Lab 12
Hash

2019. 05. 30

lab 12. Open Addressing (closed hashing)

In this assignment, you are requested to implement insert, delete, find and print operations for an open-addressing(closed hashing) hash table with three collision solution.

1. Linear Probing
2. Quadratic Probing
3. Double Hashing

Simple Specification

a. Create an empty hash table of size m. *(m is specified in the input.txt)*
b. Each integer of the input will be a value that you should insert into the hash table.
c. Using Simple Mod Hash. (value mod m)
d. Empty node's value must be 0 (zero).
e. You have to use dynamic allocation when you make hash table.

lab 12. Open Addressing (closed hashing)

- **Data Structure Specification**

```
typedef struct ListNode *Position;
typedef Position List;
typedef HashTbl *HashTable;

struct ListNode{
    int Element;
}
struct HashTbl{
    int TableSize;
    List* TheLists;
}
```

- **Function specification**
  - void Insert(HashTable H, int value, int solution)
  - void delete(HashTable H, int value, int solution)
  - int find(HashTable H, int value, int solution)
  - void print(HashTable H)
  - int Hash(int value, int Size, int i, int solution);
    - h(value) : value mod Size
    - h(value) + i → (linear)
    - h(value) + i * i → (quadratic)
    - (h(value)+i*(7-(value mod 7))) mod Size → (Double)

e.g Solution number

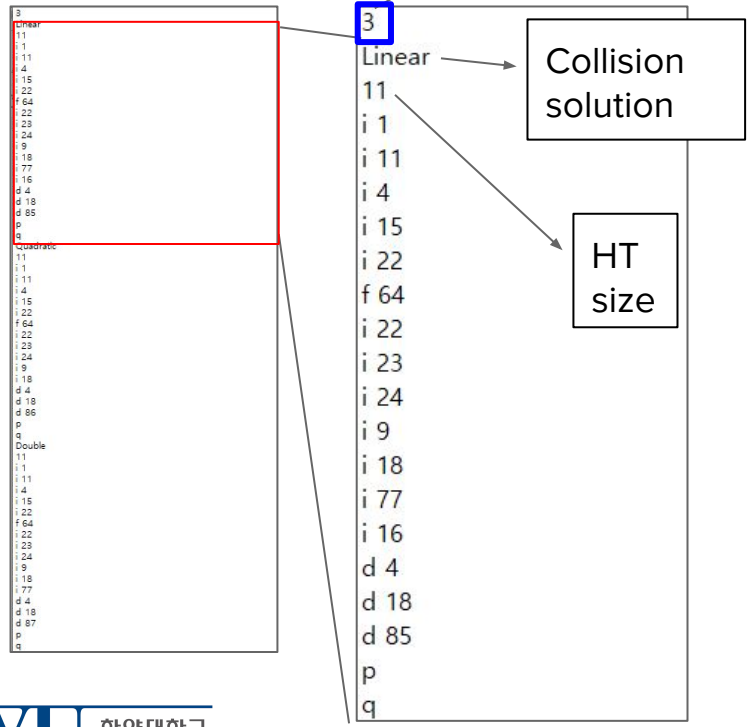
linear : 1
quadratic : 2
Double : 3

lab 12. Open Addressing (closed hashing)

- **Input Specification**
  - First line represents size of test cases.
  - First line of each test cases means collision solution. (Linear, Quadratic, Double)
  - Second line of each test cases means hash table size.
  - Remaining line of each test cases means instruction.
    - **i x** : insert a value “x” to the hash table and print “Inserted x”. If “x” already exists in hash table, print “Already exists”. value x is always positive integer. (0 can not be the key.)
    - **d x** : delete a value “x” in the hash table and set the value 0. After, print “Deleted x”. If the “x” doesn’t exist in the hash table, print “x not exists”.
    - **f x** : find a value “x” in the hash table. Print the hash table index if the “x” exist, else print “not found”.
    - **p** : print the hash table from index 0 to (m – 1). Print all values at index in order. If there is no value at any index, print “0”.
    - **q** : finish test case.
- **You have to use file I/O like the previous assignment.**

lab 12. Open Addressing (closed hashing)

Input



Output

```
Linear
Inserted 1
Inserted 11
Inserted 4
Inserted 15
Inserted 22
Not found
Already exists
Inserted 23
Inserted 24
Inserted 9
Inserted 18
Inserted 77
Inserted 16
Deleted 4
Deleted 18
85 not exists
11 1 22 23 0 15 24 0 77 9 16

Quadratic
Inserted 1
Inserted 11
Inserted 4
Inserted 15
Inserted 22
Not found
Already exists
Inserted 23
Inserted 24
Inserted 9
Inserted 18
Deleted 4
Deleted 18
86 not exists
11 1 23 24 0 15 0 0 0 22 9

Double
Inserted 1
Inserted 11
Inserted 4
Inserted 15
Inserted 22
Not found
Already exists
Inserted 23
Inserted 24
Inserted 9
Inserted 18
Inserted 77
Inserted 16
Deleted 4
Deleted 18
87 not exists
11 1 24 77 0 23 22 0 0 9 15
```

Uploaded in http://bislab.hanyang.ac.kr/index.php?mid=DataStructure

lab 12. Open Addressing (closed hashing)

- Submission
  - Project directory name : lab12
  - Source file name : p12.c
  - Executable file name : p12.out
  - You should upload in the hconnect (Gitlab) server.

DeadLine

Wednesday, 12 June, 23 : 59 pm