```
1 //HASH TABLE
 2
 3 #include <iostream>
 5 using namespace std;
7 struct Node {
8
       int key;
9
       Node* next;
10 };
11
12 struct HashTable {
13
       Node** head;
14
       int size;
15 };
16
17 HashTable createHashTable(int size = 0) {
       HashTable newTable;
18
19
       newTable.size = size;
20
       newTable.head = new Node * [size];
21
22
       for (int i = 0; i < size; i++) newTable.head[i] = nullptr;</pre>
23
24
       return newTable;
25 }
26
27 int calculateHash(int id) {
28
       return id % 10;
29 }
30
31 void insertTableNode(Node*& head, int key) {
32
       head = new Node{ key, head };
33 }
34
35 void insertItem(HashTable& table, int key) {
       int index = calculateHash(key);
37
       insertTableNode(table.head[index], key);
38 }
39
40 void deleteItem(HashTable& table, int key) {
       int index = calculateHash(key);
42
       Node* prev = nullptr, * cur = table.head[index];
43
       while (cur && cur->key != key) {
44
           prev = cur, cur = cur->next;
45
       }
46
47
       if (!cur) return;
48
       if (!prev) {
49
```

```
... {\tt CMUS \backslash Documents \backslash repos \backslash DSA \backslash Final-LyThuyet \backslash HashTable.cpp}
```

```
table.head[index] = table.head[index]->next;
       }
51
52
       else {
53
            prev->next = cur->next;
54
55
       delete cur;
56 }
57
58 void displayHashTable(HashTable table) {
       for (int i = 0; i < table.size; i++) {</pre>
            cout << i << " -> ";
60
            for (Node* p = table.head[i]; p; p = p->next) {
62
                cout << p->key << " ";
63
            }
64
           cout << endl;</pre>
65
       }
66 }
```

2