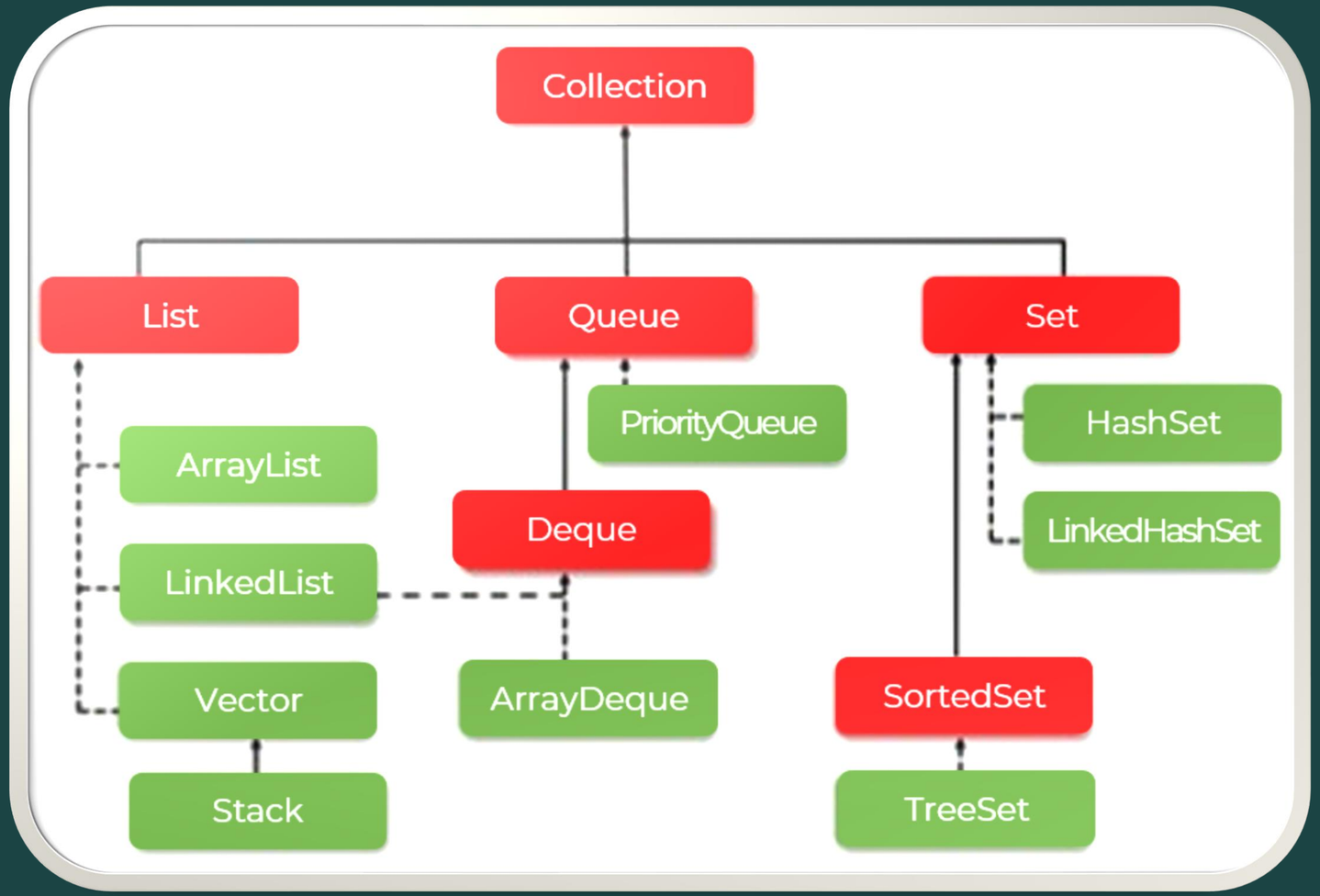
NỘI DUNG CHÍNH

1 Set

2 Map

Set

- **Không** chứa các phần tử **trùng lặp**
- Các **class** triển khai của Set
  - **HashSet**
  - **LinkedHashSet**
  - **TreeSet**
- **Hash Code??**



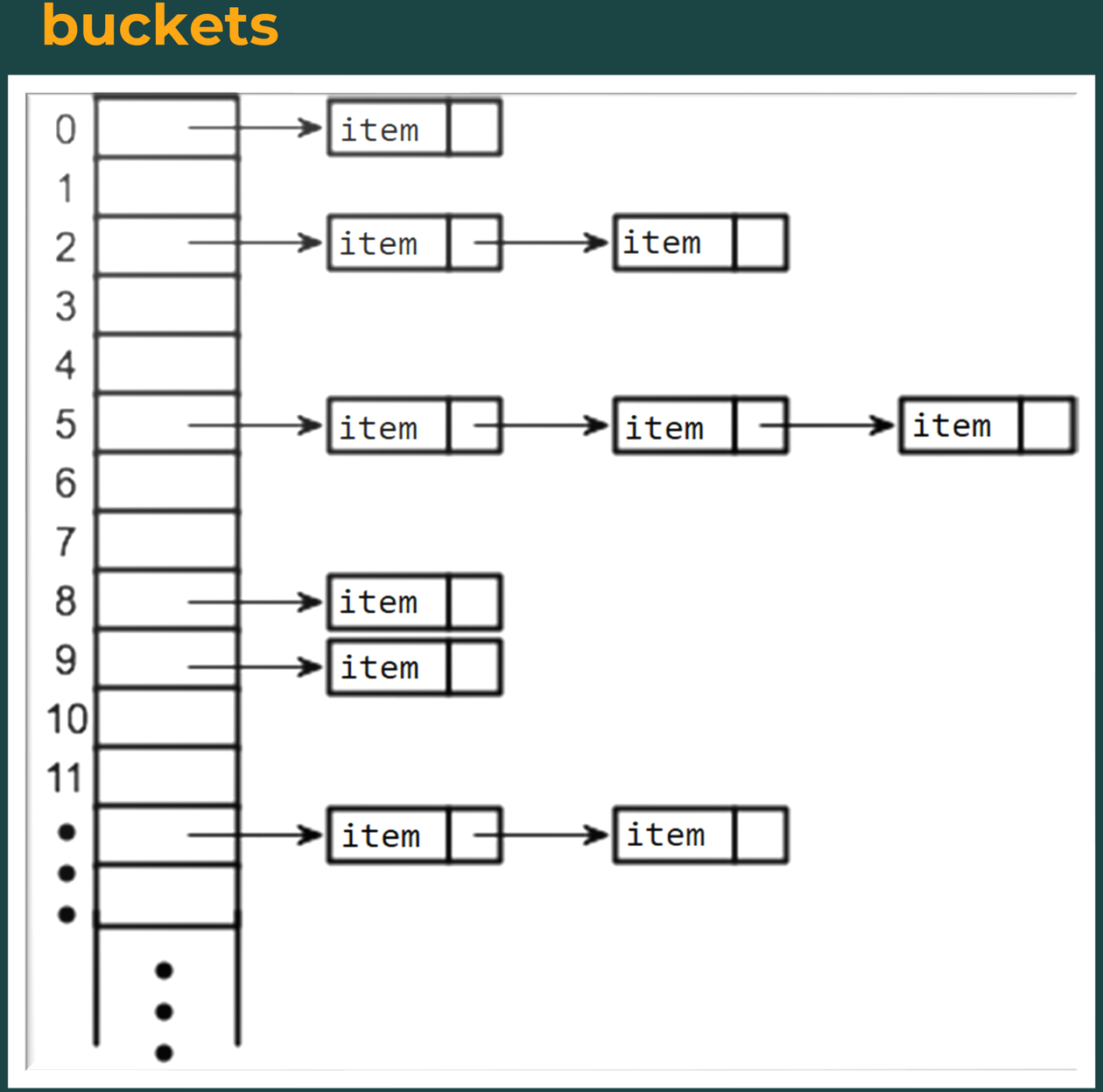
Hash Code (Mã băm)

- Là một giá trị **số nguyên** được tính toán từ **dữ liệu đầu vào**.
- **Đặc điểm**
  - Với một **đầu vào cố định**, **mã băm** phải luôn **giống nhau**.
    - new Student(1, “Huỳnh Hải Thiên", 9.6); => 1556502625
    - new Student(1, "Huỳnh Hải Thiên", 9.6); => 1556502625
  - **2** đối tượng **khác nhau** vẫn có thể xảy ra trường hợp **cùng mã băm**
    - new Student(7022, "Nguyễn Văn A", 9.97); => -1526473644
    - new Student(6357, "Nguyễn Văn B", 9.75); => -1526473644
- **Mục đích**
  - Hỗ trợ tìm kiếm **nhanh**.

HashSet

- **Đặc điểm**
  - Các phần tử trong HashSet **không có thứ tự** cụ thể
  - **Thêm**, **xóa** và **lấy** phần tử **nhanh**

- Mỗi **bucket** là **singly linked list** để lưu trữ các phần tử dựa trên **mã băm**.
- **initialCapacity** = **16** (Sức chứa ban đầu)
- **loadFactor** = **0.75** (Hệ số đầy)

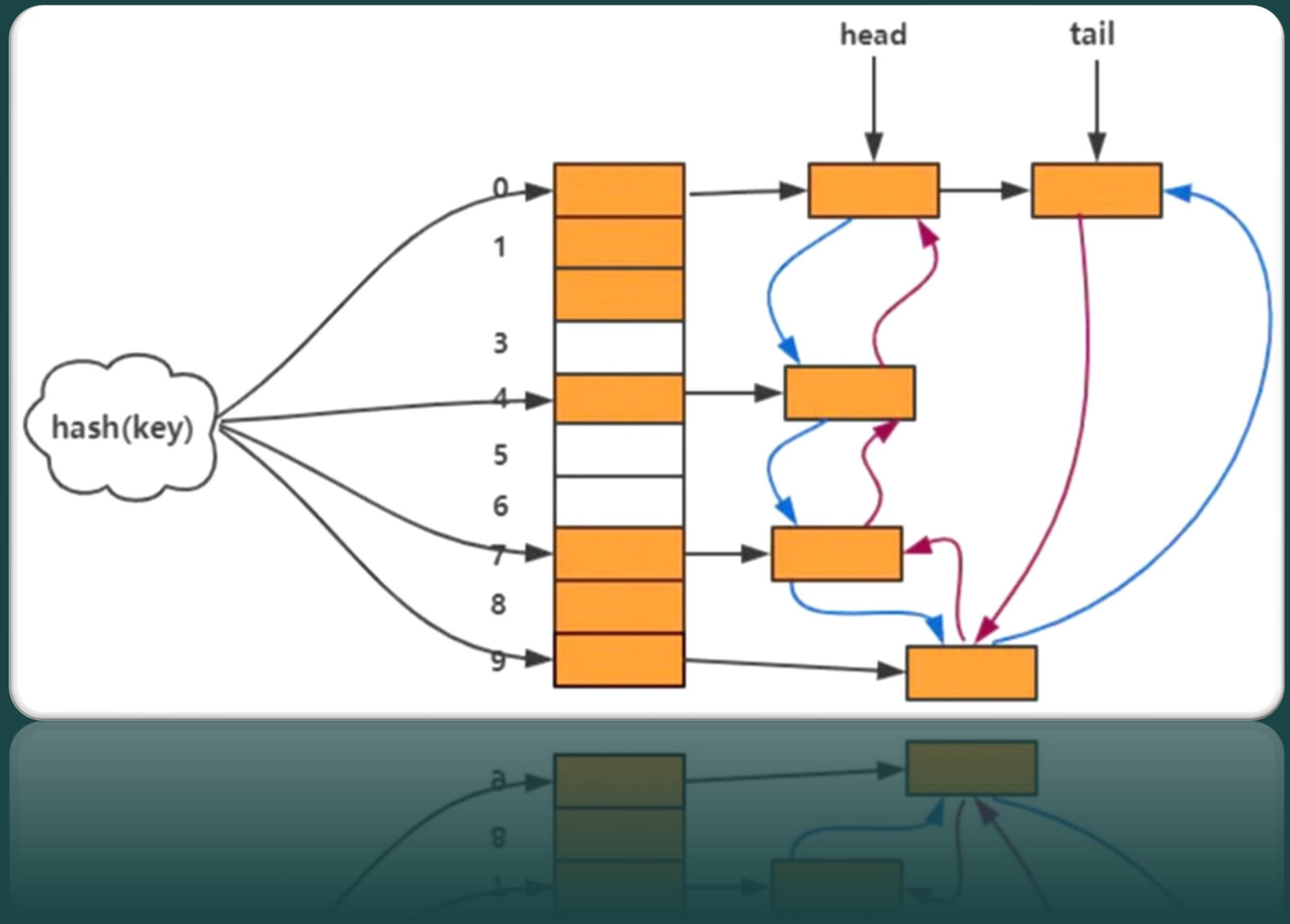


Lưu ý

- Đối với kiểu dữ liệu do chúng ta **tự định nghĩa ra cần override**
  - **equals()**
  - **hashCode()**

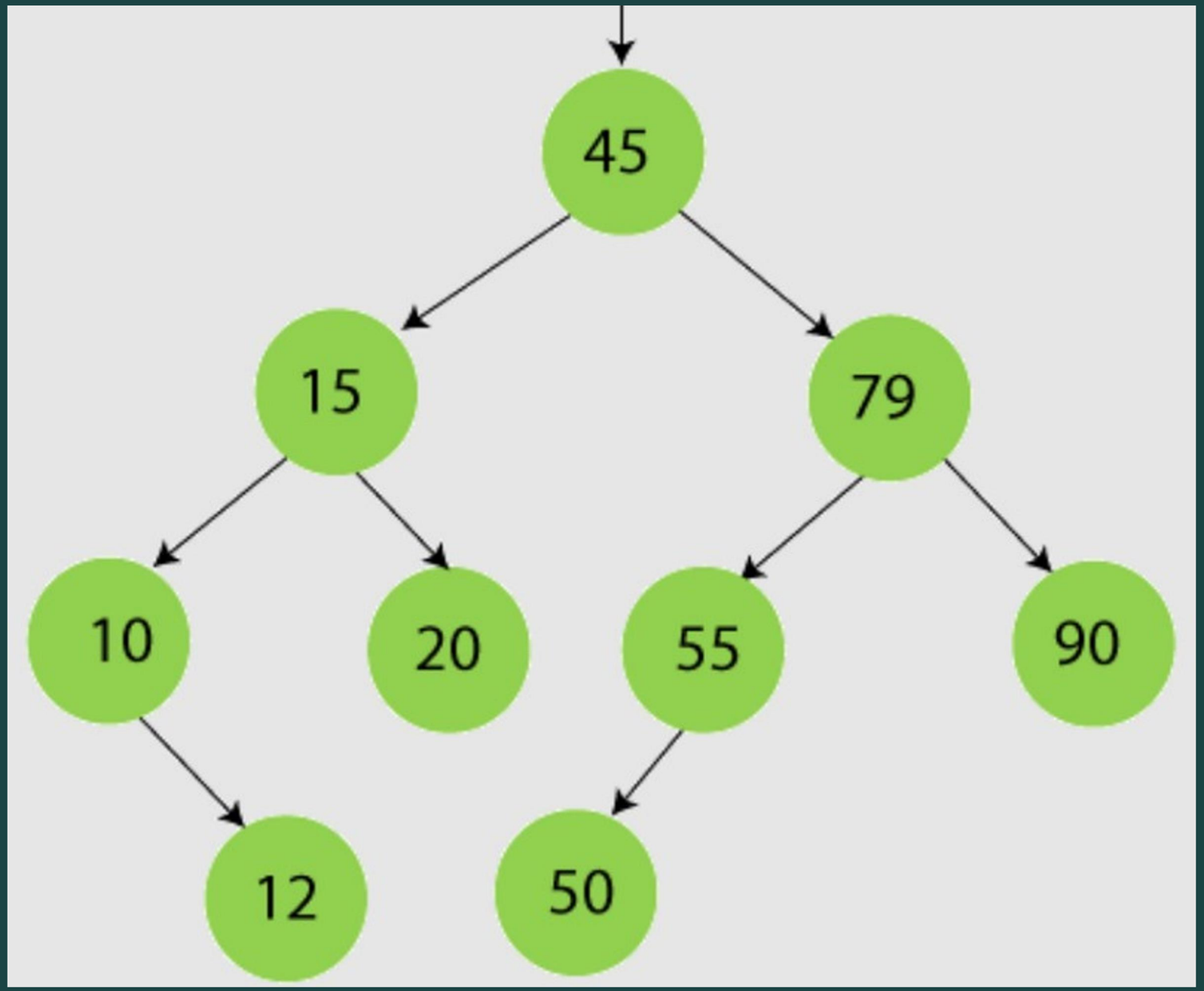
LinkedHashSet

- Là một **biến thể** của **HashSet**.
- **Điểm khác**: duy trì **thứ tự** thêm vào.



TreeSet

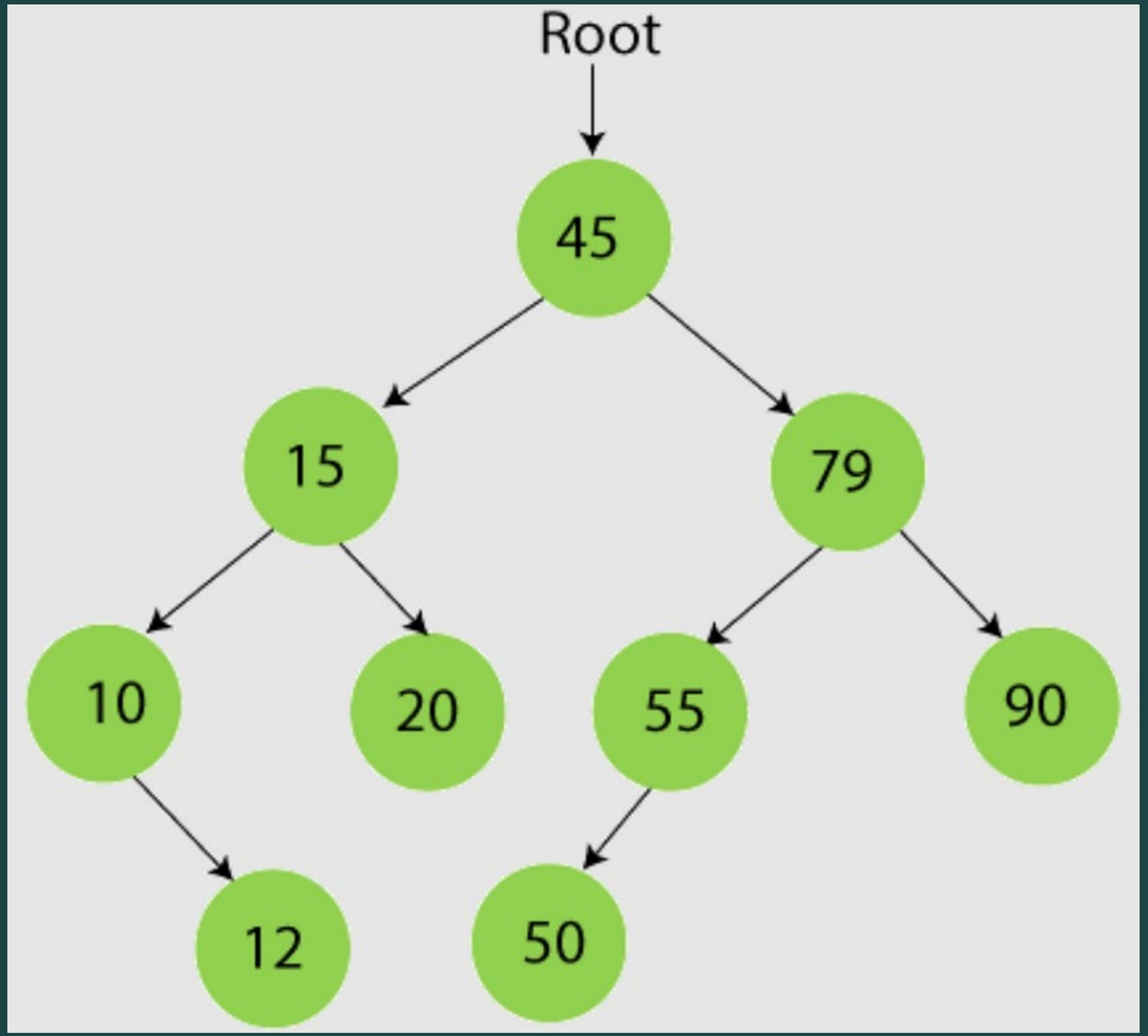
- Phần tử thêm vào được **tự động sắp xếp**
- Thao tác **thêm**, **xóa** phần tử **chậm** hơn so với **HashSet**, nhưng **tìm kiếm nhanh** hơn.



TreeSet

Cây tìm kiếm nhị phân

- Mỗi node có **tối đa hai nút con**(trái, phải).
- **Giá trị** của nút bên **trái** luôn **nhỏ hơn** giá trị của nút **cha**.
- Giá trị của nút bên **phải** luôn **lớn hơn** giá trị của nút **cha**.



Comparable & Comparator

- **Comparable** và **Comparator** giúp **so sánh** 2 đối tượng
  - Trả về **số nguyên**
    - **> 0**: Đối tượng 1 > Đối tượng 2
    - **< 0**: Đối tượng 1 < Đối tượng 2
    - **= 0**: Đối tượng 1 = Đối tượng 2
- **Khác biệt**
  - **Comparable**: **implements** trong class chính.
  - **Comparator**: **Không** cần **thay đổi** class chính.
- **Ứng dụng**: So sánh đối tượng theo nhiều tiêu chí, sắp xếp.

Lưu ý

- **Không** thể chứa phần tử **null**
- Đối với kiểu dữ liệu do **chúng ta tự định nghĩa** ra cần **implement** interface **Comparable** hoặc sử dụng **Comparator**

Map

- Sử dụng để lưu trữ và truy xuất theo cặp khóa (**key**) và giá trị (**value**)
- Mỗi cặp **key-value** được gọi là **entry**
- Map **không** cho phép 2 **key trùng lặp**
- **Mỗi key** tương ứng với **một value**.

Các triển khai của Map

- **HashMap**
  - **HashSet** được xây dựng trên nền tảng **HashMap**.
    - *PRESENT* = new Object(); // Đối tượng giả (dummy)
    - hashSet.add(element) <=> hashMap.put(element, *PRESENT*)

- **LinkedHashMap**
  - **LinkedHashSet** được xây dựng trên nền tảng **LinkedHashMap**.

- **TreeMap**
  - **TreeSet** được xây dựng trên nền tảng **TreeMap**.

DATABASE

THANK YOU