课 程 设 计 报 告

**课程名称 计算机程序设计基础2**

**班 级 ：无98**

**学 号：2019080030**

**姓 名：林宝诚**

**2020年9月13日**

**学生考勤管理系统**

目 录

[1. 系统需求分析 3](file:///C:\Users\liew\Downloads\2020程设2大作业任务书.docx#_Toc171224774)

[2. 总体设计 3~4](file:///C:\Users\liew\Downloads\2020程设2大作业任务书.docx#_Toc171224775)

[3. 详细设计 5~9](file:///C:\Users\liew\Downloads\2020程设2大作业任务书.docx#_Toc171224776)

[4. 系统调试 10](file:///C:\Users\liew\Downloads\2020程设2大作业任务书.docx#_Toc171224777)

[5. 测试结果 11~14](file:///C:\Users\liew\Downloads\2020程设2大作业任务书.docx#_Toc171224778)

[6. 系统说明书与结果分析 15~16](file:///C:\Users\liew\Downloads\2020程设2大作业任务书.docx#_Toc171224778)

[7. 总结 16](file:///C:\Users\liew\Downloads\2020程设2大作业任务书.docx#_Toc171224779)

附录1：源程序清单

**系统需求分析**

成绩管理系统记录了学生的成绩情况，其中包括了学生的学号、姓名、各科成绩、总成绩（总平均）和成绩分布。另外，为了将程序量控制在合理范围内，此系统是针对一个小型班级来设计的。根据以上需求，系统提供了以下功能（详细操作指示在后边说明书部分）：

1. 输入学生成绩（仅能由主任操作）：主任负责建立一个新学生的档案并输入所有成绩。
2. 修改学生的成绩（仅能由老师操作）：即对成绩的任意部分进行修改，并显示修改结果
3. 查询成绩：此功能根据使用者的身份略有不同。对于学生，学生能够查询自己的各科成绩和总成绩。对于教师，科任教师能够查询自己执教科目的学生成绩，而主任则能够查询任一学生的成绩和显示所有学生的成绩。
4. 统计成绩：系统会为科任老师显示平均分和学生成绩分布（以表的形式）。此外，主任也能根据学生的总成绩显示出最终的排名。
5. 系统以身份识别和菜单方式运行：根据身份的不同，菜单的显示有别（以避免功能混乱），且操作简单，因此能够确保系统运行正常的同时也使用户感到便捷。

**总体设计**

学生成绩管理系统总共包含了四大功能，分别为身份识别、输入、修改、查询和统计成绩。首先，用户需要使用学号和密码登录系统。系统进行验证后会根据用户的身份显示特定的菜单首页。接着，用户只需根据菜单上的指示进行操作即可。

**学生：**

若选择进行成绩查询，系统会直接显示学生的姓名、学号、各科成绩和总平均。

**科任教师：**

若选择进行成绩查询，系统会自动根据老师的任课科目显示所有学生的科目成绩，并计算出平均分。老师也能够选择是否显示出学生成绩分布。

若选择进行成绩修改，教师需要输入学生的学号和修改后的分数，修改成功后，系统会显示学生科目成绩以供检查。

**主任：**

若选择进行成绩查询，主任需要输入对象学生的学号（如果要查询任一学生）或者直接让系统显示出所有学生的成绩（系统也会学生的总平均进行排名）。

若选择进行成绩录入，主人需要输入学生学号、姓名和各科成绩。录入成功后，系统会出现相应的提示。

\*任何操作完成后，系统会自动将界面返回主菜单。若想登出，用户只需在菜单上选择退出选项即可。

**学生成绩管理系统模块图：**

图【1】

查询科目成绩

查询个别学生或全体学生成绩

录入新学生的成绩档案

查询科目成绩

科任教师

主任

学生

查询成绩

修改科目成绩

登入&身份识别

学生成绩管理系统

**详细设计**

**成绩管理系统的类层次图：**

basicUser类（抽象）

student类

teacher类

headTeacher类

图【2】 学生成绩管理系统中五个类的类层次图

**学生考勤管理系统中各功能模块的实现：**

输入学号（教师证件号）和密码

登入

&

识别身份

利用文件读写方式核实证号和密码

显示符合身份的菜单

图【3】学生成绩管理系统中登入函数的功能图

学生菜单

登出

查询成绩

图【4】学生菜单

查询成绩

显示学生基本资料（姓名、学号）

显示各科成绩和总平均

返回菜单

图【5】学生查询成绩功能模块

科任教师菜单

查询科目成绩

登出

修改科目成绩

图【6】科任教师菜单

显示科任教师基本资料（姓名、证号、任课科目）

查询科目成绩

显示任课科目的学生成绩和平均分

显示学生成绩分布（可选）

返回菜单

图【7】科任教师查询成绩功能模块

输入学生学号和新成绩

修改科目成绩

更新文件中学生成绩

显示修改后任课科目的学生成绩和平均分

返回菜单

图【8】科任教师修改成绩功能模块

主任菜单

查询科目成绩

添加新学生成绩

查询所有学生成绩

登出

查询任意学生成绩

图【9】主任菜单

输入学生学号

查询任意学生成绩

显示主任基本资料（姓名、证号）

显示学生各科成绩和总平均

返回菜单

图【10】主任查询任意学生成绩功能模块

显示主任基本资料（姓名、证号）

查询所有学生成绩

显示所有学生各科成绩和总平均

显示学生排名（含学号、总平均）

返回菜单

图【11】主任查询所有学生成绩功能模块

输入科目、显示主任基本资料（姓名、证号）

查询科目成绩

显示科目的学生成绩和平均分

显示学生成绩分布（可选）

返回菜单

图【12】主任查询成绩功能模块

添加

新学生

成绩

输入新学生学号、姓名、各科成绩

将新学生所有资料添加到文件中

返回菜单

图【13】主任添加新学生成绩功能模块

**basicUser**

-id:string

-username:string

-job:string

-fileInput:bool

+basicUser(userid: string)

+DisplayMainMenu()=0:virtual void

+chooseOption():int

**teacher**

-subjectName:string

-dist[10]:int

- totalStu:int

- subjectResult: vector<tuple<string, string, float>>

+teacher(teacherId:string)

+ DisplayMainMenu():void

+ Option(option:int):virtual void

+loadSubjectResult(subjectName:string,countDistribution:bool):void

+calculateMean(numStu:int):float

+DisplaySubjectResult(countDistribution:bool):void

**headTeacher**

+headTeacher(teacherId:string) +addResult():void

+DisplayResult(studentId:string):void +DisplayResult():void

+ Option(option:int):virtual void + DisplayMainMenu():void

+prepareRanking(studentId[]:string):void

**student**

-classNum:string

-result:vector<float>

+student(studentId:string)

+DisplayMainMenu():void

+Option(option:int):void

+loadResult():void

+DisplayResult():void

+calculateAvg(result:vector<float>):float

+operator <(A:student):bool

基类

派生类

派生类

派生类

图【14】学生考勤管理系统中四个类的UML图

**系统调试**

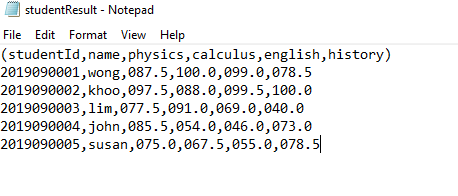
调试过程中，最令我印象深刻且较严重的问题是数组越界和string忘了重置。其中，前者是发生在学生科目成绩分布的统计。当时的实际情况为我忘记将100分考虑在内，因此在100/10=10后（基本上想法是把分数除10以10分为单位进行分类），dist[10]的数组发生了越界问题。解决方案是利用一个if函数对100分进行特殊处理（其实就是除10后减去1）。另外，string忘了重置的问题发生了较多次（毕竟是第一次使用string 类，有些不熟悉），此问题使我的数据变得混乱，有时甚至会造成程序死机。不过，解决方式较为简单，就是使用cout将数据输出进行检查即可发现问题所在，之后在相应位子补上（string变量名）.clear()即可（反正之后就是要养成将需要重复使用的string清空的习惯）。

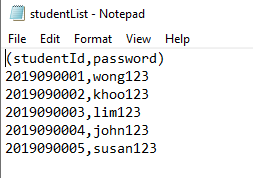
另一个使我较头痛的部分是科任老师修改学生成绩函数。此部分主要操作是根据学生证号和科目寻找相应位置进行修改。为了能够找到准确的位置，我将各科分数所占空间都设定成一致（XXX.X），并利用seekg函数找寻位置将新成绩覆盖掉旧成绩。这部分我尝试了许久才成功，毕竟之前不太理解要怎么覆盖旧数据，后来发现到使用seekg（-位置）（参照代码）就能成功执行。

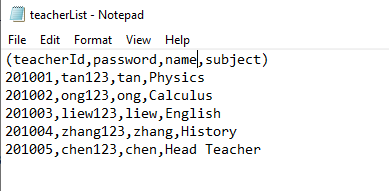
此外，为了使输出显得更加整齐和美观，我在显示方面也耗费了不少时间。调试此部分让我领悟的一个重要道理是在使用完setiosflags后要立马resetiosflags，否则会影响之后系统的输出。

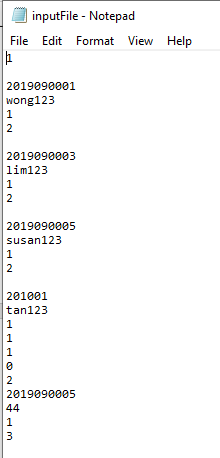
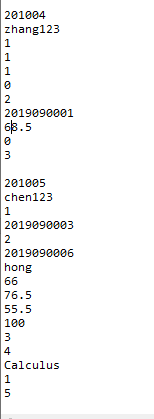
事实上，我的调试过程还算顺利，除去上述的错误和一些一开始被遗漏的部分（如最初我忘了分数是浮点数的可能性）外，我的代码逻辑部分并未有太大的问题。或许这与我在设计较复杂或较不熟练的函数时会额外建立一个新源文件先行尝试的习惯有关。在我看来，这举动其实帮助我避免在代码量较多的情况下产生混乱，进而减少我调试所需的时间。

**测试结果**

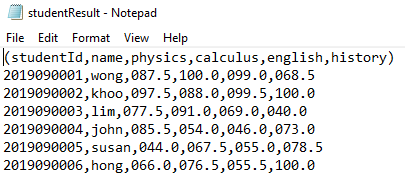
此学生成绩管理系统的测试数据为：

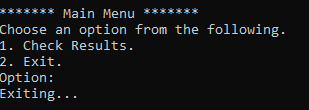
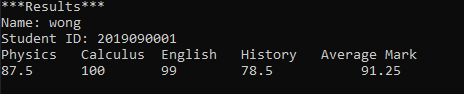
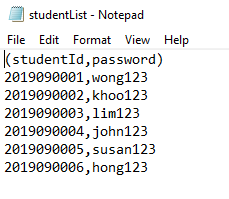


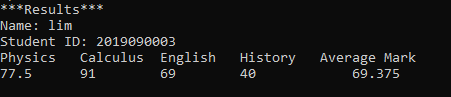
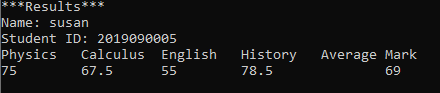


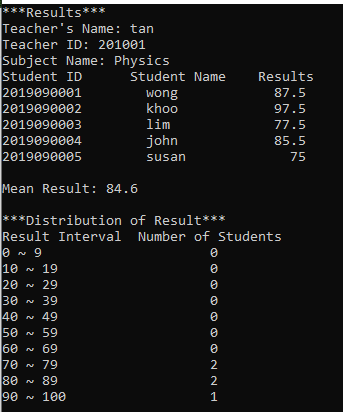
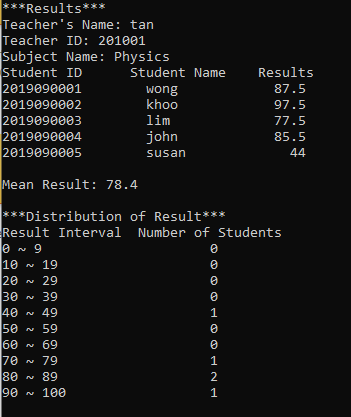


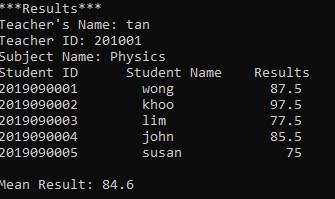
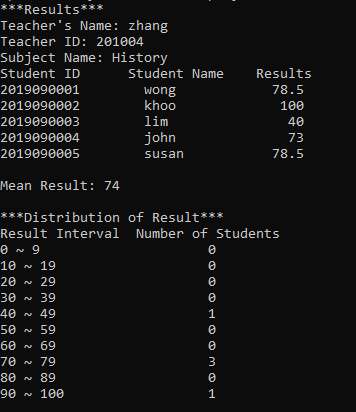
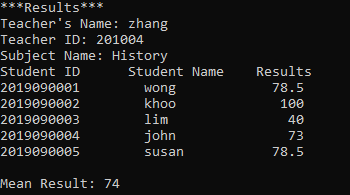
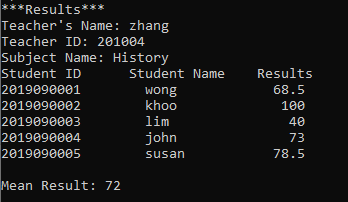
测试结果的截图如下。

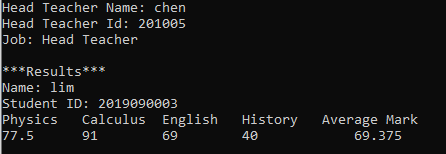


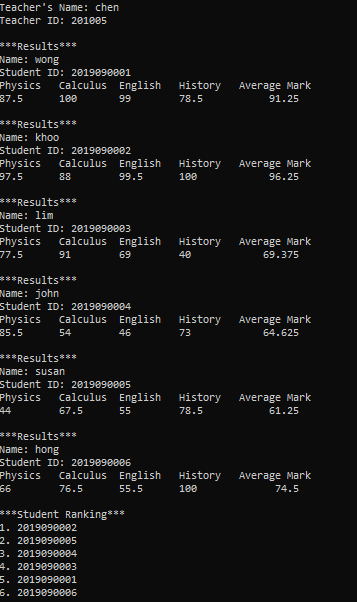
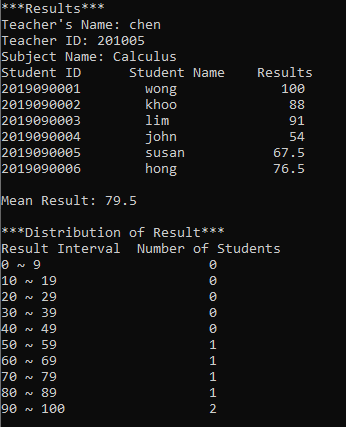












**系统使用说明书与测试结果分析**

**系统使用说明书**

首先，需要准备一个教师资料的文件（上述测试文件的teacherList图片），其中必须包含一个“Head Teacher”以添加学生成绩。另外，也需准备studentList和studentResult文件依照上述图片输入各自的第一行备注（可不输入任何学生资料）。图片中的inputFile文件为输入所用文件。下面将介绍inputFile的书写格式（其实更建议使用控制台并参照菜单和指示）。

|  |  |
| --- | --- |
| 第一行 | 1：使用文件输入 0：使用控制台输入 |
| 第二行 | 留空 |
| 第三行 | 输入证号 |
| 第四行 | 输入密码 |

不同身分接下来的输入方式有别

**学生**

|  |  |
| --- | --- |
| 接下来的每一行 | 1：显示成绩 2：退出 |
| 若选择2，则需重新会到登录页面（需要输入证号和输入密码） | |

**科任老师**

|  |
| --- |
| 接下来的每一行 |
| 查询成绩：1 🡪 显示成绩分布：1 不显示成绩分布：0 |
| 更改成绩：2 🡪 输入学生证号 🡪 输入新分数 🡪 显示成绩分布：1 不显示成绩分布：0 |
| 退出：3 |
| 若选择3（退出），则需重新会到登录页面（需要从输入证号和输入密码那里开始） |

**主任**

|  |
| --- |
| 接下来的每一行 |
| 查询任意学生成绩：1 🡪 输入学生证号 |
| 添加新学生：2 🡪 输入学生证号 🡪 输入名字 🡪 输入四科科目成绩 |
| 显示所有学生成绩：3 |
| 查询任意科目成绩：4 🡪 输入科目名称（第一个字母大写，其他小写）🡪 显示成绩分布：1 不显示成绩分布：0 |
| 退出：5 |
| 若选择5（退出），则需重新会到登录页面（需要从输入证号和输入密码那里开始） |

**测试结果分析**

首先，测试文件分别以2019090001,2019090003,2019090005三位学生的证号进行登录，并显示成绩。接着文件以201001和201004老师的证号进行登录，并进行科目成绩显示和修改成绩的操作。修改成绩的部分可从2019090005学生的物理（physics）成绩由75变成44，2019090001学生的历史（history）成绩由78.5变成68.5.最后，测试文件再以201005主任的证号进行登录，并进行查询2019090003学生成绩、添加新学生2019090006、查询微积分（Calculus）成绩和显示所有成绩这四个操作。

**总结**

在这次的设计中，我尽力将我所有学过的知识和方法运用在程序中。另外，我也在可扩展性方面下了些功夫。虽然我调试所用的数据较为简单，但是代码本身可以处理任意的数据量（基本上就是善用动态数组和少用固定的变量），所以即使学生人数增多，系统也能顺利运行。由于工作量原因，我需要做一些假设如固定的科目、学生每科都得有成绩、老师的数量等等。因此，此次的设计还能进行改进和扩展。值得一提的事，一开始进行编程时，我经常更换类的安排，也曾尝试许多不合理的设计方案，耗费了不少时间。在摸索了一番后，我才决定以上这个最终设计。无可否认，这次大作业让我在类的规划和使用上更加熟练（当然还是有很多进步空间）。总结地说，这次的大作业让我对面向对象编程有更深的认识，当然，今后我也会不断努力加强自己的编程技术，以配合科技发达时代的需求。

**附录1：源代码**

#include <iostream>

#include<string>

#include<fstream>

#include<vector>

#include<algorithm>

#include<tuple>

#include<iterator>

#include<iomanip>

using namespace std;

//设定全局变量

//用户证号、班级、科目、科目数量、测试文件（若使用文件输入）

string idGlobal;

string classNameGlobal = " Class 98";

string allSubjects[4] = { "Physics","Calculus","English","History" };

int subjectNum = sizeof(allSubjects) / sizeof(allSubjects[0]);

ifstream inputFile("D:/inputFile.txt");

//抽象类

class basicUser {

public:

basicUser(string userid = idGlobal, bool x=true);

virtual void DisplayMainMenu() = 0;

int chooseOption();

protected:

//证号、用户名、职务（学生或老师）、文件输入（判定是否）

string id;

string username;

string job;

bool fileInput;

};

//构造函数

basicUser::basicUser(string userid, bool x) {

id = userid;

fileInput = x;

}

//读用户选择函数

int basicUser::chooseOption() {

int option;

cout << "Option: ";

if (fileInput) {

string temp;

getline(inputFile, temp);

option = stoi(temp);

}

else

cin >> option;

return option;

}

//学生类

class student : public basicUser {

public:

student(string studentId = idGlobal, bool x = true);

void DisplayMainMenu();

bool Option(int option);

void loadResult();

void DisplayResult();

float calculateAvg(vector<float> result);

bool operator <(student A);

private:

//学生班级和个人成绩

string classNum;

vector<float> result;

};

//学生构造函数

student::student(string studentId,bool x) :basicUser(studentId,x) {

job = "student";

classNum = classNameGlobal;

}

//学生版显示主页函数

void student::DisplayMainMenu() {

cout << endl;

cout << "\*\*\*\*\*\*\* Main Menu \*\*\*\*\*\*\*" << endl;

cout << "Choose an option from the following." << endl;

cout << "1. Check Results." << endl;

cout << "2. Exit." << endl;

}

//学生选项函数

bool student::Option(int option) {

switch (option) {

case 1:

loadResult();

DisplayResult();

break;

case 2:

cout << endl;

cout << "Exiting..." << endl;

return 1;

default:

cout << "Enter a valid option!" << endl;

}

return 0;

}

//从文件将学生成绩读入属性result(根据对象的证号）

void student::loadResult() {

ifstream studentResult;

string temp, idTemp, mark;

string::iterator it;

studentResult.open("D:/studentResult.txt");

if (studentResult.is\_open()) {

getline(studentResult, temp);

while (studentResult.good()) {

getline(studentResult, temp);

for (it = temp.begin(); \*it != ','; it++)

idTemp.push\_back(\*it);

if (id.compare(idTemp) == 0)

{

for (it = temp.begin() + 11; \*it != ','; it++) {

username.push\_back(\*it);

}

for (int i = 0; i < subjectNum - 1; i++) {

for (it = it + 1; \*it != ','; it++) {

mark.push\_back(\*it);

}

result.push\_back(stof(mark));

mark.clear();

}

for (it = it + 1; it != temp.end(); it++) {

mark.push\_back(\*it);

}

result.push\_back(stof(mark));

}

idTemp.clear();

}

}

studentResult.close();

}

//显示学生成绩（根据属性中的result）

void student::DisplayResult() {

cout << endl;

cout << "\*\*\*Results\*\*\*" << endl << "Name: " << username << endl << "Student ID: " << id << endl;

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

cout << setw(10) << setiosflags(ios\_base::left) << allSubjects[i];

}

cout << setw(10) << "Average Mark" << endl;

for (auto itr = result.begin(); itr != result.end(); itr++) {

cout << setw(10) << setiosflags(ios\_base::left) << \*itr;

cout << resetiosflags(ios\_base::left);

}

cout << setw(10) << calculateAvg(result) << endl;

}

//两位学生的总平均比较（在主任显示排名时会用到）

bool student::operator<(student A) {

this->loadResult();

A.loadResult();

if (calculateAvg(this->result) < calculateAvg(A.result))

return 1;

else

return 0;

}

//计算学生总平均

float student::calculateAvg(vector<float> result) {

float total = 0;

for (auto itr = result.begin(); itr != result.end(); itr++) {

total += \*itr;

}

return total / size(result);

}

//老师类（科任教师）

class teacher :public basicUser {

public:

teacher(string teacherId = idGlobal,bool x =true);

void DisplayMainMenu();

virtual bool Option(int option);

void loadSubjectResult(string subjectName, bool countDistribution);

float calculateMean(int numStu);

void DisplaySubjectResult(bool countDistribution);

protected:

//成绩统计数组、学生数量、科目名称、科目成绩

int dist[10];

int totalStu;

string subjectName;

vector<tuple<string, string, float>> subjectResult;

private:

void initialize();

void edit(string subject);

};

//老师构造函数

teacher::teacher(string teacherId, bool x) :basicUser(teacherId, x) {

job = "teacher";

initialize();

}

//老师初始化函数（用以计算学生数量和根据属性id确定老师名字和任课科目）

void teacher::initialize() {

//count number of student

string temp, idTemp;

ifstream studentResult, teacherList;

string::iterator it;

int lineCount = 0;

studentResult.open("D:/studentResult.txt");

if (studentResult.is\_open()) {

while (studentResult.good()) {

getline(studentResult, temp);

lineCount++;

temp.clear();

}

}

studentResult.close();

totalStu = lineCount - 1;

teacherList.open("D:/teacherList.txt");

if (teacherList.is\_open()) {

getline(teacherList, temp);

while (teacherList.good()) {

temp.clear();

getline(teacherList, temp);

for (it = temp.begin(); \*it != ','; it++)

idTemp.push\_back(\*it);

if (id.compare(idTemp) == 0)

{

it++;

for (int i = 0; i < 1; i++) {

while (\*it != ',') {

it++;

}

it++;

}

for (; \*it != ','; it++)

username.push\_back(\*it);

it++;

for (; it != temp.end(); it++)

subjectName.push\_back(\*it);

break;

}

idTemp.clear();

}

}

teacherList.close();

}

//老师版显示主页函数

void teacher::DisplayMainMenu() {

cout << endl;

cout << "\*\*\*\*\*\*\* Main Menu \*\*\*\*\*\*\*" << endl;

cout << "Choose an option from the following." << endl;

cout << "1. Check Results." << endl;

cout << "2. Edit Results." << endl;

cout << "3. Exit." << endl;

}

//老师版选择函数

bool teacher::Option(int option) {

int flag;

string str;

switch (option) {

case 1:

//询问是否显示学生成绩分布

cout << "Do you want to display distribution (0->No, 1->Yes)? : ";

if (fileInput) {

getline(inputFile, str);

flag = stoi(str);

}

else

cin >> flag;

loadSubjectResult(subjectName, flag);

DisplaySubjectResult(flag);

break;

case 2:

edit(subjectName);

cout << "Do you want to display distribution (0->No, 1->Yes)? : ";

if (fileInput) {

getline(inputFile, str);

flag = stoi(str);

}

else

cin >> flag;

loadSubjectResult(subjectName, flag);

DisplaySubjectResult(flag);

break;

case 3:

cout << endl;

cout << "Exiting..." << endl;

return 1;

default:

cout << "Enter a valid option!" << endl;

}

return 0;

}

//将科目成绩录入属性subjectResult

void teacher::loadSubjectResult(string subjectName, bool countDistribution) {

ifstream studentResult;

string temp, idTemp, mark, nameTemp;

string::iterator it;

int position, count, i = 0;

for (int j = 0; j < 10; j++)

dist[j] = 0;

subjectResult.clear();

studentResult.open("D:/studentResult.txt");

for (position = 0; position < subjectNum; position++) {

if (subjectName.compare(allSubjects[position]) == 0)

break;

}

if (studentResult.is\_open()) {

getline(studentResult, temp);

while (studentResult.good()) {

count = 0;

temp.clear();

getline(studentResult, temp);

//cout << temp << endl;

for (it = temp.begin(); \*it != ','; it++)

idTemp.push\_back(\*it);

for (it = temp.begin() + 11; \*it != ','; it++) {

nameTemp.push\_back(\*it);

}

for (it = temp.begin(); it != temp.end(); it++) {

if (\*it == ',') {

count++;

}

if (position == count - 2) {

it++;

if (position == subjectNum - 1) {

for (; it != temp.end(); it++) {

mark.push\_back(\*it);

}

}

else {

for (; \*it != ','; it++) {

mark.push\_back(\*it);

}

}

break;

}

}

//若选择显示成绩分布

if (countDistribution) {

int index = int(stof(mark) / 10);

if (index == 10)

index -= 1;

dist[index]++;

}

subjectResult.push\_back(make\_tuple(idTemp, nameTemp, stof(mark)));

mark.clear();

idTemp.clear();

nameTemp.clear();

}

}

studentResult.close();

}

//计算科目平均分

float teacher::calculateMean(int numStu) {

float total = 0;

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

total += get<2>(subjectResult[i]);

}

return total / numStu;

}

//显示科目成绩

void teacher::DisplaySubjectResult(bool countDistribution) {

cout << endl;

cout << "\*\*\*Results\*\*\*" << endl << "Teacher's Name: " << username << endl << "Teacher ID: " << id << endl;

cout << "Subject Name: " << subjectName << endl;

cout << "Student ID" << '\t' << "Student Name" << '\t' << "Results" << endl;

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

cout << get<0>(subjectResult[i]) << "\t " << get<1>(subjectResult[i]) <<'\t'<< setiosflags(ios\_base::right) << setw(14) << get<2>(subjectResult[i]) << resetiosflags(ios\_base::right) << endl;

}

cout << endl << "Mean Result: " << calculateMean(totalStu) << endl;

if (countDistribution) {

cout << endl << "\*\*\*Distribution of Result\*\*\*" << endl;

cout << "Result Interval" << setw(20) << "Number of Students" << endl;

for (int i = 0; i < 10; i++) {

if (i == 9)

cout << i \* 10 << " ~ " << i \* 10 + 10 << setw(19) << dist[i] << endl;

else if (i == 0)

cout << i \* 10 << " ~ " << i \* 10 + 9 << setw(22) << dist[i] << endl;

else

cout << i \* 10 << " ~ " << i \* 10 + 9 << setw(20) << dist[i] << endl;

}

}

}

//修改成绩函数

void teacher::edit(string subject) {

string editId;

cout << "Enter student Id to edit result: ";

if (fileInput) {

getline(inputFile, editId);

}

else

cin >> editId;

fstream stu;

int position, count = 0, flag = 0, row = 0;

char c;

string temp, idTemp;

string::iterator it;

for (position = 0; position < subjectNum; position++) {

if (subject.compare(allSubjects[position]) == 0)

break;

}

stu.open("D:/studentResult.txt", ios\_base::in | ios\_base::out);

if (stu.is\_open()) {

getline(stu, temp);

while (stu.good()) {

getline(stu, temp);

row++;

for (it = temp.begin(); \*it != ','; it++)

idTemp.push\_back(\*it);

if (editId.compare(idTemp) == 0)

{

flag = 1;

break;

}

idTemp.clear();

}

if (flag == 0) {

cout << "Student Id is wrong. Edit failed" << endl;

return;

}

stu.seekg(0, ios\_base::beg);

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

getline(stu, temp);

}

while (stu.get(c)) {

if (c == ',')

count++;

if (count == position + 2)

break;

}

float a;

cout << "Enter new result of " << subject << " : ";

if (fileInput) {

string str;

getline(inputFile, str);

a = stof(str);

}

else

cin >> a;

stu.seekg(5, ios\_base::cur);

stu.seekg(-5, ios\_base::cur);

stu << setfill('0') << setw(5) << setiosflags(ios\_base::fixed) << setprecision(1) << a;

cout << resetiosflags(ios\_base::fixed);

}

stu.close();

}

//主任类

class headTeacher :public teacher {

public:

headTeacher(string teacherId = idGlobal, bool x = true);

void DisplayMainMenu();

virtual bool Option(int option);

private:

void addResult();

void DisplayResult(string studentId);

void DisplayResult();

void prepareRanking(string studentId[]);

void initialize();

};

//主任构造函数（科目名称为空）

headTeacher::headTeacher(string teacherId, bool x):teacher(teacherId,x) {

job = "Head Teacher";

subjectName = "";

initialize();

}

//主任初始化函数（用以计算学生数量）\*\*原先打算分为科目的学生数量和总学生数量，由于工作量原因就没法完成，可是还是保留此函数作为可扩展部分\*\*

void headTeacher::initialize() {

//count number of student

string temp, idTemp;

ifstream studentResult, teacherList;

string::iterator it;

int lineCount = 0;

studentResult.open("D:/studentResult.txt");

if (studentResult.is\_open()) {

while (studentResult.good()) {

getline(studentResult, temp);

lineCount++;

temp.clear();

}

}

studentResult.close();

totalStu = lineCount - 1;

}

//增添新学生函数

void headTeacher::addResult() {

ofstream studentResult, studentList;

string temp;

float tempMark;

studentResult.open("D:/studentResult.txt", ios\_base::app);

studentList.open("D:/studentList.txt", ios\_base::app);

if (studentResult.is\_open()) {

studentResult.seekp(0, ios\_base::end);

studentResult << endl;

cout << "Enter student Id: ";

if (fileInput) {

getline(inputFile, temp);

}

else

cin >> temp;

studentResult << temp << ",";

if (studentList.is\_open()) {

studentList.seekp(0, ios\_base::end);

studentList << endl << temp << ",";

}

cout << "Enter student name: ";

if (fileInput) {

getline(inputFile, temp);

}

else

cin >> temp;

studentResult << temp;

if (studentList.is\_open()) {

studentList << temp << "123";

studentList.close();

}

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

cout << "Enter " << allSubjects[i] << " Result: ";

if (fileInput) {

getline(inputFile, temp);

tempMark = stof(temp);

}

else

cin >> tempMark;

studentResult << "," << setfill('0') << setw(5) << setiosflags(ios\_base::fixed) << setprecision(1) << tempMark;

cout << resetiosflags(ios\_base::fixed);

}

cout << "Student result is updated. A new student is added." << endl;

}

studentList.close();

studentResult.close();

}

//显示全部学生成绩和排名函数

void headTeacher::DisplayResult() {

string\* studentId = new string[totalStu];

ifstream studentResult;

string temp, tempEle;

string::iterator it;

int i = 0;

cout << endl << "Teacher's Name: " << username << endl << "Teacher ID: " << id << endl;

studentResult.open("D:/studentResult.txt");

if (studentResult.is\_open()) {

getline(studentResult, temp);

while (studentResult.good()) {

getline(studentResult, temp);

for (it = temp.begin(); \*it != ','; it++)

tempEle.push\_back(\*it);

studentId[i++] = tempEle;

tempEle.clear();

}

}

studentResult.close();

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

student A(studentId[i]);

A.loadResult();

A.DisplayResult();

}

cout << endl;

prepareRanking(studentId);

delete[]studentId;

}

//显示某个学生成绩函数（需提供学生证号）

void headTeacher::DisplayResult(string studentId) {

ifstream stu;

string temp, idTemp;

string::iterator it;

int flag = 0;

stu.open("D:/studentResult.txt", ios\_base::in);

if (stu.is\_open()) {

getline(stu, temp);

while (stu.good()) {

getline(stu, temp);

for (it = temp.begin(); \*it != ','; it++)

idTemp.push\_back(\*it);

if (studentId.compare(idTemp) == 0)

{

flag = 1;

break;

}

idTemp.clear();

}

if (flag == 0) {

cout << "Student Id does not exist." << endl;

return;

}

}

stu.close();

cout << endl;

cout << "Head Teacher Name: " << username << endl;

cout << "Head Teacher Id: " << id << endl;

cout << "Job: " << job << endl;

student tempObj(studentId);

tempObj.loadResult();

tempObj.DisplayResult();

}

//计算学生排名函数（利用了学生类的 < 重载）

void headTeacher::prepareRanking(string studentId[]) {

for (int i = 0; i < totalStu - 1; i++) {

for (int j = 1; j < totalStu; j++) {

if (student(studentId[i]) < student(studentId[j])) {

string temp = studentId[i];

studentId[i] = studentId[j];

studentId[j] = temp;

}

}

}

cout << "\*\*\*Student Ranking\*\*\*" << endl;

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

cout << i + 1 << ". " << studentId[i] << endl;

}

//主任版显示主页函数

void headTeacher::DisplayMainMenu() {

cout << endl;

cout << "\*\*\*\*\*\*\* Main Menu \*\*\*\*\*\*\*" << endl;

cout << "Choose an option from the following." << endl;

cout << "1. Check One Student Results." << endl;

cout << "2. Add Results." << endl;

cout << "3. Display All Results." << endl;

cout << "4. Check any subject results." << endl;

cout << "5. Exit." << endl;

}

//主任版选项函数

bool headTeacher::Option(int option) {

string studentId;

string checkSubject;

string str;

int flag;

switch (option) {

case 1:

cout << "Enter student Id you want to check: ";

if (fileInput) {

getline(inputFile, studentId);

}

else

cin >> studentId;

DisplayResult(studentId);

break;

case 2:

addResult();

break;

case 3:

DisplayResult();

break;

case 4:

//主任查询科目成绩前需输入想查询科目名称

cout << "Enter a subject to check: ";

if (fileInput) {

getline(inputFile, checkSubject);

}

else

cin >> checkSubject;

cout << "Do you want to display distribution (0->No, 1->Yes)? : ";

if (fileInput) {

getline(inputFile, str);

flag = stoi(str);

}

else

cin >> flag;

loadSubjectResult(checkSubject, flag);

subjectName = checkSubject;

DisplaySubjectResult(flag);

break;

case 5:

cout << "Exiting..." << endl;

return 1;

default:

cout << "Enter a valid option!" << endl;

}

return 0;

}

//用户登入系统（核对证号、密码）并根据证号确定身份

int idVerify(bool file) {

string id, pwd, temp, idTemp, pwdTemp, jobTemp;

string::iterator it;

ifstream studentList, teacherList;

cout << "Enter credentials." << endl;

cout << "UserId: ";

if (file) {

getline(inputFile, id);

}

else

cin >> id;

cout << "Password: ";

if (file) {

getline(inputFile, pwd);

}

else

cin >> pwd;

studentList.open("D:/studentList.txt");

if (studentList.is\_open()) {

getline(studentList, temp);

while (studentList.good()) {

temp.clear();

getline(studentList, temp);

for (it = temp.begin(); \*it != ','; it++)

idTemp.push\_back(\*it);

if (id.compare(idTemp) == 0)

{

for (it = temp.begin() + 11; it != temp.end(); it++)

pwdTemp.push\_back(\*it);

if (pwd.compare(pwdTemp) == 0) {

idGlobal = id;

return 1;

}

}

idTemp.clear();

pwdTemp.clear();

}

}

else {

cout << "Student data file not found!" << endl;

exit(0);

}

temp.clear();

studentList.close();

teacherList.open("D:/teacherList.txt");

if (teacherList.is\_open()) {

getline(teacherList, temp);

while (teacherList.good()) {

temp.clear();

getline(teacherList, temp);

for (it = temp.begin(); \*it != ','; it++)

idTemp.push\_back(\*it);

if (id.compare(idTemp) == 0)

{

for (it = temp.begin() + 7; \*it != ','; it++)

pwdTemp.push\_back(\*it);

if (pwd.compare(pwdTemp) == 0) {

idGlobal = id;

it++;

while (\*it != ',')

it++;

for (it = it + 1; it != temp.end(); it++)

jobTemp.push\_back(\*it);

if (jobTemp.compare("Head Teacher") == 0)

return 3;

else

return 2;

}

}

idTemp.clear();

pwdTemp.clear();

}

}

else {

cout << "Teacher data file not found!" << endl;

exit(0);

}

return 4;

}

//成绩系统启动函数

void resultSystem(int identity, bool file) {

int exitSwitch = 0;

if (identity == 1) {

while (!exitSwitch) {

student A(idGlobal, file);

A.DisplayMainMenu();

int option = A.chooseOption();

exitSwitch = A.Option(option);

}

}

else if (identity == 2) {

teacher A(idGlobal, file);

while (!exitSwitch) {

A.DisplayMainMenu();

int option = A.chooseOption();

exitSwitch = A.Option(option);

}

}

else if (identity == 3) {

while (!exitSwitch) {

headTeacher A(idGlobal, file);

A.DisplayMainMenu();

int option = A.chooseOption();

exitSwitch = A.Option(option);

}

}

else {

cout << "You have no permission to access the result system." << endl;

exit(0);

}

}

//主函数

int main() {

bool file;

string blank;

//确定是否使用文件输入（否则使用控制台）

cout << "File input or not: " << endl;

getline(inputFile, blank);

file = stoi(blank);

getline(inputFile, blank);

if (file) {

while (inputFile.good()) {

int identity = idVerify(file);

resultSystem(identity, file);

getline(inputFile, blank);

cout << endl;

}

inputFile.close();

}

else {

inputFile.close();

int identity = idVerify(file);

resultSystem(identity, file);

}

}

**多用途计算器**

目 录

[1. 系统需求分析 46](file:///C:\Users\liew\Downloads\2020程设2大作业任务书.docx#_Toc171224774)

[2. 总体设计 46~47](file:///C:\Users\liew\Downloads\2020程设2大作业任务书.docx#_Toc171224775)

[3. 详细设计 48~51](file:///C:\Users\liew\Downloads\2020程设2大作业任务书.docx#_Toc171224776)

[4. 测试结果 52~54](file:///C:\Users\liew\Downloads\2020程设2大作业任务书.docx#_Toc171224778)

[5. 系统说明书与结果分析 55~56](file:///C:\Users\liew\Downloads\2020程设2大作业任务书.docx#_Toc171224778)

[6. 系统调试 56](file:///C:\Users\liew\Downloads\2020程设2大作业任务书.docx#_Toc171224777)

[7. 总结 56](file:///C:\Users\liew\Downloads\2020程设2大作业任务书.docx#_Toc171224779)

\*\*源程序在压缩文件包内

\*\*原先选题为蛇棋，后来更改为多用途计算器（总结部分有简单说明情况）

**系统需求分析**

多用途计算器包含三个主要部分，分别是实数的运算、虚数的加乘和图形的计算。实数的运算包括整数和浮点数加减乘除。另外，图形包括了二维和三维。二维图形的函数包括面积和周长的计算。三维图形则是表面积和体积的计算。此计算器突出的地方在于其能进行任意大数目的运算，也是为了解决double或long long在数字太大时无法完整储存数据的问题。

除此之外，计算器也有储存计算历史的功能。用户能够显示计算历史或清除历史。计算历史功能是为了让用户能够随时参照之前的计算，为用户提供了便利。

1. 实数四则运算：顾名思义，就是根据输入的数字和运算符进行运算，并输出结果。
2. 虚数的运算：同样是根据输入的数字和运算符进行运算，并输出结果。
3. 图形计算：根据用户输入的图形类型和参数，进行面积、周长或表面积、体积的运算，并输出结果
4. 显示计算历史：从文件读入从前计算历史，并显示在控制台上。
5. 删除计算历史：将存放计算历史的文件重置

**总体设计**

在进行操作时，用户只需跟随控制台上的菜单和指示将数字和运算符（或图形参数）输入到系统中，就能得到最终的结果（文件输入在后边说明书部分有详细介绍）。如上所述，系统分为三个部分：

* **实数的加减乘除：**将输入的数字制成一个类numString，并利用已重载的＋-\*/进行四则运算，如此以来，计算器理论上可计算任意大小的数字。
* **虚数的加乘：**虚数分为实部和虚部两个部分，同样是使用已重载的＋\*进行运算。由于过程中以numString类为运算核心，因此理论上可计算任意大小的数字。
* **图形计算（也使用numString类）：**2d图形包括长方形、圆形和三角形，可计算面积和周长（除三角形）。3d图形包括长方体、圆柱体和球体，皆可计算表面积和体积。

**多用途计算器系统模块图：**

多用途计算器

实数四则运算

清除计算历史

显示计算历史

虚数运算（加乘）

图形运算（二维&三维）

输入图形类别、相应参数

输入数字、运算符

计算&显示最后结果

写入历史条件：超出两条计算历史||选择显示历史功能||退出程序

写入计算历史

图【1】

**详细设计**

**多用途计算器类层次图：**

系统类：

System类

核心类（实数四则运算）：

numString类

虚数运算：

Complex类

图形运算：

shape3d类（抽象类）

shape2d类（抽象类）

Circle类

Cylinder类

Cuboid类

Sphere类

Triangle类

Rectangle类

图【2】 多用途计算器中十个类的类层次图

输入数字

实数四则运算

输入运算符

运算并显示结果

退出至主菜单

图【3】实数四则运算功能模块图

输入数字、输入运算符

虚数运算

处理输入字串，分离数字

运算并显示结果

退出至主菜单

图【4】虚数运算功能模块图

输入图形名称，参数

图形运算

处理输入字串，确认图形类别分离参数

运算并显示结果

退出至主菜单

图【5】图形运算功能模块图

打开历史文件并逐行读入计算历史

显示计算历史

显示计算历史

退出至主菜单

图【6】显示计算历史功能模块

numString类

-num:string

-length:int

+numString(x:string)

+operator+(p:numString):numString

+operator-(p:numString):numString

+operator\*(p:numString):numString

+operator/(p:numString):numString

+ stringSplit(p1: string&,p2: string&, swapStr: bool): string\*

+ findDotPos(p1: string&, p2: string&):int

System类

- dateTime:string

-history: queue<string>

-fileInput:bool

+System(str:char\*, file:bool)

+DisplayMainMenu():void

+initialize():void

+readHistory():void

+writeHistory():void

+deleteHistory():void

+basicNum():void

+complexNum():void

+shapes():void

Complex类

-real:string

-imag:string

+Complex(x:string,y:string)

+operator+(p:Complex):Complex

+operator\*(p:Complex):Complex

基类

shape2d类

-perimeter:string

-area:string

+calcArea():virtual void

+showResult():virtual string

派生类

派生类

派生类

Triangle类

-base:string

-height:string

+Triangle(x:string,y:string)

+calcArea():void

+showResult():string

Circle类

-radius:string

+Circle(x:radius)

+calcArea():void

+calcPerimeter():void

+showResult():string

Rectangle类

-length:string

-width:string

+Rectangle(x:string,y:string)

+calcArea():void

+calcPerimeter():void

+showResult():string

基类

shape3d类

-surfaceArea:string

-volume:string

+calcSurfaceArea():virtual void

+calcVol():virtual void

+showResult():virtual string

派生类

派生类

派生类

Cylinder类

-height:string

-radius:string

+Cylinder(x:string,y:string)

+calcSurfaceArea():void

+calcVol():void

+showResult():string

Sphere类

-radius:string

+Sphere(x:string)

+calcSurfaceArea():void

+calcVol():void

+showResult():string

Cuboid类

-length:string

-width:string

-height:string

+Cuboid(x:string,y:string,z:string)

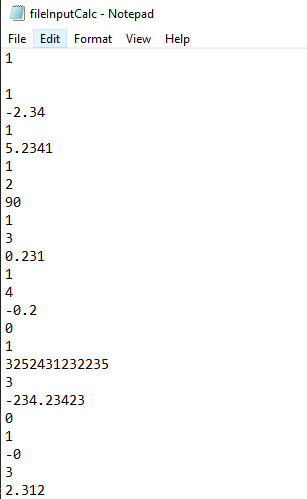
+calcSurfaceArea():void

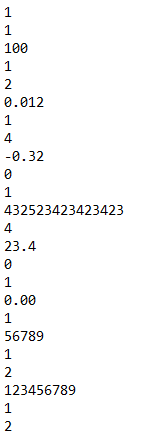
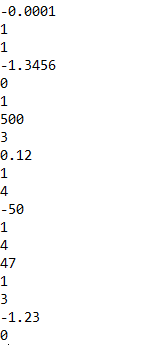
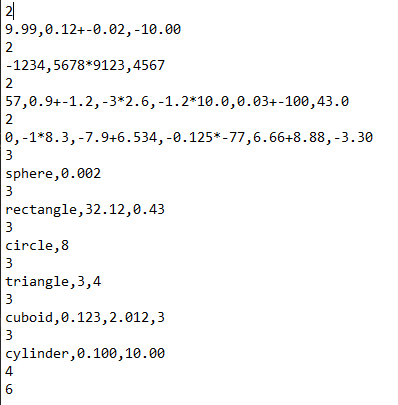
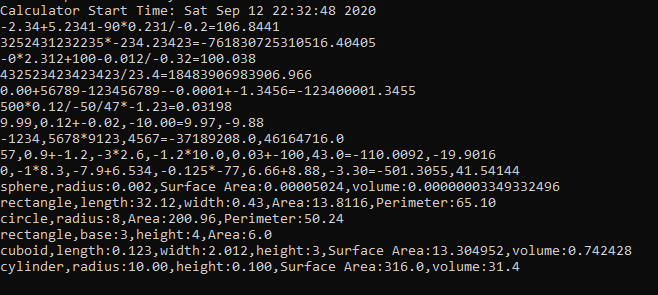
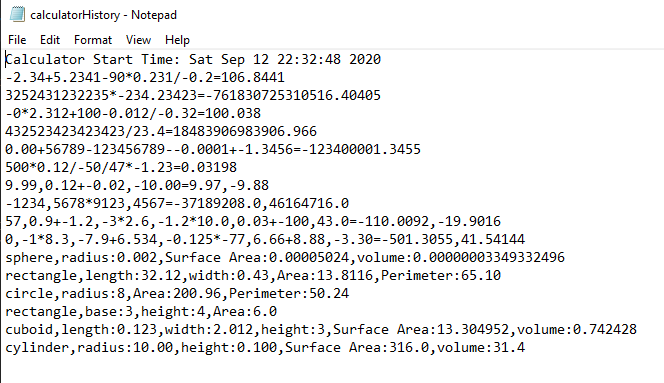
+calcVol():void

+showResult():string

图【7】多用途计算器十个类的UML图

**测试结果**

**测试文件：**

**测试结果：**

**系统使用说明书与结果分析**

**使用说明书:**

此系统可利用文件输入与控制台输入两种形式进行操作，推荐使用控制台操作（更加简单）。下面将说明文件输入的形式。

\*\*若使用控制台输入也需建立文件并将第一行标记为0

|  |  |
| --- | --- |
| 第一行 | 1：文件输入 0：控制台输入 |
| 第二行 | 空 |
| 第三行 | 1：实数四则运算 2：虚数运算 3：图形运算 4：显示计算历史 5：清空历史 6：退出（结束程序） |

接下来的每一行根据功能选择的不同操作有别（若使用控制台，将会显示菜单和指示 ）

\*\*每次完成一次完整的操作后都会退回主菜单

**实数四则运算**（\*\*除数不能过大，其他并无限制）

|  |  |
| --- | --- |
| 输入第一个数字 | |
| 输入运算符（1：+ 2：- 3：\* 4：/） | |
| 输入第二个数字 | |
| 是否继续 1：是 | 0：否 |
| 输入运算符 | 计算结果并退回主菜单 |
| 输入接下来的数字 |  |

循环

**虚数运算**

|  |
| --- |
| 输入格式：实部1，虚部1+或\*实部2，虚部2 （可任意个虚数） |

**图形**

|  |
| --- |
| 输入格式：图形名称（全小写），参数1，参数2（如有），参数3（如有） |

**图形参数表**

|  |  |
| --- | --- |
| 长方形 rectangle | 参数1：长度 参数2：宽度 |
| 三角形triangle | 参数1：底部长度 参数2：高度 |
| 圆形 circle | 参数1：半径 |
| 长方体 cuboid | 参数1：长度 参数2：宽度 参数3：高度 |
| 圆柱体 cylinder | 参数1：高度 参数2：半径 |
| 球体 sphere | 参数1：半径 |

**结果分析：**

测试结果如上述的图片calculatorHistory所示。经过科学计算器验证，结果都是正确的（进行除法后结果会有些误差，因为程序只取三位小数点）。另外，结果也符合先乘除后加减的规则。

**系统调试**

坦白地说，调试过程挺折磨人的。由于此次计算器的核心是任意数目的四则运算，因此我必须将负数和浮点数列入考虑范围，也必须考虑先乘除后加减的条规，造成代码较复杂，调试较麻烦。首先，最常见的问题是string的越界。例如，在计算0.01\*0.01此类的数字时，由于我对浮点数乘法的处理是将小数点移位、消掉多余的零，并在进行乘积和消掉前置0后再移位小数。这时，若未补上0则会造成string越界的错误。因此我需要增加个if 语句来解决此问题。此外，在进行四则运算遇到负号时，我是直接将this->num的负号消除后再进行运算，后来调试时发现到会引起问题，如在虚数的乘法时，会因负号的消失而是计算产生错误（因原号码改变了），因此，在进行任何运算前，我都会将原有数据进行复制，再进行运算。另外，另一个调试难点在于数字的输入。以虚数的输入为例，我的设计是用户将所需进行的运算一连串地输入进控制台（可包含任意个运算符），这部分的代码是将输入的string 分离成数字和运算符再进行运算，较为复杂。所幸调试时只遭遇了vector容器和string 忘了清除这些较容易发现的错误，因此调试过程还算顺利（不枉费我花那么长时间编写）。

**总结**

与学生成绩管理系统一样，多用途计算器的设计过程让我的编程水平更上一层楼。计算器其实是我的第二个主题（原先的主题是蛇棋）。更换的原因是蛇棋的图形编程方面较复杂（时间限制来不及完成），而若放弃图形编程，蛇棋又过于简单。言归正传，多用途计算器的难点主要是任意数字的四则运算，设计过程中不仅需要考虑浮点数也要考虑负数，因此考验了我全面思考的能力。至于编程技术，除了将已学过的技术添加到设计中，我也尝试将多线程技术整合到代码中。虽然只是涉及入门的运用，可是也是一个接触新知识的好机会。值得一提的是，我原本对此设计工作量的预计在600至700行代码左右，没想到完成后竟然超过1000行。由于过程耗费许多心思和精力，我决定不删减代码，在此麻烦助教了。另外，我认为此次设计的不足在于其功能有限，不能进行根号或三角函数等运算，或许我应该放弃图形运算和虚数的部分，并专注在于实数吧。综上所述，此次设计不仅提高了对课程所学知识的运用熟练度，也让我有机会学习些新知识，如。程序设计的大作业就此完结。在此对老师和助教真挚地道谢！