#algorithms #datastructure

자료구조

정의

- 컴퓨터에 자료를 저장하고 접근하는 방식을 말한다.

종류

- 배열, 스택, 큐, 해시, 트리, ...

배열

정의

- 배열은 차원과 무관하게 메모리에 연속 할당된 자료구조
- 배열에서 원소들의 순서는 실제 원소들이 저장된 메모리 주소 순서와 일치하며 끊기지 않고 연결된다.

차원이란?

- 물리학 용어 차원과 유사하다. 1차원은 점, 2차원은 점이 모여서 이루어진 선, 3차원은 선이 모여서 이루어진 면. 이처럼 배열 또한 차원이 커질 수록 하위 차원이 모여서 이루어졌다.

```
// 물리학의 점(dot)을 다루는 자료형이 있다고 가정하자
dot arr[2] // 점 두 개를 저장할 수 있는 1차원 배열.
dot arr[3][2] // 점 두 개가 저장된 1차원 배열을 3개 저장할 수 있는 2차원 배열.
dot arr[4][3][2] // 선 3개를 저장할 수 있는 2차원 배열을 4개 저장할 수 있는 3차원 배열
```

배열의 효율성

- 데이터 접근 : O(1). 배열은 각 원소에 접근할 때, 단번에 닿을 수 있다.
- 삽입 : O(max - cur + 1). 배열은 메모리를 순차적으로 저장한다. 그렇기에 삽입 위치(cur)에서 뒤에 있는 원소들의 메모리 위치를 한 칸 씩 밀어야 한다.
- 삭제 : 삽입과 마찬가지. 왜냐하면 삭제하려는 원소들의 뒤에 있는 원소를 전부 메모리에서 한 칸 씩 앞으로 당겨야 하기 때문이다.

배열을 사용할 때 고려할 점

- 메모리 크기 확인 : 문제에서 요구하는 바를 그대로 적용했을 때, 너무 배열이 커지면 메모리 초과. 한 번 어림짐작으로 계산해보고 안 되면 다른 수를 강구하라.
- 중간에 데이터 삽입이 많은가? : 배열은 데이터 삽입/삭제에서 가장 많은 시간이 소모되는 자료구조.

풀어보자

- 두 개를 뽑아서 더하기 : https://programmers.co.kr/learn/courses/30/lessons/68644
- 모의고사 : https://programmers.co.kr/learn/courses/30/lessons/42840

스택

정의

- 스택은 일방향으로 데이터를 쌓고 빼는 자료구조
- 후입선출(LIFO) 방식으로 데이터를 뺄 때는 들어간 순서의 역순으로 가장 마지막에 넣은 데이터부터 뺄 수 있다

스택 구조

- 스택은 기본적으로 원소를 넣기(push), 빼기(pop), 가장 최근에 넣은 값 주소 보관(top), 비어있는 지(empty), 가득 찼는지(full)를 확인하는 함수들을 포함한다

```
// 스택 선언
stack st;

// 스택에 자료를 넣는다.
st.push(10);
st.push(20);
st.push(30);

// 스택이 비어있을 때까지 꼭대기(마지막 값) 값을 출력하고 뺀다.
while (!st.empty()) {
    printf(st.top());
    st.pop();
}
```

- 위 예시를 보고 직접 자신이 쓰는 프로그래밍 언어로 스택을 사용해보자

스택 사용 시 고려할 점

- 중간값 접근 X : 스택은 가장 끝값부터 참조할 수 있다. 중간값을 보려면 스택 원소들의 메모리 주소를 전부 외우던가 아니면 구현상 허점을 파고들어야 한다. 어느 쪽이든 스택 자료구조를 사용하는 본질에서 벗어나므로 피하자.
- 원소 삽입, 삭제 시 주의 : 삭제 시에는 스택 공간이 비어 있는 지(empty), 삽입 시에는 스택 공간이 가득 찼는 지(full) 확인하는 버릇을 들이는 게 좋다. 내가 사용하는 스택 라이브러리가 해당 행위에 대한 별도 예외처리가 되어 있는 지 모른다면 더더욱.

풀어보자

- 짝지어 제거하기 : https://school.programmers.co.kr/learn/courses/30/lessons/12973

- (난이도 상) 표 편집 : https://school.programmers.co.kr/learn/courses/30/lessons/81303

큐

정의

- 큐는 일방향으로 데이터를 쌓아 빼는 자료구조
- 선입선출(FIFO) 방식으로 스택과 다르게 가장 먼저 삽입한 원소부터 빠져나갈 수 있다

큐 구조

- 큐는 기본적으로 스택과 유사하나 값을 확인하는 방식이 보통 front와 rear로 나뉜다. 왜냐하면 큐 특성 상 push를 하면 마지막 값(rear)에 변동이 생기고 pop을 하면 첫번째 값(front)에 변동이 생기기 때문이다. 그래서 양쪽 값 주소를 전부 알 필요가 있다.

```
// 큐 선언
queue q;

// 큐에 자료를 넣는다
q.push(10);
q.push(20);
q.push(30);

// 큐가 완전히 빌 때까지 첫번째 값을 뺀다
while (!q.empty()) {
    printf(q.front());
    q.pop();
}
```

큐 사용 시 고려할 점

- 스택과 유사한 점을 고려한다.

풀어보자

- https://school.programmers.co.kr/learn/courses/30/lessons/159994