자료구조와 알고리즘

▶ 자료구조란 무엇인가?

✔ 템플릿

직관적이지 않은 동적 자원를 구성하는 코드때문에 템플릿 사용하여 페이지 작성

✔ 데이터 베이스

- 웹 서버는 동적 자원 코드 실행 시 종종 데이터베이스에서 데이터 로드
- 웹 서버와 데이터베이스 사이의 인터페이스는 해커들의 목표 대상이 된다.
- SQL 데이터 베이스, NoSql 데이터 베이스

자료구조란

▶ 자료구조란 무엇인가?

자료구조는 데이터를 효율적으로 저장하고 조직화하는 방법을 뜻한다.
자료구조는 알고리즘의 성능을 좌우하며, 올바른 자료구조 선택은 시스템의 성능에 큰 영향을 미친다.

✔ 자료구조와 알고리즘의 관계

자료구조는 데이터를 저장하는 방식이며, 알고리즘은 그 데이터에 대한 연산 방법을 정의

재귀와 귀납적 사고

▶ 재귀와 귀납적 사고

✔ 재귀

함수내부에서 자기 자신을 호출하여 문제를 해결하는 기법

문제를 더 작은 부분 문제로 나누어 해결할 때 유용

✔ 재귀와 수학적 귀납법

수학적 귀납법은 무한히 많은 경우에 대해 어떤 명제가 참임을 증명하는 도구

간단히 말해서, 하나의 작은 경우부터 출발해 모든 경우에 대해 참임을 차근차근 증명하는 방식

재귀와 수학적 귀납법은 구조적으로 유사하며, 재귀적 알고리즘의 정확성을 증명하는 데

수학적 귀납법이 자주 사용

알고리즘 복잡도

▶ 알고리즘 복잡도

✔ 알고리즘 복잡도란?

함수알고리즘이 실행되는 시간과 공간을 측정하는 척도

- 시간복잡도 : 입력 크기에 따른 알고리즘 실행 시간의 증가율
- 공간 복잡도: 알고리즘이 필요로 하는 메모리 공간을 측정
- Big-O 표기법: 알고리즘의 최악의 경우 성능을 분석하는 도구

리스트

▶ 리스트란?

자료구조에서 가장 기본적이고 중요한 형태 중 하나로 리스트는 데이터를 순차적으로 저장하는 자료구조이다.

리스트는 여러 개의 데이터를 순차적으로 저장할 수 있으며, 각 데이터는 인덱스를 통해 접근할 수 있다.

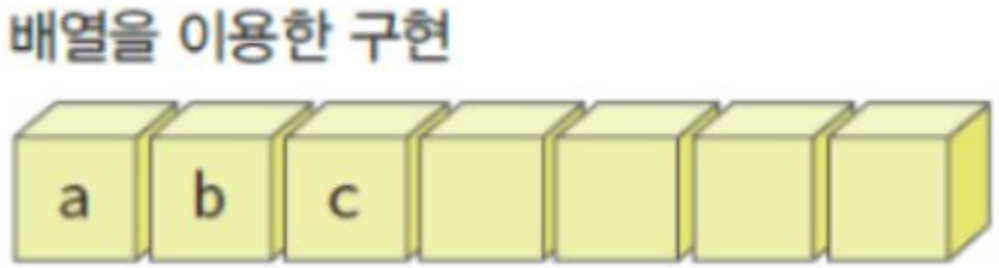
배열 리스트와 연결 리스트가 있다.

▶ 배열 리스트

고정된 크기의 메모리 블록에 데이터를 순차적으로 저장하고 인덱스로 접근 가능하다.

ArrayList

- 장점: 인덱스를 통한 접근 속도가 빠르며, 메모리 사용이 효율적입니다.
- 단점: 크기를 미리 정해야 하며, 크기를 변경할 때 전체 요소를 새 배열로 복사해야 하므로 비효율적입니다. 새로운 요소 추가,제거시 다른 요소들을 밀거나 당겨 위치를 이동해야 한다.



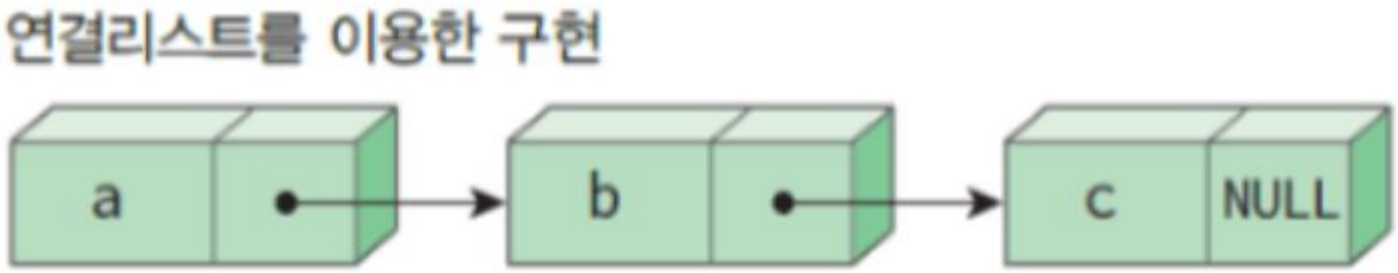
▶ 연결 리스트

요소가 노드로 구성되며, 각 노드는 데이터와 다음 노드에 대한 주소를 가지고 있다.

LinkedList

- 장점: 동적으로 크기를 조절할 수 있으며, 삽입 및 삭제 연산이 용이합니다.
- 단점: 메모리 사용이 비효율적이며, 인덱스를 통한 접근 속도가 느립니다(순차 탐색).

단일 연결 리스트, 이중 연결 리스트, 원형 연결 리스트



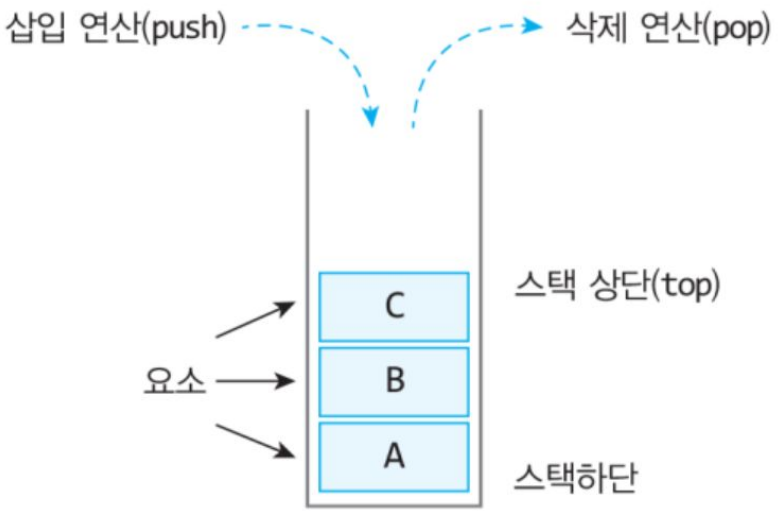
스택 & 큐

▶ 스택

✔ 스택이란?

스택(Stack)은 데이터의 저장 및 접근 방식을 정의하는 자료구조로,
후입선출(LIFO, Last In First Out) 원칙에 따라 동작한다.

주요 연산: Push(삽입), Pop(삭제)



▶ 큐

✔ 큐란?

큐(Queue)는 데이터 구조에서 선입선출(FIFO, First In First Out) 원칙에 따라 동작하는 자료구조이다. 먼저 들어간 데이터가 먼저 나오는 구조이다.

주요 연산 : 인큐, 디큐, 피크, isEmpty

▶ 우선순위 큐

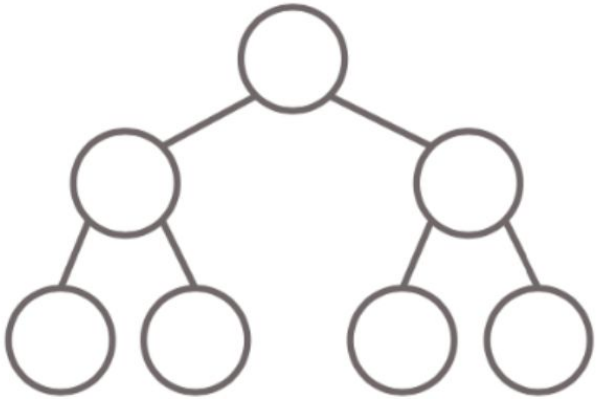
✔ 우선 순위 큐란?

우선순위 큐는 각 요소에 우선순위를 부여하여, 우선순위가 높은 요소가 먼저 처리되는 큐이다.

우선순위 큐에서는 요소가 추가되는 순서와 관계없이 우선순위에 따라 요소가 제거된다.

주요연산 : 삽입, 삭제, peek, isEmpty

일반적인 큐



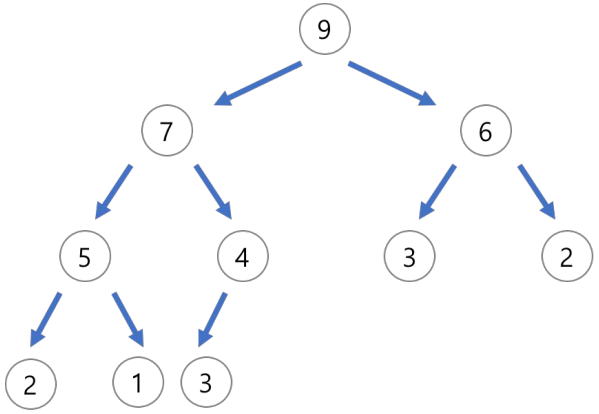
우선순위 큐

▶ 힙

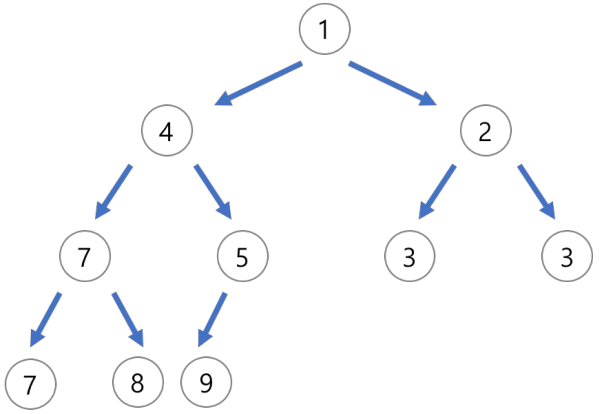
✔ 힙이란?

힙(Heap)은 완전 이진 트리의 한 형태로, 특정한 성질을 가지고 있는 자료구조
주로 우선순위 큐를 구현하는 데 사용

최소 힙(min heap) : 부모 노드의 값이 자식 노드의 값보다 작거나 같다.
최대 힙(max heap) : 모 노드의 값이 자식 노드의 값보다 크거나 같다.



-최대 힙(max heap)-



-최소 힙(min heap)-

정렬

▶ 정렬

✔ 정렬이란?

데이터 집합을 특정 기준에 따라 특정한 순서로 재배열하는 과정

정렬된 데이터는 검색,분석에 용이하고, 데이터를 효율적으로 관리할 수 있다.

정렬 기준 : 오름차순(Ascending), 내림차순(Descending)

▶ 정렬

✔ 정렬 알고리즘

주어진 데이터 집합을 특정한 기준에 따라 정렬하는 방법을 제시하는 알고리즘.

정렬 알고리즘은 데이터 구조와 관련된 중요한 개념으로, 데이터의 정렬 방식에 따라 검색 속도와 데이터 처리 효율성이 다르다.

✔ 주요 정렬 알고리즘

- 선택 정렬 (Selection Sort) : 현재 위치에서 가장 작은 값을 찾아 해당 위치에 배치하여 정렬
- 버블 정렬(Bubble Sort) : 인접한 두 요소를 비교하여 큰 값을 뒤로 보내 정렬.
- 삽입 정렬(Insertion Sort) : 각 요소를 적절한 위치에 삽입하여 정렬.
- 퀵 정렬(Quick Sort) : 피벗을 기준으로 데이터를 분할하여 정렬.
- 병합 정렬(Merge Sort) : 데이터를 반으로 나눈 후 병합하여 정렬.

색인과 이진 검색 트리

▶ 색인과 이진 검색 트리

✔ 이진 검색 트리

색인(Indexing)이란 데이터베이스, 검색 엔진, 파일 시스템 등에서 데이터를 효율적으로 검색할 수 있도록 돕는 구조.

검색속도 향상으로 원하는 데이터를 빠르게 찾을 수 있다.

✔ 색인 특징

- 검색 효율성: 색인을 사용하면 전체 데이터 집합을 탐색하지 않고도 원하는 데이터를 빠르게 검색 가능
- 다양한 형태: 색인은 해시 테이블, B-트리, 비트맵 색인 등 여러 형태로 구현.
- 동적 업데이트: 데이터가 추가되거나 삭제될 때 색인을 업데이트하는 비용이 발생. 색인 유지 관리의 주요 요소

▶ 색인과 이진 검색 트리

✔ 트리란?

트리(Tree)는 계층적 데이터 구조의 하나로, 노드(Node)들로 구성
각 노드는 데이터와 자식 노드에 대한 참조를 가집고 트리는 부모-자식 관계를 기반으로 하여 데이터를 조직하는 데 사용

트리 종류: 이진트리(Binary Tree), 이진 검색 트리(Binary Search Tree), B-트리,
균형 이진 트리(Balanced Binary Tree) 등

✔ 트리 장단점

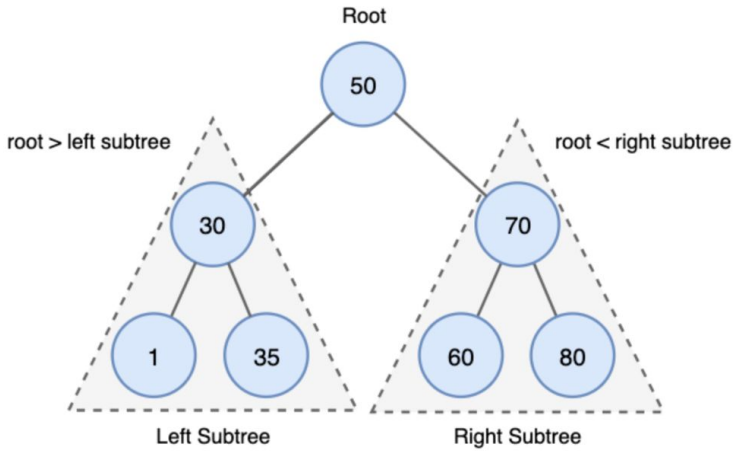
장점: 계층적 데이터 표현, 효율적 데이터 검색, 동적 데이터 관리
단점: 메모리 사용, 구현의 복잡성, 최악의 경우 성능 저하

▶ 색인과 이진 검색 트리

✔ 이진 검색 트리

원소가 정렬되어 있다는 특징을 이용해 반씩 쪼개가며 탐색하는 이진 탐색(Binary Search)의 아이디어를 채용한 자료구조

왼쪽 서브트리: 부모 노드의 키 값보다 작은 모든 노드를 포함
오른쪽 서브트리: 부모 노드의 키 값보다 큰 모든 노드를 포함



▶ 색인과 이진 검색 트리

✔ 이진 검색 트리 장점

- 효율적인 검색: 정렬된 구조 덕분에 평균적인 검색 성능이 뛰어나다.
- 동적 데이터 관리: 삽입과 삭제가 용이하여 데이터의 동적 관리가 가능하다.
- 직관적인 구조: 이진 트리이므로 구조가 간단하고 이해하기 쉽다.

✔ 이진 검색 트리 단점

- 불균형 상태: 삽입 순서에 따라 트리가 편향될 수 있어 성능이 저하될 수 있다.
- 균형 유지 필요: 최적의 성능을 위해 균형을 유지해야 하며, 이는 구현이 복잡해질 수 있다.

그래프

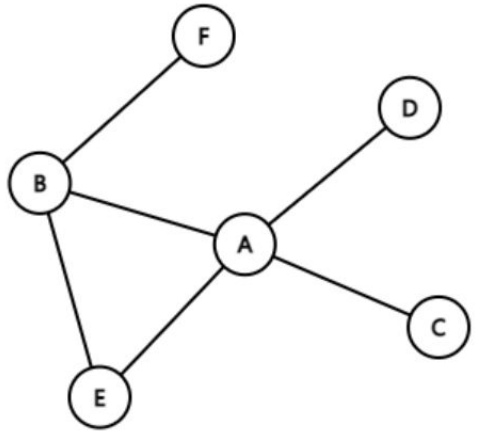
▶ 그래프

✔ 그래프란?

그래프(Graph)는 노드(정점)와 이들 간의 관계를 나타내는 엣지(간선)로 구성된 비선형 데이터 구조이다.
네트워크, 소셜 미디어, 지도 등과 같은 복잡한 관계를 표현하는 데 널리 사용

✔ 기본 구성

- 정점(Vertex) : 그래프의 기본 단위
  데이터 또는 개체를 표현
- 간선(Edge) : 정점 간의 관계를 나타내는 선
  두 정점을 연결



▶ 그래프

✔ 그래프 종류

- 유향 그래프(Directed Graph): 각 간선에 방향이 존재, 특정한 흐름이나 관계표현
- 무향 그래프(Undirected Graph): 간선이 방향이 없고, 두 정점 간의 관계가 대칭적
- 가중치 그래프(Weighted Graph): 각 간선에 가중치(비용)가 할당, 최단 경로 문제와 같은 다양한 문제를 해결하는 데 사용
- 비가중치 그래프(Unweighted Graph): 간선에 가중치가 없고 관계의 존재만을 표현

▶ 그래프

✔ 그래프 종류

- 사이클 그래프(Cyclic Graph): 정점 간의 경로에서 시작 정점으로 다시 돌아오는 경로가 존재하는 그래프
- 비순환 그래프(Acyclic Graph): 사이클이 없는 그래프로, 일반적으로 방향성이 있는 그래프에서 사용
- 연결 그래프(Connected Graph): 모든 정점 간에 경로가 존재하는 그래프
- 비연결 그래프(Disconnected Graph): 정점 간에 연결되지 않은 부분 그래프가 존재하는 그래프