src/functions.cpp

```cpp
// Title  : functions.cpp
// Desc   : Implementation of functions
// Name   : An Tran

#include <iostream>
    using std::cout;
    using std::cin;
    using std::endl;

#include <iomanip>
    using std::setfill;
    using std::setw;
    using std::left;
    using std::right;

#include <fstream>

#include <sstream>

#include "functions.h"

void greeting() {
    cout << "Press Enter to Continue";
    cin.get();
};

int recordCount() {
    std::ifstream file("data/data.txt");
    if (!file.is_open()) {
        std::cerr << "Failed to open the file." << std::endl;
        return -1;
    };
    std::string line;
    int recordCount = 0;
    bool isPreviousLineEmpty = true; // to handle the case where file starts directly with data

    while (getline(file, line)) {
        if (line.empty()) {
            recordCount++;
            isPreviousLineEmpty = true;
        } else {
            isPreviousLineEmpty = false;
        };
    };

    if (!isPreviousLineEmpty) { // if last line read was not empty, increment count for the last record
        recordCount++;
    }

    file.close();
    return recordCount;
};
```

```cpp

void getInput(vector<CollegePerson*> collegeRecords) {;
    std::ifstream file("data/data.txt");
    if (!file.is_open()) {
        std::cerr << "Failed to open the file." << std::endl;
        return;
    };

    std::string line;
    int recordIndex = 0; // keep an index to track which CollegePerson we're updating.

    while (recordIndex < collegeRecords.size() && std::getline(file, line)) {
        if (line.empty()) {
            continue;  // skip empty lines between records
        };

        std::istringstream iss(line);
        string name, university, college;
        int age, id;
        float earned = 0.0, total = 0.0;

        // parsing first line of data for personal info
        getline(iss, name, ',');
        iss >> age;
        iss.ignore();
        getline(iss, university, ',');
        getline(iss, college, ',');
        iss >> id;

        CollegePerson* person = collegeRecords[recordIndex];
        person->setName(name);
        person->setAge(age);
        person->setUniv(university);
        person->setCollege(college);
        person->setID(id);

        // read all grades for this person until an empty line is found.
        while (std::getline(file, line) && !line.empty()) {
            std::istringstream issGrade(line);
            float gradeEarned, gradeTotal;
            string gradeType;
            getline(issGrade, gradeType, ','); // skip to the grade value
            issGrade >> gradeEarned;
            issGrade.ignore();
            issGrade >> gradeTotal;

            earned += gradeEarned;
            total += gradeTotal;
        };

        // set the total earned and total points
        person->setEarned(earned);
        person->setTotal(total);

        recordIndex++;
```

```
108     };
109     file.close();
110 };
111 
112 void calcGrades(vector<CollegePerson*> collegeRecords) {
113     float grade{0.0};
114     for (int i = 0; i < collegeRecords.size(); i++) {
115         if (collegeRecords[i]->getTotal() > 0) { // ensure no division by zero
116             grade = (collegeRecords[i]->getEarned() / collegeRecords[i]->getTotal())
* 100;
117             collegeRecords[i]->setGrade(grade);
118 
119             // apply grading scale
120             if (grade > 94) {
121                 collegeRecords[i]->setLetterGrade("A+");
122                 collegeRecords[i]->setgpa(4.5);
123             } else if (grade > 89) {
124                 collegeRecords[i]->setLetterGrade("A");
125                 collegeRecords[i]->setgpa(4.0);
126             } else if (grade > 84) {
127                 collegeRecords[i]->setLetterGrade("B+");
128                 collegeRecords[i]->setgpa(3.5);
129             } else if (grade > 79) {
130                 collegeRecords[i]->setLetterGrade("B");
131                 collegeRecords[i]->setgpa(3.0);
132             } else if (grade > 74) {
133                 collegeRecords[i]->setLetterGrade("C+");
134                 collegeRecords[i]->setgpa(2.5);
135             } else if (grade > 69) {
136                 collegeRecords[i]->setLetterGrade("C");
137                 collegeRecords[i]->setgpa(2.0);
138             } else if (grade > 64) {
139                 collegeRecords[i]->setLetterGrade("D+");
140                 collegeRecords[i]->setgpa(1.5);
141             } else if (grade > 60) {
142                 collegeRecords[i]->setLetterGrade("D");
143                 collegeRecords[i]->setgpa(1.0);
144             } else {
145                 collegeRecords[i]->setLetterGrade("F");
146                 collegeRecords[i]->setgpa(0);
147             };
148         } else {
149             // If total points are 0
150             collegeRecords[i]->setLetterGrade("F");
151             collegeRecords[i]->setgpa(0);
152         };
153     };
154 };
155 
156 void display(vector<CollegePerson*> collegeRecords) {
157     //display UCD Students
158     cout << "UCD" << endl
159          << left << setw(15) << "Name" << left << setw(15) << "Age" << left <<
setw(15) << "ID"
160          << left << setw(15) << "College" << left << setw(15) << "GPA" << left <<
setw(15) << "Grade" << endl;
161     for(int i = 0; i < collegeRecords.size() ; i++)
```

```
162     {
163         if (collegeRecords[i]->getUniv() == "UCD") {
164             cout << left << setw(15) << collegeRecords[i]->getName() << left <<
setw(15) << collegeRecords[i]->getAge() << left << setw(15) << collegeRecords[i]->
getID()
165                 << left << setw(15) << collegeRecords[i]->getCollege() << left <<
setw(15) << collegeRecords[i]->getgpa() << left << setw(15) << collegeRecords[i]->
getLetterGrade() << endl;
166         };
167     };
168
169     //display Metro State Students
170     cout << "Metro State" << endl
171          << left << setw(15) << "Name" << left << setw(15) << "Age" << left <<
setw(15) << "ID"
172          << left << setw(15) << "College" << left << setw(15) << "GPA" << left <<
setw(15) << "Grade" << endl;
173     for(int i = 0; i < collegeRecords.size() ; i++)
174     {
175         if (collegeRecords[i]->getUniv() == "Metro State") {
176             cout << left << setw(15) << collegeRecords[i]->getName() << left <<
setw(15) << collegeRecords[i]->getAge() << left << setw(15) << collegeRecords[i]->
getID()
177                 << left << setw(15) << collegeRecords[i]->getCollege() << left <<
setw(15) << collegeRecords[i]->getgpa() << left << setw(15) << collegeRecords[i]->
getLetterGrade() << endl;
178         };
179     };
180 };
181
```