1. Nêu các đặc điểm của Queue Interface, Dequeue Interface?

1.1. Queue Interface:

* Queue interface lưu trứ dữ liệu theo nguyên tắc FIFO, nghĩa là phần tử được thêm vào đầu tiên sẽ được xử lý đầu tiên.
* Không cho phép lưu giá trị null.
* Các phương thức chính**:**
  + add(E e): Thêm phần tử e vào hàng đợi.
  + offer(E e): Cũng thêm phần tử vào hàng đợi, trả về false nếu không thể thêm.
  + remove(): Lấy và loại bỏ phần tử đầu tiên của hàng đợi..
  + poll(): Lấy và loại bỏ phần tử đầu tiên của hàng đợi, trả về null nếu hàng đợi trống.
  + element(): Lấy nhưng không loại bỏ phần tử đầu tiên của hàng đợi. Ném NoSuchElementException nếu hàng đợi trống.
* Ưu điểm :
  + Lưu trữ và truy xuất phần tử theo thứ tự FIFO.
  + Sử dụng đa dạng các cấu trúc dữ liệu vd linkedlist và priority.
  + Có cơ chế thread-safe an toàn trong môi trường đa luồng.
* Nhược điểm :
  + Không phù hợp khi sử dụng với các tình huống yêu cầu các cấu trúc hoặc thuật toán phức tạp hơn cấu trúc FIFO.
  + Tiêu tón nhiều bộ nhớ hơn do phải lưu trữ thứ tự của các phần tử

1.2 Dequeue Interface?

* Deque cho phép các phần tử được thêm vào hoặc xóa khỏi cả hai đầu của hàng đợi (đầu và cuối).
* Ưu điểm :
  + Hỗ trợ cả 2 cấu trúc FIFO và LIFO.
  + Không hỗ trợ lưu giá trị null.
  + Cải thiện hiệu năng khi muốn thao tác thêm hoặc xoá ở 2 đầu dữ liệu.
* Nhược điểm :
  + Không có cơ chế đánh index cho phần tử trong hàng đợi , tốc đọ truy cập đến phần riêng lẻ chậm .
  + Không an toàn trong môi trường đa luồng.

1. Class triển khai từ Queue Interface, Dequeue Interface , phân biệt trường hợp sử dụng tương ứng.

2.1 Các class triển khai từ Queue Interface :

* PriorityQueue:
  + Hàng đợi ưu tiên. Các phần tử được sắp xếp tự nhiên hoặc theo thứ tự xác định bởi một comparator khi được thêm vào hàng đợi.
  + Không hỗ trợ null: Không thể chứa giá trị null.
  + Ứng dụng: Được sử dụng khi cần ưu tiên các phần tử theo giá trị hoặc thứ tự.
* LinkedList:
  + Triển khai cả List và Queue. Là một hàng đợi theo kiểu FIFO thông thường, đồng thời hỗ trợ các thao tác như danh sách liên kết.
  + Hỗ trợ null: Có thể chứa giá trị null.
  + Ứng dụng: Được sử dụng cho các hàng đợi bình thường và đơn giản.
* ArrayBlockingQueue:
  + Một hàng đợi giới hạn, nơi kích thước hàng đợi được xác định khi khởi tạo và không thể thay đổi sau đó.
  + Không hỗ trợ null.
  + Ứng dụng: Thường được dùng trong lập trình đa luồng, nơi cần quản lý tài nguyên giới hạn.
* PriorityBlockingQueue:
  + Giống PriorityQueue, nhưng được thiết kế để hoạt động an toàn trong môi trường đa luồng.
  + Không hỗ trợ null.
  + Ứng dụng: Sử dụng khi cần ưu tiên các tác vụ hoặc đối tượng trong lập trình đa luồng.

2.2 Các class triển khai từ Deque Interface:

* LinkedList:
  + Đặc điểm: Triển khai cả Deque và List. Hỗ trợ chèn và xóa ở cả hai đầu.
  + Hỗ trợ null.
  + Ứng dụng: Khi cần thao tác như hàng đợi hai đầu hoặc như ngăn xếp (stack).
* ArrayDeque:
  + Đặc điểm: Một triển khai của Deque sử dụng một mảng động, không giới hạn về kích thước (trừ khi bộ nhớ bị đầy).
  + Không hỗ trợ null.
  + Hiệu suất cao hơn LinkedList: Thực hiện nhanh hơn trong hầu hết các thao tác so với LinkedList.
  + Ứng dụng: Được dùng khi cần một hàng đợi hai đầu nhanh và không cố định về kích thước.