

## LAB 8

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

### Program:

```
package lab;
public class ArrayOutOfBounds {

    public static void main(String[] args) {

        int arr[]=new int[8];

        System.out.println("hello guys");

        try {
            System.out.println(arr[10]); //checks the length of array element
        }
        catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Error: Array Index is out of bound"
                               + " " + e.getMessage());
            //prints an error message to the console if an
            ArrayIndexOutOfBoundsException is thrown.
        }
    }
}
```

### Output:

```
hello guys
Error: Array Index is out of bound Index 10 out of bounds for length 8
```

Q.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 lab;
public class DivisionByArithmetic {

    public static void main(String[] args) {
        try {
            int result = 10 / 0;    // performing division by zero

            System.out.println(result); //printing the result
        }
        catch (ArithmeticException e) {
            // Handles the exception if division by zero occurs

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

**Output:**

A screenshot of a terminal window with a black border. The text "Error: Division by zero is not allowed!" is displayed in a monospaced font. The text is color-coded: "Error:" is blue, "Division by zero" is red, and "is not allowed!" is blue. There is a vertical cursor line at the beginning of the line.

Q.3 . Write a java program that reads an integer input from the user and throws on IllegalArgumentException if the input is negative. Display an appropriate message when the exception is caught.

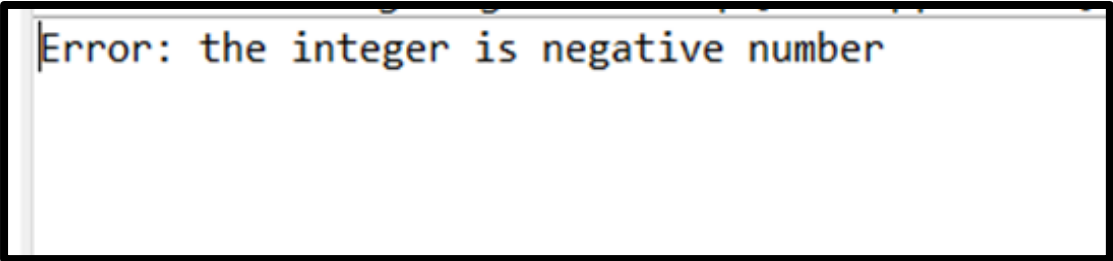
**Program:**

```
package lab;
public class IllegalArgumentExcep {
    public static void main(String[] args) {

        int a=-4; //user input for test //will throw an exception

        try {
            checkInteger(a);//this will check if the integer is positive or negative
            System.out.println("The integer is positive number");// This won't
execute if there's an exception
        }
        catch(IllegalArgumentException e) { //catches any
IllegalArgumentException thrown in the try block.
            System.out.println("Error: "+e.getMessage());
        }
    }
    public static void checkInteger(int num) throws
IllegalArgumentException{//declaring exception
        if(num < 0) { //throws IllegalArgumentException If the number is negative
            throw new IllegalArgumentException("the integer is negative
number");
        }
    }
}
```

**Output:**



```
Error: the integer is negative number
```

Q.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 lab;
public class CustomInvalidAgeExcep {
    static void checkage(int n) throws InvalidAgeException //declaring exception
    {
        if(n>18)
        {
            throw new InvalidAgeException("User is allowed....");
        }
        else {
            throw new InvalidAgeException("User is not allowed....");
        }
    }
    public static void main(String[] args) {
        int age=17; //input age for test
        try {
            checkage(age);
            // takes an age as input and checks if it's less than 18.

        }
        catch(InvalidAgeException e) {
            // catches an InvalidAgeException thrown in the try block.

            System.out.println("Error: "+e.getMessage());
            //printing the valid output
        }
    }
}
class InvalidAgeException extends Exception{
    //InvalidAgeException class extending Exception.
    public InvalidAgeException(String message) {
        super(message);
    }
}
```

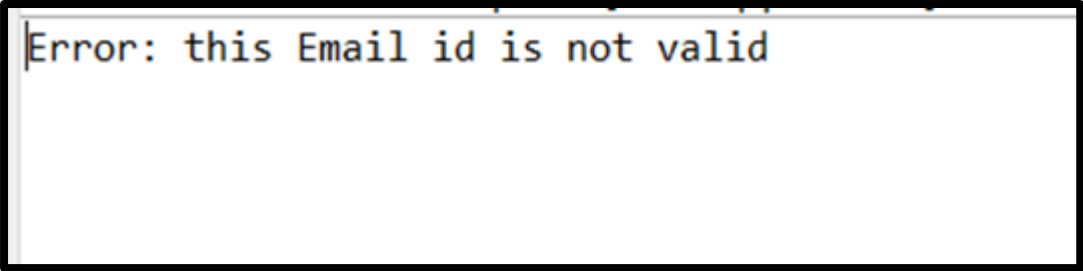
**Output:**

```
Error: User is not allowed....
```

Q.5. Write a java program that has a method to validate a user's email address. The method should throw a custom exception InvalidEmail Exception. If the email does not contain "@" and "." Handle the exception in the main method.

**Program:**

```
package lab;
public class EmailException {
    public static void main(String[] args) {
        String email="ABC.mail";//this will throw an exception
        try {
            validateEmail(email);
            System.out.println("this Email id is valid");//won't execute if there is
an exception
        }
        catch(InvalidEmailException e) { //catches InvalidEmailException thrown in
the try block.
            System.out.println("Error: "+e.getMessage());
        }
    }
    public static void validateEmail(String email) throws
InvalidEmailException{//declaring exception
        if(!email.contains("@")|| !email.contains(".")) { //checks if the email id
contains both "@" and "." characters.
            throw new InvalidEmailException("this Email id is not
valid");//prints output as not valid if email id not contains "@" and "."
        }
    }
}
class InvalidEmailException extends Exception{
    public InvalidEmailException(String message) {
        super(message);
    }
}
```

**Output:**

```
Error: this Email id is not valid
```