<해시(Hash)>

데이터를 효율적으로 관리하기 위해, 임의의 길이 데이터를 고정된 길이의 데이터로 매핑하는 것

해시 함수를 구현하여 데이터 값을 해시 값으로 매핑한다.

```
Lee → 해싱함수 → 5
Kim → 해싱함수 → 3
Park → 해싱함수 → 2
...
Chun → 해싱함수 → 5 // Lee와 해싱값 충돌
```

결국 데이터가 많아지면, 다른 데이터가 같은 해시 값으로 충돌나는 현상이 발생함 **'collision' 현상**

1. 그래도 해시 테이블을 쓰는 이유는?

적은 자원으로 많은 데이터를 효율적으로 관리하기 위해

하드디스크나, 클라우드에 존재하는 무한한 데이터들을 유한한 개수의 해시값으로 매핑하면 작은 메모리로도 프로세스 관리가 가능해짐!

- 언제나 동일한 해시값 리턴, index 를 알면 빠른 데이터 검색이 가능해짐
- 해시테이블의 시간복잡도 O(1) - (이진탐색트리는 O(logN))

2. 충돌 문제 해결

1) **체이닝** : 연결리스트로 노드를 계속 추가해나가는 방식 (제한 없이 계속 연결 가능, but 메모리 문제)

2) **Open Addressing** : 해시 함수로 얻은 주소가 아닌 다른 주소에 데이터를 저장할 수 있도록 허용 (해당 키 값에 저장되어있으면 다음 주소에 저장)

3) **선형 탐사** : 정해진 고정 폭으로 옮겨 해시값의 중복을 피함

4) **제곱 탐사** : 정해진 고정 폭을 제곱수로 옮겨 해시값의 중복을 피함

<Reference>

1. https://ratsgo.github.io/data%20structure&algorithm/2017/10/25/hash/
2. https://github.com/gyoogle/tech-interview-for-developer/blob/master/Computer%20Science/Data%20Structure/Hash.md