别人家的康少

第五课-哈希表、映射、集合的实现与特性

hash table

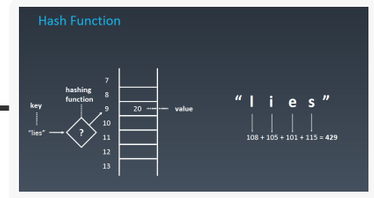
def

- 哈希表（Hash Hash table），也叫散列表，是根据关键码值（Key value）而直接进行访问的数据结构。
- 它通过把关键码值映射到表中一个位置来访问记录，以加快查找的速度。
- 这个映射函数叫作散列（Hash Hash Function Function），存放记录的数组叫 作哈希表（或散列

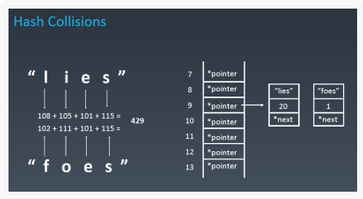
工程实践

- 电话号码簿
- 用户信息表
- 缓存（LRU Cache Cache）
- 键值对存储（Redis Redis）

hash function



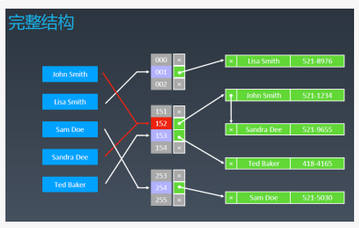
hash collisions



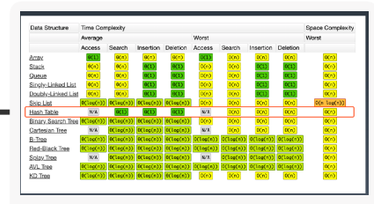
- 对于不同的东西，通过hash function得到的结果是一样的
  - eg:
    - lies
    - foes

    429
- 于是加了一个链表
- O(1)查询
  - 但是如果linked list进去了以后，且设计较差 — 会达到0(n)
  - 设计好的话可以有效避免冲突 O(1)

完整结构



复杂度分析



Data Structure	Time Complexity								Space Complexity
	Average				Worst				Worst
	Access	Search	Insertion	Deletion	Access	Search	Insertion	Deletion	
Array	O(1)	O(n)	O(n)	O(n)	O(1)	O(n)	O(n)	O(n)	O(n)
Stack	O(n)	O(n)	O(1)	O(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Queue	O(n)	O(n)	O(1)	O(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Singly-Linked List	O(n)	O(n)	O(1)	O(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Doubly-Linked List	O(n)	O(n)	O(1)	O(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Skip List	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)	O(n)	O(n)	O(n)	O(n log(n))
Hash Table	N/A	O(1)	O(1)	O(1)	N/A	O(n)	O(n)	O(n)	O(n)
Binary Search Tree	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)	O(n)	O(n)	O(n)	O(n)
Cartesian Tree	N/A	O(log(n))	O(log(n))	O(log(n))	N/A	O(n)	O(n)	O(n)	O(n)
B-Tree	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)
Red-Black Tree	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)
Splay Tree	N/A	O(log(n))	O(log(n))	O(log(n))	N/A	O(log(n))	O(log(n))	O(log(n))	O(n)
AVL Tree	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)
KD Tree	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)	O(n)	O(n)	O(n)	O(n)

JAVA Code

做题四部

- four steps:
- //1. clarification
- //2. possible soltions --> optimal (time & space)
- //3. code
- //4. test cases

Exercise

242. 有效的字母异位词

- 暴力

```
class Solution {
    public boolean isAnagram(String s, String t) {
        //anagram means that the numbers of each elements are the same
        // n + 1 time a=3 times g =1 m =1 r=1

        //four steps:
        //1. clarification
        //2. possible soltions --> optimal (time & space)
        //3. code
        //4. test cases

        // method 1:暴力求解
        //sort : 然后比较各元素相等
        if(s.length() != t.length()) return false;
        char str1[] = s.toCharArray();
        char str2[] = t.toCharArray();
        Arrays.sort(str1);
        Arrays.sort(str2);
        return Arrays.equals(str1,str2);


    }
}
```

242. 有效的字母异位词

  - string->charArray
  - sort
  - 比较长度
- hash