Kiểm tra 15 phút

Để quản lý hồ sơ học sinh của trường THPT, người ta cần quản lý những thông tin như sau:

- Các thông tin về: lớp, khoá học, kỳ học, và các thông tin cá nhân của mỗi học sinh.
- Với mỗi học sinh, các thông tin cá nhân cần quản lý gồm có: họ và tên, ngày sinh, quê quán.

1. Hãy xây dựng lớp School để quản lý các thông tin cá nhân của mỗi học sinh.
2. Cài đặt chương trình thực hiện các công việc sau:

- Nhập vào một danh sách gồm n học sinh (n nhập từ bàn phím)
- Hiển thị ra màn hình tất cả những học sinh sinh năm 1985 và quê ở Thái Nguyên
- Hiển thị ra màn hình tất cả những học sinh của lớp 10A1

Comparable

```
public static void main(String args[]) throws Exception {

        List<Course> a = new ArrayList();
        a.add(new Course("PRJ311", 110));
        a.add(new Course("DBI202", 150));
        a.add(new Course("PRF192", 120));
        Collections.sort(a);

}
```

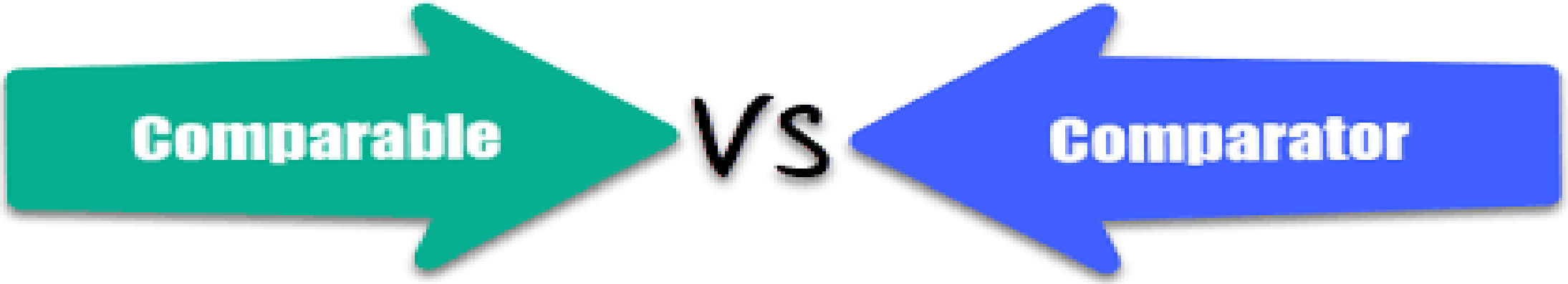
```
public class Course implements Comparable<Course>{
    private double fee;
    private String name;

    public Course(String name, double fee) {
        this.name = name;
        this.fee = fee;
    }

    public String getName() {
        return name;
    }

    public double getFee() {
        return fee;
    }

    @Override
    public int compareTo(Course o) {
        return (int) (this.fee - o.getFee());
    }
}
```

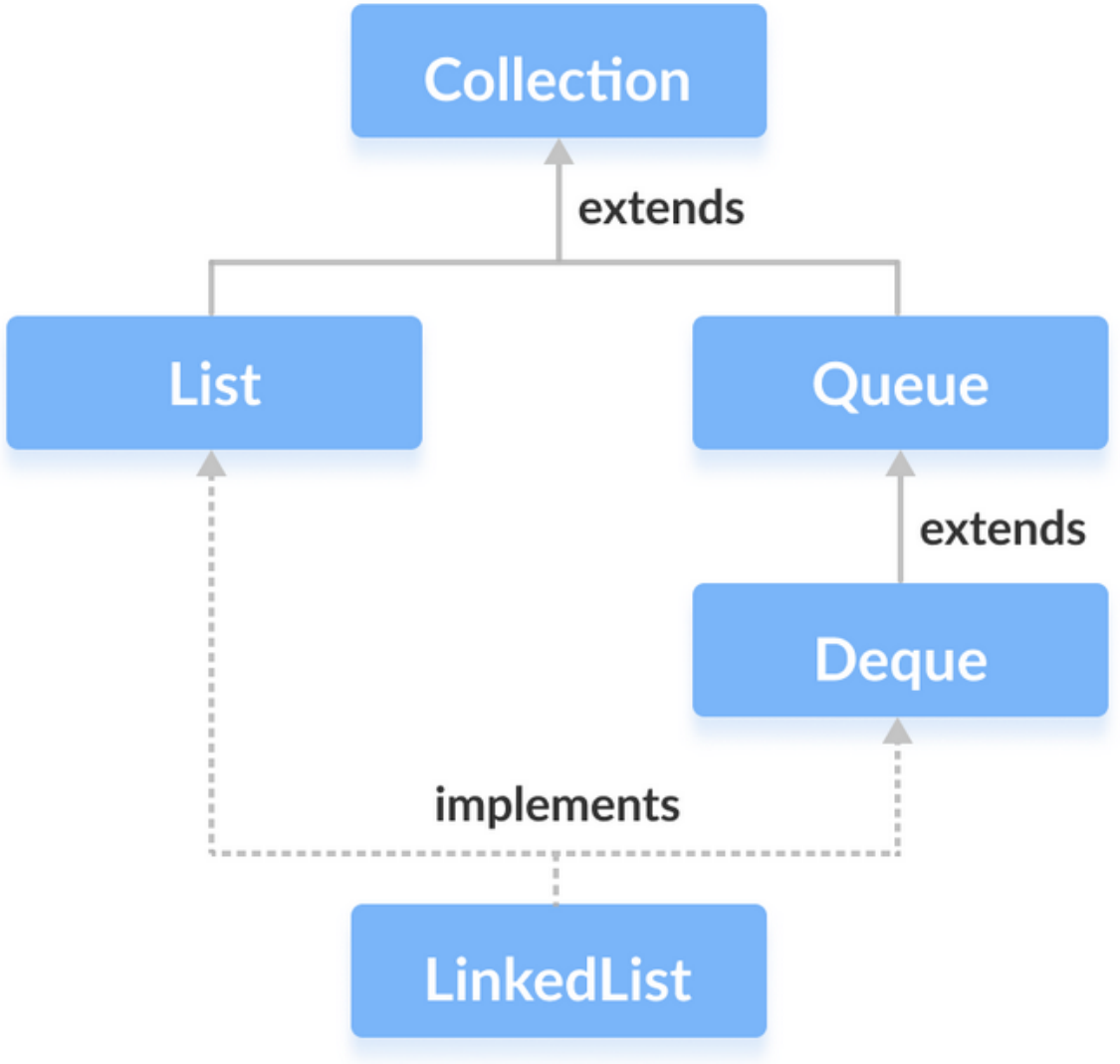


- Phải implements Comparable cho lớp đối tượng cần được so sánh
- Sử dụng compareTo() để sắp xếp các phần tử
- Collections.sort(List)

- Không cần implements Comparator cho lớp đối tượng cần được so sánh
- Sử dụng compare() để sắp xếp các phần tử
- Collections.sort(List, Comparator)

LinkedList

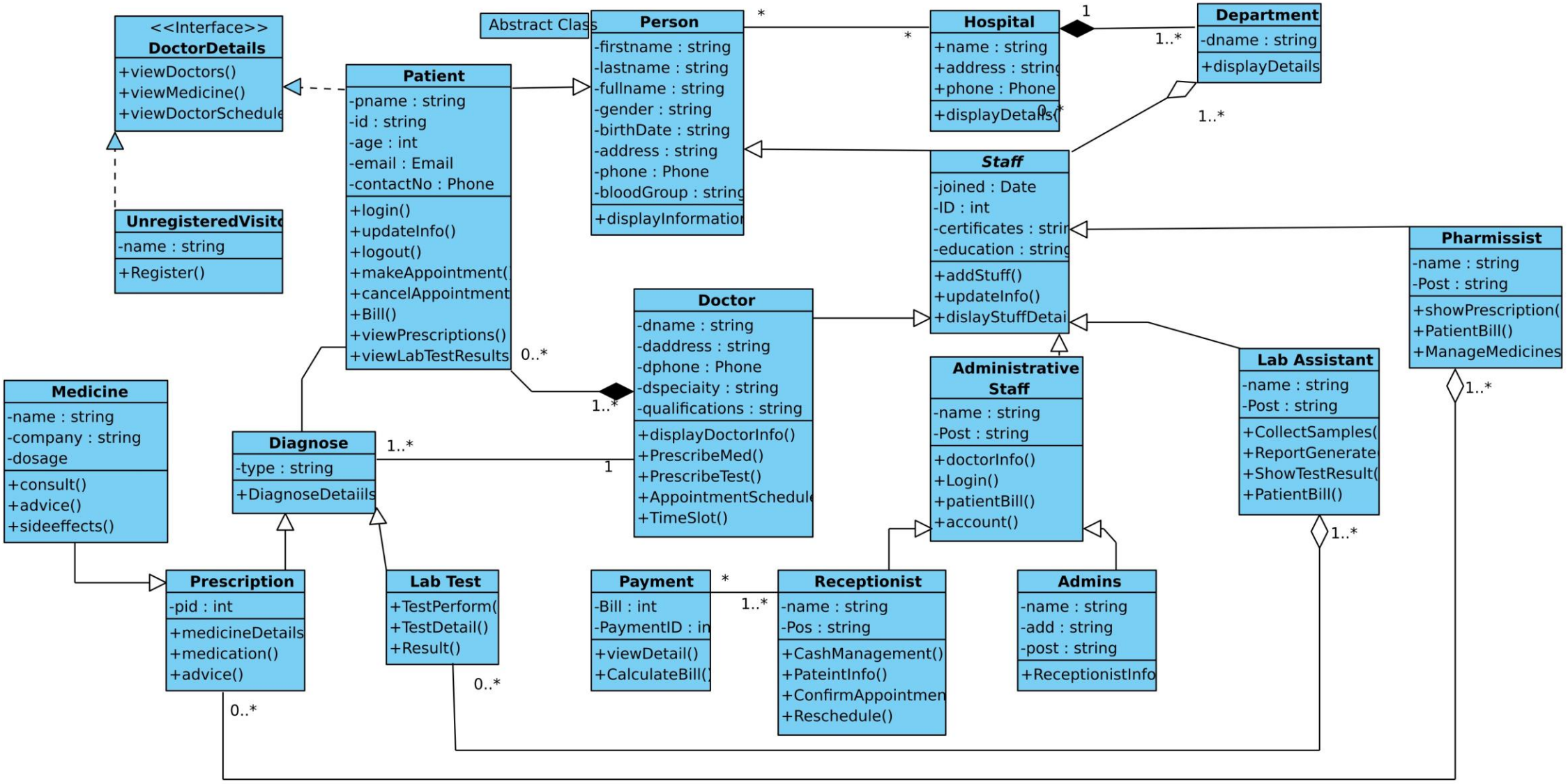
- LinkedList là một danh sách liên kết để lưu trữ phần tử. Mỗi phần thử được gọi là 1 node trong danh sách.



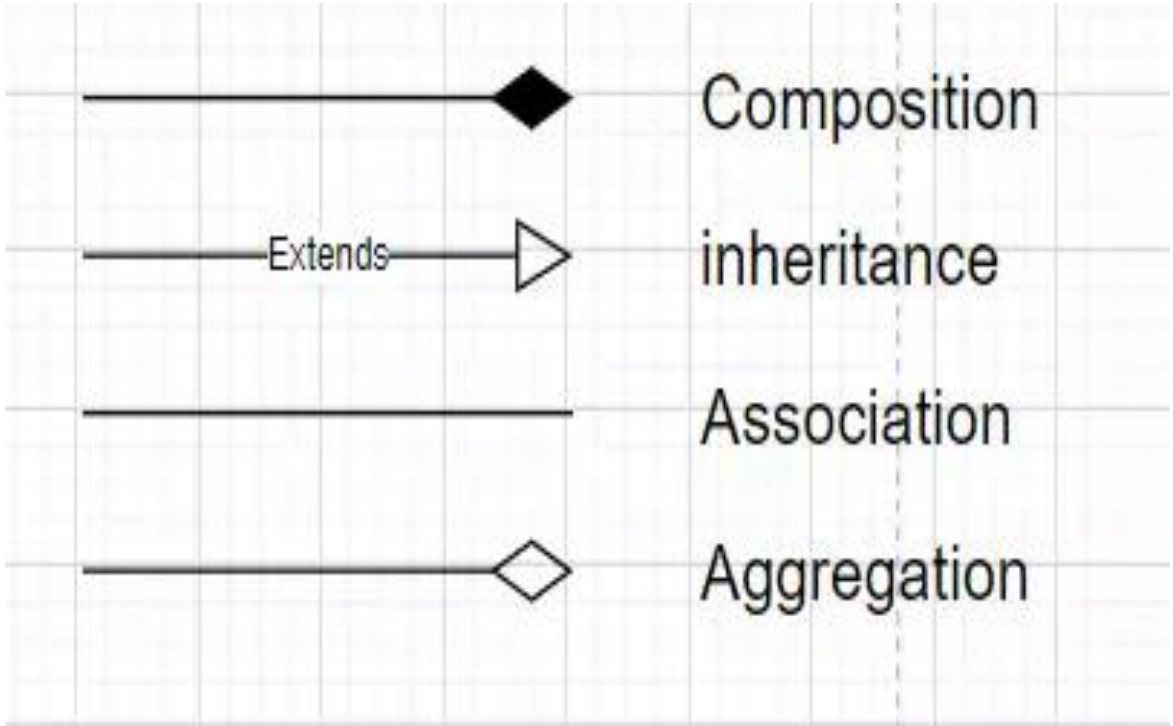
Cách khởi tạo

```
LinkedList<Integer> linkedList = new LinkedList<>();
List<Integer> list = new LinkedList<>();
Queue<Integer> queue = new LinkedList<>();
Deque<Integer> deque = new LinkedList<>();
```

Class Diagram

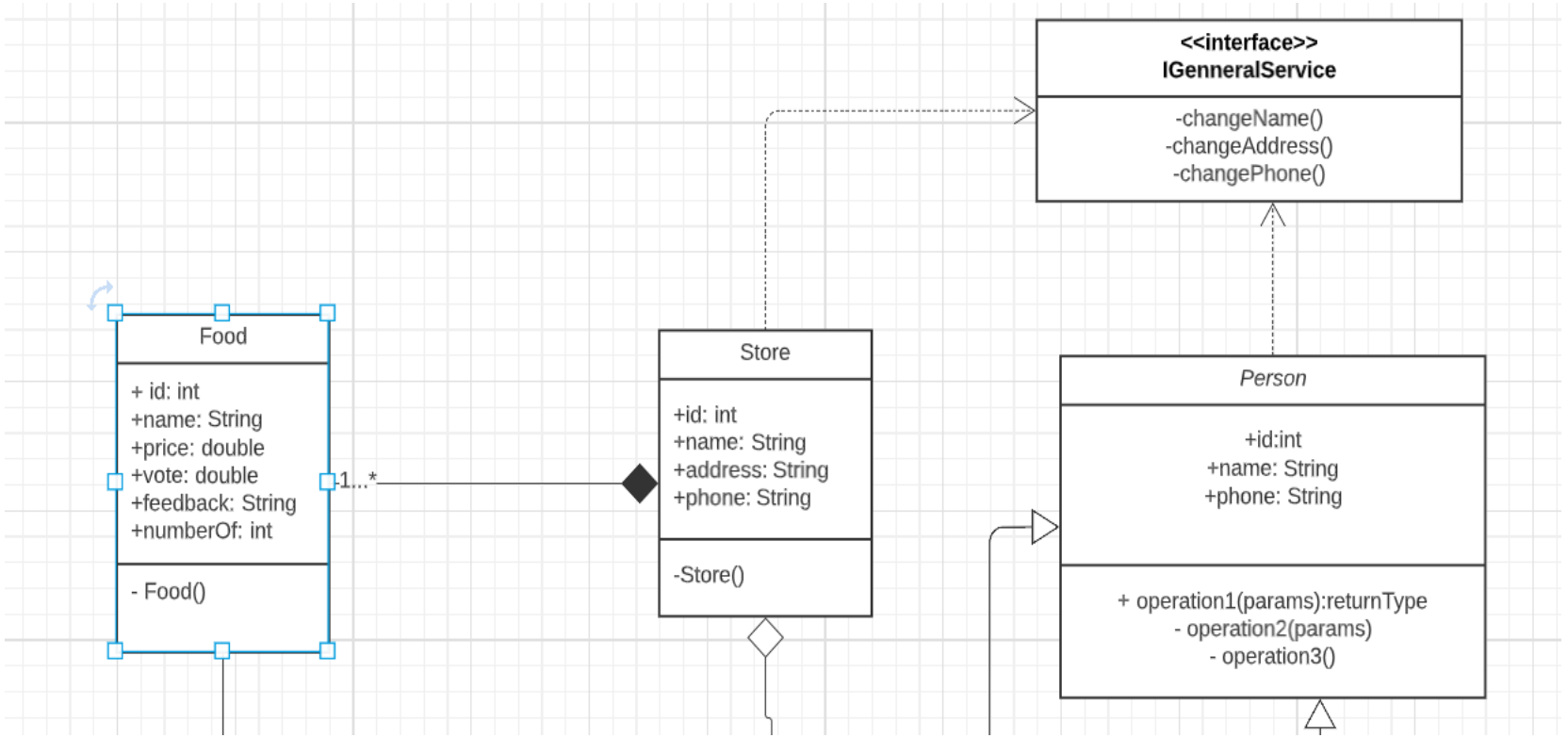


Chú thích



- Composition: Thể hiện mối quan hệ rằng obj A là một phần của obj B (Nếu B tồn tại thì A mới được tồn tại)
- Inheritance: kế thừa
- Association: Thế hiện A và B có mối liên hệ với nhau nhưng k rõ rang
- Aggregation: Thể hiện mối quan hệ rằng obj A là một phần của obj B (Nếu B không tồn tại thì A vẫn tồn tại)

Chú thích



0…*: Có thể không có hoặc có nhiều lớp đó
m...n: Có nhiều nhất n lớp và có ít nhất m lớp
VD: 1…4 : phải có ít nhất 1 lớp và nhiều nhất 4 lớp

Mũi tên nét đứt: implements

- : private
+ : public
# : protected
~ : default
ClassName: Tên class viết nghiêng thể hiện abstract class