|  |  |  |  |
| --- | --- | --- | --- |
|  | **Vector** | **List** | **Deque** |
| **Khái niệm** | - bản chất là dynamic array (mảng động)  - có khả năng tự động thay đổi kích thước được chèn hoặc xóa.  - dùng bộ nhớ liền kề | - là danh sách chứa các đối tượng liên kết với nhau, chèn thêm xóa bất kì đối tượng nào trong danh sách  - Là double linked list | - là cấu trúc dữ liệu chứa 0 hoặc nhiều phần tử có cùng kiểu dữ liệu và được biểu diễn bằng một hàng có phần tử đầu (front) và phần tử cuối (last). |
| **Big O** | -Truy cập random: O(1)  - Thêm xóa random O(N) |  |  |
|  |  |  |  |
| **Ưu điểm** | - Đẩy dữ liệu vào cuối (push\_back)  - Truy suất random ez  - Thêm vào cuối, lấy ra ez | - Đẩy dữ liệu vào 2 đầu (push\_back, push\_front)  - Thêm xóa phần tử giữa mảng, đàu mảng, ko có nhu cầu truy suất random 1 cách thường xuyên | - Lưu dữ liệu dưới dạng FIFO (vào đầu ra trước)  - Xóa thêm phần tử đầu dãy (DPT 1 vector(n))  - Giống vector |
|  |  |  |  |
| **Nhược điểm** | - chèn phẩn tử chậm | - truy suất phần tử chậm, phải duyệt danh sách tới khi có được phần tử cần |  |
| **Mức độ sử dụng** | - Phổ biến nhất |  | - Có thể thay thế vector, CTDL này sinh sau nên ko phổ biến bằng |
| **Hàm** | push\_back() | push\_back() | push\_back() |
| Pop\_back() | Pop\_back() | Pop\_back() |
|  | push\_front() | push\_front() |
|  | Pop\_front | Pop\_front |
| .at(index) |  | .at(index) |

**1. Sequence containers: vector, list, deque**

**2. Accociative containers: map, set**

|  |  |  |
| --- | --- | --- |
|  | **Set/multiset** | **Map/multimap** |
| **Khái niệm** | **-** Là cấu trúc dữ liệu lưu những phần tử không trùng lặp, sắp xếp theo thứ tự tăng dần or giảm  - Cấu trúc cây nhị phân | - sắp xếp theo thứ tự cụ thể, mà mỗi phần tử trong đó được hình thành bởi sự kết hợp của một cặp khóa và giá trị (key & value), key duy nhất  - Cấu trúc cây nhị phân |
| **BIG O** | - Truy cập random: chậm  - Thêm xóa random: O(log N) |  |
| default | **Sx Từ bé đến lớn** |  |
| **Ưu điểm** | **-** Ko cho phần tử trùng lặp  - Thêm xóa tìm kiếm nhanh hơn vector | **-** 2 chỉ số, sắp xếp theo key, vùng bộ nhớ lớn hơn set |
| **Nhược điểm** | **-** Ko thể truy cấp ngẫu nhiên hoặc bất kì trong set  Tốc độ truy cập chậm hơn vector |  |
|  |  |  |
|  | **set<int> s(arr.begin(), arr.end())** |  |
|  | **Advance**(\*it,1): trỏ con trỏ tới index 1 |  |
|  | **s**.**Find**(k): tìm k  + ko tìm đc return end() của set  + tìm đc trỏ đến pt mang giá trị |  |
|  | **s.upper\_bound(k):**  + ko tìm đc return end() của set  + return iterator đến vị trí nhỏ nhất mà lớn hơn khóa **k.** |  |
|  | **s.ower\_bound(k):**  + ko tìm đc return end() của set  + return iterator đến vị trí nhỏ nhất, lớn hơn hoặc bằng khóa **k.** |  |
|  | Linh hoạt sử dụng | |

|  |  |  |
| --- | --- | --- |
|  | **Stack** | **queue** |
| **Khái niệm** | là một cấu trúc dữ liệu hoạt động theo nguyên tắc **LIFO –** vào sau ra trước | **FIFO –** vào trước ra trước |
|  | - Vị trí cuối cùng là đỉnh (top) |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Binary tree, linked list, nạp chồng toán tử**