

BÀI 7

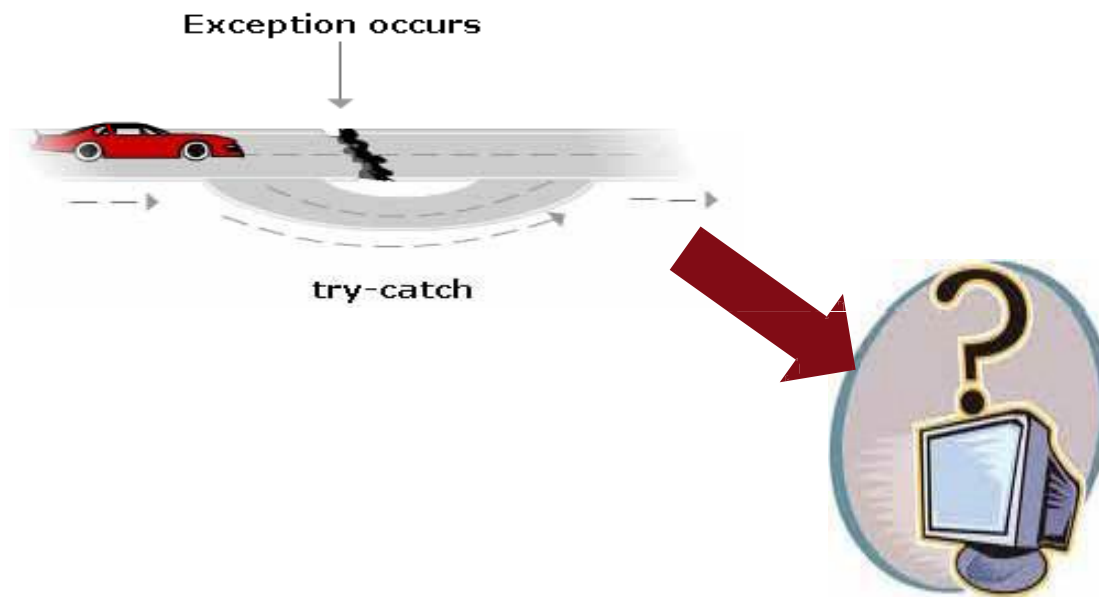
LẬP TRÌNH XỬ LÝ NGOẠI LỆ

TRONG JAVA

TÌNH HUỐNG DẪN NHẬP

Bài toán: Xây dựng hệ thống cảnh báo lỗi

Hôm nay trên đường đi làm Nam gặp phải một đoạn đường đang làm dở, Nam rất bực mình vì mình không thể tiếp tục đi vì không thể cho xe chạy qua đoạn đường đó. Sau một hồi vòng vo, Nam quyết định sẽ rẽ sang đoạn đường khác để đi. Vậy là Nam đã đến cơ quan muộn. Nam nghĩ nếu như đoạn đường đang làm được đưa thông báo thì mình sẽ không bị muộn làm.



Vậy theo Anh Chị nên xây dựng hệ thống cảnh báo lỗi thế nào để Nam biết để tránh lỗi có khả năng sẽ xảy đến.

MỤC TIÊU

Trình bày khái niệm về ngoại lệ.

Mô tả cách thức bắt lỗi, xử lý lỗi trong Java.

Sử dụng một số lớp bắt lỗi được cung cấp sẵn trong Java.

Xây dựng được lớp bắt lỗi của người dùng.

Xây dựng chương trình sử dụng một số lớp bắt lỗi trong Java và lớp bắt lỗi do người dùng tự định nghĩa.

NỘI DUNG

1

Giới thiệu về ngoại lệ.

2

Cách bắt các ngoại lệ.

3

Phân loại ngoại lệ Checked và Unchecked Exception.

4

Ngoại lệ do người dùng định nghĩa.

1. NGOẠI LỆ (EXCEPTION)

Exception là một sự kiện xảy ra trong quá trình thực thi chương trình, thông thường là những trường hợp không mong muốn xuất hiện trong quá trình xử lý.



Trên đường đua người đua xe không ngờ tai nạn xảy đến với mình



Người lái xe không ngờ xe của mình bị lao xuống bể bơi

1. NGOẠI LỆ (EXCEPTION) (tiếp theo)

- Ngoại lệ trong java là một đối tượng miêu tả trạng thái lỗi xảy ra trong một đoạn mã. Khi một trạng thái lỗi nảy sinh, đối tượng đại diện cho ngoại lệ đó được tạo ra và được chuyển cho phương thức gây nên lỗi. Phương thức này có thể tự xử lý ngoại lệ này hoặc cho qua.
- Các tình huống xảy ra Exception:
 - Lỗi chương trình.
 - Lỗi do mã người dùng.
 - Lỗi không thuộc phạm vi kiểm soát của chương trình.
- Trong Java, các lớp Exception được dùng để mô tả các đối tượng ngoại lệ, chứa các thông tin về ngoại lệ phục vụ cho quá trình xử lý.
- Sử dụng ngoại lệ giúp cho quá trình xử lý lỗi nhanh và chính xác.

CÂU HỎI TƯƠNG TÁC



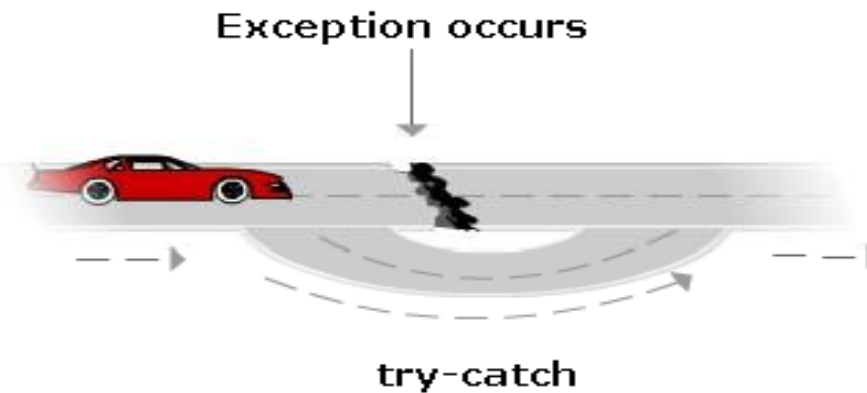
Vì sao chúng ta phải sử dụng ngoại lệ?

2. TUNG VÀ BẮT EXCEPTION

- Cách bắt Exception: Sử dụng cấu trúc try...catch...finally...
- Cách tung Exception: Sử dụng từ khóa throw và throws.

2. CÁCH BẮT CÁC EXCEPTION

Sử dụng khối lệnh `try ... catch`:

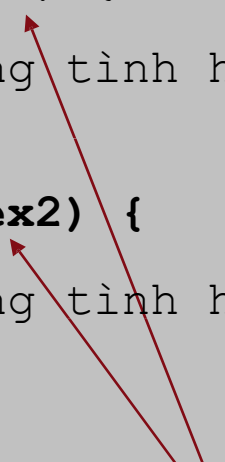


```
try {  
    //Các câu lệnh có khả năng tung ra ngoại lệ  
} catch (Exception ex) {  
    //Các câu lệnh xử lý trong tình huống bắt được ngoại lệ  
}
```

Exception: Đối tượng để bắt ngoại lệ

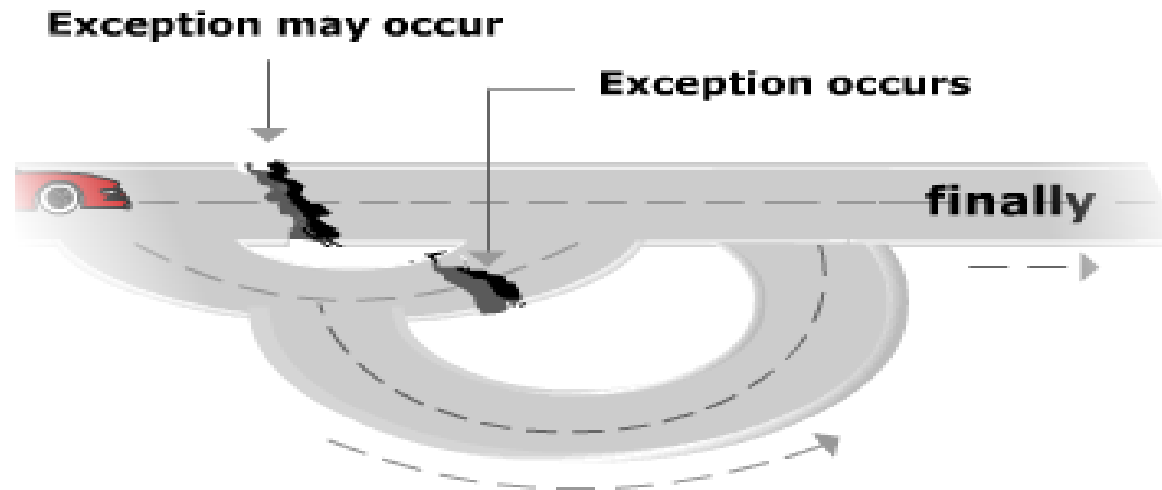
SỬ DỤNG KHỐI NHIỀU KHỐI try...catch LIỀN NHAU

```
try {  
    //Các câu lệnh có khả năng tung ra ngoại lệ  
} catch (<ExceptionType1> ex) {  
    //Các câu lệnh xử lý trong tình huống bắt được ngoại lệ  
    ExceptionType1  
} catch (<ExceptionType2> ex2) {  
    //Các câu lệnh xử lý trong tình huống bắt được ngoại lệ  
    ExceptionType2  
}
```

A diagram consisting of two red arrows. One arrow originates from the text 'ExceptionType1' in the first catch block and points to the text '<ExceptionType2>' in the second catch block. The second arrow originates from the text 'ExceptionType2' in the second catch block and points to the text '<ExceptionType1>' in the first catch block. This indicates that ExceptionType1 must inherit from ExceptionType2.

- Exceptiontype1, Exceptiontype2: Đối tượng để bắt ngoại lệ;
- Trong cây phân cấp kế thừa: Đối tượng Exceptiontype1 phải thuộc cùng cấp với Exceptiontype2, hoặc là lớp con của Exceptiontype2.

SỬ DỤNG KHỐI `finally`



```
try {  
    //Các câu lệnh có khả năng tung ra ngoại lệ  
}catch(Exception ex) {  
    //Các câu lệnh xử lý trong tình huống bắt được ngoại lệ  
}finally{  
    //Các câu lệnh trong khối finally luôn được thực thi dù  
    có hay không có ngoại lệ  
}
```

Một khối Try theo sau là một hoặc nhiều khối catch hoặc chỉ một khối Finally

VÍ DỤ

Đoạn Code có khả năng
sinh lỗi

```
public class DemoNoUseException {  
    public static void main(String[] args) {  
        System.out.println("ket qua phép toan 1/0: ");  
        System.out.println(1/0);  
        System.out.println("Loi da xay ra!");  
    }  
}
```

```
ket qua 1/0:  
Exception in thread "main" java.lang.ArithmeticException:  
/ by zero  
    at TestStudent.main(DemoNoException.java:4)  
Java Result: 1
```

VÍ DỤ (tiếp theo)

```
public class DemoUseException{
    public static void main(String[] args) {
        System.out.println("ket qua phep toan 1/0: ");
        try {
            System.out.println(1/0);
        } catch (ArithmeticException aex) {
            System.out.println("Loi: " + aex.getMessage() + " xay ra.");
            aex.printStackTrace();
        }
        System.out.println("Loi da xay ra!");
    }
}
```

```
java.lang.ArithmeticException: / by zero
ket qua phep toan 1/0:
Loi: / by zero xay ra.
at TestStudent.main(TestStudent.java:7)
Loi da xay ra!
```

VÍ DỤ (tiếp theo)

Sử dụng đối tượng bắt lỗi không chính xác vẫn làm cho chương trình kết thúc.

```
public class DemoUseException{
    public static void main(String[] args) {
        System.out.println("ket qua phep toan 1/0: ");
        try {
            System.out.println(1 / 0);
        } catch (ArrayIndexOutOfBoundsException aex) {
            System.out.println("Loi:" + aex.getMessage() + " xay ra.");
        }
        System.out.println("Loi da xay ra!");
    }
}
```

```
ket qua phep toan 1/0:
Exception in thread "main" java.lang.ArithmeticException:
/ by zero
    at TestStudent.main(TestStudent.java:7)
```

VÍ DỤ (tiếp theo)

Nếu không biết đối tượng nào bắt lỗi. Sử dụng lớp exception là lớp cha của các lớp xử lý lỗi để bắt

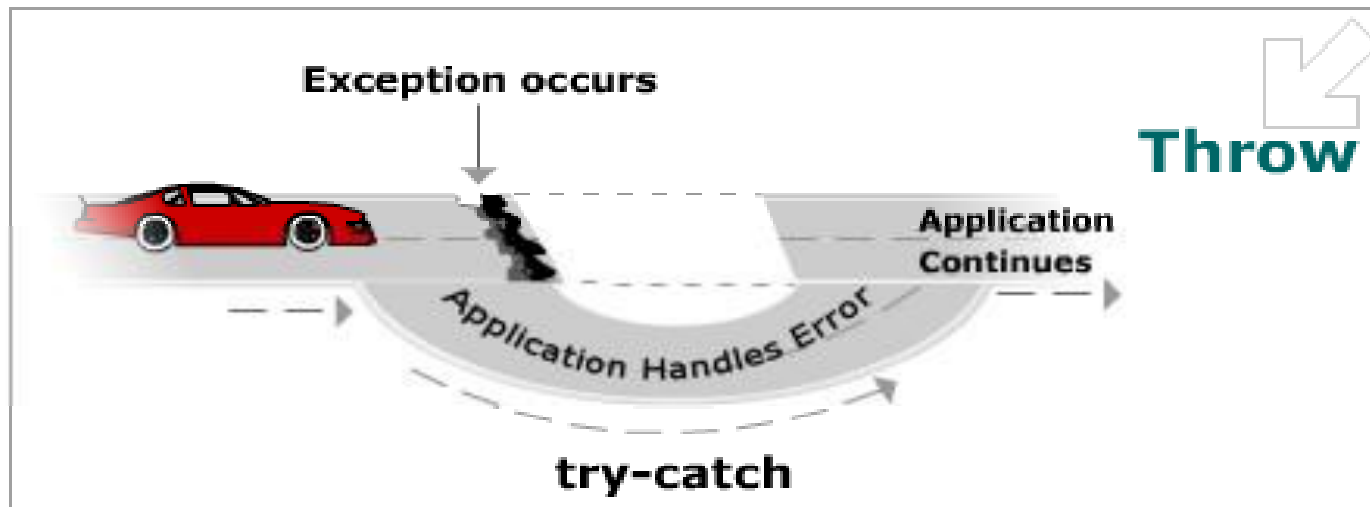
```
public class DemoUseException {  
    public static void main(String[] args) {  
        System.out.println("ket qua phep toan 1/0: ");  
        try {  
            System.out.println(1 / 0);  
        } catch (ArrayIndexOutOfBoundsException aex) {  
            System.out.println("Loi:" + aex.getMessage() + "xay ra.");  
        } catch (Exception ex) {  
            System.out.println("Loi:" + ex.getMessage() + "xay ra.");  
            ex.printStackTrace();  
        }  
        System.out.println("Loi da xay ra!");  
    }  
}
```

```
ket qua phep toan 1/0:  
java.lang.ArithmeticException: / by zero  
Loi: / by zero xay ra.  
Loi da xay ra!  
    at TestStudent.main(TestStudent.java:7)
```

2. CÁCH TUNG EXCEPTION

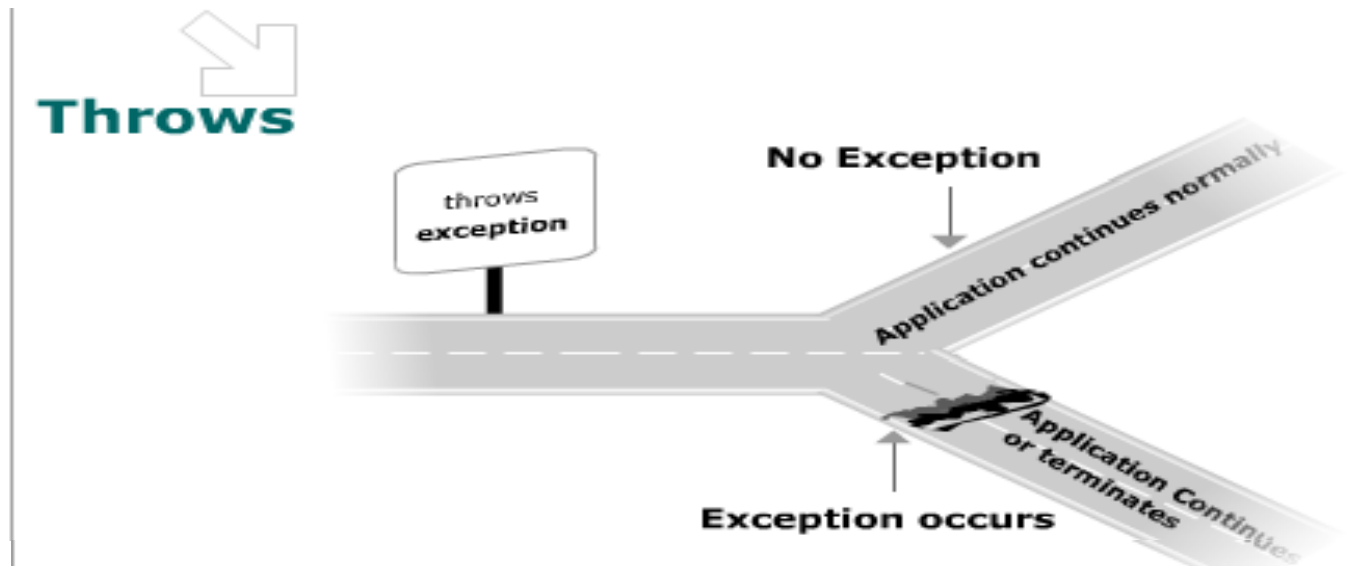
Tung ra Exception sử dụng từ khóa **throw**:

```
throw new Exception("This is an exception");
```



2. CÁCH TUNG EXCEPTION (tiếp theo)

Cảnh báo một phương thức có khả năng tung ra các Exception sử dụng từ khóa **throws**:



```
public void methodName() throws ExceptionType {  
    //body method  
    throw new ExceptionType("This is an exception");  
}
```

VÍ DỤ

```
public class ThrowDemo{
    public static void demoproc() {
        try {
            throw new NullPointerException("demo");
        } catch (NullPointerException e) {
            System.out.println("Caught inside demoproc.");
            throw e;
        }
    }
    public static void main(String[] args) {
        try {
            demoproc();
        } catch (NullPointerException e) {
            System.out.println("Recaught: " + e);
        }
    }
}
```

Caught inside demoproc.

Recaught: java.lang.NullPointerException: demo

CÂU HỎI TRẮC NGHIỆM

Câu hỏi 1 trên 4 ▾

Điểm: 10

Ngoại lệ là một sự kiện xảy ra trong quá trình thực thi chương trình, thông thường là những trường hợp không mong muốn xuất hiện trong quá trình xử lý.

- ☐ A. Đúng.
- ☐ B. Sai.

PROPERTIES

On passing, 'Finish' button:

On failing, 'Finish' button:

Allow user to leave quiz:

User may view slides after quiz:

User may attempt quiz:

Goes to Next Slide

Goes to Next Slide

At any time

At any time

Unlimited times



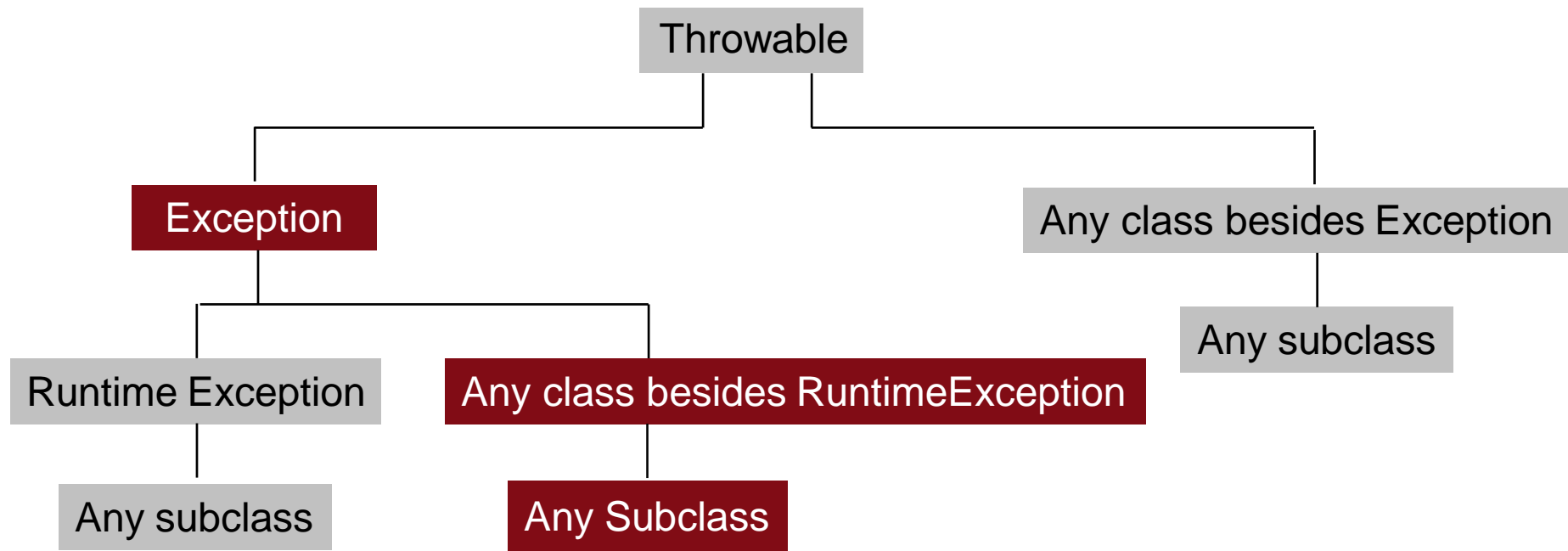
Properties...



Edit in Quizmaker

3. PHÂN LOẠI CÁC EXCEPTION

- Unchecked Exception;
- Checked Exception.



Ngoại lệ thuộc loại unchecked



Ngoại lệ thuộc loại checked

3.1. CHECKED EXCEPTION

Khi một Checked Exception được tung ra thì bắt buộc đoạn mã xử lý phải bắt ngoại lệ này.

Ngoại lệ (Exception)	Mô tả
InstantiationException	Ngoại lệ này xảy ra khi người sử dụng cố tình tạo ra một thể hiện của lớp trừu tượng.
InterruptedException	Ngoại lệ này xảy ra khi một tác vụ đang được thực thi bị ngừng hoạt động.
NoSuchMethodException	Ngoại lệ này xảy ra khi máy ảo java (JVM) không thể kiểm tra sự truy xuất vào các phương thức được gọi.

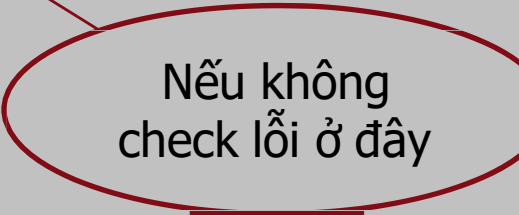
3.2. UNCHECKED EXCEPTION

Khi một Unchecked Exception được tung ra lập trình viên có thể bắt hoặc không bắt ngoại lệ này.

Ngoại lệ (Exception)	Mô tả
RuntimeException	Ngoại lệ này xảy ra trong quá trình thực thi chương trình lỗi bất kỳ làm chương trình ngưng hoạt động.
ArithmeticException	Ngoại lệ này xảy ra khi các lỗi liên quan đến phép toán bị sai phạm.
ArrayIndexOutOfBoundsException	Ngoại lệ này xảy ra khi truy xuất vào phần tử null trong mảng, hoặc truy xuất phần tử không tồn tại trong mảng.
IllegalArgumentException	Ngoại lệ này xảy ra khi kích thước mảng khởi tạo nhỏ hơn 0.
NumberFormatException	Ngoại lệ này xảy ra khi chuyển từ chuỗi sang số không hợp lệ.
StringIndexOutOfBoundsException	Ngoại lệ này xảy ra khi chỉ số mảng chuỗi là số âm hoặc không hợp lệ.

VÍ DỤ MINH HỌA VỀ CHECKED EXCEPTION

```
//Lop nay minh hoa ve checked exception
public class CheckedExceptionDemo {
    //phuong thuc nay goi den ham canthrowCheckedException de nen loi
    public void doChecked() {
        // Buoc phai check exception o day! Khong cach nao khac
        canThrowCheckedException();
        System.out.println("OK");
    }
    private int canThrowCheckedException() throws Exception {
        throw new Exception("Failure");
    }
    public static void main(String[] args) {
        new CheckedExceptionDemo().doChecked();
        System.out.println("Lenh sau bat loi");
    }
}
```



Nếu không check lỗi ở đây

```
Exception in thread "main" java.lang.RuntimeException: Uncompilable
source code - unreported exception java.lang.Exception; must be caught
or declared to be thrown
at CheckedExceptionDemo.doChecked(CheckedExceptionDemo.java:6)
at CheckedExceptionDemo.main(CheckedExceptionDemo.java:15)
Java Result: 1
```

VÍ DỤ MINH HỌA VỀ UNCHECKED EXCEPTION

```
/**
 * Client.java
 */
public class Client {
    public void doUnchecked(String value) {
        // Can phải check exception, nếu không --> bug
        int result=canThrowUncheckedException(value);
        System.out.println("result="+result);
    }
    private int canThrowUncheckedException(String value) throws
    NumberFormatException{
        return Integer.parseInt(value);
    }
    public void doChecked() {
        try {
            // Bước phải check exception ở đây! Không cách nào khác
            canThrowCheckedException();
            System.out.println("OK");
        } catch (Exception ex) {
            System.out.println(ex);
        }
    }
    private int canThrowCheckedException() throws Exception{
        throw new Exception("Failure");
    }
}
```

Bài tập 1

4. XÂY DỰNG CÁC NGOẠI LỆ

- Sử dụng cách thức kế thừa các ngoại lệ đã được định nghĩa trước để tạo ra các ngoại lệ mới.
- Ví dụ tạo một checked Exception:

```
public class ZeroNumberException extends Exception {  
    public ZeroNumberException() {  
        super("Zero number!");  
    }  
}
```

```
...  
public void methodTest(int n) throws ZeroNumberException{  
    if(n==0) {  
        throw new ZeroNumberException();  
    }  
}
```

VÍ DỤ

```
public class MyException extends Exception {
    MyException() {
        super("My Exception");
    }
}

////////////////////////////////////
public class YourException extends Exception {
    YourException() {
        super("Your Exception");
    }
}

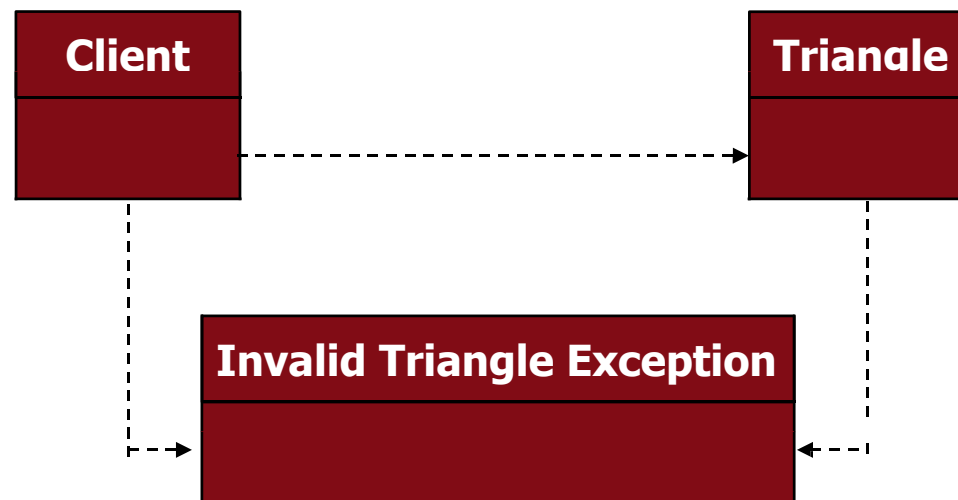
////////////////////////////////////
public class ChainDemo {
    public static void main(String[] args) {
        try {
            someMethod1();
        } catch (MyException e) {
            e.printStackTrace();
        }
    }

    static void someMethod1() throws MyException {
        try {
            someMethod2();
        } catch (YourException e) {
            System.out.println(e.getMessage());
            MyException e2 = new MyException();
            e2.initCause(e);
            throw e2;
        }
    }

    static void someMethod2() throws YourException {
        throw new YourException();
    }
}
```

BÀI TẬP

Tạo một `InvalidTriangleException` bằng cách kế thừa lớp `Exception`. Sử dụng `Exception` để phục vụ quá trình bắt lỗi trong quá trình khởi tạo đối tượng tam giác (`Triangle`), nếu người dùng khởi tạo không đúng các cạnh của tam giác sẽ throw ra `InvalidTriangleException`.



Cạnh tam giác không hợp lệ khi:

- Giá trị cạnh < 0 ;
- Giá trị cạnh là số quá lớn (xét trong giới hạn khoảng số nguyên).

Bài tập 2

CÂU HỎI TRẮC NGHIỆM

Câu hỏi 1 trên 3 ▾

Điểm: 10

Đọc các tình huống sau:

Câu 1: Khi một Checked Exception được tung ra thì bắt buộc đoạn mã xử lý phải bắt ngoại lệ này

Câu 2: Khi một Unchecked Exception được tung ra thì có thể bắt hoặc không bắt ngoại lệ này.

- ☐ A. Cả hai câu đều đúng.
- ☐ B. Cả hai câu đều sai.
- ☐ C. Câu 1 đúng, Câu 2 sai.
- ☐ D. Câu 2 đúng, Câu 1 sai.

PROPERTIES

On passing, 'Finish' button:

On failing, 'Finish' button:

Allow user to leave quiz:

User may view slides after quiz:

User may attempt quiz:

Goes to Next Slide

Goes to Next Slide

At any time

At any time

Unlimited times



Properties...



Edit in Quizmaker

TÓM LƯỢC CUỐI BÀI

Sau khi học xong bài này chúng ta đã nắm được các kiến thức sau:

- Nắm được khái niệm về ngoại lệ;
- Sử dụng cách bắt ngoại lệ;
- Xây dựng ngoại lệ người dùng;
- Xây dựng chương trình sử dụng ngoại lệ của hệ thống và do người dùng tự định nghĩa.

CÂU HỎI TRẮC NGHIỆM

Câu hỏi 1 trên 4 ▾

Điểm: 10

Trong chương trình thứ tự của các lệnh như thế nào là sai?

- ☐ A. Try...catch...finally.
- ☐ B. Try...catch...catch...finally.
- ☐ C. Try...finally.
- ☐ D. Try.

PROPERTIES

On passing, 'Finish' button:

On failing, 'Finish' button:

Allow user to leave quiz:

User may view slides after quiz:

User may attempt quiz:

Goes to Next Slide

Goes to Next Slide

At any time

At any time

Unlimited times



Properties...



Edit in Quizmaker

Phân biệt Unchecked Exception và Checked Exception?

Phân biệt Unchecked Exception và Checked Exception?

Gợi ý:

Checked Exception: Khi một Checked Exception được tung ra thì bắt buộc đoạn mã xử lý phải bắt ngoại lệ này.

Unchecked Exception: Khi một Unchecked Exception được tung ra lập trình viên có thể bắt hoặc không bắt ngoại lệ này.

Phương thức `printStackTrace` của lớp `Exception` dùng để làm gì?

Tại sao không nên `catch (Exception ex)`?

Phương thức `getMessage()` của lớp `Exception` dùng để làm gì?

Exception là gì?

PROPERTIES

Allow user to leave interaction:

Show 'Next Slide' Button:

Completion Button Label:

[Anytime](#)

[Don't show](#)

[Next Slide](#)





A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Select a term:

Checked Exception

Lớp Exception

Từ khóa finally

Từ khóa throw

Từ khóa throws

Unchecked Exception

Checked Exception

Khi một Checked Exception được tung ra thì bắt buộc đoạn mã xử lý phải bắt ngoại lệ này.

PROPERTIES

Allow user to leave interaction:

Show 'Next Slide' Button:

Completion Button Label:

[Anytime](#)[Don't show](#)[Next Slide](#)