```
1  /*//
2  Justin Dang
3  Student ID : 1148267
4  //
5  FUNCTION OF THE FOLLOWING CODE >>
6  Create a hash table class/struct.
7
8  Define an array that holds 27 elements.
9
10 Define a function called Hash(int)
11 -This function returns the modulo of that int by the size of the table (array).
12
13 Define an add function that takes an integer.
14 -This function takes the integer, determines the hash of that number by calling the above hash function, then adds it to the table using linear probing for collision resolution.
15
16 Define a function that looks up a value, it takes an integer, return -1 if the value is not in the table.
17 Create a main that allows the user to add and lookup items in the table.
18
19 //
20 WORKS CITED >>
21  - Professor's provided work
22 *///
23
24 #include <iostream>
25 #include <string>
26 using namespace std;
27
28 const int ARRAY_SIZE = 27;
29 const int EMPTY = -1;
30
31 class HashTable {
32 private:
33     int array[ARRAY_SIZE];                    // Holds elements entered by user.
34 public:
35     HashTable() {
36         for (int x = 0; x < ARRAY_SIZE; x++) {  // Arranges array to work with methods.
37             array[x] = EMPTY;
38         }
39     }
```

```
40
41      int Hash(int data) {
42          return data % ARRAY_SIZE;               // Hashes number ?
43      }
44
45      void Add(int data) {
46          if (array[Hash(data)] == EMPTY) {       // Checks if array index is in
              use.
47              cout << endl << "Inserting " << data << " at " << Hash(data) << endl;
48              array[Hash(data)] = data;           // If not we enter the data into
                  our array.
49              return;
50          }                                       // If the index is taken we
              display that to the user.
51          cout << endl << "Cannot insert " << data << " at " << Hash(data) << endl;
52      }
53
54      int Search(int data) {
55          if (array[Hash(data)] == data)
56              return data;                        // Checks to see if an integer
                  exists within our array.
57          else
58          {
59              cout << data << " does not exist in table." << endl;
60              return EMPTY;                       // Returns -1 if the integer does
                    not exist in our array
61          }
62      }
63  };
64
65  int main()
66  {
67      int userInput;
68      bool hash = true;
69      HashTable* table = new HashTable();
70      while (hash) {
71          cout << endl << "HashTable Commands(Enter command number to execute): "
              << endl
72              << "1) Add(int)" << endl
73              << "2) Search(int)" << endl
74              << "3) Quit" << "\n\n";
75          cin >> userInput;
76          switch (userInput) {
77          case 1:
78              cout << "Enter an integer to add: ";
79              cin >> userInput;
80              table->Add(userInput);
81              break;
82          case 2:
83              cout << "Enter a number to search for in the table: ";
84              cin >> userInput;
85              if (table->Search(userInput) != EMPTY) {
```

```
 86                  cout << endl << userInput << " | Exists within the table";
 87                  break;
 88              }
 89              cout << endl << userInput << " | Does not exists within the table";
 90              break;
 91          case 3:
 92              hash = false;
 93              break;
 94          default:
 95              cout << "\n\nPlease enter a valid command.\n\n";
 96              break;
 97          }
 98      }
 99  }
100  /*//------------------------------------------------ case 1:
101  HashTable Commands(Enter command number to execute):
102  1) Add(int)
103  2) Search(int)
104  3) Quit
105
106  1
107  Enter an integer to add: 1
108
109  Inserting 1 at 1
110
111  HashTable Commands(Enter command number to execute):
112  1) Add(int)
113  2) Search(int)
114  3) Quit
115
116  2
117  Enter a number to search for in the table: 1
118
119  1 | Exists within the table
120  HashTable Commands(Enter command number to execute):
121  1) Add(int)
122  2) Search(int)
123  3) Quit
124
125  1
126  Enter an integer to add: -3214
127
128  Cannot insert -3214 at -1
129
130  HashTable Commands(Enter command number to execute):
131  1) Add(int)
132  2) Search(int)
133  3) Quit
134
135  1
136  Enter an integer to add: 19380247
137
```

```
138  Inserting 19380247 at 25
139
140  HashTable Commands(Enter command number to execute):
141  1) Add(int)
142  2) Search(int)
143  3) Quit
144
145  2
146  Enter a number to search for in the table: 19380247
147
148  19380247 | Exists within the table
149  HashTable Commands(Enter command number to execute):
150  1) Add(int)
151  2) Search(int)
152  3) Quit
153
154  3
155  *///------------------------------------------------
156
```