screenshot shows the Eclipse IDE interface with two tabs open: 'CompoundInterest.java' and 'Console'. The code in 'CompoundInterest.java' is as follows:

```

1 package operatorsprograms;
2 import java.util.Scanner;
3 public class CompoundInterest {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter principal amount: ");
8         double principal = sc.nextDouble();
9         System.out.print("Enter rate of interest: ");
10        double rate = sc.nextDouble();
11        System.out.print("Enter time in years: ");
12        double time = sc.nextDouble();
13
14        double compoundinterest = principal * Math.pow((1 + rate / 100), time);
15        System.out.println("Compound Interest: " + compoundinterest);
16    }
17
18
19 }

```

In the 'Console' tab, the output of the program is shown:

```

<terminated> CompoundInterest [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\plugins\org.eclipse.jdt.ls.core\bin\operatorsprograms\CompoundInterest
Enter principal amount: 2000
Enter rate of interest: 2
Enter time in years: 2
Compound Interest: 80.80000000000018

```

9. Write a program to check whether a given character is a vowel or consonant.

The screenshot shows the Eclipse IDE interface with several tabs at the top: FahrenheitTo..., numberDivisi..., CompoundInte..., vowelOrConso..., and another unnamed tab. The main editor area contains Java code for determining if a character is a vowel or consonant. The code uses a Scanner to read input from System.in and prints output to System.out. The console window below shows the application's output after running it.

```
1 package operatorsprograms;
2 import java.util.Scanner;
3 public class vowelOrConsonent {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter a character: ");
8         char ch = sc.next().charAt(0);
9
10        if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u'
11            || ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U') {
12            System.out.println("The character is a vowel.");
13        } else {
14            System.out.println("The character is a consonant.");
15        }
16    }
17
18 }
19
20 }
```

```
Console <terminated> vowelOrConsonent [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32
Enter a character: 1
The character is a consonant.
```

10. Write a program to find the maximum of three numbers using conditional operator.

The screenshot shows the Eclipse IDE interface with several tabs at the top: numberDivisi..., maximumOfTh..., CompoundInte..., vowelOrConso..., and another unnamed tab. The main editor area contains Java code for finding the maximum of three numbers. The code uses a Scanner to read three integers from the user and then determines the maximum using a ternary operator. The console window below shows the application's output after running it.

```
1 package operatorsprograms;
2 import java.util.Scanner;
3 public class maximumOfThreeNumbers {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter first number: ");
8         int num1 = sc.nextInt();
9         System.out.print("Enter second number: ");
10        int num2 = sc.nextInt();
11        System.out.print("Enter third number: ");
12        int num3 = sc.nextInt();
13
14        int max = (num1 > num2) ? (num1 > num3 ? num1 : num3) : (num2 > num3 ? num2 : num3);
15        System.out.println("The maximum number is: " + max);
16    }
17
18 }
19
20 }
```

```
Console <terminated> maximumOfThreeNumbers [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024
Enter first number: 30
Enter second number: 34
Enter third number:
5000
The maximum number is: 5000
```

11. Write a program to find the sum of digits of a number using while loop.

The screenshot shows the Eclipse IDE interface. The top part displays the Java code for calculating the sum of digits of a number using a while loop. The bottom part shows the output of the program in the 'Console' tab, where it prompts for a number, receives input '234566788', and prints the result 'Sum of digits: 49'.

```
1 package operatorsprograms;
2 import java.util.Scanner;
3 public class sumOfDigitsUsingWhile {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter a number: ");
8         int num = sc.nextInt();
9
10        int sum = 0;
11        while (num != 0) {
12            sum += num % 10;
13            num /= 10;
14        }
15        System.out.println("Sum of digits: " + sum);
16    }
17
18 }
19
```

```
<terminated> sumOfDigitsUsingWhile [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-wi
Enter a number: 234566788
Sum of digits: 49
```

12. Write a program to check whether a given number is palindrome or not using recursion.

The screenshot shows the Eclipse IDE interface. The top part displays the Java code for checking if a number is a palindrome using a recursive helper method. The bottom part shows the output of the program in the 'Console' tab, where it prompts for a number, receives input '456', and prints the result 'The number is not a palindrome.'

```
1 package practiseprograms;
2 import java.util.Scanner;
3 public class palindromOfNumber {
4
5     static int reverse(int num, int rev) {
6         if (num == 0) {
7             return rev;
8         }
9         rev = rev * 10 + num % 10;
10        return reverse(num / 10, rev);
11    }
12
13    public static void main(String[] args) {
14        Scanner sc = new Scanner(System.in);
15        System.out.print("Enter a number: ");
16        int num = sc.nextInt();
17        int reversedNum = reverse(num, 0);
18
19        if (num == reversedNum) {
20            System.out.println("The number is a palindrome.");
21        } else {
22            System.out.println("The number is not a palindrome.");
23        }
24    }
25
26 }
```

```
<terminated> palindromOfNumber [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-
Enter a number: 456
The number is not a palindrome.
```

13. Write a program to check whether a given number is prime or not using for loop.

```
primenumber... × maximumOfThree... × sumOfDigits... × palindromOf... × 95
1 package practiseprograms;
2
3 import java.util.Scanner;
4
5 public class primenumber {
6
7     public static void main(String[] args) {
8         Scanner sc=new Scanner(System.in);
9         System.out.println("enter the number");
10        int number=sc.nextInt();
11        for(int i=0;i<=number;i++) {
12            if(number%2!=0&&number%3!=0) {
13                System.out.println("is a prime number");
14            }
15            else {
16                System.out.println(" not a prime");
17            }
18        }
19    }
20
21
22
23
24
25
```

```
Console ×
<terminated> primenumber [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32
enter the number
7
is a prime number
```

14. Write a program to find the factorial of a number using recursion.

```
primenumber... × sumOfDigits... × palindromOf... × fibUsingRecursion... × 95
1 package practiseprograms;
2
3 import java.util.Scanner;
4
5 public class fibUsingRecursion {
6     static int fibonacci(int n) {
7         if (n <= 1) {
8             return n;
9         }
10         return fibonacci(n - 1) + fibonacci(n - 2);
11     }
12
13     public static void main(String[] args) {
14         Scanner sc = new Scanner(System.in);
15         System.out.print("Enter number of terms: ");
16         int n = sc.nextInt();
17
18         System.out.println("Fibonacci series:");
19         for (int i = 0; i < n; i++) {
20             System.out.print(fibonacci(i) + " ");
21         }
22
23
24
```

```
Console ×
<terminated> fibUsingRecursion [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R
Enter number of terms: 20
Fibonacci series:
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181
```

15. Write a program to calculate the power of a number using recursion.

```
1 package practiseprograms;
2 import java.util.Scanner;
3 public class power {
4
5     static int power(int base, int exponent) {
6         if (exponent == 0) {
7             return 1;
8         }
9         return base * power(base, exponent - 1);
10    }
11
12    public static void main(String[] args) {
13        Scanner sc = new Scanner(System.in);
14        System.out.print("Enter base: ");
15        int base = sc.nextInt();
16        System.out.print("Enter exponent: ");
17        int exponent = sc.nextInt();
18
19        System.out.println("Result: " + power(base, exponent));
20    }
21
22 }
```

```
<terminated> power [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_6
Enter base: 5
Enter exponent: 26
Result: 839070905
```

16. Write a program to print the Fibonacci series using recursion

```
1 package practiseprograms;
2 import java.util.Scanner;
3 public class fibUsingRecursion {
4     static int fibonacci(int n) {
5         if (n <= 1) {
6             return n;
7         }
8         return fibonacci(n - 1) + fibonacci(n - 2);
9     }
10
11    public static void main(String[] args) {
12        Scanner sc = new Scanner(System.in);
13        System.out.print("Enter number of terms: ");
14        int n = sc.nextInt();
15
16        System.out.println("Fibonacci series:");
17        for (int i = 0; i < n; i++) {
18            System.out.print(fibonacci(i) + " ");
19        }
20    }
21
22 }
```



```
<terminated> fibUsingRecursion [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32
Enter number of terms: 45
Fibonacci series:
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765 10946 17
```

17. Write a program to reverse a string using recursion.

The screenshot shows the Eclipse IDE interface. The top tab bar has several tabs: fibUsingRec..., poerofnumbe..., powerjava, ReversiveOfS..., and "9s". The main editor area contains Java code for reversing a string:

```
1 package practiseprograms;
2 import java.util.Scanner;
3 public class ReversiveOfString {
4     static String reverse(String str) {
5         if (str.isEmpty()) {
6             return str;
7         }
8         return reverse(str.substring(1)) + str.charAt(0);
9     }
10    public static void main(String[] args) {
11        Scanner sc = new Scanner(System.in);
12        System.out.print("Enter a string: ");
13        String str = sc.nextLine();
14
15        System.out.println("Reversed string: " + reverse(str));
16    }
17}
18
19
20
21
22
```

The bottom part of the interface shows the 'Console' tab with the output of the program:

```
<terminated> ReversibleOfString [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win3
Enter a string: ammu
Reversed string: umma
```

18. Write a program to calculate the sum of natural numbers up to a given term

The screenshot shows the Eclipse IDE interface. The top tab bar has several tabs: fibUsingRec..., powerjava, ReversiveOfS..., sumOfNatural..., and "9s". The main editor area contains Java code for calculating the sum of natural numbers:

```
1 package practiseprograms;
2 import java.util.Scanner;
3 public class sumOfNatural {
4
5     static int sum(int n) {
6         if (n == 1) {
7             return 1;
8         }
9         return n + sum(n - 1);
10    }
11
12    public static void main(String[] args) {
13        Scanner sc = new Scanner(System.in);
14        System.out.print("Enter a number: ");
15        int n = sc.nextInt();
16
17        System.out.println("Sum of first " + n + " natural numbers:");
18    }
19}
20
```

The bottom part of the interface shows the 'Console' tab with the output of the program:

```
<terminated> sumOfNatural [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x8
Enter a number: 50
Sum of first 50 natural numbers: 1275
```

19. Write a program to check whether a given year is leap year or not using conditional operator

The screenshot shows the Eclipse IDE interface with several open tabs at the top: fibUsingRec..., ReversiveOfS..., sumOfNatural..., Leap19.java, and another unnamed tab. The Leap19.java tab is active, displaying the following Java code:

```
1 package practiseprograms;
2 import java.util.Scanner;
3 public class Leap19 {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter a year: ");
8         int year = sc.nextInt();
9
10        String result = ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0));
11        System.out.println(result);
12    }
13 }
14
15 }
16 }
```

Below the editor, the Eclipse Console window is visible, showing the output of the program:

```
<terminated> Leap19 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\
Enter a year: 2003
Not a Leap Year
```

20. Write a program to find the LCM (Least Common Multiple) of two numbers.

The screenshot shows the Eclipse IDE interface with several open tabs at the top: ReversibleOfS..., sumOfNatural..., Leap19.java, and LCM20.java. The LCM20.java tab is active, displaying the following Java code:

```
1 package practiseprograms;
2 import java.util.Scanner;
3 public class LCM20 {
4
5     static int lcm(int a, int b) {
6         int max = (a > b) ? a : b;
7         while (true) {
8             if (max % a == 0 && max % b == 0) {
9                 return max;
10            }
11            max++;
12        }
13    }
14
15    public static void main(String[] args) {
16        Scanner sc = new Scanner(System.in);
17        System.out.print("Enter first number: ");
18        int a = sc.nextInt();
19        System.out.print("Enter second number: ");
20        int b = sc.nextInt();
21
22        System.out.println("LCM of " + a + " and " + b + " is: " +
23    }
24
25 }
26
27 }
```

Below the editor, the Eclipse Console window is visible, showing the output of the program:

```
<terminated> LCM20 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\
Enter first number: 467
Enter second number: 432
LCM of 467 and 432 is: 201744
```

21. Write a program to calculate the area of a triangle using Heron's formula

```
sumOfNatural...  Leap19.java  LCM20.java  HeronsJava ×  "qs
1 package practiseprograms;
2 import java.util.Scanner;
3 public class Herons {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter first side: ");
8         double a = sc.nextDouble();
9         System.out.print("Enter second side: ");
10        double b = sc.nextDouble();
11        System.out.print("Enter third side: ");
12        double c = sc.nextDouble();
13
14        double s = (a + b + c) / 2;
15        double area = Math.sqrt(s * (s - a) * (s - b) * (s - c));
16
17        System.out.println("Area of the triangle: " + area);
18    }
19
20}
21
22
23
```

```
Console ×
<terminated> Herons [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter first side: 36
Enter second side: 42
Enter third side: 12
Area of the triangle: 200.2373591515829
```

22. Write a program to find the sum of all even numbers between 1 to 100 using for loop.

```
Leap19.java  LCM20.java  HeronsJava  sum1To100.java ×  "qs
1 package practiseprograms;
2
3 public class sum1To100 {
4
5     public static void main(String[] args) {
6         int sum = 0;
7         for (int i = 1; i <= 100; i++) {
8             if (i % 2 == 0) {
9                 sum += i;
10            }
11        }
12        System.out.println("Sum of all even numbers between 1 to 100: " + sum);
13    }
14
15}
16
17
```

```
Console ×
<terminated> sum1To100 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Sum of all even numbers between 1 to 100: 2550
```

23. Write a program to calculate the simple interest.

The screenshot shows the Eclipse IDE interface. The top tab bar has several tabs: Leap19.java, LCM20.java, Heronsjava, sum1To100.java, Sijava, and "24". The main editor window contains the following Java code:

```
1 package practiseprograms;
2 import java.util.Scanner;
3 public class Si {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter principal amount: ");
8         double principal = sc.nextDouble();
9         System.out.print("Enter rate of interest: ");
10        double rate = sc.nextDouble();
11        System.out.print("Enter time in years: ");
12        double time = sc.nextDouble();
13
14        double interest = (principal * rate * time) / 100;
15        System.out.println("Simple Interest: " + interest);
16    }
17 }
18
19 }
20
```

Below the editor is the Console tab, which displays the program's output:

```
<terminated> Si [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\eclipse
Enter principal amount: 3000
Enter rate of interest: 30
Enter time in years: 3
Simple Interest: 2700.0
```

24. Write a program to print the multiplication table of a given number using for loop.

The screenshot shows the Eclipse IDE interface. The top tab bar has several tabs: Heronsjava, sum1To100.java, Sijava, and tables24.java, with "24" selected. The main editor window contains the following Java code:

```
1 package practiseprograms;
2 import java.util.Scanner;
3 public class tables24 {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter a number: ");
8         int num = sc.nextInt();
9
10        System.out.println("Multiplication table of " + num + ":");
11        for (int i = 1; i <= 10; i++) {
12            System.out.println(num + " x " + i + " = " + (num * i));
13        }
14    }
15 }
16
17
```

Below the editor is the Console tab, which displays the program's output:

```
<terminated> tables24 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\eclipse
Enter a number: 342
Multiplication table of 342:
342 x 1 = 342
342 x 2 = 684
342 x 3 = 1026
342 x 4 = 1368
342 x 5 = 1710
342 x 6 = 2052
342 x 7 = 2394
342 x 8 = 2736
342 x 9 = 3078
342 x 10 = 3420
```

25. Write a program to check whether a given number is Armstrong or not using while loop.

```

1 sum1To100.java  2 Sjava  3 tables24.java  4 armstrong.java > 45
2 import java.util.Scanner;
3 public class armstrong {
4
5     public static void main(String[] args) {
6
7         Scanner scanner = new Scanner(System.in);
8
9         System.out.print("Enter a number: ");
10        int num = scanner.nextInt();
11        int originalNum = num;
12        int sum = 0;
13
14        while (num != 0) {
15            int digit = num % 10;
16            sum += digit * digit * digit;
17            num /= 10;
18        }
19
20        System.out.println(originalNum + " is an Armstrong number");
21    }
22
23
24
25

```

Console

```

<terminated> armstrong [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter a number: 401
401 is an Armstrong number: false

```

Arrays

1. Write a program to find the sum of all elements in an array.

```

1 printnamejava  2 sumofevennum...  3 swapingjava  4 primenumber...  5 factorialjava  6 qua
7 package practiseprograms;
8 import java.util.Scanner;
9 public class areaoftriangle {
10
11     public static void main(String[] args) {
12         Scanner sc=new Scanner(System.in);
13         System.out.println("enter the value");
14         double base = sc.nextDouble();
15         System.out.println("enter the value");
16         double height = sc.nextDouble();
17         double area = 0.5 * base * height;
18         System.out.println("Area of the triangle: " + area);
19     }
20
21
22
23
24
25

```

Console

```

<terminated> areaoftriangle [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\eclipse
enter the value
20
enter the value
30
Area of the triangle: 300.0

```

2. Write a program to find the largest and smallest elements in an array.

```

1 package arrayPrograms;
2 import java.util.Scanner;
3
4 public class array2 {
5
6     public static void main(String[] args) {
7         Scanner sc = new Scanner(System.in);
8         System.out.print("Enter number of elements: ");
9         int n = sc.nextInt();
10        int[] arr = new int[n];
11
12        System.out.print("Enter elements: ");
13        for (int i = 0; i < n; i++) {
14            arr[i] = sc.nextInt();
15        }
16
17        int largest = arr[0], smallest = arr[0];
18
19        for (int i = 1; i < n; i++) {
20            if (arr[i] > largest) largest = arr[i];
21            if (arr[i] < smallest) smallest = arr[i];
22        }
23    }
}

```

3. Write a program to copy elements from one array to another.

```

1 Sijava
2
3 import java.util.Scanner;
4 public class array3 {
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter number of elements: ");
8         int n = sc.nextInt();
9         int[] arr1 = new int[n], arr2 = new int[n];
10
11         System.out.println("Enter elements: ");
12         for (int i = 0; i < n; i++) {
13             arr1[i] = sc.nextInt();
14         }
15
16         for (int i = 0; i < n; i++) {
17             arr2[i] = arr1[i];
18         }
19
20         System.out.println("Array 2 after copy: ");
21         for (int i = 0; i < n; i++) {
22             System.out.print(arr2[i] + " ");
23         }
24     }
}

```

4. Write a program to remove duplicate elements from an array.

The screenshot shows the Eclipse IDE interface. The top part displays the code for 'array4.java' in the editor. The code reads elements from the user, stores them in an array, and then prints the array after removing duplicates using a HashSet. The bottom part shows the 'Console' tab with the output of the application. The user enters 5 elements: 2, 45, 36, 78, 46. The output shows the array [2, 36, 45, 78, 46] with duplicates removed.

```
1 package arrayPrograms;
2 import java.util.*;
3 public class array4 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         System.out.print("Enter number of elements: ");
7         int n = sc.nextInt();
8         int[] arr = new int[n];
9
10        System.out.println("Enter elements: ");
11        for (int i = 0; i < n; i++) {
12            arr[i] = sc.nextInt();
13        }
14
15        Set<Integer> set = new HashSet<>();
16        for (int i = 0; i < n; i++) {
17            set.add(arr[i]);
18        }
19
20        System.out.println("Array after removing duplicates: " + se
21    }
22 }

```

```
Console ×
<terminated> array4 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Enter number of elements: 5
Enter elements:
2
45
36
78
46
Array after removing duplicates: [2, 36, 45, 78, 46]
```

5. Write a program to reverse an array

The screenshot shows the Eclipse IDE interface. The top part displays the code for 'array5.java' in the editor. The code reads elements from the user, stores them in an array, and then prints the array in reverse order. The bottom part shows the 'Console' tab with the output of the application. The user enters 8 elements: 2, 4, 3, 6, 2, 7, 2, 8. The output shows the reversed array [8, 2, 7, 2, 6, 3, 4, 2].

```
1 package arrayPrograms;
2 import java.util.Scanner;
3 public class array5 {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter number of elements: ");
8         int n = sc.nextInt();
9         int[] arr = new int[n];
10
11        System.out.println("Enter elements: ");
12        for (int i = 0; i < n; i++) {
13            arr[i] = sc.nextInt();
14        }
15
16        System.out.println("Reversed array: ");
17        for (int i = n - 1; i >= 0; i--) {
18            System.out.print(arr[i] + " ");
19        }
20    }
21 }

```

```
Console ×
<terminated> array5 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Enter number of elements: 8
Enter elements:
2
4
3
6
2
7
2
8
Reversed array:
8 2 7 2 6 3 4 2
```

6. Write a program to sort an array in ascending and descending order.

```

array4.java array5.java array12.java array25.java array6.java array7.java × 94
1 package arrayPrograms;
2 import java.util.Arrays;
3 public class array6 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         System.out.print("Enter number of elements: ");
7         int n = sc.nextInt();
8         int[] arr = new int[n];
9
10        System.out.println("Enter elements: ");
11        for (int i = 0; i < n; i++) {
12            arr[i] = sc.nextInt();
13        }
14
15        Arrays.sort(arr);
16        System.out.println("Array in ascending order: " + Arrays.to
17
18        // Descending order
19        for (int i = 0; i < n / 2; i++) {
20            int temp = arr[i];
21            arr[i] = arr[n - 1 - i];
22            arr[n - 1 - i] = temp;
23        }
24        System.out.println("Array in descending order: " + Arrays.t
25    }
26}
27
28
29

```

Console ×

```

<terminated> array6 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\cli
Enter number of elements: 5
Enter elements:
3
4
5
3
4
Array in ascending order: [3, 3, 4, 4, 5]
Array in descending order: [5, 4, 4, 3, 3]

```

7. Write a program to find the frequency of each element in an array

```

array4.java array5.java array25.java array6.java array7.java × 94
1 package arrayPrograms;
2 import java.util.*;
3 public class array7 {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter number of elements: ");
8         int n = sc.nextInt();
9         int[] arr = new int[n];
10
11        System.out.println("Enter elements: ");
12        for (int i = 0; i < n; i++) {
13            arr[i] = sc.nextInt();
14        }
15
16        Map<Integer, Integer> frequencyMap = new HashMap<>();
17        for (int i = 0; i < n; i++) {
18            frequencyMap.put(arr[i], frequencyMap.getOrDefault(arr[
19        }
20
21        System.out.println("Frequency of elements: " + frequencyMap
22    }
23}
24
25
26

```

Console ×

```

<terminated> array7 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\cli
Enter number of elements: 6
Enter elements:
4
5
6
23
3
4
Frequency of elements: {3=1, 4=2, 5=1, 6=1, 23=1}

```

8. Write a program to merge two sorted arrays.

```

sum1To100.java  Sljava  table24.java  armstrong.java  arry3.java  array.java  Console X
1 package arrayPrograms;
2 import java.util.Scanner;
3 public class array8 {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter number of elements in array 1: ");
8         int n1 = sc.nextInt();
9         int[] arr1 = new int[n1];
10
11         System.out.println("Enter elements for array 1: ");
12         for (int i = 0; i < n1; i++) {
13             arr1[i] = sc.nextInt();
14         }
15
16         System.out.print("Enter number of elements in array 2: ");
17         int n2 = sc.nextInt();
18         int[] arr2 = new int[n2];
19
20         System.out.println("Enter elements for array 2: ");
21         for (int i = 0; i < n2; i++) {
22             arr2[i] = sc.nextInt();
23         }
24
25         int[] merged = new int[n1 + n2];
26         System.arraycopy(arr1, 0, merged, 0, n1);
27         System.arraycopy(arr2, 0, merged, n1, n2);
28
29         System.out.println("Merged array: ");
30         for (int i : merged) {
31             System.out.print(i + " ");
32         }
33     }
34 }
35
36

```

9. Write a program to find the intersection of two arrays.

```

array4.java  array5.java  array10.java  array11.java  array12.java  array25.java  Console X
1 import java.util.*;
2 public class array9 {
3
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter number of elements in array 1: ");
8         int n1 = sc.nextInt();
9         int[] arr1 = new int[n1];
10
11         System.out.println("Enter elements for array 1: ");
12         for (int i = 0; i < n1; i++) {
13             arr1[i] = sc.nextInt();
14         }
15
16         System.out.print("Enter number of elements in array 2: ");
17         int n2 = sc.nextInt();
18         int[] arr2 = new int[n2];
19
20         System.out.println("Enter elements for array 2: ");
21         for (int i = 0; i < n2; i++) {
22             arr2[i] = sc.nextInt();
23         }
24
25         Set<Integer> set1 = new HashSet<>();
26         Set<Integer> intersection = new HashSet<>();
27
28         for (int i = 0; i < n1; i++) {
29             set1.add(arr1[i]);
30         }
31
32         for (int i = 0; i < n2; i++) {
33             if (set1.contains(arr2[i])) {
34                 intersection.add(arr2[i]);
35             }
36         }
37
38         System.out.println("Intersection of arrays: " + intersection);
39     }
40 }
41
42
43

```

10. Write a program to check whether an array is palindrome or not.

The screenshot shows the Eclipse IDE interface with multiple tabs at the top. The active tab is 'array10.java'. The code in the editor is as follows:

```
1 package arrayPrograms;
2 import java.util.Scanner;
3 public class array10 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         System.out.print("Enter number of elements: ");
7         int n = sc.nextInt();
8         int[] arr = new int[n];
9
10        System.out.println("Enter elements: ");
11        for (int i = 0; i < n; i++) {
12            arr[i] = sc.nextInt();
13        }
14
15        boolean isPalindrome = true;
16        for (int i = 0; i < n / 2; i++) {
17            if (arr[i] != arr[n - 1 - i]) {
18                isPalindrome = false;
19                break;
20            }
21        }
22
23        if (isPalindrome) {
24            System.out.println("Array is a palindrome.");
25        } else {
26            System.out.println("Array is not a palindrome.");
27        }
28    }
29
30
31
32}
```

The right side of the interface shows the 'Console' tab with the output of the program. It reads 'Enter number of elements: 4', then lists the numbers 34, 34, 56, and 67. Finally, it prints 'Array is not a palindrome.'

11. Write a program to find the sum of all positive numbers in an array.

The screenshot shows the Eclipse IDE interface with multiple tabs at the top. The active tab is 'array11.java'. The code in the editor is as follows:

```
1 package arrayPrograms;
2 import java.util.Scanner;
3 public class array11 {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter number of elements: ");
8         int n = sc.nextInt();
9         int[] arr = new int[n];
10        int sum = 0;
11
12        System.out.println("Enter elements: ");
13        for (int i = 0; i < n; i++) {
14            arr[i] = sc.nextInt();
15            if (arr[i] > 0) {
16                sum += arr[i];
17            }
18        }
19
20        System.out.println("Sum of positive numbers: " + sum);
21    }
22
23
24
```

The right side of the interface shows the 'Console' tab with the output of the program. It reads 'Enter number of elements: 7', then lists the numbers 2, 34, 15, 36, 16, 47, and 32. Finally, it prints 'Sum of positive numbers: 182'.

12. Write a program to find the sum of all negative numbers in an array.

```
array10java array11java array12.java × array9.java »95
4 public class array12 {
5
6     public static void main(String[] args) {
7         Scanner sc = new Scanner(System.in);
8         System.out.print("Enter number of elements: ");
9         int n = sc.nextInt();
10        int[] arr = new int[n];
11        int sum = 0;
12
13        System.out.println("Enter elements: ");
14        for (int i = 0; i < n; i++) {
15            arr[i] = sc.nextInt();
16            if (arr[i] < 0) {
17                sum += arr[i];
18            }
19        }
20
21        System.out.println("Sum of negative numbers: " + sum);
22    }
23
24
25
26
```

Console ×

```
<terminated> array12 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter number of elements: 4
Enter elements:
-34
26
50
-21
Sum of negative numbers: -55
```

13. Write a program to find the product of all elements in an array.

```
array10java array11java array12.java × array13.java »95
1 package arrayPrograms;
2 import java.util.Scanner;
3 public class array13 {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter number of elements: ");
8         int n = sc.nextInt();
9         int[] arr = new int[n];
10        int product = 1;
11
12        System.out.println("Enter elements: ");
13        for (int i = 0; i < n; i++) {
14            arr[i] = sc.nextInt();
15            product *= arr[i];
16        }
17
18        System.out.println("Product of elements: " + product);
19    }
20
21
22
23
```

Console ×

```
<terminated> array13 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter number of elements: 5
Enter elements:
34
24
123
33
23
Product of elements: 76179312
```

14. Write a program to find the second largest and second smallest elements in an array

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows files array11.java, array12.java, array13.java, array14.java, and array15.java.
- Code Editor:** Displays the content of array14.java. The code defines a class array14 with a main method. It uses Scanner to read input from System.in, sorts the array, and then prints the second smallest and largest elements.
- Console:** Shows the output of running the program. It asks for the number of elements (5), then lists the entered elements (3, 5, 6, 2, 4). Finally, it outputs "Second smallest element: 4" and "Second largest element: 6".

```
array14.java
package arrayPrograms;
import java.util.Arrays;
public class array14 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of elements: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter elements: ");
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }
        Arrays.sort(arr);
        System.out.println("Second smallest element: " + arr[1]);
        System.out.println("Second largest element: " + arr[n - 2]);
    }
}
```

```
terminated> array14 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\arrayPrograms
Enter number of elements: 5
Enter elements:
3
5
6
2
4
Second smallest element: 4
Second largest element: 6
```

15. Write a program to find the index of a given element in an array.

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows files array12.java, array13.java, array14.java, and array15.java.
- Code Editor:** Displays the content of array15.java. The code defines a class array15 with a main method. It reads elements into an array and then searches for a target element, printing its index if found.
- Console:** Shows the output of running the program. It asks for the number of elements (3), then lists the entered elements (23, 14, 25). When asked to enter the element to search for (34), it outputs "Element not found."

```
array15.java
package arrayPrograms;
import java.util.Scanner;
public class array15 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of elements: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter elements: ");
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }
        System.out.print("Enter element to search: ");
        int target = sc.nextInt();
        int index = -1;
        for (int i = 0; i < n; i++) {
            if (arr[i] == target) {
                index = i;
                break;
            }
        }
    }
}
```

```
terminated> array15 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\arrayPrograms
Enter number of elements: 3
Enter elements:
23
14
25
Enter element to search: 34
Element not found.
```

16. Write a program to rotate an array to the left or right.

```

array13.java array14.java array15.java array16.java array17.java
1 package arrayPrograms;
2 import java.util.Scanner;
3 public class array16 {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter number of elements: ");
8         int n = sc.nextInt();
9         int[] arr = new int[n];
10
11        System.out.println("Enter elements: ");
12        for (int i = 0; i < n; i++) {
13            arr[i] = sc.nextInt();
14        }
15
16        System.out.print("Enter the number of positions to rotate: ");
17        int k = sc.nextInt();
18
19        // Left rotation
20        int[] rotatedLeft = new int[n];
21        for (int i = 0; i < n; i++) {
22            rotatedLeft[i] = arr[(i + k) % n];
23        }
}

```

Console X

```

<terminated> array16 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\
Enter number of elements: 3
Enter elements:
23
45
67
Enter the number of positions to rotate: 2
Array after left rotation:
67 23 45
Array after right rotation:
45 67 23

```

17. Write a program to print the elements of a 2D array in spiral order.

```

array4.java array5.java array10.java array11.java array12.java array6.java
1 public class array17 {
2
3     public static void main(String[] args) {
4         Scanner sc = new Scanner(System.in);
5         System.out.print("Enter number of rows: ");
6         int m = sc.nextInt();
7         System.out.print("Enter number of columns: ");
8         int n = sc.nextInt();
9         int[][] matrix = new int[m][n];
10
11        System.out.println("Enter matrix elements: ");
12        for (int i = 0; i < m; i++) {
13            for (int j = 0; j < n; j++) {
14                matrix[i][j] = sc.nextInt();
15            }
16        }
17
18        int top = 0, bottom = m - 1, left = 0, right = n - 1;
19
20        while (top <= bottom && left <= right) {
21            for (int i = left; i <= right; i++) {
22                System.out.print(matrix[top][i] + " ");
23            }
24            top++;
25
26            for (int i = top; i <= bottom; i++) {
27                System.out.print(matrix[i][right] + " ");
28            }
29            right--;
30
31            if (top <= bottom) {
32                for (int i = right; i >= left; i--) {
33                    System.out.print(matrix[bottom][i] + " ");
34                }
35                bottom--;
36            }
37
38            if (left <= right) {
39                for (int i = bottom; i >= top; i--) {
40                    System.out.print(matrix[i][left] + " ");
41                }
42                left++;
43            }
44        }
}

```

Console X

```

<terminated> array17 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\
Enter number of rows: 3
Enter number of columns: 3
Enter matrix elements:
23
12
14
25
3
6
5
27
26
23 12 14 6 56 67 5 25 3

```

18. Write a program to check whether two arrays are equal or not

```

1 package arrayPrograms;
2 import java.util.Scanner;
3 public class array18 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         System.out.print("Enter number of elements in array 1: ");
7         int n1 = sc.nextInt();
8         int[] arr1 = new int[n1];
9
10        System.out.print("Enter elements for array 1: ");
11        for (int i = 0; i < n1; i++) {
12            arr1[i] = sc.nextInt();
13        }
14
15        System.out.print("Enter number of elements in array 2: ");
16        int n2 = sc.nextInt();
17        int[] arr2 = new int[n2];
18
19        System.out.print("Enter elements for array 2: ");
20        for (int i = 0; i < n2; i++) {
21            arr2[i] = sc.nextInt();
22        }
23
24        boolean isEqual = (n1 == n2);
25
26        if (isEqual) {
27            for (int i = 0; i < n1; i++) {
28                if (arr1[i] != arr2[i]) {
29                    isEqual = false;
30                    break;
31                }
32            }
33        }
34
35        if (isEqual) {
36            System.out.println("Arrays are equal.");
37        } else {
38            System.out.println("Arrays are not equal.");
39        }
40    }
41 }
42

```

The console output shows the program running and comparing two arrays. It asks for the number of elements and then the elements themselves for both arrays. Finally, it prints whether the arrays are equal or not.

19. Write a program to find the sum of elements in the upper triangle of a

```

1 package arrayPrograms;
2 import java.util.Scanner;
3
4 public class array19 {
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter number of rows and columns: ");
8         int n = sc.nextInt();
9         int[][] matrix = new int[n][n];
10
11        System.out.println("Enter matrix elements: ");
12        for (int i = 0; i < n; i++) {
13            for (int j = 0; j < n; j++) {
14                matrix[i][j] = sc.nextInt();
15            }
16        }
17
18        int sum = 0;
19        for (int i = 0; i < n; i++) {
20            for (int j = i; j < n; j++) {
21                sum += matrix[i][j];
22            }
23        }
24
25        System.out.println("Sum of upper triangle elements: " + sum);
26    }
27 }
28
29
30

```

The console output shows the program running and calculating the sum of elements in the upper triangle of a 4x4 matrix. It asks for the size of the matrix and then its elements, finally printing the sum of the upper triangle elements.

20. Write a program to find the sum of elements in the lower triangle of a matrix

```
array17.java array18.java array19.java array20.java × "95"
1 package arrayPrograms;
2 import java.util.Scanner;
3 public class array20 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         System.out.print("Enter number of rows and columns: ");
7         int n = sc.nextInt();
8         int[][] matrix = new int[n][n];
9
10        System.out.println("Enter matrix elements: ");
11        for (int i = 0; i < n; i++) {
12            for (int j = 0; j < n; j++) {
13                matrix[i][j] = sc.nextInt();
14            }
15        }
16
17        int sum = 0;
18        for (int i = 0; i < n; i++) {
19            for (int j = 0; j <= i; j++) {
20                sum += matrix[i][j];
21            }
22        }
23
24        System.out.println("Sum of lower triangle elements: " + sum)
25    }
}
```

```
Console ×
<terminated> array20 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter number of rows and columns: 3
Enter matrix elements:
41
52
63
74
85
96
75
75
Sum of lower triangle elements: 445
```

21. Write a program to find the sum of elements in each row and column of a matrix.

```
array18.java array19.java array20.java array21.java × "95"
16
17
18
19        System.out.println("Sum of elements in each row:");
20        for (int i = 0; i < m; i++) {
21            int rowSum = 0;
22            for (int j = 0; j < n; j++) {
23                rowSum += matrix[i][j];
24            }
25            System.out.println("Row " + (i + 1) + ": " + rowSum);
26
27
28        System.out.println("Sum of elements in each column:");
29        for (int j = 0; j < n; j++) {
30            int colSum = 0;
31            for (int i = 0; i < m; i++) {
32                colSum += matrix[i][j];
33            }
34            System.out.println("Column " + (j + 1) + ": " + colSum);
}
}

Console ×
<terminated> array21 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter number of rows: 3
Enter number of columns: 2
Enter matrix elements:
74
85
65
5
5
585
Sum of elements in each row:
Row 1: 159
Row 2: 70
Row 3: 590
Sum of elements in each column:
Column 1: 144
Column 2: 675
```

22. Write a program to check whether a given matrix is symmetric or not.

```
array10java array11java array12java array13java array14java a Console <terminated>
1 package arrayPrograms;
2 import java.util.Scanner;
3 public class array22 {
4
5
6    public static void main(String[] args) {
7        Scanner sc = new Scanner(System.in);
8        System.out.print("Enter number of rows and columns: ");
9        int n = sc.nextInt();
10       int[][] matrix = new int[n][n];
11
12       System.out.println("Enter matrix elements: ");
13       for (int i = 0; i < n; i++) {
14           for (int j = 0; j < n; j++) {
15               matrix[i][j] = sc.nextInt();
16           }
17       }
18
19       boolean isSymmetric = true;
20       for (int i = 0; i < n; i++) {
21           for (int j = 0; j < n; j++) {
22               if (matrix[i][j] != matrix[j][i]) {
23                   isSymmetric = false;
24                   break;
25               }
26           }
27           if (!isSymmetric) break;
28       }
29
30       if (isSymmetric) {
31           System.out.println("Matrix is symmetric.");
32       } else {
33           System.out.println("Matrix is not symmetric.");
34       }
35   }
36
37
38 }
```

23. Write a program to find the saddle point of a matrix.

24. Write a program to find the kth smallest and kth largest elements in an array

```

array24.java  array15.java  array16.java  array17.java  array18.java  array19.java  array20.java  Console X
1 package arrayPrograms;
2 import java.util.Arrays;
3 public class array24 {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Enter number of elements: ");
8         int n = sc.nextInt();
9         int[] arr = new int[n];
10
11         System.out.print("Enter elements: ");
12         for (int i = 0; i < n; i++) {
13             arr[i] = sc.nextInt();
14         }
15
16         System.out.print("Enter value of k: ");
17         int k = sc.nextInt();
18
19         Arrays.sort(arr);
20
21         if (k <= n) {
22             System.out.println(k + "th smallest element: " + arr[k - 1]);
23             System.out.println(k + "th largest element: " + arr[n - k]);
24         } else {
25             System.out.println("k is greater than the size of the array.");
26         }
27     }
28 }

```

Enter number of elements: 4
 Enter elements:
 74
 85
 79
 76
 Enter value of k: 100
 k is greater than the size of the array.

25. Write a program to count the number of negative, positive, and zero elements in an array.

```

array25.java  array15.java  array16.java  array17.java  array18.java  array19.java  array20.java  Console X
1 package arrayPrograms;
2 import java.util.Scanner;
3
4 public class array25 {
5
6     public static void main(String[] args) {
7         Scanner sc = new Scanner(System.in);
8         System.out.print("Enter Number of elements: ");
9         int n = sc.nextInt();
10        int[] arr = new int[n];
11
12        System.out.print("Enter elements: ");
13        for (int i = 0; i < n; i++) {
14            arr[i] = sc.nextInt();
15        }
16
17        int positive = 0, negative = 0, zero = 0;
18
19        for (int i = 0; i < n; i++) {
20            if (arr[i] > 0) {
21                positive++;
22            } else if (arr[i] < 0) {
23                negative++;
24            } else {
25                zero++;
26            }
27        }
28
29        System.out.println("Positive numbers: " + positive);
30        System.out.println("Negative numbers: " + negative);
31        System.out.println("Zeroes: " + zero);
32
33    }
34 }

```

Enter Number of elements:
 4
 1
 0
 0
 Positive numbers: 4
 Negative numbers: 0
 Zeroes: 0

Data Types

1. Write a program to demonstrate the use of primitive data types in Java

```
array25.java array23.java array24.java datatype1.java X "95
1 package datatypesprograms;
2 import java.util.Scanner;
3
4 public class datatype1 {
5
6     public static void main(String[] args) {
7         Scanner sc = new Scanner(System.in);
8
9         int intValue = 10;
10        double doubleValue = 20.5;
11        boolean boolValue = true;
12        char charValue = 'A';
13
14        System.out.println("int: " + intValue + ", double: " + doub
15    }
16
17 }
```

```
Console X
<terminated> datatype1 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
int: 10. double: 20.5. boolean: true. char: A
```

2. Write a program to perform arithmetic operations using float and double data types

```
array25.java datatype1.java datatype10.java datatype2.java X "94
1 package datatypesprograms;
2 import java.util.Scanner;
3 public class datatype2 {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7
8         System.out.print("Enter a float value: ");
9         float f = sc.nextFloat();
10        System.out.print("Enter a double value: ");
11        double d = sc.nextDouble();
12
13        System.out.println("Sum: " + (f + d));
14        System.out.println("Difference: " + (f - d));
15        System.out.println("Product: " + (f * d));
16        System.out.println("Division: " + (f / d));
17    }
18
19
20 }
```

```
Console X
<terminated> datatype2 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win3
Enter a float value: 3.4
Enter a double value: 344566
Sum: 344569.40000009537
Difference: -344562.59999990463
Product: 1171524.4328603745
Division: 9.867485751256454E-6
```

3. Write a program to convert an integer to binary and vice versa

The screenshot shows the Eclipse IDE interface. The top part displays the code for `datatype3.java`. The code reads an integer from the user, prints it in binary, and then reads a binary string from the user, converts it back to an integer, and prints the result. The bottom part shows the `Console` tab with the application's output:

```
<terminated> datatype3 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Enter an integer: 234
Binary: 11101010
Enter binary string: 100111011
Integer: 315
```

4. Write a program to perform operations on characters such as converting lowercase to uppercase and vice versa

The screenshot shows the Eclipse IDE interface. The top part displays the code for `datatype4.java`. The code reads a character from the user and prints it in uppercase if it was lowercase, or in lowercase if it was uppercase. The bottom part shows the `Console` tab with the application's output:

```
<terminated> datatype4 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Enter a character: I
Uppercase: I
```

5. Write a program to demonstrate the use of boolean data type

```
datatype2.java datatype3.java datatype4.java datatype5.java
1 package datatypesprograms;
2 import java.util.Scanner;
3 public class datatype5 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a boolean value (true/false): ");
8         boolean b = sc.nextBoolean();
9
10        System.out.println("You entered: " + b);
11    }
12}
13
14
15
```

The screenshot shows the Eclipse IDE interface. The top part displays five Java files: datatype2.java, datatype3.java, datatype4.java, datatype5.java (which is the active file), and datatype5.java. The code for datatype5.java is shown above. The bottom part shows the 'Console' view with the following output:

```
Console <terminated> datatype5 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win
Enter a boolean value (true/false): true
You entered: true
```

6. Write a program to demonstrate the use of arrays in Java

```
datatype3.java datatype4.java datatype5.java datatype6.java
1 package datatypesprograms;
2 import java.util.Scanner;
3 public class datatype6 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter the size of the array: ");
8         int size = sc.nextInt();
9         int[] arr = new int[size];
10
11        System.out.println("Enter " + size + " elements:");
12        for (int i = 0; i < size; i++) {
13            arr[i] = sc.nextInt();
14        }
15
16        System.out.println("Array elements are:");
17        for (int i : arr) {
18            System.out.println(i);
19        }
20    }
}
Console <terminated> datatype6 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win
Enter the size of the array: 4
Enter 4 elements:
23
45
67
88
Array elements are:
23
45
67
88
```

7. Write a program to find the maximum and minimum values in an array of integers.

```
datatype4.java  datatype5.java  datatype6.java  datatype7.java < *95
1 package datatypesprograms;
2 import java.util.Scanner;
3 public class datatype7 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter the size of the array: ");
8         int size = sc.nextInt();
9         int[] arr = new int[size];
10
11        System.out.println("Enter " + size + " elements:");
12        for (int i = 0; i < size; i++) {
13            arr[i] = sc.nextInt();
14        }
15
16        int max = arr[0], min = arr[0];
17        for (int i = 1; i < arr.length; i++) {
18            if (arr[i] > max) max = arr[i];
19            if (arr[i] < min) min = arr[i];
20        }
21
22        System.out.println("Max: " + max + ", Min: " + min);
23    }
24}
25
26
```

Console <terminated> datatype7 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Enter the size of the array: 3
Enter 3 elements:
23
44
55
Max: 55, Min: 23

8. Write a program to calculate the average of elements in an array

```
datatype5.java  datatype6.java  datatype7.java  datatype8.java < *95
1 package datatypesprograms;
2 import java.util.Scanner;
3 public class datatype8 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter the size of the array: ");
8         int size = sc.nextInt();
9         int[] arr = new int[size];
10
11        System.out.println("Enter " + size + " elements:");
12        int sum = 0;
13        for (int i = 0; i < size; i++) {
14            arr[i] = sc.nextInt();
15            sum += arr[i];
16        }
17
18        double average = (double) sum / size;
19        System.out.println("Average: " + average);
20    }
21
22
23
24
```

Console <terminated> datatype8 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Enter the size of the array: 4
Enter 4 elements:
2
34
55
32
Average: 30.75

9. Write a program to find the length of a string

The screenshot shows the Eclipse IDE interface. The top part displays the code for datatype9.java:

```
1 package datatypesprograms;
2 import java.util.Scanner;
3 public class datatype9 {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7
8         System.out.print("Enter a string: ");
9         String str = sc.next();
10
11        System.out.println("Length of the string: " + str.length())
12    }
13
14
15 }
```

The bottom part shows the Console tab with the program's output:

```
<terminated> datatype9 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\
Enter a string: ammu is beautiful
Length of the string: 4
```

10. Write a program to concatenate two strings.

The screenshot shows the Eclipse IDE interface. The top part displays the code for datatype10.java:

```
1 package datatypesprograms;
2 import java.util.Scanner;
3 public class datatype10 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter first string: ");
8         String str1 = sc.next();
9         System.out.print("Enter second string: ");
10        String str2 = sc.next();
11
12        String result = str1 + str2;
13        System.out.println("Concatenated string: " + result);
14    }
15 }
```

The bottom part shows the Console tab with the program's output:

```
<terminated> datatype10 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\
Enter first string: amma
Enter second string: nanna
Concatenated string: ammananna
```

11. Write a program to demonstrate the use of wrapper classes in Java.

The screenshot shows the Eclipse IDE interface. The top bar has tabs for datatype10.java, datatype8.java, datatype9.java, datatype11.java (selected), and datatype11.java. The code editor contains the following Java code:

```
1 package datatypesprograms;
2
3 public class datatype11 {
4     public static void main(String[] args) {
5         int num = 100;
6         Integer wrapperInt = Integer.valueOf(num);
7         System.out.println("Wrapped Integer: " + wrapperInt);
8         int unwrapped = wrapperInt.intValue();
9         System.out.println("Unwrapped Integer: " + unwrapped);
10    }
11
12
13
14}
```

The bottom window is the Console, showing the application's output:

```
<terminated> datatype11 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Wrapped Integer: 100
Unwrapped Integer: 100
```

12. Write a program to convert a string to integer and vice versa.

The screenshot shows the Eclipse IDE interface. The top bar has tabs for datatype10.java, datatype9.java, datatype11.java, datatype12.java (selected), and datatype12.java. The code editor contains the following Java code:

```
1 package datatypesprograms;
2 import java.util.Scanner;
3 public class datatype12 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a string representing a number: ");
8         String str = sc.next();
9
10        int num = Integer.parseInt(str);
11        System.out.println("Converted integer: " + num);
12
13        String newStr = String.valueOf(num);
14        System.out.println("Converted back to string: " + newStr);
15    }
16
17
18}
```

The bottom window is the Console, showing the application's output:

```
<terminated> datatype12 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Enter a string representing a number: 45
Converted integer: 45
Converted back to string: 45
```

13. Write a program to demonstrate the use of string methods such as substring, index Of and equals.

The screenshot shows the Eclipse IDE interface with several tabs at the top: datatype10.java, datatype11.java, datatype12.java, datatype13.java, and datatype13.java. The datatype13.java tab is active. The code in the editor is:

```
1 package datatypesprograms;
2 import java.util.Scanner;
3 public class datatype13 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a string: ");
8         String str = sc.next();
9
10        System.out.print("Enter a substring to search for: ");
11        String subStr = sc.next();
12
13        System.out.println("Substring (first 3 characters): " + str.substring(0, 3));
14        System.out.println("Index of substring: " + str.indexOf(subStr));
15        System.out.println("Are the strings equal? " + str.equals(subStr));
16    }
17 }
18
```

Below the editor is a 'Console' window showing the output of the program:

```
<terminated> datatype13 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\workspace\datatype13\src\datatype13.java
Enter a string: representation
Enter a substring to search for: present
Substring (first 3 characters): rep
Index of substring: 2
Are the strings equal? false
```

14. Write a program to count the occurrences of a character in a string.

The screenshot shows the Eclipse IDE interface with several tabs at the top: datatype11.java, datatype12.java, datatype13.java, datatype14.java, and datatype14.java. The datatype14.java tab is active. The code in the editor is:

```
1 package datatypesprograms;
2 import java.util.Scanner;
3 public class datatype14 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a string: ");
8         String str = sc.next();
9
10        System.out.print("Enter the character to count: ");
11        char ch = sc.next().charAt(0);
12
13        int count = 0;
14        for (int i = 0; i < str.length(); i++) {
15            if (str.charAt(i) == ch) {
16                count++;
17            }
18        }
19
20    }
21 }
```

Below the editor is a 'Console' window showing the output of the program:

```
<terminated> datatype14 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\workspace\datatype14\src\datatype14.java
Enter a string: rama
Enter the character to count: a
Character 'a' appears 2 times.
```

15. Write a program to check whether a given string is palindrome.

```
datatype12.java datatype13.java datatype14.java datatype15.java >95
1 package datatypesprograms;
2 import java.util.Scanner;
3 public class datatype15 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a string: ");
8         String str = sc.next();
9
10        String reversed = new StringBuilder(str).reverse().toString();
11        if (str.equals(reversed)) {
12            System.out.println("Palindrome");
13        } else {
14            System.out.println("Not a Palindrome");
15        }
16    }
17 }
18
```

Console X
<terminated> datatype15 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\workspace\datatype15\src\datatype15.java
Enter a string: brother
Not a Palindrome

16. Write a program to reverse a string without using any built-in methods.

```
datatype13.java datatype14.java datatype15.java datatype16.java >95
1 package datatypesprograms;
2 import java.util.Scanner;
3 public class datatype16 {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7
8         System.out.print("Enter a string: ");
9         String str = sc.next();
10
11        String reversed = "";
12        for (int i = str.length() - 1; i >= 0; i--) {
13            reversed += str.charAt(i);
14        }
15
16        System.out.println("Reversed string: " + reversed);
17    }
18 }
```

Console X
<terminated> datatype16 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\workspace\datatype16\src\datatype16.java
Enter a string: tiger
Reversed string: regit

17. Write a program to convert a string to uppercase and vice versa.

The screenshot shows the Eclipse IDE interface. The top bar has tabs for datatype14.java, datatype15.java, datatype16.java, datatype17.java (which is the active tab), and datatype18.java. The code editor contains the following Java code:

```
1 package datatypesprograms;
2 import java.util.Scanner;
3 public class datatype17 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a string: ");
8         String str = sc.next();
9
10        System.out.println("Uppercase: " + str.toUpperCase());
11        System.out.println("Lowercase: " + str.toLowerCase());
12    }
13
14
15
16}
```

The bottom part of the interface is the 'Console' view, which displays the program's output:

```
<terminated> datatype17 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32->
Enter a string: lion
Uppercase: LION
Lowercase: lion
```

18. Write a program to demonstrate the use of StringBuilder class.

The screenshot shows the Eclipse IDE interface. The top bar has tabs for datatype15.java, datatype16.java, datatype17.java, and datatype18.java (which is the active tab). The code editor contains the following Java code:

```
1 package datatypesprograms;
2
3 public class datatype18 {
4     public static void main(String[] args) {
5         StringBuilder sb = new StringBuilder("Hello");
6
7         sb.append(" World");
8         sb.insert(5, ",");
9         sb.delete(6, 11);
10        sb.reverse();
11
12        System.out.println("StringBuilder result: " + sb.toString());
13    }
14
15
16}
```

The bottom part of the interface is the 'Console' view, which displays the program's output:

```
<terminated> datatype18 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64->
StringBuilder result: d,olleH
```

19. Write a program to find the factorial of a number using BigInteger class.

The screenshot shows the Eclipse IDE interface. The top part displays the code for `datatype19.java` which calculates the factorial of a given number using `BigInteger`. The bottom part shows the `Console` tab with the output of the program. The user enters `16` and the program outputs `Factorial of 16: 20922789888000`.

```
datatype19.java datatype17.java datatype18.java datatype19.java × "q5"
1 package datatypesprogams;
2 import java.math.BigInteger;
3 public class datatype19 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a number: ");
8         int num = sc.nextInt();
9
10        BigInteger factorial = BigInteger.ONE;
11        for (int i = 1; i <= num; i++) {
12            factorial = factorial.multiply(BigInteger.valueOf(i));
13        }
14
15        System.out.println("Factorial of " + num + ": " + factorial);
16    }
17 }
18
19
```

Console ×

```
<terminated> datatype19 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Enter a number: 16
Factorial of 16: 20922789888000
```

20. Write a program to demonstrate the use of arrays of objects.

The screenshot shows the Eclipse IDE interface. The top part displays the code for `datatype20.java`, which creates an array of `String` objects and prints them. The bottom part shows the `Console` tab with the output of the program. The user enters `3` and the program outputs three names: `ret`, `gfd`, and `abc`.

```
datatype17.java datatype18.java datatype19.java datatype20.java × "q5"
1 package datatypesprogams;
2 import java.util.Scanner;
3 public class datatype20 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter the number of objects: ");
8         int n = sc.nextInt();
9         String[] objects = new String[n];
10
11        System.out.println("Enter the names of " + n + " objects:");
12        for (int i = 0; i < n; i++) {
13            objects[i] = sc.next();
14        }
15
16        System.out.println("Objects array: ");
17        for (String object : objects) {
18            System.out.println(object);
19        }
20    }
21 }
```

Console ×

```
<terminated> datatype20 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_1
Enter the number of objects: 3
Enter the names of 3 objects:
ret
gfd
abc
Objects array:
ret
gfd
abc
```

21. Write a program to sort an array of integers in ascending and descending order.

The screenshot shows the Eclipse IDE interface with several Java files listed in the top bar: datatype18.java, datatype19.java, datatype20.java, datatype21.java, and datatype21.java. The main window displays the code for datatype21.java:

```

1 package datatypesprograms;
2
3 public class datatype21 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter the size of the array: ");
8         int size = sc.nextInt();
9         int[] arr = new int[size];
10
11        System.out.println("Enter " + size + " elements:");
12        for (int i = 0; i < size; i++) {
13            arr[i] = sc.nextInt();
14        }
15
16        Arrays.sort(arr);
17        System.out.println("Ascending order: " + Arrays.toString(ar
18
19

```

Below the code editor is the Eclipse Console window:

```

Console X
<terminated> datatype21 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter the size of the array: 4
Enter 4 elements:
75
12
63
85
Ascending order: [12, 63, 75, 85]
Descending order: [85, 75, 63, 12]

```

22. Write a program to find the second largest and second smallest elements in an array

The screenshot shows the Eclipse IDE interface with several Java files listed in the top bar: datatype19.java, datatype20.java, datatype21.java, datatype22.java, and datatype22.java. The main window displays the code for datatype22.java:

```

1 package datatypesprograms;
2 import java.util.Arrays;
3
4 public class datatype22 {
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7
8         System.out.print("Enter the size of the array: ");
9         int size = sc.nextInt();
10        int[] arr = new int[size];
11
12        System.out.println("Enter " + size + " elements:");
13        for (int i = 0; i < size; i++) {
14            arr[i] = sc.nextInt();
15        }
16
17        Arrays.sort(arr);
18        System.out.println("Second smallest: " + arr[1]);
19        System.out.println("Second largest: " + arr[arr.length - 2]
20    }
21
22
23

```

Below the code editor is the Eclipse Console window:

```

Console X
<terminated> datatype22 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter the size of the array: 4
Enter 4 elements:
75
12
84
96
Second smallest: 75
Second largest: 84

```

23. Write a program to find the sum of diagonal elements of a matrix

```

datatype20java datatype21java datatype22java datatype23.java × "qs"
1 package datatypesprograms;
2 import java.util.Scanner;
3 public class datatype23 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter the size of the matrix (n x n): ");
8         int n = sc.nextInt();
9         int[][] matrix = new int[n][n];
10        int diagonalSum = 0;
11
12        System.out.println("Enter the matrix elements:");
13        for (int i = 0; i < n; i++) {
14            for (int j = 0; j < n; j++) {
15                matrix[i][j] = sc.nextInt();
16                if (i == j) {
17                    diagonalSum += matrix[i][j];
18                }
19            }
20        }
21
22        System.out.println("Sum of diagonal elements: " + diagonalSum);
}

```

Console ×

```

<terminated> datatype23 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\datatype23\src\datatype23.java
Enter the size of the matrix (n x n): 3
Enter the matrix elements:
45
41
52
63
7
8
94
5
Sum of diagonal elements: 57

```

24. Write a program to transpose a matrix.

```

datatype21java datatype22.java datatype23.java datatype24.java × "qs"
1 package datatypesprograms;
2 import java.util.Scanner;
3 public class datatype24 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter the size of the matrix (n x n): ");
8         int n = sc.nextInt();
9         int[][] matrix = new int[n][n];
10        int[][] transpose = new int[n][n];
11
12        System.out.println("Enter the matrix elements:");
13        for (int i = 0; i < n; i++) {
14            for (int j = 0; j < n; j++) {
15                matrix[i][j] = sc.nextInt();
16            }
17        }
18
19        // Transpose the matrix
20        for (int i = 0; i < n; i++) {
21            for (int j = 0; j < n; j++) {
22                transpose[j][i] = matrix[i][j];
}

```

Console ×

```

<terminated> datatype24 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\datatype24\src\datatype24.java
Enter the size of the matrix (n x n): 2
Enter the matrix elements:
45
89
56
32
Transpose of the matrix:
45 56
89 32

```

25. Write a program to multiply two matrices.

```

1 datatype5.java 2 datatype6.java 3 datatype7.java 4 datatype8.java 5 datatype9.java 6 datatype10.java 7 datatype11.java 8 datatype12.java 9 datatype13.java 10 datatype14.java 11 datatype15.java 12 datatype16.java 13 datatype17.java 14 datatype18.java 15 datatype19.java 16 datatype20.java 17 datatype21.java 18 datatype22.java 19 datatype23.java 20 datatype24.java 21 datatype25.java 22 Console ...
23 terminated> datatype25 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win
24 Enter the number of rows and columns of the first matrix: 2
25
26 Enter elements of the first matrix:
27
28 10 20
29
30 Enter the number of rows and columns of the second matrix: 2
31
32 Enter elements of the second matrix:
33
34 30 40
35
36 Resultant matrix after multiplication:
37 2040 2080
38 2085 1985

```

STRINGS

1. Write a program to reverse a string

```

datatype23.java datatype24.java datatype25.java String1.java ...
1 package Stringsprograms;
2 import java.util.Scanner;
3 public class String1 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a string: ");
8         String str = sc.nextLine();
9
10        String reversed = new StringBuilder(str).reverse();
11        System.out.println("Reversed string: " + reversed);
12    }
13}
14
15
16

```

```

Console ...
terminated> String1 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win
Enter a string: rose flower
Reversed string: rewolf esor

```

2. Write a program to check whether a given string is palindrome or not

```
datatype24.java datatype25.java String1.java String2.java > 95
1 package Stringsprograms;
2 import java.util.Scanner;
3 public class String2 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a string: ");
8         String str = sc.nextLine();
9
10        String reversed = new StringBuilder(str).reverse().toString();
11
12        if (str.equals(reversed)) {
13            System.out.println("The string is a palindrome.");
14        } else {
15            System.out.println("The string is not a palindrome.");
16        }
17    }
18}
19
20
```

```
Console <
<terminated> String2 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter a string: random
The string is not a palindrome.
```

3. Write a program to count the number of vowels and consonants in a string.

```
datatype25java String1.java String2.java String3.java > 95
1 package Stringsprograms;
2 import java.util.Scanner;
3 public class String3 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a string: ");
8         String str = sc.nextLine().toLowerCase();
9
10        int vowels = 0, consonants = 0;
11
12        for (int i = 0; i < str.length(); i++) {
13            char ch = str.charAt(i);
14            if (ch >= 'a' && ch <= 'z') {
15                if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o'
16                    || ch == 'u')
17                    vowels++;
18                else
19                    consonants++;
20            }
21        }
22
23        System.out.println("Vowels: " + vowels);
24        System.out.println("Consonants: " + consonants);
25    }
26}
```

```
Console <
<terminated> String3 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\
Enter a string: peacock
Vowels: 3
Consonants: 4
```

18. Write a program to count the occurrences of a character in a string.

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows files String1.java, String2.java, String3.java, string4.java, and string5.java.
- Code Editor:** Displays the content of string4.java. The code uses Scanner to read a string and a character from the user, then counts the occurrences of the character in the string.
- Console:** Shows the output of running the program. It prompts for a string ("Enter a string: ") and a character ("Enter the character to count: "). The user inputs "dog is an animal" and "a". The output shows that the character 'a' appears 4 times.

```
String1.java  String2.java  String3.java  string4.java  string5.java
1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string4 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a string: ");
8         String str = sc.nextLine();
9
10        System.out.print("Enter the character to count: ");
11        char ch = sc.next().charAt(0);
12
13        int count = 0;
14        for (int i = 0; i < str.length(); i++) {
15            if (str.charAt(i) == ch) {
16                count++;
17            }
18        }
19
20        System.out.println("Character '" + ch + "' appears " + coun
21    }
22}

```

```
Console ×
<terminated> string4 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Enter a string: dog is an animal
Enter the character to count: a
Character 'a' appears 4 times.
```

18. Write a program to remove all white spaces from a string.

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows files String2.java, String3.java, string4.java, and string5.java.
- Code Editor:** Displays the content of string5.java. The code reads a string from the user and then removes all white spaces from it using the replaceAll method.
- Console:** Shows the output of running the program. It prompts for a string ("Enter a string: ") and then prints the string without any spaces.

```
String2.java  String3.java  string4.java  string5.java  string6.java
1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string5 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a string: ");
8         String str = sc.nextLine();
9
10        String result = str.replaceAll("\\s+", "");
11        System.out.println("String without spaces: " + result);
12    }
13}

```

```
Console ×
<terminated> string5 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Enter a string: varchar is a datatype
String without spaces: varcharisdatatype
```

6. Write a program to find the length of a string.

The screenshot shows the Eclipse IDE interface. The top bar has tabs for "String3.java", "String4.java", "String5.java", "String6.java", and "String6.java X". The code editor contains the following Java code:

```
1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string6 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a string: ");
8         String str = sc.nextLine();
9
10        System.out.println("Length of the string: " + str.length())
11    }
12 }
13
14
```

The console window below shows the output of running the program:

```
<terminated> string6 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Enter a string: java application
Length of the string: 16
```

8. Write a program to convert lowercase letters to uppercase and vice versa.

The screenshot shows the Eclipse IDE interface. The top bar has tabs for "String3.java", "String4.java", "String5.java", "String6.java", and "String7.java X". The code editor contains the following Java code:

```
1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string7 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a string: ");
8         String str = sc.nextLine();
9
10        System.out.println("Uppercase: " + str.toUpperCase());
11        System.out.println("Lowercase: " + str.toLowerCase());
12    }
13 }
14
```

The console window below shows the output of running the program:

```
<terminated> string7 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Enter a string: program is a set of instructions
Uppercase: PROGRAM IS ASET OF INSTRUCTIONS
Lowercase: program is a set of instructions
```

7. Write a program to concatenate two strings.

```
string4.java string5.java string6.java string7.java string8.java »4
1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string8 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter first string: ");
8         String str1 = sc.nextLine();
9
10        System.out.print("Enter second string: ");
11        String str2 = sc.nextLine();
12
13        String result = str1 + str2;
14        System.out.println("Concatenated string: " + result);
```

Console X

```
<terminated> string8 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter first string: raju
Enter second string: gari gadhi
Concatenated string: rajugari gadhi
```

9. Write a program to find the longest word in a string.

```
string5.java string6.java string7.java string8.java string9.java »4
1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string9 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a sentence: ");
8         String str = sc.nextLine();
9
10        String[] words = str.split(" ");
11        String longest = words[0];
12
13        for (String word : words) {
14            if (word.length() > longest.length()) {
15                longest = word;
16            }
17        }
18
19        System.out.println("Longest word: " + longest);
20    }
21 }
```

Console X

```
<terminated> string9 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\es
Enter a sentence: Write a program to find the longest word in a string
Longest word: program
```

10. Write a program to check whether two strings are anagrams or not.

```

string7.java   string8java   string9java   string10java X 95
1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string10 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter first string: ");
8         String str1 = sc.nextLine();
9
10        System.out.print("Enter second string: ");
11        String str2 = sc.nextLine();
12
13        if (str1.length() != str2.length()) {
14            System.out.println("Not anagrams.");
15        } else {
16            char[] arr1 = str1.toCharArray();
17            char[] arr2 = str2.toCharArray();
18            java.util.Arrays.sort(arr1);
19            java.util.Arrays.sort(arr2);
20
21            if (java.util.Arrays.equals(arr1, arr2)) {
22                System.out.println("The strings are anagrams.");
23            } else {
24                System.out.println("Not anagrams.");
25            }
26        }
27    }
28 }

Console X
<terminated> string10 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter first string: water
Enter second string: vapour
Not anagrams.

```

11. Write a program to find the first non-repeated character in a string

```

string8java   string9java   string10java   string11java X 95
1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string11 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a string: ");
8         String str = sc.nextLine();
9
10        LinkedHashMap<Character, Integer> charCount = new LinkedHashMap<Character, Integer>();
11
12        for (int i = 0; i < str.length(); i++) {
13            char c = str.charAt(i);
14            charCount.put(c, charCount.getOrDefault(c, 0) + 1);
15        }
16
17        for (char c : charCount.keySet()) {
18            if (charCount.get(c) == 1) {
19                System.out.println("First non-repeated character: " + c);
20                return;
21            }
22        }
23
24        System.out.println("No non-repeated character found.");
25    }
26
27
28 }

Console X
<terminated> string11 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter a string: string
First non-repeated character: s

```

12. Write a program to check whether a string contains only digits or not.

```

string8java string9java string10java string11java × "95
 1 package Stringsprograms;
 2 import java.util.Scanner;
 3 public class string11 {
 4     public static void main(String[] args) {
 5         Scanner sc = new Scanner(System.in);
 6
 7         System.out.print("Enter a string: ");
 8         String str = sc.nextLine();
 9
10         LinkedHashMap<Character, Integer> charCount = new LinkedHashMap<Character, Integer>();
11
12         for (int i = 0; i < str.length(); i++) {
13             char c = str.charAt(i);
14             charCount.put(c, charCount.getOrDefault(c, 0) + 1);
15         }
16
17         for (char c : charCount.keySet()) {
18             if (charCount.get(c) == 1) {
19                 System.out.println("First non-repeated character: " + c);
20                 return;
21             }
22         }
23
24         System.out.println("No non-repeated character found.");
25     }
26 }
27
28
29

```

Console ×

```
<terminated> string12 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter a string: 23anna
The string does not contain only digits.
```

13. Write a program to find all permutations of a string

```

string9java string10java string11java string13java × "95
 1
 2
 3
 4
 5
 6
 7
 8
 9
 10
 11
 12
 13     permute(str, 0, str.length() - 1);
 14
 15
 16
 17     private static void permute(String str, int l, int r) {
 18         if (l == r) {
 19             System.out.println(str);
 20         } else {
 21             for (int i = l; i <= r; i++) {
 22                 str = swap(str, l, i);
 23                 permute(str, l + 1, r);
 24                 str = swap(str, l, i); // Backtrack
 25             }
 26         }
 27
 28     private static String swap(String str, int i, int j) {
 29         char temp;
 30         char[] arr = str.toCharArray();
 31         temp = arr[i];
 32         arr[i] = arr[j];
 33         arr[j] = temp;
 34         return new String(arr);
 35     }
 36 }

Console ×           
```

```
<terminated> string13 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter a string: eat
eat
eta
aet
ate
tae
tea
```

18. Write a program to find the words in of each character in a string.

The screenshot shows the Eclipse IDE interface with two tabs open: 'string14.java' and 'Console'. The code in 'string14.java' is as follows:

```
1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string14 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         System.out.print("Enter a string: ");
7         String str = sc.nextLine();
8         HashMap<Character, Integer> map = new HashMap<>();
9         for (int i = 0; i < str.length(); i++) {
10             char ch = str.charAt(i);
11             map.put(ch, map.getOrDefault(ch, 0) + 1);
12         }
13         System.out.println("Character occurrences: " + map);
14     }
15 }
```

The 'Console' tab shows the application's output:

```
<terminated> string14 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_
Enter a string: download
Character occurrences: {a=1, d=2, w=1, l=1, m=1, o=2}
```

15. Write a program to reverse words a sentence.

The screenshot shows the Eclipse IDE interface with two tabs open: 'string15.java' and 'Console'. The code in 'string15.java' is as follows:

```
1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string15 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         System.out.print("Enter a sentence: ");
7         String sentence = sc.nextLine();
8         String[] words = sentence.split(" ");
9         StringBuilder reversedSentence = new StringBuilder();
10        for (int i = words.length - 1; i >= 0; i--) {
11            reversedSentence.append(words[i]).append(" ");
12        }
13        System.out.println("Reversed sentence: " + reversedSentence);
14    }
15 }
```

The 'Console' tab shows the application's output:

```
<terminated> string15 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\java
Enter a sentence: java in eclipse
Reversed sentence: eclipse in java
```

18. Write a program to find the duplicate characters in a string.

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows files string13.java, string14.java, string15.java, string16.java, and string17.java.
- Code Editor:** Displays the Java code for string16.java. The code uses a HashMap to count character occurrences and then prints out any character that appears more than once.
- Console:** Shows the output of running the program. It prompts for a string ("Enter a string: ") and receives "eclipse". It then prints "Duplicate characters: e".

```
1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string16 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a string: ");
8         String str = sc.nextLine();
9
10        HashMap<Character, Integer> map = new HashMap<>();
11        System.out.print("Duplicate characters: ");
12        for (int i = 0; i < str.length(); i++) {
13            char ch = str.charAt(i);
14            map.put(ch, map.getOrDefault(ch, 0) + 1);
15        }
16
17        for (char ch : map.keySet()) {
18            if (map.get(ch) > 1) {
19                System.out.print(ch + " ");
20            }
21        }
22    }
23 }
24 }
```

```
<terminated> string16 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_6
Enter a string: eclipse
Duplicate characters: e
```

18. Write a program to find the first repeating character in a string.

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows files string14.java, string15.java, string16.java, string17.java, and string18.java.
- Code Editor:** Displays the Java code for string17.java. It uses a HashSet to track characters seen so far. As it iterates through the string, it checks if the current character is already in the set. If it is, it prints the character and returns. If not, it adds the character to the set.
- Console:** Shows the output of running the program. It prompts for a string ("Enter a string: ") and receives "smallest window". It then prints "First repeating character: l".

```
1 package Stringsprograms;
2 import java.util.Scanner;
3
4 public class string17 {
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7
8         System.out.print("Enter a string: ");
9         String str = sc.nextLine();
10
11        HashSet<Character> set = new HashSet<>();
12        for (int i = 0; i < str.length(); i++) {
13            char ch = str.charAt(i);
14            if (!set.add(ch)) {
15                System.out.println("First repeating character: " +
16                    return;
17            }
18        }
19        System.out.println("No repeating character found.");
20    }
21 }
22 }
23 }
24 }
```

```
<terminated> string17 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_6
Enter a string: smallest window
First repeating character: l
```

18 Write a program to capitalize the first letter of each word in a sentence.

```

1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string18 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a sentence: ");
8         String sentence = sc.nextLine();
9
10        String[] words = sentence.split(" ");
11        StringBuilder capitalizedSentence = new StringBuilder();
12
13        for (String word : words) {
14            capitalizedSentence.append(word.substring(0, 1).toUpperCase());
15        }
16
17        System.out.println("Capitalized sentence: " + capitalizedSentence);
18    }
19
20
21
22

```

Console

```

<terminated> string18 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter a sentence: check whether a string is a rotation
Capitalized sentence: Check Whether A String Is A Rotation

```

19. Write a program to check whether a string is a rotation of another string.

```

1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string19 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter first string: ");
8         String str1 = sc.nextLine();
9
10        System.out.print("Enter second string: ");
11        String str2 = sc.nextLine();
12
13        if (str1.length() != str2.length()) {
14            System.out.println("Strings are not rotations of each other.");
15        } else {
16            String temp = str1 + str1;
17            if (temp.contains(str2)) {
18                System.out.println("The second string is a rotation of the first string.");
19            } else {
20                System.out.println("Strings are not rotations of each other.");
21            }
22        }
23    }

```

Console

```

<terminated> string19 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Enter first string: papa
Enter second string: apap
Strings are not rotations of each other.

```

20. Write a program to check whether a string is a substring of another string.

```
string17.java string18.java string19.java string20.java >95
1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string20 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter first string: ");
8         String str1 = sc.nextLine();
9
10        System.out.print("Enter second string: ");
11        String str2 = sc.nextLine();
12
13        if (str1.contains(str2)) {
14            System.out.println("The second string is a substring of");
15        } else {
16            System.out.println("The second string is not a substrin");
17        }
18    }
19
20
21
```



```
Console X
<terminated> string20 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\...
Enter first string: user
Enter second string: name
The second string is not a substring of the first string.
```

21. Write a program to remove duplicates from a string.

```
string18.java string19.java string20.java string21.java >95
1 package Stringsprograms;
2 import java.util.Scanner;
3
4 public class string21 {
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7
8         System.out.print("Enter a string: ");
9         String str = sc.nextLine();
10
11         LinkedHashSet<Character> set = new LinkedHashSet<>();
12         for (char c : str.toCharArray()) {
13             set.add(c);
14         }
15
16         StringBuilder result = new StringBuilder();
17         for (char c : set) {
18             result.append(c);
19         }
20
21         System.out.println("String without duplicates: " + result.t
22     }
23
24 }
```



```
Console X
<terminated> string21 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_6...
Enter a string: bottle
String without duplicates: bottle
```

22. Write a program to find the common characters in two given strings.

```

1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string22 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter first string: ");
8         String str1 = sc.nextLine();
9
10        System.out.print("Enter second string: ");
11        String str2 = sc.nextLine();
12
13        HashSet<Character> set1 = new HashSet<>();
14        HashSet<Character> set2 = new HashSet<>();
15        for (char c : str1.toCharArray()) {
16            set1.add(c);
17        }
18
19        for (char c : str2.toCharArray()) {
16            set2.add(c);
17        }
20
21        set1.retainAll(set2);
22
23
24    }

```

Console X

```

<terminated> string22 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32->
Enter first string: girl
Enter second string: boy
Common characters: []

```

23. Write a program to check whether a given string is a pangram or not.

```

1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string23 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter a string: ");
8         String str = sc.nextLine().toLowerCase();
9
10        boolean[] alphabet = new boolean[26];
11        int index = 0;
12
13        for (int i = 0; i < str.length(); i++) {
14            if (str.charAt(i) >= 'a' && str.charAt(i) <= 'z') {
15                index = str.charAt(i) - 'a';
16                alphabet[index] = true;
17            }
18        }
19
20        boolean isPangram = true;
21        for (int i = 0; i < 26; i++) {
22            if (!alphabet[i]) {
23                isPangram = false;
24            }
25        }
26
27    }

```

Console X

```

<terminated> string23 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64>
Enter a string: smart
The string is not a pangram.

```

24. Write a program to find the smallest window in a string containing all characters of another string.

```

1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string24 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter first string: ");
8         String str1 = sc.nextLine();
9
10        System.out.print("Enter second string: ");
11        String str2 = sc.nextLine();
12
13        System.out.println("Smallest window: " + smallestWindow(str1, str2));
14    }
15
16    public static String smallestWindow(String str1, String str2) {
17        if (str1.length() < str2.length()) {
18            return "No such window";
19        }
20
21        int[] count = new int[256];
22        for (int i = 0; i < str2.length(); i++) {
23            count[str2.charAt(i)]++;
}

```

Console output:

```

<terminated> string24 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Enter first string: plugins
Enter second string: in java
Smallest window: No such window

```

25. Write a program to find the longest common prefix among an array of strings.

```

1 package Stringsprograms;
2 import java.util.Scanner;
3 public class string25 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         System.out.print("Enter the number of strings: ");
8         int n = sc.nextInt();
9         sc.nextLine(); // Consume newline
10
11        String[] strArray = new String[n];
12
13        System.out.println("Enter the strings:");
14        for (int i = 0; i < n; i++) {
15            strArray[i] = sc.nextLine();
16        }
17
18        System.out.println("Longest common prefix: " + longestCommonPrefix(strArray));
19    }
20
21    public static String longestCommonPrefix(String[] strs) {
22        if (strs == null || strs.length == 0) {
23            return "";
}

```

Console output:

```

<terminated> string25 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R
Enter the number of strings: 2
Enter the strings:
backspace

```

Encapsulation

- 1.create a class representing student with private member variables (name, roll number, age) and public methods (getters and setters).

The screenshot shows the Eclipse IDE interface. The code editor displays a Java class named 'Phone' with private member variables 'brand', 'model', and 'price', and public accessor methods. The terminal window below shows the execution of the code and its output.

```
1 import java.util.*;
2 class Phone{
3     private String brand;
4     private String model;
5     private float price;
6     public void setBrand(String brand) {
7         this.brand=brand;
8     }
9     public void setModel(String model) {
10        this.model=model;
11    }
12    public void setPrice(float price) {
13        this.price=price;
14    }
15    public String getBrand() {
16        return brand;
17    }
18    public String getModel() {
19        return model;
20    }
21    public float getPrice() {
22    }
23 }
```

```
<terminated>:Encapsulation3 [Java Application] C:\Program Files\Java\jdk-23\bin\javaw.exe 2 Mar 2023 12:41:57 am - 12:43:11 am elapsed: 00:01:14 [pid:21612]
enter brand name
oppoF19pro
enter model of mobile
9025
enter the price
23500
Phone brand is oppoF19pro
Phone model is 9025
Phone price is 23500.0
```

2. Write a program to demonstrate encapsulation by accessing private member variables Through public accessor methods.

The screenshot shows the Eclipse IDE interface. The code editor displays a Java class named 'Student' with private member variables 'name', 'rollnumber', and 'age', and public accessor methods. The terminal window below shows the execution of the code and its output.

```
1 import java.util.*;
2 class Student{
3     private String name;
4     private int rollnumber;
5     private int age;
6     public void setName(String name) {
7         this.name=name;
8     }
9     public void setRollnumber(int rollnumber) {
10        this.rollnumber=rollnumber;
11    }
12    public void setAge(int age) {
13        this.age=age;
14    }
15    public String getName() {
16        return name;
17    }
18    public int getRollnumber() {
19        return rollnumber;
20    }
21    public int getAge() {
22    }
23 }
```

```
<terminated>:Encapsulation1 [Java Application] C:\Program Files\Java\jdk-23\bin\javaw.exe 2 Mar 2023 12:47:08 am - 12:47:20 am elapsed: 00:00:12 [pid:10256]
enter name
anusha
enter roll number
21
enter age
21
Student name is anusha
Student roll number is 21
Student age is 21
```

● eclipse-workspace - Samplesrc.com/Bankaccount.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Simple.java Multiplication Transpose Diagonal Inverses SumofIntegers Max Java Multiplication Abstraction Animals.java Bankaccount .java

```
1 package com;
2 public class Bankaccount {
3     public static void main(String[] args) {
4         abstract class Bankaccount {
5             protected double balance;
6             public abstract void deposit(double amount);
7             public abstract void withdraw(double amount);
8             public double getbalance() {
9                 return balance;
10            }
11        }
12        class SavingsAccount extends Bankaccount {
13            @Override
14            public void deposit(double amount) {
15                if (amount > 0) {
16                    balance += amount;
17                    System.out.println("Deposited " + amount + " into Savings Account. Current Balance: " + balance);
18                } else {
19                    System.out.println("Deposit amount must be positive.");
20                }
21            }
22        }
23    }
24 }
```

Console x

```
<terminated: Bankaccount [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (25 Feb 2025, 3:05:55 am - 3:05:57 am Elapsed 00:00:18) [pid 17006]
Deposited 1000.0 into Savings Account. Current Balance: 1000.0
Withdraw 500.0 from Savings Account. Current Balance: 500.0
Deposited 2000.0 into Current Account. Current Balance: 2000.0
Withdraw 1500.0 from Current Account. Current Balance: 500.0
Savings Account Balance: 500.0
Current Account Balance: 500.0
```

The screenshot shows the Eclipse IDE interface. The top menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help. The left sidebar shows a package structure with a single package named 'com'. The main editor area contains the following Java code:

```
private String model;
private String color;
private int price;
public void setModel(String model) {
    this.model=model;
}
public void setColor(String color) {
    this.color=color;
}
public void setPrice(int price) {
    this.price=price;
}
String getModel() {
    return model;
}
String getColor() {
    return color;
}
int getPrice() {
}
```

The bottom right corner of the editor has a small preview window showing the code's output.

The bottom part of the interface is a terminal window titled 'Console' with the following text:

```
-generated: E:\eclipse\JavaApplication1\src\Main.java (2 Mar 2015, 12:47:01 pm - 12.460 ms elapsed, 00026305 [line: 4])
```

```
enter name of the car
bmw
enter color
grey
enter price
200000
Car name is bmw
Car color is grey
Car price is 200000
```

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - Sample03/Encapsulation2.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard toolbar icons.
- Project Explorer:** Shows multiple Java files under the Encapsulation package.
- Code Editor:** Displays the Java code for the Book class:

```
1 import java.util.Scanner;
2
3 class Book {
4     private String title;
5     private String author;
6     private int price;
7     public void setTitle(String title) {
8         this.title=title;
9     }
10    public void setAuthor(String author) {
11        this.author=author;
12    }
13    public void setPrice(int price) {
14        this.price=price;
15    }
16    public String getTitle() {
17        return title;
18    }
19    public String getAuthor() {
20        return author ;
21    }
22 }
```
- Console:** Shows the output of the application's execution:

```
<terminated> Encapsulation2 [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (2 Mar 2025, 12:44:14 am - 124ms elapsed 0003165) [pid 2662]
enter title of the book
romanesque
enter author of the book
valmiki
enter price of the book
79
The name of book romanesque
the author of the book valmiki
The price of book is79
```

The screenshot shows the Eclipse IDE interface with two tabs open. The top tab displays a Java class named 'Student' with private attributes 'name' and 'age', and public methods to set and get these values. The bottom tab shows the console output where the program prompts for name and age, and then prints the student's name, roll number, and age.

```
1 package sample;
2
3 import java.util.*;
4
5 class Student{
6     private String name;
7     private int rollnumber;
8     private int age;
9     public void setName(String name) {
10         this.name=name;
11     }
12     public void setRollnumber(int rollnumber) {
13         this.rollnumber=rollnumber;
14     }
15     public void setAge(int age) {
16         this.age=age;
17     }
18     public String getName() {
19         return name;
20     }
21     public int getRollnumber() {
22         return rollnumber;
23     }
24     public int getAge() {
25
26 }
27 }
28
29
30 <Console>
31 <terminated> Encapsulation (Java Application) C:\Program Files\java\jdk-23\bin\java.exe (2 Mar 2023 12:47:38 am - 12:47:39 am elapsed: 30013.720) [pid:10256]
32 enter name:
33 enter name: anusha
34 enter roll number:
35 enter roll number: 21
36 enter age:
37 enter age: 21
38 Student name is anusha
39 Student roll number is 21
40 Student age is 21
```

The screenshot shows the Eclipse IDE interface with two tabs open. The top tab displays a Java class named 'Circle' with a static final variable 'pi=3.14', a private attribute 'radius', and public methods to set and get the radius and area. The bottom tab shows the console output where the program calculates and prints the area and circumference of a circle with a radius of 7.

```
1 package sample;
2
3 import java.util.*;
4
5 class Circle{
6     private static final double pi=3.14;
7     private int radius;
8     private double area,circum;
9     public void setRadius(int r) {
10         radius=r;
11     }
12     public double getArea() {
13         area=pi*radius*radius;
14         return area;
15     }
16     public double getCircum() {
17         circum=2*pi*radius;
18         return circum;
19     }
20 }
21
22 public class EncapsulationII {
23
24     public static void main(String[] args) {
25
26 }
27 }
28
29 <Console>
30 <terminated> EncapsulationII (Java Application) C:\Program Files\java\jdk-23\bin\java.exe (2 Mar 2023 12:48:52 am - 12:48:53 am elapsed: 30005.590) [pid:1253]
31 enter radius:
32 enter radius: 7
33 The area of circle is 153.99999999999999
34 The circumference of circle is 43.98
```

```
edgecumputer:~$ javac Encapsulation1.java & java Encapsulation1
enter employee name: ravi
enter employee id number: 11001
enter salary: 20000
Student name is: ravi
Student roll number is: 11001
Student age is: 20000.0
```

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - Sample10/Encapsulation (Java - Eclipse IDE)
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard toolbar icons.
- Left Margin:** Shows line numbers from 1 to 20.
- Code Editor:** Contains the following Java code:

```
1 import java.util.*;  
2 class Phone{  
3     private String brand;  
4     private String model;  
5     private float price;  
6     public void setBrand(String brand) {  
7         this.brand=brand;  
8     }  
9     public void setModel(String model) {  
10        this.model=model;  
11    }  
12    public void setPrice(float price) {  
13        this.price=price;  
14    }  
15    public String getBrand() {  
16        return brand;  
17    }  
18    public String getModel() {  
19        return model;  
20    }  
21    public float getPrice() {  
22        return price;  
23    }  
24}
```
- Console View:** Shows the terminal output:

```
terminated> Encapsulation$ Java Application [C:\Program Files\Java\jdk-23\bin\javaw.exe] [2 Mar 2023 12:41:57 am – 12:43:31 am elapsed: 00:01:34.039] [pid: 2161]  
enter brand name  
oppoF19pro  
enter model of mobile  
R9s  
enter the price  
2500  
Phone brand is oppoF19pro  
Phone model is R9s  
Phone price is 2500.0
```

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - Sample10/Encapsulation03.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard toolbar icons for file operations.
- Left Margin:** Shows a list of open files: Encapsulation01, Encapsulation02, Encapsulation03, Encapsulation04, Encapsulation05, Encapsulation06, BankAccount, Encapsulation08, Encapsulation09, and Encapsulation010.
- Code Editor:** Displays the Java code for `Encapsulation03`. The code defines a `Rectangle` class with private attributes `length` and `breadth`, and a protected attribute `area`. It includes methods to set the length and breadth, and a method to calculate the area.
- Console:** Shows the terminal output of the application. The application prompts for length and breadth, calculates the area as 40.0, and exits.

OVERLOADING

1. Write a program to find the sum of two integers.

The screenshot shows the Eclipse IDE interface with several open tabs at the top. The active tab is 'overloading1.java'. The code defines a class 'overloading1' with two static methods: 'sum' for integers and 'sum' for floats. It uses a Scanner to read input from the user and prints the sum to the console. The 'Console' tab at the bottom shows the output of running the application, where it prompts for two integers (456 and 289), calculates their sum (7245), then prompts for two floats (34.5 and 78.6), calculates their sum (113.1), and finally exits.

```
1 package overloading;
2 import java.util.Scanner;
3 public class overloading1 {
4
5     public static int sum(int a, int b) {
6         return a + b;
7     }
8
9     // Overloaded method for floats
10    public static float sum(float a, float b) {
11        return a + b;
12    }
13
14    public static void main(String[] args) {
15        Scanner sc = new Scanner(System.in);
16
17        System.out.print("Enter first integer: ");
18        int num1 = sc.nextInt();
19        System.out.print("Enter second integer: ");
20        int num2 = sc.nextInt();
21        System.out.println("Sum of integers: " + sum(num1, num2));
22
23        System.out.print("Enter first float: ");
24
25        System.out.print("Enter second float: ");
26        float num3 = sc.nextFloat();
27        System.out.println("Sum of floats: " + sum(num1, num2));
28    }
29}
```

```
Console ×
<terminated> overloading1 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Enter first integer: 456
Enter second integer: 289
Sum of integers: 7245
Enter first float: 34.5
Enter second float: 78.6
Sum of floats: 113.1
```

2. Write a program to find the sum of two floats.

The screenshot shows the Eclipse IDE interface with several open tabs at the top. The active tab is 'overloading2.java'. The code defines a class 'overloading2' with a static method 'sum' for floats. It uses a Scanner to read input from the user and prints the sum to the console. The 'Console' tab at the bottom shows the output of running the application, where it prompts for two floats (78.5 and 97.33), calculates their sum (175.83), and finally exits.

```
1 package overloading;
2 import java.util.Scanner;
3 public class overloading2 {
4
5     public static float sum(float a, float b) {
6         return a + b;
7     }
8
9     public static void main(String[] args) {
10        Scanner sc = new Scanner(System.in);
11
12        System.out.print("Enter first float: ");
13        float num1 = sc.nextFloat();
14        System.out.print("Enter second float: ");
15        float num2 = sc.nextFloat();
16        System.out.println("Sum of floats: " + sum(num1, num2));
17    }
18}
```

```
Console ×
<terminated> overloading2 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32
Enter first float: 78.5
Enter second float: 97.33
Sum of floats: 175.83
```

3. Write a program to find the area of a rectangle.

```

string25.java  overloading...  overloading...  overloading...  overloading...  ×  »9%
3  public class overloading3 {
4
5      // Overloaded method for integer dimensions
6      public static int area(int length, int width) {
7          return length * width;
8      }
9
10     // Overloaded method for float dimensions
11     public static float area(float length, float width) {
12         return length * width;
13     }
14
15     public static void main(String[] args) {
16         Scanner sc = new Scanner(System.in);
17
18         System.out.print("Enter length of rectangle (integer): ");
19         int length1 = sc.nextInt();
20         System.out.print("Enter width of rectangle (integer): ");
21         int width1 = sc.nextInt();
22         System.out.println("Area of rectangle: " + area(length1, wi
23
24         System.out.print("Enter length of rectangle (float): ");
25         float length2 = sc.nextFloat();
26         System.out.print("Enter width of rectangle (float): ");

```

Console ×

```

<terminated> overloading3 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Enter length of rectangle (integer): 74
Enter width of rectangle (integer): 963
Area of rectangle: 71262
Enter length of rectangle (float): 4.8
Enter width of rectangle (float): 6.77
Area of rectangle: 32.496002

```

4. Write a program to find the volume of a cylinder.

```

overloading...  overloading...  overloading...  overloading...  ×  »9%
1  package overloading;
2  import java.util.Scanner;
3  public class overloading4 {
4
5      public static double volume(int radius, int height) {
6          return Math.PI * Math.pow(radius, 2) * height;
7      }
8
9      // Overloaded method for float radius and height
10     public static double volume(float radius, float height) {
11         return Math.PI * Math.pow(radius, 2) * height;
12     }
13
14     public static void main(String[] args) {
15         Scanner sc = new Scanner(System.in);
16
17         System.out.print("Enter radius of cylinder (integer): ");
18         int radius1 = sc.nextInt();
19         System.out.print("Enter height of cylinder (integer): ");
20         int height1 = sc.nextInt();
21         System.out.println("Volume of cylinder: " + volume(radius1,
22
23         System.out.print("Enter radius of cylinder (float): ");
24         float radius2 = sc.nextFloat();

```

Console ×

```

<terminated> overloading4 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x
Enter radius of cylinder (integer): 14
Enter height of cylinder (integer): 23
Volume of cylinder: 14162.299682382787
Enter radius of cylinder (float): 74.5
Enter height of cylinder (float): 94.2
Volume of cylinder: 1642529.9865179162

```

5. Write a program to calculate the factorial of a number.

```
overloading... overloading... overloading... overloading... >95
2 import java.util.Scanner;
3 public class overloading5 {
4
5     public static int factorial(int n) {
6         int fact = 1;
7         for (int i = 1; i <= n; i++) {
8             fact *= i;
9         }
10        return fact;
11    }
12
13    // Method for long factorial
14    public static long factorial(long n) {
15        long fact = 1;
16        for (long i = 1; i <= n; i++) {
17            fact *= i;
18        }
19        return fact;
20    }
21
22    public static void main(String[] args) {
23        Scanner sc = new Scanner(System.in);
24
25        System.out.print("Enter an integer to find factorial: ");
26
27        int num = sc.nextInt();
28
29        long result = factorial(num);
30
31        System.out.println("Factorial of " + num + " is: " + result);
32    }
33}
```

6. Write a program to find the average of three numbers

```
1 package overloading;
2 import java.util.Scanner;
3 public class overloading6 {
4
5     public static double average(int a, int b, int c) {
6         return (a + b + c) / 3.0;
7     }
8
9     public static void main(String[] args) {
10        Scanner sc = new Scanner(System.in);
11
12        System.out.print("Enter first number: ");
13        int num1 = sc.nextInt();
14        System.out.print("Enter second number: ");
15        int num2 = sc.nextInt();
16        System.out.print("Enter third number: ");
17        int num3 = sc.nextInt();
18
19        System.out.println("Average of three numbers: " + average(n
20    }
21
22
23 }
```

Console >

<terminated> overloading6 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_

Enter first number: 45

Enter second number: 12

Enter third number: 23

Average of three numbers: 26.666666666666668

7. Write a program to find the maximum of two numbers.

```

1 package overloading;
2 import java.util.Scanner;
3 public class overloading7 {
4     public static int max(int a, int b) {
5         return (a > b) ? a : b;
6     }
7
8     public static float max(float a, float b) {
9         return (a > b) ? a : b;
10    }
11
12    public static void main(String[] args) {
13        Scanner sc = new Scanner(System.in);
14
15        System.out.print("Enter first integer: ");
16        int int1 = sc.nextInt();
17        System.out.print("Enter second integer: ");
18        int int2 = sc.nextInt();
19        System.out.println("Maximum of two integers: " + max(int1,
20
21        System.out.print("Enter first float: ");
22        float float1 = sc.nextFloat();
23        System.out.print("Enter second float: ");
24        float float2 = sc.nextFloat();

```

Console X

```

<terminated> overloading7 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86.
Enter first integer: 58
Enter second integer: 54
Maximum of two integers: 58
Enter first float: 78945
Enter second float: 12398.12575
Maximum of two floats: 78945.0

```

8. Write a program to find the minimum of two numbers.

```

1 package overloading;
2 import java.util.Scanner;
3 public class overloading8 {
4
5     public static int min(int a, int b) {
6         return (a < b) ? a : b;
7     }
8
9     public static float min(float a, float b) {
10    return (a < b) ? a : b;
11 }
12
13    public static void main(String[] args) {
14        Scanner sc = new Scanner(System.in);
15
16        System.out.print("Enter first integer: ");
17        int int1 = sc.nextInt();
18        System.out.print("Enter second integer: ");
19        int int2 = sc.nextInt();
20        System.out.println("Minimum of two integers: " + min(int1,
21
22        System.out.print("Enter first float: ");
23        float float1 = sc.nextFloat();
24        System.out.print("Enter second float: ");
25        float float2 = sc.nextFloat();
26        System.out.println("Minimum of two floats: " + min(float1,
27    }
28
29

```

Console X

```

<terminated> overloading8 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86.
Enter first integer: 4785
Enter second integer: 55566
Minimum of two integers: 4785
Enter first float: 124579.144
Enter second float: 1257896.34456
Minimum of two floats: 124579.14

```

9. Write a program to calculate the power of a number.

```

1 package overloading;
2 import java.util.Scanner;
3 public class overloading9 {
4     public static void main(String[] args) {
5         Scanner scanner = new Scanner(System.in);
6         System.out.print("Enter the base number: ");
7         double base = scanner.nextDouble();
8         System.out.print("Enter the exponent (int): ");
9         int exponentInt = scanner.nextInt();
10        System.out.print("Enter the exponent (double): ");
11        double exponentDouble = scanner.nextDouble();
12
13        System.out.println("Result with int exponent: " + power(base, exponentInt));
14        System.out.println("Result with double exponent: " + power(base, exponentDouble));
15    }
16
17    public static double power(double base, int exponent) {
18        return Math.pow(base, exponent);
19    }
20
21    public static double power(double base, double exponent) {
22        return Math.pow(base, exponent);
23    }
24
25}

```

Console X

```

<terminated> overloading9 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_
Enter the base number: 45
Enter the exponent (int): 78
Enter the exponent (double): 5423
Result with int exponent: 8.924339317248996E128
Result with double exponent: Infinity

```

10. Write a program to check whether a number is prime or not.

```

1 package overloading;
2 public class overloading10 {
3
4     public static void main(String[] args) {
5         Scanner scanner = new Scanner(System.in);
6         System.out.print("Enter a number: ");
7         int number = scanner.nextInt();
8
9         System.out.println("Is " + number + " prime (int method)? ");
10        System.out.println("Is " + number + " prime (long method)? ");
11    }
12
13
14    public static boolean isPrime(int number) {
15        if (number <= 1) return false;
16        for (int i = 2; i <= Math.sqrt(number); i++) {
17            if (number % i == 0) {
18                return false;
19            }
20        }
21        return true;
22    }
23
24    public static boolean isPrime(long number) {
25        if (number <= 1) return false;
26        for (long i = 2; i <= Math.sqrt(number); i++) {
27            if (number % i == 0) {
28                return false;
29            }

```

Console X

```

<terminated> overloading10 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_
Enter a number: 23
Is 23 prime (int method)? true
Is 23 prime (long method)? true

```

11. Write a program to find the square root of a number

```

1 package overloading;
2 import java.util.Scanner;
3
4 public class overloading11 {
5
6     public static void main(String[] args) {
7         Scanner scanner = new Scanner(System.in);
8         System.out.print("Enter a number (int): ");
9         int numInt = scanner.nextInt();
10        System.out.print("Enter a number (double): ");
11        double numDouble = scanner.nextDouble();
12
13        System.out.println("Square root (int method): " + sqrt(numInt));
14        System.out.println("Square root (double method): " + sqrt(numDouble));
15    }
16
17    public static double sqrt(int num) {
18        return Math.sqrt(num);
19    }
20
21    public static double sqrt(double num) {
22        return Math.sqrt(num);
23    }
24}
25
26
27

```

Console X

```

<terminated> overloading11 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Enter a number (int): 4785
Enter a number (double): 145789
Square root (int method): 69.17369442208505
Square root (double method): 381.8232575420204

```

12. Write a program to calculate the perimeter of a rectangle.

```

1 package overloading;
2 import java.util.Scanner;
3 public class overloading12 {
4
5     public static void main(String[] args) {
6         Scanner scanner = new Scanner(System.in);
7         System.out.print("Enter the length of the rectangle (int): ");
8         int lengthInt = scanner.nextInt();
9         System.out.print("Enter the width of the rectangle (int): ");
10        int widthInt = scanner.nextInt();
11        System.out.print("Enter the length of the rectangle (double): ");
12        double lengthDouble = scanner.nextDouble();
13        System.out.print("Enter the width of the rectangle (double): ");
14        double widthDouble = scanner.nextDouble();
15
16        System.out.println("Perimeter (int method): " + perimeter(lengthInt, widthInt));
17        System.out.println("Perimeter (double method): " + perimeter(lengthDouble, widthDouble));
18    }
19
20    public static int perimeter(int length, int width) {
21        return 2 * (length + width);
22    }
23
24    public static double perimeter(double length, double width) {
25        return 2 * (length + width);
26    }
27}

```

Console X

```

<terminated> overloading12 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Enter the length of the rectangle (int): 74
Enter the width of the rectangle (int): 58
Enter the length of the rectangle (double): 98
Enter the width of the rectangle (double): 17
Perimeter (int method): 264
Perimeter (double method): 230.0

```

13. Write a program to find the area of a circle.

```

1 package overloading;
2 import java.util.Scanner;
3 public class overloading13 {
4     public static void main(String[] args) {
5         Scanner scanner = new Scanner(System.in);
6         System.out.print("Enter the radius of the circle (int): ");
7         int radiusInt = scanner.nextInt();
8         System.out.print("Enter the radius of the circle (double): ");
9         double radiusDouble = scanner.nextDouble();
10
11         System.out.println("Area (int method): " + area(radiusInt));
12         System.out.println("Area (double method): " + area(radiusDouble));
13     }
14
15     public static double area(int radius) {
16         return Math.PI * radius * radius;
17     }
18
19     public static double area(double radius) {
20         return Math.PI * radius * radius;
21     }
22 }
23
24
25

```

Console output:

```

<terminated> overloading13 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x8
Enter the radius of the circle (int): 47
Enter the radius of the circle (double): 4579.123
Area (int method): 6939.778171779853
Area (double method): 6.5874069135955006E7

```

14. Write a program to find the area of a triangle.

```

5
6     public static void main(String[] args) {
7         Scanner scanner = new Scanner(System.in);
8
9         // For base and height
10        System.out.print("Enter the base of the triangle (int): ");
11        int baseInt = scanner.nextInt();
12        System.out.print("Enter the height of the triangle (int): ");
13        int heightInt = scanner.nextInt();
14
15        // For base and height as double
16        System.out.print("Enter the base of the triangle (double): ");
17        double baseDouble = scanner.nextDouble();
18        System.out.print("Enter the height of the triangle (double): ");
19        double heightDouble = scanner.nextDouble();
20
21        System.out.println("Area using base and height (int): " + a
22        System.out.println("Area using base and height (double): " )
23
24
25        // Method to calculate area using base and height (int)
26        public static double area(int base, int height) {
27            return 0.5 * base * height;
28        }
29
30        // Method to calculate area using base and height (double)
31        public static double area(double base, double height) {

```

Console output:

```

<terminated> overloading14 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x8
Enter the base of the triangle (int): 78
Enter the height of the triangle (int): 74
Enter the base of the triangle (double): 74587
Enter the height of the triangle (double): 45874
Area using base and height (int): 2886.0
Area using base and height (double): 1.710802019E9

```

15. Write a program to convert temperature from Celsius to Fahrenheit.

The screenshot shows the Eclipse IDE interface with two tabs open in the editor: "overloading..." and "overloading15". The "overloading15" tab contains the following Java code:

```
1 package overloading;
2 import java.util.Scanner;
3 public class overloading15 {
4     public static void main(String[] args) {
5         Scanner scanner = new Scanner(System.in);
6         System.out.print("Enter temperature in Celsius (int): ");
7         int celsiusInt = scanner.nextInt();
8         System.out.print("Enter temperature in Celsius (double): ")
9         double celsiusDouble = scanner.nextDouble();
10
11        System.out.println("Celsius to Fahrenheit (int method): " +
12        System.out.println("Celsius to Fahrenheit (double method): ");
13    }
14
15    public static double celsiusToFahrenheit(int celsius) {
16        return (celsius * 9/5.0) + 32;
17    }
18
19    public static double celsiusToFahrenheit(double celsius) {
20        return (celsius * 9/5.0) + 32;
21    }
22}
```

The "Console" tab shows the output of running the program:

```
<terminated> overloading15 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Enter temperature in Celsius (int): 7845
Enter temperature in Celsius (double): 75458
Celsius to Fahrenheit (int method): 14153.0
Celsius to Fahrenheit (double method): 135856.4
```

16. Write a program to convert temperature from Fahrenheit to Celsius.

The screenshot shows the Eclipse IDE interface with two tabs open in the editor: "overloading..." and "overloading16". The "overloading16" tab contains the following Java code:

```
1 package overloading;
2 import java.util.Scanner;
3
4 public class overloading16 {
5     public static void main(String[] args) {
6         Scanner scanner = new Scanner(System.in);
7         System.out.print("Enter temperature in Fahrenheit (int): ");
8         int fahrenheitInt = scanner.nextInt();
9         System.out.print("Enter temperature in Fahrenheit (double): ");
10        double fahrenheitDouble = scanner.nextDouble();
11
12        System.out.println("Fahrenheit to Celsius (int): " + fahrenheitInt);
13        System.out.println("Fahrenheit to Celsius (double): " + fahrenheitDouble);
14    }
15
16    public static double fahrenheitToCelsius(int fahrenheit) {
17        return (fahrenheit - 32) * 5 / 9.0;
18    }
19
20    public static double fahrenheitToCelsius(double fahrenheit) {
21        return (fahrenheit - 32) * 5 / 9.0;
22    }
23}
```

The "Console" tab shows the output of running the program:

```
<terminated> overloading16 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Enter temperature in Fahrenheit (int): 457
Enter temperature in Fahrenheit (double): 12457
Fahrenheit to Celsius (int): 236.1111111111111
Fahrenheit to Celsius (double): 6902.777777777777
```

17. Write a program to find the volume of a sphere.

The screenshot shows the Eclipse IDE interface. At the top, there are four tabs labeled "overloading..." which are all pointing to the same code editor window. The code in the editor is as follows:

```
1 package overloading;
2 import java.util.Scanner;
3 public class overloading17 {
4     public static void main(String[] args) {
5         Scanner scanner = new Scanner(System.in);
6         System.out.print("Enter radius (int): ");
7         int radiusInt = scanner.nextInt();
8         System.out.print("Enter radius (double): ");
9         double radiusDouble = scanner.nextDouble();
10
11        System.out.println("Volume (int method): " + volume(radiusInt));
12        System.out.println("Volume (double method): " + volume(radiusDouble));
13    }
14
15    public static double volume(int radius) {
16        return (4 / 3.0) * Math.PI * Math.pow(radius, 3);
17    }
18
19    public static double volume(double radius) {
20        return (4 / 3.0) * Math.PI * Math.pow(radius, 3);
21    }
22 }
23
24
25
```

Below the code editor is the Eclipse Console window, titled "Console X". It displays the output of the program's execution:

```
<terminated> overloading17 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_
Enter radius (int): 4578
Enter radius (double): 74589
Volume (int method): 4.018981196899893E11
Volume (double method): 1.738252903058805E15
```

18. Write a program to find the average of an array of integers.

The screenshot shows the Eclipse IDE interface. At the top, there are four tabs labeled "overloading..." which are all pointing to the same code editor window. The code in the editor is as follows:

```
1 package overloading;
2 import java.util.Scanner;
3 public class overloading18 {
4
5     public static void main(String[] args) {
6         Scanner scanner = new Scanner(System.in);
7         System.out.print("Enter the number of elements: ");
8         int n = scanner.nextInt();
9         int[] arr = new int[n];
10
11        System.out.println("Enter elements: ");
12        for (int i = 0; i < n; i++) {
13            arr[i] = scanner.nextInt();
14        }
15
16        System.out.println("Average: " + average(arr));
17    }
18
19    public static double average(int[] arr) {
20        double sum = 0;
21        for (int num : arr) {
22            sum += num;
23        }
24        return sum / arr.length;
25    }
26 }
```

Below the code editor is the Eclipse Console window, titled "Console X". It displays the output of the program's execution:

```
<terminated> overloading18 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_
Enter the number of elements: 5
Enter elements:
12
45
78
96
63
Average: 58.8
```

19. Write a program to calculate the compound interest.

```
overloading... overloading... overloading... overloading... > "os"
1 package overloading;
2 import java.util.Scanner;
3 public class overloading19 {
4     public static void main(String[] args) {
5         Scanner scanner = new Scanner(System.in);
6         System.out.print("Enter principal amount: ");
7         double principal = scanner.nextDouble();
8         System.out.print("Enter annual interest rate (in %): ");
9         double rate = scanner.nextDouble();
10        System.out.print("Enter time in years: ");
11        int time = scanner.nextInt();
12        System.out.print("Enter number of times interest is compounded per year: ");
13        int n = scanner.nextInt();
14
15        System.out.println("Compound Interest: " + compoundInterest(principal, rate, time, n));
16    }
17
18    public static double compoundInterest(double principal, double rate, int time, int n) {
19        return principal * Math.pow(1 + (rate / (100 * n)), n * time);
20    }
21
22
23
24}
```

```
Console >
<terminated> overloading19 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Enter principal amount: 78945
Enter annual interest rate (in %): 20
Enter time in years: 5
Enter number of times interest is compounded per year: 7
Compound Interest: 132662.27944538492
```

20. Write a program to find the area of a trapezoid.

```
overloading... overloading... overloading... overloading... > "os"
1 package overloading;
2 import java.util.Scanner;
3 public class overloading20 {
4     public static void main(String[] args) {
5         Scanner scanner = new Scanner(System.in);
6         System.out.print("Enter the first base length (int): ");
7         int base1Int = scanner.nextInt();
8         System.out.print("Enter the second base length (int): ");
9         int base2Int = scanner.nextInt();
10        System.out.print("Enter the height (int): ");
11        int heightInt = scanner.nextInt();
12
13        System.out.println("Area (int method): " + area(base1Int, base2Int, heightInt));
14    }
15
16    public static double area(int base1, int base2, int height) {
17        return 0.5 * (base1 + base2) * height;
18    }
19
20
21
22}
```

```
Console >
<terminated> overloading20 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Enter the first base length (int): 74
Enter the second base length (int): 554
Enter the height (int): 45
Area (int method): 14130.0
```

21. Write a program to find the area of a parallelogram.

```
overloading... overloading... overloading... overloading... >95
1 package overloading;
2 import java.util.Scanner;
3 public class overloading21 {
4
5     public static void main(String[] args) {
6         Scanner scanner = new Scanner(System.in);
7         System.out.print("Enter base (int): ");
8         int baseInt = scanner.nextInt();
9         System.out.print("Enter height (int): ");
10        int heightInt = scanner.nextInt();
11
12        System.out.println("Area (int method): " + area(baseInt, he
13    }
14
15    public static double area(int base, int height) {
16        return base * height;
17    }
18}
19
20
21
```

```
Console >
<terminated> overloading21 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32
Enter base (int): 45
Enter height (int): 124
Area (int method): 5580.0
```

22. Write a program to calculate the simple interest.

```
overloading... overloading... overloading... overloading... >95
1 package overloading;
2 import java.util.Scanner;
3
4 public class overloadidng22 {
5
6     public static void main(String[] args) {
7         Scanner scanner = new Scanner(System.in);
8         System.out.print("Enter principal amount: ");
9         double principal = scanner.nextDouble();
10        System.out.print("Enter annual interest rate (in %): ");
11        double rate = scanner.nextDouble();
12        System.out.print("Enter time in years: ");
13        int time = scanner.nextInt();
14
15        System.out.println("Simple Interest: " + simpleInterest(prin
16    }
17
18    public static double simpleInterest(double principal, double ra
19        return (principal * rate * time) / 100;
20}
```



```
Console >
<terminated> overloadidng22 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Enter principal amount: 4500
Enter annual interest rate (in %): 23
Enter time in years: 4
Simple Interest: 4140.0
```

23. Write a program to find the area of a square.

```
1 package overloading;
2 import java.util.Scanner;
3 public class overloading23 {
4     public static void main(String[] args) {
5         Scanner scanner = new Scanner(System.in);
6         System.out.print("Enter side length (int): ");
7         int sideInt = scanner.nextInt();
8         System.out.print("Enter side length (double): ");
9         double sideDouble = scanner.nextDouble();
10
11        System.out.println("Area (int method): " + area(sideInt));
12        System.out.println("Area (double method): " + area(sideDoub
13    }
14
15    public static double area(int side) {
16        return side * side;
17    }
18
19    public static double area(double side) {
20        return side * side;
21    }
22}
```

```
Console X
<terminated> overloading23 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_
Enter side length (int): 90
Enter side length (double): 80
Area (int method): 8100.0
Area (double method): 6400.0
```

24. Write a program to find the area of a rhombus.

```
1 package overloading;
2 import java.util.Scanner;
3 public class overloading24 {
4     public static void main(String[] args) {
5         Scanner scanner = new Scanner(System.in);
6         System.out.print("Enter diagonals (int): ");
7         int diagonal1Int = scanner.nextInt();
8         int diagonal2Int = scanner.nextInt();
9
10        System.out.println("Area (int method): " + area(diagonal1In
11    }
12
13    public static double area(int diagonal1, int diagonal2) {
14        return 0.5 * diagonal1 * diagonal2;
15    }
16
17}
```

```
Console X
<terminated> overloading24 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_
Enter diagonals (int): 956
345
Area (int method): 164910.0
```

25. Write a program to find the area of a regular polygon.

```

1 package overloading;
2 import java.util.Scanner;
3 public class overloading25 {
4     public static void main(String[] args) {
5         Scanner scanner = new Scanner(System.in);
6         System.out.print("Enter the number of sides: ");
7         int n = scanner.nextInt();
8         System.out.print("Enter the side length (int): ");
9         int sideLengthInt = scanner.nextInt();
10        System.out.print("Enter the side length (double): ");
11        double sideLengthDouble = scanner.nextDouble();
12
13        System.out.println("Area (int method): " + area(n, sideLengthInt));
14        System.out.println("Area (double method): " + area(n, sideLengthDouble));
15    }
16
17    public static double area(int n, int sideLength) {
18        return (n * sideLength * sideLength) / (4 * Math.tan(Math.PI / n));
19    }
20
21    public static double area(int n, double sideLength) {
22        return (n * sideLength * sideLength) / (4 * Math.tan(Math.PI / n));
23    }
24 }

```

Console

```

<terminated> overloading25 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\overloading25
Enter the number of sides: 5
Enter the side length (int): 34
Enter the side length (double): 3456
Area (int method): 1988.8718750808457
Area (double method): 2.054927196208098E7

```

Static

1 Write a program to count the number of objects created for a class using a static variable

```

1 package staticprograms;
2 public class static1 {
3     static int count = 0;
4
5     public static1() {
6         count++;
7     }
8
9     public static void main(String[] args) {
10        static1 obj1 = new static1();
11        static1 obj2 = new static1();
12        static1 obj3 = new static1();
13
14        System.out.println("Number of objects created: " + count);
15    }
16 }

```

Console

```

<terminated> static1 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\static1
Number of objects created: 3

```

2. Implement a static method to find the factorial of a number

```
overloading... static1java static2java static3java static4java
1 package staticprograms;
2 public class static2 {
3     public static int static2(int n) {
4         if (n == 0 || n == 1) {
5             return 1;
6         }
7         return n * static2(n - 1);
8     }
9
10    public static void main(String[] args) {
11        int num = 5;
12        System.out.println("Factorial of " + num + " is " + static2
13    }
14
15
16
17
```

Console <terminated> static2 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ecli
Factorial of 5 is 120

3. Write a program to calculate the area of a circle using a static method

```
overloading... static1java static2java static3java static4java
1 package staticprograms;
2 public class static3 {
3     public static double area(double radius) {
4         return Math.PI * radius * radius;
5     }
6
7     public static void main(String[] args) {
8         double radius = 7;
9         System.out.println("Area of the circle: " + area(radius));
10    }
11
12
```

Console <terminated> static3 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ecli
Area of the circle: 153.93804002589985

4. Write a program to find the sum of elements in an array using a static method

```
static1java static2java static3java static4java
1 package staticprograms;
2 public class static4 {
3     public static int sum(int[] arr) {
4         int sum = 0;
5         for (int num : arr) {
6             sum += num;
7         }
8         return sum;
9     }
10
11     public static void main(String[] args) {
12         int[] arr = {1, 2, 3, 4, 5};
13         System.out.println("Sum of elements in the array: " + sum(a
14     }
15
16
```

Console <terminated> static4 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ecli
Sum of elements in the array: 15

5.
Write a program to find the maximum of two numbers using a static method

```
static1.java × static2.java static3.java static4.java static5.java × »44
practiseprograms/src/staticprograms/static1.java
1
2 public class static5 {
3     public static int max(int a, int b) {
4         return (a > b) ? a : b;
5     }
6
7     public static void main(String[] args) {
8         int a = 10, b = 20;
9         System.out.println("Maximum of " + a + " and " + b + " is "
10    }
11
12 }
13
14

Console ×
<terminated> static5 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\cli
Maximum of 10 and 20 is 20
```

6. Write a program to find the minimum of two numbers using a static method

```
static2.java static3.java static4.java static5.java static6.java × »44
1 package staticprograms;
2
3 public class static6 {
4     public static int min(int a, int b) {
5         return (a < b) ? a : b;
6     }
7
8     public static void main(String[] args) {
9         int a = 10, b = 20;
10        System.out.println("Minimum of " + a + " and " + b + " is "
11    }
12
13 }
```



```
Console ×
<terminated> static6 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\
Minimum of 10 and 20 is 10
```

7. Write a program to find the power of a number using a static method

```
static3.java static4.java static5.java static6.java static7.java × »44
1 package staticprograms;
2
3 public class static7 {
4     public static double power(double base, int exponent) {
5         return Math.pow(base, exponent);
6     }
7
8     public static void main(String[] args) {
9         double base = 2, exponent = 3;
10        System.out.println(base + " raised to the power " + exponent
11    }
12
13 }
14

Console ×
<terminated> static7 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\cli
2.0 raised to the power 3.0 is 8.0
```

8. Write a program to calculate the square root of a number using a static method

```
static4.java static5.java static6.java static7.java sttic8.java "94
1 package staticprograms;
2
3 public class sttic8 {
4     public static double sqrt(double number) {
5         return Math.sqrt(number);
6     }
7
8     public static void main(String[] args) {
9         double number = 25;
10        System.out.println("Square root of " + number + " is " + sq
11    }
12
13 }
```

Console <terminated> sttic8 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Square root of 25.0 is 5.0

9. Write a program to find the area of a triangle using a static method

```
static6.java static7.java sttic8.java static9.java static10.java "94
1 package staticprograms;
2
3 public class static9 {
4     public static double area(double base, double height) {
5         return 0.5 * base * height;
6     }
7
8     public static void main(String[] args) {
9         double base = 5, height = 10;
10        System.out.println("Area of the triangle: " + area(base, he
11    }
12
13
14 }
```

Console <terminated> static9 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Area of the triangle: 25.0

10. Write a program to calculate the simple interest using a static method

```
static6.java static7.java sttic8.java static9.java static10.java "94
1 package staticprograms;
2
3 public class static10 {
4     public static double calculateSimpleInterest(double principal,
5         return (principal * rate * time) / 100;
6     }
7
8     public static void main(String[] args) {
9         double principal = 1000, rate = 5, time = 2;
10        System.out.println("Simple Interest: " + calculateSimpleInt
11    }
12
13 }
```

Console <terminated> static10 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Simple Interest: 100.0

11. Write a program to find the volume of a cylinder using a static method

```
static11.java static9.java static10.java static11.java > 95
1 package staticprograms;
2
3 public class static11 {
4     public static double volume(double radius, double height) {
5         return Math.PI * Math.pow(radius, 2) * height;
6     }
7
8     public static void main(String[] args) {
9         double radius = 7, height = 10;
10        System.out.println("Volume of the cylinder: " + volume(radi
11    }
12
13}
14
```

Console >

```
<terminated> static11 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Volume of the cylinder: 1539.3804002589986
```

12. Write a program to calculate the compound interest using a static method

```
static9.java static10.java static11.java static12.java > 95
1 package staticprograms;
2
3 public class static12 {
4     public static double calculateCompoundInterest(double principal
5         return principal * Math.pow(1 + (rate / n), n * time) - pri
6     }
7
8     public static void main(String[] args) {
9         double principal = 1000, rate = 5, time = 2;
10        int n = 4; // Quarterly compounding
11        System.out.println("Compound Interest: " + calculateCompoun
12    }
13
14
15}
16
```

Console >

```
<terminated> static12 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Compound Interest: 655840.8355712891
```

13 Write a program to find the area of a rectangle using a static method

```
static9.java static11.java static12.java static13.java > 95
1 package staticprograms;
2
3 public class static13 {
4     public static double area(double length, double width) {
5         return length * width;
6     }
7
8     public static void main(String[] args) {
9         double length = 5, width = 8;
10        System.out.println("Area of the rectangle: " + area(length,
11    }
12
13
14
15}
```

Console >

```
<terminated> static13 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Area of the rectangle: 40.0
```

14.

Write a program to find the area of a square using a static method

```
static11java static12java static13java static14java >95
1 package staticprograms;
2
3 public class static14 {
4     public static double area(double side) {
5         return side * side;
6     }
7
8     public static void main(String[] args) {
9         double side = 4;
10        System.out.println("Area of the square: " + area(side));
11    }
12
13 }
```

Console >
<terminated> static14 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ecl
Area of the square: 16.0

15. Write a program to find the area of a rhombus using a static method

```
static12java static13java static14java static15.java >95
1 package staticprograms;
2
3 public class static15 {
4     public static double area(double diagonal1, double diagonal2) {
5         return 0.5 * diagonal1 * diagonal2;
6     }
7
8     public static void main(String[] args) {
9         double diagonal1 = 6, diagonal2 = 8;
10        System.out.println("Area of the rhombus: " + area(diagonal1
11    )
12
13 }
```

Console >
<terminated> static15 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ecl
Area of the rhombus: 24.0

16. Write a program to find the area of a parallelogram using a static method

```
static13java static14java static15java static16.java >95
1 package staticprograms;
2
3 public class static16 {
4     public static double area(double base, double height) {
5         return base * height;
6     }
7
8     public static void main(String[] args) {
9         double base = 5, height = 7;
10        System.out.println("Area of the parallelogram: " + area(base
11    )
12
13 }
```

Console >
<terminated> static16 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ecl
Area of the parallelogram: 35.0

17. Write a program to find the area of a trapezoid using a static method.

The screenshot shows the Eclipse IDE interface with several tabs at the top: static14.java, static15.java, static16.java, static17.java (selected), and static18.java. The static17.java tab contains the following Java code:

```
1 package staticprograms;
2
3 public class static17 {
4     public static double area(double base1, double base2, double height) {
5         return 0.5 * (base1 + base2) * height;
6     }
7
8     public static void main(String[] args) {
9         double base1 = 5, base2 = 7, height = 4;
10        System.out.println("Area of the trapezoid: " + area(base1,
11    })
12
13
14
15}
```

The Console tab at the bottom shows the output: <terminated> static17 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\static17.java Area of the trapezoid: 24.0

18. Write a program to find the area of a regular polygon using a static method.

The screenshot shows the Eclipse IDE interface with several tabs at the top: static15.java, static16.java, static17.java, static18.java (selected), and static19.java. The static18.java tab contains the following Java code:

```
1 package staticprograms;
2
3 public class static18 {
4     public static double area(int sides, double length) {
5         return (sides * length * length) / (4 * Math.tan(Math.PI / sides));
6     }
7
8     public static void main(String[] args) {
9         int sides = 6;
10        double length = 5;
11        System.out.println("Area of the polygon: " + area(sides, length));
12    }
13
14}
```

The Console tab at the bottom shows the output: <terminated> static18 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\static18.java Area of the polygon: 64.9519052838329

19. Write a program to convert temperature from Celsius to Fahrenheit using a static method

The screenshot shows the Eclipse IDE interface with several tabs at the top: static16.java, static17.java, static18.java, static19.java (selected), and static20.java. The static19.java tab contains the following Java code:

```
1 package staticprograms;
2
3 public class static19 {
4     public static double celsiusToFahrenheit(double celsius) {
5         return (celsius * 9/5) + 32;
6     }
7
8     public static void main(String[] args) {
9         double celsius = 25;
10        System.out.println(celsius + " Celsius is " + celsiusToFahrenheit(celsius));
11    }
12
13
14}
```

The Console tab at the bottom shows the output: <terminated> static19 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\static19.java 25.0 Celsius is 77.0 Fahrenheit

20. Write a program to Fahrenheit to Celsius using static S convert temperature from Fahrenheit to Celsius using a static

The screenshot shows the Eclipse IDE interface with several tabs at the top: static17java, static18java, static19.java, static20.java (selected), and static21.java. The static20.java tab contains the following Java code:

```
1 package staticprograms;
2
3 public class static20 {
4     public static double fahrenheitToCelsius(double fahrenheit) {
5         return (fahrenheit - 32) * 5/9;
6     }
7
8     public static void main(String[] args) {
9         double fahrenheit = 77;
10        System.out.println(fahrenheit + " Fahrenheit is " + fahrenh
11    }
12 }
13
```

Below the editor is the Eclipse Console window, which displays the output of the program:

```
<terminated> static20 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ecli
77.0 Fahrenheit is 25.0 Celsius
```

21. Write a program to find the factorial of a number using recursion and a static method.

The screenshot shows the Eclipse IDE interface with several tabs at the top: static18java, static19java, static20.java, static21.java (selected), and static22.java. The static21.java tab contains the following Java code:

```
1 package staticprograms;
2
3 public class static21 {
4     public static int factorial(int n) {
5         if (n == 0) return 1;
6         return n * factorial(n - 1);
7     }
8
9     public static void main(String[] args) {
10        int num = 5;
11        System.out.println("Factorial of " + num + " is " + factori
12    }
13 }
14
15
```

Below the editor is the Eclipse Console window, which displays the output of the program:

```
<terminated> static21 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ecli
Factorial of 5 is 120
```

22 Write a program to find the sum of digits of a number using recursion and a static method.

The screenshot shows the Eclipse IDE interface with several tabs at the top: static19.java, static20.java, static21.java, static22.java (selected), and static23.java. The static22.java tab contains the following Java code:

```
1 package staticprograms;
2
3 public class static22 {
4     public static int sumOfDigits(int n) {
5         if (n == 0) return 0;
6         return n % 10 + sumOfDigits(n / 10);
7     }
8
9     public static void main(String[] args) {
10        int num = 123;
11        System.out.println("Sum of digits of " + num + " is " + sum
12    }
13 }
14
```

Below the editor is the Eclipse Console window, which displays the output of the program:

```
<terminated> static22 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Sum of digits of 123 is 6
```

23. Write a program to find the power of a number using recursion and a static method

```
static20java static21java static22java static23java static24java
1 package staticprograms;
2
3 public class static23 {
4     public static int power(int base, int exponent) {
5         if (exponent == 0) return 1;
6         return base * power(base, exponent - 1);
7     }
8
9     public static void main(String[] args) {
10        int base = 2, exponent = 3;
11        System.out.println(base + " raised to the power " + exponent +
12    }
13 }
14
```

Console X
<terminated> static23 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
2 raised to the power 3 is 8

24. Write a program to find the Fibonacci series using recursion and a static method

```
static21java static22java static23java static24java static25java
1 package staticprograms;
2
3 public class static24 {
4     public static int fibonacci(int n) {
5         if (n <= 1) return n;
6         return fibonacci(n - 1) + fibonacci(n - 2);
7     }
8
9     public static void main(String[] args) {
10        int num = 6;
11        System.out.println("Fibonacci of " + num + " is " + fibonacci(n
12    }
13 }
14
```

Console X
<terminated> static24 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Fibonacci of 6 is 8

25. Write a program to reverse a string using recursion and a static method

```
static22java static23java static24java static25java static26java
1 package staticprograms;
2
3 public class static25 {
4     public static String reverse(String str) {
5         if (str.isEmpty()) return str;
6         return reverse(str.substring(1)) + str.charAt(0);
7     }
8
9     public static void main(String[] args) {
10        String str = "hello";
11        System.out.println("Reversed string: " + reverse(str));
12    }
13 }
14
```

Console X
<terminated> static25 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Reversed string: olleh

Exception Handling:

1. Write a program to handle Array Index Out Of Bounds Exception.

```
① swapningjava ② primenumber... ③ areaoftriang... ④ handle1.java × *95
1 package exceptionHandling;
2
3 public class handle1 {
4
5     public static void main(String[] args) {
6         try {
7             int[] arr = new int[5];
8             arr[10] = 100; // This will throw ArrayIndexOutOfBoundsException
9         } catch (ArrayIndexOutOfBoundsException e) {
10             System.out.println("Exception: Array index out of bound")
11         }
12     }
13
14
15
16
```

Console ×

```
<terminated> handle1 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: Array index out of bounds!
```

2. Implement a program to handle Arithmetic Exception (division by zero).

```
① swapningjava ② primenumber... ③ handle1.java ④ handle2.java × *95
1 package exceptionHandling;
2
3 public class handle2 {
4     public static void main(String[] args) {
5         try {
6             int result = 10 / 0; // This will throw ArithmeticException
7         } catch (ArithmaticException e) {
8             System.out.println("Exception: Cannot divide by zero!")
9         }
10    }
11
12
13
14
```

Console ×

```
<terminated> handle2 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: Cannot divide by zero!
```

3. Write a program to handle Null Pointer Exception.

```
① swapningjava ② handle1.java ③ handle2.java ④ handle3.java × *95
1 package exceptionHandling;
2
3 public class handle3 {
4     public static void main(String[] args) {
5         try {
6             String str = null;
7             System.out.println(str.length()); // This will throw NullPointerException
8         } catch (NullPointerException e) {
9             System.out.println("Exception: Null pointer encountered")
10        }
11    }
12
13
14
15
```

Console ×

```
<terminated> handle3 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: Null pointer encountered!
```

4. Implement a program to handle File Not Found Exception

```
handle1java handle2java handle3java handle4java × "95"
1 package exceptionHandling;
2 import java.io.*;
3 public class handle4 {
4     public static void main(String[] args) {
5         try {
6             FileReader file = new FileReader("nonexistentfile.txt")
7         } catch (FileNotFoundException e) {
8             System.out.println("Exception: File not found!");
9         }
10    }
11}
12
13
14
```

Console ×

```
<terminated> handle4 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: File not found!
```

5. Write a program to handle Number Format Exception.

```
handle2java handle3java handle4java handle5java × "95"
1 package exceptionHandling;
2 public class handle5 {
3     public static void main(String[] args) {
4         try {
5             int num = Integer.parseInt("abc"); // This will throw
6         } catch (NumberFormatException e) {
7             System.out.println("Exception: Invalid number format!");
8         }
9     }
10}
11
12
13
```

Console ×

```
<terminated> handle5 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: Invalid number format!
```

6. Implement a program to handle IO Exception.

```
handle3java handle4java handle5java handle6java × "95"
1 package exceptionHandling;
2 import java.io.*;
3
4 public class handle6 {
5     public static void main(String[] args) {
6         try {
7             FileInputStream file = new FileInputStream("somefile.tx
8         } catch (IOException e) {
9             System.out.println("Exception: I/O error occurred!");
10        }
11    }
12}
13
14
15
```

Console ×

```
<terminated> handle6 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: I/O error occurred!
```

7. Write a program to handle Class Not Found Exception.

```
handle4.java handle5.java handle6.java handle7.java × *95
1 package exceptionHandling;
2 public class handle7 {
3     public static void main(String[] args) {
4         try {
5             Class.forName("NonExistentClass"); // This will throw
6         } catch (ClassNotFoundException e) {
7             System.out.println("Exception: Class not found!");
8         }
9     }
10 }
11
12
13
```

```
Console ×
<terminated> handle7 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: Class not found!
```

8. Implement a program to handle Stack Over flow Error.

```
handle5.java handle6.java handle7.java *handle8.java × *95
1 package exceptionHandling;
2 public class handle8 {
3     public static void recursiveMethod() {
4         recursiveMethod(); // This will throw StackOverflowError
5     }
6
7     public static void main(String[] args) {
8         try {
9             recursiveMethod();
10        } catch (StackOverflowError e) {
11            System.out.println("Error: Stack Overflow!");
12        }
13    }
14
15
16
```

```
Console ×
<terminated> handle8 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Error: Stack Overflow!
```

9. Write a program to handle Negative Array Size Exception

```
handle6.java handle7.java handle8.java handle9.java × *95
1 package exceptionHandling;
2 public class handle9 {
3     public static void main(String[] args) {
4         try {
5             int[] arr = new int[-5]; // This will throw NegativeAr
6         } catch (NegativeArraySizeException e) {
7             System.out.println("Exception: Negative array size!");
8         }
9     }
10
11
12
13
```

```
Console ×
<terminated> handle9 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: Negative array size!
```

10. Implement a program to handle Interrupted Exception.

```
handle7.java handle8.java handle9.java handle10.java >95
1 package exceptionHandling;
2
3 public class handle10 {
4
5     public static void main(String[] args) throws InterruptedException {
6         try {
7             Thread.sleep(5000); // Thread will sleep for 5 seconds
8             Thread.currentThread().interrupt(); // Interrupt the thread
9             Thread.sleep(5000); // This will throw InterruptedException
10        } catch (InterruptedException e) {
11            System.out.println("Exception: Thread was interrupted!");
12        }
13    }
14
15
16
```

Console >
<terminated> handle10 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\Java Exception: Thread was interrupted!

11. Write a program to handle Array Store Exception.

```
handle7.java handle8.java handle9.java handle10.java >95
1 package exceptionHandling;
2
3 public class handle10 {
4
5     public static void main(String[] args) throws InterruptedException {
6         try {
7             Thread.sleep(5000); // Thread will sleep for 5 seconds
8             Thread.currentThread().interrupt(); // Interrupt the thread
9             Thread.sleep(5000); // This will throw InterruptedException
10        } catch (InterruptedException e) {
11            System.out.println("Exception: Thread was interrupted!");
12        }
13    }
14
15
16
```

Console >
<terminated> handle11 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\Java Exception: Array store exception!

12. Implement a program to handle Illegal State Exception.

```
handle8.java handle9.java handle10.java handle12.java >95
1 package exceptionHandling;
2
3 public class handle12 {
4     public static void main(String[] args) {
5         try {
6             throw new IllegalStateException("Illegal state encountered");
7         } catch (IllegalStateException e) {
8             System.out.println("Exception: " + e.getMessage());
9         }
10    }
11
12
13
14
```

Console >
<terminated> handle12 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\Java Exception: Illegal state encountered!

13. Write a program to handle NoSuchElementException.

```
handle9.java handle10.java handle12.java handle13.java handle14.java
1 package exceptionHandling;
2 import java.util.*;
3
4 public class handle13 {
5     public static void main(String[] args) {
6         try {
7             List<String> list = new ArrayList<>();
8             list.add("Element 1");
9             Iterator<String> iterator = list.iterator();
10            iterator.next(); // This will return the first element
11            iterator.next(); // This will throw NoSuchElementException
12        } catch (NoSuchElementException e) {
13            System.out.println("Exception: No such element found!");
14        }
15    }
16 }
```

Console <
<terminated> handle13 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: No such element found!

14. Implement a program to handle Unsupported Operation Exception.

```
handle10.java handle12.java handle13.java handle14.java
1 package exceptionHandling;
2 import java.util.*;
3
4 public class handle14 {
5     public static void main(String[] args) {
6         try {
7             List<String> list = Collections.unmodifiableList(new Ar
8             list.add("New Element"); // This will throw Unsupported
9         } catch (UnsupportedOperationException e) {
10             System.out.println("Exception: Unsupported operation!");
11         }
12     }
13 }
14
15 //15 also same
16
```

Console <
<terminated> handle14 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: Unsupported operation!

15. Write a program to handle Unsupported Operation Exception.

```
handle10.java handle12.java handle13.java handle14.java
1 package exceptionHandling;
2 import java.util.*;
3
4 public class handle14 {
5     public static void main(String[] args) {
6         try {
7             List<String> list = Collections.unmodifiableList(new Ar
8             list.add("New Element"); // This will throw Unsupported
9         } catch (UnsupportedOperationException e) {
10             System.out.println("Exception: Unsupported operation!");
11         }
12     }
13 }
14
15 //15 also same
16
```

Console <
<terminated> handle14 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: Unsupported operation!

16. Implement a program to handle Concurrent Modification Exception.

The screenshot shows the Eclipse IDE interface with the code editor open. The code is as follows:

```
4  public class handle16 {
5      public static void main(String[] args) {
6          try {
7              List<String> list = new ArrayList<>();
8              list.add("Element 1");
9              list.add("Element 2");
10
11             for (String element : list) {
12                 list.remove(element); // This will throw ConcurrentModificationException
13             }
14         } catch (ConcurrentModificationException e) {
15             System.out.println("Exception: Concurrent modification exception occurred!");
16         }
17     }
18 }
```

The code demonstrates attempting to remove elements from a list while it is being modified, which triggers a `ConcurrentModificationException`. The catch block prints a message to the console.

17. Write a program to handle Illegal Argument Exception.

The screenshot shows the Eclipse IDE interface with the code editor open. The code is as follows:

```
1 package exceptionHandling;
2
3 public class handle17 {
4     public static void main(String[] args) {
5         try {
6             throw new IllegalArgumentException("Illegal argument passed!");
7         } catch (IllegalArgumentException e) {
8             System.out.println("Exception: " + e.getMessage());
9         }
10    }
11
12
13 }
```

The code throws an `IllegalArgumentException` with a specific message. The catch block catches this exception and prints its message to the console.

18. Implement a program to handle Security Exception.

The screenshot shows the Eclipse IDE interface with the code editor open. The code is as follows:

```
1 package exceptionHandling;
2
3 public class handle18 {
4     public static void main(String[] args) {
5         try {
6             System.setSecurityManager(new SecurityManager());
7             throw new SecurityException("Security exception occurred!");
8         } catch (SecurityException e) {
9             System.out.println("Exception: Security violation detected!");
10    }
11
12
13 }
```

The code sets a security manager and then throws a `SecurityException`. The catch block catches this exception and prints a message to the console.

19. Write a program to handle Date Time Parse Exception.

```
handle19.java handle17Java handle18Java handle19.java × *95
1 package exceptionHandling;
2+ import java.time.*;
3
4 public class handle19{
5     public static void main(String[] args) {
6         try {
7             String invalidDate = "2025-02-31"; // Invalid date
8             LocalDate.parse(invalidDate); // This will throw DateTimeParseException
9         } catch (DateTimeParseException e) {
10             System.out.println("Exception: Invalid date format!");
11         }
12     }
13 }
14
15
16
17
```

Console ×
<terminated> handle19 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: Invalid date format!

20. Implement a program to handle Pattern Syntax Exception.

```
handle17.java handle18Java handle19Java handle20java × *95
1 package exceptionHandling;
2 import java.util.regex.*;
3
4 public class handle20 {
5     public static void main(String[] args) {
6         try {
7             Pattern.compile("["); // This will throw PatternSyntaxException
8         } catch (PatternSyntaxException e) {
9             System.out.println("Exception: Invalid regular expression!");
10        }
11    }
12 }
13
14
15
```

Console ×
<terminated> handle20 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: Invalid regular expression!

21. Write a program to handle Missing Resource Exception.

```
handle18.java handle19Java handle20Java handle21java × *95
1 package exceptionHandling;
2 import java.util.*;
3
4 public class handle21 {
5     public static void main(String[] args) {
6         try {
7             ResourceBundle rb = ResourceBundle.getBundle("nonexistent");
8         } catch (MissingResourceException e) {
9             System.out.println("Exception: Resource bundle not found!");
10        }
11    }
12 }
13
14
15
```

Console ×
<terminated> handle21 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: Resource bundle not found!

22. Implement a program to handle FormatterClosedException.

```
handle22.java handle20java handle21java handle22java >95
1 package exceptionHandling;
2 import java.util.*;
3
4 public class handle22{
5     public static void main(String[] args) {
6         try (Formatter formatter = new Formatter()) {
7             formatter.format("Formatted String: %d", 100);
8             formatter.close();
9             formatter.format("This will throw exception"); // This
10        } catch (FormatterClosedException e) {
11            System.out.println("Exception: Formatter is closed!");
12        }
13    }
14}
15
16
```

Console >

```
<terminated> handle22 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: Formatter is closed!
```

23. Write a program to handle Buffer Over flow Exception.

```
handle20.java handle21java handle22java handle23.java >95
1 package exceptionHandling;
2 import java.nio.*;
3
4 public class handle23 {
5     public static void main(String[] args) {
6         try {
7             ByteBuffer buffer = ByteBuffer.allocate(2);
8             buffer.put((byte) 1);
9             buffer.put((byte) 2);
10            buffer.put((byte) 3); // This will throw BufferOverflowException
11        } catch (BufferOverflowException e) {
12            System.out.println("Exception: Buffer overflow occurred");
13        }
14    }
15}
16
```

Console >

```
<terminated> handle23 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: Buffer overflow occurred!
```

24. Implement a program to handle Buffer Underflow Exception.

```
handle21java handle22java handle23java handle24.java >95
1 package exceptionHandling;
2 import java.nio.*;
3
4 public class handle24 {
5     public static void main(String[] args) {
6         try {
7             ByteBuffer buffer = ByteBuffer.allocate(2);
8             buffer.put((byte) 1);
9             buffer.flip();
10            buffer.get(); // This will work
11            buffer.get(); // This will throw BufferUnderflowException
12        } catch (BufferUnderflowException e) {
13            System.out.println("Exception: Buffer underflow occurred");
14        }
15    }
16}
```

Console >

```
<terminated> handle24 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Exception: Buffer underflow occurred!
```

1. Write a program to demonstrate Array List by adding, removing, and iterating over elements.

The screenshot shows the Eclipse IDE interface. The top bar has tabs for handle22.java, handle23.java, handle24.java, and collection1... (which is the active tab). Below the tabs is the code editor with collection1.java:

```
1 package collectionprogram;
2 import java.util.ArrayList;
3
4 public class collection1 {
5     public static void main(String[] args) {
6         ArrayList<String> list = new ArrayList<>();
7
8         // Adding elements
9         list.add("Apple");
10        list.add("Banana");
11        list.add("Cherry");
12
13        // Removing element
14        list.remove("Banana");
15
16        // Iterating over elements
17        for (String item : list) {
18            System.out.println(item);
19        }
20    }
21 }
```

Below the code editor is the Console view, which shows the output:

```
<terminated> collection1 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Apple
Cherry
```

1. Implement a program to demonstrate LinkedList by adding, removing, and iterating over elements.

The screenshot shows the Eclipse IDE interface. The top bar has tabs for handle22.java, handle23.java, handle24.java, and collection1... (which is the active tab). Below the tabs is the code editor with collection2.java:

```
5
6     public static void main(String[] args) {
7         LinkedList<String> list = new LinkedList<>();
8
9         // Adding elements
10        list.add("Apple");
11        list.add("Banana");
12        list.add("Cherry");
13
14        // Removing element
15        list.remove("Banana");
16
17        // Iterating over elements
18        for (String item : list) {
19            System.out.println(item);
20        }
21    }
22 }
```

Below the code editor is the Console view, which shows the output:

```
<terminated> collection2 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Apple
Cherry
```

- 2.

3. Write a program to demonstrate HashSet by adding, removing, and iterating over elements.

The screenshot shows the Eclipse IDE interface. The top bar displays several open files: handle24.java, collection1..., collection2..., collection3..., and a file ending in "95". The main editor window contains the following Java code:

```
1 package collectionprogram;
2 import java.util.HashSet;
3
4 public class collection3 {
5     public static void main(String[] args) {
6         HashSet<String> set = new HashSet<>();
7
8         // Adding elements
9         set.add("Apple");
10        set.add("Banana");
11        set.add("Cherry");
12
13        // Removing element
14        set.remove("Banana");
15
16        // Iterating over elements
17        for (String item : set) {
18            System.out.println(item);
19        }
20    }
21
22 }
```

The bottom window is the Console, showing the output of the program:

```
<terminated> collection3 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Apple
Cherry
```

- Implement a program to demonstrate
4. TreeSet by adding, removing, and
iterating over elements.

The screenshot shows the Eclipse IDE interface. In the top tab bar, the 'collection4...' tab is selected. Below it, the code editor displays a Java class named 'collection4' which demonstrates the use of a TreeSet. The code adds three elements ('Apple', 'Banana', 'Cherry') to the set, removes one ('Banana'), and then iterates over the remaining elements, printing them to the console. The execution output window at the bottom shows the printed results: 'Apple' and 'Cherry'.

```
1 package collectionprogram;
2
3 import java.util.TreeSet;
4
5 public class collection4 {
6     public static void main(String[] args) {
7         TreeSet<String> set = new TreeSet<>();
8
9         // Adding elements
10        set.add("Apple");
11        set.add("Banana");
12        set.add("Cherry");
13
14        // Removing element
15        set.remove("Banana");
16
17        // Iterating over elements
18        for (String item : set) {
19            System.out.println(item);
20        }
21    }
22 }
```

Console <terminated> collection4 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12

Apple
Cherry

5. Write a program to demonstrate
HashMap by adding and retrieving
key-value pairs.

Implement a program to demonstrate

The screenshot shows the Eclipse IDE interface. The top part displays the code for `collection5.java`. The code creates a `HashMap` and adds three key-value pairs: (1, "Apple"), (2, "Banana"), and (3, "Cherry"). It then prints the value associated with key 2. The bottom part shows the `Console` tab with the output: `<terminated> collection5 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\collection5.java:14: Key 2: Banana`.

```
1 package collectionprogram;
2 import java.util.HashMap;
3
4 public class collection5 {
5     public static void main(String[] args) {
6         HashMap<Integer, String> map = new HashMap<>();
7
8         // Adding key-value pairs
9         map.put(1, "Apple");
10        map.put(2, "Banana");
11        map.put(3, "Cherry");
12
13        // Retrieving value by key
14        System.out.println("Key 2: " + map.get(2));
15    }
16
17
18 }
```

6. TreeMap by adding and retrieving key- value pairs.

The screenshot shows the Eclipse IDE interface. The top part displays the code for `collection6.java`. The code creates a `TreeMap` and adds three key-value pairs: (1, "Apple"), (2, "Banana"), and (3, "Cherry"). It then prints the value associated with key 2. The bottom part shows the `Console` tab with the output: `<terminated> collection6 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\collection6.java:14: Key 2: Banana`.

```
1 package collectionprogram;
2 import java.util.TreeMap;
3
4 public class collection6 {
5     public static void main(String[] args) {
6         TreeMap<Integer, String> map = new TreeMap<>();
7
8         // Adding key-value pairs
9         map.put(1, "Apple");
10        map.put(2, "Banana");
11        map.put(3, "Cherry");
12
13        // Retrieving value by key
14        System.out.println("Key 2: " + map.get(2));
15    }
16
17 }
```

7. Write a program to demonstrate
LinkedHashMap by adding and
retrieving key- value pairs.

Implement a program to demonstrate

The screenshot shows the Eclipse IDE interface. The top bar has tabs for "collection4...", "collection5.java", "collection6...", "collection7...", and "collection8...". The "collection7..." tab is active. The code in the editor is:

```
1 package collectionprogram;
2 import java.util.LinkedHashMap;
3
4 public class collection7 {
5     public static void main(String[] args) {
6         LinkedHashMap<Integer, String> map = new LinkedHashMap<>();
7
8         // Adding key-value pairs
9         map.put(1, "Apple");
10        map.put(2, "Banana");
11        map.put(3, "Cherry");
12
13        // Retrieving value by key
14        System.out.println("Key 2: " + map.get(2));
15    }
16}
17
18
```

The bottom part of the interface shows the "Console" tab with the output:

```
<terminated> collection7 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\
Key 2: Banana
```

8. Queue by adding, removing, and iterating over elements.

The screenshot shows the Eclipse IDE interface. The top bar has tabs for "collection5.java", "collection6...", "collection7...", and "collection8...". The "collection8..." tab is active. The code in the editor is:

```
1 package collectionprogram;
2 import java.util.LinkedList;
3
4 public class collection8 {
5     public static void main(String[] args) {
6         Queue<String> queue = new LinkedList<>();
7
8         // Adding elements
9         queue.add("Apple");
10        queue.add("Banana");
11        queue.add("Cherry");
12
13        // Removing element
14        System.out.println("Removed: " + queue.remove());
15
16        // Iterating over elements
17        for (String item : queue) {
18            System.out.println(item);
19        }
20    }
21}
22
```

The bottom part of the interface shows the "Console" tab with the output:

```
<terminated> collection8 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\
Removed: Apple
Banana
Cherry
```

- Implement a program to demonstrate
9. Write a program to demonstrate
PriorityQueue by adding,
removing, and iterating over
elements.

The screenshot shows the Eclipse IDE interface. In the top bar, there are tabs for "collection6...", "collection7...", "collection8...", "collection9...", and "collection9...". The "collection9..." tab is active. Below the tabs, the code editor displays the following Java code:

```
1 package collectionprogram;
2 import java.util.PriorityQueue;
3
4 public class collection9 {
5     public static void main(String[] args) {
6         PriorityQueue<String> queue = new PriorityQueue<>();
7
8         // Adding elements
9         queue.add("Apple");
10        queue.add("Banana");
11        queue.add("Cherry");
12
13        // Removing element
14        System.out.println("Removed: " + queue.remove());
15
16        // Iterating over elements
17        for (String item : queue) {
18            System.out.println(item);
19        }
20    }
21 }
22 }
```

A tooltip is visible in the center-right area of the code editor, showing the type information for the variable "item": "@ String item - collectionprogram.collection9.main String[]".

Below the code editor is the "Console" view, which shows the terminal output of the application's execution:

```
<terminated> collection9 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\collection9\src\collectionprogram\collection9.java
Removed: Apple
Banana
Cherry
```

- Implement a program to demonstrate
10. Stack by adding, removing, and
iterating over elements.

The screenshot shows the Eclipse IDE interface. In the top tab bar, there are tabs for "collection7...", "collection8...", "collection9...", "collection10...", and "collection11...". The "collection10..." tab is active. Below the tabs, the code for "collection10" is displayed:

```
1 package collectionprogram;
2 import java.util.Stack;
3
4 public class collection10 {
5     public static void main(String[] args) {
6         Stack<String> stack = new Stack<>();
7
8         // Pushing elements onto the stack
9         stack.push("Apple");
10        stack.push("Banana");
11        stack.push("Cherry");
12
13        // Popping an element from the stack
14        System.out.println("Popped: " + stack.pop());
15
16        // Iterating over elements
17        for (String item : stack) {
18            System.out.println(item);
19        }
20    }
21
22 }
```

Below the code editor is the "Console" view, which shows the output of the program:

```
<terminated> collection10 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Popped: Cherry
Apple
Banana
```

11. Write a program to demonstrate
ArrayDeque by adding, removing,
and iterating over elements

The screenshot shows the Eclipse IDE interface. In the top tab bar, there are tabs for "collection8...", "collection9...", "collection10...", "collection11...", and "collection12...". The "collection11..." tab is active. Below the tabs, the code for "collection11" is displayed:

```
1 package collectionprogram;
2 import java.util.ArrayDeque;
3
4 public class collection11 {
5     public static void main(String[] args) {
6         ArrayDeque<String> deque = new ArrayDeque<>();
7
8         // Adding elements
9         deque.add("Apple");
10        deque.add("Banana");
11        deque.add("Cherry");
12
13        // Removing elements
14        System.out.println("Removed: " + deque.remove());
15
16        // Iterating over elements
17        for (String item : deque) {
18            System.out.println(item);
19        }
20    }
21
22 }
```

Below the code editor is the "Console" view, which shows the output of the program:

```
<terminated> collection11 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Removed: Apple
Banana
Cherry
```

- Implement a program to demonstrate
12. EnumSet by adding, removing, and
iterating over elements

The screenshot shows the Eclipse IDE interface. The top part displays a Java code editor with the following code:

```
1 package collectionprogram;
2 import java.util.EnumSet;
3
4 enum Fruits { APPLE, BANANA, CHERRY }
5
6 public class collection12 {
7     public static void main(String[] args) {
8         EnumSet<Fruits> set = EnumSet.of(Fruits.APPLE, Fruits.BANANA);
9
10        // Adding element
11        set.add(Fruits.CHERRY);
12
13        // Removing element
14        set.remove(Fruits.BANANA);
15
16        // Iterating over elements
17        for (Fruits fruit : set) {
18            System.out.println(fruit);
19        }
20    }
21 }
22 }
```

The bottom part shows the Eclipse Console window with the output:

```
<terminated> collection12 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
APPLE
CHERRY
```

13. Write a program to demonstrate
BitSet by adding, removing, and
iterating over elements.

The screenshot shows the Eclipse IDE interface. The top part displays a Java code editor with the following code:

```
1 package collectionprogram;
2 import java.util.BitSet;
3
4 public class collection13 {
5     public static void main(String[] args) {
6         BitSet bitSet = new BitSet();
7
8         // Setting bits
9         bitSet.set(0);
10        bitSet.set(2);
11
12        // Clearing bit
13        bitSet.clear(2);
14
15        // Iterating over bits
16        for (int i = 0; i < bitSet.length(); i++) {
17            if (bitSet.get(i)) {
18                System.out.println("Bit " + i + " is set.");
19            }
20        }
21    }
22 }
```

The bottom part shows the Eclipse Console window with the output:

```
<terminated> collection13 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Bit 0 is set.
```

14. Hashtable by adding and retrieving
key-value pairs

Implement a program to demonstrate

The screenshot shows the Eclipse IDE interface. The top part displays a Java code editor with the following code:

```
1 package collectionprogram;
2 import java.util.Hashtable;
3
4 public class collection14 {
5     public static void main(String[] args) {
6         Hashtable<Integer, String> table = new Hashtable<>();
7
8         // Adding key-value pairs
9         table.put(1, "Apple");
10        table.put(2, "Banana");
11        table.put(3, "Cherry");
12
13        // Retrieving value by key
14        System.out.println("Key 2: " + table.get(2));
15    }
16
17
18}
```

The bottom part shows the Eclipse Console window with the output of the program:

```
<terminated> collection14 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Key 2: Banana
```

15.

Write a program to demonstrate Properties by adding and retrieving key-value pairs.

The screenshot shows the Eclipse IDE interface. The top part displays a Java code editor with the following code:

```
1 package collectionprogram;
2 import java.util.Properties;
3
4 public class collection15 {
5     public static void main(String[] args) {
6         Properties properties = new Properties();
7
8         // Adding key-value pairs
9         properties.setProperty("username", "admin");
10        properties.setProperty("password", "password123");
11
12        // Retrieving value by key
13        System.out.println("Username: " + properties.getProperty("u
14    }
15
16
17
18}
```

The bottom part shows the Eclipse Console window with the output of the program:

```
<terminated> collection15 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Username: admin
```

16.

Vector by adding, removing, and iterating over elements.

Implement a program to demonstrate

```
1 package collectionprogram;
2 import java.util.Vector;
3
4 public class collection16 {
5     public static void main(String[] args) {
6         Vector<String> vector = new Vector<>();
7
8         // Adding elements
9         vector.add("Apple");
10        vector.add("Banana");
11        vector.add("Cherry");
12
13        // Removing element
14        vector.remove("Banana");
15
16        // Iterating over elements
17        for (String item : vector) {
18            System.out.println(item);
19        }
20    }
21 }
22
```

Console <terminated> collection16 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Apple
Cherry

17.

Write a program to demonstrate
Enumeration by iterating over
elements of a collection.

```
1 package collectionprogram;
2 import java.util.Vector;
3
4 public class collection17 {
5     public static void main(String[] args) {
6         Vector<String> vector = new Vector<>();
7
8         // Adding elements
9         vector.add("Apple");
10        vector.add("Banana");
11        vector.add("Cherry");
12
13        // Iterating using Enumeration
14        Enumeration<String> enumeration = vector.elements();
15        while (enumeration.hasMoreElements()) {
16            System.out.println(enumeration.nextElement());
17        }
18    }
19 }
20
21
22
23
```

Console <terminated> collection17 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Apple
Banana
Cherry

18.

Implement a program to demonstrate ListIterator by iterating over elements of a list.

The screenshot shows the Eclipse IDE interface with two tabs open: 'collection18' and 'Console'. The code in 'collection18' demonstrates the use of ListIterator to iterate over elements of an ArrayList. The 'Console' tab shows the output: Apple, Banana, Cherry.

```
1 package collectionprogram;
2 import java.util.ArrayList;
3
4 public class collection18 {
5     public static void main(String[] args) {
6         ArrayList<String> list = new ArrayList<>();
7
8         // Adding elements
9         list.add("Apple");
10        list.add("Banana");
11        list.add("Cherry");
12
13        // Iterating using ListIterator
14        ListIterator<String> iterator = list.listIterator();
15        while (iterator.hasNext()) {
16            System.out.println(iterator.next());
17        }
18    }
19
20}
21
22
23
```

```
Console <terminated> collection17 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Apple
Banana
Cherry
```

19.

Write a program to demonstrate Iterator by iterating over elements of a collection.

The screenshot shows the Eclipse IDE interface with two tabs open: 'collection19' and 'Console'. The code in 'collection19' demonstrates the use of Iterator to iterate over elements of an ArrayList. The 'Console' tab shows the output: Apple, Banana, Cherry.

```
1 package collectionprogram;
2 import java.util.ArrayList;
3
4 public class collection19 {
5     public static void main(String[] args) {
6         ArrayList<String> list = new ArrayList<>();
7
8         // Adding elements
9         list.add("Apple");
10        list.add("Banana");
11        list.add("Cherry");
12
13        // Iterating using Iterator
14        Iterator<String> iterator = list.iterator();
15        while (iterator.hasNext()) {
16            System.out.println(iterator.next());
17        }
18    }
19
20}
21
22
```

```
Console <terminated> collection19 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Apple
Banana
Cherry
```

20.

ArrayBlockingQueue by adding, removing, and Q iterating over elements.

Implement a program to demonstrate

The screenshot shows the Eclipse IDE interface. The top part displays the code for `collection20` which uses `ArrayBlockingQueue`. The bottom part shows the `Console` tab with the output: `Removed: Apple` and `Banana`.

```
1 package collectionprogram;
2 import java.util.concurrent.ArrayBlockingQueue;
3
4 public class collection20 {
5     public static void main(String[] args) throws InterruptedException {
6         ArrayBlockingQueue<String> queue = new ArrayBlockingQueue<>
7
8             // Adding elements
9             queue.put("Apple");
10            queue.put("Banana");
11
12            // Removing elements
13            System.out.println("Removed: " + queue.take());
14
15            // Iterating over elements
16            for (String item : queue) {
17                System.out.println(item);
18            }
19        }
20    }
21
22 }
```

Console

```
<terminated> collection20 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Removed: Apple
Banana
```

21.

Write a program to demonstrate
LinkedBlockingQueue by adding,
removing, and iterating over
elements.

The screenshot shows the Eclipse IDE interface. The top part displays the code for `collection21` which uses `LinkedBlockingQueue`. The bottom part shows the `Console` tab with the output: `Removed: Apple` and `Banana`.

```
1 package collectionprogram;
2 import java.util.concurrent.LinkedBlockingQueue;
3
4 public class collection21 {
5     public static void main(String[] args) throws InterruptedException {
6         LinkedBlockingQueue<String> queue = new LinkedBlockingQueue<>
7
8             // Adding elements
9             queue.put("Apple");
10            queue.put("Banana");
11
12            // Removing elements
13            System.out.println("Removed: " + queue.take());
14
15            // Iterating over elements
16            for (String item : queue) {
17                System.out.println(item);
18            }
19        }
20    }
21
22 }
```

Console

```
<terminated> collection21 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Removed: Apple
Banana
```

Implement a program to demonstrate

22.

Priority BlockingQueue by adding, removing, and iterating over elements.

The screenshot shows the Eclipse IDE interface. The top part displays a Java code editor with the following code:

```
1 package collectionprogram;
2 import java.util.concurrent.PriorityBlockingQueue;
3
4 public class collection22 {
5     public static void main(String[] args) throws InterruptedException {
6         PriorityBlockingQueue<String> queue = new PriorityBlockingQueue<String>();
7
8         // Adding elements
9         queue.put("Apple");
10        queue.put("Banana");
11
12        // Removing elements
13        System.out.println("Removed: " + queue.take());
14
15        // Iterating over elements
16        for (String item : queue) {
17            System.out.println(item);
18        }
19    }
20 }
21
22 }
```

The bottom part shows the Eclipse Console window with the output of the program:

```
<terminated> collection22 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Removed: Apple
Banana
```

23.

Write a program to demonstrate Synchronous Queue by adding and removing elements.

The screenshot shows the Eclipse IDE interface. The top part displays a Java code editor with the following code:

```
1 package collectionprogram;
2 import java.util.concurrent.SynchronousQueue;
3
4 public class collection23 {
5     public static void main(String[] args) throws InterruptedException {
6         SynchronousQueue<String> queue = new SynchronousQueue<String>();
7
8         // Adding and removing elements
9         new Thread(() -> {
10             try {
11                 queue.put("Apple");
12             } catch (InterruptedException e) {
13                 Thread.currentThread().interrupt();
14             }
15         }).start();
16
17         // Removing element
18         System.out.println("Removed: " + queue.take());
19     }
20 }
21
22 }
```

The bottom part shows the Eclipse Console window with the output of the program:

```
<terminated> collection23 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
Removed: Apple
```

Implement a program to demonstrate

24.

DelayQueue by adding and retrieving elements.

The screenshot shows the Eclipse IDE interface. In the top editor area, there is a Java code snippet for a class named `DelayedItem` that implements the `Delayed` interface. The code includes methods for setting item and delay time, and overriding `getDelay` and `compareTo`. In the bottom console window, the output shows the word "Banana" being printed.

```
1 package collectionprogram;
2 import java.util.concurrent.DelayQueue;
3
4 class DelayedItem implements Delayed {
5     private String item;
6     private long delayTime;
7
8     public DelayedItem(String item, long delayTime) {
9         this.item = item;
10        this.delayTime = delayTime + System.currentTimeMillis();
11    }
12
13    @Override
14    public long getDelay(TimeUnit unit) {
15        return unit.convert(delayTime - System.currentTimeMillis(), unit);
16    }
17
18    @Override
19    public int compareTo(Delayed o) {
20        return Long.compare(this.delayTime, ((DelayedItem) o).delayTime);
21    }
22}
23
24
```

Console ×
collection24 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\eclipse\plu
Taken: Banana

25.

Write a program to demonstrate ConcurrentLinkedQueue by adding, removing, and iterating over elements.

The screenshot shows the Eclipse IDE interface. In the top editor area, there is a Java code snippet for a class named `collection25`. It contains a static main method that creates a `ConcurrentLinkedQueue`, adds "Apple" and "Banana" to it, prints the removed item, and then iterates over the queue to print each item. In the bottom console window, the output shows the removal of "Apple" followed by the iteration output of "Banana".

```
1 package collectionprogram;
2 import java.util.concurrent.ConcurrentLinkedQueue;
3
4 public class collection25 {
5     public static void main(String[] args) {
6         ConcurrentLinkedQueue<String> queue = new ConcurrentLinkedQueue();
7         queue.add("Apple");
8         queue.add("Banana");
9
10        System.out.println("Removed: " + queue.poll());
11        for (String item : queue) {
12            System.out.println(item);
13        }
14    }
15}
16
```

Console ×
terminated > collection25 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\eclipse\plu
Removed: Apple
Banana

Multi Threading:

Implement a program to demonstrate

1. Write a Java program to create multiple threads and display their names.

```
1 package multithreading;
2
3 public class multil extends Thread {
4     public void run() {
5         System.out.println("Thread Name: " + Thread.currentThread());
6     }
7
8     public static void main(String[] args) {
9         multil t1 = new multil();
10        multil t2 = new multil();
11        t1.start();
12        t2.start();
13    }
14
15
16
17
18 }
```

Console >

```
<terminated> multi1 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\eclip
Thread Name: Thread-0
Thread Name: Thread-1
```

2. Implement a program to demonstrate thread synchronization using synchronized blocks.

```
1 package multithreading;
2
3     class Counter {
4         private int count = 0;
5
6         public void increment() {
7             synchronized (this) {
8                 count++;
9             }
10        }
11
12        public int getCount() {
13            return count;
14        }
15    }
16
17    public class SynchronizedBlockmulti2 {
18        public static void main(String[] args) throws InterruptedException {
19            Counter counter = new Counter();
20            Thread t1 = new Thread(() -> { for (int i = 0; i < 1000; i++)
21                counter.increment(); });
22            Thread t2 = new Thread(() -> { for (int i = 0; i < 1000; i+
23                counter.increment()); });
24            t1.start();
25            t2.start();
26            t1.join();
27            t2.join();
28        }
29    }
```

Console >

```
<terminated> SynchronizedBlockmulti2 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-
Final Count: 2000
```

3. Write a Java program to create multiple threads and display their priorities.

The screenshot shows the Eclipse IDE interface. The top part displays a Java code editor with the file `multi3.java` open. The code creates two threads, `t1` and `t2`, setting their priorities to `MIN_PRIORITY` and `MAX_PRIORITY` respectively before starting them. The bottom part shows the `Console` tab with the output:

```
<terminated> multi3 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Thread-1 Priority: 10
Thread-0 Priority: 1
```

4. Implement a program to create a thread pool and execute multiple tasks using Executor Service.

The screenshot shows the Eclipse IDE interface. The top part displays a Java code editor with the file `multi4.java` open. It defines a task class `Task` that implements `Runnable`. The `main` method creates a fixed thread pool of size 3 using `Executors.newFixedThreadPool(3)`. It then submits 5 tasks to the pool. A tooltip is visible over the `submit` method call. The bottom part shows the `Console` tab with the output:

```
<terminated> multi4 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ec
Task executed by: pool-1-thread-3
Task executed by: pool-1-thread-2
Task executed by: pool-1-thread-1
Task executed by: pool-1-thread-1
Task executed by: pool-1-thread-3
```

5. Write a Java program to create multiple threads and join them.

```
① Synchronized... ② multi3.java ③ multi4.java ④ multi5.java × »95
1 package multithreading;
2
3 public class multi5 extends Thread {
4
5     public void run() {
6         System.out.println("Thread " + Thread.currentThread().getNa
7     }
8
9     public static void main(String[] args) throws InterruptedException
10    {
11        multi5 t1 = new multi5 ();
12        multi5 t2 = new multi5 ();
13        t1.start();
14        t2.start();
15        t1.join();
16        t2.join();
17        System.out.println("Completed");
18    }
19
20
21 }
```

```
Console <terminated> multi5 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ecli  
Thread Thread-1 is executing  
Thread Thread-0 is executing  
All threads completed
```

6. Implement a program to demonstrate deadlock condition in multithreading.

```
multi3.java multi4java multi5java multi6java >95
1 package multithreading;
2 class A {
3     synchronized void methodA(B b) {
4         System.out.println("Thread 1: Holding lock on A");
5         try { Thread.sleep(100); } catch (Exception e) {}
6         System.out.println("Thread 1: Waiting for lock on B");
7         b.last();
8     }
9
10    synchronized void last() { System.out.println("Inside A's last() me
11    })
12
13 class B {
14     synchronized void methodB(A a) {
15         System.out.println("Thread 2: Holding lock on B");
16         try { Thread.sleep(100); } catch (Exception e) {}
17         System.out.println("Thread 2: Waiting for lock on A");
18         a.last();
19     }
20
21     synchronized void last() { System.out.println("Inside B's last() me
22    })
23
24 public class multi6{
25     public static void main(String[] args) {
Console >
multi6 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\eclipse\plugins\or
Thread 2: Holding lock on B
Thread 1: Holding lock on A
Thread 2: Waiting for lock on A
Thread 1: Waiting for lock on B
```

7. Write a Java program to create multiple threads and interrupt them

The screenshot shows the Eclipse IDE interface with the code editor and a terminal window. The code in the editor is as follows:

```
multi7.java
1 package multithreading;
2
3     class multi7 extends Thread {
4         public void run() {
5             try {
6                 System.out.println("Thread " + Thread.currentThread().getId());
7                 Thread.sleep(5000);
8                 System.out.println("Thread " + Thread.currentThread().getId());
9             } catch (InterruptedException e) {
10                 System.out.println("Thread " + Thread.currentThread().getId());
11             }
12         }
13
14     public static void main(String[] args) throws InterruptedException {
15         multi7 t1 = new multi7();
16         multi7 t2 = new multi7();
17         t1.start();
18         t2.start();
19         Thread.sleep(2000);
20         t1.interrupt();
21         t2.interrupt();
22     }
23 }
```

The terminal window below shows the output of running the program:

```
Console <terminated> multi7 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\eclip
Thread Thread-1 is sleeping.
Thread Thread-0 is sleeping.
Thread Thread-1 was interrupted.
Thread Thread-0 was interrupted.
```

8. Implement a program to demonstrate thread local variables.

The screenshot shows the Eclipse IDE interface with the code editor and a terminal window. The code in the editor is as follows:

```
multi8.java
1 package multithreading;
2
3
4     class mutli8 extends Thread {
5         private static ThreadLocal<Integer> threadLocal = ThreadLocal.w
6
7         public void run() {
8             threadLocal.set(threadLocal.get() + 1);
9             System.out.println("Thread " + Thread.currentThread().getNa
10         }
11
12     public static void main(String[] args) {
13         mutli8 t1 = new mutli8();
14         mutli8 t2 = new mutli8();
15         t1.start();
16         t2.start();
17     }
18
19
20
21 }
```

The terminal window below shows the output of running the program:

```
Console <terminated> mutli8 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\eclip
Thread Thread-1 value: 1
Thread Thread-0 value: 1
```

9. Write a Java program to create multiple threads and wait for them to complete using CountDownLatch.

```
multi5java multi6java multi7java mutli8java multi9java *94
1 package multithreading;
2 import java.util.concurrent.CountDownLatch;
3
4 class TaskWithLatch implements Runnable {
5     private CountDownLatch latch;
6
7     TaskWithLatch(CountDownLatch latch) {
8         this.latch = latch;
9     }
10
11    public void run() {
12        System.out.println(Thread.currentThread().getName() + " is exec
13        try { Thread.sleep(2000); } catch (InterruptedException e) {}}
14        latch.countDown();
15    }
16 }
17
18 public class multi9 {
19     public static void main(String[] args) throws InterruptedException
20         CountDownLatch latch = new CountDownLatch(2);
21
22         Thread t1 = new Thread(new TaskWithLatch(latch));
23         Thread t2 = new Thread(new TaskWithLatch(latch));
24
25         t1.start();
```

10. Implement a program to demonstrate thread priorities in Java

```
multi6.java multi7.java multi8.java multi9.java multi10.java <terminated> 94
1 package multithreading;
2
3
4     class multi10 extends Thread {
5         public void run() {
6             System.out.println(Thread.currentThread().getName() + " with Priority: " + Thread.currentThread().getPriority());
7         }
8
9     public static void main(String[] args) {
10        multi10 t1 = new multi10();
11        multi10 t2 = new multi10();
12        t1.setPriority(Thread.MIN_PRIORITY);
13        t2.setPriority(Thread.MAX_PRIORITY);
14        t1.start();
15        t2.start();
16    }
17 }
18
19
20
```

11. Write a Java program to create multiple threads and use thread group.

```
multi7java multi8java multi9java multi10java multi11java >94
1 package multithreading;
2
3     class multi11 extends Thread {
4         public void run() {
5             System.out.println(Thread.currentThread().getName() + " in
6         }
7
8         public static void main(String[] args) {
9             ThreadGroup group = new ThreadGroup("Group1");
10
11             multi11 t1 = new multi11();
12             multi11 t2 = new multi11();
13
14             Thread t1Thread = new Thread(group, t1);
15             Thread t2Thread = new Thread(group, t2);
16
17             t1Thread.start();
18             t2Thread.start();
19         }
20
21
22
23
24 }
```

Console >
<terminated> multi11 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\cli
Thread-2 in group Group1
Thread-3 in group Group1

12. Implement a program to demonstrate thread communication using wait() and notify() methods.

```
multi9java multi10java multi11.java multi12.java >95
1 package multithreading;
2
3     class multi12 {
4         private static final Object Lock = new Object();
5
6         public static void main(String[] args) throws InterruptedException {
7             Thread producer = new Thread(() -> {
8                 synchronized (Lock) {
9                     try {
10                         System.out.println("Producer is waiting");
11                         Lock.wait();
12                     } catch (InterruptedException e) { e.printStackTrace();
13                         System.out.println("Producer resumed");
14                     }
15                 });
16
17                 Thread consumer = new Thread(() -> {
18                     synchronized (Lock) {
19                         System.out.println("Consumer is notifying");
20                         Lock.notify();
21                     }
22                 });
23
24                 producer.start();
25                 consumer.start();
26             });
27
28 }
```

Console >
<terminated> multi12 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\cli
Producer is waiting
Consumer is notifying
Producer resumed

13. Write a Java program to create multiple threads and use thread local variables.

The screenshot shows the Eclipse IDE interface. The top part displays the code for `multi13.java`. The code creates two threads, each incrementing a shared counter stored in a `ThreadLocal` variable. The output in the console shows the final values of the counter for each thread.

```
multi10.java multi11.java multi12.java multi13.java × "95
1 package multithreading;
2
3 class multi13 extends Thread {
4     private static ThreadLocal<Integer> threadLocal = ThreadLocal.w
5
6     public void run() {
7         threadLocal.set(threadLocal.get() + 1);
8         System.out.println("Thread " + Thread.currentThread().getNa
9     }
10
11     public static void main(String[] args) {
12         multi13 t1 = new multi13();
13         multi13 t2 = new multi13();
14         t1.start();
15         t2.start();
16     }
17
18
19
20
```

Console output:

```
<terminated> multi13 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\el
Thread Thread-0 value: 1
Thread Thread-1 value: 1
```

14. Implement a program to demonstrate thread communication using volatile keyword

The screenshot shows the Eclipse IDE interface. The top part displays the code for `multi14.java`. It uses a `volatile` boolean flag to coordinate between two threads. One thread sets the flag to true, and the other thread waits for it to be set before exiting. The output in the console shows the sequence of events.

```
multi11.java multi12.java multi13.java multi14.java × "95
1 package multithreading;
2
3 public class multi14 {
4     private static volatile boolean flag = false;
5
6     public static void main(String[] args) throws InterruptedException {
7         Thread t1 = new Thread(() -> {
8             while (!flag) {
9                 // Do something
10            }
11            System.out.println("Flag is set, thread 1 terminated.");
12        });
13
14        Thread t2 = new Thread(() -> {
15            System.out.println("Setting flag to true");
16            flag = true;
17        });
18
19        t1.start();
20        t2.start();
21
22        t1.join();
23        t2.join();
24    }
25
```

Console output:

```
<terminated> multi14 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\el
Setting flag to true
Flag is set, thread 1 terminated.
```

15. Write a Java program to create multiple threads and use Executors framework.

```
multi12.java × multi13.java × multi14.java × *multi15.java × "95
practiseprograms/src/multithreading/multi12.java
 2
 3*   import java.util.concurrent.ExecutorService;□
 5
 6   class multi15 implements Runnable {
 7*     public void run() {
 8       System.out.println(Thread.currentThread().getName() + " is
 9     }
10
11*   public static void main(String[] args) {
12     ExecutorService executor = Executors.newFixedThreadPool(3)
13     for (int i = 0; i < 5; i++) {
14       executor.submit(new multi15 ());
15     }
16     executor.shutdown();
17   }
18
19
20
21
```

```
Console ×
<terminated> multi15 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64
pool-1-thread-3 is executing
pool-1-thread-1 is executing
pool-1-thread-2 is executing
pool-1-thread-1 is executing
pool-1-thread-3 is executing
```

16. Implement a program to demonstrate thread interruption in Java.

```
multi12.java multi13.java multi14.java multi15.java >95
1 package multithreading;
2
3* import java.util.concurrent.ExecutorService;
4
5
6 class multi15 implements Runnable {
7     public void run() {
8         System.out.println(Thread.currentThread().getName() + " is
9     }
10
11    public static void main(String[] args) {
12        ExecutorService executor = Executors.newFixedThreadPool(3);
13        for (int i = 0; i < 5; i++) {
14            executor.submit(new multi15 ());
15        }
16        executor.shutdown();
17    }
18}
19
20
21
```

```
Console X:
multi16 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\eclipse\plugins
Thread Thread-0 is sleeping.
Thread Thread-1 is sleeping.
```

17. Write a Java program to create multiple threads and use Callable and Future.

```
multi13.java multi14.java multi15.java multi17.java >95
1 package multithreading;
2     import java.util.concurrent.*;
3
4
5 class multi17 implements Callable<String> {
6     public String call() {
7         return "Thread " + Thread.currentThread().getName() + " is
8     }
9
10    public static void main(String[] args) throws ExecutionException {
11        ExecutorService executor = Executors.newFixedThreadPool(3);
12        multi17 task = new multi17();
13        Future<String> future = executor.submit(task);
14        System.out.println(future.get());
15        executor.shutdown();
16    }
17
18
19
```

```
Console X:
<terminated> multi17 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\ecli
Thread pool-1-thread-1 is executing.
```

18. Implement a program to demonstrate thread communication using BlockingQueue.

```

1 multi14.java   2 multi15.java   3 multi17.java   4 multi18.java  *  5
1 package multithreading;
2 import java.util.concurrent.*;
3
4 class multi18 {
5     public static void main(String[] args) throws InterruptedException {
6         BlockingQueue<Integer> queue = new LinkedBlockingQueue<>(10);
7
8         Thread producer = new Thread(() -> {
9             try {
10                 for (int i = 0; i < 5; i++) {
11                     queue.put(i);
12                     System.out.println("Produced: " + i);
13                 }
14             } catch (InterruptedException e) {
15                 e.printStackTrace();
16             }
17         });
18
19         Thread consumer = new Thread(() -> {
20             try {
21                 for (int i = 0; i < 5; i++) {
22                     Integer value = queue.take();
23                     System.out.println("Consumed: " + value);
24                 }
25             } catch (InterruptedException e) {
26                 e.printStackTrace();
27             }
28         });
29     }
30 }

```

Console output:

```

<terminated> multi18 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\edi
Consumed: 0
Produced: 0
Produced: 1
Consumed: 1
Consumed: 2
Produced: 2
Produced: 3
Produced: 4
Consumed: 3
Consumed: 4

```

19. Write a Java program to create multiple threads and use Phaser

```

1 multi15.java   2 multi17.java   3 multi18.java   4 multi19.java  *  5
1 package multithreading;
2 import java.util.concurrent.*;
3
4 class multi19 {
5     public static void main(String[] args) {
6         Phaser phaser = new Phaser(1);
7
8         Runnable task = () -> {
9             System.out.println(Thread.currentThread().getName() + " is doing the work.");
10            phaser.arriveAndAwaitAdvance();
11        };
12
13         Thread t1 = new Thread(task);
14         Thread t2 = new Thread(task);
15
16         t1.start();
17         t2.start();
18
19         phaser.arriveAndAwaitAdvance();
20         System.out.println("All threads completed.");
21     }
22 }

```

Console output:

```

<terminated> multi19 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\edi
All threads completed.
Thread-1 is doing the work.
Thread-0 is doing the work.

```

20. Implement a program to demonstrate thread communication using CyclicBarrier

```
multi17.java multi18.java multi19.java multi20.java X "95
1 package multithreading;
2
3 import java.util.concurrent.*;
4
5 class multi20 {
6     public static void main(String[] args) throws InterruptedException {
7         CyclicBarrier barrier = new CyclicBarrier(2, () -> System.out.println("All threads reached the barrier"));
8
9         Runnable task = () -> {
10             System.out.println(Thread.currentThread().getName() + " is waiting at the barrier");
11             try {
12                 barrier.await();
13             } catch (InterruptedException | BrokenBarrierException e) {
14                 e.printStackTrace();
15             }
16         };
17
18         Thread t1 = new Thread(task);
19         Thread t2 = new Thread(task);
20
21         t1.start();
22         t2.start();
23         t1.join();
24         t2.join();
25     }
}
```

```
Console X
<terminated> multi20 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86
Thread-0 is waiting at the barrier
Thread-1 is waiting at the barrier
All threads reached the barrier
```

21. Write a Java program to create multiple threads and use Semaphore

```

multi18.java multi19.java multi20.java multi21.java *95
1 package multithreading;
2
3     import java.util.concurrent.*;
4
5     class multi21 {
6         private static Semaphore semaphore = new Semaphore(1);
7
8         public static void main(String[] args) throws InterruptedException {
9             Runnable task = () -> {
10                 try {
11                     semaphore.acquire();
12                     System.out.println(Thread.currentThread().getName());
13                     Thread.sleep(2000);
14                     System.out.println(Thread.currentThread().getName());
15                     semaphore.release();
16                 } catch (InterruptedException e) {
17                     e.printStackTrace();
18                 }
19             };
20
21             Thread t1 = new Thread(task);
22             Thread t2 = new Thread(task);
23
24             t1.start();
25             t2.start();
26         }
27     }

```

Console

```

multi21 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\eclipse\plugins\
Thread-0 has acquired the semaphore.
Thread-0 releasing the semaphore.
Thread-1 has acquired the semaphore.

```

22. Implement a program to demonstrate thread communication using Exchanger.

```

multi19.java multi20.java multi21.java multi22.java *96
1 F practiseprograms/src/multithreading/multi19.java
2     import java.util.concurrent.*;
3
4     class multi22 {
5         public static void main(String[] args) throws InterruptedException {
6             Exchanger<String> exchanger = new Exchanger<>();
7
8             Runnable producer = () -> {
9                 try {
10                     String message = "Hello from " + Thread.currentThread().getName();
11                     System.out.println(Thread.currentThread().getName());
12                     exchanger.exchange(message);
13                 } catch (InterruptedException e) {
14                     e.printStackTrace();
15                 }
16             };
17
18             Runnable consumer = () -> {
19                 try {
20                     String message = exchanger.exchange(null);
21                     System.out.println(Thread.currentThread().getName());
22                 } catch (InterruptedException e) {
23                     e.printStackTrace();
24                 }
25             };
26         }
27     }

```

Console

```

terminated> multi22 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_6
Thread-0 exchanging message: Hello from Thread-0
Thread-1 received message: Hello from Thread-0

```

23. Write a Java program to create multiple threads and use CompletionService

```

1 package multithreading;
2 import java.util.concurrent.*;
3
4 class multi23 {
5     public static void main(String[] args) throws InterruptedException {
6         ExecutorService executor = Executors.newFixedThreadPool(3);
7         CompletionService<String> completionService = new ExecutorCompletionService(executor);
8
9         for (int i = 0; i < 5; i++) {
10             completionService.submit(() -> {
11                 return "Completed by " + Thread.currentThread().getThreadName();
12             });
13         }
14
15         for (int i = 0; i < 5; i++) {
16             System.out.println(completionService.take().get());
17         }
18     }
19 }
20
21
22
23

```

Console >
 <terminated> multi22 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\eccl
 Thread-0 exchanging message: Hello from Thread-0
 Thread-1 received message: Hello from Thread-0

24. Implement a program to demonstrate thread communication using TransferQueue.

```

1 multi21.java X 2 multi22.java 3 multi23.java 4 multi24.java X 5
practiseprograms/src/multithreading/multi21.java
  2 import java.util.concurrent.*;
  3
  4 class multi24 {
  5     public static void main(String[] args) throws InterruptedException {
  6         TransferQueue<Integer> queue = new LinkedTransferQueue<>();
  7
  8         Thread producer = new Thread(() -> {
  9             try {
10                 queue.transfer(1);
11                 System.out.println("Produced: 1");
12             } catch (InterruptedException e) {
13                 e.printStackTrace();
14             }
15         });
16
17         Thread consumer = new Thread(() -> {
18             try {
19                 Integer value = queue.take();
20                 System.out.println("Consumed: " + value);
21             } catch (InterruptedException e) {
22                 e.printStackTrace();
23             }
24         });
25

```

Console >
 <terminated> multi24 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\eccl
 Produced: 1
 Consumed: 1

25. Write a Java program to create multiple threads and use Scheduled Executor Service.

The screenshot shows the Eclipse IDE interface. At the top, there are tabs for multiple Java files: multi22.java, multi23.java, multi24.java, multi25.java, and multi26.java. The multi25.java tab is active. Below the tabs is the code for multi25.java:

```
1 package multithreading;
2 import java.util.concurrent.*;
3
4 class multi25 {
5     public static void main(String[] args) {
6         ScheduledExecutorService scheduler = Executors.newScheduled
7
8             Runnable task = () -> System.out.println("Task executed at:
9
10            scheduler.scheduleAtFixedRate(task, 0, 2, TimeUnit.SECONDS)
11        }
12    }
13}
14
15
```

At the bottom of the screen is the Eclipse Console window, which displays the output of the program:

```
Console ×
multi25 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\eclipse\plugins\org
Task executed at: 1740910310354
Task executed at: 1740910312353
```

26. Implement a program to demonstrate thread communication using Lock and Condition.

The screenshot shows the Eclipse IDE interface. At the top, there are tabs for multiple Java files: multi23.java, multi24.java, multi25.java, and multi26.java. The multi26.java tab is active. Below the tabs is the code for multi26.java:

```
1 package multithreading;
2 import java.util.concurrent.locks.*;
3
4 class multi26 {
5     private static Lock lock = new ReentrantLock();
6     private static Condition condition = lock.newCondition();
7
8     public static void main(String[] args) throws InterruptedException {
9         Thread producer = new Thread(() -> {
10             lock.lock();
11             try {
12                 System.out.println("Producer is working.");
13                 condition.await();
14                 System.out.println("Producer resumed.");
15             } catch (InterruptedException e) {
16                 e.printStackTrace();
17             } finally {
18                 lock.unlock();
19             }
20         });
21
22         Thread consumer = new Thread(() -> {
23             lock.lock();
24             try {
25                 System.out.println("Consumer is notifying");
26             } finally {
27                 lock.unlock();
28             }
29         });
30
31         producer.start();
32         consumer.start();
33
34         producer.join();
35         consumer.join();
36
37         System.out.println("Both threads have completed.");
38     }
39 }
```

At the bottom of the screen is the Eclipse Console window, which displays the output of the program:

```
Console ×
multi26 [Java Application] C:\Users\HP\Downloads\eclipse-jee-2024-12-R-win32-x86_64\eclipse\plugins\org
Consumer is notifying
Producer is working.
```