| Cơ sở để so sánh | STACK | QUEUE |
| --- | --- | --- |
| Nguyên tắc làm việc | LIFO (Last in First out) | FIFO (First in First out) |
| Structure | Dùng một đầu để chèn và xóa các phần tử dữ liệu | Có 2 đầu để xử lý dữ liệu, một đầu chèn một đầu xóa |
| Số con trỏ được sử dụng | Một | Hai (Trong trường hợp đơn giản) |
| Hoạt động được thực hiện | Push và Pop | Enqueue và dequeue |
| Kiểm tra empty condition | Top == -1 | Front == -1 |
| Examination full condition | Top == Max - 1 | Rear == Max - 1 |
| Biến thể | Không có biến thể | Nó có các biến thể như hàng đợi tròn, hàng đợi ưu tiên, hàng đợi kết thúc gấp đôi. |
| Thực hiện | Đơn giản | Tương đối phức tạp |

Thực hiện Stack

Static implementation (triển khai tĩnh) sử dụng mảng để tạo stack. Triển khai tĩnh mặc dù là một kỹ thuật dễ dàng nhưng không phải là cách tạo linh hoạt, vì việc khai báo kích thước của ngăn xếp phải được thực hiện trong quá trình thiết kế chương trình, sau đó kích thước không thể thay đổi. Ngoài ra, triển khai tĩnh không hiệu quả lắm đối với việc sử dụng bộ nhớ. Vì một mảng (để thực hiện ngăn xếp) được khai báo trước khi bắt đầu hoạt động (tại thời điểm thiết kế chương trình). Bây giờ nếu số lượng phần tử được sắp xếp rất ít trong ngăn xếp, bộ nhớ được phân bổ tĩnh sẽ bị lãng phí. Mặt khác, nếu có nhiều số phần tử được lưu trữ trong ngăn xếp thì chúng ta không thể thay đổi kích thước của mảng để tăng dung lượng của nó, để nó có thể chứa các phần tử mới.

Dynamic implementation(triển khai động) cũng được gọi là biểu diễn danh sách được liên kết và sử dụng các con trỏ để thực hiện kiểu ngăn xếp của cấu trúc dữ liệu.

Thực hiện Queue

Static implementation (triển khai tĩnh): Nếu hàng đợi được triển khai bằng mảng, số lượng phần tử chính xác mà chúng tôi muốn lưu trữ trong hàng đợi phải được đảm bảo trước, bởi vì kích thước của mảng phải được khai báo tại thời điểm thiết kế hoặc trước khi quá trình xử lý bắt đầu. Trong trường hợp này, phần đầu của mảng sẽ đứng trước của hàng đợi và vị trí cuối cùng của mảng sẽ đóng vai trò là phía sau của hàng đợi. Mối quan hệ sau đây cung cấp cho toàn bộ các phần tử tồn tại trong hàng đợi khi được triển khai bằng cách sử dụng các mảng.

Dynamic implementation(triển khai động): thực hiện các hàng đợi bằng cách sử dụng các con trỏ, nhược điểm chính là một nút trong biểu diễn được liên kết sẽ tiêu tốn nhiều không gian bộ nhớ hơn một phần tử tương ứng trong biểu diễn mảng. Vì có ít nhất hai trường trong mỗi nút một cho trường dữ liệu và trường khác để lưu địa chỉ của nút tiếp theo trong khi trong biểu diễn chỉ có trường dữ liệu là có. Công dụng của việc sử dụng biểu diễn được liên kết trở nên rõ ràng khi được yêu cầu chèn hoặc xóa một phần tử ở giữa một nhóm các phần tử khác.

Như vậy Stack và Queue là các cấu trúc dữ liệu tuyến tính khác nhau theo các cách nhất định như cơ chế làm việc, cấu trúc, cách thực hiện, các biến thể nhưng cả hai đều được sử dụng để lưu trữ các phần tử trong danh sách và thực hiện các thao tác trong danh sách như thêm và xóa các phần tử.