

Exception Handling



ประเภทของ Error

1. Syntax errors

- เขียนโปรแกรมที่ผิดไวยากรณ์ของหลักภาษา
- เกิดขึ้นในตอน compile

2. Runtime error

- เกิดขึ้นในขณะที่โปรแกรมรัน
- อาจทำให้โปรแกรม crash ได้

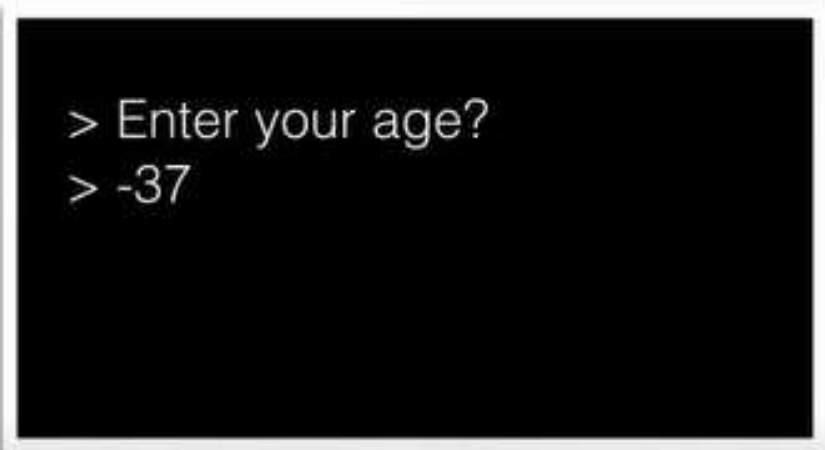
3. Bugs

- โปรแกรมทำงานแต่ไม่ได้ผลลัพธ์ตามที่เราคาดหวัง

Runtime Error

Runtime error คือ error ที่เกิดขึ้นในขณะที่โปรแกรมกำลังทำงาน

ตัวอย่าง Runtime Error



```
> Enter your age?  
> -37
```

พยายามเปิดไฟล์ที่ไม่มี



```
File file = new File("test.txt");  
Scanner fileScanner = new Scanner(file);
```

การเขียนโปรแกรมที่ดีควรตรวจสอบข้อผิดพลาดที่อาจเกิดขึ้น

```
File file = new File("test.txt");  
if(file.exists()) {  
    Scanner fileScanner = new Scanner(file);  
}
```

Exception คือ?

- คือ ข้อผิดพลาดใดๆ ที่ทำให้โปรแกรมหยุดการทำงานจากการทำงานปกติ
- เมื่อเกิด exception โปรแกรมจะหยุดทำงานและแสดงข้อความแจ้งข้อผิดพลาด

```
Exception in thread "main" java.io.FileNotFoundException: G:\886201 OOP\Lab OK\Solution lab11\tex.txt (The system cannot find the file specified)
    at java.io.FileInputStream.open0(Native Method)
    at java.io.FileInputStream.open(Unknown Source)
    at java.io.FileInputStream.<init>(Unknown Source)
    at java.util.Scanner.<init>(Unknown Source)
    at ex3.Test.main(Test.java:12)
```

Exception เกิดเมื่อไหร่

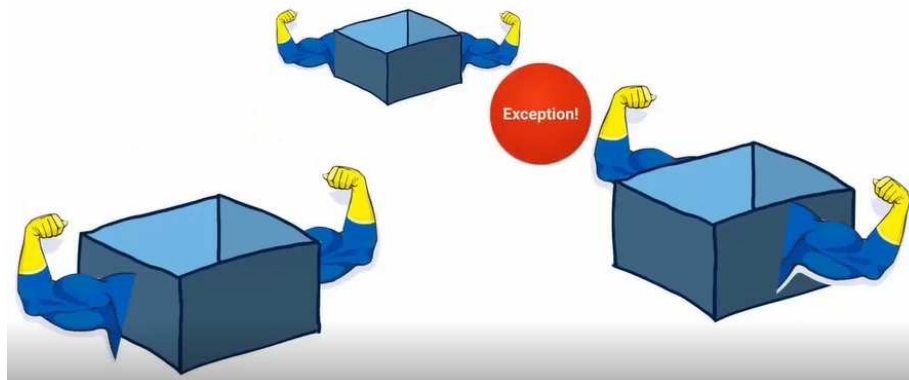
- สามารถเกิดได้ทั้งขณะ runtime (runtime exceptions) และ compile-time (compile-time exceptions)
- exception ที่เกิดขณะโปรแกรมทำงาน
 - ผู้เขียนโปรแกรมต้องเขียนคำสั่งเพื่อจัดการข้อผิดพลาดที่อาจจะเกิดขึ้น เพื่อให้โปรแกรมดำเนินต่อไป
 - ตัวอย่างเช่น
 - ArithmeticException
 - ArrayIndexOutOfBoundsException
 - NullPointerException

ต้องทำอะไรเมื่อเกิด Exception

- เมื่อเกิดข้อผิดพลาดขึ้น จะมีการสร้าง exception object ที่สัมพันธ์กับ exception class

```
public void openFile(String fileName) throws FileNotFoundException {  
    // Opens a file here  
}
```

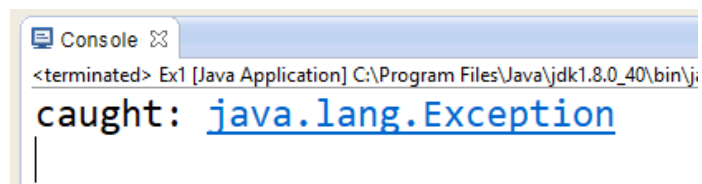
- จากนั้นโยน (throw) exception นี้ให้ใครก็ตามที่รอดักจับอยู่
 - เมื่อมีผู้จับ exception ได้ สามารถเลือกได้ว่าจะโยนให้คนอื่นต่อ หรือ จะจัดการ exception นั้น



throw Statements

- ใช้สำหรับโยน exception ในตำแหน่งที่ต้องการออกมา
- throw จะตามด้วย instance ของ exception ที่จะโยนออกมา
 - หาก instance นั้นมีอยู่ก่อน ก็สามารถโยนออกมาได้เลย
 - ถ้าไม่มีต้องทำการสร้างขึ้นมาก่อนด้วยคำสั่ง new ก่อน
- คำสั่ง throw อาจอยู่ใน try block ที่มีการดักจับ exception หรืออยู่ใน method ที่มีการระบุว่าส่ง exception นั้นออกมา

```
public class Ex1 {  
    public static void div(int x, int y) {  
        try {  
            if (y == 0)  
                throw new Exception();  
            System.out.println("div = " + x / y);  
        } catch (Exception e) {  
            System.out.println("caught: " + e);  
        }  
    }  
  
    public static void main(String[] args) {  
        div(1, 0);  
    }  
}
```



Console

<terminated> Ex1 [Java Application] C:\Program Files\Java\jdk1.8.0_40\bin\j
caught: [java.lang.Exception](#)

Method ที่มีการ throws exception

- คาดหมายว่า method นี้จะมีการ throws exception ออกมา
- ใช้คำสั่ง throws ที่หัวเมธอดหลังวงเล็บปิดของรายการพารามิเตอร์
- ระบุคำว่า throws ตามด้วยคลาสของ exception ที่จะโยนออกมา

```
public void openFile(String fileName) throws FileNotFoundException {  
    // Opens a file here  
}
```

ตัวอย่าง

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class Ex1 {
    public static void main(String[] a) throws IOException {
        BufferedReader stdin = new BufferedReader
            (new InputStreamReader(System.in));

        String inData;
        int num;
        System.out.println("Enter an integer:");
        inData = stdin.readLine();
        num = Integer.parseInt(inData); // convert inData to int
        System.out.println("The square of " + inData + " is " + num * num);
    }
}
```

Exception handing

- ในภาษาจาวามีรูปแบบการใช้งาน ดังนี้

```
try{  
    <statements>;  
} catch (Throwable1 t) {  
    <statements>;  
} catch (Throwable2 t) {  
    <statements>;  
} catch (Throwable3 t) {  
    <statements>;  
}
```

ตัวอย่าง

```
try{
    File file = new File("somefile.txt");
} catch (FileNotFoundException e){
    System.out.println("File missing!");
}
```

จัดการ exception
โดยแสดงข้อความให้ user ทราบ

```
try{
    File file = new File("somefile.txt");
} catch (FileNotFoundException e){
    throw e;
}
```

ดักจับได้แล้วโยน exception ให้
ผู้ที่เรียก method นี้ไปจัดการต่อ

ตัวอย่าง 2

```
void main(String[] args) throws Exception {  
    openFile();  
}
```

```
void openFile() throws Exception {  
    File file = new File("somefile.txt");  
}
```

เมื่อเกิด exception ในเมธอด openFile() จะโยน exception ให้ main
main ก็โยนต่อ ผลก็คือ โปรแกรมหยุดทำงานแล้วแสดง exception msg

การจับ exceptions

- ภายในบล็อก catch เราสามารถเลือกได้ว่าจะดักจับ exception ประเภทไหน
- แล้วเมื่อจับได้จะดำเนินการอย่างไร
 - เช่น ข้อความผิดพลาดให้ผู้ใช้ทราบ

```
try{
    openFile("somefile.txt") ;
} catch(FileNotFoundException exception){
    // Handle the situation by letting the user know what happened
    System.out.println("Cannot find that file");
}
```


การจับ exceptions (2)

- จับได้แล้วไม่จัดการอะไร โยนต่อ

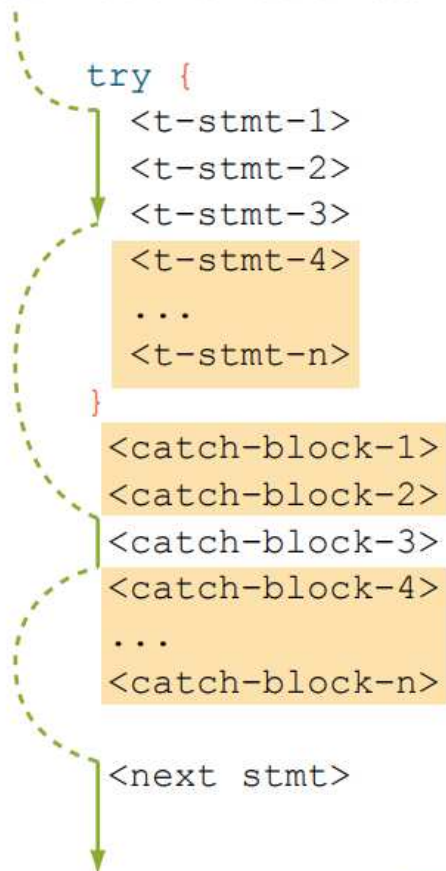
```
try{
    openFile("somefile.txt");
} catch(FileNotFoundException exception){
    // Running away from the responsibility
    throw exception;
}
```

การทำงานของ Exception Handling

- คำสั่งที่อยู่ใน try block จะทำงานปกติ
- หากทำงานจนจบโดยไม่มีข้อผิดพลาดเกิดขึ้น ก็จะทำงานหลังคำสั่ง catch block สุดท้าย
- ถ้ามี exception เกิดขึ้นใน try block โปรแกรมจะหยุดทำงานที่บรรทัดนั้น แล้วสร้าง instance ของ exception แล้ว throws ออกไป
- ถ้าคำสั่งที่มีความผิดพลาดนั้นมี catch block ที่มีค่าพารามิเตอร์ตรงกับ exception ที่เกิดขึ้น คำสั่งใน catch block นั้นก็จะทำงาน เมื่อทำเสร็จก็ไปทำงานคำสั่งหลัง catch block สุดท้าย

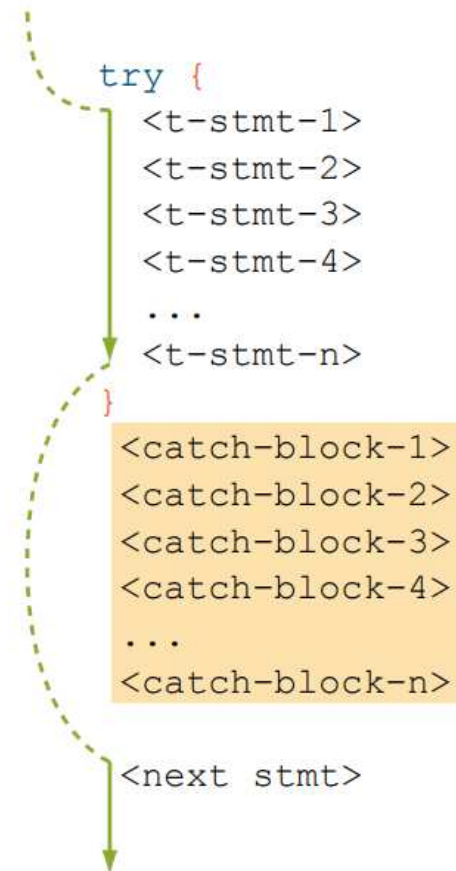
Exception

Assume **<t-stmt-3>** throws an exception and **<catch-block-3>** is the matching **catch** block.



Skipped portion

No exception



การทำงานของ Exception Handling (ต่อ)

- ถ้า Exception ที่ถูก throws ออกมาจากคำสั่ง try block ไม่มี catch block ดักจับ
 - Exception ที่ถูกส่งออกมาจาก method ที่เกิดความผิดพลาดจะถูกส่งไปยัง method ผู้เรียก
 - ถ้า method ที่เรียกใช้มีการจับ exception กระบวนการจัดการก็เสร็จสิ้น
 - หาก method นั้นไม่มีการจับ exception มันจะส่งกับเรื่อยๆ จนออกจาก main (เรียกว่า propagation)
- เมื่อออกจาก main จะถูก java จัดการดังนี้
 - แสดง exception
 - พิมพ์ activation stack เพื่อให้รู้จุดกำเนิดและเส้นทาง propagation
 - หยุดการทำงานของ JVM

เขียน code แบบไม่มี exception

```
public class TestArithmetic {  
  
    public static void main(String[] args) {  
        int a = 30, b =0;  
        int c = a/b;  
        System.out.println("Result = "+c);  
    }  
}
```

```
Exception in thread "main" java.lang.ArithmeticException: / by zero  
    at TestArithmetic.main(TestArithmetic.java:6)
```

แบบฝึกหัด

- ให้นิสิตเขียนโปรแกรมเพื่อดักจับ exception ที่ชื่อว่า ArithmeticException
- จากนั้นจัดการ exception นี้โดยแสดงข้อความว่า “Can’t divide a number by 0”

เขียน code แบบไม่มี exception

```
public class TestNullPointer {  
  
    public static void main(String[] args) {  
        String a = null;  
        System.out.println(a.charAt(0));  
    }  
}
```

```
Exception in thread "main" java.lang.NullPointerException  
    at TestNullPointer.main(TestNullPointer.java:6)
```

|

แบบฝึกหัด

- ให้นิสิตเขียนโปรแกรมเพื่อดักจับ exception ที่ชื่อว่า NullPointerException
- จากนั้นจัดการ exception นี้โดยแสดงข้อความว่า “You create a NullPointerException”

เขียน code แบบไม่มี exception

```
public class TestStringIndexOutOfBounds {  
  
    public static void main(String[] args) {  
        String a= "This is a book";  
        for(int i=a.length(); i>=0; i--) {  
            System.out.println(a.charAt(i));  
        }  
    }  
}
```

```
Exception in thread "main" java.lang.StringIndexOutOfBoundsException: String index out of range: 14  
    at java.lang.String.charAt(Unknown Source)  
    at TestStringIndexOutOfBounds.main(TestStringIndexOutOfBounds.java:7)
```

แบบฝึกหัด

- ให้นิสิตเขียนโปรแกรมเพื่อดักจับ exception ที่ชื่อว่า `StringIndexOutOfBoundsException`
- จากนั้นจัดการ exception นี้โดยแสดงข้อความว่า “You create a `StringIndexOutOfBoundsException`”

เขียน code แบบไม่มี exception

```
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;

public class TestFileNotFound {

    public static void main(String[] args) throws FileNotFoundException {
        File file = new File("E:/file.txt");
        Scanner input = new Scanner(file);
    }
}
```

```
Exception in thread "main" java.io.FileNotFoundException: E:\file.txt (The system cannot find the path specified)
    at java.io.FileInputStream.open0(Native Method)
    at java.io.FileInputStream.open(Unknown Source)
    at java.io.FileInputStream.<init>(Unknown Source)
    at java.util.Scanner.<init>(Unknown Source)
    at TestFileNotFound.main(TestFileNotFound.java:9)
```

แบบฝึกหัด

- ให้นิสิตเขียนโปรแกรมเพื่อดักจับ exception ที่ชื่อว่า FileNotFoundException
- จากนั้นจัดการ exception นี้โดยแสดงข้อความว่า “File does not exist”

เขียน code แบบไม่มี exception

```
import java.util.Scanner;

public class TestInputMissMatch {

    public static void main(String[] args) {
        Scanner kb = new Scanner(System.in);
        int num = kb.nextInt();

        System.out.println("num = " + num);
    }
}
```

test

```
Exception in thread "main" java.util.InputMismatchException
    at java.util.Scanner.throwFor(Unknown Source)
    at java.util.Scanner.next(Unknown Source)
    at java.util.Scanner.nextInt(Unknown Source)
    at java.util.Scanner.nextInt(Unknown Source)
    at TestInputMissMatch.main(TestInputMissMatch.java:7)
```

แบบฝึกหัด

- ให้นิสิตเขียนโปรแกรมเพื่อดักจับ exception ที่ชื่อว่า InputMismatchException
- จากนั้นจัดการ exception นี้โดยแสดงข้อความว่า “Your input invalid”

เขียน code แบบไม่มี exception

```
public class TestArrayIndexOutOfBounds {  
  
    public static void main(String[] args) {  
        int a[] = new int[5];  
        a[6] = 9;  
    }  
}
```

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 6  
    at TestArrayIndexOutOfBounds.main(TestArrayIndexOutOfBounds.java:6)
```

แบบฝึกหัด

- ให้นิสิตเขียนโปรแกรมเพื่อดักจับ exception ที่ชื่อว่า `ArrayIndexOutOfBoundsException`
- จากนั้นจัดการ exception นี้โดยแสดงข้อความว่า “Array index out of bounds”

Multiple catch statements

- เนื่องจากใน try block อาจจะมีมากกว่า 1 คำสั่ง และ เมธอดอาจโยน exception ออกมา มากกว่า 1 ประเภท
- เราสามารถสร้างการดักจับประเภทต่างๆของ exception ไปพร้อมกันในครั้งเดียว
 - สามารถสร้าง exception type ได้มากเท่าที่ต้องการ จนแน่ใจว่าครอบคลุม exception ที่อาจจะเกิดขึ้นและ ถูกโยนออกมาจาก try block

```
try{
    openFile("somefile.txt");
    array[index]++;
} catch(FileNotFoundException exception){
    // Handle all the possible file-not-found-related issues here
} catch(IndexOutOfBoundsException exception){
    // Handle all the possible index-out-of-bounds-related issues here
}
```

Catching all exceptions

- อีกแนวทางหนึ่งในการจับทุก exception type คือใช้ type Exception
 - ไม่ว่าจะเป็น exception อะไรที่ถูกโยนออกมาจาก try-catch block จะถูกจับด้วยจัดการด้วย catch statement นี้

```
try{  
    openFile("somefile.txt");  
    array[index]++;  
} catch(Exception exception){  
    // Handle all the possible exceptions here  
}
```

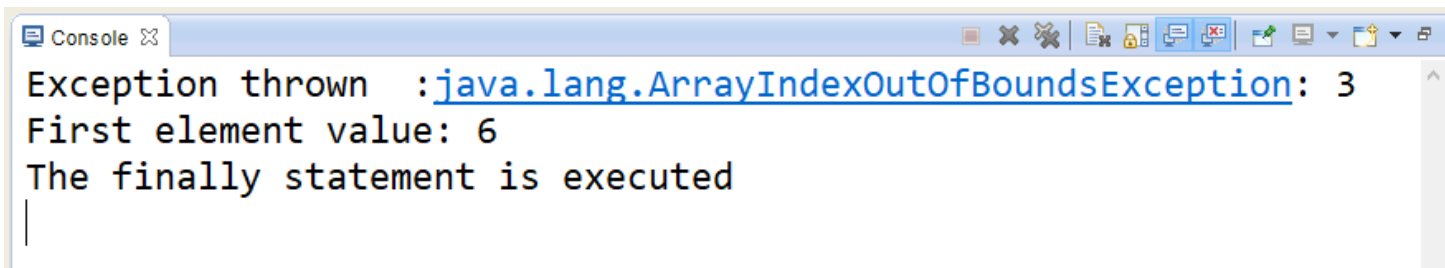
finally block

- เป็นบล็อกที่มีหรือไม่มีก็ได้
- ถ้ามี มีได้เพียง 1 block เป็นบล็อกที่ทำงานเสมอ ไม่ว่าโปรแกรมจะผ่าน try หรือ catch block หรือไม่

```
try {  
    <statements>;  
}catch (<parameter>) {  
    <statements>;  
}finally {  
    <statements>;  
}
```

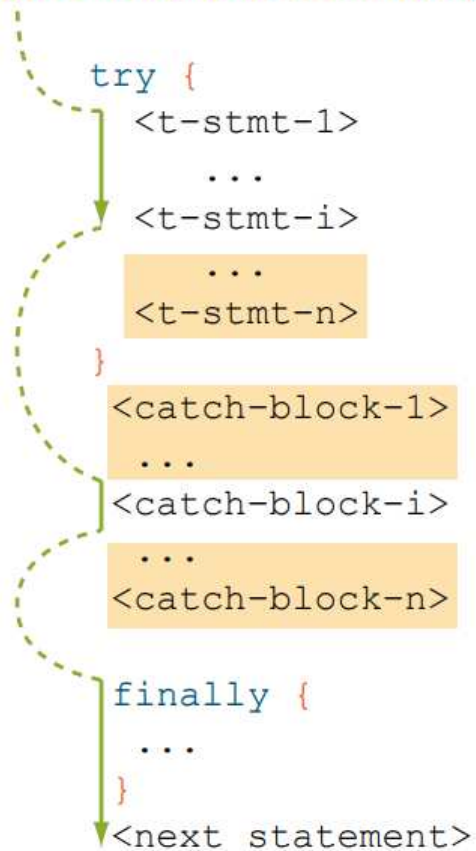
ตัวอย่าง

```
public class Ex1 {  
    public static void main(String args[]) {  
        int a[] = new int[2];  
        try {  
            System.out.println("Access element three :" + a[3]);  
        } catch (ArrayIndexOutOfBoundsException e) {  
            System.out.println("Exception thrown :" + e);  
        } finally {  
            a[0] = 6;  
            System.out.println("First element value: " + a[0]);  
            System.out.println("The finally statement is executed");  
        }  
    }  
}
```



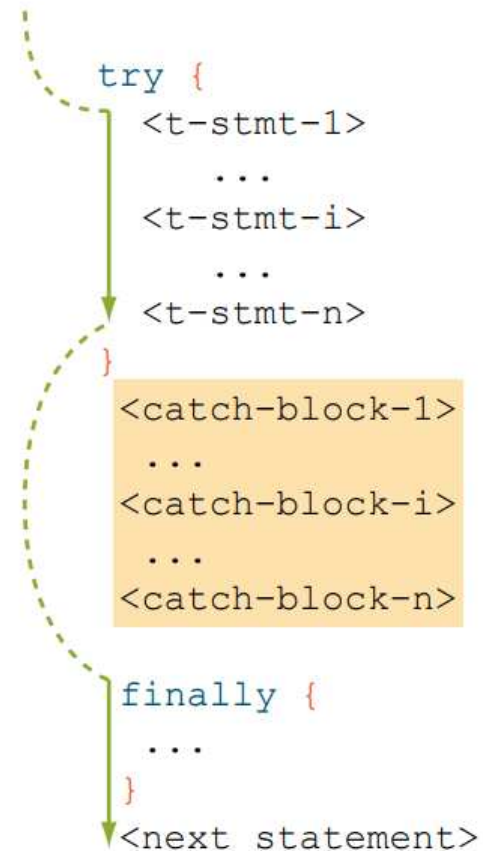
Exception

Assume **<t-stmt-i>** throws an exception and **<catch-block-i>** is the matching **catch** block.



Skipped portion

No exception



แบบฝึกหัด

```
class Test extends Exception { }

class Main {
    public static void main(String args[]) {
        try {
            throw new Test();
        }
        catch(Test t) {
            System.out.println("Got the Test Exception");
        }
        finally {
            System.out.println("Inside finally block ");
        }
    }
}
```

A

Got the Test Exception
Inside finally block

B

Got the Test Exception

C

Inside finally block

D

Compiler Error

แบบฝึกหัด 2

Output of following Java program?

```
class Main {  
    public static void main(String args[]) {  
        int x = 0;  
        int y = 10;  
        int z = y/x;  
    }  
}
```

A

Compiler Error

B

Compiles and runs fine

C

Compiles fine but throws ArithmeticException exception

แบบฝึกหัด 3

```
class Test
{
    public static void main (String[] args)
    {
        try
        {
            int a = 0;
            System.out.println ("a = " + a + "\n");
            int b = 20 / a;
            System.out.println ("b = " + b);
        }

        catch(ArithmeticException e)
        {
            System.out.println ("Divide by zero error");
        }

        finally
        {
            System.out.println ("inside the finally block");
        }
    }
}
```

A

Compile error

B

Divide by zero error

C

a = 0
Divide by zero error
inside the finally block

D

a = 0

E

inside the finally block

แบบฝึกหัด 4

```
class Test
{
    public static void main(String[] args)
    {
        try
        {
            int a[]= {1, 2, 3, 4};
            for (int i = 1; i <= 4; i++)
            {
                System.out.println ("a[" + i + "]= " + a[i] + "\n");
            }
        }

        catch (Exception e)
        {
            System.out.println ("error = " + e);
        }

        catch (ArrayIndexOutOfBoundsException e)
        {
            System.out.println ("ArrayIndexOutOfBoundsException");
        }
    }
}
```

A

Compiler error

B

Run time error

C

ArrayIndexOutOfBoundsException

D

Error Code is printed

E



Array is printed

เฉลย

```
1
2 public class Test {
3
4     public static void main(String[] args) {
5         try {
6             int a[] = {1,2,3,4};
7             for(int i=1;i<=4;i++) {
8                 System.out.println("a[" + i + "]= " + a[i]);
9             }
10        }catch (Exception e) {
11            System.out.println("error = " + e);
12        }catch (ArrayIndexOutOfBoundsException e) {
13            Syst
14        }
15    }
16
17 }
18
```

Unreachable catch block for ArrayIndexOutOfBoundsException. It is already handled by the catch block for Exception

2 quick fixes available:

-  [Remove catch clause](#)
-  [Replace catch clause with throws](#)

Press 'F2' for focus

แบบฝึกหัด 5

| | |
|---|--------|
| A | abdef |
| B | abdec |
| C | abdefc |

```
class Test
{
    String str = "a";

    void A()
    {
        try
        {
            str += "b";
            B();
        }
        catch (Exception e)
        {
            str += "c";
        }
    }

    void B() throws Exception
    {
        try
        {
            str += "d";
            C();
        }
        catch (Exception e)
        {
            throw new Exception();
        }
        finally
        {
            str += "e";
        }

        str += "f";
    }
}
```

```
void C() throws Exception
{
    throw new Exception();
}

void display()
{
    System.out.println(str);
}

public static void main(String[] args)
{
    Test object = new Test();
    object.A();
    object.display();
}
```

แบบฝึกหัด 6

```
public static void main(String[] args) {  
    String[] months = {"Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"};  
    Scanner scanner = new Scanner(System.in);  
    try {  
        int month = scanner.nextInt();  
        System.out.print(months[month]);  
    } catch (IndexOutOfBoundsException exception) {  
        System.out.print("Index is out of bounds");  
    } catch (InputMismatchException exception) {  
        System.out.print("Input mismatch");  
    }  
}
```

1. ถ้าผู้ใช้ป้อน 3 โปรแกรมจะแสดงผลลัพธ์อะไร
2. ถ้าผู้ใช้ป้อน 99 โปรแกรมจะแสดงผลลัพธ์อะไร
3. ถ้าผู้ใช้ป้อน aaa โปรแกรมจะแสดงผลลัพธ์อะไร

Thanks!!!

Udacity Course : Object Oriented Programming in Java

<https://classroom.udacity.com/courses/ud283>

Wu, C. Thomas. (2008). A comprehensive introduction to object-oriented programming with Java.

Wu, C. Thomas. (2010). An introduction to object-oriented programming with Java.

Cay Horstmann. (2010). Big Java.

Cay Horstmann (2017). Java Concepts Late Objects,

Yakov Fain. (2004). Java Programming for Kids, Parents and GrandParents.