TUGAS SESI 11
PEMROGRAMAN BERORIENTASI OBJEK

Nama : Arsyika Ma'rifatika Suryana Putri
NIM : 20210040063
Kelas : TI21E

Analisa Percobaan 5

```
public class Exception5 {

    public static void main(String[] args) {
        int bil=10;
        try
        {
            System.out.println(bil/0);
        }
        catch(ArithmeticException e)
        {
            System.out.println("Pesan error:  ");
            System.out.println(e.getMessage());
            System.out.println("Info stack erase");
            e.printStackTrace();
            e.printStackTrace(System.out);
        }
        catch(Exception e)
        {
            System.out.println("Ini menghandle error yang terjadi");
        }
    }

}
```

Pada blok kode try diatas memiliki kesalahan pada **bil / 0** yang dimana kesalahan ini akan dikirimkan ke catch tipe **ArithmaticException.**

Analisa percobaan 6

```
public class ThrowExample {

    static void demo(){
        NullPointerException t;
        t = new NullPointerException("Coba Throw");
        throw t;
        // Baris ini tidak lagi dikerjakan;
        //System.out.println("Ini tidak lagi dicetak");
    }
    Run | Debug
    public static void main(String[] args) {
        try{
            demo();
            System.out.println("Selesai");
        }
        catch(NullPointerException e){
            System.out.println("Ada pesan error: "+e);
        }
    }
}
```

Fungsi demo() memiliki kesalahan yang dibuat manual yaitu **NullPointerException,** ketika pada blok
try **NullPointerException** akan dilemparkan dan ditangkap oleh catch bertipe **NullPointerException.**

Analisa percobaan 7

```
public class ThrowExample2 {

    public static void main(String[] args) {
        try
        {
            throw new Exception("Here's my Exception");
        }
        catch(Exception e)
        {
            System.out.println("Caught Exception");
            System.out.println("e.getMessage():"+e.getMessage());
            System.out.println("e.toString():"+e.toString());
            System.out.println("e.printStackTrace():");
            e.printStackTrace();
        }
    }
}
```

Pada blok try akan dilemparkan **Exception** manual dan ditangkap pada catch.

Analisa Percobaan 8

```java
import java.io.*;

public class Test3 {

    public void methodA(){
        System.out.println("Method A");
    }

    public void methodB() throws IOException{
        System.out.println(20/0);
        System.out.println("Method B");
    }
}
```

```java
class Utama {
    Run | Debug
    public static void main(String[] args) throws IOException{
        Test3 o = new Test3();
        o.methodA();

        try {
            o.methodB();

        } catch (Exception e) {
            System.out.println("Error di method B");
            System.out.println(e.getMessage());
        }
        finally{
            System.out.println("selalu dicetak");
        }
    }
```

Pada fungsi MethodB memiliki sebuah kesalahan, ketika fungsi MethodB dipanggil dalam fungsi try maka kesalahan pada MethodB akan di tangkap catch.