**- Array와 LinkedList가 각각 무엇인지, 어떻게 다른지 설명해주세요.**

- Array는 순서가 있는 데이터를 저장하기 위한 선형 자료구조이며, 고정 길이의 크기를 가진 자료구조입니다. 순서가 있기 때문에 각 데이터의 순서를 의미하는 Index가 존재하며, Index를 통해 해당 데이터에 접근하는 Random Access를 지원합니다. Index를 통해 값을 색인하면 O(1)의 시간복잡도가 소요됩니다.

- 하지만, 중간에 값을 삽입 혹은 삭제에 대해선, 뒤의 요소들을 모두 한 칸씩 밀거나 당겨야 하므로 Worst Case의 경우 O(N)의 시간복잡도를 갖습니다.

- 고정길이, 느린 삽입 삭제 연산을 해결하기 위해서 나온 자료구조가 바로 LinkedList 입니다.

- 데이터를 저장하고 있는 각 노드를 서로 연결 시켜 만든 선형 자료구조로, 가변 길이의 크기를 가지고 있습니다.

- 중간에 값을 삽입하는 과정에서 기존의 연결을 끊고, 새로운 연결을 맺는 과정만 하면 되므로 O(1)의 시간 복잡도, 삭제하는 과정에서도 기존의 연결을 끊고 그 뒤의 노드와 연결하는 O(1)의 시간복잡도를 갖습니다.

- 반면 특정 값을 찾는 과정은 연결리스트 내부를 모두 완전탐색해야 하므로 worst case의 경우 O(N)인 단점이 존재합니다.

**- Stack과 Queue에 대해 설명하고, 각각 어디에 사용되었는지 설명해주세요.**

- 스택은, 선입후출의 특성을 가진 자료구조입니다. 스택을 구현하기 위해서는 배열 / 연결리스트 사용이 가능하며, 주로 배열을 통해 구현합니다. 스택이 실제로 사용되는 부분은 브라우저의 '뒤로가기' 기능이나, 메모리 영역의 스택 영역에 사용됩니다.

- 큐는, 선입선출의 특성을 가진 자료구조입니다. 큐를 구현하기 위해서는 배열 / 연결리스트 사용이 가능하며, 주로 연결리스트를 통해 구현합니다. 큐가 실제로 사용되는 부분은 프린터 큐, CPU 스케줄링의 Ready Queue 등에 사용됩니다.

**- Priority Queue가 무엇이고, 어떻게 구현할 수 있는지 설명할 수 있나요?**

- 선입선출의 구조를 가진 큐와 다르게, 우선순위가 높은 원소가 우선적으로 빠져나오게 되는 자료구조입니다.

- 우선순위 큐는 배열, 연결리스트, Heap으로 구현할 수 있으며, 일반적인 경우에는 삽입과 삭제 연산이 모두 O(logN)이 보장된 Heap 자료구조를 사용해서 우선순위 큐를 구현합니다.

**- 해시 테이블(Hash Table)에 대해서 설명할 수 있나요?**

- 해시 테이블은 Key-Value 쌍으로 값을 저장하는 자료구조로, 검색 시간 복잡도로 Θ(1)를 갖고 있는 것이 특징입니다.

- 해시 테이블에 접근할 수 있는 Key와, Key와 매핑되어 있는 Value로 저장되어 있는 것이 특징입니다.

**- 해시 테이블(Hash Table)의 검색 시간 복잡도는 어떻게 O(1)을 유지할 수 있나요?**

- 해시 테이블에 Key로 접근하면, Hash Function에 Key 값을 집어 넣어 나온 Hash 값을 저장소의 인덱스로

활용합니다. 해당 인덱스에 값을 저장하기 때문에, Hash Function의 시간 복잡도를 제외하면 시간 복잡도

O(1)을 보장할 수 있습니다.

**- 체이닝 방식과 개방주소법의 장단점을 비교해주세요.**

- 체이닝 방식 :

장점 :

1) 한정된 저장소(Bucket)을 효율적으로 사용할 수 있다.

2) 해시 함수(Hash Function)을 선택하는 중요성이 상대적으로 적다.

3) 상대적으로 적은 메모리를 사용한다. 미리 공간을 잡아 놓을 필요가 없다.

단점 :

1) 한 Hash에 자료들이 계속 연결된다면(쏠림 현상) 검색 효율을 낮출 수 있다.

2) 외부 저장 공간을 사용한다.

3) 외부 저장 공간 작업을 추가로 해야 한다.

- 개방주소법 방식 :

장점 :

1) 또 다른 저장공간 없이 해시테이블 내에서 데이터 저장 및 처리가 가능하다.

2) 또 다른 저장공간에서의 추가적인 작업이 없다.

단점 :

1) 해시 함수(Hash Function)의 성능에 전체 해시테이블의 성능이 좌지우지된다.

2) 데이터의 길이가 늘어나면 그에 해당하는 저장소를 마련해 두어야 한다.

**- BST의 성능을 개선하기 위한 방법에는 무엇이 있나요?**

위에서 언급했듯이, BST의 양쪽 높이의 균형을 맞추는 것이 성능을 개선하는 방식입니다.

대표적으로 개량된 BST 트리로써, AVL 트리, Red-Black 트리가 있습니다.