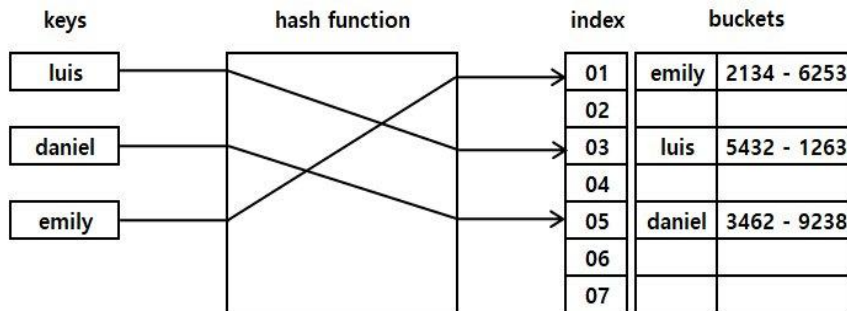


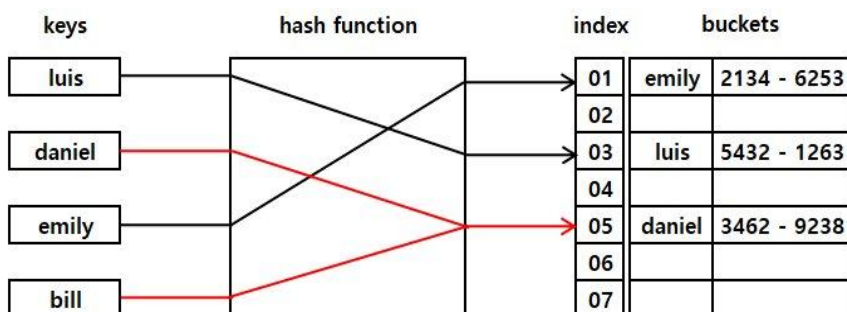
해시테이블

해시함수를 사용하여 키를 해시값으로 매핑하고, 이 해시값을 색인(index) 삼아 데이터의 값(value)을 키와 함께 저장하는 자료구조를 해시테이블(hash table)이라고 합니다. 이 때 데이터가 저장되는 곳을 버킷(bucket) 또는 슬롯(slot)이라고 합니다.



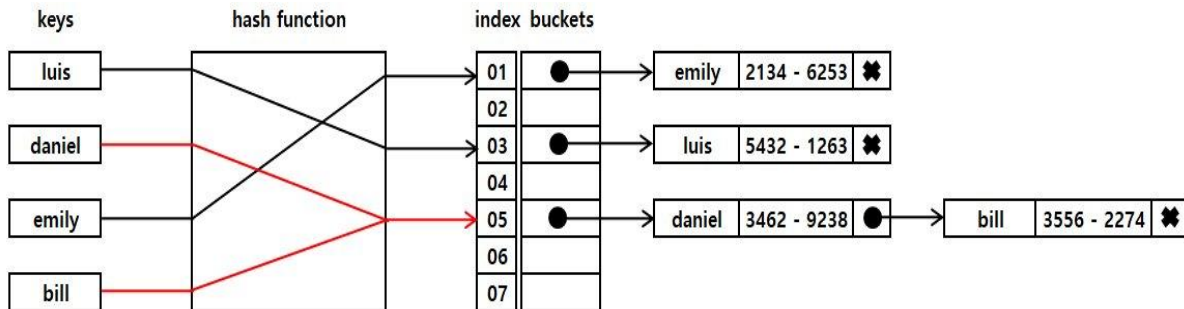
충돌

해시함수는 해시값의 개수보다 대개 많은 키값을 해시값으로 변환하기 때문에 해시함수가 서로 다른 두 개의 키에 대해 동일한 해시값을 내는 해시충돌(collision)이 발생하게 됩니다. 아래 그림은 이름-전화번호를 매핑하기 위한 해시함수를 개념적으로 나타냈습니다. 예시의 해시함수는 'daniel'과 'bill'를 모두 '05'로 매핑해 해시충돌을 일으키고 있습니다.



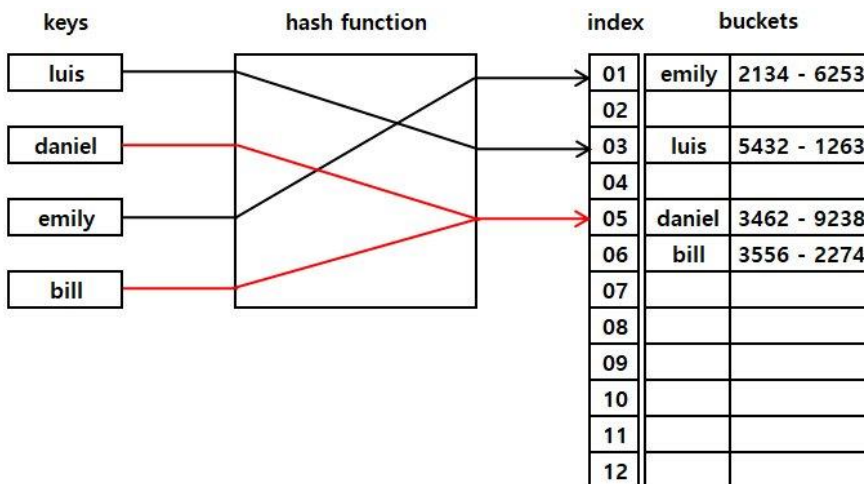
충돌해결(체인법 : Separate Chaining)

해시충돌 문제를 해결하기 위한 간단한 아이디어 가운데 하나는 한 버킷당 들어갈 수 있는 엔트리의 수에 제한을 두지 않음으로써 모든 자료를 해시테이블에 담는 것입니다. 해당 버킷에 데이터가 이미 있다면 체인처럼 노드를 추가하여 다음 노드를 가리키는 방식으로 구현(연결리스트)하기 때문에 체인이라고 합니다.



충돌해결(개방번지화 : Open Addressing)

open addressing은 chaining과 달리 한 버킷당 들어갈 수 있는 엔트리가 하나뿐인 해시테이블입니다. 해시함수로 얻은 주소가 아닌, 다른 주소에 데이터를 저장할 수 있도록 허용한다는 취지에서 open addressing이라고 합니다. 해시충돌이 빈번히 생길 수 있습니다.



1) 선형탐사(Linear probing)

최초 해시값에 해당하는 버킷에 다른 데이터가 저장돼 있으면 해당 해시값에서 고정 폭(예컨대 1칸)을 옮겨 다음 해시값에 해당하는 버킷에 액세스(삽입, 삭제, 탐색)합니다. 여기에 데이터가 있으면 고정폭으로 또 옮겨 액세스합니다.

2) 제곱탐사(Quadratic probing)

최초 해시값에 해당하는 버킷에 다른 데이터가 저장돼 있으면 해당 해시값에서 그 폭이 제곱수로 늘어난다는 특징이 있습니다. 예컨대 임의의 키값에 해당하는 데이터에 액세스할 때 충돌이 일어나면 1^2 칸을 옮깁니다. 여기에서도 충돌이 일어나면 이번엔 2^2 칸, 그 다음엔 3^2 칸 옮기는 식입니다.

3) 이중해싱(Double hashing)

탐사할 해시값의 규칙성을 없애버려서 clustering을 방지하는 기법입니다. 2개의 해시함수를 준비해서 하나는 최초의 해시값을 얻을 때, 또 다른 하나는 해시충돌이 일어났을 때 탐사 이동폭을 얻기 위해 사용합니다.

해시테이블의 시간복잡도

"해시테이블의 탐색, 삽입, 삭제의 시간복잡도는 평균적으로 $O(1)$ 을 갖지만 충돌이 빈번하게 일어나면 최악의 경우인 $O(n)$ 의 시간복잡도로 수렴할 수 있습니다."

HashMap

메서드	설 명
void clear()	HashMap에 저장된 모든 객체를 제거
boolean containsKey(Object key)	HashMap에 저장된 키(key)가 포함되었는지 알려준다. (포함되어 있으면 true)
Object get(Object key)	지정된 키(key의 값(객체))을 반환, 못찾으면 null 반환
Object getOrDefault(Object key, Object defaultValue)	지정된 키(key)의 값(객체)을 반환한다. 키를 못찾으면, 기본 값(defaultValue)로 지정된 객체를 반환
boolean isEmpty()	HashMap이 비어있는지 알려준다.
Set keySet()	HashMap에 저장된 모든 키가 저장된 Set을 반환
Object put(Object key, Object value)	지정된 키와 값을 HashMap에 저장
Object remove(Object key)	HashMap에 지정된 키로 저장된 값(객체)를 제거
int size()	저장된 객체의 개수를 반환한다.

HashMap 사용예:

```

class Solution {
    public void solution(String s){
        HashMap<Character, Integer> sH = new HashMap<>();
        for(char x : s.toCharArray()){
            sH.put(x, sH.getOrDefault(x, 0) + 1);
        }
        for(char key : sH.keySet()){
            System.out.println(key + " " + sH.get(key));
        }
    }
    public static void main(String[] args){
        Solution T = new Solution();
        T.solution("statitsics");
    }
}

```

학급 회장

학급 회장을 뽑는데 후보로 기호 A, B, C, D, E 후보가 등록을 했습니다.

투표용지에는 반 학생들이 자기가 선택한 후보의 기호(알파벳)가 쓰여져 있으며 선생님은 그 기호를 발표하고 있습니다.

매개변수 s에 투표용지에 쓰여져 있던 각 후보의 기호가 선생님이 발표한 순서대로 문자열로 주어지면 어떤 기호의 후보가 학급 회장이 되었는지 반환하는 프로그램을 작성하세요.

반드시 한 명의 학급회장이 선출되도록 투표결과가 나왔다고 가정합니다.

입출력 예:

s	answer
"BACBACCACCBDEDE"	C
"AAAAABBCCDDDD"	A
"AABBCCDDEEABCB"	B

제한사항:

- 문자열 s의 길이는 100을 넘지 않습니다.

입력예제 1 설명 :

A기호 3표, B기호 3표, C기호 5표, D기호 2표, E기호 2표 를 받아 C가 학급회장이 되었습니다.

한 번 사용한 최초문자

문자열에서 한번만 사용한 문자를 찾으려고 합니다.

매개변수 `s`에 문자열이 주어지면 한번만 사용한 문자 중 문자열에서 가장 먼저 나타난 문자의 인덱스 번호를 반환하는 프로그램을 작성하세요. 인덱스는 1부터 시작합니다. 한번만 사용한 문자가 없을 경우 `-1`를 반환하세요.

입출력 예:

s	answer
"statitsics"	3
"aabb"	-1
"stringshowtime"	3
"abcdeabcfg"	5

제한사항:

- 문자열 `s`의 길이는 100을 넘지 않습니다.
- 문자열은 소문자로만 이루어져 있습니다.

입력예제 1 설명 :

한번만 사용한 문자는 `a`, `c`이고, 문자열에서 먼저 나타난 것은 `a`이고 인덱스는 3입니다.

같은 빈도수 만들기

소문자 a, b, c, d, e로 이루어진 문자열이 주어지면 해당 문자열에서 a, b, c, d, e의 최소의 개수를 추가하여 a, b, c, d, e의 빈도수가 동일하게 되도록 해야 합니다. 동일빈도수가 되는 최소 추가 개수를 알파벳 a, b, c, d, e순으로 배열에 저장하여 반환하는 프로그램을 작성하세요.

만약 주어진 문자열이 "aaabc" 라면 빈도수는 a:3 , b:1, c:1, d:0, e:0 이고 최소 개수를 추가하여 동일 빈도수가 되게 하려면 b를 2개, c를 2개, d를 3개, e를 3개 추가하면 모두 빈도수가 3개로 동일해집니다.

입출력 예:

s	answer
"aaabc"	[0, 2, 2, 3, 3]
"aabb"	[0, 0, 2, 2, 2]
"abcde"	[0, 0, 0, 0, 0]
"abcdeabc"	[0, 0, 0, 1, 1]
"abbccdde"	[1, 0, 0, 0, 0]

제한사항:

- 문자열 s의 길이는 100을 넘지 않습니다.

자기 분열수

자기 분열수란 배열의 원소 중 자기 자신의 숫자만큼 빈도수를 갖는 숫자를 의미합니다.

만약 배열이 [1, 2, 3, 1, 3, 3, 2, 4] 라면

1의 빈도수는 2,

2의 빈도수는 2,

3의 빈도수는 3,

4의 빈도수는 1입니다.

여기서 자기 자신의 숫자와 같은 빈도수를 갖는 자기 분열수는 2와 3입니다.

매개변수 nums에 자연수가 원소인 배열이 주어지면 이 배열에서 자기 분열수 중 가장 작은 수를 찾아 반환하는 프로그램을 작성하세요. 자기 분열수가 존재하지 않으면 -1을 반환하세요.

입출력 예:

nums	answer
[1, 2, 3, 1, 3, 3, 2, 4]	2
[1, 2, 3, 3, 3, 2, 4, 5, 5, 5]	1
[1, 1, 2, 5, 5, 5, 5, 5, 3, 3, 3, 3, 5]	-1
[7, 6, 7, 7, 8, 8, 8, 8, 7, 5, 7, 7, 7, 8, 8]	7
[11, 12, 5, 5, 3, 11, 7, 12, 15, 12, 12, 11, 12, 12, 7, 8, 12, 11, 12, 7, 12, 5, 15, 20, 15, 12, 15, 12, 15, 14, 12]	12

제한사항:

- nums의 길이 $3 \leq n \leq 500,000$
- 배열 nums의 원소는 자연수입니다. $1 \leq \text{nums}[i] \leq 1,000,000$

팰린드롬 길이

문자열이 주어지면 해당 문자열의 문자들을 가지고 만들 수 있는 최대길이 팰린드롬을 만들고 그 길이를 구하세요. 문자열은 소문자로만 이루어져 있습니다.

만약 "abcbbbccaaeee" 가 주어진다면 만들 수 있는 가장 긴 팰린드롬은 "ebbcaaacbbe"이고 답은 11입니다.

입출력 예:

s	answer
"abcbbbccaaeee"	11
"aabbccddee"	10
"fgfgabtetaaaetytceefcecekefefkcccckbsgaafffg"	41
"aabcefagcefbcabbcc"	17
"abcbbbccaa"	9

제한사항:

- s의 길이는 1,000을 넘지 않습니다.

프로그래머스 Lv.0 - 한 번만 등장한 문자

<https://school.programmers.co.kr/learn/courses/30/lessons/120896>

정답코드

```
import java.util.*;
class Solution {
    public String solution(String s) {
        HashMap<Character, Integer> sH = new HashMap<>();
        for(char x : s.toCharArray()){
            sH.put(x, sH.getOrDefault(x, 0) + 1);
        }
        StringBuilder sb = new StringBuilder();
        for(char key : sH.keySet()){
            if(sH.get(key) == 1) sb.append(key);
        }
        String tmp = sb.toString();
        char[] answer = tmp.toCharArray();
        Arrays.sort(answer);
        return String.valueOf(answer);
    }
}
```

프로그래머스 Lv.0 - 최빈값 구하기

<https://school.programmers.co.kr/learn/courses/30/lessons/120812>

정답코드

```
import java.util.*;
class Solution {
    public int solution(int[] array) {
        HashMap<Integer, Integer> nH = new HashMap<>();
        for(int x : array){
            nH.put(x, nH.getOrDefault(x, 0) + 1);
        }
        int max = 0;
        for(int key : nH.keySet()){
            if(nH.get(key) > max) max = nH.get(key);
        }
        ArrayList<Integer> al = new ArrayList<>();
        for(int key : nH.keySet()){
            if(nH.get(key) == max) al.add(key);
        }
        if(al.size() > 1) return -1;
        return al.get(0);
    }
}
```

프로그래머스 Lv.1 - 추억 점수

<https://school.programmers.co.kr/learn/courses/30/lessons/176963>

정답코드

```
import java.util.*;

class Solution {
    public int[] solution(String[] name, int[] yearning, String[][] photo) {
        int[] answer = new int[photo.length];
        HashMap<String, Integer> sH = new HashMap<>();
        for(int i = 0; i < name.length; i++){
            sH.put(name[i], yearning[i]);
        }
        for(int i = 0; i < photo.length; i++){
            int score = 0;
            for(String x : photo[i]){
                score += sH.getOrDefault(x, 0);
            }
            answer[i] = score;
        }
        return answer;
    }
}
```

프로그래머스 Lv.1 - 신고 결과 받기

<https://school.programmers.co.kr/learn/courses/30/lessons/92334>

정답코드

```
import java.util.*;

class Solution {
    public int[] solution(String[] id_list, String[] report, int k){
        int[] answer = new int[id_list.length];
        HashMap<String, HashSet<String>> reportHash = new HashMap<>();
        HashMap<String, Integer> stoped = new HashMap<>();
        HashSet<String> reportSet = new HashSet<>(Arrays.asList(report));
        for(String x : reportSet){
            String a = x.split(" ")[0];
            String b = x.split(" ")[1];
            reportHash.putIfAbsent(a, new HashSet<String>());
            reportHash.get(a).add(b);
            stoped.put(b, stoped.getOrDefault(b, 0) + 1);
        }
        for(int i = 0; i < id_list.length; i++){
            int cnt = 0;
            if(reportHash.get(id_list[i]) == null) continue;
            for(String user : reportHash.get(id_list[i])){
                if(stoped.get(user) >= k) cnt++;
            }
            answer[i] = cnt;
        }
        return answer;
    }
}
```