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 MyPackage;

public class ArrayBoundExcep

{

Public static void main (String[] args) {

//declaring an array of integers

Int[] array = {1, 2, 3, 4, 5};

Try {

Int element = array[5]; // Trying to access an element outside the array bounds

System.out.println(“Element at index 5: “ + element);

} catch(ArrayIndexOutOfBoundsException e) {

//Handles exception by user friendly message

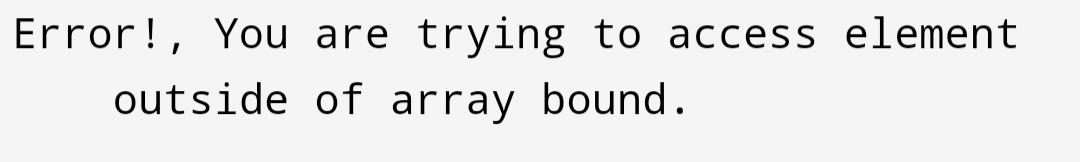
System.out.println(“Error!, You are trying to access element outside of array bound.”);

}

}

}

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.

**Code**:-

package MyPackage;

import java.lang.Exception;

public class DivideByZero

{

Public static void main (String[] args)

{

//declaring 2 integers

Int num1 = 10;

Int num2 = 0;

Try {

Int result = num1 / num2; //trying to divide number by zero

System.out.println(“Result : “ + result);

} catch (ArithmeticException e) {

//handling the exception

e.printStackTrace();

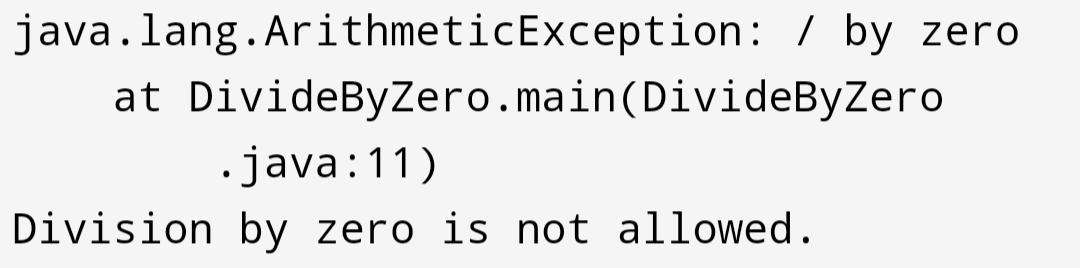
System.out.println(“Division by zero is not allowed.”);

}

}

}

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.

**Code**:-

package MyPackage;

import java.util.Scanner;

public class IllArguExcep

{

Public static void main (String[] args)

{

//creating object of scanner class

Scanner sc = new Scanner(System.in);

System.out.print(“Enter an integer number : “); //taking number from user as input and storing it

Int number = sc.nextInt();

Try {

//checks if number is negative

If (number < 0) {

Throw new IllegalArgumentException(“The input must be a non-negative integer.”);

}

//prints number if positive

System.out.println(“The number is : “ + number);

} catch (IllegalArgumentException e) {

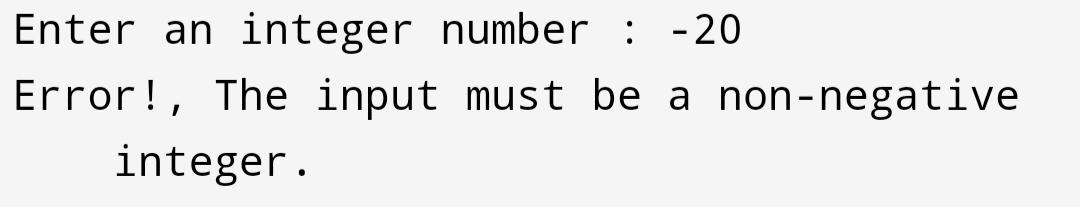
//handles exception

System.out.println(“Error!, “ + e.getMessage());

}

}

}

Output:-

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

Code:-

package MyPackage;

public class ArithExcep

{

//defining mathod to throw exception

Public static void display(int num1, int num2) throws ArithmeticException {

If (num2 == 0) {

Throw new ArithmeticException(“Cannot divide by zero.”);

}

Int result = num1/num2;

System.out.println(result);

}

Public static void main (String[] args)

{

//declaring 2 integers

Int num1 = 10;

Int num2 = 0;

Try {

Display(num1, num2); //calls method

System.out.println(“Result : “);

} catch (ArithmeticException e) {

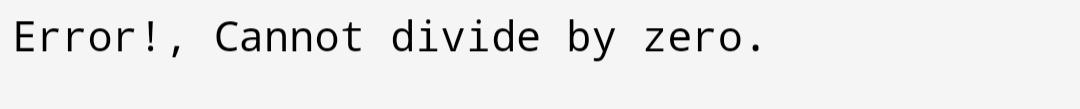
//handle exception

System.out.println(“Error!, “ + e.getMessage());

}

}

}

Output:-

5.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:-

package MyPackage;

public class InvalidAgeExcep

{

//defining custom exception

Public static class InvalidAgeException extends Exception {

Public InvalidAgeException(String message) {

Super(message);

}

}

//creating method to throw custom exception

Public static void checkAge(int age) throws InvalidAgeException {

If (age < 18) {

Throw new InvalidAgeException (“Invalid age, you are not eligible for driving.”);

} else {

System.out.println(“You are eligible for driving.”);

}

}

Public static void main (String[] args)

{

//declaring integer age

Int age = 17;

Try {

checkAge(age); //calls method

} catch(InvalidAgeException e) {

//handles exception

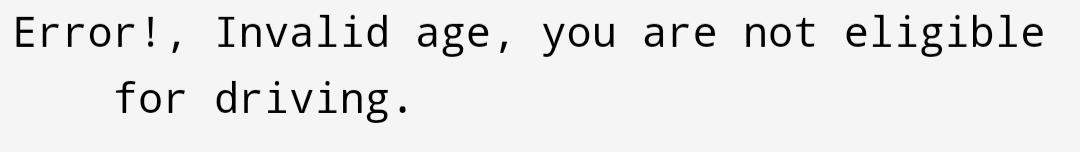
System.out.println(“Error!, “ + e.getMessage());

}

}

}

Output:-



6.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:-

package MyPackage;

public class EmailValidator {

//defining custom exception

Public static class InvalidEmailException extends Exception {

Public InvalidEmailException(String message) {

Super(message);

}

}

//creating method to throw custom exception

Public static void validateEmail(String email) throws InvalidEmailException {

If (!email.contains(“@”) || !email.contains(“.”)) {

Throw new InvalidEmailException(“Invalid email format: “ + email);

}

}

Public static void main(String[] args)

{

//declaring input emails

String email1 = [example@example.com](mailto:example@example.com);

String email2 = “example@example”;

Try {

validateEmail(email1); //calls method

System.out.println(“Valid email: “ + email1);

} catch (InvalidEmailException e) {

//handle exception if thrown

System.out.println(e.getMessage());

}

Try {

validateEmail(email2); //calls method

System.out.println(“Valid email: “ + email2);

} catch (InvalidEmailException e) {

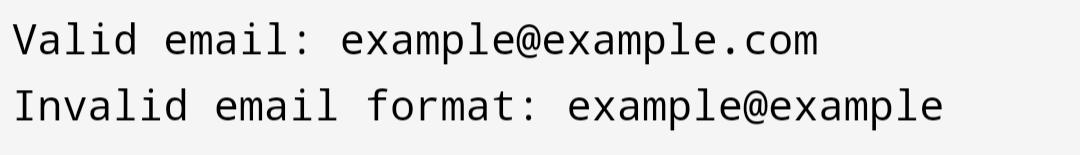
//handles exception

System.out.println(e.getMessage());

}

}

}

Output:-