**9장**

Hashing : Searching algorithm의 시간복잡도가 O(1)이 되도록하는 자료구조

Core idea : 데이터 x는 x번째 index에 저장

Q1 : 데이터 x가 string이면?

Q2 : Wasted memory 해시 테이블을 구현할 때, 전체 테이블 크기와 데이터가 실제로 차지하는 공간 간의 비효율성 때문에 메모리가 낭비될 수 있습니다.

* 데이터로부터 index를 hash function으로 유도/계산하여 index를 구하자(index = h(data))
* Hash fucntion의 조건 : 낮은 시간 복잡도, 고른 데이터 분포

Hash function

1. Division method : index = h(x) = x mod size\_of\_array = x를 어레이의 크기로 나눈 나머지

* Insertion, deletion, searching이 모두 O(1)
* Index가 충돌하는 경우 발생, -> multiplication method

1. Multiplication method :

텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

자동 생성된 설명

1. Load factor :

로드 팩터(Load Factor)는 해시 테이블의 효율성을 측정하는 지표로, 해시 테이블이 얼마나 채워져 있는지를 나타냅니다. 이는 해시 테이블의 성능과 관련이 있으며, 충돌과 메모리 사용 효율성에 영향을 미칩니다.

**로드 팩터의 정의**

로드 팩터는 다음과 같이 정의됩니다:

Load Factor=n/m

여기서,

* nnn은 해시 테이블에 저장된 요소의 개수,
* mmm은 해시 테이블의 버킷(bucket) 또는 슬롯(slot)의 총 개수입니다.

로드 팩터는 0에서 1 사이의 값이며, 1에 가까울수록 해시 테이블이 꽉 차 있음을 의미합니다.

Q3: Duplicated index = hash collision

1. open hashing : chaining : 동일한 해시 값을 가진 여러 요소가 있을 때, 각 요소를 해시 테이블의 각 슬롯(slot)의 연결 리스트에 추가하여 충돌을 해결합니다. Single linked llist 의 헤더는 insertion operation에 따라 업데이트 된다.

Load factor가 1보다 커질 수 있다.

1. Closed hashing : 충돌이 발생할 때 해시 테이블 내의 다른 위치를 찾아 데이터를 저장

* Linear hashing : 충돌 발생 시, 다음 index을 확인하여 빈 index을 찾을 때까지 이동.
* Quadratic probibng : 충돌 발생 시, 제곱 간격으로 이동하여 빈 버킷을 찾음 🡺 신속하게 1차 군집을 탈출할 수 있음
* Double hashing : 이중 해싱은 두 개의 다른 해시 함수를 사용하여 충돌이 발생할 때 두 번째 해시 함수의 값을 간격으로 사용해 다음 위치를 탐색합니다.

10장