Học lập trình Java cơ bản

LOTUS ACADEMY

Bài 4:
Mảng và danh sách trong Java

Hanoi 2022

NỘI DUNG CHÍNH

Mảng một/nhiều chiều

Lớp Arrays trong Java

Collection & Collections trong Java

Phần 1

Mảng một / nhiều chiều

Mảng một / nhiều chiều

Khai mảng
Và
Truy Cập:

Mảng 1 chiều:

Kiểu[] TênMảng = new **Kiểu**[số_phần_tử];

Kiểu[] TênMảng = new **Kiểu[]** {gt1,gt2,gt3…,gtn};

Mảng n chiều:

Kiểu[][]…[] TênMảng = new **Kiểu**[số_phần_tử][số_phần_tử]… [số_phần_tử];

Kiểu [][]…[] TênMảng = new **Kiểu [][]…[]**{ { … {gt_i1 ,.. gt_in }, …}, …};

Thuộc tính/phương thức	Chức năng
Tên_mảng[vị trí x]	Truy cập vào giá trị ở một vị trí x trong mảng một chiều
Tên_mảng[vị trí x, vị trí y,..]	Truy cập vào giá trị ở một vị trí x, y,.. trong mảng nhiều chiều
length	Trả lại độ dài của mảng đối với mảng một chiều

Phần 2

Lớp Arrays trong Java

Lớp Arrays trong Java

Phương thức	Chức năng
sort()	Sắp xếp các phần tử của mảng theo thứ tự tăng dần theo thuật toán Quick sort
toString()	Trả về chuỗi của tất cả các phần tử của một mảng. Chuỗi này bao gồm tất cả các phần tử được bao quanh trong “[]”.
binarySearch()	Tìm vị trí của phần tử trong mảng bằng phương thức tìm kiếm nhị phân (binary search). Các phần tử trong mảng phải được sắp xếp trước khi gọi phương thức này.
fill()	Gán giá trị xác định cho mỗi phần tử của một mảng. Phương thức này hữu ích trong việc khởi tạo tất cả các phần tử của một mảng với một giá trị.
copyOf()	Sao chép mảng được chỉ định vào mảng mới của cùng một kiểu.
asList()	Tạo một danh sách từ một mảng được chỉ định.
equals()	So sánh hai mảng có bằng nhau hay không. Phương pháp này lấy hai mảng làm tham số và trả về true nếu cả hai mảng có cùng một số phần tử và các cặp tương ứng của các phần tử của cả hai mảng đều bằng nhau.

Lớp Arrays trong Java

Phương thức sort() trong Arrays

Cú pháp:

- Arrays.sort(<Tên_mảng>)
- Arrays.sort(<Tên_mảng>, start_index, stop_index)
- Arrays.sort(<Tên_mảng>, Comparator)
- Arrays.sort(<Tên_mảng>, start_index, stop_index, Comparator)

VD: Cho 1 mảng số nguyên gồm 10 phần tử bất kì.

1, Hãy sắp xếp mảng đó theo thứ tự tăng dần.

```
int[] arr = { 2, 1, 4, 6, 3, 8, 9, 10, 0, 5};
Arrays.sort(arr);
```

2, Hãy sắp xếp các phần tử từ vị trí thứ 2 đến vị trí thứ 7

```
int[] arr = { 2, 1, 4, 6, 10, 9, 8, 3, 0, 5};
Arrays.sort(arr, 2, 8);
```

Lớp Arrays trong Java

Phương thức toString() trong Arrays

Cú pháp: Arrays.toString(<Tên_mảng>)

VD: Cho 4 mảng : số nguyên, số thực, chuỗi, và kí tự . Mỗi mảng gồm 5 phần tử bất kì.

Hãy sử dụng phương thức toString để chuyển các mảng trên thành 1 chuỗi

```
int[] arr_Int = { 2, 1, 4, 6, 10};
double[] arr_Double = {2.333, 1.333, 5, 3.5, 6.7};
String[] arr_Str = {"Apple", "PineApple", "Orange","Lemon", "Grapes"};
char[] arr_Char = {'H','E', 'L', 'L','O'};
System.out.println(Arrays.toString(arr_Int));
System.out.println(Arrays.toString(arr_Double));
System.out.println(Arrays.toString(arr_Str));
System.out.println(Arrays.toString(arr_Char));
```

Kết quả:

```
[2, 1, 4, 6, 10]
[2.333, 1.333, 5.0, 3.5, 6.7]
[Apple, PineApple, Orange, Lemon, Grapes]
[H, E, L, L, O]
```

Lớp Arrays trong Java

Phương thức fill() trong Arrays

Cú pháp:
- Arrays.fill(<Tên_mảng>, giá_trị)
- Arrays.fill(<Tên_mảng>, start_index, stop_index, giá_trị)

VD: Cấp phát bộ nhớ cho 1 mảng với 10 phần tử.

Hãy sử dụng phương thức fill để:

1, Lấp đầy toàn bộ mảng với giá trị 1

```
//khoi tao 1 mang so nguyen gom 10 phan tu
int[] arr_Int = new int[10];
//Lap day mang voi gia tri 1
Arrays.fill(arr_Int, 1);
```

2, Lấp đầy từ phần tử ở vị trí số 2 đến số 8 với giá trị 10

```
//khoi tao 1 mang so nguyen gom 10 phan tu
int[] arr_Int = new int[10];
//Lap day mang tu phan tu thu 2 den 8 voi gia tri 10
Arrays.fill(arr_Int, 2,8,10);
```

Phần 3

Kiểu danh sách

Kiểu danh sách

Khai báo
Và
Truy Cập:

Cần khai báo không gian tên : using System.Collection.Generic;
List<Kiểu> TênDS = new **ArrayList<Kiểu>()**;
List<Kiểu> TênDS = new **ArrayList <Kiểu>()** {**Arrays.asList**(gt1,gt2,gt3…,gtn)};
Truy cập : **TênDS.get(vị trí)**
vị_trí có giá trị từ 0 đến nhỏ hơn số_phần_tử

Thuộc tính/phương thức	Chức năng
size()	Trả lại số lượng phần tử hiện có trong danh sách
add(giá trị)	Thêm vào cuối một giá trị cùng kiểu vào trong danh sách
addAll(List<T> ds)	Thêm vào cuối danh sách hiện tại một danh sách khác cùng kiểu
remove(vị trí)	Bỏ khỏi danh sách một phần tử ở vị trí cụ thể
removeAll(List<T> ds)	Xoá bỏ 1 danh sách các phần tử khỏi danh sách cũ
clear()	Xóa trống danh sách

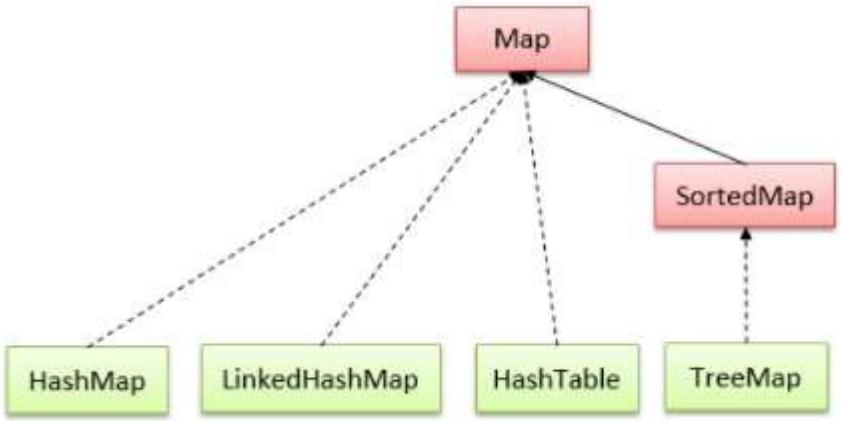
Phần 3

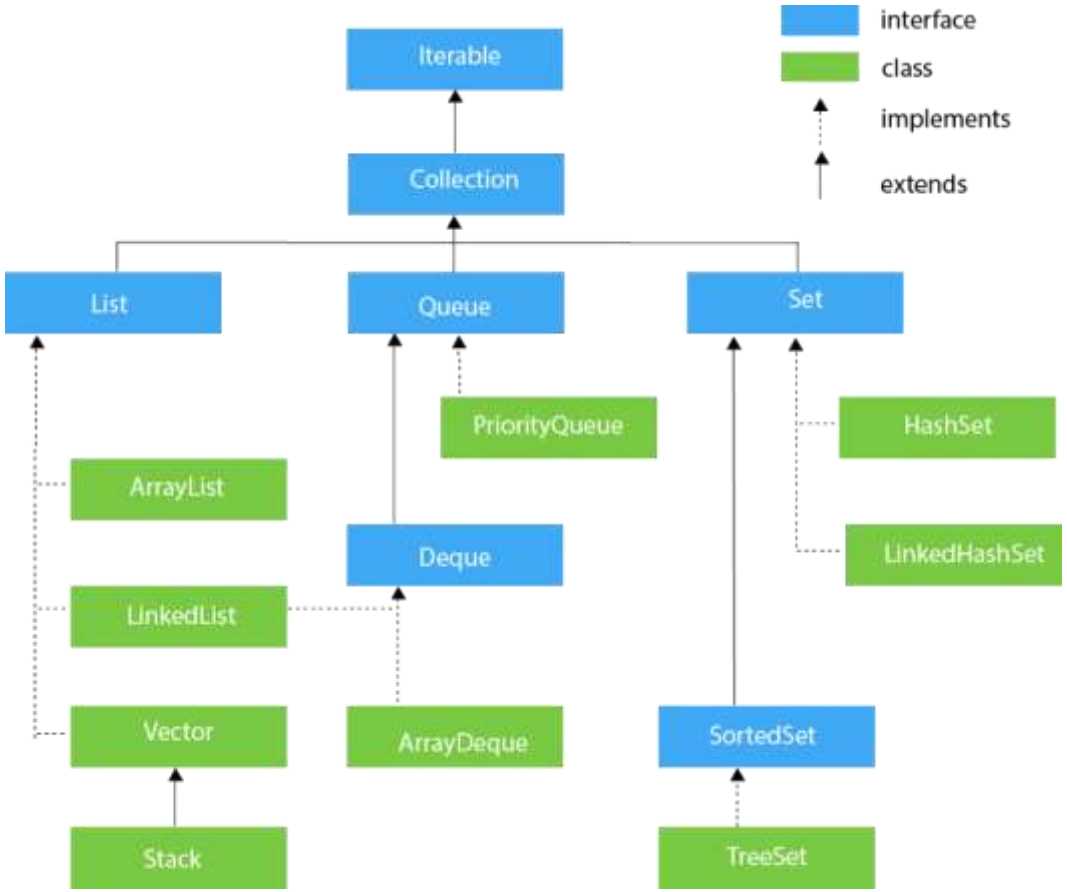
Collection & Collections trong Java

Collection & Collections trong Java

Collections trong java là một khuôn khổ cung cấp một kiến trúc để lưu trữ và thao tác tới nhóm các đối tượng. Tất cả các hoạt động mà bạn thực hiện trên một dữ liệu như tìm kiếm, phân loại, chèn, xóa,... có thể được thực hiện bởi Java Collections.

Collection trong java là một root interface trong hệ thống cấp bậc Collection. Java Collection cung cấp nhiều interface (Set, List, Queue, Deque, …) và các lớp (ArrayList, Vector, LinkedList, PriorityQueue, HashSet, LinkedHashSet, TreeSet, …).





Collection & Collections trong Java

Interface	Mô tả
Set	- là một collection không thể chứa 2 giá trị trùng lặp.
List	- là một collection có thứ. List có thể chứa các phần tử trùng lặp. Thường có quyền kiểm soát chính xác vị trí các phần tử được chèn vào và có thể truy cập chúng bằng chỉ số (vị trí của chúng).
Queue	- là một collection được sử dụng để chứa nhiều phần tử trước khi xử lý. Bên cạnh các thao tác cơ bản của collection, Queue cung cấp các thao tác bổ sung như chèn, lấy ra và kiểm tra. - Queue có thể được sử dụng như là FIFO (first-in, first-out - vào trước, ra trước)
Deque	- là một collection được sử dụng để chứa nhiều phần tử trước khi xử lý. - Ngoài các thao tác cơ bản của collection, một Deque cung cấp các thao tác bổ sung như chèn, lấy ra và kiểm tra. - Deques có thể được sử dụng như là FIFO (first-in, first-out - vào trước, ra trước) và LIFO (last-in, first-out - vào sau, ra trước). Trong một Deque, tất cả các phần tử mới có thể được chèn vào, lấy ra và lấy ra ở cả hai đầu.
Map	- là một đối tượng ánh xạ mỗi key tương úng với một giá trị. Map không thể chứa giá trị trùng lặp. Mỗi key có thể ánh xạ đến nhiều nhất một giá trị.
SortedSet	- là một Set chứa các phần tử theo thứ tự tăng dần.
SortedMap	- là một Map chứa các phần tử được sắp xếp theo thứ tự tăng dần của key của chúng.

LTS Lotus Technology Services

LQA Lotus Quality Assurance

Xin trân trọng cảm ơn

Lotus Academy

lotusacademy.edu.vn