Experiment #5

1.

//main.cpp

#include<iostream>

#include<fstream>

#include<cstdlib>

#include<ctime>

#include<iomanip>

#include<cmath>

using namespace std;

int main() {

//Part I

srand(time(0));

ofstream outFile(".\\MyFile\\Record.txt", ios::out);

if (!outFile) {

cerr << "File couldn't be opened.\n";

exit(1);

}

outFile << setfill('0');

for (int i = 0; i < 100; ++i) {

outFile << "No." << i + 1 << ":\t"; //No.x:

for (int j = 0; j < 4; ++j) {

outFile << setw(3) << rand() % 256;

if (j != 3)

outFile << '.';

} //xxx.xxx.xxx.xxx

outFile << ' '

<< 2021 << '-'

<< 05 << '-'

<< setw(2) << rand() % 15 + 1 //2021-05-DD(before 15)

<< ' '

<< setw(2) <<rand() % 24 << ':'

<< setw(2) << rand() % 60 << ':'

<< setw(2) << rand() % 60 //HH:MM:SS

<< endl;

}

//Part II

ifstream inFile(".\\MyFile\\Record.txt", ios::in);

if (!inFile) {

cerr << "File couldn't be opened.\n";

exit(1);

}

int A = 0, B = 0, C = 0, D = 0, E = 0;

char\* temp1 = new char[100]; //to store the data read from the txt

int x = 1;

while (inFile.getline(temp1,100)) {

for (int i = 0; i < 100; ++i) {

if (temp1[i] == '\t') { //to find the address

int temp2 = 0;

for (int j = 0; j < 3; ++j)

temp2 += int(temp1[i + j + 1] - '0') \* pow(10, 2 - j); //convey from char to int

if (temp2 >= 0 && temp2 < 128)

++A;

else if (temp2 >= 128 && temp2 < 192)

++B;

else if (temp2 >= 192 && temp2 < 224)

++C;

else if (temp2 >= 224 && temp2 < 240)

++D;

else if (temp2 >= 240 && temp2 < 247)

++E;

break;

}

}

delete[] temp1;

temp1 = new char[100];

}

delete[] temp1;

cout << "Number of class A address: " << A << endl

<< "Number of class B address: " << B << endl

<< "Number of class C address: " << C << endl

<< "Number of class D address: " << D << endl

<< "Number of class E address: " << E << endl;

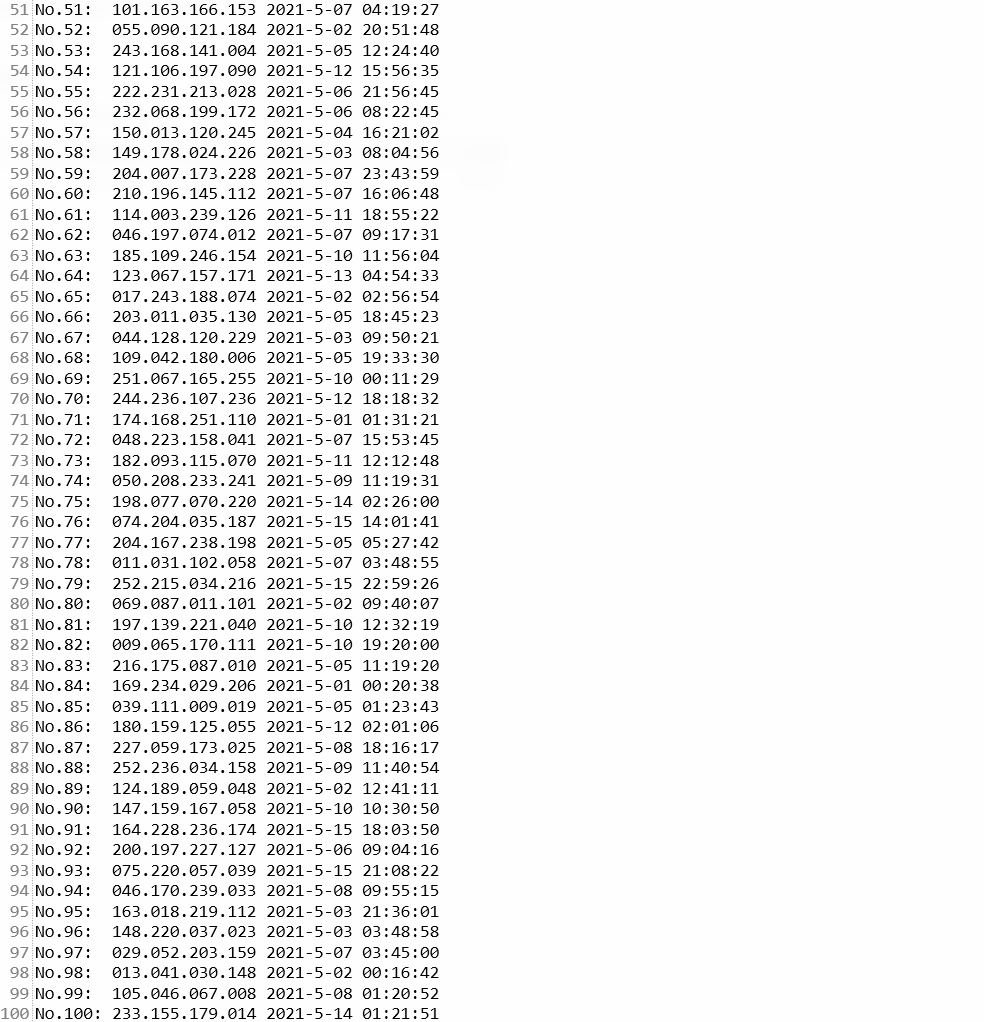
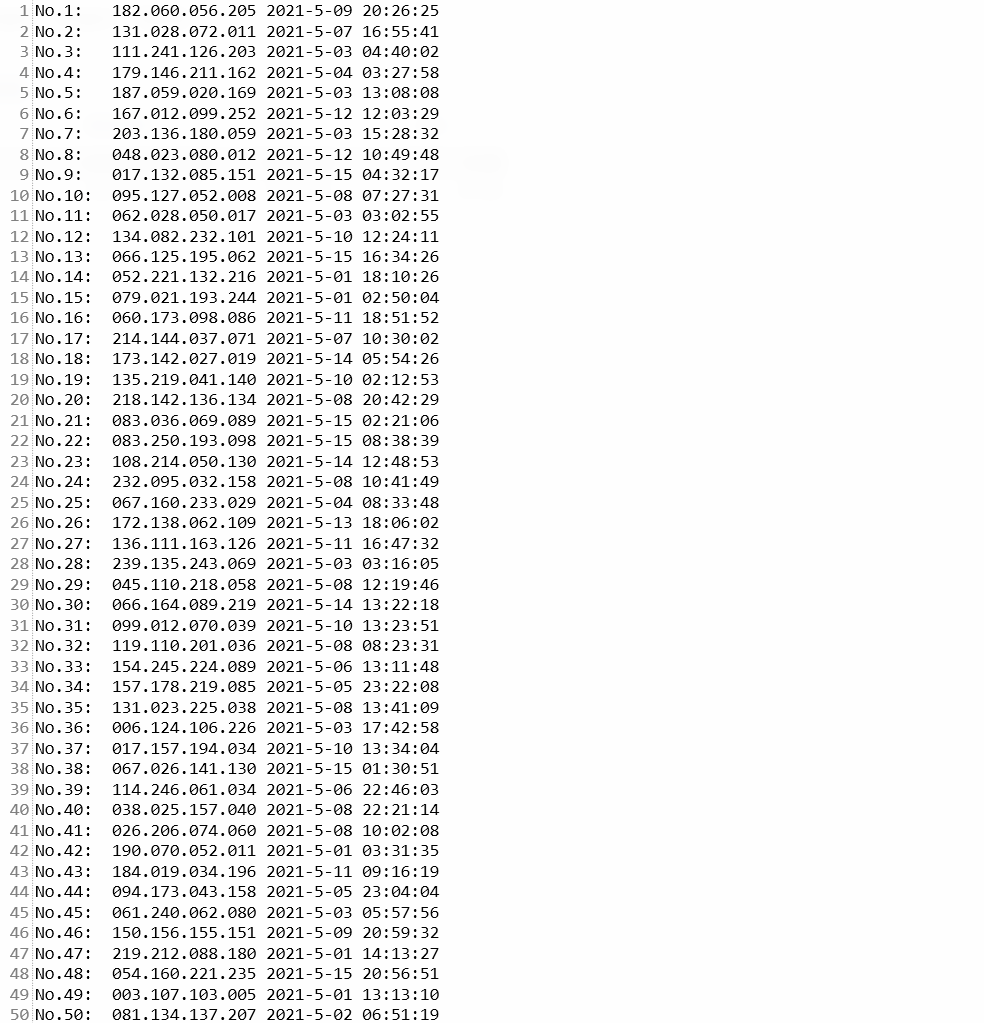
inFile.close();

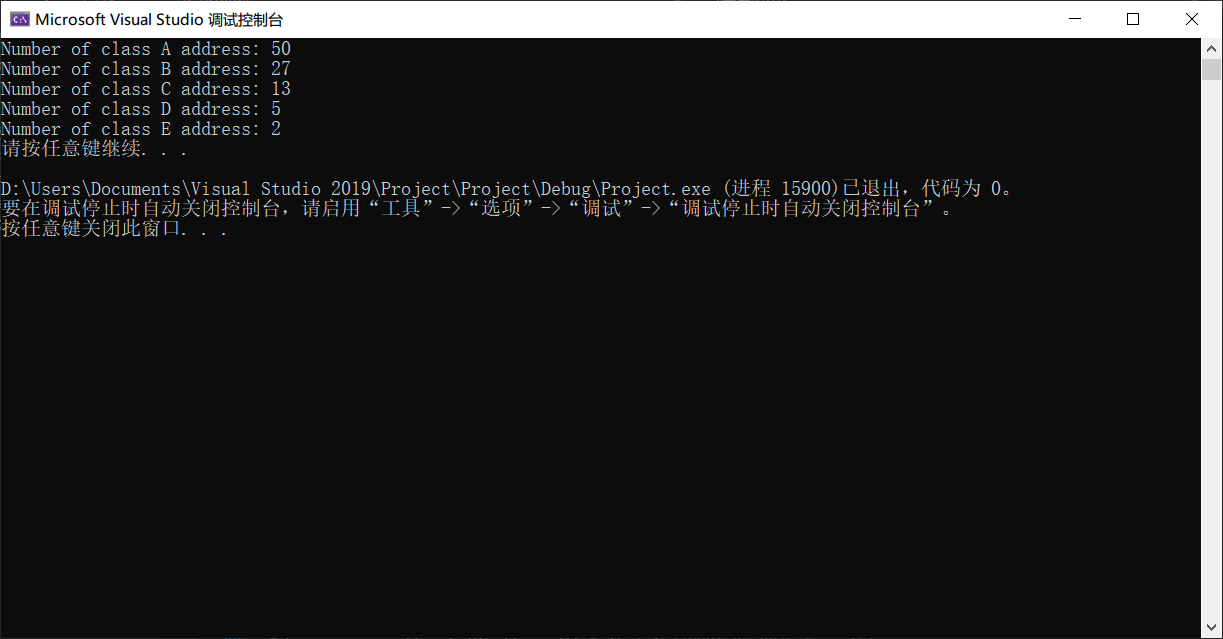
cout << "File closed." << endl;

system("pause");

return 0;

}





2.

//Student.h

#ifndef STUDENT\_H

#define STUDENT\_H

class Student {

public:

Student(string fn = "", string ln = "", double gd = 0) {

setFirstName(fn);

setLastName(ln);

grade = gd;

}

void setFirstName(string fn) {

const char\* fnValue = fn.data();

int length = fn.size();

length = (length < 50 ? length : 49);

strncpy\_s(firstName, fnValue, length);

firstName[length] = '\0';

}

void setLastName(string ln) {

const char\* lnValue = ln.data();

int length = ln.size();

length = (length < 50 ? length : 49);

strncpy\_s(lastName, lnValue, length);

lastName[length] = '\0';

}

void setGrade(double gd) {

grade = gd;

}

string setFirtName() {

return firstName;

}

string getLastName() {

return lastName;

}

double getGrade() {

return grade;

}

bool isEmpty() {

if (firstName == string("") && lastName == string("") && grade == 0)

return 1;

else

return 0;

}

void clearRecord() {

setFirstName("");

setLastName("");

setGrade(0);

}

void display() {

if (isEmpty())

cout << "Empty student record" << endl;

else {

cout << left << setw(15) << firstName

<< setw(15) << lastName

<< setw(5) << grade << endl;

}

}

private:

char firstName[50], lastName[50];

double grade;

};

#endif // !STUDENT\_H

//////////////////////////////////////////////////

//main.cpp

#include<iostream>

#include<fstream>

#include<iomanip>

#include<string>

using namespace std;

#include"Student.h"

#define ADD\_ 1

#define DELETE\_ 2

#define EDIT\_ 3

#define DISPLAY\_ 4

#define EXIT\_ 5

int GetRequest();

void AddStu(fstream&,Student&);

void DeleteStu(fstream&, Student&);

void EditStu(fstream&, Student&);

void DisplayStu(fstream&, Student&);

int main() {

fstream studentFile(".\\MyFile\\Student.dat", ios::in | ios::out | ios::binary);

if (!studentFile) {

cerr << "File couldn't be opened.\n";

exit(1);

}

else

cout << "File opened successfully.\n";

//Initialize the file

Student temp;

for (int i = 0; i < 100; ++i)

studentFile.write(reinterpret\_cast<const char\*>(&temp), sizeof(temp));

//Access the file

int request = GetRequest();

while (request != EXIT\_) {

switch (request) {

case ADD\_:

AddStu(studentFile, temp);

break;

case DELETE\_:

DeleteStu(studentFile, temp);

break;

case EDIT\_:

EditStu(studentFile, temp);

break;

case DISPLAY\_:

DisplayStu(studentFile, temp);

break;

}

request = GetRequest();

}

studentFile.close();

cout << "File closed.\n";

system("pause");

return 0;

}

int GetRequest() {

cout << "\nEnter request:"

<< "\n1 - Add a student's profile"

<< "\n2 - Delete a student's profile"

<< "\n3 - Edit a student's profile"

<< "\n4 - Display a student's (or all students') profile"

<< "\n5 - Exit of run" << endl;

int request = 0;

do {

cout << "\n?";

cin >> request;

} while (request< ADD\_ || request >EXIT\_);

return request;

}

void AddStu(fstream& File, Student& Stu) {

cout << "Enter the ID of the student you want to add (1-100):";

int ID;

check:

cin >> ID;

while (ID < 1 || ID>100)

cin >> ID;

File.seekg((ID - 1) \* sizeof(Stu));

File.read(reinterpret\_cast<char\*>(&Stu), sizeof(Stu));

while (!Stu.isEmpty()) {

cout << "This ID has existed in the list, please enter another ID: ";

goto check;

}

string name;

double grade;

cout << endl;

cout << "First Name: ";

cin >> name;

Stu.setFirstName(name);

cout << "Last Name: ";

cin >> name;

Stu.setLastName(name);

cout << "Grade: ";

cin >> grade;

Stu.setGrade(grade);

File.seekp((ID - 1) \* sizeof(Stu));

File.write(reinterpret\_cast<const char\*>(&Stu), sizeof(Stu));

}

void DeleteStu(fstream& File, Student& Stu) {

cout << "Enter the ID of the student you want to delete (1-100):";

int ID;

cin >> ID;

while (ID < 1 || ID>100)

cin >> ID;

File.seekg((ID - 1) \* sizeof(Stu));

File.read(reinterpret\_cast<char\*>(&Stu), sizeof(Stu));

cout << endl;

if (!Stu.isEmpty()) {

Stu.clearRecord();

File.seekp((ID - 1) \* sizeof(Stu));

File.write(reinterpret\_cast<const char\*>(&Stu), sizeof(Stu));

}

cout << "Profile deleted.\n";

}

void EditStu(fstream& File, Student& Stu) {

cout << "Enter the ID of the student you want to edit (1-100):";

int ID;

cin >> ID;

while (ID < 1 || ID>100)

cin >> ID;

File.seekg((ID - 1) \* sizeof(Stu));

File.read(reinterpret\_cast<char\*>(&Stu), sizeof(Stu));

cout << endl;

cout << "Original student profile:\n";

Stu.display();

string name;

double grade;

cout << endl;

cout << "New first name: ";

cin >> name;

Stu.setFirstName(name);

cout << "New last name: ";

cin >> name;

Stu.setLastName(name);

cout << "New grade: ";

cin >> grade;

Stu.setGrade(grade);

File.seekp((ID - 1) \* sizeof(Stu));

File.write(reinterpret\_cast<const char\*>(&Stu), sizeof(Stu));

}

void DisplayStu(fstream& File,Student& Stu) {

cout << "a - Display certain student's profile.\n"

<< "b - Display all students' profiles.\n";

char choice;

cin >> choice;

cout << endl;

switch (choice) {

case 'a':

cout << "Enter the ID of the student you want to display (1-100):";

int ID;

cin >> ID;

while (ID < 1 || ID>100)

cin >> ID;

File.seekg((ID - 1) \* sizeof(Stu));

File.read(reinterpret\_cast<char\*>(&Stu), sizeof(Stu));

cout << "Original student profile:\n";

Stu.display();

break;

case 'b':

for (int i = 0; i < 100; ++i) {

File.seekg(i \* sizeof(Stu));

File.read(reinterpret\_cast<char\*>(&Stu), sizeof(Stu));

if (!Stu.isEmpty()) {

cout << "ID: " << i+1 << '\t';

Stu.display();

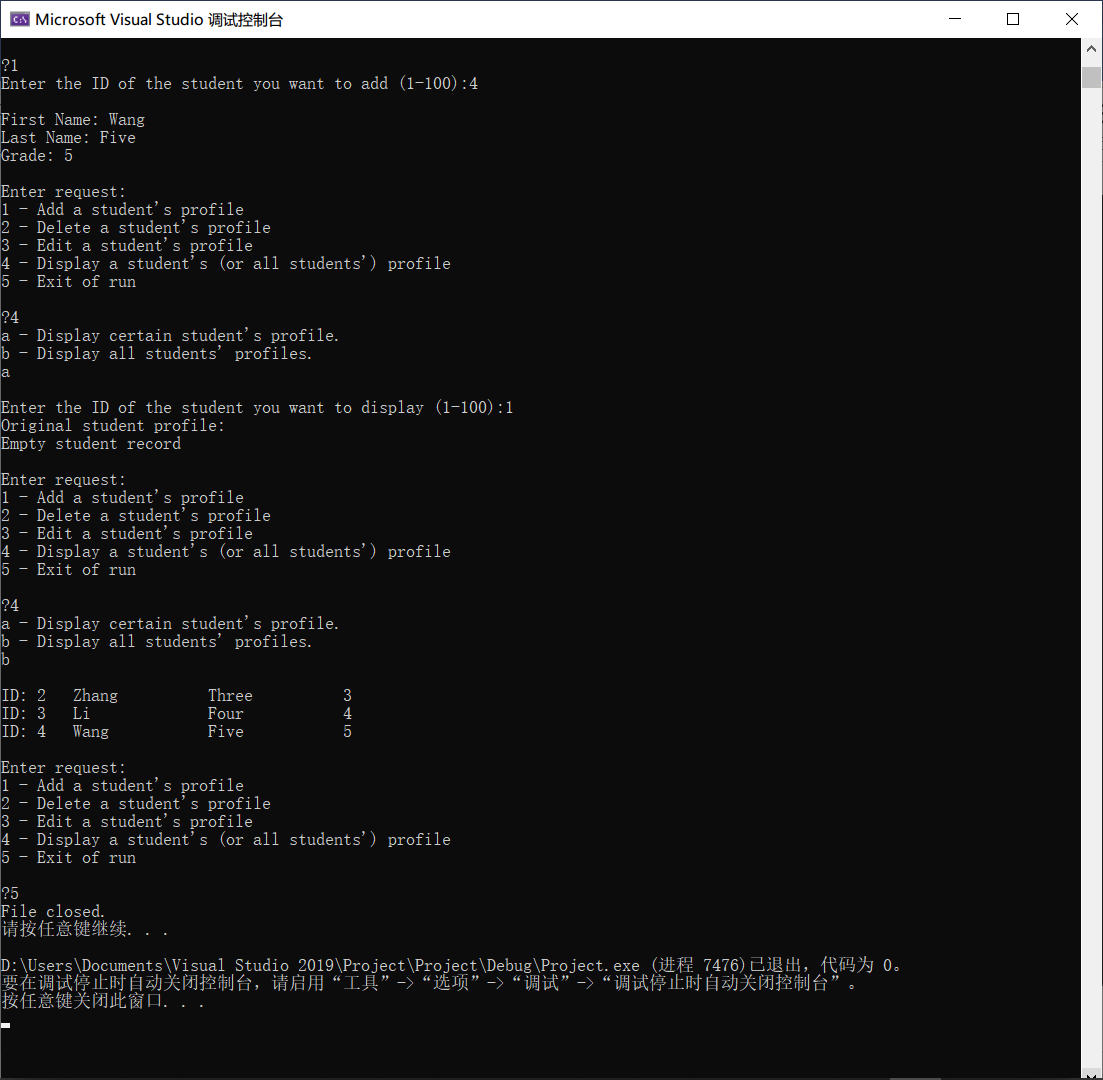
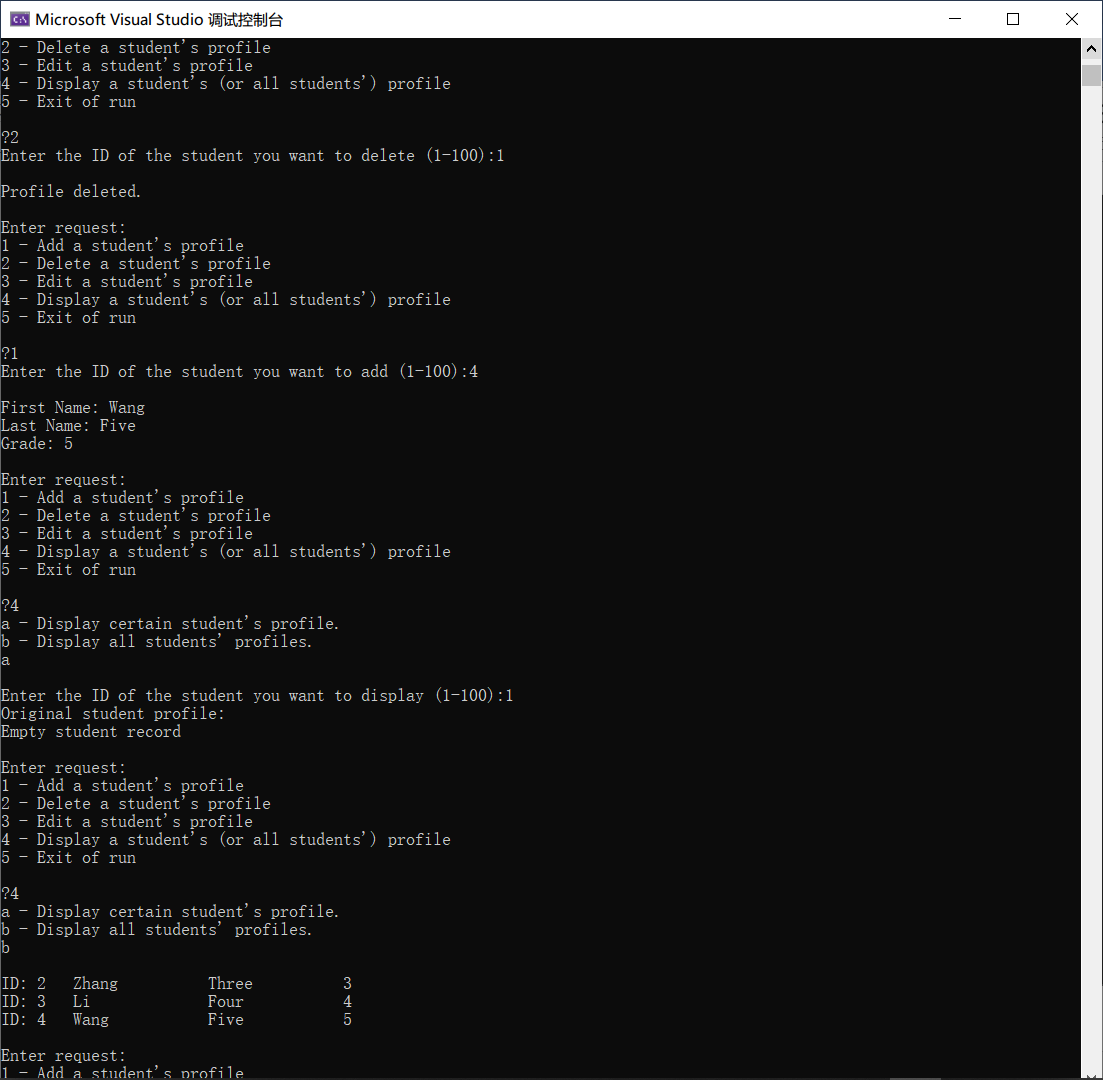
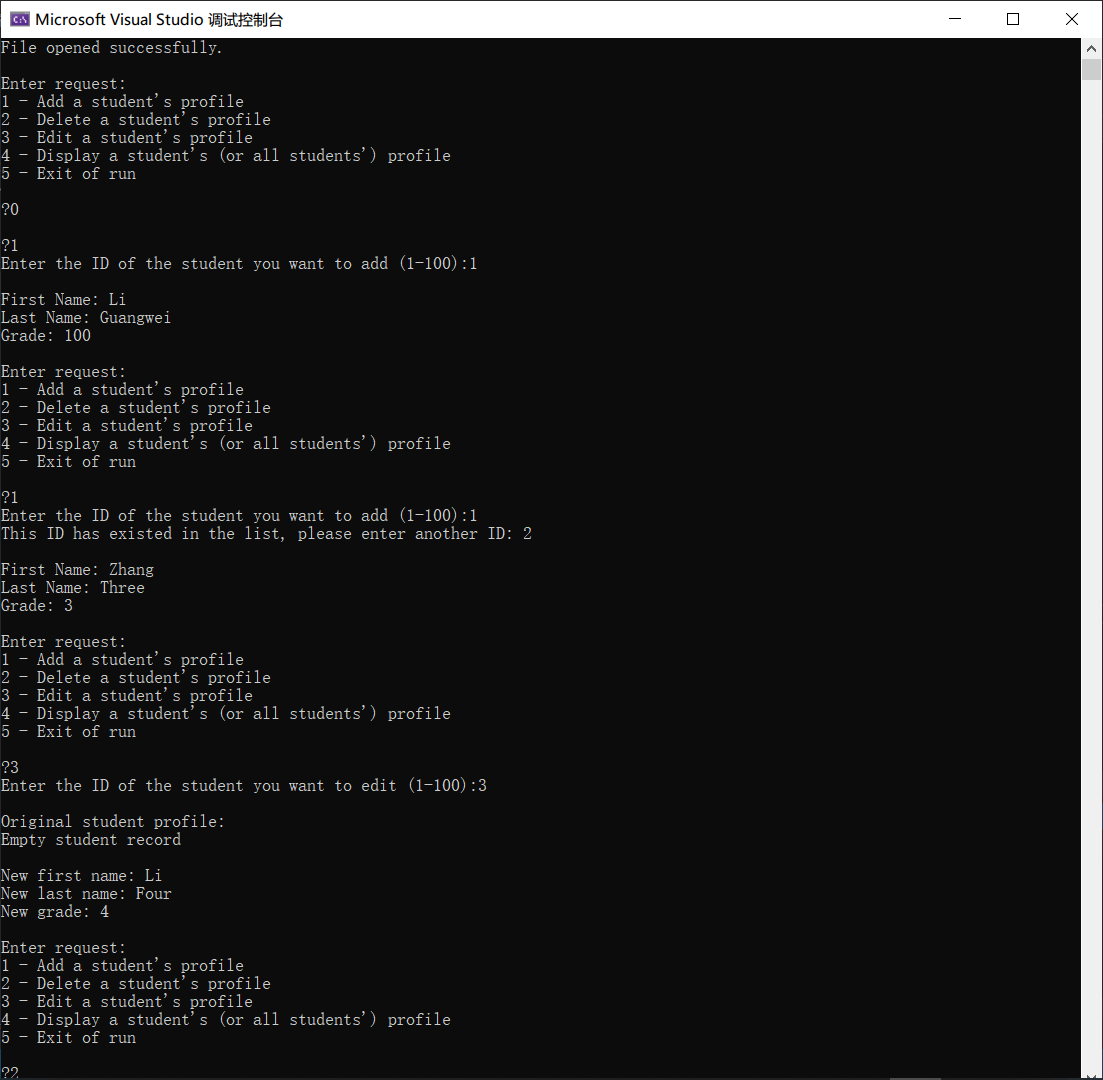
}

}

break;

}

}



调试时已经测试过了所有的功能，均正常。