1. **Stack 스택**

**특징**

LIFO (Last In First Out) 구조 - 한 쪽 끝에서만 자료를 넣고 뺄 수 있는 구조

스택에 데이터를 push하면 항상 top에 들어가고, pop하면 가장 최근에 push한 데이터가 나옴.

접근 가능한 요소는 맨 위 요소만 가능.

자료가 없을 때 pop하는 오류를 stack underflow, 스택의 크기 이상의 자료를 push 하려고 할 때의 오류를 stack overflow라고 함.

**시간 복잡도**

- 원소를 삽입/삭제하는 경우 : O(1)

**장점**

데이터의 삽입과 삭제가 빠르다. (맨 위 원소 접근 O(1))

**단점**

맨 위 요소만 접근이 가능함.

그래서 전체 탐색을 하려면 원소를 하나하나 꺼내서 옮겨가면서 해야함.

**사용법**

재귀 알고리즘에서 유용하게 사용

역추적할 때 (문서 작업 시 실행 취소)

1. **Queue 큐**

**특징**

FIFO(First-In-First-Out) 구조 - 먼저 넣은 데이터가 먼저 나오는 구조

Queue의 사전적 정의는 줄. 기다리는 줄에서 먼저 선 사람이 먼저 나갈 수 있는 것처럼, 먼저 들어간 데이터가 먼저 나가는 것에서 붙여진 이름.

데이터가 삽입(push)되는 곳을 front, 제거(pop)되는 곳을 back이라고 한다.

**시간 복잡도**

- 원소를 삽입/삭제하는 경우 : O(1)

**장점**

데이터의 삽입과 삭제가 빠르다.

**단점**

Queue의 중간에 위치한 데이터로의 접근이 어렵다.

- 선형 큐 :

Front는 고정, Back을 이동하면서 데이터를 삭제하는 경우: 데이터를 제거했을 때, 나머지 데이터를 한 칸마다 전부 옮겨야 함.

둘 다 이동하면서 삽입, 삭제를 할 경우, 배열의 끝에 저장되어 있는 상황에서 Back을 더 이상 이동시킬 수 없어서 overflow 발생.

- 순환 큐(환형 큐) : 선형 큐를 보완하기 위한 방식. front가 큐의 끝에 닿으면 큐의 맨 앞으로 자료를 보내서 원형으로 연결.

**사용법**

데이터를 입력된 순서대로 처리해야 할 때

BFS (너비 우선 탐색) 구현할 때

1. **Deque 덱**

**특징**

Deque(Double Ended Queue), queue와 비슷하지만 queue는 front에서만 삭제하고, end에서 삽입하는데, deque는 front와 end에서 삭제와 삽입이 모두 가능하다.

연속적인 메모리를 기반으로 하는 '시퀀스 컨테이너'이다. 따라서, 임의 접근 반복자 제공.

여러 개의 메모리 단위로 데이터를 저장한다. vector는 메모리를 재할당하고 모든 요소를 복사하여야 하는데, deque는 새로운 메모리 단위를 할당하여 요소를 추가한다.또 데이터 요소를 저장하는 여러 개의 메모리 단위를 갖습니다.

크기가 가변적이다. (선언 후에 변경할 수 있다.)

중간 요소가 삽입, 삭제될 때, 요소들을 앞/뒤로 밀 수 있으므로 vector보다는 좋은 성능을 갖음. 그래도, 앞/뒤에서의 삽입/삭제 성능은 좋지만 중간에서는 좋지 않다.

**시간 복잡도**

삽입/삭제

- 원소를 앞/뒤에 삽입하는 경우 : O(1)

삭제

- 원소를 앞/뒤에 삽입하는 경우 : O(1)

탐색

- 원소를 탐색하는 경우 : O(1) (index 접근)

**장점**

데이터의 삽입과 삭제가 빠르다.

크기가 가변적이다.

앞, 뒤에서 데이터를 삽입/삭제할 수 있다.

index로 임의 원소 접근이 가능하다.

새로운 원소 삽입 시에, 메모리를 재할당하고 복사하지 않고 새로운 단위의 메모리 블록을 할당하여 삽입한다.

**단점**

deque의 중간에서의 삽입과 삭제가 어렵다.

구현이 어렵다

**사용법**

앞과 뒤에서 삽입, 삭제가 자주 일어나는 경우

데이터의 개수가 가변적일 경우ㄴ

데이터 검색을 거의 하지 않을 경우 (랜덤 요소에 접근해야할 때)

참고사이트:

<https://ko.wikipedia.org/wiki/%EC%8A%A4%ED%83%9D>

<https://ko.wikipedia.org/wiki/%ED%81%90_(%EC%9E%90%EB%A3%8C_%EA%B5%AC%EC%A1%B0)>

<https://blog.naver.com/justkukaro/220515795433?proxyReferer=https%3A%2F%2Fwww.google.com%2F>

<https://velog.io/@choiiis/%EC%9E%90%EB%A3%8C%EA%B5%AC%EC%A1%B0-%EC%8A%A4%ED%83%9DStack%EA%B3%BC-%ED%81%90Queue>

<http://blog.naver.com/PostView.nhn?blogId=sooftware&logNo=221516440423&parentCategoryNo=&categoryNo=9&viewDate=&isShowPopularPosts=true&from=search>