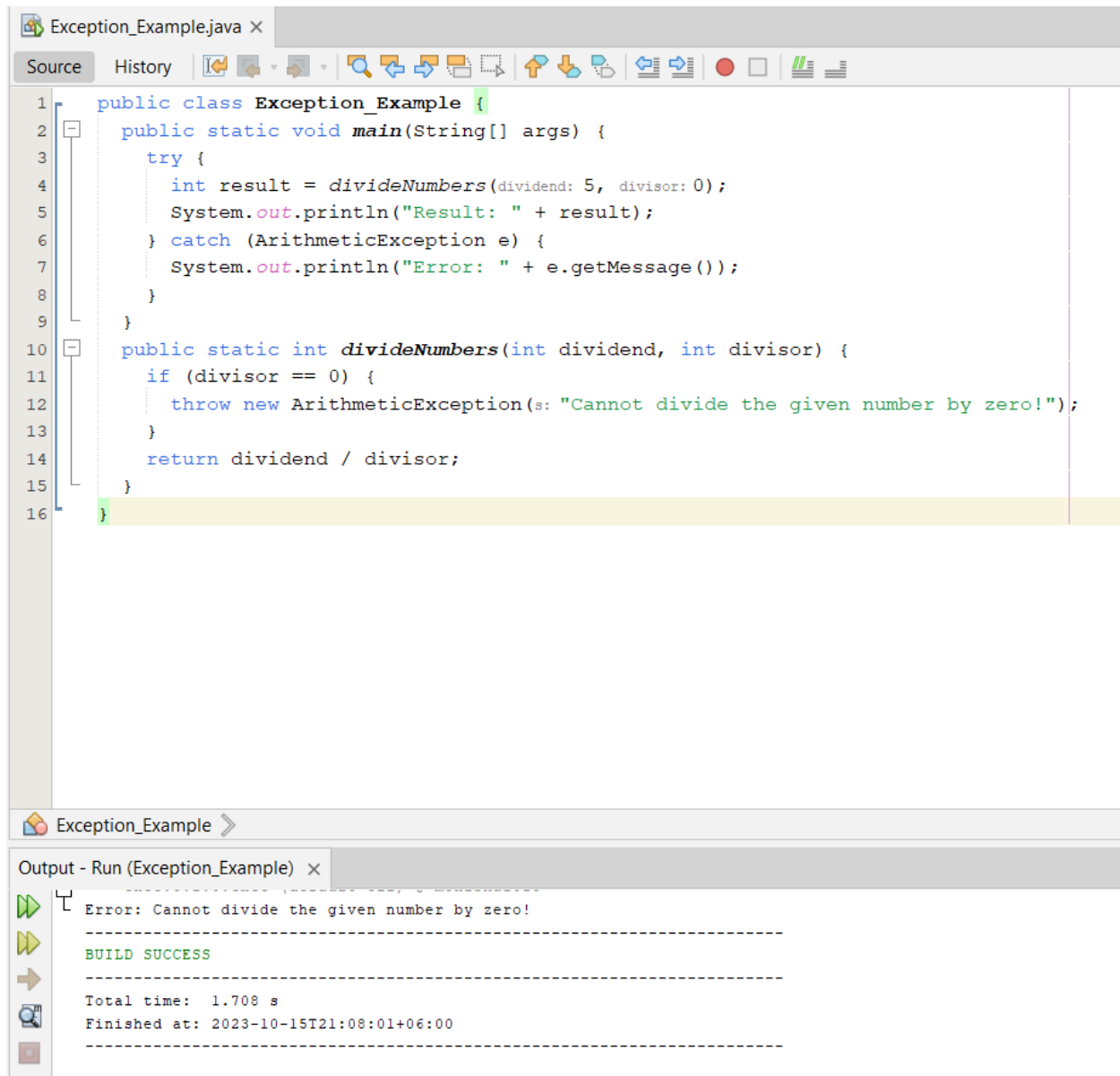


First Task: Write a java program that throws an exception and catch it using a try-catch block.



The screenshot shows an IDE with a Java file named 'Exception_Example.java'. The code defines a class 'Exception_Example' with a 'main' method and a 'divideNumbers' method. The 'main' method calls 'divideNumbers' with 5 as the dividend and 0 as the divisor. The 'divideNumbers' method checks if the divisor is zero; if so, it throws an 'ArithmeticException' with the message 'Cannot divide the given number by zero!'. The 'main' method uses a try-catch block to catch this exception and print an error message. The output window shows the execution result: 'Error: Cannot divide the given number by zero!', 'BUILD SUCCESS', and the total time of 1.708 s.

```
1 public class Exception_Example {
2     public static void main(String[] args) {
3         try {
4             int result = divideNumbers(dividend: 5, divisor: 0);
5             System.out.println("Result: " + result);
6         } catch (ArithmeticException e) {
7             System.out.println("Error: " + e.getMessage());
8         }
9     }
10    public static int divideNumbers(int dividend, int divisor) {
11        if (divisor == 0) {
12            throw new ArithmeticException(s: "Cannot divide the given number by zero!");
13        }
14        return dividend / divisor;
15    }
16 }
```

Output - Run (Exception_Example) x

```
Error: Cannot divide the given number by zero!
-----
BUILD SUCCESS
-----
Total time: 1.708 s
Finished at: 2023-10-15T21:08:01+06:00
-----
```

Discussion: In the above exercise, the divide Numbers method takes two integers as input and checks if the divisor is zero. If the divisor is zero, it throws an Arithmetic Exception with the message "Cannot divide the given number by zero!"

In the main method, we call the divide Numbers method with 10 as the dividend and 0 as the divisor. Since the divisor is zero, it throws an exception. Using a try-catch block, we catch the exception and print the error message "Error: Cannot divide by zero".

Second Task: Write a java program to create a method that takes an integer as a parameter and throws an exception if the number is odd.

```
1 public class Exception_OddNumber {
2     public static void main(String[] args) {
3         int n = 18;
4         trynumber(n);
5         n = 7;
6         trynumber(n);
7     }
8
9     public static void trynumber(int n) {
10        try {
11            checkEvenNumber(number:n);
12            System.out.println(n + " is even.");
13        } catch (IllegalArgumentException e) {
14            System.out.println("Error: " + e.getMessage());
15        }
16    }
17
18    public static void checkEvenNumber(int number) {
19        if (number % 2 != 0) {
20            throw new IllegalArgumentException(number + " is odd.");
21        }
22    }
23 }
24
```

Exception_OddNumber >

Output - Run (Exception_OddNumber) X

```
Compiling 5 source files to C:\Users\monis\OneDrive\Documents\NetBeansProjects\monishal813\target\classes
--- exec:3.1.0:exec (default-cli) @ monishal813 ---
18 is even.
Error: 7 is odd.
-----
BUILD SUCCESS
-----
Total time: 1.565 s
Finished at: 2023-10-15T21:35:43+06:00
-----
```

Discussion: In the main method, an integer n is declared and assigned 18. The try number method is then called with n as an argument. The try number method handles the exception. It contains a try-catch block. Inside the try block, the method check Even Number is called, passing n as an argument. If the number is even, the message "[number] is even." is printed. If an exception occurs in the try block, it is caught by the catch block, which handles Illegal Argument Exception. In this case, the error message "Error: [exception message]" is printed. After the first call to try number(n), the value of n is updated to 7, and the try number method is called again. This time, since 7 is an odd number, an exception is thrown.