2018-2019 Term2 程序设计实践作业报告

王昊恩 3018216021

本次作业内容已全部上传至 $Github$ 。

$Github$ 仓库地址：https://github.com/FireSSang/18-19-Summer-Programming-Practice-Homework

第一部分

一、实践训练题目： 10 选 6 ，每题 10 分，共 60 分

1、输入一个长度小于 100 的字符串，判断其是否为回文串。

2、输入长度为 n 的整形数组，分别使用选择和冒泡排序进行排序。

3、输入长度为 n 的整形数组，使用快速排序或者归并排序进行排序。

4、实现只支持加号、减号的计算器。

5、实现二进制转 10 进制的转换器（输入二进制数，输出十进制数）

6、输入一个日期 $(xxxx-xx-xx)$ ，计算 n 天以后的日期。

7、输入一个正整数，判断这个数是否为素数（质数）。

8、输入长度小于 10 的字符串，输出所有子串。

9、输入两个矩阵（分别为 $m*n$ 及 $n*q$ ），编码输出两个矩阵相乘的结果。

10、对一个长度为 n 的升序数列，用二分法进行查询。

完成情况：

完成了 `1` ，`2` ，`3` ，`5` ，`6` ，`7` ，`8` ，`9` ，`10` 共 9 题，此部分程序代码在 `Text1` 文件夹下。

每个程序内包含详细的操作指引，并提前告知了可能的异常情况，在运行程序时输出。

例如 `5` 题：

> In this program, you can enter several binary strings, it will convert them to several decimal integers. Please note that the string length should be within 64 bits, otherwise "The string is too long." will be output. The binary string should be represented by the complement, that is, the highest bit represents its symbol. A positive number should start with "0" and a negative number should start with "1". Therefore, the length of the string should be at least 2, for example, a positive number "1" should be entered as "01". Now enter the binary string, or enter "END" to terminate the program:

第二部分

二、 课程设计： 二选一，共 40 分

评分标准：用户界面不作为评分项（用Dos界面即可），评分项包括代码风格、注释完整性、功能完整性、文档完整性。

1、基于面向对象设计并实现学生成绩管理系统，用于统计单科成绩、加权分计算、年级排名等。设计发散功能可作为加分项。

2、自拟题目进行系统设计与实现，要求使用面向对象。

完成情况：

程序使用 $CLion$ 开发， `C++ 14.0` 标准。全部文件（包括存储数据的 $data.txt$ ）使用 GBK 编码，以避免中文引起的乱码。

1.项目功能介绍

项目实现了以下功能（引用程序内操作指引）：

> 欢迎使用学生成绩管理系统 Designed by Wang Haoen 系统正在初始化，请稍候... 系统初始化完成！ 您可以输入序号进行以下操作：
>
> 1.输入学生成绩

2.修改学生成绩

3.删除学生成绩

4.查询学生信息

5.查询单科信息

6.查看成绩总览

7.查看年级排名

0.安全退出系统 提示：建议您安全退出，强制终止程序可能会导致数据丢失。

1. 输入学生成绩

   您可以输入序号进行以下操作：

   1.建立新的学生档案

   2.输入学生学号以添加成绩

   3.输入学生姓名以添加成绩

   0.返回上一级

   建立新的学生档案时检测是否出现学号重复情况，不检测重名。

2. 修改学生成绩

   您可以输入序号进行以下操作：

   1.输入学生学号以修改成绩

   2.输入学生姓名以修改成绩

   0.返回上一级

3. 删除学生成绩

   您可以输入序号进行以下操作：

   1.输入学生学号以删除成绩

   2.输入学生姓名以删除成绩

   0.返回上一级

4. 查询学生信息

   您可以输入序号进行以下操作：

   1.输入学生学号以查询信息

   2.输入学生姓名以查询信息

   0.返回上一级

5. 查询单科信息

   请输入科目名称（输入0以结束操作并返回上一级）：

6. 查看成绩总览

   科目名称： 科目学分： 平均成绩： 选课人数： 排名 学号 姓名 成绩 GPA

7. 查看年级排名

   年级排名如下：

   排名 学号 姓名 成绩 GPA

8. 安全退出

   系统正在保存数据，请稍候... 数据保存成功！ 期待您的下次使用！

2.项目功能优化

1. 错误处理

   项目内设置了报错界面，在用户进行了无效的操作后提示用户操作无效并返回上一级。

   可能的报错原因：

   1. 选择操作时输入错误的操作代码；
   2. 新建学生信息时学号重复；
   3. 新建、修改、删除成绩信息时学生不存在或科目不存在；
   4. 输入的成绩不在 $[0, 100]$ 范围内；

2. 数据读取与存储

打开系统时，程序会读取存放在 `cmake-build-debug` 目录下的 $data.txt$ 文件。

关闭程序时建议使用安全退出，否则会丢失当次使用时进行的可能的修改操作。

关闭程序前，程序会将原 $data.txt$ 清空并按原格式写入新的数据信息。

3.项目结构介绍

项目包含以下 4 个类，下面将逐个进行简单介绍（在每个类的 $.h$ 和 $.cpp$ 中包含了详细的介绍）：

1. $Method$

   $Method$ 是一个抽象类，是 $System$ 的基类，主要用于约束 $System$ 类中需要实现的功能。 $System$ 中大多数函数是通过 $override\ Method$ 中的函数实现的。

   代码如下 $(Method.h)$：

```
#include <bits/stdc++.h>

/**
 * 抽象类
 * 该类下定义了需要在 System 类中 override 的函数
 * 在 System.cpp 下对每个函数进行详细解释
 */
class Method
{
public:
    //初始化及退出
    virtual void system_initialization() = 0;
    virtual void system_exit() = 0;

    //界面
    virtual void guide_interface() = 0;
    virtual void input_interface() = 0;
    virtual void modify_interface() = 0;
    virtual void delete_interface() = 0;
    virtual void query_interface() = 0;
    virtual void error_interface() = 0;

    //操作
    virtual void add_data() = 0;
    virtual void input_data(int) = 0;
    virtual void modify_data(int)= 0;
    virtual void delete_data(int) = 0;
    virtual void query_student(int) = 0;
    virtual void query_subject() = 0;
    virtual void output_all_subject() = 0;
    virtual void output_total_ranking() = 0;

    //定位及查重
    virtual int find_by_student_ID(std::string) = 0;
    virtual int find_by_student_name(std::string) = 0;
    virtual int find_by_subject_name(std::string) = 0;
    virtual bool is_duplicate_names(std::string) = 0;
};
```

2. $Sysetm$

   $System$ 是成绩管理系统类，是项目的主要实现部分。

   该类继承了抽象类 $Method$ 。除 $override\ Method$ 中的函数外，该类还分别定义了 $Student$ 对象和 $Subject$ 对象的 $vector$ 以存储学生和科目信息。

   代码如下 $(System.h)$：

```
#include <bits/stdc++.h>
#include "Method.h"
#include "Student.h"
#include "Subject.h"

/**
 * 操作系统类
```

```
 * 该类下 override 了抽象类 Method 中的方法
 * 在 System.cpp 下对每个函数进行详细解释
 */
class System : Method
{
public:
    //初始化及安全退出
    void system_initialization() override;
    void system_exit() override;

    //界面
    void guide_interface() override;
    void input_interface() override;
    void modify_interface() override;
    void delete_interface() override;
    void query_interface() override;
    void error_interface() override;

    //操作
    void add_data() override;
    void input_data(int) override;
    void modify_data(int) override;
    void delete_data(int) override;
    void query_student(int) override;
    void query_subject() override;
    void output_all_subject() override;
    void output_total_ranking() override;

    //定位及查重
    int find_by_student_ID(std::string student_ID) override;
    int find_by_student_name(std::string student_name) override;
    int find_by_subject_name(std::string subject_name) override;
    bool is_duplicate_names(std::string student_name) override;

private:
    std::vector<Student> stud;
    std::vector<Subject> subj;

    //比较函数
    static bool sort_with_total_score(Student, Student);
    static bool sort_with_credit(Subject, Subject);
};
```

3. $Student$

该类用于保存学生信息。

代码如下 $(Student.h)$：

```
#include <bits/stdc++.h>

/**
 * 类 Student 的解释：
 * 本类中定义了学生的学号、姓名、总评成绩、绩点和所选科目数
 * 本类中使用结构体定义了学生需要记录的科目信息，包括学生所选科目名称、科目学分、科目成绩和科目绩点
 * 本类中使用 STL 中的容器 vector 存储科目信息
 * 本类中定义了如下函数，右侧标注了其主要功能
 * 在该类对应的.cpp文件中，对每个函数进行了详细的解释
 */
class Student
{
public:
    Student();                                              //无参构造函数
    Student(std::string, std::string);                      //含参构造函数
    std::string get_ID();                                   //读取学生学号
    std::string get_name();                                 //读取学生姓名
    double get_total_score();                               //读取学生加权
    double get_total_credit();                              //读取学生学分
    void set_score(std::string, double, double);            //写入学生成绩
    double get_GPA();                                       //读取学生绩点
    void set_rank(int);                                     //写入学生排名
    int get_rank();                                         //获取学生排名
    int get_number_of_subject();                            //获取选课数量
```

```
    void modify_score(std::string, double);                 //修改学生成绩
    void delete_score(std::string);                         //删除学生成绩
    void show_information();                                //展示科目信息
    bool check_subject(std::string);                        //查询是否已选

    void get_subject(int, std::string&, double&, double&);  //保存文件函数

private:
    std::string ID;
    std::string name;
    double total_score;
    double total_credit;
    double GPA;
    int rank;
    int number_of_subject;
    struct subject
    {
        std::string subject_name;
        double subject_credit;
        double subject_score;
        double subject_GPA;
    };
    std::vector<subject> list;

    static double calculate_single_GPA(double);             //计算单科绩点
    double calculate_total_GPA();                           //计算学生绩点
    double calculate_total_score();                         //计算学生加权
    static bool sort_with_credit(subject, subject);         //比较函数，将科目按学分降序排序
};
```

4. *Subject*

该类用于保存学生信息。

代码如下 $(Subject.h)$：

```
#include <bits/stdc++.h>

/**
 * 类 Subject 的解释：
 * 本类中定义了科目的名称、学分、平均成绩和选课人数
 * 本类中使用结构体定义了选修了该学科学生的信息，包括学生学号、学生姓名、学生成绩和学生绩点
 * 本类中使用 STL 中的容器 vector 存储学生信息
 * 本类中定义了如下函数，右侧标注了其主要功能
 * 在该类对应的.cpp文件中，对每个函数进行了详细的解释
 */
class Subject
{
public:
    Subject();                                                  //无参构造函数
    Subject(std::string, double);                               //含参构造函数
    std::string get_subject_name();                             //获取科目名称
    double get_subject_credit();                                //获取科目学分
    int get_number_of_student();                                //获取学生人数
    void set_student_score(std::string, std::string, double);   //录入学生成绩
    void modify_score(std::string, double);                     //修改学生成绩
    void delete_student(std::string);                           //删除学生成绩
    void show_information();                                    //展示学生信息

private:
    std::string name;
    double credit;
    double average_score;
    int number_of_student;
    struct student
    {
        std::string student_ID;
        std::string student_name;
        double student_score;
        double student_GPA;
    };
    std::vector<student> list;
```

```
    void calculate_average_score();                          //计算平均成绩
    static double calculate_GPA(double);                     //计算学生绩点
    static bool sort_with_score(student a, student b);       //比较函数，将学生按成绩降序排序
};
```

5. $x.cpp$

   由于篇幅有限，不在该文档中展示 $.cpp$ 文件内容，但每个函数都在 $.cpp$ 文件中进行了详细的解释，可以清楚地了解操作方式及功能。