CX1107 Data Structures and Algorithms **2020/21 Semester 2**

Assignment 6: Searching

School of Computer Science and Engineering *Nanyang Technological University*

Q1 Modify the closed addressing hash table in Lab 6 Question 4 to perform insertion and deletion by using a doubly linked list below.

```c
typedef struct _listnode{
    int key;
    struct _listnode *next;
    struct _listnode *pre;
} ListNode;

typedef struct _linkedlist{
   int size;
   ListNode *head;
} HashTableNode;

typedef struct _hashTable{
   int hSize;
   int nSize;
   HashTableNode *Table;
} HashTable;
```

The following utility functions are provided. The details can refer to the given template:

```c
ListNode* HashSearch(HashTable, int);
```

The insertion and deletion function prototypes are given as follows:

```c
int HashInsert(HashTable* Q1, int key);
int HashDelete(HashTable* Q1, int key);
```

Both functions' return value indicates success (1) or failure (0).

Q2 Implement an open addressing hash table by double hashing to perform insertion and deletion. The structure of hash slots is given below and a hash table with 37 hashslots is created in the main function.

```c
enum Marker {EMPTY,USED,DELETED};

typedef struct _slot{
    int key;
    enum Marker indicator;
} HashSlot;
```

The hash functions are provided. The $hash2()$ is the incremental hash function for double hashing.

```c
int hash1(int key);
int hash2(int key);
```

The insertion and deletion function prototypes are given as follows:

```c
int HashInsert(int key, HashSlot hashTable[]);
int HashDelete(int key, HashSlot hashTable[]);
```

Both functions' return value is the number of key comparisons done during insertion and deletion respectively. Inserting a duplicate key will return -1. If the number of key comparisons is more than the table size, it implies that the table is full. For the deletion function, it will return -1 if the deleting key does not exist. The number of key comparisons in deletion cannot be more than table size. Once the key is deleted, you are not allowed to read it. You only can check its marker if it is a deleted slot.

Q3 Coalesced hashing is a combination of closed addressing and linear probing. Each slot is not only storing the key, but also the link (index) to the next slot. The structure of hash slots is given below.

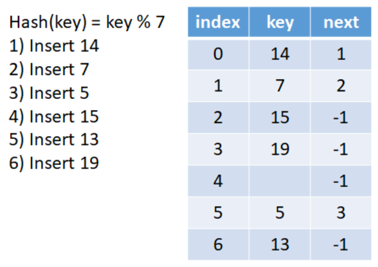
```c
enum Marker {EMPTY,USED};

typedef struct _slot{
    int key;
    enum Marker indicator;
    int next;
} HashSlot;
```

The insertion and searching function prototypes are given as follows:

```c
int HashInsert(int key, HashSlot hashTable[]);
int HashFind(int key, HashSlot hashTable[]);
```

Both functions' return value is an index of the slot where the key is inserted and searched respectively. For insertion function, inserting a duplicate key will return -1. Return value larger than the table size implies that the table is full. For searching function, finding an non-existing key will return -1.

For example,



Hash(key) = key % 7
1) Insert 14
2) Insert 7
3) Insert 5
4) Insert 15
5) Insert 13
6) Insert 19

index	key	next
0	14	1
1	7	2
2	15	-1
3	19	-1
4		-1
5	5	3
6	13	-1