```
// Attached: HW 2(d)
// File: HW_2d.pdf
//
//
// Programmer: Gage Alvarez
// Class: CS 1B
// Instructor: Med Mogasemi
//
//
// Program: Display Student Grades
//
// Description:
// The program reads grades from a text file, and converts them to GPA, which is
// then displayed for each student and each subject.
//
#include <fstream>
#include <iomanip>
#include <iostream>

const int g_STUDENTS = 5;
const int g_CLASSES = 3;

// function prototypes
void loadGradesFromFile(char student_grades[][g_CLASSES]);
void displayGrades(char student_grades[][g_CLASSES]);
float calculateStudentGPA(char student_grades[][g_CLASSES], int &student);
void displayStudentGPA(char student_grades[][g_CLASSES]);
float calcuateGPABySubject(char student_grades[][g_CLASSES],
                           const int &subject);
void displayGPAPerClass(char student_grades[][g_CLASSES]);

int main() {
  // variable init
  char student_grades[g_STUDENTS][g_CLASSES]{};

  loadGradesFromFile(student_grades);
  displayGrades(student_grades);
  displayStudentGPA(student_grades);
  displayGPAPerClass(student_grades);
} // end of main
// function loadGradesFromFile gets the file, and reads the contents from the
// file into the 2d array. Inputs: grades.txt, the grades file, and the
// student_grades array to load the contents of the file
void loadGradesFromFile(char student_grades[][g_CLASSES]) {
  std::fstream inFile("grades.txt");
  if (inFile.fail()) {
     std::cout << "File failed to open" << '\n';
  }

  for (int i = 0; i < g_STUDENTS; i++) {
     for (int j = 0; j < g_CLASSES; j++) {
        inFile >> student_grades[i][j];
     }
```

```
  }
} // end of loadGradesFromFile

// function displayGrades takes the contents of the 2d array and prints them to
// the console Inputs: student_grades array
void displayGrades(char student_grades[][g_CLASSES]) {
  std::cout << "Display overall grades" << '\n';
  for (int i = 0; i < g_STUDENTS; i++) {
      std::cout << "Grades for student " << i + 1 << '\t';
      for (int j = 0; j < g_CLASSES; j++) {
          std::cout << std::setw(3) << student_grades[i][j] << " ";
      }
      std::cout << '\n';
  }
} // end of displayGrades

// function calculateStudentGPA calculates the students GPA based on the grades
// in the 2d array Inputs: student_grades array as well as a numerical student
// parameter to specify which student's grade is being calculated Outputs: the
// total GPA of the student returned to function call
float calculateStudentGPA(char student_grades[][g_CLASSES], int &student) {
  float total = 0.0;
  for (int i = 0; i < g_CLASSES; i++) {
      if (student_grades[student][i] == 'A') {
          total += 4.0;
      } else if (student_grades[student][i] == 'B') {
          total += 3.0;
      } else if (student_grades[student][i] == 'C') {
          total += 2.0;
      } else if (student_grades[student][i] == 'D') {
          total += 1.0;
      } else {
          total += 0.0;
      }
  }
  return total / g_CLASSES;
} // end of calculateStudentGPA

// function displayStudentGPA displays the completed calculations of
// calculateStudentGPA Inputs: student_grades array and number returned by
// calculateStudentGPA
void displayStudentGPA(char student_grades[][g_CLASSES]) {
  std::cout << "student GPAs" << '\n';
  for (int i = 0; i < g_STUDENTS; i++) {
      const float GPA = calculateStudentGPA(student_grades, i);
      std::cout << "GPA for student " << i + 1 << ": " << GPA << '\n';
  }
} // end of displayStudentGPA

// function calcuateGPABySubject calculates the average gpa across one subject
// instead of for a specific student Inputs: student_grades array and a subject
// parameter to specify the subject Outputs: returns average GPA for specified
// subject to function call
float calcuateGPABySubject(char student_grades[][g_CLASSES],
                           const int &subject) {
```

```
  float total = 0.0;
  for (int i = 0; i < g_STUDENTS; i++) {
     if (student_grades[i][subject] == 'A') {
        total += 4.0;
     } else if (student_grades[i][subject] == 'B') {
        total += 3.0;
     } else if (student_grades[i][subject] == 'C') {
        total += 2.0;
     } else if (student_grades[i][subject] == 'D') {
        total += 1.0;
     } else {
        total += 0.0;
     }
  }
  return total / g_STUDENTS;
} // end of calcuateGPABySubject

// function displayGPAPerClass calls and displays calcuateGPABySubject
// Inputs: student_grades array as well as calcuateGPABySubject value
// prints results  to console
void displayGPAPerClass(char student_grades[][g_CLASSES]) {
  std::cout << "Average GPA per subject\n";
  std::cout << std::setw(3) << "English\tHistory\tMath\n";
  for (int i = 0; i < g_CLASSES; i++) {
     float classGPA = calcuateGPABySubject(student_grades, i);
     std::cout << std::setw(3) << classGPA << "\t";
  }
} // end of displayGPAPerClass
```