Tutorial – 5

1. Find type of exception occurs in given code. Rewrite code that handle Exception using try and catch block.

```
public class ExceptionDemo1
{
   public static void main(String args[])
   {
      int number=50/0;
      System.out.println("number=" + number);
   }
}
```

Code:

```java
public class Main1 {
    public static void main(String args[]) {
        try {
            int number = 50 / 0;
            System.out.println("number=" + number);
        } catch (ArithmeticException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

Output Screenshot:

```
G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>javac Main1.java

G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>java Main1
Error: / by zero

G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>
```

2. Create array of numbers with size= 5. try to access element of index 10 and find is there any exception occurs? Write code to handle exception.

Code:

```
public class Main2 {
    public static void main(String[] args) {
        int[] numbers = new int[5];
        try {
            numbers[10] = 100; // Trying to access an element of index 10
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Error: " + e.getMessage()); // Handling the
exception
        }
    }
}
```

Output Screenshot:

```
G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>javac Main2.java

G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>java Main2
Error: Index 10 out of bounds for length 5

G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>
```

3. Find type of exception occurs in given code. Rewrite code that handle Exception using nested- try and catch block.

```
public class nestedtry
{
public static void main(String args[])
{

   int b=30/0;
   System.out.println("going to divide by 0");
   int a[]=new int[5];
   a[5]=10;
   System.out.println(a[5]);
}
}
```

Code:

```
public class Main3 {
    public static void main(String args[]) {
        try {
            int b = 30 / 0;
            System.out.println("going to divide by 0");
            try {
                int a[] = new int[5];
                a[5] = 10;
                System.out.println(a[5]);
            } catch (ArrayIndexOutOfBoundsException e) {
                System.out.println("Error: " + e.getMessage());
            }
        } catch (ArithmeticException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

Output Screenshot:

```
G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>javac Main3.java

G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>java Main3
Error: / by zero

G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>
```

4. Demonstrate the different usage of "final" in JAVA. give difference of all.

Code:

```
final class MyClass {
    final int MAX_VALUE = 100;

    public final void display(final int num) {
        final int value = 10;
        // value = 20;  // This will give an error as the variable is
declared final
        // num = 30;    // This will give an error as the parameter is
declared final
        System.out.println("MAX_VALUE: " + MAX_VALUE);
        System.out.println("num: " + num);
        System.out.println("value: " + value);
    }
}

public class Main4 {
    public static void main(String[] args) {
        MyClass obj = new MyClass();
        obj.display(50);
    }
}
```

Output Screenshot:

```
G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>javac Main4.java

G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>java Main4
MAX_VALUE: 100
num: 50
value: 10

G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>
```

5. Demonstrate the usage of finally block.

Code:

```
import java.util.Scanner;

public class Main5 {
    public static void main(String[] args) {
        Scanner scanner = null;

        try {
            scanner = new Scanner(System.in);
            System.out.print("Enter a number: ");
            int num = scanner.nextInt();
            System.out.println("Square of " + num + " is " + (num * num));
        } catch (Exception e) {
            System.out.println("Exception caught: " + e.getMessage());
        } finally {
            if (scanner != null) {
                scanner.close();
            }
            System.out.println("Finally block executed.");
        }
    }
}
```

Output Screenshot:

```
G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>javac Main5.java

G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>java Main5
Enter a number: 50
Square of 50 is 2500
Finally block executed.

G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>
```

6. Create user define exception name MyException and throw that user define exception using constructor and toString method.

Code:

```
class MyException extends Exception {
    private String message;

    public MyException(String message) {
        this.message = message;
    }

    @Override
    public String toString() {
        return "MyException: " + message;
    }
}

public class Main6 {
    public static void main(String[] args) {
        try {
            throw new MyException("This is a custom exception message");
        } catch (MyException e) {
            System.out.println(e);
        }
    }
}
```

Output Screenshot:

```
G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>javac Main6.java

G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>java Main6
MyException: This is a custom exception message

G:\OBJECT ORIENTED PROGRAMMING WITH JAVA\Tutorial\Experiment - 5>
```