```cpp
//****************************************************************************
**
//
//              File:                           studentList.cpp
//
//              Student:                        Sean Herrick
//
//              Assignment:                     Program #08
//
//              Course Name:            Data Structures II
//
//              Course Number:          COSC 3100-01
//
//              Due:                            October 7th, 2022
//
//
//              This program displays, prints, and enters in student data.
//
//              Other files required:
//                      1.      Node.h
//                      2.      HashTable.h
//                      3.      student.h
//
//****************************************************************************
**

#include <iostream>
#include <fstream>
#include <new>
#include <iomanip>

using namespace std;

#include "HashTable.h"
#include "student.h"

//****************************************************************************
**

void process ( HashTable <Student> & );
char getChoice ( );
void buildList ( HashTable <Student> & );
void displayStudents ( HashTable <Student> & );
void printStudents ( HashTable <Student> & studentList );
void addStudent ( HashTable <Student> & );
void removeStudent ( HashTable <Student> &);
void findStudent ( HashTable <Student> &);
void displayStatistics ( HashTable <Student> &);

//****************************************************************************
**

int main ( )
{
        HashTable <Student> studentList ( 61 );

        buildList ( studentList );
        process ( studentList );

        return 0;
}

//****************************************************************************
**
```

```cpp
void buildList ( HashTable <Student>& studentList )
{
        Student student;
        ifstream studentData;

        studentData.open ( "studentFile.txt" );

        while ( studentData >> student.id )
        {
                studentData.ignore ( );
                studentData.getline ( student.name, 50 );
                studentData.getline ( student.citystate, 50 );
                studentData >> student.phone >> student.gender >> student.year >> student.credits
                                        >> student.gpa >> student.major;
                studentList.insert ( student );
        }
}

//**************************************************************************************************
**

void process ( HashTable <Student> & studentList )
{
        char choice;

        do
        {
                choice = getChoice ( );

                switch ( choice )
                {
                case 'A':
                        addStudent ( studentList );
                        break;
                case 'F':
                        findStudent ( studentList );
                        break;
                case 'R':
                        removeStudent ( studentList );
                        break;
                case 'S':
                        displayStatistics ( studentList );
                        break;
                case 'D':
                        displayStudents ( studentList );
                        break;
                case 'P':
                        printStudents ( studentList );
                        break;
                case 'Q':
                        break;
                }
        }
        while ( choice != 'Q' );
}

//**************************************************************************************************
**

char getChoice ( )
{
        char choice = ' ';
        bool valid;
```

```cpp
        cout << "======== MENU ========\n"
             << "A:     Add a new Student\n"
             << "F:     Find a Student Record\n"
             << "R:     Remove a Student\n"
             << "S:     Statistics\n"
             << "D:     Display Student Records\n"
             << "P:     Print Student Records\n"
             << "Q:     Quit\n"
             << "Enter a choice: ";

        do
        {
                cin >> choice;
                choice = toupper ( choice );

                switch ( choice )
                {
                case 'A':
                        valid = true;
                        break;
                case 'F':
                        valid = true;
                        break;
                case 'R':
                        valid = true;
                        break;
                case 'S':
                        valid = true;
                        break;
                case 'D':
                        valid = true;
                        break;
                case 'P':
                        valid = true;
                        break;
                case 'Q':
                        valid = true;
                        break;
                default:
                        valid = false;
                        cout << "\ainvalid choice\n" << "Please try again: ";
                        break;
                }
        }
        while ( ! ( valid ) );

        return choice;
}

//********************************************************************************************
**

void addStudent ( HashTable <Student> & studentList )
{
        Student student;
        bool success = false;

        cout << "Enter new student ID: ";
        cin >> student.id;

        cin.ignore ( );
        cout << "Enter new student name: ";
        cin.getline ( student.name, 50 );

        cout << "Enter new student city and state: ";
```

```cpp
        cin.getline ( student.citystate, 50 );

        cin.ignore ( );
        cout << "Enter new student phone number: ";
        cin.getline ( student.phone, 12 );

        cout << "Enter new student year: ";
        cin >> student.year;

        cout << "Enter new student gender: ";
        cin >> student.gender;

        cin.ignore ( );
        cout << "Enter new student major: ";
        cin.getline ( student.major, 50 );

        cout << "Enter new student credits: ";
        cin >> student.credits;

        cout << "Enter new student gpa: ";
        cin >> student.gpa;

        if ( studentList.insert ( student ) )
        {
                cout << "New student added!" << endl << endl;
        }

        else
        {
                cout << "New student was not added." << endl << endl;
        }
}
//************************************************************************************************
**

void displayStudents ( HashTable <Student> & studentList )
{
        studentList.displayTable ( );
}

//************************************************************************************************
**

void printStudents ( HashTable <Student>& studentList )
{
        studentList.writeFile ( );
}


void removeStudent ( HashTable <Student>& studentList )
{
        Student student;
        bool success = false;

        cout << "Enter the student ID that you want to remove: ";
        cin >> student.id;

        if ( studentList.remove ( student ) )
        {
                cout << "Student successfully removed!" << endl << endl;
                success = true;
        }

        else
```

```cpp
        {
                cout << "Student could not be found." << endl << endl;
        }

}

//************************************************************************************
**

void findStudent ( HashTable <Student> & studentList )
{
        Student student;

        cout << "Enter the student id you want to find: ";
        cin >> student.id;

        if ( studentList.retrieve ( student ) )
        {
                cout << endl << student.id << endl;
                cout << student.name << endl;
                cout << student.gender << endl;
                cout << student.citystate << endl;
                cout << student.phone << endl;
                cout << student.major << endl;
                cout << student.credits << endl;
                cout << student.year << endl;
                cout << student.gpa << endl;

                cout << endl << "Student successfully retrieved!" << endl << endl;
        }

        else
        {
                cout << "Student was not found." << endl << endl;
        }
}

//************************************************************************************
**

void displayStatistics ( HashTable <Student> & studentList )
{
        Student student;

        studentList.statistics ( );

        if ( studentList.isEmpty ( ) )
        {
                cout << endl << endl << "The table is empty" << endl << endl;
        }

        else
        {
                cout << endl << endl << "The table is not empty" << endl << endl;
        }
}

//************************************************************************************
**

/*

Table size:                     67

Number of Elements:             61
```

```
Empty Positions:          28

Num. of Chains:           17

Max Chain Length:         4

Num. of Collisions:       22

Avg. Chain Length:        1.3

Percent Collisions:       36.1%

Load Factor:              58.2%

Avg # Search Steps:       1.508


The table is not empty.

*/
```