Description of C++ final assignment
Jingjing Yang 2104210
Grade counter Program include the following functions:
1. Add student.
Here the user can input student name, amount of accepted weekly exercises, grade of final assignment. Those data are read and stored.

2. Print out the student info including course grade.
Prints list of students that have been input to the program. The user can decide if the list is sorted based on student names alphabetically or grade ascending. Each row shows name of the student, amount of accepted weekly exercises, grade of final assignment, precise grade as well as rounded grade.
The amount of students in the list is also displayed.

3. Search student.(additions)
Searchs student by certain name or certain grade. The user can decide if the search is based on student name or grade.
If the student(s) under search have been input to the program, then print the student's info. Each row shows name of the student, amount of accepted weekly exercises, grade of final assignment, precise grade as well as rounded grade.
If not, print a message telling the user that the student is not found.

4. Settings.
Here the user can choose emphasis of the weekly exercises and final assignment to the final grade. The default values are 40%/60%.
After reseting, the precise grade and rounded grade will change accordingly when print out.

Some functions used in this program:
bool,nested struct, array[], getline(), setw(x), round(), static_cast<int>
a++, ||, &&, If, nested if, while, for(), switch
subroutie void, &, *,->

grade counter.h

```
#include <iostream>     // for cin and cout streams
using namespace std;
#include <string>      //for string data type
#include <iomanip>         //for stream manipulator setw(x) setprecision(x)
#include <cmath>       //for round()

/****************************************************************************
 *                          struct Student{}                              *
 *    Declaration of structure "Student"                                     *
 *    To group 5 variables related to a student together.                  *
 ***************************************************************************/
struct Student
{
    string stuName;            //student name
    int numExercise;       //amount of acceepted weekly exercises(0-40)
    double gradeAssignment;  //grade of the final assignemnt(0-5)

    double gradeExercise;  //grade from the exercises(0-5)
    double preciseGrade;   //precise final grade precise grade based on emphasis of
exercises and assignment
    int roundGrade;            //rounded final grade 0-5
};

/****************************************************************************
 *                          struct StuList{}                              *
 *    Declaration of structure "StuList"                                     *
 *    To group multiple variables related to the list together.              *
 ***************************************************************************/
struct StuList
{
    struct Student stuArray[1000];  //array of students in the program, using nested
structure,allowing maximum 1000 elements
    int percentA;                  //emphasis of the weekly exercises to the final grade
    int percentB;                  //emphasis of the final assignment to the final grade
    int size;                          //amount of students that have been put in the
program
};
/*******************-----Function Protoypes-----****************************/
void showMenu();
void addStudent(StuList* stulist);
void countGrade(StuList* stulist);
void sortByAlphabet(StuList* stulist);
void sortByGrade(StuList* stulist);
void printStudent(StuList* stulist);
void searchByName(StuList* stulist);
void searchByGrade(StuList* stulist);
void settings(StuList* stulist);
```

grade counter.cpp

```
#include "grade counter.h"

/*******************-----Main Function-----********************************/

int main()
{

    //Variable Declarations
    StuList stulist;      //define structure variable "stulist" in the program
    stulist.percentA = 40;  //initialize emphasis of weekly exercises as 40%, but the user can reset its value from "6.Reset emphasis"
    stulist.percentB = 60;  //initialize emphasis of final assignment as 60%, but its value will change if percentA is reset
    stulist.size = 0;      //initialize amount of student input is 0,add the value will increase 1 everytime a new student added

    int select = 0;

    while (1)//the user can continue choosing another subfunction from the menu, as long as "0. Exit" is not chosen yet
    {
        showMenu();//to call function "showMenu()"

        //check non-numeric input
        bool error;
        do
        {
            error = false;
            cin >> select;//read input of grade for final assignment
            if (cin.fail())
            {
                cout << "Error! Please enter a number between 0-6: ";
                error = true;
                cin.clear();
                cin.ignore(80, '\n');
            }
        } while (error);

        switch (select)
        {
            //Function Calls for different user choice
            //using reference parameter to modify contents of structure variable
        case 1://to add a student
            addStudent(&stulist);
            break;
        case 2://to print the student list based on name Alphabetically
            countGrade(&stulist);
            sortByAlphabet(&stulist);
            printStudent(&stulist);
            break;
        case 3://to print the student list based on final precise grade
            countGrade(&stulist);
            sortByGrade(&stulist);
            printStudent(&stulist);
            break;
        case 4://to find a student by his/her name
            searchByName(&stulist);
            break;
        case 5://to search students with certain fianl score
            searchByGrade(&stulist);
            break;
        case 6://to reset emphasis of exercise and assignment
            settings(&stulist);
            break;
        case 0://to exit program
            system("pause");
            return 0;
            break;
        default:
            cout << "Invaid input! Try again." << endl;//if the user input is outside 0-6
            break;
        }
    }
    system("pause");
    return 0;
}
```

grade counter definition.cpp

```
#include "grade counter.h"

/******************-----Function Definitions-----************************/

/**********************************************************************
 *                              showMenu()                               *
 *      To show 7 different sub-functions in this program.              *
 **********************************************************************/
void showMenu()
{
	cout << "\t***| ————********————|***" << endl;
	cout << "\t***| 1.Add student                    |***" << endl;
	cout << "\t***| 2.Print students Alphabeticaly|***" << endl;
	cout << "\t***| 3.Print students by grade      |***" << endl;
	cout << "\t***| 4.Search student by name        |***" << endl;
	cout << "\t***| 5.Search student by grade       |***" << endl;
	cout << "\t***| 6.Reset emphasis                 |***" << endl;
	cout << "\t***| 0.Exit                           |***" << endl;
	cout << "\t***| ———————————— |***" << endl;

	cout << "\t     Welcome to Grade Mangement System! " << endl;//welcome message
	cout << " Please select a Function from the menu by entering a number 0-6: " << endl;//a prompt to insturct the user to choose a subfunction
}

/**********************************************************************
 *                              addStudent()                             *
 *      To   get input of student name, amount of accepted weekly exercises,   *
 *        and grade for the final assignment.                          *
 **********************************************************************/
void addStudent(StuList* stulist)
{
	string name;
	cout << "\nPlese enter student name: ";// display a prompt to insturct the user to enter data
	cin.ignore();
	getline(cin, name);//read input of student name
	stulist->stuArray[stulist->size].stuName = name;

	int exercise;
	cout << "Please enter score from weekly exercises(0-100 inclusive): ";
```

```
//check non-numeric input
bool error;
do
{
        error = false;
        cin >> exercise;//read input of amount of accepted weekly exercises
        if (cin.fail())
        {
                cout << "Error! Please enter a number between 0-40: ";
                error = true;
                cin.clear();
                cin.ignore(80, '\n');
        }
} while (error);

//to limit input between 0-40
while (exercise < 0 || exercise>40)
{
        cout << "Error! Please enter a number between 0-40: ";
        cin >> exercise;
}
stulist->stuArray[stulist->size].numExercise = exercise;

double assignment;
cout << "Please enter grade for the final assignment(0-5 inclusive): ";

//check non-numeric input
do
{
        error = false;
        cin >> assignment;//read input of grade for final assignment
        if (cin.fail())
        {
                cout << "Error! Please enter a number between 0-5: ";
                error = true;
                cin.clear();
                cin.ignore(80, '\n');
        }
} while (error);

//to limit input between 0-5
while (assignment < 0 || assignment> 5)
{
        cout << "Error! Please enter a number between 0-5: " << endl;
        cin >> assignment;
    }
    stulist->stuArray[stulist->size].gradeAssignment = assignment;

    //update amount of students in the program after a new adding
    stulist->size++;

    cout << "\nStudent added!" << endl;

    system("pause");//press any key to continue
    system("cls");    //clear the screen
}

/*************************************************************************
 *                              countGrade()                              *
 *    To calculate final precise grade and rounded grade.              *
 *    0-19 exercises -> grading for week exercises 0                     *
 *    20-23 exercises -> grading for week exercises 1                    *
 *    24-27 exercises -> grading for week exercises 2                    *
 *    28-32 exercises -> grading for week exercises 3                    *
 *    33-36 exercises -> grading for week exercises 4                    *
 *    37-40 exercises -> grading for week exercises 5                    *
 *************************************************************************/
void countGrade(StuList* stulist)
{
    for (int i = 0; i < stulist->size; i++)
    {
        //rules for grade level 0-5 based on amount of accepted exercises
        if (stulist->stuArray[i].numExercise >= 0 && stulist->stuArray[i].numExercise <= 19)
            stulist->stuArray[i].gradeExercise = 0;
        else  if  (stulist->stuArray[i].numExercise  >=  20  &&  stulist->stuArray[i].numExercise  <=
23)
            stulist->stuArray[i].gradeExercise = 1;
        else  if  (stulist->stuArray[i].numExercise  >=  24  &&  stulist->stuArray[i].numExercise  <=
27)
            stulist->stuArray[i].gradeExercise = 2;
        else  if  (stulist->stuArray[i].numExercise  >=  28  &&  stulist->stuArray[i].numExercise  <=
32)
            stulist->stuArray[i].gradeExercise = 3;
        else  if  (stulist->stuArray[i].numExercise  >=  33  &&  stulist->stuArray[i].numExercise  <=
36)
            stulist->stuArray[i].gradeExercise = 4;
        else  if  (stulist->stuArray[i].numExercise  >=  37  &&  stulist->stuArray[i].numExercise  <=
40)
```

```
                    stulist->stuArray[i].gradeExercise = 5;

                //final precise grade based on grade and emphasis of exercises and assignment
                stulist->stuArray[i].preciseGrade = stulist->stuArray[i].gradeExercise * stulist->percentA
/ 100 + stulist->stuArray[i].gradeAssignment * stulist->percentB / 100;

                //using static_cast<int> to convert float to int
                stulist->stuArray[i].roundGrade = round((stulist->stuArray[i].preciseGrade));
        }
}

/****************************************************************************
 *                              printStudent()                              *
 *      To print list of students that have been inputted to the program.   *
 *      Each row shows name of the student, amount of accepted weekly       *
 *        exercises, grade of final assignment,                             *
 *        precise grade as well as rounded grade.                           *
 ****************************************************************************/
void printStudent(StuList* stulist)
{
        if (stulist->size == 0)
        {
                cout << "\nNo student added yet!" << endl;
        }
        else
        {
                cout << "\n>>>>>>>>>> " << stulist->size << " student(s) have been added to the list
<<<<<<<<<<" << endl;

                cout << "          " << "Name:\t" << "Excercises:\t" << "Assignment:\t" << "Precise
grade:\t" << "Rounded grade:" << endl;

                for (int i = 0; i < stulist->size; i++)
                {
                        cout << i + 1 << "    " << stulist->stuArray[i].stuName << "\t" <<
                                stulist->stuArray[i].numExercise << "\t\t" <<
                                static_cast<int>(stulist->stuArray[i].gradeAssignment) << "\t\t" <<
                                setprecision(1) << fixed << showpoint << stulist->stuArray[i].preciseGrade <<
"\t\t" <<
                                stulist->stuArray[i].roundGrade << endl;
                }
        }
        system("pause");//press any key to continue
        system("cls");    //clear the screen
```

```
}

/**************************************************************************
 *                                  sortByGrade()                                   *
 *      To sort the students on the list, basing on precise grade ascending.    *
 **************************************************************************/
void sortByGrade(StuList* stulist)
{
	for (int i = 0; i < stulist->size - 1; i++)//bubble sorting
	{
		for (int j = 0; j < stulist->size - i - 1; j++)
		{
			if (stulist->stuArray[j].preciseGrade > stulist->stuArray[j + 1].preciseGrade)
			{
				struct Student temp = stulist->stuArray[j];
				stulist->stuArray[j] = stulist->stuArray[j + 1];
				stulist->stuArray[j + 1] = temp;
			}
		}
	}
}

/**************************************************************************
 *                                  sortByAlphabet()                                *
 *      the list is sorted based on student names alphabetically.               *
 **************************************************************************/
void sortByAlphabet(StuList* stulist)
{
	for (int i = 0; i < stulist->size - 1; i++)//bubble sorting
	{
		for (int j = 0; j < stulist->size - i - 1; j++)
		{
			if (stulist->stuArray[j].stuName > stulist->stuArray[j + 1].stuName)
			{
				struct Student temp = stulist->stuArray[j];
				stulist->stuArray[j] = stulist->stuArray[j + 1];
				stulist->stuArray[j + 1] = temp;
			}
		}
	}
}


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

```
 *                                       searchByName()                                     *
 *               To find a student by his/her name.                                    *
 ***************************************************************************/
void searchByName(StuList* stulist)
{
	cout << "\nPlease enter the name of the student you are searching: ";
	string name;
	cin.ignore();
	getline(cin, name);

	bool isExist = false;//set a flag

	//check first if the student is in the list
	for (int i = 0; i < stulist->size - 1; i++)
	{
		if (stulist->stuArray[i].stuName == name)
		{
			isExist = true;
			cout << "Name:\t" << "Excercises:\t" << "Assignment:\t" << "Precise grade:\t" <<
"Rounded grade:" << endl;
			cout << stulist->stuArray[i].stuName << "\t" <<
				stulist->stuArray[i].numExercise << "\t\t" <<
				static_cast<int>(stulist->stuArray[i].gradeAssignment) << "\t\t" <<
				setprecision(1) << fixed << showpoint << stulist->stuArray[i].preciseGrade <<
"\t\t" <<
				stulist->stuArray[i].roundGrade << endl;
		}
	}
	if (isExist == false)
	{
		cout << "Can not find the student!" << endl;
	}

	system("pause");//press any key to continue
	system("cls");    //clear the screen
}

/***************************************************************************
 *                                       searchByGrade()                                    *
 *             To find students with certain final assignment grade.                 *
 ***************************************************************************/
void searchByGrade(StuList* stulist)
{
	cout << "\nPlease enter the grade of the student's final assignment: ";
	double score;

	//check non-numeric input
	bool error;
	do
	{
		error = false;
		cin >> score;//read input of grade for final assignment
		if (cin.fail())
		{
			cout << "Error! Please enter a number between 0-5: ";
			error = true;
			cin.clear();
			cin.ignore(80, '\n');
		}
	} while (error);

	bool isExist = false;//set a flag

	//check first if the student is in the list
	for (int i = 0; i < stulist->size - 1; i++)
	{
		if (stulist->stuArray[i].gradeAssignment == score)
		{
			isExist = true;
			cout << "Name:\t" << "Excercises:\t" << "Assignment:\t" << "Precise grade:\t" <<
"Rounded grade:" << endl;
			cout << stulist->stuArray[i].stuName << "\t" <<
				stulist->stuArray[i].numExercise << "\t\t" <<
				static_cast<int>(stulist->stuArray[i].gradeAssignment) << "\t\t" <<
				setprecision(1) << fixed << showpoint << stulist->stuArray[i].preciseGrade <<
"\t\t" <<
				stulist->stuArray[i].roundGrade << endl;
		}
	}
	if (isExist == false)
	{
		cout << "Can not find the student!" << endl;
	}

	system("pause");//press any key to continue
	system("cls");	//clear the screen
}
```

```
/*************************************************************************
 *                          settings()                                   *
 *          To allow the user reset the emphasis of scores from          *
            the weekly exercises and final assignment to the final score         *
 *************************************************************************/
void settings(StuList* stulist)
{
    int a;
    cout << "\nPlease choose emphasis of the weekly exercises(1-100): ";

    //check non-numeric input
    bool error;
    do
    {
        error = false;
        cin >> a;//read input of grade for final assignment
        if (cin.fail())
        {
            cout << "Error! Please enter a number between 0-100: ";
            error = true;
            cin.clear();
            cin.ignore(80, '\n');
        }
    } while (error);

    //to limit value between 1-100
    while (a < 0 || a >100)
    {
        cout << "Error! Please enter a number between 1-100: ";
        cin >> a;
    }
    stulist->percentA = a;
    int b = 100 - a;//total percentage is 100%
    stulist->percentB = b;
    cout << "\n\tThe emphasis of the weekly exercises is now set to be " << a << "%." << endl;
    cout << "\n\tThe emphasis of the final assignment is then " << b << " %." << endl;

    system("pause");//press any key to continue
    system("cls");    //clear the screen
}
```