# ใบงานการทดลองที่ 12

## เรื่อง การใช้งานคำสั่ง try catch และ throw exception

### 1. จุดประสงค์ทั่วไป

1.1. รู้และเข้าใจการใช้วัตถุ การทำหลายงานพร้อมกัน และการติดต่อระหว่างงาน

1.2. รู้และเข้าใจการจัดการกับความผิดปกติในการเขียนโปรแกรมเชิงวัตถุ

### 2. เครื่องมือและอุปกรณ์

เครื่องคอมพิวเตอร์1 เครื่อง ที่ติดตั้งโปรแกรม Eclipse

### 3. ทฤษฎีการทดลอง

3.1. Java Exception คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

เป็นวิธีการจัดการข้อผิดพลาดที่เกิดขึ้นในขณะที่โปรแกรมทำงาน

```
import java.util.Scanner;
public class TestException1{
 public static void main (String[] args) {
 Scanner reader = new Scanner(System.in);
 int x;
 System.out.print("Enter number: ");
 x = reader.nextInt();
 System.out.println("Your number is " + x); }}
```

3.2. คำสั่ง try มีลักษณะการทำงานอย่างไร?

```
Try { Jorhvsosd
Int answer = j / s ;}
```

3.3. คำสั่ง catch มีลักษณะการทำงานอย่างไร?

```
catch (InputMismatchException ex) {
 System.out.println("Exception occurred: " + ex);
 }
```

ตรวจจับ exception ที่จะเกิดขึ้นและจัดการกับมัน จากตัวอย่างด้านบน เราได้ปรับปรุงโปรแกรมให้สามารถ

จัดการกับ exception ได้

3.4. คำสั่ง finally มีลักษณะการทำงานอย่างไร?

เป็นคำสงวนใจภาษา Java ซึ่งเราสามารถใช้ได้กับการประกาศ ตัวแปร, method และ class ได้ด้วย โดยที่มันจะมีความหมายแตกต่างกันออกไปขึ้นอยู่กับว่าเราไปใช้ในการประกาศอะไร

```
finally {
 System.out.println("bar's finally"); }
```

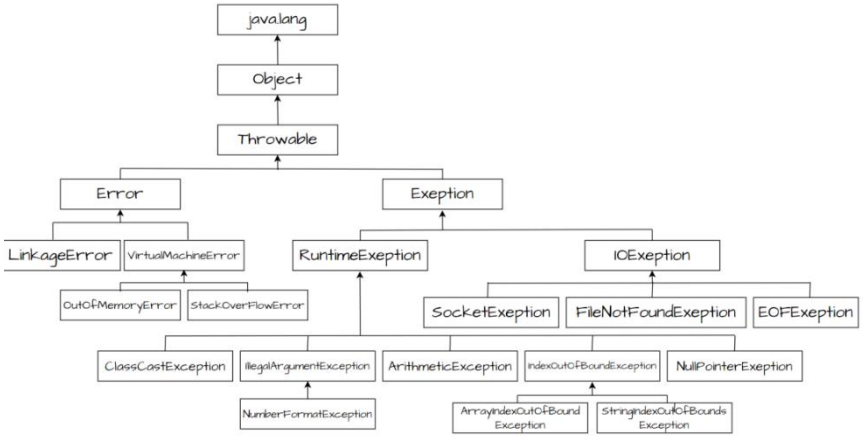
3.5. ลักษณะโครงสร้างของคำสั่ง try catch เป็นอย่างไร? อธิบายพร้อมยกตัวอย่างประกอบ

เป็นคำสั่งที่ใช้สำหรับกำหนดบล็อคเพื่อตรวจสอบและจัดการกับข้อผิดพลาดที่อาจจะเกิดขึ้นในโปรแกรม

```
package com.java.myapp;
public class MyClass {
public static void main(String[] args) {
 try {
 int x = 200 ;
 int y = 0 ;
 int z = x / y;
 System.out.println(" x / y = " + z) ;
 } catch(Exception e) {
 System.out.println("Error : " + e.getMessage());
 }}}
```

## 4. ลำดับขั้นการปฏิบัติการ

4.1. จากผังงานต่อไปนี้ จงเขียนโค้ดโปรแกรมเพื่อแสดงตัวอย่างการจัดการความผิดปกติของคลาสการจัดการสิ่งผิดปกติจนครบทุก คลาส (เน้นเฉพาะส่วนของ Error และ Exception)



ตัวอย่างโค้ด โปรแกรมการจัดการสิ่งผิดปกติในส่วนของ Error

```
try {
    check(5);
}catch(Error e) {
    System.out.println("Error");
}// End try..catch ---| Error


System.out.println("  This is LinkageError");


try {
    check(5);
}catch(VirtualMachineError e) {
    System.out.println("  This is VirtualMachineError");
}// End try..catch ---| VirtualMachineError

    try {
        int arrSize = 15;
        long memoryConsumed = 0;

        long[] memoryAllocated = null;
        for (int loop = 0; loop < Integer.MAX_VALUE; loop++) {
            memoryAllocated = new long[arrSize];
            memoryAllocated[0] = 0;
            memoryConsumed += arrSize * Long.SIZE;
            arrSize *= arrSize * 2;
            Thread.sleep(100);
        }
    }catch(OutOfMemoryError e) {
        System.out.println("   ----| This is OutOfMemoryError");
    }// End try..catch ---| OutOfMemoryError

    try {
        check(5);
    }catch(StackOverflowError e) {
        System.out.println("   ----| This is StackOverflowError");
    }// End try..catch ---| StackOverflowError
```

ตัวอย่างโค้ด โปรแกรมการจัดการสิ่งผิดปกติในส่วนของ Exeption

```
try {
    String a = "1234 ";
    Integer.parseInt(a);
}catch(Exception e) {
    System.out.println("Exception");
}// End try..catch ---| Exception

    try {
        int[] arrayin = {1,2,3};
        System.out.println(arrayin[10]);
    }catch(RuntimeException e) {
        System.out.println("  This is RuntimeException");
    }// End try..catch ---| RuntimeException

        try {
            String objStr = "123";
            BigDecimal result = addOne(objStr);
            System.out.println(result);
        }catch( ClassCastException e ) {
            System.out.println("   ----| This is ClassCastException");
        } // End try..catch ---| ClassCastException

        try {
            a();
        } catch (IllegalStateException e) {
            System.out.println("   ----|_ This is IllegalStateException");
        }// End try..catch ---| IllegalStateException

        try {
            String a = "1234 ";
            Integer.parseInt(a);
        }catch(NumberFormatException e) {
            System.out.println("        |----> This is NumberFormatException");
        }// End try..catch ---| NumberFormatException

        try {
            int a = 5;
            int b = 0;
            int ans = a / b ;
        }catch(ArithmeticException e) {
            System.out.println("   ----| This is ArithmeticException");
        }// End try..catch ---| ArithmeticException
```

```java
try {
    int[] arrayin = {1,2,3};
    System.out.println(arrayin[10]);
}catch(IndexOutOfBoundsException e) {
    System.out.println("    ----|_ This is IndexOutOfBoundsException");
}// End try..catch ---| IndexOutOfBoundsException

try {
    int[] arrayin = {1,2,3};
    System.out.println(arrayin[10]);
}catch(ArrayIndexOutOfBoundsException e) {
    System.out.println("        |----> This is ArrayIndexOutOfBoundsException");
}// End try..catch ---| ArrayIndexOutOfBoundsException

try {
    String st = "arm";
    System.out.println(st.charAt(4));
}catch(StringIndexOutOfBoundsException e) {
    System.out.println("        |----> This is StringIndexOutOfBoundsException");
}// End try..catch ---| StringIndexOutOfBoundsException

try {
    Path file = null;
    Files.delete(file);
} catch (NullPointerException e) {
        System.out.println("    ----| This is NullPointerException");
}// End try..catch ---| NullPointerException


 try {
    FileInputStream f = new FileInputStream("code.txt");
    }catch(IOException e) {
        System.out.println("  This is IOException");
    }// End try..catch ---| IOException

            try {
                createConnection();
                System.out.println("Second test");
                initiateIO();
            }catch(SocketException e) {
                    System.out.println("    ----| This is SocketException");
            }// End try..catch ---| SocketException


            try {
            FileInputStream f = new FileInputStream("code.txt");
            }catch(FileNotFoundException e) {
                System.out.println("    ----| This is FileNotFoundException");
            }// End try..catch ---| FileNotFoundException

            try {
            DataInputStream dis = new DataInputStream(new FileInputStream("C:\\data.txt"));
            while (true) {
                char ch ;
                ch = dis.readChar();
                System.out.println(ch);
            }
            }catch(EOFException e) {
                System.out.println("    ----| This is EOFException");
            }// End try..catch ---| EOFException
    System.out.println("_____________________________________________________");
```

**5. สรุปผลการปฏิบัติการ**

การใช้ try เป็นส่วนหนึ่งที่ทำให้โปรแกรม exception ขึ้นได้ และสามารถทำ exception ให้ตรวจจับข้อมูลที่มีส่วนผิดพลาดได้

**6. คำถามท้ายการทดลอง**

6.1. เพราะเหตุใดการใช้catch( Exception e ) ; จึงไม่เหมาะสมกับการจัดการสิ่งผิดปกติที่ดีที่สุด

เพราะตัวโปรแกรมที่สร้างมามันอาจ ตรวจจับส่วนของ error ได้ ว่ามันจะ error ตรงไหน

6.2. การจัดการสิ่งผิดปกติจากการตัวเลขต่างๆ ด้วยเลขศูนย์ควรเลือกใช้วิธีใด?

Catch( ArithmeticException e ){ }

6.3. การจัดการสิ่งผิดปกติจากการเรียกใช้งาน Element เกินขนาดของอาเรย์ควรเลือกใช้วิธีใด?

Catch( ArrayindexOutOfBoundsException e ){ }