1. Các đặc điểm Map Interface**.**

* Map lưu trữ dữ liệu dưới dạng cặp key - value , trong dó key là duy nhất và ánh xạ đến 1 value duy nhất.
* Key là duy nhất và không có key trùng lặp trong 1 map .
* Không quan tâm đến thứ tự khi lưu trữ phần tử.
* Có thể lưu giá trị null cho key và value.

1. Class triển khai Map Interface và trường hợp sử dụng.

2.1. HashMap

* Mỗi key trong hashmap là duy nhất , nếu lưu cùng key thì map sẽ overide lại.
* Cho phép lưu giá trị null ở cả key và value.
* Không an toàn trong môi trường đa luồng
* Trường hợp sử dụng : HashMap rất phù hợp trong các tình huống cần hiệu suất cao và không yêu cầu đồng bộ hóa hoặc thứ tự phần tử.

2.2. LinkedHashMap

* LinkedHashMap duy trì thứ tự chèn của các phần tử, giúp dễ dàng truy xuất theo thứ tự các phần tử đã thêm vào.
* LinkedHashMap còn có tùy chọn để sắp xếp các mục dựa trên thứ tự truy cập.
* Hiệu suất tốt vì được xây dựng dựa trên hashCode.
* Làm cache hiệu quả: Như đã đề cập, với cơ chế dựa trên thứ tự truy cập, LinkedHashMap có thể được sử dụng để xây dựng bộ nhớ đệm (cache) .
* Hiệu năng chậm hơn hash map vì phải duy trì sắp xếp và truy cập .
* Trường hợp sử dụng : khi cần vừa truy xuất nhanh vừa duy trì thứ tự chèn hoặc truy cập

2.3. TreeMap

* TreeMap lưu trữ các phần tử theo thứ tự tự nhiên của các khóa (dựa trên việc sắp xếp tự nhiên hoặc dựa trên Comparator được cung cấp).
* TreeMap thích hợp khi bạn cần lấy dữ liệu theo thứ tự hoặc cần các thao tác phạm vi (ví dụ: lấy tất cả các khóa trong một phạm vi nhất định.
* Tiêu tốn nhiều bộ nhớ .
* không hỗ trọ kiểu dữ liêuh null.
* Trường hợp sử dụng :
  + Khi cần duy trì thứ tự tự nhiên hoặc sắp xếp tùy chỉnh của các khóa.
  + Khi bạn cần thao tác với một phạm vi của các phần tử, ví dụ như tìm kiếm trong một khoảng khóa nhất định.
  + Khi các thao tác truy xuất theo thứ tự khóa quan trọng đối với bài toán của bạn.