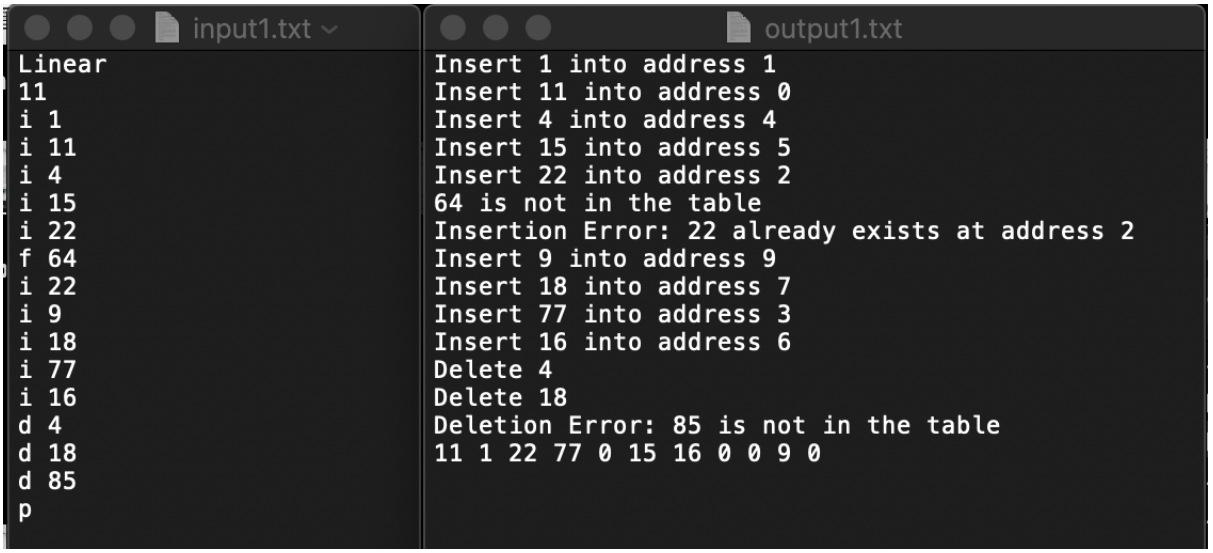
2023 Data Structure (CSE2010 12282)

Lab 11 (5/24) Closed Hashing

- arranged by TA Baomaoo Kim

Due : ~2023.05.31.(Wed) 23:59 **Late Submission : ~2023.06.01.(Thr) 23:59**

- **Closed Hashing**을 구현하고**, insert, delete, find, print** 기능을 구현한다**.**
- 아래의 사진과 같이 **[input].txt** 파일을 입력받아 **Closed Hashing** 기능을 수행하고 결과값을 **[output].txt** 파일에 출력하여 저장한다**.**
- **Implement Closed Hashing, insert, delete, find, print function.**
- **As shown in the picture below, it receives the [input].txt file and performs the closed hashing function. Print the output to the [output].txt file and save it.**



input1.txt

```
Linear
11
i 1
i 11
i 4
i 15
i 22
f 64
i 22
i 9
i 18
i 77
i 16
d 4
d 18
d 85
p
```

output1.txt

```
Insert 1 into address 1
Insert 11 into address 0
Insert 4 into address 4
Insert 15 into address 5
Insert 22 into address 2
64 is not in the table
Insertion Error: 22 already exists at address 2
Insert 9 into address 9
Insert 18 into address 7
Insert 77 into address 3
Insert 16 into address 6
Delete 4
Delete 18
Deletion Error: 85 is not in the table
11 1 22 77 0 15 16 0 0 9 0
```

- **<input> :** 각 **line** 마다 **Command** 가 주어짐**.**
  : Collision Solution type, size, insert, find, print
- **<output> :** 각 **Command** 에 맞는 **result** 출력**.**
- **<Input> : Command is given for each line**
  : Collision Solution type, size, insert, find, print
- **<Output> : Output result for each command.**

<collision Solution type 및 **Hash Table size** 설정**>**

- 위 사진에서처럼 첫째줄에 **“Linear”** 혹은 **“Quadratic”**로 입력됩니다.
- 위 사진에서처럼 둘째줄에 **100** 이하의 자연수로 입력됩니다.
- As in the picture above, “Linear” or “Quadratic” is entered in the first line.
- As in the picture above, a natural number less than 100 is entered in the second line.

<Insert>

- solution을 이용해서 Hash table에 값을 insert합니다.
- 위의 사진에서처럼 **“i [insert할 key값]"** 로 표현합니다.
- insertion 성공시 : **"Insert key값 into address entry인덱스"**
  중복된 key값 insert시 : **"Insertion Error: key값 already exists at address entry인덱스"**
  HashTable이 다 차있는 경우 : **“Insertion Error: table is full”**
- As in the picture above, it is expressed as “i **[key value to insert]**”
- Successful insertion: "Insert key value into address **entry index**"
- When inserting duplicate key value: "Insertion Error: **key value** already exists at address **entry index**"
- When HashTable is full: **“Insertion Error: table is full”**

<Find>

- 위의 사진처럼 **"f [search할 key값]"** 로 표현합니다.
- key 값을 찾았을 경우 출력 형식 : **"key값 is in the table"**
- key 값을 찾았을 경우 1을 반환합니다.
- key 값이 존재하지 않는 경우 출력 형식 : **"key값 is not in the table"**
- key 값이 존재하지 않는 경우 0을 반환합니다.
- As in the picture above, it is expressed as "f **[key value to search]**".
- Output format when key value is found: "**key value** is in the table"
- return 1, when key is found.
- Output format when key value does not exist: "**key value** is not in the table"
- return 0, when key is not found.

<Delete>

- 위의 사진처럼 **"d [delete할 key값]"** 로 표현합니다.
- delete 성공시 출력 형식은 **"Delete key값"** 입니다.
- key 값이 존재하지 않는 경우 **"Deletion Error: key값 is not in the table"** 입니다.
- As in the picture above, it is expressed as "f **[key value to search]**".
- Output format when key value is found: "**key value** is in the table"
- Output format when key value does not exist: "**key value** is not in the table"

<PrintTable>

- 위 사진처럼 **"p"** 로 표현 합니다.
- "p"의 경우, Hash Table의 index 순서대로 Hash Table 안에 있는 Key 값들을 출력하시면 됩니다.
- 비어있는 Hash Table Entry의 경우 위 사진처럼 0으로 출력하시면 됩니다.
- It is expressed as "**p**" as shown in the picture above.
- In the case of "p", print out the key values in the Hash Table in the order of index in the Hash Table.
- In the case of an empty Hash Table Entry, you can output it as 0 as shown in the picture above.

<CreateTable>

- table의 size를 입력으로 받아 Hash table을 생성합니다.
  - table의 size는 input파일의 두번째 줄에서 입력받습니다.
- Receives the size of the table and creates a hash table.
  - table size is entered in the second line of the input file.

<Hash Function & Collision Solution>

- **h(value) = value % Size , i = iteration #**
- **hi = (h(value) + i) % Size** (for linear) , **hi = (h(value) + i*i) % Size** (for Quadratic)

<Structure & Function Format>

Structure

```
typedef int ElementType;
typedef ElementType List;
typedef struct HashTbl* HashTable;
typedef struct HashTbl{
    int TableSize;
    List *TheLists;
}HashTbl;
```

Function

```
HashTable createTable(int TableSize);
void Insert(HashTable H, ElementType Key, int solution);
void Delete(HashTable H, ElementType Key, int solution);
int Find(HashTable H, ElementType Key, int solution);
void printTable(HashTable H);
void deleteTable(HashTable H);
```

- 위 사진과 같은 Struct 구조체를 사용하셔야 합니다.
- 위 사진과 같은 함수들을 형식에 맞게 구현해주시면 됩니다.
- Struct format above should be used for implementation
- Functions should be implemented in appropriate format as above.

<File Name Format>

- [학번].c ex) 20XXXXXXXX.c

<실행 방법>

- gcc 2020XXXXXX.c -o 2020XXXXXX_lab11
- ./2020XXXXXX_lab11 [input_file_name] [output_file_name]
- 꼭 제공되는 testcase로 실행시켜보시기 바랍니다.
- !!! Run your solution code with the provided test case above and check whether it works properly !!!

<Issue>

- 코드 작성시 주석을 적어주시기 바랍니다. 주석이 없는경우 Cheating으로 간주될 수 있습니다.
- 제공된 testcase는 채점 case에 포함됩니다. 모두 알맞게 나오는지 확인해보시기 바랍니다.
- 파일 입출력은 *argv[]* 를 사용하여 구현해주시기 바랍니다.
- 제출 마감 시간 이전의 가장 최신 버전의 commit을 기준으로 채점할 예정입니다.
- 제출 파일과, 폴더 naming 은 꼭 지정된 형식으로 해주셔야 합니다.
- Please write down the detailed comments when writing the code. If there is no comment, it might be considered cheating.
- Provided test case is included in the test case for grading. Please check to see if it makes a proper result.
- Do not use a fixed file name when inputting and outputting files, but implement it using argv[] as in skeleton code.
- Scoring will be based on the latest version of commit before the deadline.
- The names of the .c file and directory should be named in proper format.

<Directory Format>

- 아래와 같이 git 프로젝트 폴더에 “lab11” 폴더 생성후, "lab11" 폴더 안에 "20XXXXXXXX.c" 파일을 위치시키시면 됩니다.
- After creating the "lab10" directory in the git-project-directory as below, place the "20XXXXXXXXXX.c" file in the "lab10" directory.

>>

2020_CSE2010_20XXXXXXXX/ (GitLab clone 폴더)

---...

---lab09/
-------2020XXXXXX.c
-------...
---lab10/
-------2020XXXXXX.c
-------...
---lab11/
-------2020XXXXXX.c
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> QnA : 2023ds12282@gmail.com