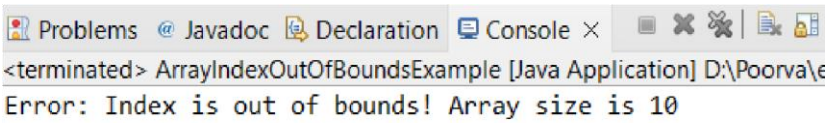
1. Write a program that tries to access an element outside the bounds of an array and handles the ArrayIndexOutOfBoundsException by printing a user-friendly message.

Program :

```
package demo; public class
ArrayIndexOutOfBoundsExample { public static void
main(String[] args) { // TODO Auto-generated
method stub int[] numbers = {1, 2, 3, 4, 5, 6, 7 ,
8, 9, 10}; // Try to access an element outside the
bounds try { int element = numbers[10];

System.out.println("Element at index 10: " + element); // This line won't be executed
} catch (ArrayIndexOutOfBoundsException e) {

System.out.println("Error: Index is out of bounds! Array size is " + numbers.length);

}
}
}
```

Output :

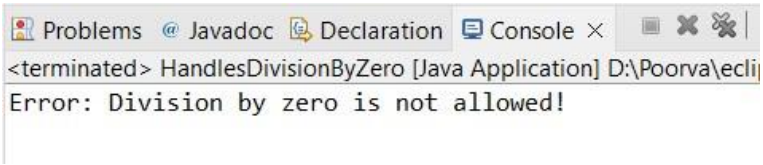


2. Write a program that attempts to divide a number by zero and handles the ArithmeticException by printing a message that division by zero is not allowed.

Program :

```
package demo; public class

HandlesDivisionByZero { public static

void main(String[] args) { // TODO Auto-

generated method stub int numerator = 10;

int denominator = 0; // Try to divide by

zero try { int result = numerator /

denominator;

System.out.println("Result: " + result); // This line won't be executed

} catch (ArithmeticException e) {

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

}

}

}
```

Output :



3. 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.

Program :

```
package demo; import java.util.Scanner;

public class IntegerInputChecker { public

static void main(String[] args) { // TODO

Auto-generated method stub Scanner scanner

= new Scanner(System.in); try {


System.out.print("Please enter an integer: "); // Prompting the user to enter an

integer int number = scanner.nextInt(); // Reading the integer input from the user if

(number < 0) { // Checking if the entered number is negative


throw new IllegalArgumentException("Error: The number cannot be negative."); //
Throwing an IllegalArgumentException if the number is negative

}

System.out.println("You have entered: " + number); // Printing the entered number if
it is not negative

} catch (IllegalArgumentException e) { // Catching the IllegalArgumentException

System.out.println("Caught an exception: " + e.getMessage()); // this indicating that
an exception was caught along with the exception's message

} finally { scanner.close();


}

}

}
```

Output :

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> IntegerInputChecker [Java Application] D:\Poorva\eclipse\plugins\org.eclip
Please enter an integer: -9
Caught an exception: Error: The number cannot be negative.
```

4. 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.

Program :

```
package demo;

//Define a custom exception class class

InvalidAgeException extends Exception { public

InvalidAgeException(String message) {

super(message);


} } public class AgeValidation { // Method to check if age is valid

public static void validateAge(int age) throws InvalidAgeException {

if (age < 18) {


// Throw custom exception if age is less than 18 throw
new InvalidAgeException("Age is less than 18.");

} } public static void main(String[]
args) { // TODO Auto-generated method
stub int age = 15; try {

// Check the provided age validateAge(age);

// If age is valid, print a success message

System.out.println("Age is valid.");

} catch (InvalidAgeException e)
{

// Catch the exception and show the message

System.out.println("Exception caught: " + e.getMessage());

}

}
```

```
}
```

Output :

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> AgeValidation [Java Application] D:\Poorva\eclipse\p
Exception caught: Age is less than 18.
```

5 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.

Program :

```
package demo;

//Custom exception class for invalid email class InvalidEmailException
extends Exception { public InvalidEmailException(String message) {
super(message); // Call the constructor of the parent class Exception

} } public class EmailValidation { // Method to validate email public static

void validateEmail(String email) throws InvalidEmailException { if

(!email.contains("@") || !email.contains(".")) { // If email does not have @

or ., throw InvalidEmailException throw new InvalidEmailException("Email must

contain '@' and '.'"); } } public static void main(String[] args) {


// TODO Auto-generated method stub

String email = "bpoorvagmail.com"; // Set an example email for testing
try { validateEmail(email); // Validate the provided email

System.out.println("Email is valid."); // Print if email is valid

} catch (InvalidEmailException e) {


System.out.println("Exception caught: " + e.getMessage()); // Catch and print the
exception message

}

}

}
```

Output :

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> EmailValidation [Java Application] D:\Poorva\eclipse\plugins
Exception caught: Email must contain '@' and '.'
```