Day 11 - Hash

Hash(Java, Python, C++, C)

1. **배열**
    a. 기본적으로 **동일한 타입**의 데이터(Java, C++, C)를 연속적으로 가질수 있는 구조
        i. Python에서는 서로 다른 타입의 종류도 연속적으로 가질 수 있다.
    b. ex) 3이라는 숫자가 몇번째에 있는지 찾아라
    [1,2,3,4,5 …..]
        1. 찾고자하는 데이터의 탐색
        2. 위치를 알 수 있다.
        3. 하지만 데이터를 찾는데 index를 사용하지 않고 데이터를 탐색해야한다면?

    - 배열만 가지고는 특정한 문제를 해결할 수 없다.
    - 데이터에 더 많은 의미를 부여하여 특정 위치에 존재하는 데이터를 빠르게 찾는 방법을 효율적인 방식으로 대응 해야 한다.
        - 이러한 대응 방식중에 가장 많이 이용하는 것이 “Hash Table”

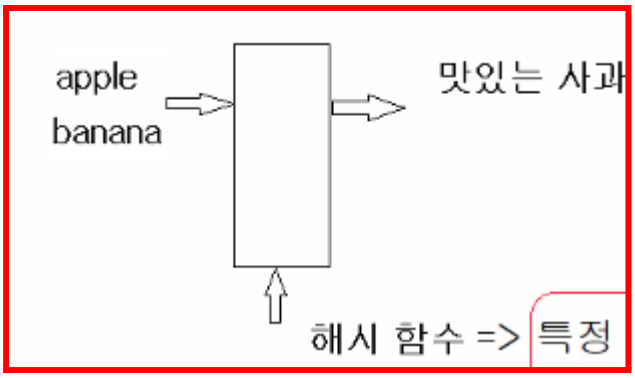
해시

- 특정한 데이터를 변환 함수를 통하여 고정된 크기의 데이터 값으로 변환

특정 데이터를 찾아보자

1. Java
2. C++
3. Python
4. Programming

- 특정 값을 입력 받으면 입력한 값의 길이에 상관없이 항상 일정한 결과값을 만들어 낸다.
- 즉 결과를 알고 싶다면 값만 알고 있으면 된다는 의미



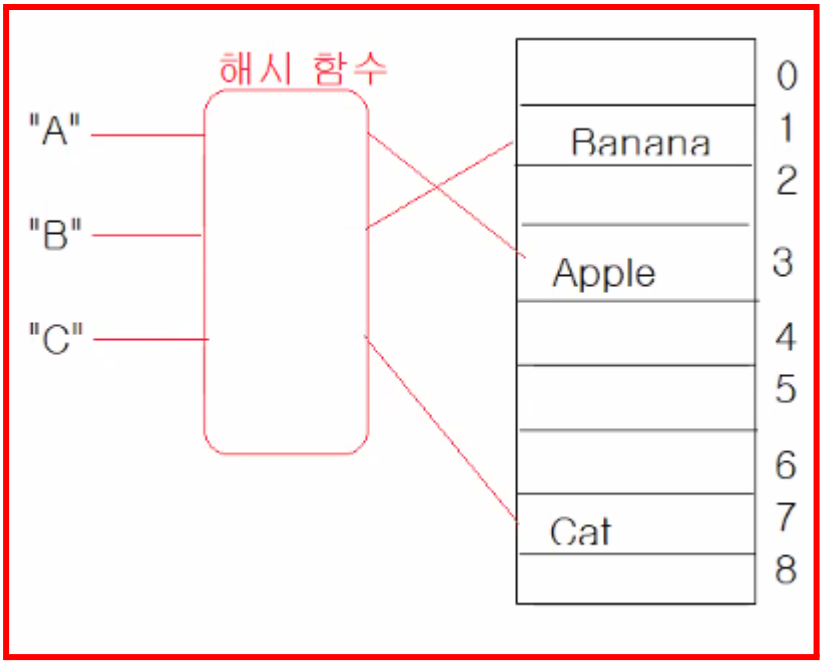
해시 함수 (Hash Function)

- 특정 단어나 데이터를 받아서 해시 함수를 사용하여 변환하는 작업을 “Hashing” 이라고 하고 그 결과값은 고정된 길이의 새로운 값이 만들어 진다.

해시 테이블 (Hash Table)

- 이러한 해시 함수, 해시의 작동 원리를 사용하여 만든 자료형이 해시 테이블 (Hash table) 이다.
- 파이썬에서는 “Dictionary” { key:value }
- 해싱을 사용하여 데이터를 테이블 형태로 저장하는 구조 ⇒ Hash table
- 선언한 키(key)는 해시 함수를 통해 특정한 값으로 변환되어 내부에서 관리한다. 그에 대응하는 데이터가 연결되는 자료구조를 의미한다.

ex) Hash Table



- 해시 함수를 통해서 바로 값을 찾아준다.

*** 딕셔너리 key : value**

{key : value}

info = {“name”:”lee”, “age : 20}

값을 호출 할 때에는 info[‘name’]

- 배열처럼 키를 호출한다고 생각하면 된다.

코딩테스트에서는

- 문자열과 숫자만을 사용하여 문제를 푸는데 아무런 제약이 없다.

예시문제)

- 사과 4개, 바나나 11개, 귤 7개 ⇒ 이 내용을 어떻게 데이터로 구성 가능할까?
- 위의 데이터를 배열로 구현 ⇒ 각각의 과일의 개수를 배열로 만든다.
    - 각 과일의 개수를 배열로 구현한다면 순서대로 각 종류의 과일을 나열한다. 이러한 방법은 너무 비효율적이다.

- **파이썬으로 구현하면**

  fruit = {‘Apple’ : 4, ‘Banana’ : 2, ‘mandarin’ : 7 }

  fruit[‘Apple’] ⇒ 4

    - **for 구문을 이용하여 조회하는 코드를 작성**

```
for i in fruit:
    # 코드작성
    print([fruit])
```

    - **위의 형식을 배열로 선언**

      [‘Apple’, ‘Banana’, ‘mandarin’]

      [4, 11, 7]

2개의 배열이든 1개의 배열이든 순서를 지켜야한다. (순서가 다르면 값이 달라진다)

이러한 불편한 점을 **“해시 테이블”**을 사용하여 개선할 수 있다.

1. **탐색에 걸리는 시간 비용이 크게 줄어든다.**
    - (주의할 점은 **정렬을 자주한다거나 혹은 전체 탐색**을 한다거나, **‘최대값 / 최소값 찾기’** 처럼 자료구조를 많이 이용하는 연산일수록 해시 테이블의 효율이 크게 떨어진다.)

해시 예시문제):

Untitled0.ipynb