1. 해시

해시란 암호학에서 사용하는 방법이다. 블록체인 장부에 사용한다고 요즘 말이 많다.

해시란 임의의 길이의 데이터를 일정한 길이의 데이터로 바꾸는 것이다. 데이터가 살짝 달라져도 해시값은 완전히 변한다고 할 수 있다.

데이터(키) --------------(해시 알고리즘)-----------------------🡪 해시값

이런 식으로 진행이 된다.

해시 값은 보통 테이블에 저장이 되며 해시값 자체를 인덱스로 사용한다..

이 경우 데이터를 찾을 때 해시값을 바로 찾으니 시간복잡도는 O(1)으로 개빠르다고 할 수 있다.

근데 문제가 만약 키 값은 다른데 해시값이 같게 나온다면 어떻게 될까? 이를 바로 충돌(Collision)이 일어났다고 한다. 이러한 경우에는 배열에 충돌이 일어나게 된다.

해시함수의 핵심은 바로 이 충돌을 최대한 적게 내는거라고 할 수 있다.

2. 해시테이블

해시값들을 저장하는 자료구조로 배열의 인덱스에 해시값을 사용한다

적은 리소스로 많은 데이터를 효율적으로 관리하기 위해 사용하며,

시간복잡도는 바로 직으로 꽂으니 O(1)이 나온다. 빠르다.

하지만 충돌이 일어나면 저장하는데 곤란함이 있다. 이를 해결하기 위한 방법이 Chaining과 오픈 어드레싱이라는 방법이 있다.

3. 체이닝

체이닝은 해시테이블을 Linked List연결하는 것이다. 그렇게 하면 중복값이 일어나도 연결리스트로 엮으니 해시값이 겹쳐도 다른 곳으로 이동할 필요가 없다.

하지만 링크드 리스트를 구현해야 해서 노드와 포인터를 설정해준다는 점에서 리소스가 많이 소모된다는 점과 만약 해시테이블의 하나의 인덱스에 모든 해시 값이 들어간다면 시간복잡도는

O(1)에서 O(N)이 된다. 그러니 안겹치게 함수를 짜도록 하자

이를 해결하는 방법 중 하나 자바의 해시맵인데 아직 공부 안했다.

4. 오픈어드레싱

오픈 어드레싱은 겹치는 해시값이 있어 인덱스가 차있으면 다음 인덱스로 옮기는 방법이다.

크게 리니어 프로빙, 쿼드라 프로빙, 더블해시가 있다.

리니어 프로빙은 인덱스가 차있으면 다음 인덱스로 쳐박는 방법이다. 근데 가면 갈수록 인덱스가 차버리면 결국에는 못들어가 오버헤드가 발동되며 +1을 한다는 점에서 한 장소에 집중적으로 위치하는 현상인 프라이머리 클러스터링이 발생하게 된다. 이를 해결하는 방법이 쿼드라 프로빙이다.

쿼드라 프로빙은 인덱스가 차있으면 충돌횟수에 따라 수식을 정해 정하고 거기에 쳐박는 방법이다. 이또한 하다보면 오버헤드가 발동되며 특정수식은 가면 갈수록 역여서 한 군대에 뭉티기 보다는 주기적으로 뭉치게 된다 이러한 현상은 세컨더리 클러스터링이라고 한다.