**LAB-8**

1. Write a program that tries to access an element outside the bounds of an array and handles the ArrayIndexOutOfBoundsException by printing a user-friendly message.

**Code**:-

**package** Anudip;

**public** **class** ArrayIndexExample {

**public** **static** **void** main(String[] args) {

**int**[] arr = { 1, 2, 3, 4, 5 };

**try** {

// Trying to access an element outside the bounds of the array

**int** value = arr[6];

} **catch** (ArrayIndexOutOfBoundsException e) {

// Handling the exception and printing a user-friendly message

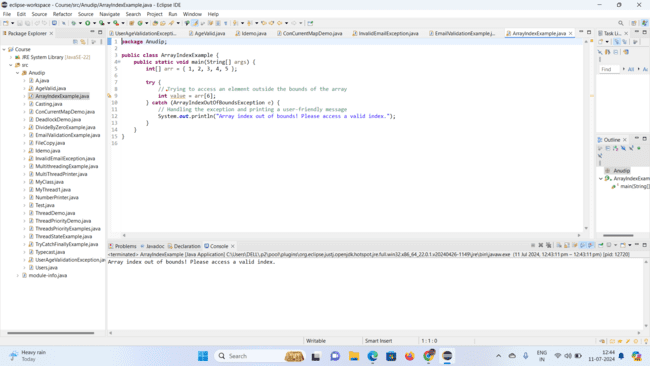
System.***out***.println("Array index out of bounds! Please access a valid index.");

}

}

}

**Output**:-



1. Write a program that attempts to divide a number by zero and handles the ArithmeticException by printing a message that division by zero is not allowed.

**Code**:-

**package** Anudip;

**public** **class** DivideByZeroDemo {

**public** **static** **void** main(String[] args) {

**int** numerator = 10;

**int** denominator = 0;

**try** {

// Attempting to divide by zero

**int** result = numerator / denominator;

} **catch** (ArithmeticException e) {

// Handling the exception and printing a message

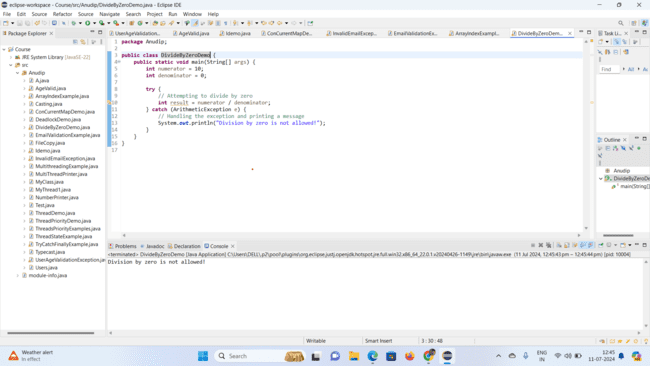
System.***out***.println("Division by zero is not allowed!");

}

}

}

**Output**:-



1. Write a Java program that reads an integer input from the user and throws an IllegalArgumentException if the input is negative. Display an appropriate message when the exception is caught.

**Code**:-

**package** Anudip;

**import** java.util.Scanner;

**public** **class** IllegalArgumentExceptionDemo {

**public** **static** **void** main(String[] args) {

Scanner scanner = **new** Scanner(System.***in***);

System.***out***.print("Enter a positive integer: ");

**int** number = scanner.nextInt();

**try** {

*checkPositive*(number);

System.***out***.println("You entered: " + number);

} **catch** (IllegalArgumentException e) {

System.***out***.println("Error: " + e.getMessage());

}

}

**public** **static** **void** checkPositive(**int** number) {

**if** (number < 0) {

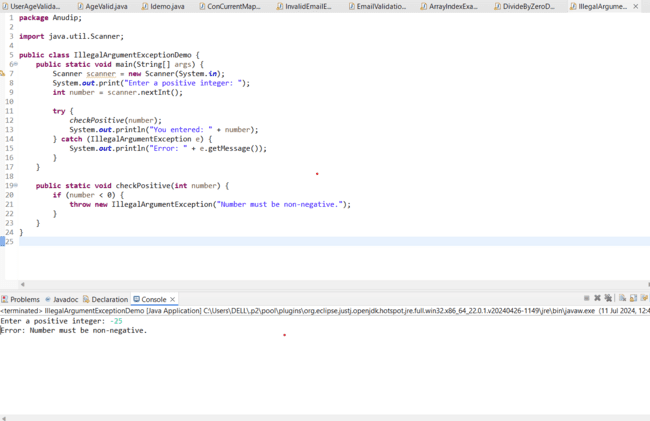
**throw** **new** IllegalArgumentException("Number must be non-negative.");

}

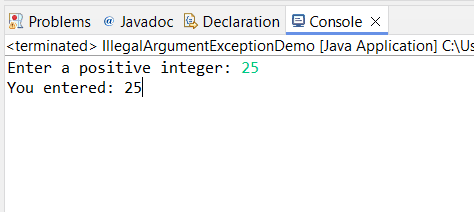
}

}

**Output**:-



**When User enter positive number then output like this**:-



1. Create a Java method that divides two numbers and declares that it throws an ArithmeticException. Handle the exception in the main method.

**Code**:-

**package** Anudip;

**public** **class** ArithmeticExceptionMethod {

**public** **static** **void** main(String[] args) {

**try** {

**int** result = *divide*(10, 0);

System.***out***.println("Result: " + result);

} **catch** (ArithmeticException e) {

System.***out***.println("Error: Division by zero is not allowed.");

}

}

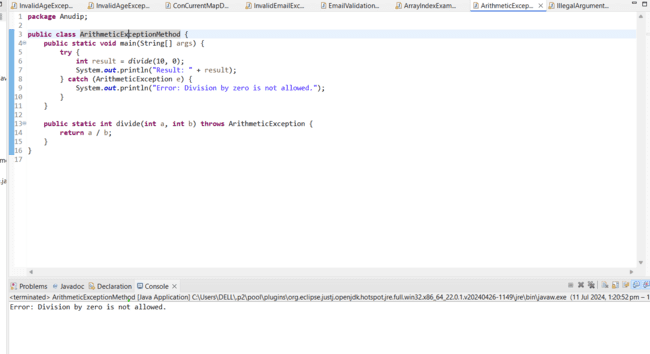
**public** **static** **int** divide(**int** a, **int** b) **throws** ArithmeticException {

**return** a / b;

}

}

**Output**:-



1. Define a custom exception called InvalidAgeException. Write a Java program that throws this exception if the age provided is less than 18. Handle the exception and display an appropriate message.

**Code**:-

InvalidAgeException.java Class

**package** Anudip;

**import** java.lang.Exception;

**class** InvalidAgeException **extends** Exception {

**public** InvalidAgeException(String message) {

**super**(message);

}

}

InvalidAgeExceptionExample.java Class

**package** Anudip;

**import** java.lang.Exception;

//Define the custom exception

**public** **class** InvalidAgeExceptionExample {

**public** **static** **void** main(String[] args) {

**try** {

*checkAge*(16);

} **catch** (InvalidAgeException e) {

System.***out***.println("Error: " + e.getMessage());

}

}

**public** **static** **void** checkAge(**int** age) **throws** InvalidAgeException {

**if** (age < 18) {

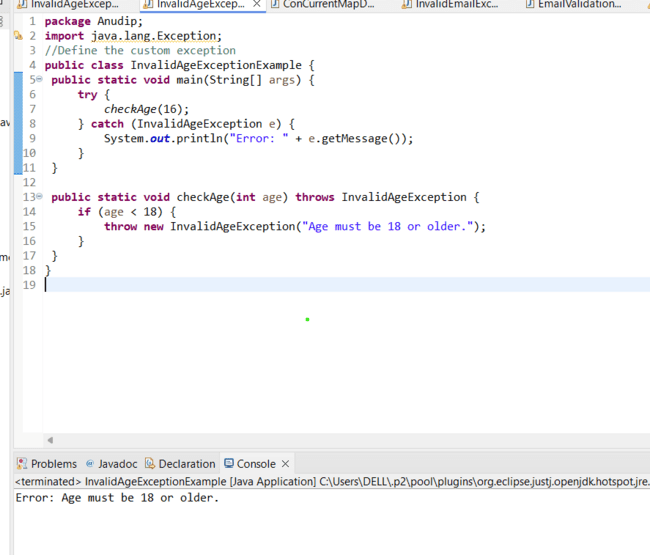
**throw** **new** InvalidAgeException("Age must be 18 or older.");

}

}

}

**Output**:-



1. Write a Java program that has a method to validate a user's email address. The method should throw a custom exception InvalidEmailException if the email does not contain @ and .. Handle the exception in the main method.

**Code**:-

InvalidEmailException.java Class

**package** Anudip;

//Define the custom exception

**public** **class** InvalidEmailException **extends** Exception {

**public** InvalidEmailException(String message) {

**super**(message);

}

}

EmailValidationExample.java Class

**package** Anudip;

**public** **class** EmailValidationExample {

**public** **static** **void** main(String[] args) {

String email = "xyz@gmail.com";

**try** {

*validateEmail*(email); // Example of an invalid email

System.***out***.println("Your Email is right."+email);

} **catch** (InvalidEmailException e) {

System.***out***.println("Error: " + e.getMessage());

}

}

**public** **static** **void** validateEmail(String email) **throws** InvalidEmailException {

**if** (!email.contains("@") || !email.contains(".")) {

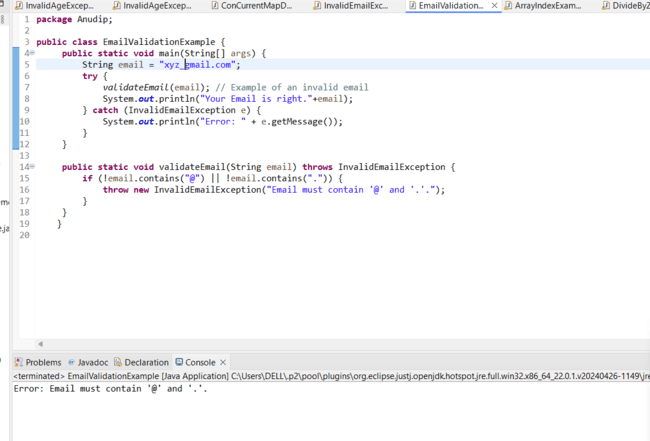
**throw** **new** InvalidEmailException("Email must contain '@' and '.'.");

}

}

}

**Output**:-



**When Email is correct then output like this**:-

