Course Grades

Stephen Gerkin
September 2, 2019
CIST 2362 – C++ II
CRN 22282
Programming Lab 03

Program Description

Course Grades program gets and stores information about student course grades, calculates their average, determines the course grade, and displays the information to the user. This is accomplished using a structure to store the information about each individual student, an array to store the grades for course tests, and functions that determine the average and course grade.

The user is prompted to input the number of tests for the course, the number of students in the course, and then the information for each student including name and each test score. Average and course grade are then calculated and determined. A table containing each student ID, name, and course information is then displayed to the user.

The program then frees all memory used to store the information and the user is prompted if they wish to repeat the program.

Function Definitions

courseGrade.cpp

Signature	Description
courseGrade	Main application logic for the program.
getValidIntegerInput	Displays an out message to the user and gets input
constructStudentCourseGrade	Constructs a StudentCourseGrade structure
getStudentName	Gets the name of a student from the user
getTestScore	Gets a single test score from the user
calculateAverage	Calculates the average of tests for an array of tests
determineGrade	Determines the letter grade based off a number value
displayStudentGrades	Displays all student course and grade information
printLineBreak	Prints a new line separator between table elements
repeatProgram	Checks if user wants to repeat program
main	Entry point for program. Calls courseGrade while user wants to repeat the application

Function Details (courseGrade.cpp)

courseGrade

void courseGrade()
Course Grade application start point. Gets number of tests from user and number of students in course. Allocates an array of StudentCourseGrade structures to store information. Calls functions to fill array elements and get all necessary information before calling function to display information to the user.

getValidIntegerInput

unsigned int getValidInteger(const std::string &outMsg)
Displays a message string to the user to get an unsigned integer value. Validates input. Does not allow values less than or equal to 0.

Parameters:
outMsg – String to display to the user indicating what information is to be entered.

Returns:
User input as an unsigned integer value.

constructStudentCourseGrade

void constructStudentCourseGrade(StudentCourseGrade &studentCourseGrade,
unsigned int &numTests)
Constructs a StudentCourseGrade structure object. Gets a student name and test scores from the user. Calls function to calculate the average and determine the grade to store inside the object.

Parameters:
studentCourseGrade – Structure object to store information in.
numTests – Number of tests to get values for.

getStudentName

std::string getStudentName(const unsigned int &idNum)
Gets a student name from the user for the corresponding ID number. Does not allow empty string entry.

Parameters:
idNum – The student ID number to get the name for.

Returns:
User input for student name as a string

getTestScore

double getTestScore(const int &testNumber)
Gets a single test score from the user. Validates for valid unsigned floating-point number entry. Does not allow entries below 0.

Parameters:
testNumber – The displayed test number to get the score for.

Returns:
User input for a test score as an unsigned double value.

Function Details (courseGrade.cpp)

calculateAverage

double calculateAverage(const double *arr, const unsigned int &arrSize)
Calculates the arithmetic mean (average) of an array by adding the elements and dividing by total number of elements.

Parameters:
*arr – A pointer to an array of double values.
arrSize – The number of elements in the array.

Returns:
The calculated average value of the array as a double value.

determineGrade

char determineGrade(const double &average)
Determines the letter grade representation for an averaged value using the following criteria:

91-100(+)	**A**
81-90	**B**
71-80	**C**
61-70	**D**
60 or below	**F**

Paramters:
average – The average score to get the letter grade for.

Returns:
A char letter representation of the score.

displayStudentGrades

void displayStudentGrades(const StudentCourseGrade *students, const int &numStudents)
Displays a table containing student information including:

1. Student ID Number
2. Student Name
3. Student Average Course Grade
4. Associated Letter Grade for average

Parameters:
*students – A pointer to an array of StudentCourseGrade elements with student information for the current course.
numStudents – Number of students in the array

printLineBreak

void printLineBreak(const int &charsPerLine)
Prints a new line to the console and a series of dashes to indicate a new line in the displayed table. Prints a second new line to the console.

Parameters:
charsPerLine – Number of characters per line of the table.

Function Details (courseGrade.cpp)

repeatProgram

bool repeatProgram()
Determines if the user wants to repeat the program again with a simple “Y” or “N” prompt (case insensitive

Returns:
True if user wants to repeat the program

main

int main()
Main entry point for the program

Returns:
0 to shell to indicate termination of program

InputValidation.h

Signature	Description
validYN	Validates input against “Y” or “N” input
validUnsignedInt	Validates input against unsigned integer input
validUnsignedFloat	Validates input against unsigned floating point number input

Function Details (InputValidation.h)

validYN

bool validYN(const std::string &input)
Uses regular expression parsing to match input against valid “Y” or “N” entry only. Case insensitive.

Parameters:
input – String value to validate

Returns:
True if input matches “Y” or “N”

validUnsignedInt

bool validUnsignedInt(const std::string &input)
Uses regular expression parsing to match input against valid unsigned integer values.

Parameters:
input – String value to validate

Returns:
True if input consists only of an unsigned integer value

Function Details (InputValidation.h)

validUnsignedFloat

bool validUnsignedFloat(const std::string &input)
Uses regular expression parsing to match input against valid unsigned floating-point number values.

Parameters:
input – String value to validate

Returns:
True if the input consists only of an unsigned floating-point number value

Structure Definitions

StudentCourseGrade (location: "StudentCourseGrade.h")

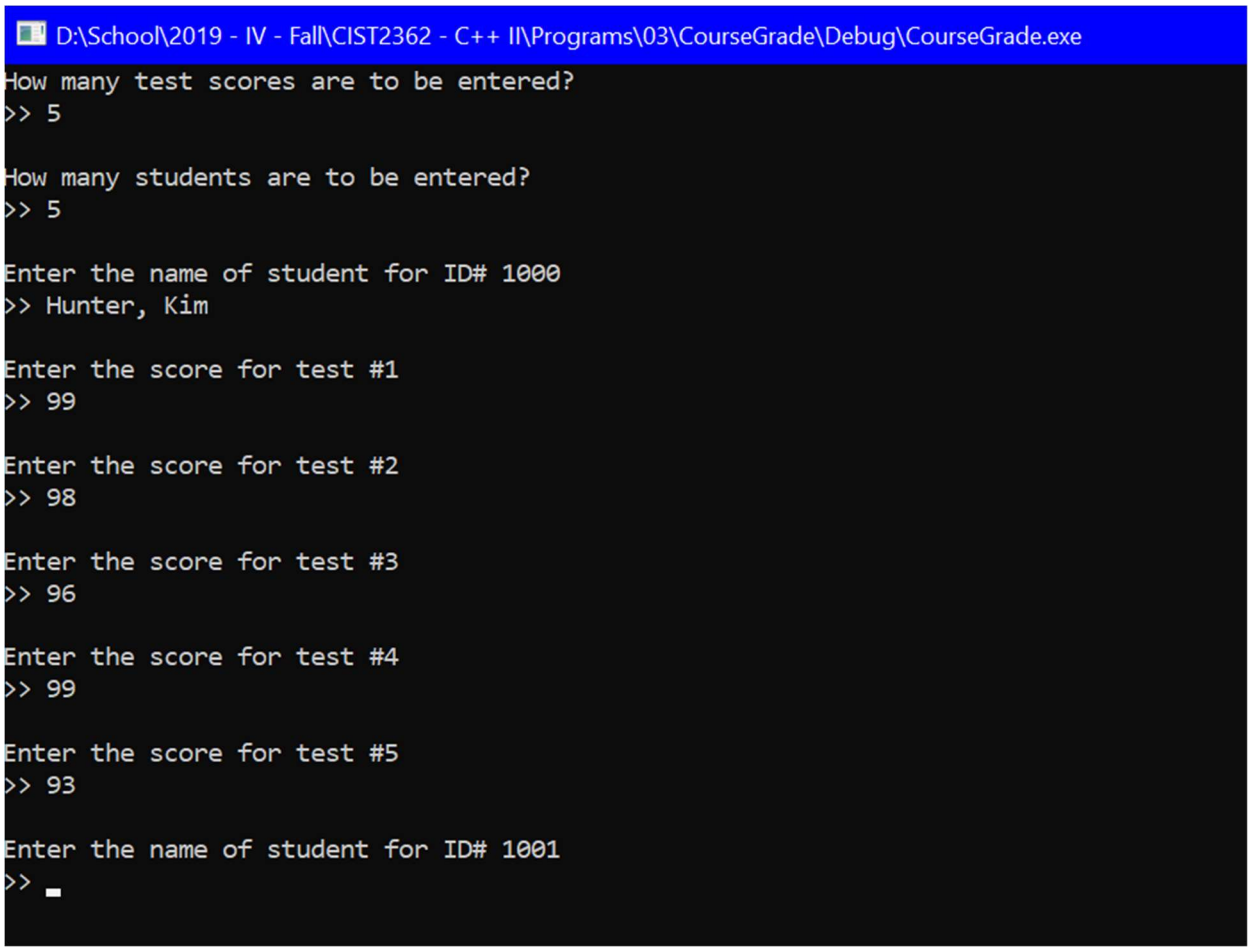
Member type and name	Description
std::string name	Stores a student name
unsigned int idNum	Stores a student ID Number
double * testScores	Stores a pointer to an array of test scores for a student
double average	Stores the average of values in testScores array
char grade	Stores the letter grade of the student's average

Constructors and Destructors	Description
Default (no arguments)	None
Destructor	Frees any memory associated with testScores array if not null

Methods and return type	Description
None	

Sample Program Output

Getting number of tests and students



Displaying table information: