

Lập trình JAVA

Biên soạn: ZendVN

Hướng dẫn: Lưu Trường Hải Lân

Hỗ trợ Skype: zendvn.support

Chương 01:

Nền tảng lập trình JAVA

- 
- 1. *Tổng quan lập trình JAVA*
 - 2. *Biến số, hằng số và kiểu dữ liệu*
 - 3. *Toán tử và hàm toán học*
 - 4. *Phát biểu điều kiện*
 - 5. *Sử dụng Vòng lặp*
 - 6. *Ứng dụng máy tính bỏ túi*
 - 7. *Ứng dụng Game “Đi tìm ẩn số”*
 - 8. *Ứng dụng tháp hình*
 - 9. *Ứng dụng mô phỏng máy ATM*
 - 10. *Tình huống thực hành có đáp án*

Chương 01:

Nền tảng lập trình JAVA

- 1. Tổng quan lập trình JAVA*
- 2. Biến số, hằng số và kiểu dữ liệu*
- 3. Toán tử và hàm toán học*
- 4. Phát biểu điều kiện*
- 5. Sử dụng Vòng lặp*
- 6. Ứng dụng máy tính bỏ túi*
- 7. Ứng dụng Game “Đi tìm ẩn số”*
- 8. Ứng dụng tháp hình*
- 9. Ứng dụng mô phỏng máy ATM*
- 10. Tình huống thực hành có đáp án*

Chương 01:

Nền tảng lập trình JAVA

Phần 01 - Tổng quan lập trình JAVA

JAVA là gì ?

- JAVA là ngôn ngữ lập trình hướng đối tượng do Sun Microsystem sáng lập
- JAVA được tạo ra với tiêu chí “Write Once, Run Anywhere”



(Chương trình Java nhờ có máy ảo Java nên có thể thực thi được trên nhiều nền tảng khác nhau, không phụ thuộc vào phần cứng, hệ điều hành, ...)

Tại sao là JAVA ?

- Ngôn ngữ lập trình hướng đối tượng dễ tiếp cận
- Tài liệu phong phú đa dạng (API, source, documents, article, ebook, video, ...)
- Khá nhiều công cụ hỗ trợ lập trình: Netbeans, Eclipse, ...
- Hoàn toàn miễn phí và có cộng đồng phát triển rộng lớn
- Xuất hiện ở khắp nơi: web, mobile, desktop, ...

Phần mềm học JAVA

Bộ cài đặt JDK (Java Development Kit)

- <http://www.oracle.com/technetwork/java/javase/downloads/index.html>
- Máy ảo JAVA (Java Virtual Machine)

Môi trường phát triển IDE (Integrated Development Environment)

- Netbeans
- Eclipse
- JCreator



Hello World!

Khởi tạo và thực thi project trong Eclipse

- Tạo project JAVA
- Thực thi tập tin JAVA

Thực thi tập tin .java bằng CMD (Command Line)

- Thiết lập biến môi trường
- Biên dịch và thực thi tập tin java: javac, java

Chương 01:

Nền tảng lập trình JAVA

- 
- 1. *Tổng quan lập trình JAVA*
 - 2. *Biến số, hằng số và kiểu dữ liệu*
 - 3. *Toán tử và hàm toán học*
 - 4. *Phát biểu điều kiện*
 - 5. *Sử dụng Vòng lặp*
 - 6. *Ứng dụng máy tính bỏ túi*
 - 7. *Ứng dụng Game “Đi tìm ẩn số”*
 - 8. *Ứng dụng tháp hình*
 - 9. *Ứng dụng mô phỏng máy ATM*
 - 10. *Tình huống thực hành có đáp án*

Chương 01:

Nền tảng lập trình JAVA

Phần 02 – Biến, hằng và kiểu dữ liệu

Biến là gì ?

- Biến là một vùng nhớ dùng để lưu trữ giá trị mà giá trị đó có thể thay đổi khi chương trình thực thi.
- Mỗi biến gắn liền với một kiểu dữ liệu và một định danh duy nhất gọi là tên biến

Cú pháp khai báo biến

Cú pháp: **[Kiểu dữ liệu] [Tên biến]**

int, char

1. Khai báo biến
2. Hiển thị biến (in biến)
3. Khởi tạo (gán) giá trị ban đầu cho biến

- Bắt đầu bằng một chữ cái hoặc ký tự gạch dưới (_)
- Chỉ bao gồm các ký tự chữ, ký tự số và ký tự gạch dưới
- Không chứa ký tự khoảng trắng trong tên biến
- Phân biệt chữ hoa và chữ thường
- Không trùng với từ khóa trong JAVA

Hằng số

- Khác với biến, hằng số là giá trị không thể thay đổi được (bất biến)
- Cú pháp: **final** [Kiểu dữ liệu] [Tên hằng số] = [Giá trị]
- Quy tắc đặt tên hằng số tương tự như quy tắc đặt tên cho biến số
- Thống nhất hằng số sẽ được thể hiện ở dạng **chữ in hoa + underscore**

```
final int YOUR_BIRTHDAY = 1989;
```

Kiểu dữ liệu

Ngoài kiểu dữ liệu int và char đã tiếp xúc thì trong JAVA còn có kiểu dữ liệu nào nữa hay không ?

Trong Java **kiểu dữ liệu** được chia thành hai loại:

- Các kiểu dữ liệu nguyên thủy (primitive): byte, short, int, long, float, double, boolean, char và **String**
- Các kiểu dữ liệu tham chiếu (reference): array, class, interface

Kiểu dữ liệu nguyên thủy

Kiểu	Độ dài (bit)	Phạm vi biểu diễn giá trị
byte	8	-128 đến 127
char	16	'\u0000' to '\uffff'
boolean	1	“True” hoặc “False”
short	16	-32768 đến 32767

Kiểu	Độ dài (bit)	Phạm vi biểu diễn giá trị
int	32	-2,147,483,648 đến +2,147,483,648
long	64	-9,223,372,036'854,775,808 đến +9,223,372,036'854,775,808
float	32	-3.40292347E+38 đến +3.40292347E+38
double	64	-1,79769313486231570E+308 đến +1,79769313486231570E+308

Reference <https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html>

Chuyển đổi kiểu dữ liệu

- Nới rộng (widening): chuyển đổi kiểu có kích thước nhỏ sang kiểu có kích thước lớn (không làm mất thông tin)
- Thu hẹp (narrowwing): chuyển đổi kiểu có kích thước lớn sang kiểu có kích thước nhỏ (làm mất thông tin)

```
int var1      = 12;
double var2   = 1.23;
int var3      = var1 + (int)var2;    // Sự thu nhỏ (narrowwing)
double var4   = (double)var1 + var2; // Sự nới rộng (widening)
```

Chương 01:

Nền tảng lập trình JAVA

- 
- 1. *Tổng quan lập trình JAVA*
 - 2. *Biến số, hằng số và kiểu dữ liệu*
 - 3. ***Toán tử và hàm toán học***
 - 4. *Phát biểu điều kiện*
 - 5. *Sử dụng Vòng lặp*
 - 6. *Ứng dụng máy tính bỏ túi*
 - 7. *Ứng dụng Game “Đi tìm ẩn số”*
 - 8. *Ứng dụng tháp hình*
 - 9. *Ứng dụng mô phỏng máy ATM*
 - 10. *Tình huống thực hành có đáp án*

Chương 01:

Nền tảng lập trình JAVA

Phần 03 – Toán tử và hàm toán học

Trong JAVA có các loại toán tử nào ?

- **Toán tử số học** + - * / %
- **Toán tử gán** += -= *= /= %=
- **Toán tử ++ --**
- **Toán tử so sánh** > < >= <= == !=
- **Toán tử logic** && || !
- **Toán tử dịch bit**

Hàm toán học

- Số lớn nhất (**max**), số nhỏ nhất (**min**), căn bậc 2 (**sqrt**), lũy thừa (**pow**)
- Làm tròn số: tròn lên (**ceil**), tròn xuống (**floor**), tròn gần nhất (**round**)
- Phát sinh số ngẫu nhiên trong khoảng 0 đến 1 (**random**)

1. *Tìm số lớn nhất trong 3 số tự nhiên cho trước*
2. *Phát sinh số tự nhiên ngẫu nhiên nằm trong khoảng từ 14 đến 19*

Chương 01:

Nền tảng lập trình JAVA

- 
- 1. *Tổng quan lập trình JAVA*
 - 2. *Biến số, hằng số và kiểu dữ liệu*
 - 3. *Toán tử và hàm toán học*
 - 4. Phát biểu điều kiện**
 - 5. *Sử dụng Vòng lặp*
 - 6. *Ứng dụng máy tính bỏ túi*
 - 7. *Ứng dụng Game “Đi tìm ẩn số”*
 - 8. *Ứng dụng tháp hình*
 - 9. *Ứng dụng mô phỏng máy ATM*
 - 10. *Tình huống thực hành có đáp án*

Chương 01:

Nền tảng lập trình JAVA

Phần 04 – Phát biểu điều kiện

Phát biểu điều kiện là gì ?

- Phát biểu điều kiện (câu điều kiện) là các phát biểu giúp chúng ta thực hiện những hành động khác nhau theo những điều kiện khác nhau
- Phát biểu điều kiện được sử dụng rất thường xuyên đối với bất kỳ ngôn ngữ lập trình nào

Có mấy loại câu điều kiện trong JAVA?

Câu điều kiện IF ... ELSE

- Câu điều kiện IF
- Câu điều kiện IF ... ELSE
- Câu điều kiện IF ... ELSE IF ... ELSE

Câu điều kiện SWITCH

Chương 01:

Nền tảng lập trình JAVA

- 
- 1. *Tổng quan lập trình JAVA*
 - 2. *Biến số, hằng số và kiểu dữ liệu*
 - 3. *Toán tử và hàm toán học*
 - 4. *Phát biểu điều kiện*
 - 5. *Sử dụng Vòng lặp*
 - 6. *Ứng dụng máy tính bỏ túi*
 - 7. *Ứng dụng Game “Đi tìm ẩn số”*
 - 8. *Ứng dụng tháp hình*
 - 9. *Ứng dụng mô phỏng máy ATM*
 - 10. *Tình huống thực hành có đáp án*

Chương 01:

Nền tảng lập trình JAVA

Phần 05 – Sử dụng vòng lặp

Vòng lặp trong JAVA

Sử dụng vòng lặp khi: Một đoạn mã lệnh trong chương trình được thực hiện lặp đi lặp lại cho đến khi thỏa mãn một điều kiện nào đó

Các vòng lặp thường sử dụng trong JAVA

- For
- While
- Do ... While

Break & Continue

Câu lệnh **break** được sử dụng để nhảy ra khỏi một vòng lặp.

Câu lệnh **continue** có chức năng dừng vòng lặp tại giá trị đó và nhảy sang giá trị khác trong vòng lặp

```
while (condition) {  
    continue;  
    break;  
}
```

Chương 01:

Nền tảng lập trình JAVA

- 1. *Tổng quan lập trình JAVA*
- 2. *Biến số, hằng số và kiểu dữ liệu*
- 3. *Toán tử và hàm toán học*
- 4. *Phát biểu điều kiện*
- 5. *Sử dụng Vòng lặp*
- 6. *Ứng dụng máy tính bỏ túi*
- 7. *Ứng dụng Game “Đi tìm ẩn số”*
- 8. *Ứng dụng tháp hình*
- 9. *Ứng dụng mô phỏng máy ATM*
- 10. *Tình huống thực hành có đáp án*

Chương 01:

Nền tảng lập trình JAVA

Phần 06

Xây dựng ứng dụng máy tính bỏ túi

Xây dựng ứng dụng máy tính bỏ túi

Mô phỏng máy tính điện tử

Số thứ nhất

Phép toán

Số thứ hai

Xem kết quả

Kết quả: $25 \% 2 = 1$

Chương 01:

Nền tảng lập trình JAVA

- 
- 1. *Tổng quan lập trình JAVA*
 - 2. *Biến số, hằng số và kiểu dữ liệu*
 - 3. *Toán tử và hàm toán học*
 - 4. *Phát biểu điều kiện*
 - 5. *Sử dụng Vòng lặp*
 - 6. *Ứng dụng máy tính bỏ túi*
 - 7. ***Ứng dụng Game “Đi tìm ẩn số”***
 - 8. *Ứng dụng tháp hình*
 - 9. *Ứng dụng mô phỏng máy ATM*
 - 10. *Tình huống thực hành có đáp án*

Chương 01:

Nền tảng lập trình JAVA

Phần 07
Xây dựng ứng dụng Game “Đi tìm ẩn số”

Xây dựng ứng dụng game “Đi tìm ẩn số”



Chương 01:

Nền tảng lập trình JAVA

- 
- 1. *Tổng quan lập trình JAVA*
 - 2. *Biến số, hằng số và kiểu dữ liệu*
 - 3. *Toán tử và hàm toán học*
 - 4. *Phát biểu điều kiện*
 - 5. *Sử dụng Vòng lặp*
 - 6. *Ứng dụng máy tính bỏ túi*
 - 7. *Ứng dụng Game “Đi tìm ẩn số”*
 - 8. *Ứng dụng tháp hình*
 - 9. *Ứng dụng mô phỏng máy ATM*
 - 10. *Tình huống thực hành có đáp án*

Chương 01:

Nền tảng lập trình JAVA

Phần 08

Sử dụng vòng lặp xử lý các tháp hình

Sử dụng vòng lặp xử lý các tháp hình

```
*
```

```
* *
```

```
* * *
```

```
* * * *
```

```
* * * * *
```

```
* * * * *
```

```
* * * *
```

```
* * *
```

```
*
```

```
*
```

```
* *
```

```
* * *
```

```
* * *
```

```
* * * *
```

```
* * * *
```

```
*
```

```
* * * * *
```

```
* * * *
```

```
* * *
```

```
*
```

```
1
```

```
1 2
```

```
1 2 3
```

```
1 2 3 4
```

```
1 2 3 4 5
```

```
1
```

```
212
```

```
32123
```

```
4321234
```

```
543212345
```

```
* * * * *
```

```
*
```

```
*
```

```
*
```

```
* * * * *
```

```
*
```

```
* * *
```

```
* * * *
```

```
* * * * *
```

```
* * * * *
```

Chương 01:

Nền tảng lập trình JAVA

- 1. *Tổng quan lập trình JAVA*
- 2. *Biến số, hằng số và kiểu dữ liệu*
- 3. *Toán tử và hàm toán học*
- 4. *Phát biểu điều kiện*
- 5. *Sử dụng Vòng lặp*
- 6. *Ứng dụng máy tính bỏ túi*
- 7. *Ứng dụng Game “Đi tìm ẩn số”*
- 8. *Ứng dụng tháp hình*
- 9. *Ứng dụng mô phỏng máy ATM*
- 10. *Tình huống thực hành có đáp án*

Chương 01:

Nền tảng lập trình JAVA

Phần 09: Mô phỏng máy ATM

Mô phỏng máy ATM



Mô phỏng máy ATM

Vui lòng nhập vào số tiền quý khách muốn thực hiện giao dịch

3450000

Rút tiền

Mệnh giá

Mệnh giá 500.000

Mệnh giá 200.000

Mệnh giá 50.000

Số lượng

6

2

1

Thành tiền

3.000.000

400.000

50.000

Tổng tiền: 3.450.000

Chương 01:

Nền tảng lập trình JAVA

- 
- 1. *Tổng quan lập trình JAVA*
 - 2. *Biến số, hằng số và kiểu dữ liệu*
 - 3. *Toán tử và hàm toán học*
 - 4. *Phát biểu điều kiện*
 - 5. *Sử dụng Vòng lặp*
 - 6. *Ứng dụng máy tính bỏ túi*
 - 7. *Ứng dụng Game “Đi tìm ẩn số”*
 - 8. *Ứng dụng tháp hình*
 - 9. *Ứng dụng mô phỏng máy ATM*
 - 10. *Tình huống thực hành có đáp án***

Chương 01:

Nền tảng lập trình JAVA

Phần 10: Tình huống thực hành

The image shows a diamond-shaped arrangement of asterisks (*). There are seven asterisks along each side of the diamond. The pattern is symmetrical, with one asterisk at the top center, two on each of the four sides, and five at the bottom center.

A decorative pattern consisting of a grid of asterisks (*). The pattern is arranged in a roughly triangular shape, with the highest density of stars at the bottom and a single star at the top right.

A decorative pattern consisting of a grid of asterisks (*). The grid has 6 rows and 6 columns. The asterisks are arranged such that they form a central diamond shape with a star-like pattern around it. The pattern is symmetric along both the horizontal and vertical axes.

A vertical column of asterisks used as a decorative separator or list marker. The asterisks are arranged in horizontal rows, with the number of asterisks increasing from one at the top to seven at the bottom.

1 2 3 4 5
1 2 3 4
1 2 3
1 2
1

1
2 1
3 2 1
4 3 2 1
5 4 3 2 1

Chương 02:

Lập trình hướng đối tượng

1. *Class & Object*
2. *Property & Method*
3. *Overloading method*
4. *Constructor*
5. *Inheritance*
6. *Overriding method*
7. *Access Modifier*
8. *Static variable & static method*
9. *Xây dựng class Fraction*
10. *Tình huống thực hành có đáp án*

Chương 02:

Lập trình hướng đối tượng

- 1. Class & Object*
- 2. Property & Method*
- 3. Overloading method*
- 4. Constructor*
- 5. Inheritance*
- 6. Overriding method*
- 7. Access Modifier*
- 8. Static variable & static method*
- 9. Xây dựng class Fraction*
- 10. Tình huống thực hành có đáp án*

Chương 02:

Lập trình hướng đối tượng

Phần 1 – Class & Object

Lập trình hướng thủ tục & hướng đối tượng

Lập trình hướng thủ tục

- Giải quyết vấn đề từng bước đến khi đạt yêu cầu
- Lập trình từ trên xuống
- Lập trình theo hàm → chỉ tạo ra hàm xử lý khi gặp vấn đề nào đó

Lập trình hướng đối tượng

- Dựa trên nền tảng các lớp đã xây dựng sẵn
- Xác định trước các chức năng cần phải thực hiện

Lập trình hướng đối tượng

- OOP - Object oriented programming là kiểu lập trình lấy đối tượng làm nền tảng
- Đơn giản hóa việc phát triển chương trình
- Tạo ra các chương trình có tính mềm dẻo và linh động cao
- Dễ dàng phát triển, bảo trì và nâng cấp.

Phân biệt Class & Object

Class chỉ một cái gì đó chung chung, object là một cái cụ thể

- Công thức làm bánh quy là một Class → bánh quy là một Object
- Con gái là một Class → Hồng là một Object
- Con mèo là một Class → Con mèo Mimi nhà tôi là một Object
- Bản vẽ là một Class → Ngôi nhà của tôi là một Object

Khai báo Class

Tạo class Student nằm trong Main.java

```
package chap02.oop;  
public class Main {  
    public static void main(String[] args) {  
        System.out.println("Main.main");  
    }  
}  
  
class Student {  
}
```

→ Thêm từ khóa public vào trước class Student ?

Khởi tạo đối tượng của Class

Tạo class Student nằm trong Main.java

```
package chap02.oop;  
  
public class Main {  
  
    public static void main(String[] args) {  
  
        Student studentOne      = new Student();  
        Student studentTwo      = new Student();  
  
    }  
}
```

- Tách class Student thành 1 file riêng (cùng package)
- Tách class Student thành 1 file riêng (khác package)

Chương 02:

Lập trình hướng đối tượng

1. *Class & Object*
2. *Property & Method*
3. *Overloading method*
4. *Constructor*
5. *Inheritance*
6. *Overriding method*
7. *Access Modifier*
8. *Static variable & static method*
9. *Xây dựng class Fraction*
10. *Tình huống thực hành có đáp án*

Chương 02:

Lập trình hướng đối tượng

Phần 2 – Property & Method

Hiểu thế nào về Property & Method

Property (Thuộc tính)

- Là các đặc điểm, đặc tính của một lớp

Method (Phương thức)

- Là các hành động có thể được thực hiện từ lớp
- Phương thức cũng giống như hàm, nhưng là hàm riêng của lớp

Ví dụ về phương thức và thuộc tính

Xét Object “Sinh viên Yến Trang”

- Đặc điểm: tóc dài, má lúng đồng tiền, cao 1.7 m, ... → thuộc tính
- Hành động: ăn, ngủ, dạy anh văn, đánh đàn, ... → phương thức

Xét Object “Con cún Ken nhà tôi”

- Đặc điểm: lông xoăn, màu xám, đuôi ngắn, ... → thuộc tính
- Hành động: ăn, ngủ, bắt chuột, ... → phương thức

Khai báo property

```
public class Student {  
    public String name;  
    public String code;  
    public int birthday;  
}
```

Access modifiers
(phạm vi truy xuất)

Data type
(kiểu dữ liệu)

Property
(tên của thuộc tính)

→ Truy cập và gán giá trị cho các thuộc tính

Truy cập và gán giá trị cho các thuộc tính

```
public static void main(String[] args) {  
    Student studentOne      = new Student();  
    studentOne.birthday    = 1994;  
    studentOne.name         = "John";  
    studentOne.code         = "S001";  
  
    System.out.println("birthday: " + studentOne.birthday);  
    System.out.println("name: "      + studentOne.name);  
    System.out.println("code: "      + studentOne.code);  
}
```

Phương thức getter và setter

```
public static void main(String[] args) {  
    Student studentOne = new Student();  
    studentOne.birthday = 1994;  
    studentOne.name = "John";  
    studentOne.code = "S001";  
  
    System.out.println("birthday: " + studentOne.birthday);  
    System.out.println("name: " + studentOne.name);  
    System.out.println("code: " + studentOne.code);  
}
```

setter

getter

Khai báo method

```
public class People {  
    public String name;  
  
    public String getName() {  
        return name;  
    }  
  
    public void setName(String name) {  
        this.name = name;  
    }  
}
```

[access] [data] [method] ([param], [...]) {

 // your code here
}

Chương 02:

Lập trình hướng đối tượng

1. *Class & Object*
2. *Property & Method*
- 3. *Overloading method***
4. *Constructor*
5. *Inheritance*
6. *Overriding method*
7. *Access Modifier*
8. *Static variable & static method*
9. *Xây dựng class Fraction*
10. *Tình huống thực hành có đáp án*

Chương 02:

Lập trình hướng đối tượng

Phần 3 – Overloading Method

Overloading method (nạp chồng phương thức)

Overloading method là những phương thức nằm trong cùng một class có cùng tên nhưng khác các tham số (khác kiểu dữ liệu, khác số lượng tham số)

```
public void setCode(String code) { }  
public void setCode() { }  
public void setCode(String first, int last) { }  
public void setCode(int code) { }
```

Chương 02:

Lập trình hướng đối tượng

1. *Class & Object*
2. *Property & Method*
3. *Overloading method*
- 4. Constructor**
5. *Inheritance*
6. *Overriding method*
7. *Access Modifier*
8. *Static variable & static method*
9. *Xây dựng class Fraction*
10. *Tình huống thực hành có đáp án*

Chương 02:

Lập trình hướng đối tượng

Phần 4 – Constructor

Constructor (phương thức khởi tạo)

- Constructor **được gọi tự động** và được gọi đầu tiên khi một object được khởi tạo
- Constructor không có giá trị trả về, có thể có tham số hoặc không có
- Constructor phải có **cùng tên với lớp**
- Constructor **có thể bị nạp chồng** (overloading)
- Nếu một class chưa khai báo constructor thì sẽ được JAVA cung cấp một constructor **mặc định** (default constructor).

Constructor (phương thức khởi tạo)

```
public class Student{  
    public String name;  
  
    public Student(){  
        System.out.println("Constructor 1");  
        this.setName("John");  
    }  
  
    public Student(String name){  
        System.out.println("Constructor 2");  
        this.setName(name);  
    }  
}
```

Chương 02:

Lập trình hướng đối tượng

1. *Class & Object*
2. *Property & Method*
3. *Overloading method*
4. *Constructor*
5. *Inheritance*
6. *Overriding method*
7. *Access Modifier*
8. *Static variable & static method*
9. *Xây dựng class Fraction*
10. *Tình huống thực hành có đáp án*

Chương 02:

Lập trình hướng đối tượng

Phần 5 – Inheritance

Tình huống

Student

```
+ name  
+ code  
+ score  
  
getName  
setName (String name)  
getCode ()  
setCode (String code)  
showInfo ()  
getScore ()  
setScore (Double score)
```

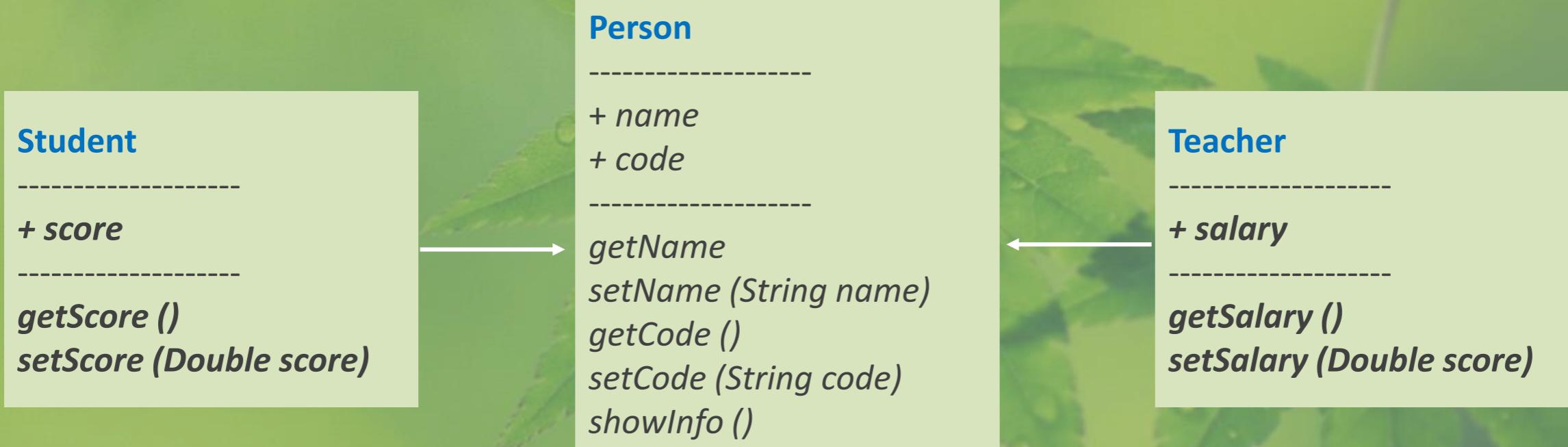
Person

```
+ name  
+ code  
  
getName  
setName (String name)  
getCode ()  
setCode (String code)  
showInfo ()
```

Teacher

```
+ name  
+ code  
+ salary  
  
getName  
setName (String name)  
getCode ()  
setCode (String code)  
showInfo ()  
getSalary ()  
setSalary (Double score)
```

Tình huống



Inheritance (sự kế thừa)

Student kế thừa từ Person được biểu diễn qua từ khóa extends

```
class Student extends Person{  
}
```

- Student gọi là class con (subclass), Person gọi là class cha (superclass)
- Student nếu đã kế thừa từ Person thì không thể kế thừa thêm lớp khác

Chống kế thừa

```
final public class Person{  
}
```

- Từ khóa final cho biết Person là lớp hằng → không lớp nào có thể kế thừa từ lớp Person được nữa

Chương 02:

Lập trình hướng đối tượng

- 1. *Class & Object*
- 2. *Property & Method*
- 3. *Overloading method*
- 4. *Constructor*
- 5. *Inheritance*
- 6. ***Overriding method***
- 7. *Access Modifier*
- 8. *Static variable & static method*
- 9. *Xây dựng class Fraction*
- 10. *Tình huống thực hành có đáp án*

Chương 02:

Lập trình hướng đối tượng

Phần 6 – Overriding

Overriding (sự ghi đè)

- Phương thức đã xuất hiện ở lớp cha và xuất hiện tiếp ở lớp con.
- Khi đối tượng thuộc lớp con gọi phương thức thì sẽ chọn lựa và chạy theo phương thức trong lớp con.
- Nếu lớp con không có phương thức đó thì mới gọi phương thức đó ở lớp cha

Overloading vs Overriding

Overloading (nạp chồng)

- Xuất hiện: trong cùng lớp
- Tên phương thức: giống nhau
- Số lượng tham số và kiểu dữ liệu của tham số: có thể khác nhau
- Đa hình (polymorphism) trong quá trình biên dịch

Overriding (ghi đè)

- Xuất hiện: ở class cha và class con
- Tên phương thức: giống nhau
- Số lượng tham số và kiểu dữ liệu của tham số: giống nhau
- Đa hình (polymorphism) trong quá trình thực thi

Chương 02:

Lập trình hướng đối tượng

1. *Class & Object*
2. *Property & Method*
3. *Overloading method*
4. *Constructor*
5. *Inheritance*
6. *Overriding method*
7. ***Access Modifier***
8. *Static variable & static method*
9. *Xây dựng class Fraction*
10. *Tình huống thực hành có đáp án*

Chương 02:

Lập trình hướng đối tượng

Phần 7 – Access modifier

Access modifier

Access modifier cấp độ truy cập cho class, *property* và method

```
public class Student {}
```

- **Private** chỉ truy cập được trong class *(property, method)*
- **Null** (rỗng) truy cập trong package *(class, property, method)*
- **Protected** truy cập trong package và các subclasses *(property, method)*
- **Public** truy cập từ bất kỳ đâu *(class, property, method)*

Chương 02:

Lập trình hướng đối tượng

1. *Class & Object*
2. *Property & Method*
3. *Overloading method*
4. *Constructor*
5. *Inheritance*
6. *Overriding method*
7. *Access Modifier*
- 8. *Static variable & static method***
9. *Xây dựng class Fraction*
10. *Tình huống thực hành có đáp án*

Chương 02:

Lập trình hướng đối tượng

Phần 8 – Static variable & Static method

Static keyword

- **Static keyword** được sử dụng chủ yếu trong việc quản lý bộ nhớ
- Static có thể được áp dụng cho:
 - *variable (class variable)*
 - *method (class method)*
 - block
 - nested class

Static variable

```
public class Student{  
    private String name;  
    private String college = "CTU";  
}
```

name = Peter
college = CTU

name = John
college = CTU

name = Mary
college = CTU

college = CTU

name = Peter

name = John

name = Mary

Static method

- Static method thuộc về lớp không thuộc về đối tượng
- Static method được gọi mà không cần tạo một thể hiện của lớp.
- Static method có thể truy cập và thay đổi giá trị của static variable
- Các property và method không thỏa static thì static method không có quyền truy cập vào

Chương 02:

Lập trình hướng đối tượng

1. *Class & Object*
2. *Property & Method*
3. *Overloading method*
4. *Constructor*
5. *Inheritance*
6. *Overriding method*
7. *Access Modifier*
8. *Static variable & static method*
9. **Xây dựng class Fraction**
10. *Tình huống thực hành có đáp án*

Chương 02:

Lập trình hướng đối tượng

Phần 9 – Xây dựng class Fraction

Class Fraction

- Nhập phân số (tử số và mẫu số)
- In phân số
- Kiểm tra phân số tối giản
- Tối giản phân số
- Thực hiện phép cộng 2 phân số
- Thực hiện phép trừ 2 phân số
- Thực hiện phép nhân 2 phân số
- Thực hiện phép chia 2 phân số

Class Fraction

G Fraction

- numerator : int
- denominator : int
- ^c Fraction(int, int)
- ^c Fraction(Fraction, Fraction, String)
- print() : String
- normalize() : void
- add(Fraction, Fraction) : void
- sub(Fraction, Fraction) : void
- multiply(Fraction, Fraction) : void
- divide(Fraction, Fraction) : void
- UCLN(int, int) : int
- checkNormalize() : boolean
- getNumerator() : int
- setNumerator(int) : void
- getDenominator() : int
- setDenominator(int) : void

Thao tác trên hai phân số

Tối giản phân số

...

Cộng hai phân số

Tìm UCLN của hai số

...

Kiểm tra phân số tối giản

Chương 02:

Lập trình hướng đối tượng

- 
- 1. *Class & Object*
 - 2. *Property & Method*
 - 3. *Overloading method*
 - 4. *Constructor*
 - 5. *Inheritance*
 - 6. *Overriding method*
 - 7. *Access Modifier*
 - 8. *Static variable & static method*
 - 9. *Xây dựng class Fraction*
 - 10. *Tình huống thực hành có đáp án***

Chương 02:

Lập trình hướng đối tượng

Phần 10 – Tình huống thực hành

Xây dựng class Book

- Nhập thông tin một quyển sách (id, tên và giá tiền của quyển sách đó)
- In thông tin quyển sách
- Cập nhật thông tin quyển sách

```
===== BOOK MANAGER =====
1. Add book
2. Edit book
3. Info book
4. Exit
Your choise [1-4]:
```

Chương 03:

Kỹ thuật xử lý mảng

1. *Khai báo mảng và truy cập mảng*
2. *Thao tác trên mảng một chiều*
3. *Sử dụng Class Arrays*
4. *Mảng đa chiều và các thao tác trên mảng*
5. *Tình huống thực hành có đáp án*

Chương 03:

Kỹ thuật xử lý mảng

- 1. Khai báo mảng và truy cập mảng*
- 2. Thao tác trên mảng một chiều*
- 3. Sử dụng Class Arrays*
- 4. Mảng đa chiều và các thao tác trên mảng*
- 5. Tình huống thực hành có đáp án*

Chương 03:

Kỹ thuật xử lý mảng

Phần 1 – Khai báo và truy cập mảng

Tình huống và khái niệm về mảng

- Lưu trữ tên của một sinh viên nào đó → variable **studentA** → lưu trữ tên của 100 sinh viên → 100 biến **studentX**
- Một biến thông thường chỉ chứa một giá trị duy nhất, nếu chúng ta muốn chứa nhiều giá trị trong một biến thì biến đó phải là một mảng
- Mảng là một biến đặc biệt và có thể lưu trữ nhiều giá trị khác nhau có cùng một kiểu dữ liệu

Các cách khai báo mảng

- **int** arrOne[];
- **int[]** arrInt ;
- **int[]** arrInt = new int[5];
- **int** arrInt[] = {30, 23, 31, 36, 34};
- **String** arrString[] = {"Zend 2", "PHP", "Java", "Javascript"};

[0]	[1]	[2]	[3]	
arrString[] →	Zend 2	PHP	Java	Javascript

Chương 03:

Kỹ thuật xử lý mảng

1. *Khai báo mảng và truy cập mảng*
2. ***Thao tác trên mảng một chiều***
3. *Sử dụng Class Arrays*
4. *Mảng đa chiều và các thao tác trên mảng*
5. *Tình huống thực hành có đáp án*

Chương 03:

Kỹ thuật xử lý mảng

Phần 2 – Thao tác trên mảng một chiều

Các thao tác trên mảng

- Truy cập vào các phần tử trên mảng
- Duyệt mảng với vòng lặp for
- Nhập mảng
- Tính tổng các phần tử trong mảng Number
- Tìm giá trị min, max trong mảng Number

Chương 03:

Kỹ thuật xử lý mảng

1. *Khai báo mảng và truy cập mảng*
2. *Thao tác trên mảng một chiều*
3. **Sử dụng Class Arrays**
4. *Mảng đa chiều và các thao tác trên mảng*
5. *Tình huống thực hành có đáp án*

Chương 03:

Kỹ thuật xử lý mảng

Phần 3 – Sử dụng class Arrays

Sử dụng `java.util.Arrays`

- Sao chép mảng `Arrays.copyOf`
- Sao chép mảng có chọn lọc `Arrays.copyOfRange`
- In mảng `Arrays.toString`
- Sắp xếp mảng `Arrays.sort` (tang dần, giảm dần)

Chương 03:

Kỹ thuật xử lý mảng

1. *Khai báo mảng và truy cập mảng*
2. *Thao tác trên mảng một chiều*
3. *Sử dụng Class Arrays*
4. ***Mảng đa chiều và các thao tác trên mảng***
5. *Tình huống thực hành có đáp án*

Chương 03:

Kỹ thuật xử lý mảng

Phần 4

Mảng đa chiều và các thao tác trên mảng

Mảng đa chiều

- Mảng đa chiều (mảng nhiều chiều) là một mảng mà các phần tử trong mảng là các mảng khác.
- Mảng hai chiều là mảng được sử dụng nhiều nhất trong số các mảng nhiều chiều. Mảng hai chiều có số hàng và số cột xác định (thường áp dụng biểu diễn cho các ma trận)
- `int[][] arrMulti = new int[2][3]; // 2 hàng 3 cột`

Mảng đa chiều

```
int[][] arrMulti = { { 2, 9, 1 }, { 3, 6, 7 } };
```

arrMulti [0] [0] = 2;

arrMulti [0] [1] = 9;

arrMulti [0] [2] = 1;

	[0]	[1]	[2] ←
→ [0]	2	9	1
[1]	3	6	7

arrMulti [1] [0] = 3;

arrMulti [1] [1] = 6;

arrMulti [1] [2] = 7;

Các thao tác trên mảng đa chiều

- Truy cập vào các phần tử trên mảng đa chiều
- Duyệt mảng với vòng lặp for
- Nhập và in ma trận
- Tính tổng các phần tử trên một dòng nào đó của ma trận
- Tính tổng các phần tử trên một cột nào đó của ma trận
- Tính tổng các phần tử trên đường chéo của ma trận vuông
- Tìm phần tử lớn nhất trong ma trận

Chương 03:

Kỹ thuật xử lý mảng

1. *Khai báo mảng và truy cập mảng*
2. *Thao tác trên mảng một chiều*
3. *Sử dụng Class Arrays*
4. *Mảng đa chiều và các thao tác trên mảng*
5. *Tình huống thực hành có đáp án*

Chương 03:

Kỹ thuật xử lý mảng

Phần 5 – Tình huống thực hành

Thao tác trên ma trận vuông

Xuất các phần tử nằm phía trên (phía dưới) đường chéo chính

Các phần tử nằm trên đường chéo chính: 2 - 6 - 8

→ Các phần tử nằm phía trên đường chéo chính: 9 - 1 - 7

→ Các phần tử nằm phía dưới đường chéo chính: 3 - 4 - 1

Xây dựng giải thuật để nhập ma trận vuông và thực hiện hai yêu cầu trên

2	9	1
3	6	7
4	1	8

Thao tác trên ma trận

Tìm các phần tử lớn nhất ở mỗi dòng và tính tổng các phần tử đó

Ma trận có 3 dòng:

- Dòng 1 có phần tử lớn nhất: 9
- Dòng 1 có phần tử lớn nhất: 7
- Dòng 1 có phần tử lớn nhất: 8

$$\text{Tổng: } 9 + 7 + 8 = 26$$

2	9	1
3	6	7
4	1	8

Xây dựng giải thuật để nhập ma trận và thực hiện yêu cầu trên

Thao tác trên ma trận

Sắp xếp ma trận tăng dần

2	9	1
3	6	7
4	1	8



1	1	2
3	4	6
7	8	9

Xây dựng giải thuật để nhập ma trận và thực hiện yêu cầu trên

Chương 04:

Xử lý ngoại lệ

1. *Exception và xử lý ngoại lệ*
2. *Sử dụng Try – catch – finally*
3. *Từ khóa throws và throw*
4. *Xây dựng ứng dụng BookStore Version 3*

Exception là gì ?

- **Exception** các bất thường xảy ra khi chương trình thực thi
- **Exception** nếu không được xử lý tốt sẽ làm cho chương trình bị dừng đột ngột, mất dữ liệu, ảnh hưởng đến các chương trình khác
- Các nguyên nhân gây ra Exception: input không phù hợp, đọc file khi file không tồn tại, tràn bộ nhớ, ...

Sử dụng Try ... catch ... finally

```
try {  
    // khôi lệnh nghi ngờ exception xảy ra  
} catch (ArithmeticException e) {  
    System.out.println("Error: " + e);  
} catch (ArrayIndexOutOfBoundsException e) {  
    System.out.println("Error: " + e);  
} catch (Exception e) {  
    System.out.println("Error: " + e);  
} finally {  
    // Khôi lệnh luôn được thực thi  
}
```

Các class thuộc Exception

- ClassNotFoundException | Không tìm thấy Class
- FileNotFoundException | Không tìm thấy file
- ArrayIndexOutOfBoundsException | Vượt quá chỉ mục của mảng
- ArithmeticException | Lỗi thực thi một phép toán
- NumberFormatException | Định dạng số không đúng
- NoSuchMethodException | Sai tên phương thức
- ...

Chương 05:

Kỹ thuật xử lý chuỗi

- 
- 1. *Kiểu dữ liệu String*
 - 2. *Sử dụng class StringBuilder*
 - 3. *Kỹ thuật xử lý chuỗi*
 - 4. *Regular Expression*
 - 5. *Kiểm tra sự hợp lệ của dữ liệu với Regular Expression*
 - 6. *Kỹ thuật tìm kiếm và thay thế chuỗi*
 - 7. *Kỹ thuật bóc tách chuỗi*
 - 8. *Ứng dụng chuẩn hóa chuỗi*
 - 9. *Ứng dụng chuyển số thành chữ*
 - 10. *Tình huống thực hành có đáp án*
 - 11. *BookStore Version 4*

Chương 05:

Kỹ thuật xử lý chuỗi

- 
- 1. Kiểu dữ liệu String**
 - 2. Sử dụng class StringBuilder**
 - 3. Kỹ thuật xử lý chuỗi**
 - 4. Regular Expression**
 - 5. Kiểm tra sự hợp lệ của dữ liệu với Regular Expression**
 - 6. Kỹ thuật tìm kiếm và thay thế chuỗi**
 - 7. Kỹ thuật bóc tách chuỗi**
 - 8. Ứng dụng chuẩn hóa chuỗi**
 - 9. Ứng dụng chuyển số thành chữ**
 - 10. Tình huống thực hành có đáp án**
 - 11. BookStore Version 4**

Kiểu dữ liệu String

- *Tìm hiểu kiểu dữ liệu String*
- *Các trường hợp khởi tạo đối tượng kiểu String*
- *Chiều dài của chuỗi*
- *Nối chuỗi*
- *So sánh chuỗi == hoặc equals hoặc equalsIgnoreCase*
- *Tạo ra vùng nhớ rác*

Chương 05:

Kỹ thuật xử lý chuỗi

- 
- 1. *Kiểu dữ liệu String*
 - 2. *Sử dụng class StringBuilder*
 - 3. *Kỹ thuật xử lý chuỗi*
 - 4. *Regular Expression*
 - 5. *Kiểm tra sự hợp lệ của dữ liệu với Regular Expression*
 - 6. *Kỹ thuật tìm kiếm và thay thế chuỗi*
 - 7. *Kỹ thuật bóc tách chuỗi*
 - 8. *Ứng dụng chuẩn hóa chuỗi*
 - 9. *Ứng dụng chuyển số thành chữ*
 - 10. *Tình huống thực hành có đáp án*
 - 11. *BookStore Version 4*

Sử dụng class StringBuilder

- *Thao tác trên các kiểu dữ liệu String và không tạo ra vùng nhớ rác*
- *Các phương thường thông dụng*
 - *append*
 - *insert*
 - *deleteCharAt*
 - *delete*

Chương 05:

Kỹ thuật xử lý chuỗi

- 
- 1. *Kiểu dữ liệu String*
 - 2. *Sử dụng class StringBuilder*
 - 3. Kỹ thuật xử lý chuỗi**
 - 4. *Regular Expression*
 - 5. *Kiểm tra sự hợp lệ của dữ liệu với Regular Expression*
 - 6. *Kỹ thuật tìm kiếm và thay thế chuỗi*
 - 7. *Kỹ thuật bóc tách chuỗi*
 - 8. *Ứng dụng chuẩn hóa chuỗi*
 - 9. *Ứng dụng chuyển số thành chữ*
 - 10. *Tình huống thực hành có đáp án*
 - 11. *BookStore Version 4*

Kỹ thuật xử lý chuỗi cơ bản

- ***length()*** *Trả về độ dài chuỗi*
- ***charAt()*** *Lấy ra ký tự nằm ở vị trí index trong chuỗi*
- ***toLowerCase()*** *Chuyển chuỗi sang chữ thường*
- ***toUpperCase()*** *Chuyển chuỗi sang chữ in hoa*
- ***trim()*** *Loại bỏ khoảng trắng 2 bên chuỗi*
- ***substring()*** *Trích xuất chuỗi con*

Chương 05:

Kỹ thuật xử lý chuỗi

- 
- 1. *Kiểu dữ liệu String*
 - 2. *Sử dụng class StringBuilder*
 - 3. *Kỹ thuật xử lý chuỗi*
 - 4. Regular Expression**
 - 5. *Kiểm tra sự hợp lệ của dữ liệu với Regular Expression*
 - 6. *Kỹ thuật tìm kiếm và thay thế chuỗi*
 - 7. *Kỹ thuật bóc tách chuỗi*
 - 8. *Ứng dụng chuẩn hóa chuỗi*
 - 9. *Ứng dụng chuyển số thành chữ*
 - 10. *Tình huống thực hành có đáp án*
 - 11. *BookStore Version 4*

Regular Expression là gì ?

- Regular expression (RE - biểu thức chính quy) bắt nguồn từ ngôn ngữ Perl, xuất hiện và được ứng dụng rất nhiều trong các ngôn ngữ lập trình
- Regular expression có thể hiểu đơn giản là một biểu thức dùng để mô tả một mẫu chuỗi nào đó (mẫu: các chuỗi văn bản tuân theo một quy luật sắp xếp nhất định).
 - dd/mm/yyyy là mẫu mô tả ngày tháng năm
 - xxxx@gmail.com là mẫu mô tả địa chỉ email

Regular Expression làm được gì ?

- *Biểu thức chính quy giúp giảm bớt những dòng mã trong quá trình xử lý chuỗi bằng những biểu thức ngắn gọn nhưng đem lại kết quả như mong đợi*
- *Regular expression thường được sử dụng trong các trường hợp*
 - *Kiểm tra giá trị các phần tử của Form*
 - *Xử lý các yêu cầu phức tạp trong chuỗi (bóc tách, thay đổi nội dung, loại bỏ ký tự, ...)*

Ký hiệu thường dùng trong RE

STT	Ký hiệu	Ý nghĩa
1	^	Tìm kiếm giá trị từ đầu chuỗi nguồn
2	\$	Tìm kiếm giá trị từ cuối chuỗi nguồn
3	.	Đại diện cho một ký tự bất kỳ
4	\	Tìm kiếm các giá trị đặt biệt trong chuỗi (^ \$. / @)
5	[]	Tìm tập hợp các ký tự
6	-	Lấy các ký tự trong một khoảng nào đó (thường dùng với ký hiệu [])
7	[^]	Nếu một ký tự ^ đứng trước một ký tự hay tập hợp có nghĩa là phủ định của ký hiệu hay tập hợp đó
8	(A B C)	Lựa chọn các giá trị A hoặc B hoặc C

Ký hiệu thường dùng trong RE

STT	Ký hiệu	Ý nghĩa
9	*	0 hoặc nhiều lần xuất hiện
10	+	1 hoặc nhiều lần xuất hiện
11	?	0 hoặc 1 lần xuất hiện
12	{n,m}	Số lần xuất hiện của ký tự từ n đến m lần
13	*?	0 lần xuất hiện
14	+?	1 lần xuất hiện
15	??	0 lần xuất hiện
16	\w	Tương đương [A-zo-9_]
17	\W	Tương đương [^A-zo-9_]

Ký hiệu thường dùng trong RE

STT	Ký hiệu	Ý nghĩa
18	\s	Tập hợp những ký tự khoảng trắng
19	\S	Tập hợp những ký tự không là ký tự khoảng trắng
20	\d	[0-9] Tập hợp những ký tự từ 0 đến 9
21	\D	[^0-9] Tập hợp những ký tự không thuộc từ 0 đến 9
22	\A	Tìm kiếm giá trị từ đầu chuỗi nguồn
23	\Z	Tìm kiếm giá trị từ cuối chuỗi nguồn

Kiểm tra ID hợp lệ

1. *ID phải có chiều dài là 7 ký tự XXX-YYY*
2. *XXX phải nằm trong tập hợp các giá trị ký tự A-Z hoặc a-z*
3. *YYY phải là các giá trị số từ 2 đến 8*

Pattern: [A-z]{3}-[2-8]{3}

Kiểm tra tên đăng nhập hợp lệ

1. *Tên đăng nhập phải bắt đầu bằng một ký tự hoặc dấu gạch dưới*
2. *Tên đăng nhập là tập hợp của các ký tự a-z, 0-9 và có thể có các ký tự như dấu chấm (.), dấu gạch dưới (_)*
3. *Độ dài tối thiểu của tên đăng nhập là 5 ký tự và độ dài tối đa là 32 ký tự*

Pattern [A-Za-z_][A-Za-z0-9_\.]{4,31}

Kiểm tra địa chỉ website hợp lệ

1. <http://www zend vn>
2. <https://www zend vn vn>
3. <http://zend vn>
4. <https://zend vn>
5. <www zend vn>

Pattern (([//\(www\.\)?|www\.\)\[a-zA-Z\-\-\]{3,20}\\(\[a-zA-Z\]{2,4}\\){1,2}](https://))

Chương 05:

Kỹ thuật xử lý chuỗi

- 
- 1. *Kiểu dữ liệu String*
 - 2. *Sử dụng class StringBuilder*
 - 3. *Kỹ thuật xử lý chuỗi*
 - 4. *Regular Expression*
 - 5. *Kiểm tra sự hợp lệ của dữ liệu với Regular Expression*
 - 6. *Kỹ thuật tìm kiếm và thay thế chuỗi*
 - 7. *Kỹ thuật bóc tách chuỗi*
 - 8. *Ứng dụng chuẩn hóa chuỗi*
 - 9. *Ứng dụng chuyển số thành chữ*
 - 10. *Tình huống thực hành có đáp án*
 - 11. *BookStore Version 4*

Kiểm tra sự hợp lệ của dữ liệu với RE

Sử dụng phương thức matches để kiểm tra chuỗi có trùng khớp với mẫu RE hay không ?

Ví dụ

- Kiểm tra số tự nhiên hợp lệ
- Kiểm tra ID hợp lệ
- Kiểm tra username hợp lệ
- Kiểm tra địa chỉ website hợp lệ

Chương 05:

Kỹ thuật xử lý chuỗi

- 
- 1. *Kiểu dữ liệu String*
 - 2. *Sử dụng class StringBuilder*
 - 3. *Kỹ thuật xử lý chuỗi*
 - 4. *Regular Expression*
 - 5. *Kiểm tra sự hợp lệ của dữ liệu với Regular Expression*
 - 6. ***Kỹ thuật tìm kiếm và thay thế chuỗi***
 - 7. *Kỹ thuật bóc tách chuỗi*
 - 8. *Ứng dụng chuẩn hóa chuỗi*
 - 9. *Ứng dụng chuyển số thành chữ*
 - 10. *Tình huống thực hành có đáp án*
 - 11. *BookStore Version 4*

Kỹ thuật tìm kiếm và thay thế chuỗi

- *replace* không kết hợp được với *RE*
- *replaceAll* kết hợp được với *RE*

Bài tập áp dụng

- *Java is vory vory oasy!* → *Java is very very easy!*
- *khoa hoc lap trinh java* → *khoa hoc lap trinh java*
- *abcdAB69CD!@#\$%^&*()0<>?_{ }+21* → *69021*

Chương 05:

Kỹ thuật xử lý chuỗi

- 
- 1. *Kiểu dữ liệu String*
 - 2. *Sử dụng class StringBuilder*
 - 3. *Kỹ thuật xử lý chuỗi*
 - 4. *Regular Expression*
 - 5. *Kiểm tra sự hợp lệ của dữ liệu với Regular Expression*
 - 6. *Kỹ thuật tìm kiếm và thay thế chuỗi*
 - 7. *Kỹ thuật bóc tách chuỗi*
 - 8. *Ứng dụng chuẩn hóa chuỗi*
 - 9. *Ứng dụng chuyển số thành chữ*
 - 10. *Tình huống thực hành có đáp án*
 - 11. *BookStore Version 4*

Kỹ thuật bóc tách chuỗi

- *token* chuỗi có giá trị xác định sau khi được bóc tách
- *delimiter* ký tự xác định dùng để bóc tách chuỗi

Ví dụ

- *Java_is_easy!*
 - *Token: java, is, easy!* *Delimiter: khoảng trắng*
- *Java--is--easy!*
 - *Token: java, is, easy!* *Delimiter: --*

Kỹ thuật bóc tách chuỗi

- Scanner: *useDelimiter*, *hasNext*, *next*
- *split*
- *StringTokenizer*

Ví dụ

- *Java_is_very_very_easy*
- *Java--is--very--very--easy*
- *Java--is_very_very--easy*
- *Java_12_is_233__very_32_very__156_easy*

Chương 05:

Kỹ thuật xử lý chuỗi

- 
- 1. *Kiểu dữ liệu String*
 - 2. *Sử dụng class StringBuilder*
 - 3. *Kỹ thuật xử lý chuỗi*
 - 4. *Regular Expression*
 - 5. *Kiểm tra sự hợp lệ của dữ liệu với Regular Expression*
 - 6. *Kỹ thuật tìm kiếm và thay thế chuỗi*
 - 7. *Kỹ thuật bóc tách chuỗi*
 - 8. *Ứng dụng chuẩn hóa chuỗi***
 - 9. *Ứng dụng chuyển số thành chữ*
 - 10. *Tình huống thực hành có đáp án*
 - 11. *BookStore Version 4*

Ứng dụng chuẩn hóa chuỗi

Chuỗi đạt chuẩn dạng 1.

- *Không có khoảng trắng ở đầu và cuối chuỗi*
- *Giữa các từ trong chuỗi chỉ tồn tại một khoảng trắng duy nhất*
- *Ký tự đầu tiên trong chuỗi phải là ký tự in hoa*
- *Các ký tự còn lại phải ở dạng chữ thường*
- *Hết câu bằng dấu chấm*

Ứng dụng chuẩn hóa chuỗi

Chuỗi Đạt Chuẩn Dạng 2.

- *Không có khoảng trắng ở đầu và cuối chuỗi*
- *Giữa các từ trong chuỗi chỉ tồn tại một khoảng trắng duy nhất*
- *Ký tự đầu tiên trong chuỗi phải là ký tự in hoa,*
- *Hết câu bằng dấu chấm*
- *Các ký tự đầu tiên ở mỗi từ phải viết được viết hoa*

Chương 05:

Kỹ thuật xử lý chuỗi

- 
- 1. *Kiểu dữ liệu String*
 - 2. *Sử dụng class StringBuilder*
 - 3. *Kỹ thuật xử lý chuỗi*
 - 4. *Regular Expression*
 - 5. *Kiểm tra sự hợp lệ của dữ liệu với Regular Expression*
 - 6. *Kỹ thuật tìm kiếm và thay thế chuỗi*
 - 7. *Kỹ thuật bóc tách chuỗi*
 - 8. *Ứng dụng chuẩn hóa chuỗi*
 - 9. *Ứng dụng chuyển số thành chữ*
 - 10. *Tình huống thực hành có đáp án*
 - 11. *BookStore Version 4*

Ứng dụng đọc số có 3 chữ số

- $976 = Chín trăm bảy mươi sáu$
- $206 = Hai trăm linh sáu$
- $115 = Một trăm mười lăm$
- $291 = Hai trăm chín mươi một$

Chương 05:

Kỹ thuật xử lý chuỗi

- 
- 1. *Kiểu dữ liệu String*
 - 2. *Sử dụng class StringBuilder*
 - 3. *Kỹ thuật xử lý chuỗi*
 - 4. *Regular Expression*
 - 5. *Kiểm tra sự hợp lệ của dữ liệu với Regular Expression*
 - 6. *Kỹ thuật tìm kiếm và thay thế chuỗi*
 - 7. *Kỹ thuật bóc tách chuỗi*
 - 8. *Ứng dụng chuẩn hóa chuỗi*
 - 9. *Ứng dụng chuyển số thành chữ*
 - 10. Tình huống thực hành có đáp án**
 - 11. *BookStore Version 4*

Kiểm tra email hợp lệ

1. Địa chỉ email phải bắt đầu bằng một ký tự.
2. Địa chỉ email là tập hợp của các ký tự a-z, 0 đến 9 và có thể có các ký tự như dấu chấm (.), dấu gạch dưới (_)
3. Độ dài tối thiểu của email là 5 ký tự và độ dài tối đa là 32 ký tự
4. Tên miền của email có thể là tên miền cấp 1 hoặc tên miền cấp 2

Pattern: ^[a-zA-Z][a-zA-Z0-9_\.]{4,31}@{1,}[a-zA-Z]{2,}(\.{1,}[a-zA-Z]{2,4}){1,2}\\$

Đọc số nhỏ hơn 999

- *9 chín*
- *12 mươi hai*
- *97 chín mươi bảy*
- *875 tám trăm bảy mươi lăm*

Chương 05:

Kỹ thuật xử lý chuỗi

- 
- 1. *Kiểu dữ liệu String*
 - 2. *Sử dụng class StringBuilder*
 - 3. *Kỹ thuật xử lý chuỗi*
 - 4. *Regular Expression*
 - 5. *Kiểm tra sự hợp lệ của dữ liệu với Regular Expression*
 - 6. *Kỹ thuật tìm kiếm và thay thế chuỗi*
 - 7. *Kỹ thuật bóc tách chuỗi*
 - 8. *Ứng dụng chuẩn hóa chuỗi*
 - 9. *Ứng dụng chuyển số thành chữ*
 - 10. *Tình huống thực hành có đáp án*
 - 11. ***BookStore Version 4***

Chương 06:

Lập trình hướng đối tượng nâng cao

1. *Abstract class*
2. *Interface*
3. *So sánh Abstract và Interface*
4. *Sử dụng class Object*
5. *Giảm thiểu sự phụ thuộc giữa các class*
6. *Inner Class trong Java*
7. *Xây dựng ứng BookStore Version 5*

Abstract class

- *Abstract class* (lớp trừu tượng) là class chứa bộ khung cơ bản, với các thuộc tính, các phương thức và các phương thức trừu tượng (*abstract method* phương thức chưa hoàn thiện: chỉ có tên, kiểu trả về, tham số)
- Các lớp *extends* từ lớp *abstract* sẽ có nhiệm vụ hoàn thiện các *abstract method*
- *Abstract class* thường được xây dựng bởi kỹ sư thiết kế hệ thống có kinh nghiệm

Interface

- *Interface không phải là class, đây là một mẫu giao diện quy định một số phương thức bắt buộc cho một class nào đó*
- *Interface không cho phép định nghĩa rõ cách hoạt động của phương thức, chỉ dùng lại ở khai báo phương thức, việc định nghĩa các phương thức này sẽ được thực hiện ở các class con*

Sử dụng default và static trong Interface

- Tạo mới một phương thức trong Interface → định nghĩa cách hoạt động của phương thức này ở các class implements từ Interface đó.
- Sử dụng từ khóa default hoặc static để khi tạo mới một phương thức thì không cần phải định nghĩa ở tất cả các class implements từ Interface.
- Sử dụng default method thì cần khởi tạo đối tượng <> Sử dụng static method thì không cần khởi tạo đối tượng

Abstract Class vs Interface

Những điểm giống nhau

- *Chứa các abstract methods*
- *Không khởi tạo trực tiếp đối tượng mà phải thông qua các class con*

Abstract Class vs Interface

Abstract class

- + Là một lớp
- + Mỗi method **không bắt buộc** phải được định nghĩa ở lớp con
- + Mỗi lớp con chỉ kế thừa từ một lớp abstract, một lớp abstract có nhiều lớp con
- + Mỗi phương thức abstract có thể ở trạng thái *protected* hoặc *public*

Interface

- + Không phải lớp
- + Mỗi method **bắt buộc** phải được định nghĩa ở lớp con
- + Mỗi lớp con kế thừa từ một hoặc nhiều Interface, một Interface có nhiều lớp con
- + Mỗi phương thức abstract chỉ ở trạng thái *public*

Sử dụng class Object

- Tất cả các class trong Java (có sẵn hay do lập trình viên định nghĩa) đều kế thừa từ class Object
- Sử dụng phương thức **toString()** để in ra thông tin của một đối tượng nào đó
- Sử dụng phương thức **equals()** để so sánh các đối tượng với nhau

Tình huống thực tế “Đi công tác”

điện thoại đặt vé
Khởi hành 20h ngày 04/07

→ Vietnam Airline

ZendVN

8h ngày 05/07/2015

Công ty
tin học X

điện thoại đặt xe
Khởi hành 19h ngày 04/07

→ Taxi Vinasun

Tình huống thực tế “Đi công tác”



Giảm thiểu sự phụ thuộc giữa các Class

- Giảm thiểu sự phụ thuộc giữa các Class tạo ra sự linh hoạt cao và tính dễ bảo trì cho ứng dụng
- Nếu giảm thiểu tốt sự phụ thuộc sẽ giúp tăng khả năng dung lại và khả năng đọc của mã nguồn, tạo cho ứng dụng tính mềm dẻo và khả năng nâng cấp bảo trì trở nên đơn giản và nhanh chóng hơn

Inner Class trong JAVA

Inner Class là một class được khai báo và tồn tại bên trong một class khác

- *Tạo nhóm quan hệ giữa các class với nhau*
- *Các class lồng nhau tăng tính dễ đọc và bảo trì cho ứng dụng*
- *Mã nguồn ngắn gọn và đơn giản hơn*

Inner Class trong JAVA

Type	Description
Member Inner Class	Class nằm trong class và ngoài các method
Anonymous Inner Class	Class ẩn danh, được tạo ra để hiện thực các interface hoặc các class mở rộng khác
Local Inner Class	Class nằm trong method
Static Nested Class	Static class được tạo ra bởi một class
Nested Interface	Interface được tạo ra bởi một class hoặc interface

Chương 07:

Sử dụng Collection trong JAVA

1. *Collections bạn là ai ?*
2. *Sử dụng List Interface*
3. *Sử dụng Set Interface*
4. *Sử dụng Map Interface*
5. *Static methods trong Collections*
6. *Sử dụng Comparable Interface*
7. *Sử dụng Comparator Interface*
8. *So sánh giữa các Collection*
9. *BookStore Version 5*

Chương 07:

Sử dụng Collection trong JAVA

1. *Collections bạn là ai ?*
2. *Sử dụng List Interface*
3. *Sử dụng Set Interface*
4. *Sử dụng Map Interface*
5. *Static methods trong Collections*
6. *Sử dụng Comparable Interface*
7. *Sử dụng Comparator Interface*
8. *So sánh giữa các Collection*
9. *BookStore Version 5*

Collections là gì ?

- *Nhược điểm của kiểu dữ liệu Array*
 - *Kích thước cố định*
 - *Các phần tử phải cùng kiểu dữ liệu và nằm liền kề nhau trong bộ nhớ*
- *Collection khắc phục tất cả các nhược điểm trên ngoài ra còn hỗ trợ sẵn các phương thức: tìm kiếm, sắp xếp, chèn, thao tác, xóa phần tử, ..*
- *Collection là một công cụ có thể chứa và thao tác với các đối tượng thuộc nhiều kiểu dữ liệu khác nhau*

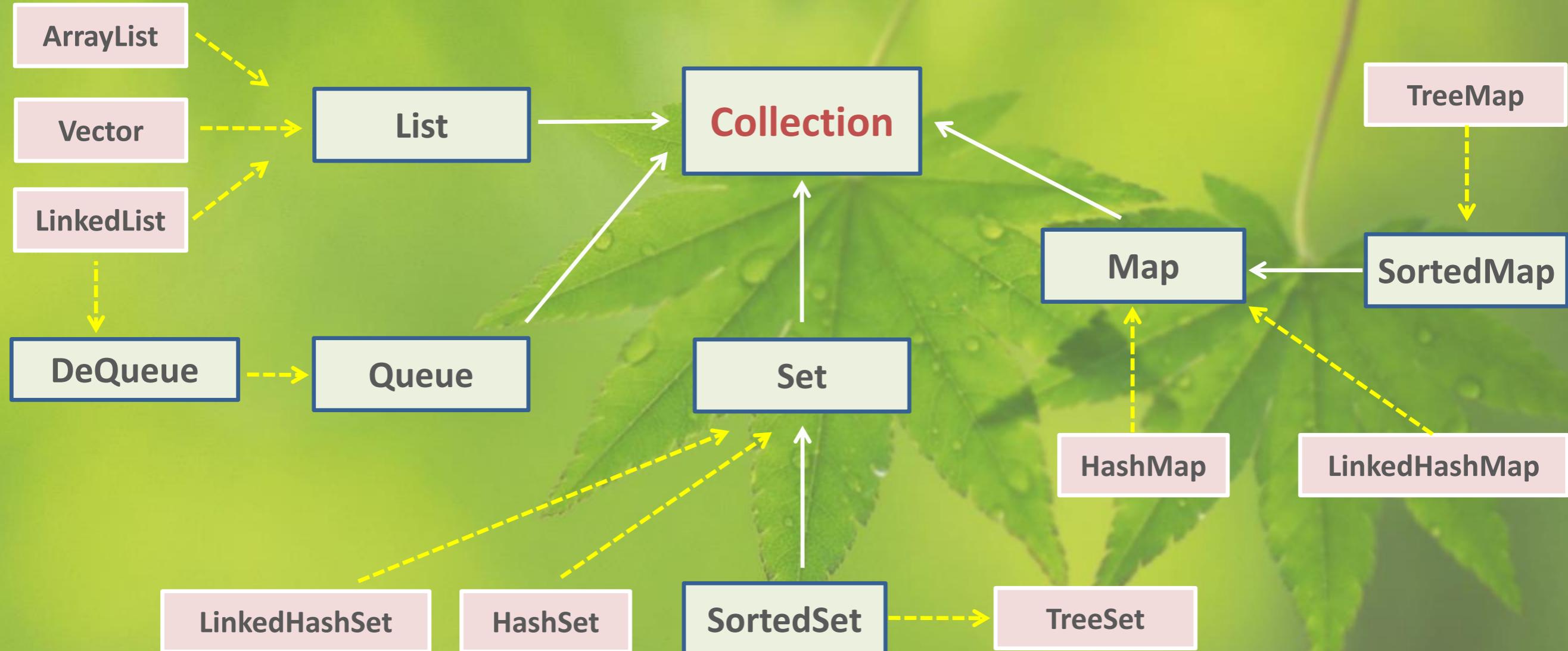
Cấu trúc Collection

Interface

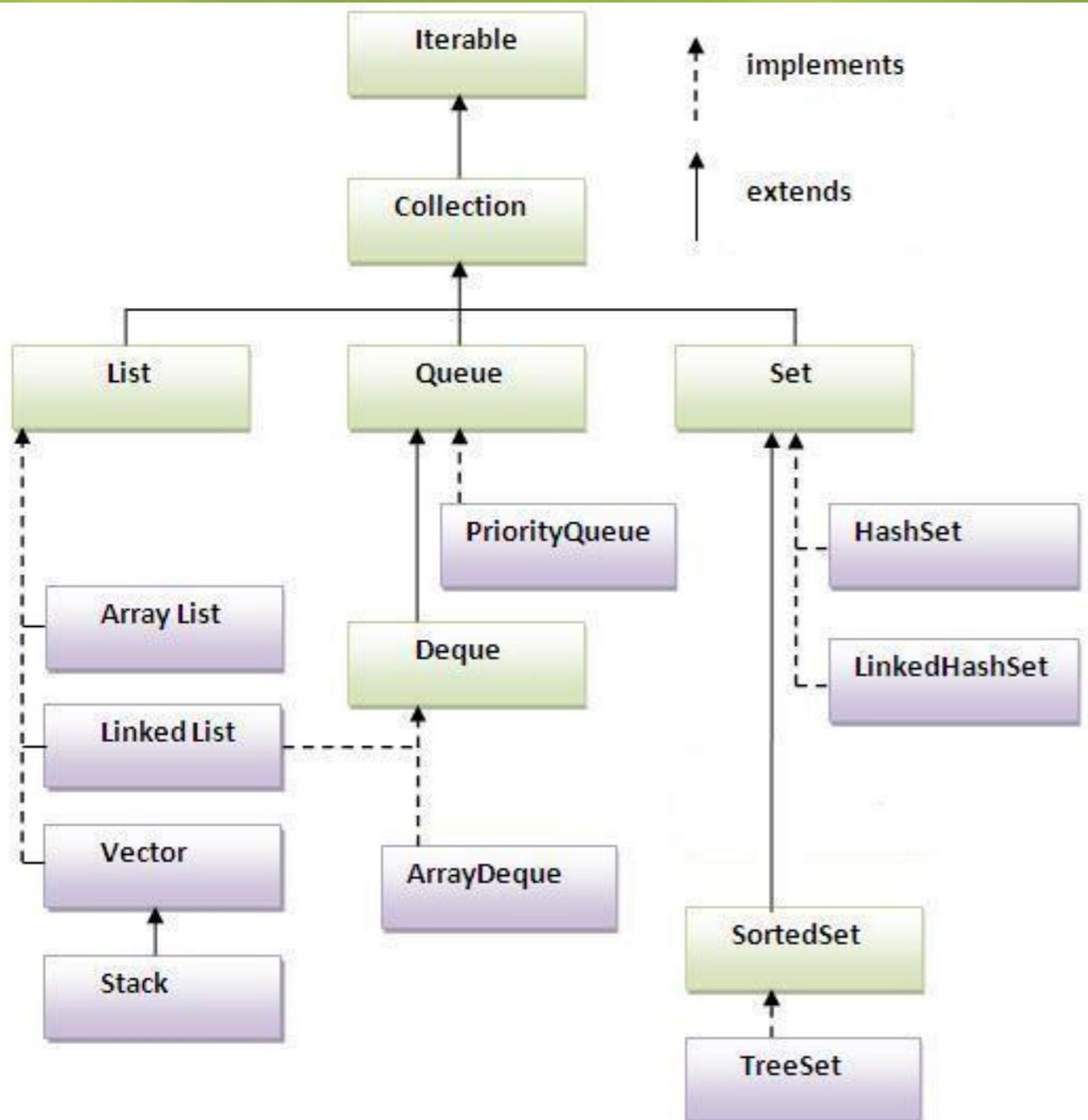
Class

extends

implements



Cấu trúc Collection



- Tất cả các class và interface của *Collection* đều nằm trong package *java.util*
- Các phương thức thường sử dụng trong *Collection*
 - *size()*
 - *add()*
 - *isEmpty()*
 - *remove()*
 - *set()*
 - *get()*

Chương 07:

Sử dụng Collection trong JAVA

1. *Collections bạn là ai ?*
2. ***Sử dụng List Interface***
3. *Sử dụng Set Interface*
4. *Sử dụng Map Interface*
5. *Static methods trong Collections*
6. *Sử dụng Comparable Interface*
7. *Sử dụng Comparator Interface*
8. *So sánh giữa các Collection*
9. *BookStore Version 5*

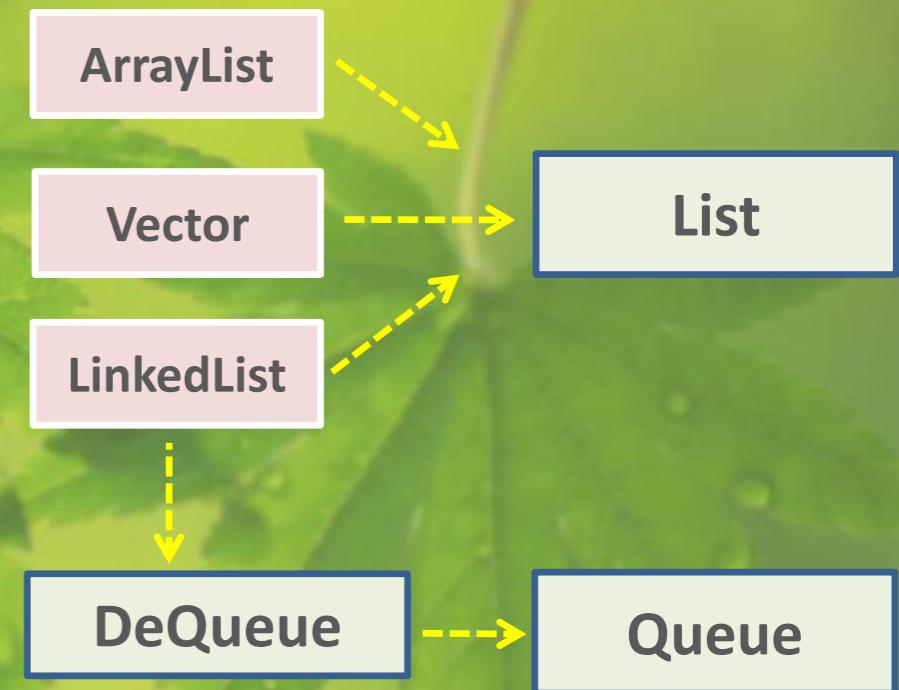
ArrayList

- Sử dụng một mảng động để lưu trữ các phần tử
- Giá trị các phần tử có thể trùng nhau
- Thứ tự thêm vào các element cũng chính là thứ tự các phần tử trong ArrayList
- Tìm hiểu các phương thức thông dụng và phương pháp duyệt các phần tử *for* và *Iterator*



LinkedList

- Sử dụng “**doubly linked list**” để lưu trữ các phần tử
- Giá trị các phần tử có thể trùng nhau
- Thứ tự thêm vào các element cũng chính là thứ tự các phần tử trong *LinkedList*
- Tìm hiểu phương pháp duyệt các phần tử: *for* và *Iterator*



ArrayList vs LinkedList

ArrayList

- Sử dụng mảng động để lưu trữ các phần tử
- Implement: List
- Better for storing and accessing data

LinkedList

- Sử dụng “doubly linked list” để lưu trữ các phần tử
- Implement: List và Dequeue
- Better for manipulating

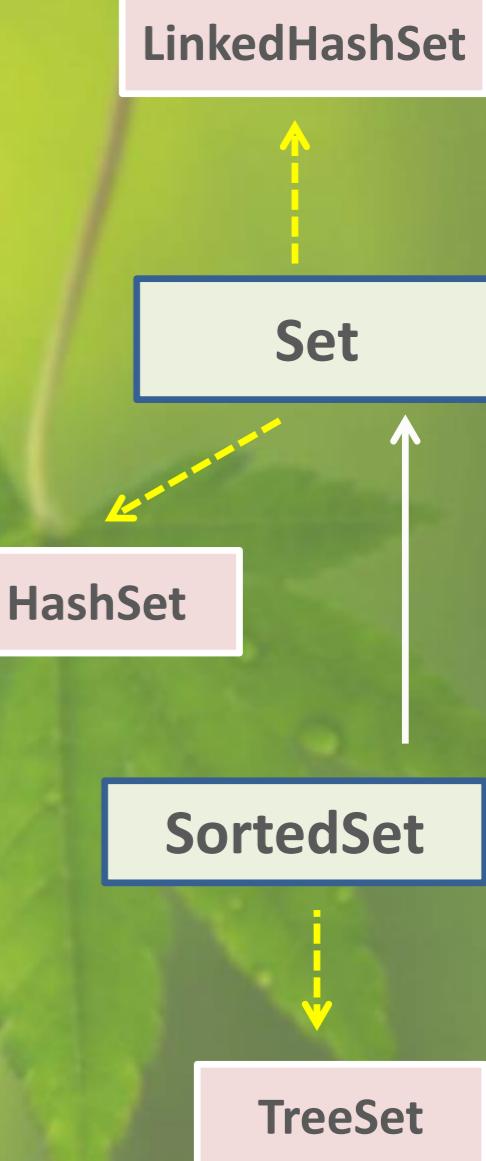
Chương 07:

Sử dụng Collection trong JAVA

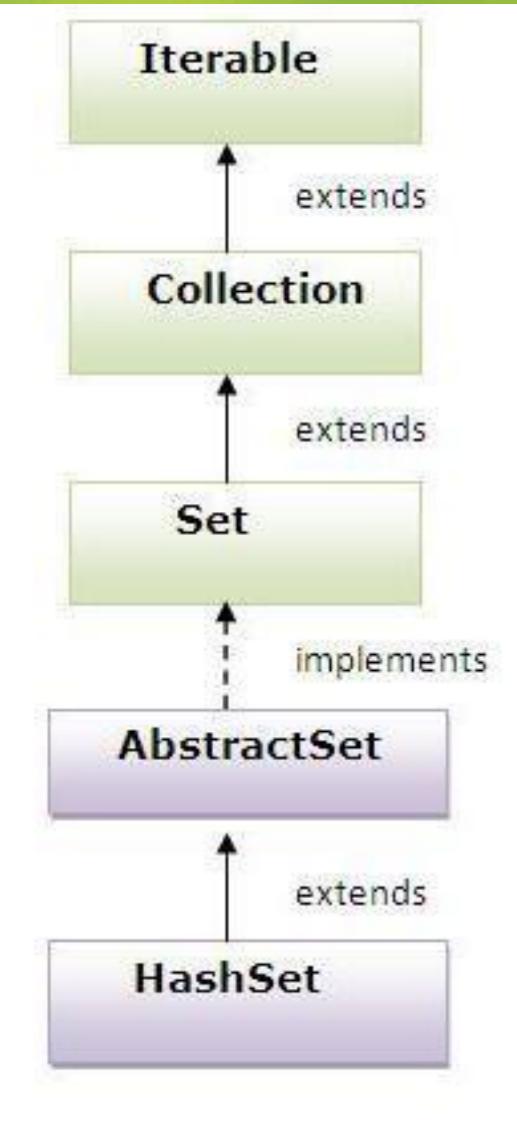
1. *Collections bạn là ai ?*
2. *Sử dụng List Interface*
3. ***Sử dụng Set Interface***
4. *Sử dụng Map Interface*
5. *Static methods trong Collections*
6. *Sử dụng Comparable Interface*
7. *Sử dụng Comparator Interface*
8. *So sánh giữa các Collection*
9. *BookStore Version 5*

Set Interface

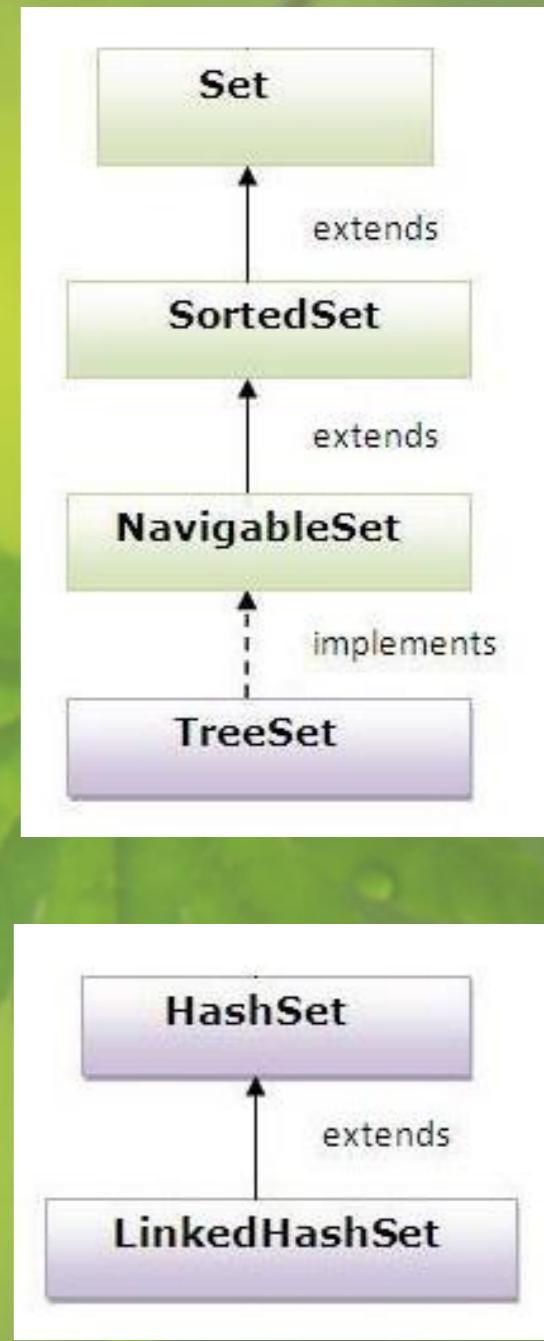
- Lưu trữ các phần tử thuộc nhiều kiểu dữ liệu khác nhau **nhưng** các phần tử **không được lưu trữ theo chỉ số**
- Không chứa các phần tử trùng nhau
- **Sự khác biệt giữa List và Set:** List có thể chứa những phần tử có giá trị giống nhau còn Set thì không
- Các class tiến hành tìm hiểu: *HashSet, LinkedHashSet, TreeSet*



HashSet – LinkedHashSet – TreeSet



- *HashSet* *thứ tự các phần tử được thêm vào bị thay đổi*
- *LinkedSet* *thứ tự các phần tử được thêm vào đảm bảo được giữ nguyên*
- *TreeSet* *thứ tự các phần tử được thêm vào được sắp xếp theo thứ tự tăng dần*



Chương 07:

Sử dụng Collection trong JAVA

1. *Collections bạn là ai ?*
2. *Sử dụng List Interface*
3. *Sử dụng Set Interface*
- 4. *Sử dụng Map Interface***
5. *So sánh giữa các Collection*
6. *Static methods trong Collections*
7. *Sử dụng Comparable Interface*
8. *Sử dụng Comparator Interface*
9. *BookStore Version 5*

Map Interface

- Lưu trữ các phần tử thuộc nhiều kiểu dữ liệu khác nhau **nhưng** các phần tử là **một đối tượng bao gồm key và value**
- Không chứa các phần tử có khóa (key) giống nhau
- Các class tiến hành tìm hiểu: *HashMap*, *LinkedHashMap*, *TreeMap*

LinkedHashMap

Map

HashMap

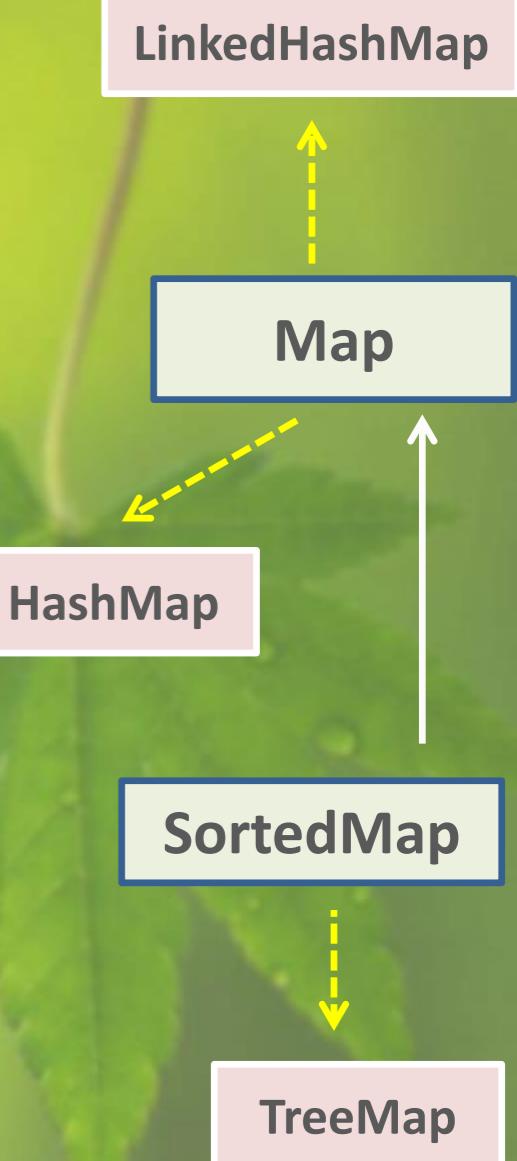
SortedMap

TreeMap



HashMap – LinkedHashMap - TreeMap

- *HashMap* thứ tự các phần tử thay đổi so với thứ tự insert vào
- *LinkedHashMap* giữ nguyên thứ tự các phần tử
- *Treemap* thứ tự các phần tử được sắp xếp theo thứ tự tang dần của key



Chương 07:

Sử dụng Collection trong JAVA

1. *Collections bạn là ai ?*
2. *Sử dụng List Interface*
3. *Sử dụng Set Interface*
4. *Sử dụng Map Interface*
5. *So sánh giữa các Collection*
6. *Static methods trong Collections*
7. *Sử dụng Comparable Interface*
8. *Sử dụng Comparator Interface*
9. *BookStore Version 5*

So sánh giữa các Collection

- **List** có thể tồn tại các phần tử có giá trị trùng nhau
- **Set** không tồn tại các phần tử có giá trị trùng nhau
- **Map** không tồn tại các phần tử có key trùng nhau

Tình huống luyện tập

- *ArrayList – LinkedList*
 - *HashSet – LinkHashSet – TreeSet*
 - *HashMap – LinkHashMap – TreeMap*
- Difference between ... and ... in Java
 - Java ... vs ...

Chương 07:

Sử dụng Collection trong JAVA

- 1. *Collections bạn là ai ?*
- 2. *Sử dụng List Interface*
- 3. *Sử dụng Set Interface*
- 4. *Sử dụng Map Interface*
- 5. *So sánh giữa các Collection*
- 6. *Static methods trong Collections*
- 7. *Sử dụng Comparable Interface*
- 8. *Sử dụng Comparator Interface*
- 9. *BookStore Version 5*

Sử dụng Static methods trong Collections

Sử dụng phương thức static của Collections để thực hiện các yêu cầu sau:

- *Sắp xếp danh sách theo thứ tự tăng dần*
- *Đảo ngược vị trí các phần tử trong một danh sách*
- *Sắp xếp danh sách theo thứ tự giảm dần*
- *Tìm kiếm phần tử trong danh sách*
- *Trộn ngẫu nhiên vị trí các phần tử nằm trong danh sách*

Chương 07:

Sử dụng Collection trong JAVA

1. *Collections bạn là ai ?*
2. *Sử dụng List Interface*
3. *Sử dụng Set Interface*
4. *Sử dụng Map Interface*
5. *So sánh giữa các Collection*
6. *Static methods trong Collections*
7. ***Sử dụng Comparable Interface***
8. *Sử dụng Comparator Interface*
9. *BookStore Version 5*

Sử dụng Collections.sort đối với object

- *Exception in thread "main" java.lang.ClassCastException: chapo6.collection.Course cannot be cast to java.lang.Comparable*
- Khắc phục bằng cách **implements Comparable** và định nghĩa phương thức **compareTo()**

```
public class Course implements Comparable{
    @Override
    public int compareTo(Object obj) {
        // your code here
    }
}
```

Chương 07:

Sử dụng Collection trong JAVA

1. *Collections bạn là ai ?*
2. *Sử dụng List Interface*
3. *Sử dụng Set Interface*
4. *Sử dụng Map Interface*
5. *So sánh giữa các Collection*
6. *Static methods trong Collections*
7. *Sử dụng Comparable Interface*
- 8. *Sử dụng Comparator Interface***
9. *BookStore Version 5*

Sử dụng Comparator Interface

- Định nghĩa class XComparator implements từ Comparator
- Sử dụng tham số thứ hai trong Collections.sort()

Chương 08:

Sử dụng Generics trong JAVA

- 
1. *Generic Method*
 2. *Generic Class*
 3. *Generic Collection*
 4. *Ràng buộc dữ liệu trong Generic*
 5. *Xây dựng ứng dụng BookStore V7*

Sử dụng Generic như thế nào

- *Generic giúp phát hiện sớm các exception trong lúc biên dịch chương trình*
- *Generic Methods*
- *Generic Class*
- *Generic Collection*

Chương 09:

Class & Thư viện mở rộng

Sử dụng class Pattern và Match trong Java Regex

Sử dụng class Pattern và Matcher trong Java Regex

```
Pattern pattern = Pattern.compile("\\s+");  
Matcher matcher = pattern.matcher("Java is easy");
```

