

```
1 *****
2 *   PROGRAMMED BY : Carlos Aguilera
3 *   CLASS          : CS1B
4 *   SECTION        : MW: 7:30p - 9:50p
5 *   LAB #8         : Lab Linked Lists
6 *****
7
8 Enter name of student to search: John Smith
9
10 FOUND
11 -----
12 Name:   John Smith
13 Age:    20
14 Major:  Math
15 GPA:    3.5
16
17 John Smith
18 -----
19 20
20 Math
21 3.5
22
23 Anna White
24 -----
25 19
26 English
27 3.2
28
29 Paul Johnson
30 -----
31 18
32 Physics
33 3.7
34
35 Connor Darling
36 -----
37 18
38 Computer Science
39 4
40
41 Carlos Aguilera
42 -----
43 21
44 Computer Science
45 4.5
46
47 Rand Om
48 -----
49 33
50 Math
51 2.3
52
53 Coo Lguy
54 -----
55 10
56 Science
57 3.4
58
59 Larry Chad
```

```
60 -----
61 14
62 Science
63 2.3
64
65 POPPING
66 Name:   John Smith
67 Age:    20
68 Major:  Math
69 GPA:    3.5
70
71 Average GPA: 3.34286
72
73 Anna White
74 -----
75 19
76 English
77 3.2
78
79 Paul Johnson
80 -----
81 18
82 Physics
83 3.7
84
85 Connor Darling
86 -----
87 18
88 Computer Science
89 4
90
91 Carlos Aguilera
92 -----
93 21
94 Computer Science
95 4.5
96
97 Rand Om
98 -----
99 33
100 Math
101 2.3
102
103 Coo Lguy
104 -----
105 10
106 Science
107 3.4
108
109 Larry Chad
110 -----
111 14
112 Science
113 2.3
```

```

1 #ifndef _HEADER_H_
2 #define _HEADER_H_
3 #include <iostream>
4 #include <string>
5 #include <fstream>
6 #include <iomanip>
7 #include <map>
8 #include <unordered_map>
9 #include "studentNode.h"
10
11 using namespace std;
12
13 void readData(StudentNode** head, map<string, StudentNode> &studentData);
14 void heading();
15 void dispList(StudentNode* &head);
16 void pop(StudentNode** head, map<string, StudentNode> &studentData);
17 void search(StudentNode* &head, map<string, StudentNode> &studentData);
18 void average(StudentNode** head);
19
20 #endif
21
22 #ifndef _STUDENTNODE_H_
23 #define _STUDENTNODE_H_
24 #include "header.h"
25
26 struct StudentNode {
27     string name;
28     int age;
29     string major;
30     float gpa;
31     StudentNode* nextNode;
32 };
33
34 #endif
35
36 #include "../include/header.h"
37 #include "../include/studentNode.h"
38
39 int main()
40 {
41     heading();
42     StudentNode* head = nullptr;
43     /*
44     Using map to get myself understanding hashing and Big O(1) with
45 hashing so I used an ordered map just to keep the elements in the map the
46 same as what was read in the file, also made the name the id to hash
47 */
48     map<string, StudentNode> studentData;
49     readData(&head, studentData); // a ptr to ptr for accesses the ptr in main
50
51     search(head, studentData);
52     pop(&head, studentData);
53     average(&head);
54     dispList(head);
55
56     return 0;
57 }

```

```

58
59 #include "../include/header.h"
60 #include "../include/studentNode.h"
61
62 void readData(StudentNode** head, map<string, StudentNode> &studentData)
63 {
64     StudentNode* node = nullptr;
65     node = new StudentNode;
66     *head = node;
67     string temp;
68
69     ifstream inFile;
70     inFile.open("inFile.txt");
71     if (inFile.is_open())
72     {
73         while (!inFile.eof())
74         {
75             getline(inFile, node->name);
76             inFile >> node->age;
77             inFile.ignore(1000, '\n');
78             getline(inFile, node->major);
79             inFile >> node->gpa;
80             inFile.ignore(1000, '\n');
81             getline(inFile, temp);
82
83             if (node->name[node->name.size() - 1] == ' ')
84             {
85                 node->name = node->name.substr(0, node->name.size() - 1);
86             }
87
88             if (node->major[node->major.size() - 1] == ' ')
89             {
90                 node->major = node->major.substr(0, node->major.size() - 1);
91             }
92
93             studentData[node->name] = *node;
94
95             if (!inFile.eof())
96             {
97                 node->nextNode = new StudentNode;
98                 node = node->nextNode;
99             }
100             else
101                 node->nextNode = nullptr;
102         }
103     }
104     else {
105         cout << "File did not open successfully!\n";
106         delete node;
107     }
108     inFile.close();
109 }
110
111 #include "../include/header.h"
112 #include "../include/studentNode.h"
113
114 void dispList(StudentNode* &head)
115 {
116     StudentNode* node = head;
117     while (node != nullptr)
118     {

```

```

118     cout << node->name << "\n";
119     cout << "-----\n";
120     cout << node->age << "\n";
121     cout << node->major << "\n";
122     cout << node->gpa << "\n\n";
123     node = node->nextNode;
124 }
125
126 }
127
128 #include "../include/header.h"
129 #include "../include/studentNode.h"
130
131 void pop(StudentNode** head, map<string, StudentNode> &studentData)
132 {
133     StudentNode* node = *head;
134     if (node != nullptr)
135     {
136         cout << left;
137         cout << "POPPING\n";
138         cout << setw(8) << "Name:" << node->name << "\n";
139         cout << setw(8) << "Age:" << node->age << "\n";
140         cout << setw(8) << "Major:" << node->major << "\n";
141         cout << setw(8) << "GPA:" << node->gpa << "\n\n";
142         cout << right;
143
144         studentData.erase(node->name);
145         *head = node->nextNode;
146         delete node;
147     }else
148         cout << "The stack is empty!\n\n";
149 }
150
151
152 #include "../include/header.h"
153 #include "../include/studentNode.h"
154
155 void search(StudentNode* &head, map<string, StudentNode> &studentData)
156 {
157     string key;
158
159     cout << "Enter name of student to search: ";
160     getline(cin, key);
161
162
163     if (studentData.find(key) != studentData.end()) //Using .find method in
the map to find key
164     {
165         StudentNode node = studentData[key];
166
167         cout << left;
168         cout << "\nFOUND\n-----";
169         cout << setw(9) << "\nName:" << node.name << "\n";
170         cout << setw(8) << "Age:" << node.age << "\n";
171         cout << setw(8) << "Major:" << node.major << "\n";
172         cout << setw(8) << "GPA:" << node.gpa << "\n\n";
173         cout << right;
174     }else
175         cout << "Student was not found!\n";
176 }

```

```
177
178 #include "../include/header.h"
179 #include "../include/studentNode.h"
180
181 void average(StudentNode** head)
182 {
183     StudentNode* node = nullptr;
184     node = *head;
185     int size = 0;
186     float totalGPA = 0;
187
188     while (node != nullptr)
189     {
190         totalGPA += node->gpa;
191         node = node->nextNode;
192         ++size;
193     }
194     cout << "Average GPA: " << totalGPA/size << "\n";
195 }
```