Module 2) Core Java

 Write a Java program to Take three numbers from the user and print the greatest number

```
C) 🔅 🚓 Share Run
                                                                                                                                                               Output
    1 · import java.util.Scanner;
                                                                                                                                                              Enter number 1:
    3 - class HelloWorld {
            public static void main(String[] args) {
                                                                                                                                                             Enter number 2:
                                                                                                                                                              Enter number 3:
8 9 10 111 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 }
                                                                                                                                                             C is greater
                  // scanner class
Scanner scan = new Scanner(System.in);
                                                                                                                                                              === Code Execution Successful ===
                  System.out.println("Enter number 1: ");
                  a = scan.nextInt();
System.out.println("Enter number 2: ");
b = scan.nextInt();
                  System.out.println("Enter number 3: ");
c = scan.nextInt();
                        System.out.println("A is greater ");
                  else if(b > a && b > c){

System.out.println("B is greater ");
                  }
else if(c > a && c > b){
   System.out.println("C is greater ");
```

 Write a Java program that takes the user to provide a single character from the alphabet. Print Vowel or Consonant, depending on the user input. If the user input is not a letter (between a and z or A and Z), or is a string of length > 1, print an error message.

 Write a Java program that takes a year from user and print whether that year is a leap year or not. B19. Write a program in Java to display the first 10 natural numbers using while loop.

```
Main.java
                                        [] 🔅
                                                    ∝ Share
                                                                Run
                                                                          Output
 1 import java.util.Scanner;
                                                                         Enter a year: 2000
3 - public class LeapYear {
                                                                         2000 is a leap year.
     public static void main(String[] args) {
                                                                         === Code Execution Successful ===
         Scanner scanner = new Scanner(System.in);
         System.out.print("Enter a year: ");
         int year = scanner.nextInt();
10
         if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {
              System.out.println(year + " is a leap year.");
              System.out.println(year + " is not a leap year.");
           scanner.close();
```

• Write a program in Java to input 5 numbers from keyboard and find their sum and average using for loop.

 Write a programin Java to display the pattern like right angle triangle with a number.

```
Main.java

| Topublic class NumberTriangle {
| Topublic class NumberTriangle {
| Topublic static void main(String[] args) {
| Topublic static void main(Stri
```

• Write a program in Java to make such a pattern like right angle triangle with number increased by 1 The pattern like:

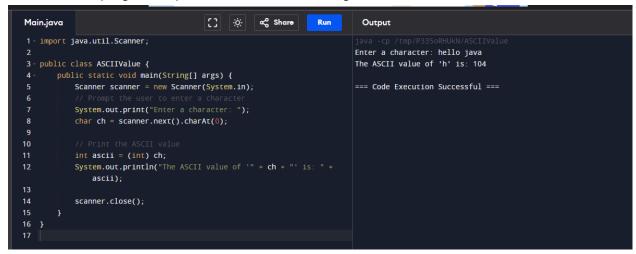
```
Main.java
                                          [] ⊹ c Share Run
                                                                               Output
                                                                                                                                                 Clear
1 public class NumberTriangleIncrement {
      public static void main(String[] args) {
          int rows = 5;
                                                                             2 3
           int num = 1;
                                                                             4 5 6
                                                                             7 8 9 10
          // Use nested loops to print the number triangle for (int i = 1; i <= rows; i++) {
                                                                             11 12 13 14 15
                                                                             === Code Execution Successful ===
                   System.out.print(num + " ");
               System.out.println();
```

• Write a Java program that reads a positive integer and count the number of digits of the number. Input an integer number less than ten billion: 125463 Number of digits in the number: 6

```
Clear
 Main.java
                                                             () 🔆 📽 Share Run
                                                                                                        Output
 1 - import java.util.Scanner;
                                                                                                      Input an integer number less than ten billion: 1293012
Number of digits in the number: 7
 3 public class DigitCounter {
        public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
                                                                                                       === Code Execution Successful ===
              long number = scanner.nextLong();
              int count = 0;
long temp = number;
11
12
13
14
15
16
17
18
              while (temp != 0) {
                   count++;
              System.out.println("Number of digits in the number: " + count);
               scanner.close();
```

• Write a Java program to count the letters, spaces, numbers and other characters of an input string.

Write a Java program to print the ASCII value of a given character.



• Write a Java program that accepts an integer (n) and computes the value of n+nn+nnn. Input number: 5 5 + 55 + 555

```
Main.java
                                         [] ⊹ oc Share
                                                                            Output
 1 * import java.util.Scanner;
                                                                           Input number: 10
3 → public class ComputeValue {
                                                                           10 + 1010 + 101010 = 102030
       public static void main(String[] args) {
          Scanner scanner = new Scanner(System.in);
                                                                          === Code Execution Successful ===
           System.out.print("Input number: ");
           int n = scanner.nextInt();
           int nn = Integer.parseInt("" + n + n);
           int nnn = Integer.parseInt("" + n + n + n);
12
           System.out.println(n + " + " + nn + " + " + nnn + " = " +
               result);
14
           scanner.close();
16
```

Write a Java program to display the system time.

• Write a Java program to print numbers between 1 to 100 which are divisible by 3, 5 and by both.

```
Main.java
                                                             Clear
    public class DivisibleNumbers {
         public static void main(String[] args) {
                                                                                                      Numbers divisible by 3:
             System.out.println("Numbers divisible by 3:");
for (int i = 1; i <= 100; i++) {
   if (i % 3 == 0) {</pre>
                                                                                                      3 6 9 12 15 18 21 24 27 30 33 36 39 42 45 48 51 54 57 60 63 66 69 72 75 78 81 84 87
                                                                                                          90 93 96 99
                                                                                                      Numbers divisible by 5:
5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100
                                                                                                      Numbers divisible by both 3 and 5:
                                                                                                      === Code Execution Successful ===
              System.out.println("\nNumbers divisible by 5:");
              for (int i = 1; i <= 100; i++) {
    if (i % 5 == 0) {
12
13
14
15
16
17
18
19
20
                       System.out.print(i + " ");
              System.out.println("\nNumbers divisible by both 3 and 5:");
              for (int i = 1; i <= 100; i++) {
  if (i % 3 == 0 && i % 5 == 0) {
                       System.out.print(i + "
21
22
23
24
```

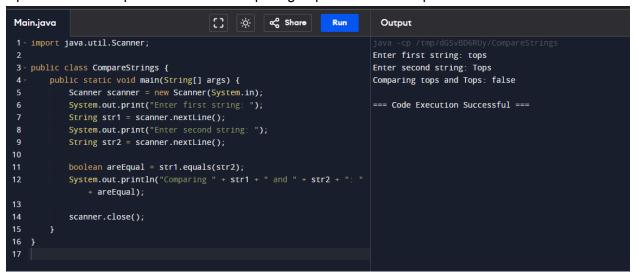
• W.A.J.P to get the character at the given index within the String. Original String = Tops Technologies! The character at position 0 is T, The character at position 10 is o

```
[] 🔆 🗬 Share
Main.java
                                                                            Output
  - import java.util.Scanner;
                                                                          Enter the index: 10
   public class CharacterAtIndex {
                                                                          The character at position 10 is o
       public static void main(String[] args) {
          String originalString = "Tops Technologies!";
                                                                          === Code Execution Successful ===
          Scanner scanner = new Scanner(System.in);
          System.out.print("Enter the index: ");
          int index = scanner.nextInt();
           if (index < 0 || index >= originalString.length()) {
               System.out.println("Invalid index.");
               char charAtPos = originalString.charAt(index);
               System.out.println("The character at position " + index +
14
                   " is " + charAtPos);
16
           scanner.close();
18
```

W.A.J.P to concatenate a given string to the end of another string.

```
Main.java
                                                     ∝ Share
                                                                   Run
                                                                             Output
   import java.util.Scanner;
                                                                           Enter first string: Tops
3 - public class ConcatenateStrings {
                                                                           Enter second string: Technologies
       public static void main(String[] args) {
                                                                           Concatenated string: TopsTechnologies
           Scanner scanner = new Scanner(System.in);
           System.out.print("Enter first string: ");
                                                                           === Code Execution Successful ===
           String str1 = scanner.nextLine();
           System.out.print("Enter second string: ");
8
           String str2 = scanner.nextLine();
           String concatenatedString = str1 + str2;
           System.out.println("Concatenated string: " +
              concatenatedString);
           scanner.close();
16
```

• W.A.J.P to compare a given string to the specified character sequence. Comparing topsint.com and topsint.com: true Comparing Topsint.com and topsint.com: false



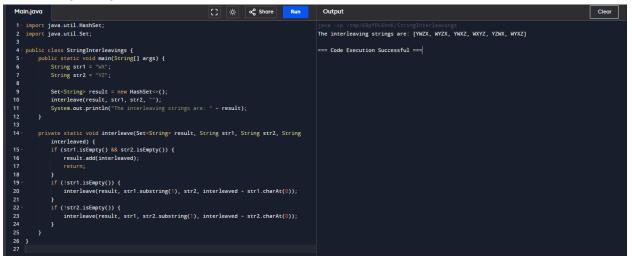
• W.A.J.P to check whether a given string ends with the contents of another string. "Java Exercises" ends with "se"? False "Java Exercise" ends with "se"? True

```
Main.java
1 - import java.util.Scanner;
                                                                           Enter the main string: test
                                                                           Enter the substring to check: st
3 - public class StringEndsWith {
      public static void main(String[] args) {
                                                                           "test" ends with "st"? true
          Scanner scanner = new Scanner(System.in);
           System.out.print("Enter the main string: ");
                                                                           === Code Execution Successful ===
          String mainString = scanner.nextLine();
           System.out.print("Enter the substring to check: ");
          String subString = scanner.nextLine();
           boolean endsWith = mainString.endsWith(subString);
           System.out.println("\"" + mainString + "\" ends with \"" +
               subString + "\"? " + endsWith);
           scanner.close();
```

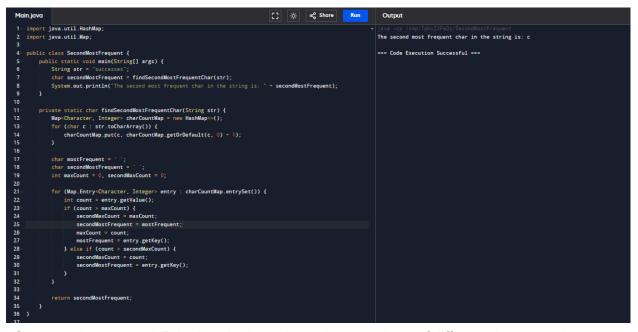
• W.A.J.P to check whether a given string starts with the contents of another string. Red is favorite color. Starts with Red? True Orange is also my favorite color. Starts with Red? False I3.

```
Main.java
                                                     ∝ Share
                                                                             Output
 1 · import java.util.Scanner;
                                                                           Enter the main string: hello
3 - public class StringStartsWith {
                                                                           Enter the substring to check: hell
      public static void main(String[] args) {
                                                                           "hello" starts with "hell"? true
           Scanner scanner = new Scanner(System.in);
           System.out.print("Enter the main string: ");
                                                                           === Code Execution Successful ===
          String mainString = scanner.nextLine();
8
           System.out.print("Enter the substring to check: ");
           String subString = scanner.nextLine();
          boolean startsWith = mainString.startsWith(subString);
           System.out.println("\"" + mainString + "\" starts with \"" +
               subString + "\"? " + startsWith);
           scanner.close();
15
```

• W.A.J.P to find all interleaving of given strings. The given strings are: WX YZ The interleaving strings are: YWZX WYZX YWXZ WXYZ YZWX WYXZ



• W.A.J.P to find the second most frequent character in a given string. The given string is: successes The second most frequent char in the string is: c



• Create a class named 'Print Number' to print various numbers of different data types by creating different methods with the same name 'printn' having a parameter for each data type.

```
| Non-journ | Color Printfulleder | Color Pr
```

• Create a class to print an integer and a character with two methods having the same name but different sequence of the integer and the character parameters. For example, if the parameters of the first method are of the form (int n, char c), then that of the second method will be of the form (char c, int n).

```
Main.java
                                                                                     Output
1 - class Example {
                                                                                   Method with int and char: 5, A
       void method(int n, char c) {
                                                                                   Method with char and int: B, 10
          System.out.println("Method with int and char: " + n + ", " + c);
                                                                                   === Code Execution Successful ===
       void method(char c, int n) {
          System.out.println("Method with char and int: " + c + ", " + n);
13 - public class TestExample {
      public static void main(String[] args) {
        Example example = new Example();
          example.method(5, 'A');
           example.method('B', 10);
18
```

• Create a class to print the area of a square and a rectangle. The class has two methods with the same name but different number of parameters. The method for printing area of a rectangle has two parameters which are length and breadth respectively while the other method for printing area of square has one parameter which is side of square.

```
Main.java
                                                 () |
                                                           ∝ Share
                                                                        Run
                                                                                   Output
 1 - class Area {
                                                                                 Area of square: 25
       void printArea(int side) {
                                                                                 Area of rectangle: 50
          int area = side * side;
           System.out.println("Area of square: " + area);
                                                                                 === Code Execution Successful ===
       void printArea(int length, int breadth) {
         int area = length * breadth;
           System.out.println("Area of rectangle: " + area);
15 - public class TestArea {
     public static void main(String[] args) {
          Area area = new Area();
         area.printArea(5); /
          area.printArea(5, 10); // Print area of rectangle
20
22
```

• Create a class with a method that prints "This is a parent class" and its subclass with another method that prints "This is child class". Now, create an object for each of the class and call 1 - method of parent class by object of parent class 2 - method of child class by object of child class 3 - method of parent class by object of child class

```
Main.java
                                                 [] ☆ < Share
                                                                        Run
                                                                                   Output
 1 - class Parent {
                                                                                  This is a parent class
       void message() {
                                                                                  This is a child class
           System.out.println("This is a parent class");
                                                                                  This is a child class
   class Child extends Parent {
       void message() {
           System.out.println("This is a child class");
15 - public class TestParentChild {
      public static void main(String[] args) {
          Parent parent = new Parent();
           Child child = new Child();
          parent.message();
           ((Parent) child).message(); // Method of parent class by object
```

• Create a class named 'Member' having the following members: 1. Data members 2. Name 3. Age 4. Phone number 5. Address 6. Salary It also has a method named 'printSalary' which prints the salary of the members.

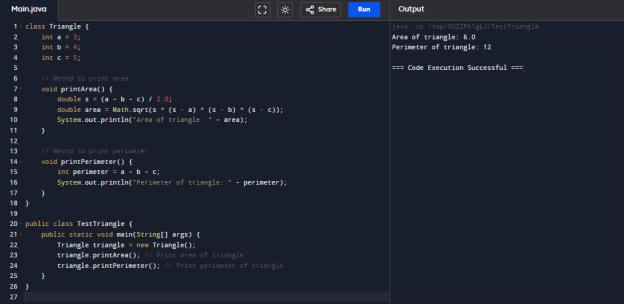
```
Main.java
                                                                         ∝ Share
                                                                                                Output
   class Member {
                                                                                               Salary: 50000.0
       String name;
        int age;
       String phoneNumber;
                                                                                               === Code Execution Successful ===
       String address;
       double salary;
       void printSalary() {
           System.out.println("Salary: " + salary);
10
12 }
14 public class TestMember {
      public static void main(String[] args) {
          Member member = new Member();
           member.name = "John Doe";
           member.age = 30;
          member.phoneNumber = "123-456-7890";
           member.address = "123 Main St";
           member.salary = 50000.0;
           member.printSalary(); // Print the salary
```

• Create a class named 'Rectangle' with two data members 'length' and 'breadth' and two methods to print the area and perimeter of the rectangle respectively. Its constructor having parameters for length and breadth is used to initialize the length and breadth of the rectangle. Let class 'Square' inherit the 'Rectangle' class with its constructor having a parameter for its side (suppose s) calling the constructor of its parent class as 'super (s, s)'. Print the area and perimeter of a rectangle and a square.

```
Holipin

1 class Recrapts {
1 class Recrapts {
2 int lamph;
3 int breath;
4 interpreter to notation interprete the second of content of the second of the se
```

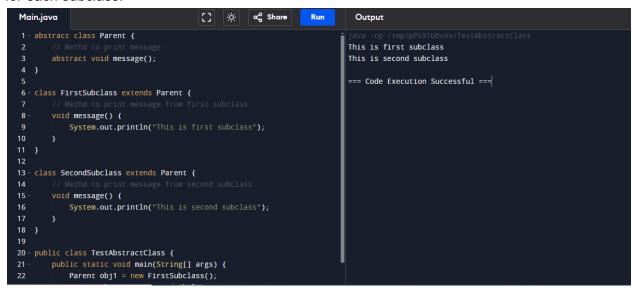
• Write a program to print the area and perimeter of a triangle having sides of 3, 4 and 5 units by creating a class named 'Triangle' without any parameter in its constructor.



• Print the sum, difference and product of two complex numbers by creating a class named 'Complex' with separate methods for each operation whose real and imaginary parts are entered by user.

```
Main.java
                                                    ∝ Share
1 · import java.util.Scanner;
                                                                         Enter real part of first complex number: 12
3 - class Complex {
                                                                         Enter imaginary part of first complex number: 12
                                                                         Enter real part of second complex number: 123
      double real:
       double imaginary;
                                                                         Enter imaginary part of second complex number: 123
                                                                         Sum: 135.0 + 135.0i
                                                                         Difference: -111.0 + -111.0i
      Complex(double real, double imaginary) {
                                                                        Product: 0.0 + 2952.0i
         this.real = real;
                                                                         === Code Execution Successful ===1
          this.imaginary = imaginary;
      Complex add(Complex other) {
         return new Complex(this.real + other.real, this.imaginary +
             other.imaginary);
       Complex subtract(Complex other) {
          return new Complex(this.real - other.real, this.imaginary
```

• Create an abstract class 'Parent' with a method 'message'. It has two subclasses each having a method with the same name 'message' that prints "This is first subclass" and "This is second subclass" respectively. Call the methods 'message' by creating an object for each subclass.



• Create an abstract class 'Bank' with an abstract method 'getBalance'. \$100, \$150 and \$200 are deposited in banks A, B and C respectively. 'BankA', 'BankB' and 'BankC' are subclasses of class 'Bank', each having a method named 'getBalance'. Call this method by creating an object of each of the three classes.

• We have to calculate the percentage of marks obtained in three subjects (each out of 100) by student A and in four subjects (each out of 100) by student B. Create an abstract class 'Marks' with an abstract method 'getPercentage'. It is inherited by two other classes 'A' and 'B' each having a method with the same name which returns the percentage of the students. The constructor of student A takes the marks in three subjects asits parameters and the marks in foursubjects as its parameters forstudent B. Create an object for each of the two classes and print the percentage of marks for both the students.

• Write a program to print the factorial of a number by defining a method named 'Factorial'. Factorial of any number n is represented by n! And is equal to 1*2*3*. *(n-1) *n. E.g.- 4! = 1*2*3*4 = 24 3! = 3*2*1 = 6 2! = 2*1 = 2 Also, 1! = 1 0! = 0

```
[] - cc Share Run
Main.java
                                                                           Output
                                                                                                                                            Clear
1 - import java.util.Scanner;
                                                                         Enter a number: 10
3 - public class Factorial {
                                                                         Factorial of 10 is 3628800
       static int factorial(int n) {
                                                                         === Code Execution Successful ===
          return n * factorial(n - 1);
       public static void main(String[] args) {
          Scanner scanner = new Scanner(System.in);
           System.out.print("Enter a number: ");
          int number = scanner.nextInt();
          System.out.println("Factorial of " + number + " is " +
              factorial(number));
           scanner.close();
```

• We have to calculate the area of a rectangle, a square and a circle. Create an abstract class 'Shape' with three abstract methods namely 'RectangleArea' taking two parameters, 'SquareArea' and 'CircleArea' taking one parameter each. The parameters of 'RectangleArea' are its length and breadth, that of 'SquareArea' is its side and that of 'CircleArea' is its radius. Now create another class 'Area' containing all the three methods 'RectangleArea', 'SquareArea' and 'CircleArea' for printing the area of rectangle, square and circle respectively. Create an object of class 'Area' and call all the three methods.

```
Main.java
                                            ∝ Share
                                                                                 Output
20
        void SquareArea(int side) {
           int area = side * side;
                                                                               Area of rectangle: 50
            System.out.println("Area of square: " + area);
                                                                               Area of square: 25
23
                                                                               Area of circle: 176.71458676442586
25
                                                                                === Code Execution Successful ===
26
        void CircleArea(double radius) {
            double area = Math.PI * radius * radius;
            System.out.println("Area of circle: " + area);
28
29
31
32 public class TestShape {
     public static void main(String[] args) {
34
            Area area = new Area():
           area.RectangleArea(5, 10); // Print area of rectangle
36
           area.SquareArea(5); // Print area of square
area.CircleArea(7.5); // Print area of circle
37
38
39
```

I3. Write a program which will ask the user to enter his/her marks (out of 100). Define a method that will display grades according to the marks entered as below: Marks Grade 1-100 A 1-90 B 1-80 B 1-70 C 1-60 D 41-50 DD40 Fail

```
Main.java
                                                     ∝ Share
                                                                  Run
                                                                             Output
1 - import java.util.Scanner;
                                                                           Enter your marks (out of 100): 59
3 → public class DisplayGrades {
                                                                           Grade: D
       static void displayGrade(int marks) {
                                                                           === Code Execution Successful ===
         if (marks >= 91 && marks <= 100) {
               System.out.println("Grade: A");
           } else if (marks >= 81 && marks <= 90) {
              System.out.println("Grade: B");
          } else if (marks >= 71 && marks <= 80) {
             System.out.println("Grade: B");
         } else if (marks >= 61 && marks <= 70) {
               System.out.println("Grade: C");
           } else if (marks >= 51 && marks <= 60) {
              System.out.println("Grade: D");
           } else if (marks >= 41 && marks <= 50) {
16
              System.out.println("Grade: DD");
           } else if (marks <= 40) {</pre>
              System.out.println("Grade: Fail");
           } else {
               System.out.println("Invalid marks");
```

• Create a class named 'Shape' with a method to print "This is this is shape". Then create two other classes named 'Rectangle', 'Circle' inheriting the Shape class, both having a method to print "This is rectangular shape" and "This is circular shape" respectively. Create a subclass 'Square' of 'Rectangle' having a method to print "Square is a rectangle". Now call the method of 'Shape' and 'Rectangle' class by the object of 'Square' class.

```
Main.java
                                         [] 🔅
                                                     ∝ Share
                                                                  Run
                                                                            Output
1 - class Shape {
                                                                           This is this is shape
        void printShape() {
                                                                          This is rectangular shape
           System.out.println("This is this is shape");
                                                                          Square is a rectangle
                                                                           === Code Execution Successful ===
8 - class Rectangle extends Shape {
10
       void printRectangle() {
           System.out.println("This is rectangular shape");
14
15 - class Circle extends Shape {
16
       void printCircle() {
           System.out.println("This is circular shape");
20
```

• W.A.J. P to demonstrate try catch block,

```
Main.java

| Typublic class TryCatchDemo {
| Typublic class TryCatchDemo {
| Typublic class TryCatchDemo {
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static void main(String[] args) {
| Array index is out of bounds!
| Typublic static vo
```

• Take two numbers from the user and perform the division operation and handle Arithmetic Exception. O/P- Enter two numbers: 10 0 Exception in thread main java.lang.ArithmeticException:/ by zero

```
Main.java
                                                        ∝ Share
 1 · import java.util.Scanner;
                                                                               Enter two numbers: 10 20
   public class DivisionDemo {
                                                                               Result: 0
        public static void main(String[] args) {
            Scanner scanner = new Scanner(System.in);
                                                                               === Code Execution Successful ===
            System.out.print("Enter two numbers: ");
            int num1 = scanner.nextInt();
            int num2 = scanner.nextInt();
10
            try {
                int result = num1 / num2; // Division operation
                System.out.println("Result: " + result);
            } catch (ArithmeticException e) {
13
                \textbf{System.out.println("} \textbf{Exception in thread main java.lang}
                     .ArithmeticException: / by zero");
15
            scanner.close();
   1
```

• W.A.J. P to demonstrate multiple catch blocks, (one is to handle divide by zero exception and another one is to handle ArrayIndexOutOfBoundException) int a [] = new int [5]; a [5]=30/0;

```
Main.java
                                          \Box
                                                                              Output
 1 - public class MultipleCatchDemo {
       public static void main(String[] args) {
                                                                            ArithmeticException: / by zero
           try {
                                                                            === Code Execution Successful ===
4
                int[] a = new int[5];
               a[5] = 30 / 0; // This will throw ArithmeticException
           } catch (ArithmeticException e) {
                System.out.println("ArithmeticException: " + e.getMessage
           } catch (ArrayIndexOutOfBoundsException e) {
                System.out.println("ArrayIndexOutOfBoundsException: " + e
                    .getMessage());
10
13
```

• W.A.J. P to implement the above program (pro.no-B27) using nesting of try-catch block. try { try {//code} catch (Exception e) {//code} catch (Exception e) {//code}

```
[] ×
                                                   ∝ Share
Main.java
                                                                           Output
                                                                                                                                           Clear
1 public class NestedTryCatchDemo {
      public static void main(String[] args) {
                                                                         Inner catch: ArithmeticException: / by zero
                                                                         Outer catch: ArrayIndexOutOfBoundsException: Index 5 out of bounds for
                                                                            length 5
                                                                         === Code Execution Successful ===
                  System.out.println("Inner catch: ArithmeticException:
                        + e.getMessage());
              int[] a = new int[5];
          } catch (ArrayIndexOutOfBoundsException e) {
              System.out.println("Outer catch
                  ArrayIndexOutOfBoundsException: " + e.getMessage());
```

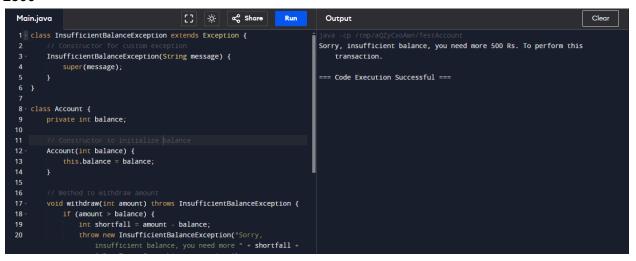
• W.A.J. P to demonstrate try catch block, take two numbers from the user by Command line argument and perform the division operation and handle Arithmetic O/PException in thread main java. Lang. Arithmetic Exception:/ by zero

```
Main.java
                                         [] ☆ < Share
 1 - public class DivisionCommandLine {
       public static void main(String[] args) {
                                                                          Please provide two numbers as command line arguments.
           if (args.length < 2) {</pre>
               System.out.println("Please provide two numbers as command
                                                                          === Code Execution Successful ===
           try {
               int num1 = Integer.parseInt(args[0]);
               int num2 = Integer.parseInt(args[1]);
              int result = num1 / num2; // Division operation
               System.out.println("Result: " + result);
           } catch (ArithmeticException e) {
              System.out.println("Exception in thread main java.lang
15
```

• W.A.J.P to create the validate method that takes integer value as a parameter. If the age is less than 18, then throw an Arithmetic Exception otherwise print a message welcome to vote. O/P- Enter your age: 16 Exception in thread main java. Lang. Arithmetic Exception: not valid

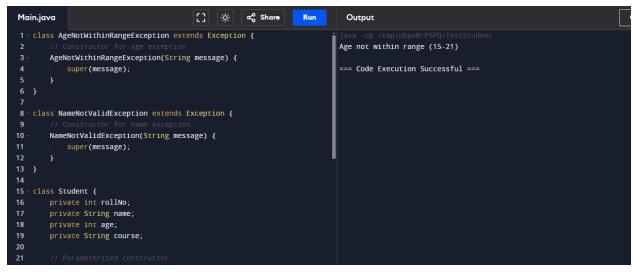
```
[] 🌣 🚓 Share
                                                                Run
                                                                                                                                            Clear
Main.java
                                                                           Output
1 - import java.util.Scanner;
                                                                          Enter your age: 12
  public class ValidateAge {
                                                                         Exception in thread main java.lang.ArithmeticException: not valid
       static void validate(int age) {
                                                                          === Code Execution Successful ===
          if (age < 18) {
              throw new ArithmeticException("not valid");
              System.out.println("Welcome to vote");
      public static void main(String[] args) {
          Scanner scanner = new Scanner(System.in);
          System.out.print("Enter your age: ");
          int age = scanner.nextInt();
              validate(age); // Validate age
          } catch (ArithmeticException e) {
```

• W.A.J.P to create a custom exception if Customer withdraw amount which is greater than account balance then program will show custom exception otherwise amount will deduct from account balance. Account balance is: 2000 Enter withdraw amount: 2500



Sorry, insufficient balance, you need more 500 Rs. To perform this transaction.

- W.A.J.P to create a class Student with attributes roll no, name, age and course. Initialize values through parameterized constructor. If age of student is not in between 15 and 21 then generate user defined exception
- "AgeNotWithinRangeException". If name contains numbers or special symbols raise exception "NameNotValidException". Define the two exception classes.



• W.A.J. P to create one thread by implementing Runnable interface in Class.



• W.A.J. P to create one thread by extending Thread class in another Class.

• W.A.J.P to create 2 threads and execute that threads by providing sleep time as 2000ms and check the execution.

```
∝ Share
 1 - class SleepThread extends Thread {
       public void run() {
                                                                           Thread started: 11
                                                                           Thread started: 10
           trv {
               System.out.println("Thread started: " + Thread
                                                                           Thread ended: 11
                   .currentThread().getId());
                                                                           Thread ended: 10
               Thread.sleep(2000); //
               System.out.println("Thread ended: " + Thread
                                                                           === Code Execution Successful ===
                   .currentThread().getId());
           } catch (InterruptedException e) {
               System.out.println("Thread interrupted.");
13 - public class TestSleepThreads {
       public static void main(String[] args) {
14
           SleepThread t1 = new SleepThread();
           SleepThread t2 = new SleepThread();
18
           t1.start(); // Start first thread
            t2.start(); // Start second threa
20
```

• W.A.J.P to start the same Thread twice by calling start () method twice. Test ThreadTwice1 t1=new TestThreadTwice1(); t1.start (); t1.start ();



• W.A.J.P to create 2 threads and make one thread as Daemon Thread by using set Daemon () method of Thread class and check whether the thread is set daemon or not by using is Daemon () method.

TestDaemonThread2 t1=new TestDaemonThread2();

TestDaemonThread2 t2=new TestDaemonThread2(); t1.start();

t1.setDaemon(true);//will throw exception here t2.start();

```
Main.java
                                                      ∝ Share
                                                                   Run
                                                                             Output
 1 class TestDaemonThread2 extends Thread {
       public void run() {
                                                                            Daemon thread running.
           System.out.println("Daemon thread running.");
                                                                            Daemon thread running.
                                                                            Is t1 a daemon thread? true
4
                                                                           === Code Execution Successful ===
6
   public class TestDaemon {
       public static void main(String[] args) {
8
           TestDaemonThread2 t1 = new TestDaemonThread2();
           TestDaemonThread2 t2 = new TestDaemonThread2();
           t1.setDaemon(true); // Set t1 as daemon thread
           t1.start(); // Start daemon thread
           t2.start(); // Start normal thread
           System.out.println("Is t1 a daemon thread? " + t1.isDaemon());
19
```

Write a Java program to create a new array list, add some colors (string) and print out the collection.

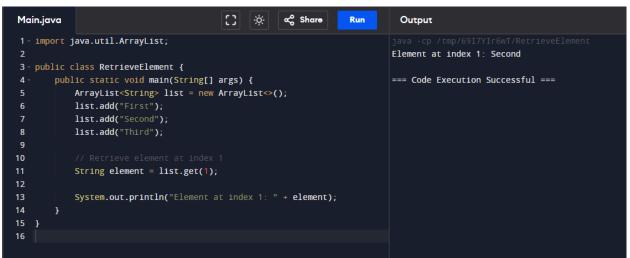
```
∝ Share
Main.java
                                                                             Output
   import java.util.ArrayList;
                                                                           ArrayList elements: [Red, Green, Blue, Yellow]
  import java.util.Collections;
4 - public class ArrayListDemo {
                                                                           Green
       public static void main(String[] args) {
                                                                           Blue
                                                                           Yellow
           ArrayList<String> colors = new ArrayList<>();
                                                                           After insertion: [Pink, Red, Green, Blue, Yellow]
           colors.add("Red");
                                                                           Element at index 2: Green
           colors.add("Green");
                                                                           After update: [Pink, Red, Purple, Blue, Yellow]
           colors.add("Blue");
                                                                           After removal: [Pink, Red, Blue, Yellow]
           colors.add("Yellow");
                                                                           Contains Green? false
                                                                           Sorted list: [Blue, Pink, Red, Yellow]
                                                                           Copied list: [Blue, Pink, Red, Yellow]
           System.out.println("ArrayList elements: " + colors);
                                                                           Shuffled list: [Yellow, Red, Pink, Blue]
                                                                           === Code Execution Successful ===
           for (String color : colors) {
               System.out.println(color);
           colors.add(0, "Pink");
```

Write a Java program to iterate through all elements in an array list.

```
[] 🔅
Main.java
                                                     ∝ Share
                                                                  Run
                                                                            Output
1 - import java.util.HashSet;
                                                                          HashSet elements: [Red, Blue, Yellow, Green]
3 - public class HashSetDemo {
       public static void main(String[] args) {
                                                                           === Code Execution Successful ===
          HashSet<String> set = new HashSet<>();
           set.add("Red");
           set.add("Green");
           set.add("Blue");
8
           set.add("Yellow");
           System.out.println("HashSet elements: " + set);
```

• Write a Java program to insert an element into the array list at the first position.

• Write a Java program to retrieve an element (at a specified index) from a given array List.



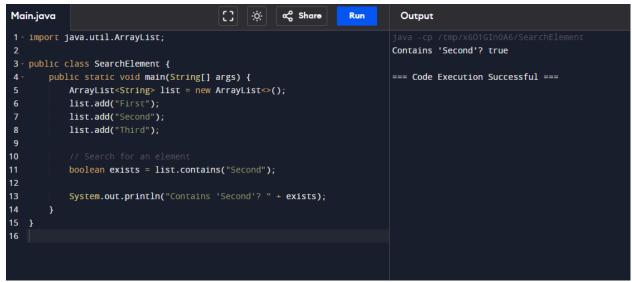
Write a Java program to update specific array element by given element.

```
Main.java
                                         [] ×
                                                    ∝ Share
                                                                 Run
1 - import java.util.ArrayList;
                                                                          ArrayList after update: [First, Updated Second, Third]
3 - public class UpdateElement {
       public static void main(String[] args) {
                                                                          === Code Execution Successful ===
          ArrayList<String> list = new ArrayList<>();
           list.add("First");
           list.add("Second");
           list.add("Third");
10
           list.set(1, "Updated Second");
           System.out.println("ArrayList after update: " + list);
  1
```

Write a Java program to remove the third element from an array list.

```
Main.java
                                                      ∝ೆ Share
                                                                   Run
                                                                              Output
 1 - import java.util.ArrayList;
                                                                            ArrayList after removal: [First, Second, Fourth]
3 → public class RemoveElement {
       public static void main(String[] args) {
                                                                            === Code Execution Successful ===
           ArrayList<String> list = new ArrayList<>();
           list.add("First");
           list.add("Second");
           list.add("Third");
           list.add("Fourth");
10
            list.remove(2);
14
           System.out.println("ArrayList after removal: " + list);
```

Write a Java program to search an element in an array list.



Write a Java program to sort a given array list.

```
() 🔅
Main.java
                                                     م$ Share
                                                                  Run
 1 - import java.util.ArrayList;
2 import java.util.Collections;
                                                                           Sorted ArrayList: [Apple, Banana, Cherry]
4 - public class SortArrayList {
                                                                           === Code Execution Successful ===
       public static void main(String[] args) {
           ArrayList<String> list = new ArrayList<>();
           list.add("Banana");
           list.add("Apple");
           list.add("Cherry");
           Collections.sort(list);
           System.out.println("Sorted ArrayList: " + list);
```

Write a Java program to copy one array list into another.

```
Run
Main.java
                                                                               Output
 1 - import java.util.ArrayList;
                                                                             Copied ArrayList: [First, Second, Third]
3 - public class CopyArrayList {
       public static void main(String[] args) {
                                                                              === Code Execution Successful ===
           ArrayList<String> originalList = new ArrayList<>();
            originalList.add("First");
           originalList.add("Second");
           originalList.add("Third");
            // Copy originalList to newList
ArrayList<String> newList = new ArrayList<>(originalList);
            System.out.println("Copied ArrayList: " + newList);
14
```

• Write a Java program to shuffle elements in an array list.

```
[] ×
Main.java
                                                     ∝ Share
                                                                            Output
1 import java.util.ArrayList;
2 import java.util.Collections;
                                                                          Shuffled ArrayList: [Third, Second, First]
4 - public class ShuffleArrayList {
                                                                          === Code Execution Successful ===
       public static void main(String[] args) {
          ArrayList<String> list = new ArrayList<>();
           list.add("First");
8
          list.add("Second");
           list.add("Third");
           Collections.shuffle(list);
           System.out.println("Shuffled ArrayList: " + list);
```

• Write a Java program to append the specified element to the end of a hash set

• Write a Java program to iterate through all elements in a hash list.

```
Main.java
                                         [] 🔅
                                                     ∝ Share
1 - import java.util.HashSet;
                                                                           Red
3 public class IterateHashSet {
                                                                           Blue
       public static void main(String[] args) {
                                                                           Green
           HashSet<String> set = new HashSet<>();
           set.add("Red");
                                                                           === Code Execution Successful ===
6
           set.add("Green");
           set.add("Blue");
9
           for (String color : set) {
               System.out.println(color);
16
```

Write a Java program to get the number of elements in a hash set.

```
Main.java
                                                                             Output
 1 - import java.util.HashSet;
                                                                            Number of elements in HashSet: 3
3 public class CountHashSetElements {
       public static void main(String[] args) {
                                                                            === Code Execution Successful ===
           HashSet<String> set = new HashSet<>();
           set.add("Red");
6
           set.add("Green");
           set.add("Blue");
           int size = set.size();
12
           System.out.println("Number of elements in HashSet: " + size);
16
```

 Write a Java program to associate the specified value with the specified key in a Hash Map.

```
Main.java
                                                       مر Share
                                                                    Run
                                                                              Output
1 - import java.util.HashMap;
                                                                             HashMap: {Bob=30, Alice=25}
3 - public class PutInHashMap {
       public static void main(String[] args) {
                                                                             === Code Execution Successful ===
           HashMap<String, Integer> map = new HashMap<>();
           map.put("Alice", 25);
9
           map.put("Bob", 30);
           System.out.println("HashMap: " + map);
12
13
```

• Write a Java program to count the number of key-value (size) mappings in a map.

```
[] ×
                                                                 Run
Main.java
                                                    ∝ Share
                                                                           Output
  import java.util.HashMap;
                                                                         Number of key-value mappings in HashMap: 2
  public class CountMapEntries {
      public static void main(String[] args) {
                                                                          === Code Execution Successful ===
         HashMap<String, Integer> map = new HashMap<>();
          map.put("Alice", 25);
          map.put("Bob", 30);
          int size = map.size();
          System.out.println("Number of key-value mappings in HashMap: "
              + size);
      1
```

Write a Java program to reverse elements in an array list.

```
Main.java
                                         α⇔ Share
                                                                  Run
                                                                            Output
 1 - import java.util.ArrayList;
                                                                          Reversed ArrayList: [Third, Second, First]
2 import java.util.Collections;
4 - public class ReverseArrayList {
                                                                           === Code Execution Successful ===
       public static void main(String[] args) {
          ArrayList<String> list = new ArrayList<>();
           list.add("First");
           list.add("Second");
8
           list.add("Third");
           Collections.reverse(list);
13
           System.out.println("Reversed ArrayList: " + list);
16
```

Write a Java program to extract a portion of an array list.

```
Main.java
                                         ∝് Share
                                                                   Run
                                                                             Output
 1 - import java.util.ArrayList;
                                                                           SubList: [Second, Third]
3 public class SubListArrayList {
       public static void main(String[] args) {
                                                                           === Code Execution Successful ===
           ArrayList<String> list = new ArrayList<>();
           list.add("First");
           list.add("Second");
8
            list.add("Third");
           list.add("Fourth");
           ArrayList<String> subList = new ArrayList<>(list.subList(1, 3
13
            System.out.println("SubList: " + subList);
16
```

Write a Java program to compare two array lists.

```
lain.java
                                       ∝ Share
                                                                Run
                                                                          Output
 import java.util.ArrayList;
                                                                        Are the two ArrayLists equal? true
 public class CompareArrayLists {
     public static void main(String[] args) {
                                                                        === Code Execution Successful ===
         ArrayList<String> list1 = new ArrayList<>();
         list1.add("A");
         list1.add("B");
         ArrayList<String> list2 = new ArrayList<>();
         list2.add("A");
         list2.add("B");
         boolean isEqual = list1.equals(list2);
         System.out.println("Are the two ArrayLists equal? " + isEqual
```

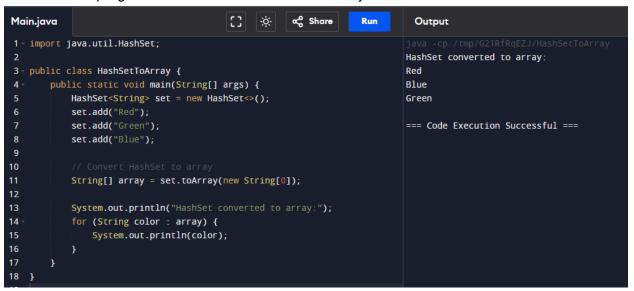
Write a Java program of swap two elements in an array list.

```
Main.java
                                         [] 🔅
                                                     ∝ Share
                                                                 Run
1 · import java.util.ArrayList;
                                                                          ArrayList after swap: [Third, Second, First]
3 - public class SwapArrayListElements {
       public static void main(String[] args) {
                                                                          === Code Execution Successful ===
          ArrayList<String> list = new ArrayList<>();
           list.add("First");
          list.add("Second");
          list.add("Third");
9
           String temp = list.get(0);
           list.set(0, list.get(2));
           list.set(2, temp);
           System.out.println("ArrayList after swap: " + list);
```

Write a Java program to join two array lists.

```
Main.java
                                          ∝ Share
                                                                  Run
                                                                             Output
 1 - import java.util.ArrayList;
                                                                           Joined ArrayList: [A, B, C, D]
3 - public class JoinArrayLists {
       public static void main(String[] args) {
                                                                           === Code Execution Successful ===
           ArrayList<String> list1 = new ArrayList<>();
           list1.add("A");
           list1.add("B");
 8
           ArrayList<String> list2 = new ArrayList<>();
9
10
           list2.add("C");
           list2.add("D");
13
14
            list1.addAll(list2);
15
            System.out.println("Joined ArrayList: " + list1);
16
18
   }
```

• Write a Java program to convert a hash set to an array.



Write a Java program to convert a hash set to a List/Array List.

```
Main.java
                                                     ∝ Share
                                                                   Run
                                                                             Output
1 - import java.util.ArrayList;
2 import java.util.HashSet;
                                                                           HashSet converted to ArrayList: [Red, Blue, Green]
4 public class HashSetToArrayList {
                                                                           === Code Execution Successful ===
       public static void main(String[] args) {
           HashSet<String> set = new HashSet<>();
6
           set.add("Red");
           set.add("Green");
           set.add("Blue");
9
           ArrayList<String> list = new ArrayList<>(set);
           System.out.println("HashSet converted to ArrayList: " + list);
```

 Write a Java program to check whether a map contains key-value mappings (empty) or not.

Write a Java program to increase the size of an array list.

```
import java.util.ArrayList;

public class IncreaseArrayListSize {
    public static void main(String[] args) {
        ArrayList < String > list = new ArrayList < > ();
        list.add("Element 1");
        list.add("Element 2");

        // Increase the size of the ArrayList
        list.ensureCapacity(10); // Ensure capacity of at least 10

        System.out.println("ArrayList capacity increased");
    }
}
```

• Write a Java program to replace the second element of an Array List with the specified element.



• Write a Java program to print all the elements of an Array List using the position of the elements.

• Write a Java program to compare two sets and retain elements which are same on both sets.

```
Main.java
                                                     ∝ Share
                                                                  Run
                                                                            Output
1 - import java.util.HashSet;
                                                                           Common elements: [B, C]
3 - public class CompareSets {
      public static void main(String[] args) {
                                                                           === Code Execution Successful ===
          HashSet<String> set1 = new HashSet<>();
          set1.add("A");
           set1.add("B");
          set1.add("C");
8
          HashSet<String> set2 = new HashSet<>();
           set2.add("B");
          set2.add("C");
           set2.add("D");
           set1.retainAll(set2);
           System.out.println("Common elements: " + set1);
```

• Write a Java program to get a collection view of the values contained in this map.