Trường Cao đẳng Công nghệ Thông tin TP.HCM

Khoa Công nghệ Thông tin – Điện tử

Chương 2:

Ngoại lệ

Giảng viên: Hà Mỹ Trinh

Email: trinhhm@itc.edu.vn

Nội dung

1. NGOẠI LỆ
2. PHÂN LOẠI NGOẠI LỆ
3. XỬ LÝ NGOẠI LỆ
4. KIỂM LỖI
5. SỬ DỤNG FINALLY
6. SỬ DỤNG THROWS VÀ THROW
7. TẠO EXCEPTION MỚI
8. NGOẠI LỆ TỰ ĐỊNH NGHĨA

1. Ngoại lệ

- Có những lỗi chỉ khi chạy chương trình mới xuất hiện và chương trình đang chạy lập tức ngừng lại và xuất hiện thông báo lỗi – đó chính là ngoại lệ (exception).
- Ngoại lệ (exception) = Exceptional event
- Định nghĩa: Ngoại lệ là một sự kiện xảy ra trong quá trình thực thi chương trình, phá vỡ luồng bình thường của chương trình
- Ví dụ: Xét chương trình chia 2 số. Nếu ta cho mẫu số = 0 thì phát sinh lỗi và đó được coi là 1 ngoại lệ.

int i = 4/0; → Error

1. Ngoại lệ

- Ngoại lệ là một lỗi đặc biệt
- Xảy ra tại thời điểm chạy chương trình (runtime)
- Khi xảy ra một ngoại lệ, nếu không xử lý thì chương trình kết thúc ngay và trả lại quyền điều khiển cho hệ điều hành.
- Kết thúc bất thường chương trình
- Kết quả thực thi không mong muốn

1. Ngoại lệ

```java
public static void main(String[] args) {
    System.out.println(x: "Three");
    // Phép chia này không có vấn đề.
    int value = 10 / 2;
    System.out.println(x: "Two");
    // Phép chia này không có vấn đề.
    value = 10 / 1;
    System.out.println(x: "One");

    // Phép chia này có vấn đề, chia cho 0.
    // Lỗi đã xảy ra tại đây.
    value = 10 / 0;
    // Và dòng code dưới đây sẽ không được thực hiện.
    System.out.println(x: "Let's go!");
}
```

```
Two
One
Exception in thread "main" java.lang.ArithmeticException: / by zero
        at ngoaile.ngoaile.main(ngoaile.java:16)
PS D:\5. ITC\demo\lthdt\baisoan>
```

1. Ngoại lệ

Luồng đi của chương trình

```
public static void main(String[] args) {
    System.out.println(x: "Three");
    // Phép chia này không có vấn đề.
    int value = 10 / 2;
    System.out.println(x: "Two");
    // Phép chia này không có vấn đề.
    value = 10 / 1;
    System.out.println(x: "One");

    // Phép chia này có vấn đề, chia cho 0.
    // Lỗi đã xảy ra tại đây.
    value = 10 / 0;
    // Và dòng code dưới đây sẽ không được thực hiện.
    System.out.println(x: "Let's go!");

}
```

(1) (2) (3) (4) (5) (6) (7)

Fix: Message “Error”

1. Ngoại lệ

Sửa code

```
System.out.println(x: "Three");
// Phép chia này không có vấn đề.
int value = 10 / 2;
System.out.println(x: "Two");
// Phép chia này không có vấn đề.
value = 10 / 1;
System.out.println(x: "One");
```

1. Ngoại lệ

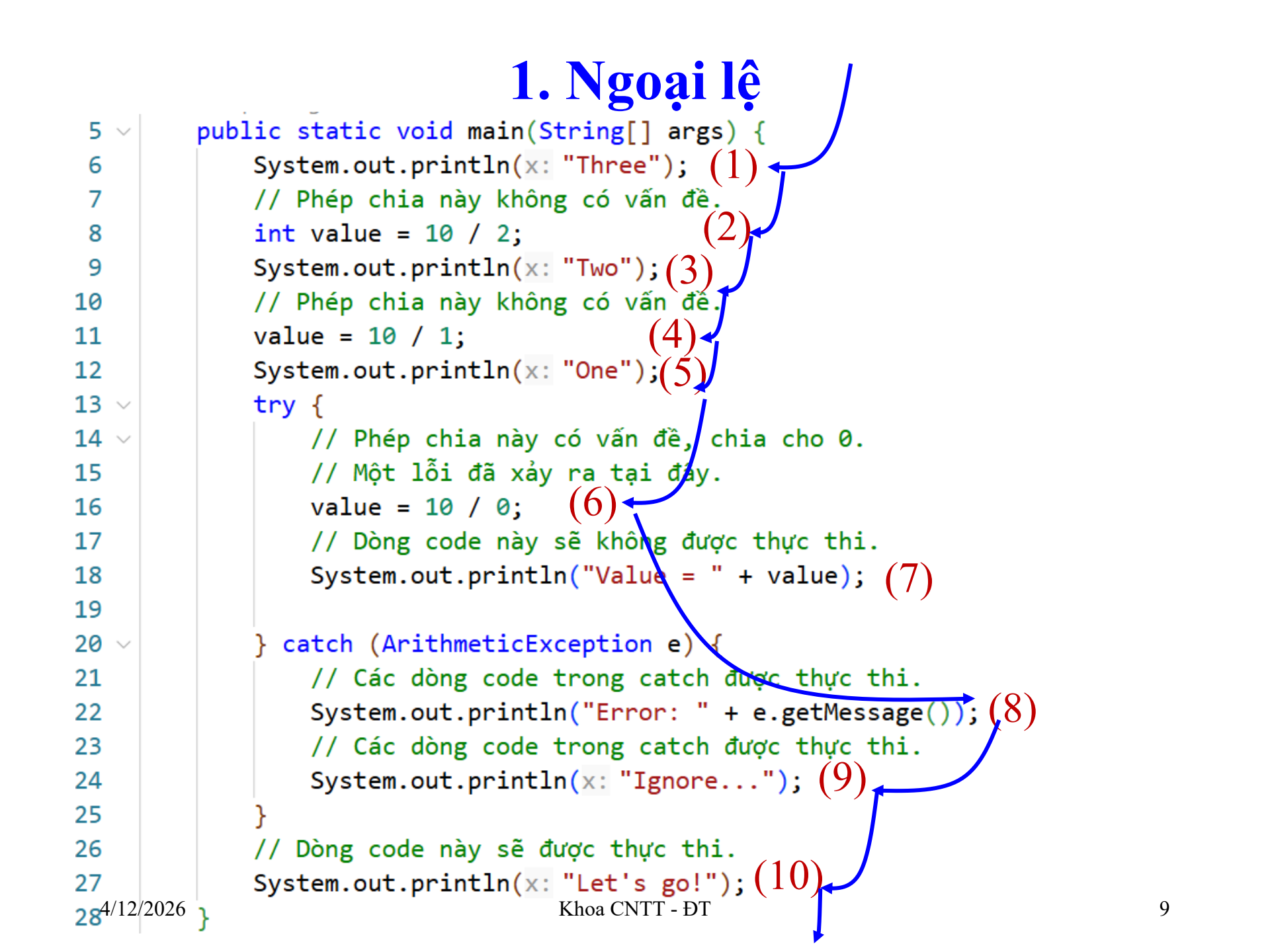
Sửa code

```
try {
    // Phép chia này có vấn đề, chia cho 0.
    // Một lỗi đã xảy ra tại đây.
    value = 10 / 0;

    // Dòng code này sẽ không được thực thi.
    System.out.println("Value = " + value);

} catch (ArithmeticException e) {
    // Các dòng code trong catch được thực thi.
    System.out.println("Error: " + e.getMessage());

    // Các dòng code trong catch được thực thi.
    System.out.println(x: "Ignore...");
}

// Dòng code này sẽ được thực thi.
System.out.println(x: "Let's go!");
```

```
Three
Two
One
Error: / by zero
Ignore...
Let's go!
```

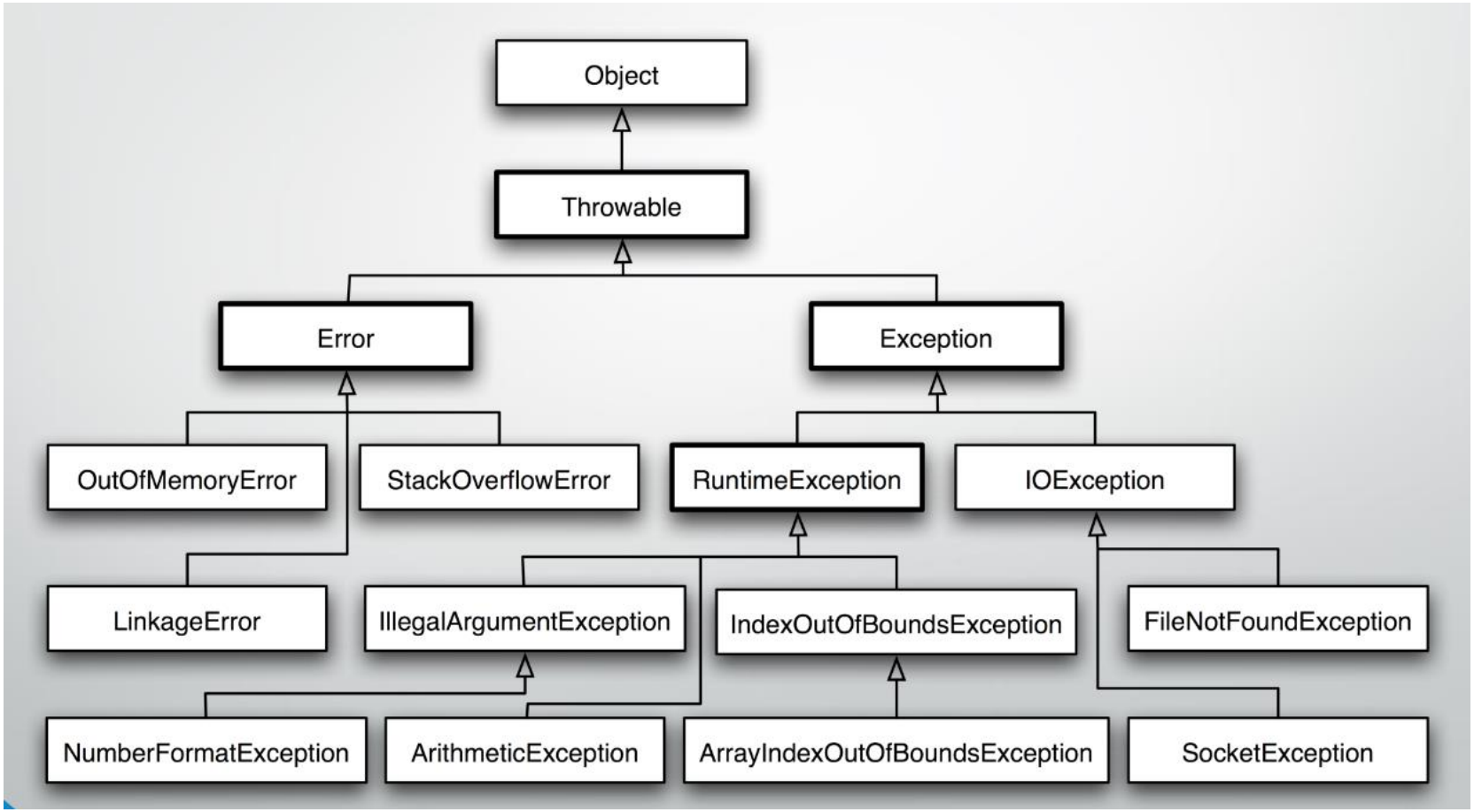
1. Ngoại lệ

```
5   public static void main(String[] args) {
6       System.out.println(x: "Three");
7       // Phép chia này không có vấn đề.
8       int value = 10 / 2;
9       System.out.println(x: "Two");
10      // Phép chia này không có vấn đề.
11      value = 10 / 1;
12      System.out.println(x: "One");
13      try {
14          // Phép chia này có vấn đề, chia cho 0.
15          // Một lỗi đã xảy ra tại đây.
16          value = 10 / 0;
17          // Dòng code này sẽ không được thực thi.
18          System.out.println("Value = " + value);
19
20      } catch (ArithmeticException e) {
21          // Các dòng code trong catch được thực thi.
22          System.out.println("Error: " + e.getMessage());
23          // Các dòng code trong catch được thực thi.
24          System.out.println(x: "Ignore...");
25      }
26      // Dòng code này sẽ được thực thi.
27      System.out.println(x: "Let's go!");
28  }
```



1. Ngoại lệ

- Class Throwable xử lý lỗi và ngoại lệ (Error, Exception).



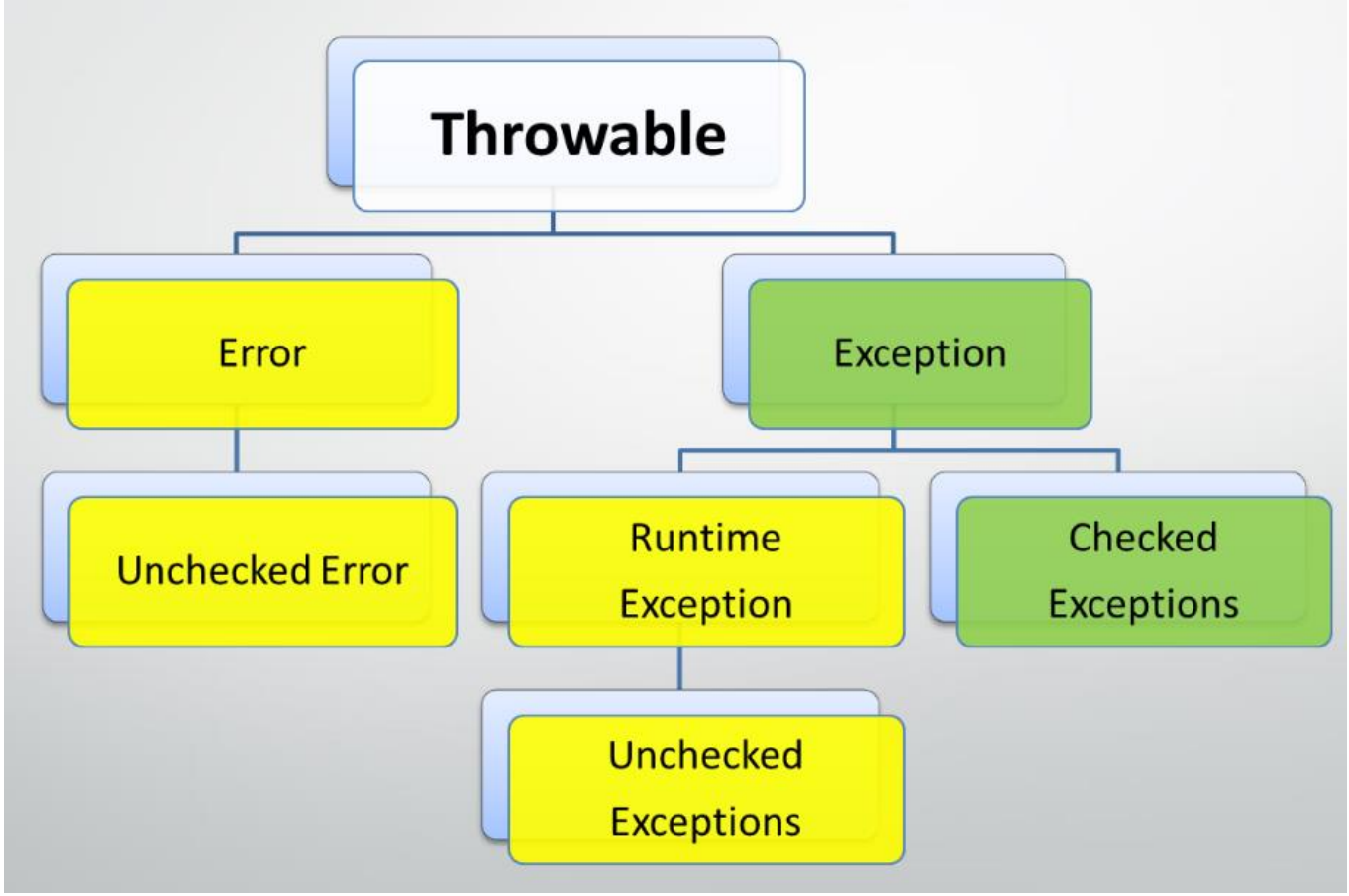
1. Ngoại lệ

- Bài tập: Chuyển đổi chuỗi sang số

```
try {
        int a = Integer.parseInt(string);
        System.out.println("Thành công");
} catch (Exception ex) {
        System.out.println("Lỗi");
}
```

2. Phân loại ngoại lệ

- Exception chia làm 2 loại là checked (xanh) và unchecked (vàng)



2. Phân loại ngoại lệ

- Ngoại lệ 'unchecked':
  - Là các ngoại lệ được kiểm tra lúc chạy
  - Bao gồm các class Error, RuntimeException và các lớp con của chúng
  - Ví dụ: Integer.parseInt("abc") vẫn dịch được nhưng chạy lỗi.
- Ngoại lệ 'checked':
  - Là các ngoại lệ được kiểm tra lúc dịch
  - Bao gồm các class exception còn lại
  - Ví dụ: new FileWriter("c:/data.txt") dịch lỗi dù file đã tồn tại

2. Phân loại ngoại lệ

- Một số ngoại lệ 'checked':
  - ClassNotFoundException
  - IOException
  - FileNotFoundException
  - EOFException
- Một số ngoại lệ 'unchecked'
  - ArithmeticException
  - IllegalArgumentException
  - IndexOutOfBoundException
  - NullPointerException
  - InputMismatchException

3. Xử lý ngoại lệ

- Sử dụng lệnh try…catch để xử lý các ngoại lệ

```
try {
    //Khối lệnh có khả năng gây ra
    //ngoại lệ
}
catch (…) {// Bắt các xử lý ngoại lệ
    //Khối lệnh xử lý ngoại lệ
}
```

3. Xử lý ngoại lệ

- Ví dụ sau xử lý lỗi chuyển chuỗi sang số nguyên

```
try {
        int a = Integer.parseInt(string);
        System.out.println("Thành công");
} catch (Exception ex) {
        System.out.println("Lỗi");
}
```

3. Xử lý ngoại lệ

Khối try/ catch lồng nhau

- Những phần nhỏ trong khối mã sinh ra một lỗi, nhưng toàn bộ cả khối thì lại sinh ra một lỗi khác → Cần có các xử lý ngoại lệ lồng nhau.
- Khi các khối try lồng nhau, khối try bên trong sẽ được thực hiện trước.

```
try {
    statement 1;
    statement 2;
    try {
        statement 1;
        statement 2;
    }
    catch(Exception e) {

    }
}
catch(Exception e) {

}
```

3. Xử lý ngoại lệ

- Nhiều khối catch
  - Một đoạn mã có thể gây ra nhiều hơn một ngoại lệ → Sử dụng nhiều khối catch.

```
try {
 // Doan ma co the gay ra nhieu ngoai le
} catch (ExceptionType1 e1) {
    // Xu ly ngoai le 1
} catch (ExceptionType2 e2) {
    // Xu ly ngoai le 2
} ...
```

  - ExceptionType1 phải là lớp con hoặc ngang hàng với ExceptionType2 (trong cây phân cấp kê thừa)

3. Xử lý ngoại lệ

```
public class TestMultiCatchBlock {
    public static void main(String args[]) {
        try {
            int a[] = new int[5];
            a[5] = 30 / 0;
        } catch (ArithmeticException e) {
            System.out.println("Loi chia cho 0");
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Loi ngoai pham vi mang");
        }
        System.out.println("The end");
    }
}
```

3. Xử lý ngoại lệ

- Ví dụ về Exception ngang hang:

```
try {
    int x = Integer.parseInt("abc");
    int y = 10 / 0;
}
catch (NumberFormatException e) {
    System.out.println("Sai định dạng số!");
}
catch (ArithmeticException e) {
    System.out.println("Lỗi chia cho 0!");
}
```

3. Xử lý ngoại lệ

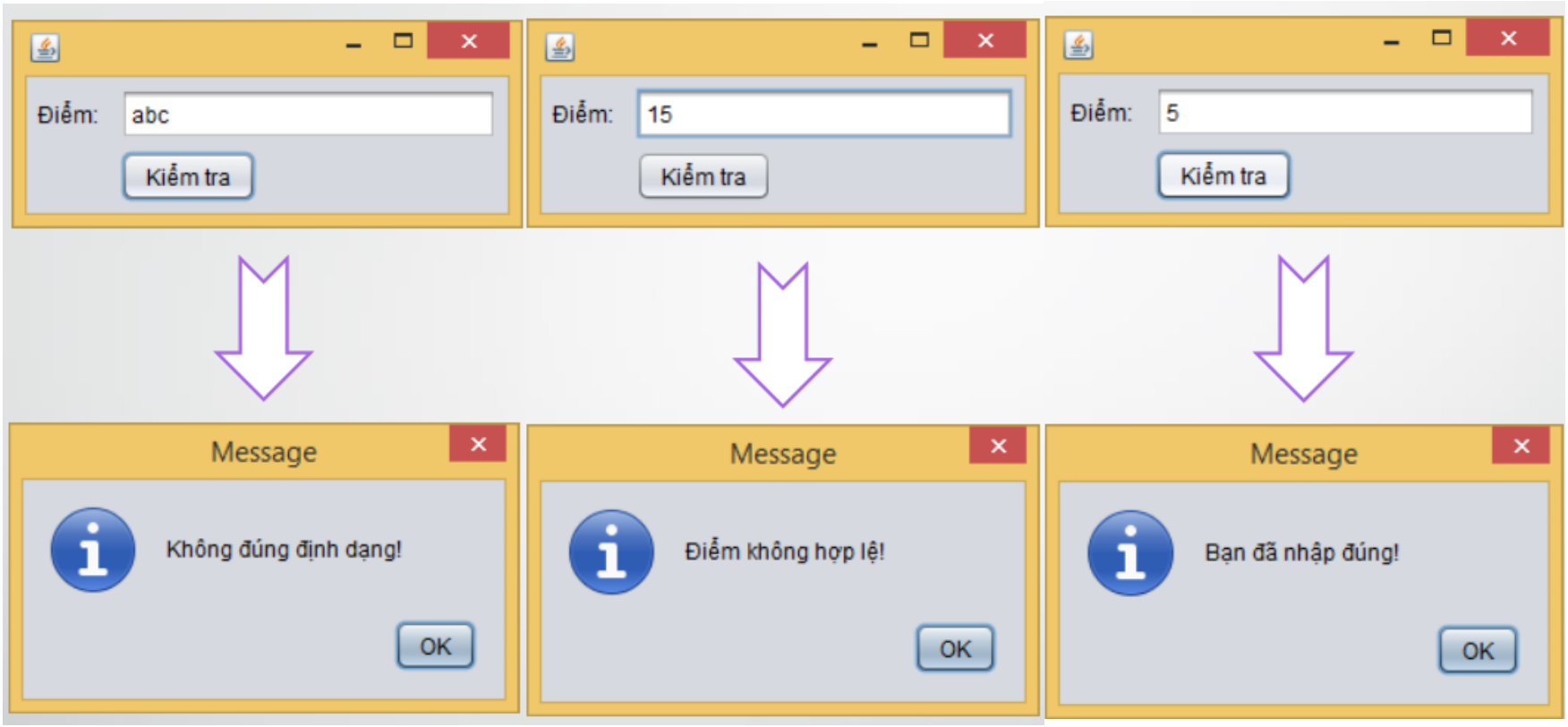
- Ví dụ SAI:

```
catch (Exception e) { }
catch (ArithmeticException e) { } // lỗi vì Exception bao trùm hết
```

- Ví dụ ĐÚNG:

```
catch (ArithmeticException e) { }
catch (Exception e) { }
```

4. Kiểm lỗi


Điểm: abc
Kiểm tra
Điểm: 15
Kiểm tra
Điểm: 5
Kiểm tra
Message
Không đúng định dạng!
OK
Message
Điểm không hợp lệ!
OK
Message
Bạn đã nhập đúng!
OK

4. Kiểm lỗi

```
1.  String text = txtDiem.getText();
2.  try {
3.      double diem = Double.parseDouble(text);
4.      if(diem < 0 || diem > 10){
5.          JOptionPane.showMessageDialog(this, "Điểm không hợp
    lệ!");
6.      }
7.      else{
8.          JOptionPane.showMessageDialog(this, "Bạn đã nhập
    đúng!");
9.      }
10. }
11. catch (Exception e) {
12.     JOptionPane.showMessageDialog(this, "Không đúng định
    dạng!");
13. }
```

4.1 Bắt lỗi chi tiết

- Khối mã try có thể có nhiều ngoại lệ xảy ra.
- Sử dụng nhiều khối catch để bắt và xử lý chi tiết các ngoại lệ đó.

```
String[] ss = {"1", "a", "2"};
try {
        int a = Integer.parseInt(ss[1]);
}
catch (NumberFormatException e1) {
        System.out.println("Không đúng định dạng số !");
}
catch (ArrayIndexOutOfBoundsException e2) {
        System.out.println("Ngoài phạm vi mảng !");
}
catch (NullPointerException e3) {
        System.out.println("Mảng chưa được khởi tạo !");
}
```

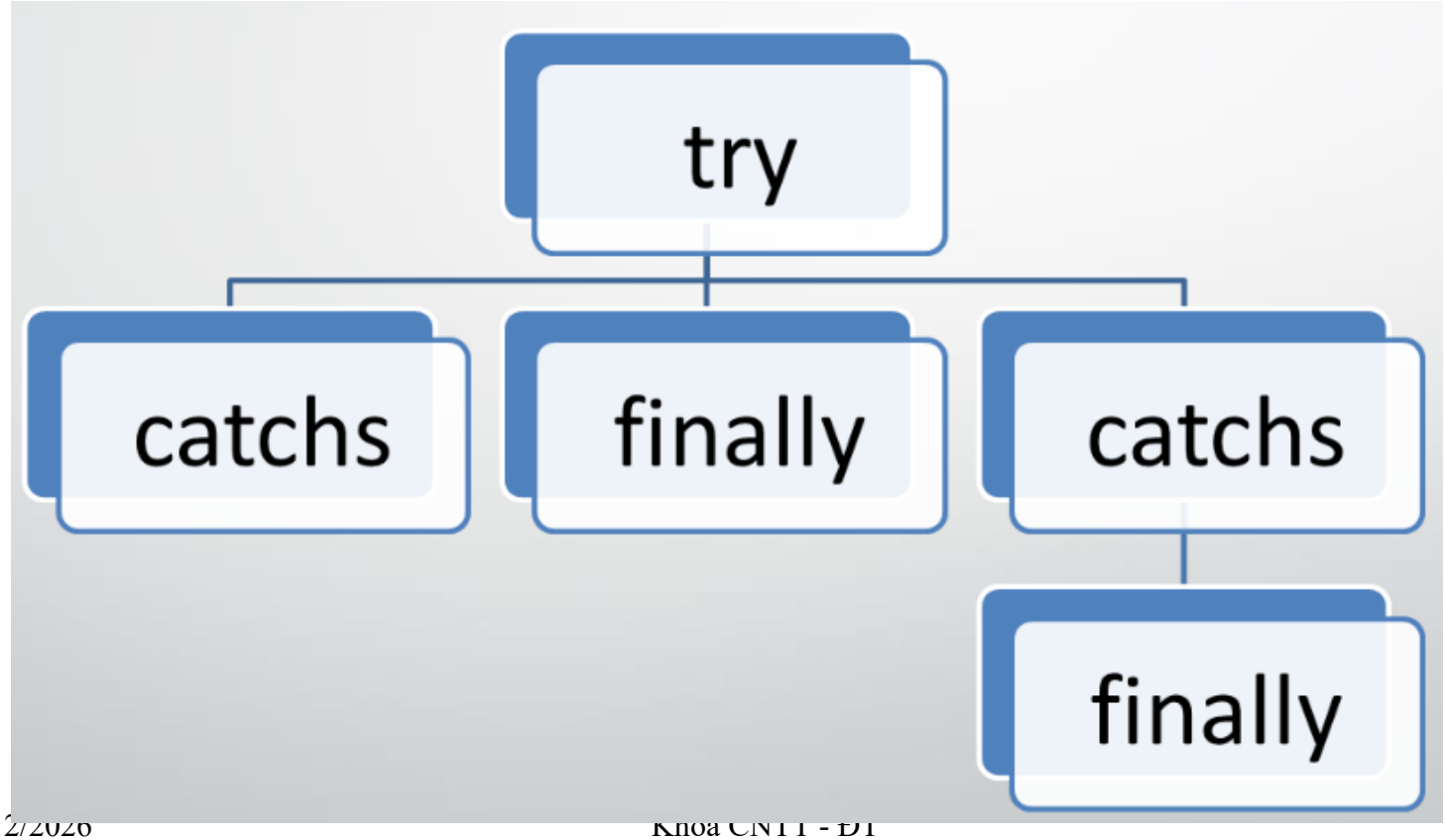
4.2 Bắt lỗi chung

- Catch thứ 2 của đoạn mã sau bắt chung cả 2 ngoại lệ NumberFormatException và NullPointerException do cả 2 ngoại lệ này đều là con của Exception
- Catch bắt ngoại lệ chung phải đặt sau cùng

```
String[] ss = {"1", "a", "2"};
try {
        int a = Integer.parseInt(ss[1]);
}
catch (NumberFormatException e1) {
        System.out.println("Không đúng định dạng số !");
}
catch (Exception e2) {
        System.out.println("Lỗi chuyển đổi số !");
}
```

5. Sử dụng Finally

- Mỗi khối try yêu cầu có ít nhất một khối catch hoặc/và duy nhất một khối finally.
- Khối finally sẽ được thực hiện dù ngoại lệ có xuất hiện hay không.



5. Sử dụng Finally

- Cú pháp try – catch – finally

```
try {
    // Khoi lenh co the sinh ngoai le
}
catch(ExceptionType e) {
    // Bat va xu ly ngoai le
}
finally {
    /* Thuc hien cac cong viec can thiet du ngoai le
     * co xay ra hay khong */
}
```

- Nếu đã có khối try thì bắt buộc phải có khối catch hoặc khối finally hoặc cả hai

5. Sử dụng Finally

```
try {
        int a = Integer.parseInt(ss[1]);
}
catch (Exception e) {
        System.out.println("Lỗi !");
}
```

Ngoại lệ đã được xử lý

```
try {
        int a = Integer.parseInt(ss[1]);
}
catch (Exception e) {
        System.out.println("Lỗi !");
}
finally {
        ss = null;
}
```

```
try {
        int a = Integer.parseInt(ss[1]);
}
finally {
        ss = null;
}
```

Ngoại lệ chưa được xử lý


Q
&
A