

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

- Exception คือการที่โปรแกรมพยายามจะทำงานบางอย่าง แต่เกิดข้อผิดพลาดขึ้น แล้วโปรแกรมไม่สามารถจัดการข้อผิดพลาดนั้นได้ ซึ่งทำให้เกิด exception ขึ้น และส่งผลทำให้โปรแกรมหยุดทำงาน Exception เกิดขึ้นในขณะที่โปรแกรมทำงาน ยกตัวอย่างเช่น โปรแกรมกำลังจะเปิดไฟล์ขึ้นมา แต่ไฟล์ที่ต้องการไม่มีอยู่ เป็นต้น รูปแบบการใช้ exception ในภาษา Java

```
try {  
    // try to do something  
} catch (Exception1 ex1) {  
    // handle for exception 1  
}  
  
...  
  
} catch (ExceptionN exN) {  
    // handle for exception N  
} finally {  
    // Always proceed this block whether  
    // an exception is thrown or not  
}
```

3.2. คำสั่ง try มีลักษณะการทำงานอย่างไร?

- เป็นส่วนของโปรแกรมที่อาจจะทำให้เกิด exception ขึ้น

3.3. คำสั่ง catch มีลักษณะการทำงานอย่างไร?

- ในแต่ละ catch บล็อกเป็นการจัดการกับ exception แต่ละแบบ

3.4. คำสั่ง finally มีลักษณะการทำงานอย่างไร?

- โปรแกรมจะเข้ามาท งานเสมอไม่ว่าจะเกิด exception ในบล็อกของค าสั่ง try หรือไม่ก็ตาม

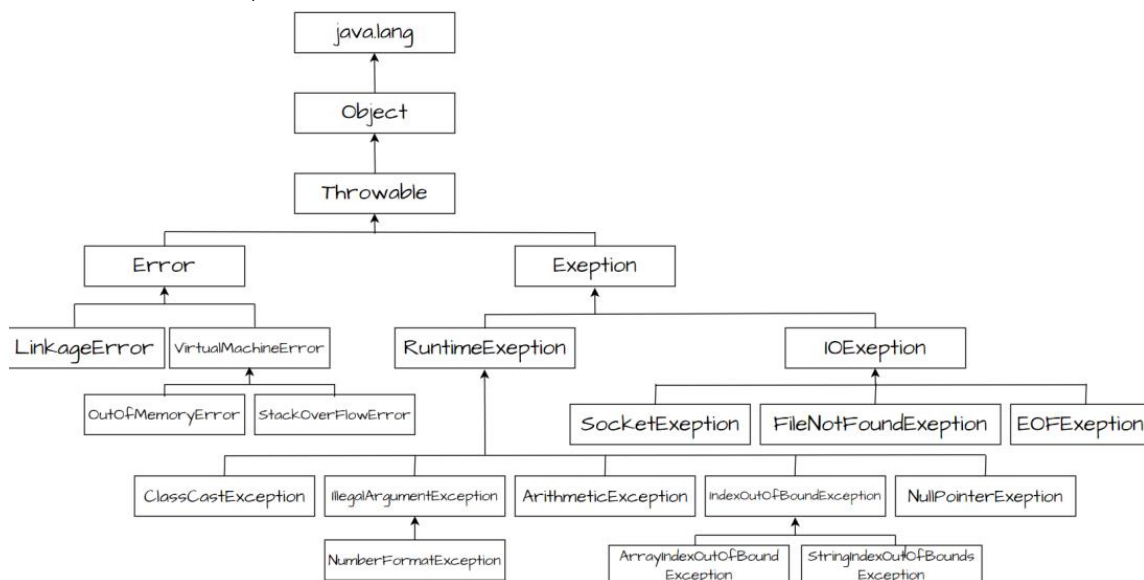
3.5. ลักษณะโครงสร้างของคำสั่ง try catch เป็นอย่างไร? อธิบายพร้อมยกตัวอย่างประกอบ

```
try {  
    // try to do something  
} catch (Exception1 ex1) {  
    // handle for exception 1  
}
```

- ในบล็อกคำสั่งของ try เป็นส่วนของโปรแกรมที่อาจจะทำให้เกิด exception ขึ้น และในแต่ละ catch บล็อกเป็นการจัดการกับ exception แต่ละแบบ และบล็อก

4. ลำดับชั้นการปฏิบัติการ

4.1. จากผังงานต่อไปนี้ จงเขียนโค้ดโปรแกรมเพื่อแสดงตัวอย่างการจัดการความผิดปกติของคลาสการจัดการสิ่งผิดปกติจนครบทุก คลาส (เน้นเฉพาะส่วนของ Error และ Exception)



ตัวอย่างโค้ด โปรแกรมการจัดการสิ่งผิด ปกในส่วนของ Error

```
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
```

ตัวอย่างโค้ดโปรแกรมการจัดการสิ่ง ผิด ปกในส่วนของ Exeption

```
    public static BigDecimal addOne(Object obj){  
        BigDecimal a =(BigDecimal)obj;  
        return a.add(BigDecimal.ONE);  
    }  
  
    public static void a0{ b0; }  
    public static void b0{ c0; }  
    public static void c0{  
        throw new IllegalStateException("Just a test");  
    }  
  
    private static void createConnection() throws Exception, IOException {  
  
        String host = null;  
        int port = 0;  
        Socket socket = new Socket(host, port);  
  
    }  
  
    private static void initiateIO() throws IOException {  
  
        Socket socket = null;  
        PrintWriter outbound = new PrintWriter(socket.getOutputStream(), true);  
  
    }  
  
    public static void check(int i){  
  
        if(i == 0)  
            return;  
        else{  
            check(i++);  
        }  
    }  
  
    public static void main( String[] args ) throws Exception {  
  
        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{
        a0;
    }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

    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("_____");

    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

}

```

5. สรุปผลการปฏิบัติการ

- Error แต่ละอย่างสามารถบ่งบอกได้ถึง error ของข้อมูลนั้นๆ ได้

6. คำถามท้ายการทดลอง

- 6.1. เพราะเหตุใดการใช้ catch(Exception e) ; จึงไม่เหมาะสมกับการจัดการสิ่งผิดปกติที่ดีที่สุด
 - เพราะ catch(Exception e) ; มันกว้างเกินไป ไม่สามารถจะรู้ error ได้
- 6.2. การจัดการสิ่งผิดปกติจากการตัวเลขต่างๆ ด้วยเลขศูนย์ควรเลือกใช้วิธีใด?
 - ใช้ ArithmeticException
- 6.3. การจัดการสิ่งผิดปกติจากการเรียกใช้งาน Element เกินขนาดของอาเรย์ควรเลือกใช้วิธีใด?
 - ใช้ indexoutofboundsException