```
// Written by Jared Dyreson
// Days bug free: -1
#include <iostream>
#include <string>
#include <fstream>
#include <sstream>
#include <algorithm>
using namespace std;
/*
Highly detailed notes inbound

How to compile me on Linux: g++ -std=c++11 Individual_Project_One.cpp -o Individual_Project_On
e

  --Pseudocode begin--
    - This is all done at first so all information is easily accessed without having to run ce
rtain options <NOTE>
    - iterate only in the first 5 lines as we are only allowed to use constants, <countLines>
in Project Two remedies this
    - initialize arrays and variables inside main <main>
    - read the file with the alotted amount of lines, [find names (first and last), calculate
score, assign correct letter grade], append them to their respective arrays
    - select from menu, case/switch combo!
      - 'A': report all iformation about students
      - 'H': Tell who has the highest grade
        - will need to be redone for Project Two to account for multiple same high scores
        - in the form of checking for duplicate records
      - 'S': Search for a student
        - Report if the student is in the records
          - If so, print high score along with their name
          - If not, tell user the student could not be found
      - 'G': Save student Information to StudentData.txt
      - 'Q': Quit
        - Break immediately
      - run until the selection is 'Q'
  --Pseudocode end--

  <NOTES>
  - cout << "\033[2J\033[1;1H"; <a way to clear the screen because I have OCD, not fundamental
ly important to runtime, just my sanity. Also was written by this guy on StackOverflow :) -->
https://stackoverflow.com/questions/4062045/clearing-terminal-in-linux-with-c-code>
*/

int main(){
  // unchanging values
  const int NAMES = 5;
  const int GRADES = 6;

  // beautiful arrays
  string namesArray[NAMES] = {};
  char letterGrade[NAMES] = {};
  int allGrades[NAMES][GRADES] = {};

  // allows us to open the file for reading
  ifstream stream;
  string line, firstName, lastName, fullName;
  stream.open("/home/jared/Desktop/CompSci-CSUF/CPSC-121/Projects/Project One/CPSC121data");
  if (!stream.is_open()){
    // cannot open file so we terminate immediately
    cerr << "Unable to open, cowardly refusing" << "\n";
    exit(1);
  }
  for (int i = 0; i < NAMES; i++){
    getline(stream, line, ',');
    // this took a while, was originally using the boost library to trim the string but I knew
```

```
 there must be a more standard method
    // there was an extra whitespace after using the comma as a delimeter that was not allowin
g me to print "Student Name: <value from array>" in one line
    stringstream namesStream(line);
    while (namesStream >> firstName >> lastName){
      fullName = firstName + " " + lastName;
      namesArray[i] = fullName;
    }
    for (int j = 0; j < GRADES; j++){
        getline(stream, line, ',');
        stringstream ss(line);
        int var;
        while (ss >> var){
          allGrades[i][j] = var;
        }
    }
  }
  for (int k = 0; k < NAMES; k++){
    int sum = 0;
    for (int l = 0; l < GRADES; l++){
      sum+=allGrades[k][l];
    }
    // calculate the score of all of the students one by one
    double curve = 90;
    int percentOfStudent = (sum / curve)*100;
    char grade;
    // check grade based on percent, can be replaced with a function
    if (percentOfStudent == 100 && percentOfStudent < 100 && percentOfStudent >= 90){
      grade = 'A';
    }
    else if (percentOfStudent < 89 && percentOfStudent >= 80){
      grade = 'B';
    }
    else if (percentOfStudent < 79 && percentOfStudent >= 70){
      grade = 'C';
    }
    else if (percentOfStudent < 69 && percentOfStudent >= 60){
      grade = 'D';
    }
    else {
      grade = 'F';
    }
    letterGrade[k] = grade;
  }
  char selection;
  do{
    // display options
    // can be replaced with a function
    cout << "+------+" << endl;
    cout << "| Menu |" << endl;
    cout << "+ -----+" << endl;
    cout << "A) Display all information about students" << endl;
    cout << "H) Highest grade in the class" << endl;
    cout << "S) Search for a student" << endl;
    cout << "G) Save Student information to StudentData.txt" << endl;
    cout << "Q) Quit" << endl;
    cout << "Selection: ";
    cin >> selection;
    switch(selection){
      case 'A':
      {
        // display all information
        cout << "\033[2J\033[1;1H";
        for (int i = 0; i < NAMES; i++){
          cout << "Student Name: " << namesArray[i] << endl;
```

```
          cout << "Grade: " << letterGrade[i] << endl;
          for (int j = 0; j < GRADES; j++) {
            if (j == (GRADES - 1)){
              cout << allGrades[i][j] << endl;
            }
            else{
              cout << allGrades[i][j] << " ";
            }
          }
          cout << endl;
        }
        break;
      }
      case 'H':
      {
        // highest grade in the class
        cout << "\033[2J\033[1;1H";
        int tempallGrades[GRADES] = {};
        int largest = 0, counter = 0, percent = 0;
        double curve = 90;
        string studentWithHighestGrade;
        for (int i = 0; i < NAMES; i++){
          for (int j = 0; j < GRADES; j++){
            counter+=allGrades[i][j];
          }
          percent = (counter / curve)*100;
          tempallGrades[i] = percent;
          counter = 0;
        }

        for (int i = 0; i < NAMES; i++) {
          if (tempallGrades[i] > largest){
            largest = tempallGrades[i];
            studentWithHighestGrade = namesArray[i];
          }
        }
        cout << "Student with Highest Score: " << studentWithHighestGrade << "\n";
        cout << "Overall grade: " << largest  << "%" << endl;
        break;
      }
      case 'S':
      {
        // search for a student
        cout << "\033[2J\033[1;1H";
        string studentSelection;
        cout << "Enter name: ";
        cin.ignore();
        getline(cin, studentSelection);
        bool studentIsFound;
        for (int i = 0; i < NAMES; i++) {
          // we found them
          if (namesArray[i] == studentSelection){
            cout << studentSelection << " is in our records" << endl;
            // this is so we can check after this loop ends if the student is not found
            studentIsFound = true;
            for (int k = 0; k < GRADES; k++){
              // display all of the scores
              // when we reach the end of the loop, we change the formatting, printing a newli
ne and allowing the rest of the menu to be displayed
              if (k == (GRADES - 1)){
                cout << allGrades[i][k] << endl;
              }
              else{
                // print all in one line while it is not the last
                cout << allGrades[i][k] << " ";
```

```
            }
          }
          // no need to continue going through the loop
          break;
        }
        else{
          // continue to say that we cannot find the student
          studentIsFound = false;
        }
      }
      // this is where having that extra boolean assignment comes in handy
      // if we did find the student, this condition is false and will not be executed
      if (!studentIsFound) {
        cerr << "We could not find " << studentSelection << " in our records" << endl;
      }
      break;
    }
    case 'G':
    {
      cout << "\033[2J\033[1;1H";
      cout << "Saving...." << endl;
      // open a new file to write to
      ofstream write;
      write.open("StudentData.txt");
      for (int i = 0; i < NAMES; i++){
        // write all information to file in one line
        write << "Student: " << namesArray[i] << "\n" << "Letter Grade: " << letterGrade[i]
<< endl;
      }
      write.close();
      cout << "Successfully wrote data to StudentData.txt" << endl;
      break;
    }
    case 'Q':
    {
      // immediately quit, semi useless function. Would be even more useless if I said exit(
0) because the do while loop is assigned to terminate the loop and assign return 0 anyways
      break;
    }
    default:
    {
      cout << "\033[2J\033[1;1H";
      cerr << "Invalid option" << endl;
      break;
    }
  }
} while (selection != 'Q');
// exit gracefully
return 0;
}
```

Terminal

File Edit View Search Terminal Help

```
+------+
| Menu |
+ -----+
A) Display all information about students
H) Highest grade in the class
S) Search for a student
G) Save Student information to StudentData.txt
Q) Quit
Selection:
```

Terminal

File Edit View Search Terminal Help

```
Student Name: John Smith
Grade: C
5 10 15 10 11 14

Student Name: Mary Doe
Grade: B
4 15 13 25 10 13

Student Name: Larry Tran
Grade: B
3 14 20 12 18 11

Student Name: Tom Johnson
Grade: C
5 10 11 15 17 12

Student Name: Bella Morrison
Grade: B
4 12 18 13 18 14

+------+
| Menu |
+ -----+
A) Display all information about students
H) Highest grade in the class
S) Search for a student
G) Save Student information to StudentData.txt
Q) Quit
Selection:
```

Terminal

File Edit View Search Terminal Help

```
Student with Highest Score: Mary Doe
Overall grade: 88%
+------+
| Menu |
+ -----+
A) Display all information about students
H) Highest grade in the class
S) Search for a student
G) Save Student information to StudentData.txt
Q) Quit
Selection:
```

Terminal

File Edit View Search Terminal Help

```
Enter name: Jared Dyreson
We could not find Jared Dyreson in our records
+------+
| Menu |
+ -----+
A) Display all information about students
H) Highest grade in the class
S) Search for a student
G) Save Student information to StudentData.txt
Q) Quit
Selection:
```

Terminal

File Edit View Search Terminal Help

```
Enter name: Mary Doe
Mary Doe is in our records
4 15 13 25 10 13
+------+
| Menu |
+ -----+
A) Display all information about students
H) Highest grade in the class
S) Search for a student
G) Save Student information to StudentData.txt
Q) Quit
Selection:
```

Terminal

File Edit View Search Terminal Help

```
Saving....
Successfully wrote data to StudentData.txt
+------+
| Menu |
+ -----+
A) Display all information about students
H) Highest grade in the class
S) Search for a student
G) Save Student information to StudentData.txt
Q) Quit
Selection:
```

Terminal

File Edit View Search Terminal Help

```
Saving....
Successfully wrote data to StudentData.txt
+------+
| Menu |
+ -----+
A) Display all information about students
H) Highest grade in the class
S) Search for a student
G) Save Student information to StudentData.txt
Q) Quit
Selection: Q
Press any key to continue...
```

```
jared@jared-xps ~/Desktop/CompSci-CSUF/CPSC-121/Projects/Project One $ cat StudentData.txt
Student: John Smith
Letter Grade: C
Student: Mary Doe
Letter Grade: B
Student: Larry Tran
Letter Grade: B
Student: Tom Johnson
Letter Grade: C
Student: Bella Morrison
Letter Grade: B
jared@jared-xps ~/Desktop/CompSci-CSUF/CPSC-121/Projects/Project One $
```