

Software Engineering Assignment SDD

Trenton MacNinch, Michael Lin, and Ying Gao

2024/11/19

Contents

1	Introduction	3
1.1	Date of Issue	3
1.2	Context	3
1.3	Scope	3
1.4	Authorship	3
1.5	Change History	3
1.6	Summary	3
2	Software Architecture	3
2.1	Overview	3
3	Detailed Design	4
3.1	Overview	4
3.2	Student Information	4
3.3	Course Information	4
3.4	Creating Output	4

1 Introduction

1.1 Date of Issue

2024/11/19

1.2 Context

1.3 Scope

1.4 Authorship

Trenton MacNinch, Michael Lin, and Ying Gao

1.5 Change History

1.6 Summary

The program is used with the following syntax:

```
calcgrades [name file] [course file] [output file]
```

The application reads two input text files and outputs one file with specific formatting. The input files are given by the user, formatted as the following:

[name file]

Student ID, Student Name

000000001, John Hay

000000002, Mary Smith

...

[course file]

Student ID, Course Code, Test 1, Test 2, Test 3, Final exam

000000001, CP317, 75.3, 80.4, 60.3, 70.5

000000002, CP414, 80.2, 90.5, 50.4, 75.6

000000001, CP414, 60.5, 70.6, 80.6, 75.6

...

The output file is formatted as the following, with special notice given to student IDs being in ascending order:

[output file]

Student ID, Student Name, Course Code, Final grade

000000001, John Hay, CP317, 66.7

000000001, John Hay, CP414,

000000002, Mary Smith, CP414, 74.8

...

2 Software Architecture

2.1 Overview

Given the requirements for our application, we have chosen to create a C++ script running on a Linux system. The overall system architecture then includes the following;

- 2 Input Files
- 1 Executable
- 1 Output File

3 Detailed Design

3.1 Overview

The application loads all student information into a map, then loads all course information into a vector of custom course objects, and then sorts the vector of course objects. The program will then proceed to iterate through all stored course information and attach student names based on IDs while placing it in the designated output file.

3.2 Student Information

The program opens the name file as a normal C++ file handle. The program then calls a function that generates a map. This function reads the info from a line in the name file, validates the given information, and adds the student info to the map if the information is acceptable.

3.3 Course Information

The program opens the course file as a file handle, and uses a function to generate a vector of course objects. The function reads the information from a line in the course file, validates the given course information, and rejects any invalid courses. The program then sorts the vector of course information by student ID to enable a more efficient outputting phase.

3.4 Creating Output

The program takes the information stored in the student id map and the course object vector, and combines it according to the output format while pushing it to the file handle for the output file. The program then closes.