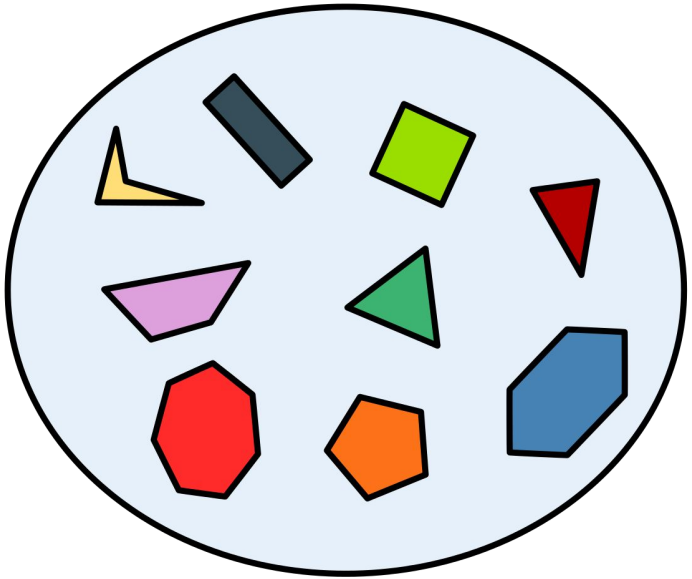
Collection-2

Nội dung

1. Set Interface
2. SortedSet Interface
3. Một số loại Set cơ bản
4. Map Interface
5. Một số loại Map cơ bản

Set Interface

- **Set interface** là lớp kế thừa lại từ **Collection Interface**
- Lưu trữ các dữ liệu không có thứ tự
- Dữ liệu lưu trữ không thể trùng lặp.



Set Interface

Một số phương thức cơ bản của Set

- `containsAll(Collection c)`
- `addAll(Collection c)`
- `retainAll(Collection c)`
- `removeAll(Collection c)`

SortedSet Interface

- **SortedSet interface** là lớp kế thừa lại từ **Set Interface**
- Giống với **Set**
- Hỗ trợ tính năng sắp xếp phần tử theo thứ tự tăng dần.

SortedSet Interface

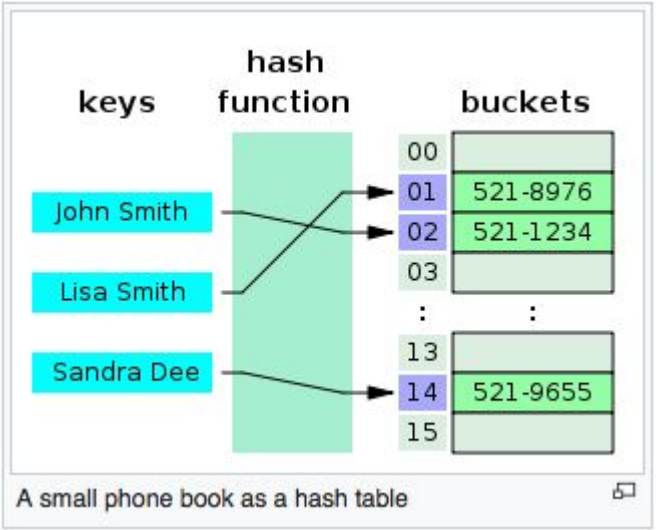
Một số phương thức cơ bản của SortedSet

- `first()`
- `last()`
- `headSet(E end)`
- `subSet(E start, E end)`
- `tailSet(E from)`

Một số loại Set cơ bản

HashSet

- **HashSet** là lớp thực thi của **interface Set**
- Sử dụng 1 **HashTable** (bảng băm) để lưu trữ dữ liệu
- Không thể truy cập trực tiếp qua qua index



HashSet

Contrucstor

- `HashSet()`
- `HashSet(Collection c)`
- `HashSet(int size)`

HashSet

Một số phương thức cơ bản của HashSet

- `add(E obj)`
- `clone()`
- `clear()`
- `contains(Object obj)`
- `size()`
- `iterator()`

HashSet

Ví dụ

```
Set<String> mySet = new HashSet<>();
mySet.add("Hello");
mySet.add("World");
mySet.add("Java");
Iterator<String> iterator = mySet.iterator() ;
while (iterator.hasNext()) {
   String str = iterator.next() ;
   System.out.println(str);
}
```

LinkedHashSet

- **LinkedHashSet** là lớp thực thi của **interface Set**
- Sự kết hợp của LinkedList và HashSet

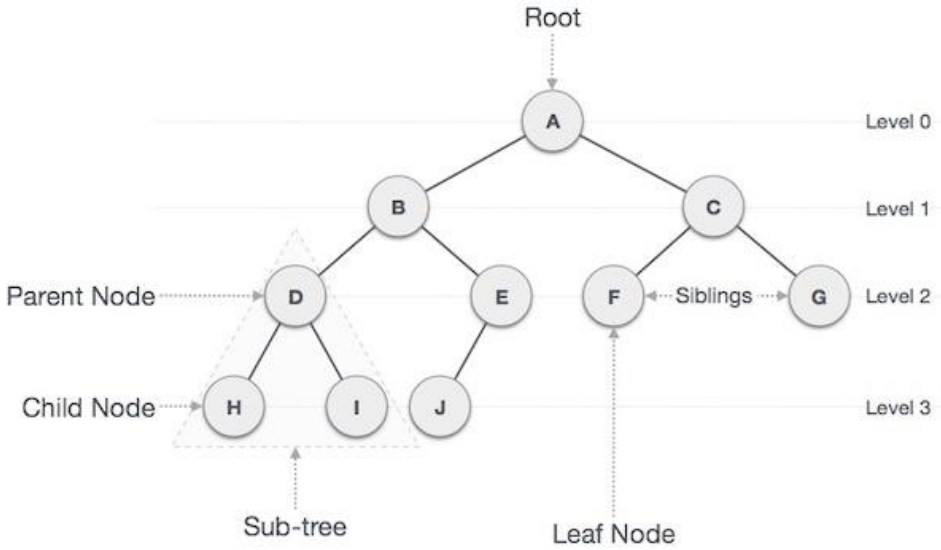
LinkedHashSet

Contrucstor

- `LinkedHashSet()`
- `LinkedHashSet(Collection c)`
- `LinkedHashSet(int size)`

TreeSet

- **TreeSet** là lớp thực thi của **interface Set**
- Sử dụng cấu trúc Tree (cây) để lưu trữ dữ liệu



TreeSet

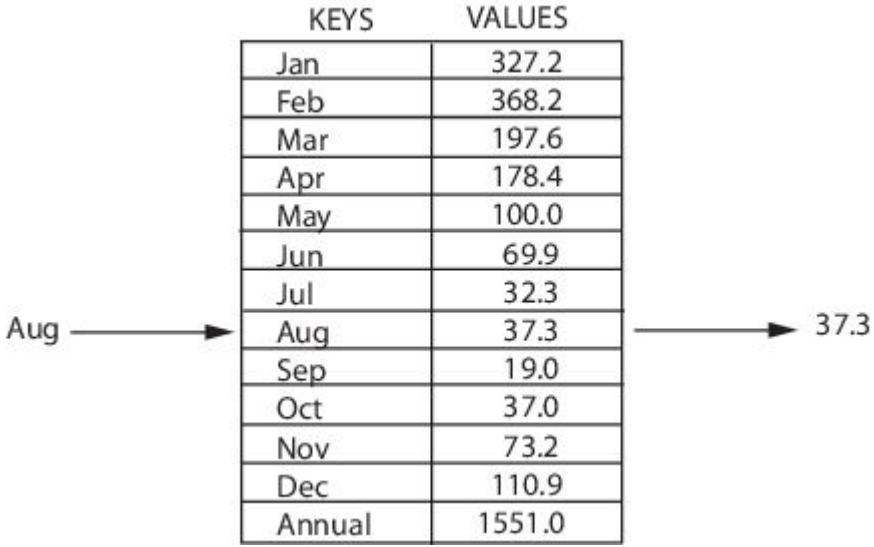
Contrucstor

- `TreeSet()`
- `TreeSet(Collection c)`
- `TreeSet(int size)`

Map

Map Interface

- **Map** là một collection lưu trữ dữ liệu bằng cấu trúc key-value.
- Mỗi 1 key sẽ có 1 value tương ứng
- Key là duy nhất nhưng Value thì có thể trùng



Map Interface

Một số phương thức cơ bản của Map

- `put(K key, V value)`
- `get(K key)`
- `size()`
- `values()`
- `containsKey(Object k)`
- `containsValue(Object v)`

HashMap

- **HashMap** là lớp thực thi của **interface Map**
- Kích thước có thể tự tăng

Contrucstor

- `HashMap()`
- `HashMap(int initialCapacity)`

HashTable

- **HashTable** là lớp thực thi của **interface Map**
- cặp giá trị key/value được lưu trong 1 hashtable

Contrucstor

- `HashTable()`
- `HashTable(int initialCapacity)`

TreeMap

- **TreeMap** là lớp thực thi của **interface Map**
- Giá trị được lưu trong 1 Trê
- Key được sắp xếp theo thứ tự

Contrucstor

- `TreeMap()`
- `TreeMap(Comparator c)`

TreeMap

Một số phương thức cơ bản của TreeMap

- `firstKey()`
- `lastKey()`
- `headMap(K toKey)`
- `tailMap(K fromKey)`

TreeMap

Ví dụ

```
Map<String, String> myMap = new HashMap<>();
myMap.put("hello", "xin chao");
myMap.put("goodbye", "tam biet");
// Lấy giá trị theo key
System.out.println(myMap.get("hello"));
System.out.println(myMap.get("goodbye"));
// Duyệt toàn bộ map
Iterator<String> keys = myMap.keySet().iterator();
while (keys.hasNext()) {
   System.out.println(myMap.get(keys.next()));
}
```