

**本科课程设计**

课 程 名 称： C++程序设计

课 程 编 号： 08060226

学 生 姓 名： 葉小選（組長）、徐佳乐

学 号： 2021152956(组长) 、2021150791

学 院： 信息科学与技术学院

系： 计算机

专 业： 计算机科学与技术

指 导 教 师： 张晓刚

教 师 单 位：

开 课 时 间：2022 ~ 2023 学年度第 2 学期

**暨南大学教务处**

**2023 年 4 月 6 日**

**目录**

1. 题目及需求分析

题目：高校水电费管理系统

1. 选做此课题的目的与意义

学习如何用C++程序来编写一个管理系统，加深对于C++语言的了解和实际运用。

1. 系统的功能需求分析

**问题描述：**

**住宿学生信息包括：学号、姓名、性别、年龄、班级、用电量、用水量等信息。教工信息包括职工号、姓名、性别、年龄、工作部门、用电量、用水量等信息。能计算出学生和教工每月所要交的电费和水费。定义一个人员类，实现学生和教工共同的信息和行为。**

**功能要求：**

**（1）添加功能：程序能够添加不同学生和教工的记录，提供选择界面供用户选择所要添加的类别，要求编号号要唯一，如果添加了重复编号的记录时，则提示数据添加重复并取消添加。**

**（2）查询功能：可根据姓名、用水量、用电量信息对已添加的学生或教工记录进行查询，如果未找到，给出相应的提示信息，如果找到，则显示相应的记录信息。**

**（3）显示功能：可显示当前系统中所有学生和教工的记录，每条记录占据一行。**

**（4）编辑功能：可根据查询结果对相应的记录进行修改，修改时注意编号的唯一性。**

**（5）删除功能：主要实现对已添加的学生或教工记录进行删除。如果当前系统中没有相应的记录，则提示“记录为空！”并返回操作。**

**（6）统计功能：能根据多种参数进行统计。能统计学生和教工的用水用电量、所要交纳的电费和水费、未交纳水电费的人员信息等。**

**（7）保存功能：可将当前系统中各类记录存入文件中，存入方式任意。**

**（8）读取功能：可将保存在文件中的信息读入到当前系统中，供用户进行使用。**

**（9）计算电费和水费。学生每月都有一定额度的水电是免费使用的，超过的部分需要交费。**

1. 系统设计

组员分工：构思、代码实现由组长（叶小选）完成，绘图、PPT制作由组员（徐佳乐）完成。

1. 根据所选题目，通过用例图描述系统总体功能。

**设计思路**：

程序在打开后的第一个画面应该是主页面，主页面需要有两个选项，一个是普通用户登录（学生、教工），另一个是管理员登录，在普通用户登录和学生用户登录都设立一个登陆界面。

在登陆界面，需要有两个输入，分别是账号输入和密码输入，并且包含初次登陆的选项，即注册账户。

在注册账户页面，有两个选项，一个是教师用户注册，一个是学生用户注册，在选择对应身份后，会先进行账号注册，要求输入一次账号和连续输出两次密码，两次密码会进行校验，如果不一样则返回错误重新输入。账号注册完成再进入注册信息补充，分别需要输入他们的用户信息，包括姓名、年龄、工（学）号、单位（班级）、性别这些基础信息，他们的用水量和用电量会初始化为0。在填写工（学）号的时候会进行校验，工（学）号将会作为关键字，不允许重复，注册时输入的工（学）号，会与后台存储的工（学）号进行对比，当发现一样的返回错误。

管理员账户不提供注册，默认为创作者本人拥有，仅能透过给与管理员密钥提升普通用户身份组进管理员才能新增管理员数量。

当登陆完成——以管理员登录的情况。会出现一个界面，包括查询未缴费名单、学生查询、教工查询和全部查询、修改用户信息、新增管理员用户、删除管理员用户。学生查询会按班查询（输入班号），然后输出整个班级的信息，教工查询会按部查询（输入部门），然后输出整个部门教工信息。信息包括用户的关键字、部门(班级），用水量、用电量、总共需要缴付的水电费。新增管理员用户则是输入关键字，把该名用户提升到管理员。删除管理员用户则是输入关键字，把该名用户从管理员身份里删除。管理员身份为一个身份标识符，每名管理员都拥有一个身份标识符，当该标识符不匹配，则不属于管理员。修改用户信息则是有两个选项，一个是显示学生的，一个是显示教工的，选择对应的类别后，会显示当前全部的学生或教工的信息，之后再输入对应的关键字，以对指定用户进行信息修改，包括对用水量，用电量和删除该用户的选项。

当登陆完成——以普通用户登录的情况。学生和教工的页面都是差不多的，但仍会分出两个页面。普通用户页面显示当前用户的个人信息，包括他的名字，学号，所属班级，年龄，用电量，用水量以及他当前应该缴付的水电费。拥有修改选项，用户修改用户的密码，需要用户输入当前密码，输入的当前密码会与改名用户的密码（未修改前）进行对比，当密码相匹配时，会进行密码修改操作，新增临时数组1存放用户第一次输入的密码，当输入第二次密码时候再新增临时数组2，输入完成后会对两个临时数组进行比较，当两个临时数组完全一样时，临时数组一的内容覆盖原本的旧密码，并且释放两个临时数组。拥有缴费选项，用于给用户进行缴费，当缴费完成后，会对该名用户的用水量、用电量以及当前应该缴付的水电费进行清0操作。

当前需要缴费的金额：（用水量-免费用水额度）\*0.5+（用电量-免费用电额度）\*0.7，当>0时，需要缴费，当<=0时，无需缴费。

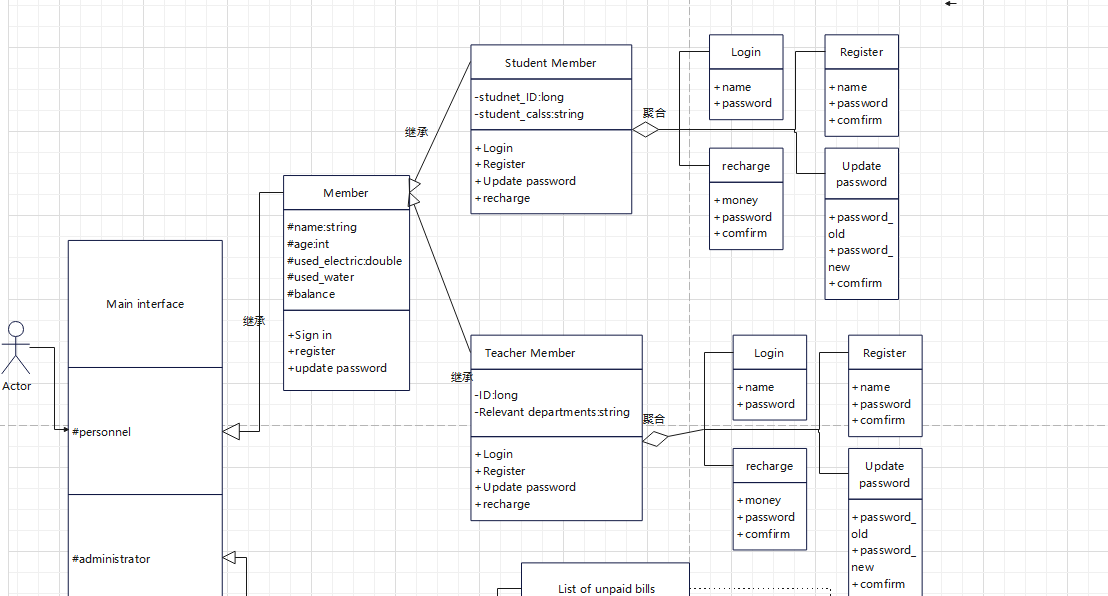
新增余额系统，余额系统会存储用户的金额进行自动扣费服务，设立cnt\_ddl计数为缴费的日期，设立pay\_ddl，当cnt\_ddl到达一个阈值会进行“缴费操作”，缴费操作流程先计算当前需要缴费的金额$p，然后$p与余额进行比较，当$p<余额，则直接进行自动缴费，并且留下记录，输出缴费成功信息。当$p>余额，则提示用户需要对余额进行储值，当到达pay\_ddl阈值，则会进行$p和余额的比较，若仍处于$p>余额，则输出请缴费，若$p<余额，则输出缴费成功。

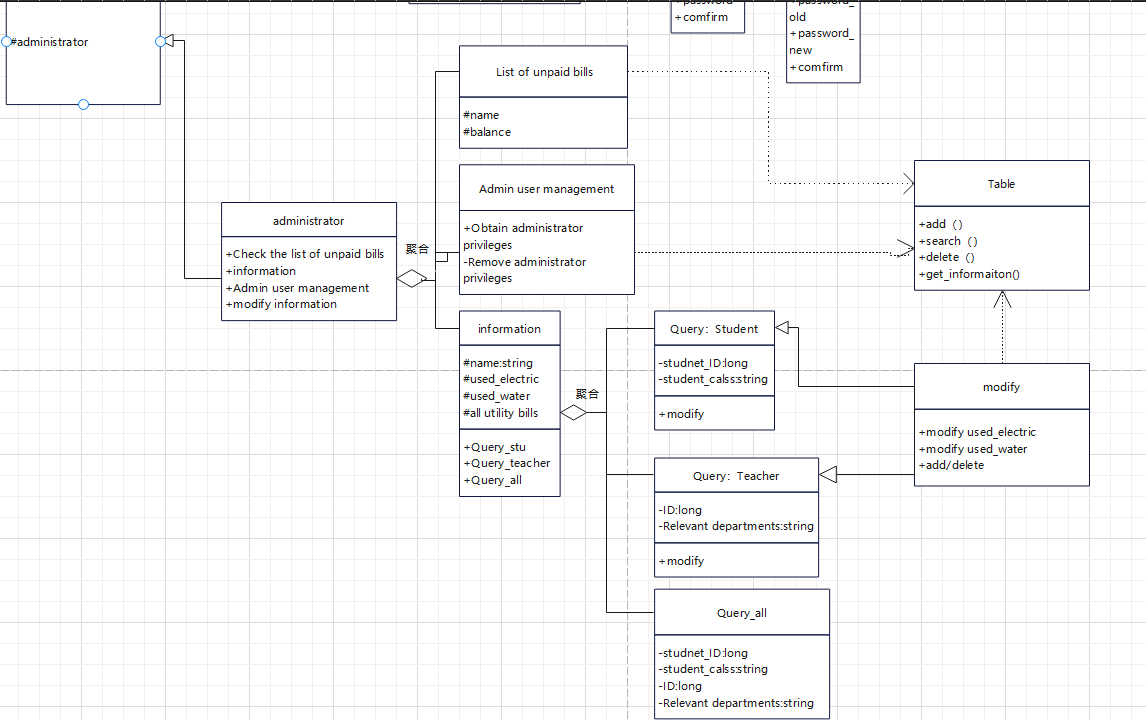
新增情况，在注册里面完成。编辑功能在用户个人页面里完成，统计功能，对全部用户的统计仅管理员内部完成，而对用户个人的统计则又用户部分独立完成。

保存方面-把当前的用户信息以txt文件保存，实现对文件的输入。

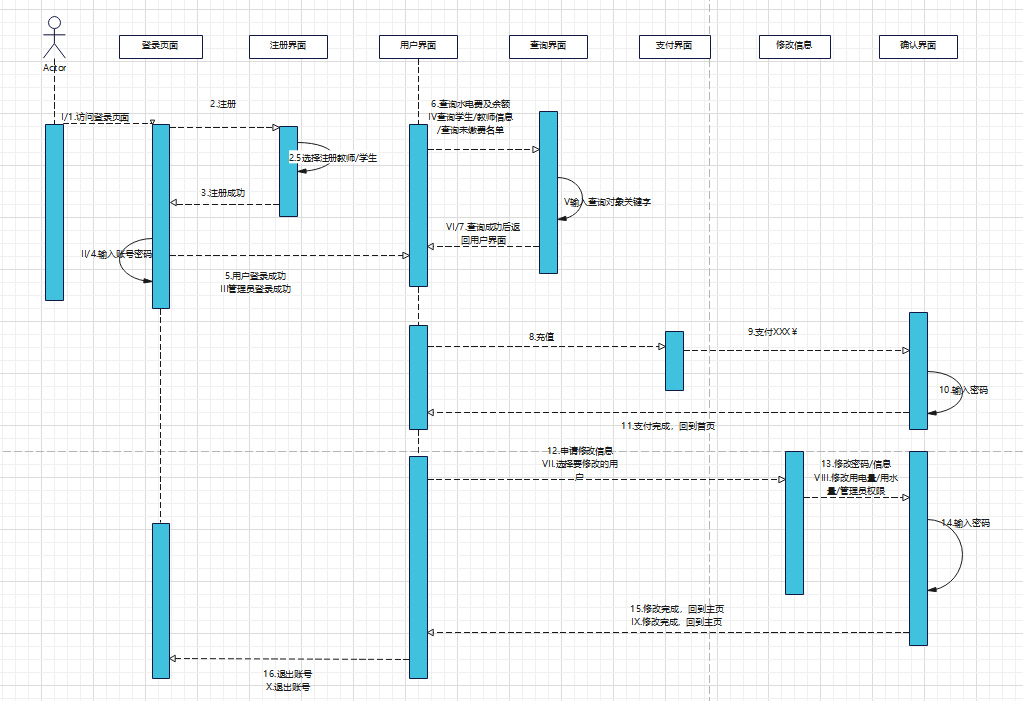
读取方面，对txt文件进行读取，并且显示在页面上。

2. 使用类图描述系统总体设计。





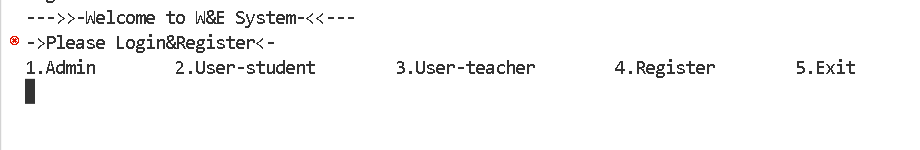
3. 使用时序图描述系统的主要业务操作。



三、系统运行及调试

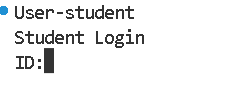
1. 系统运行及调试

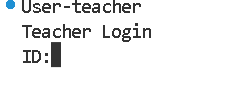
主菜单界面：



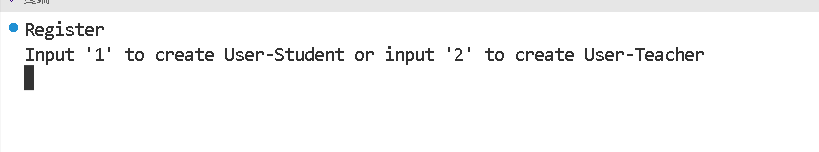
登陆界面：

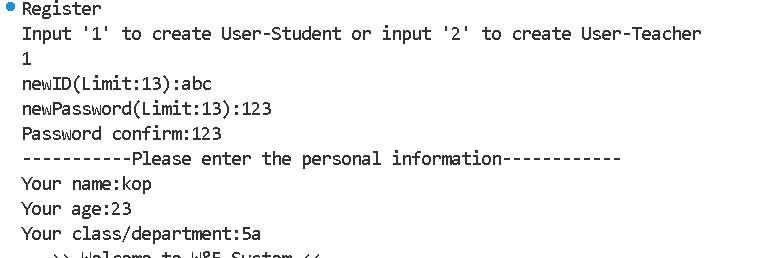




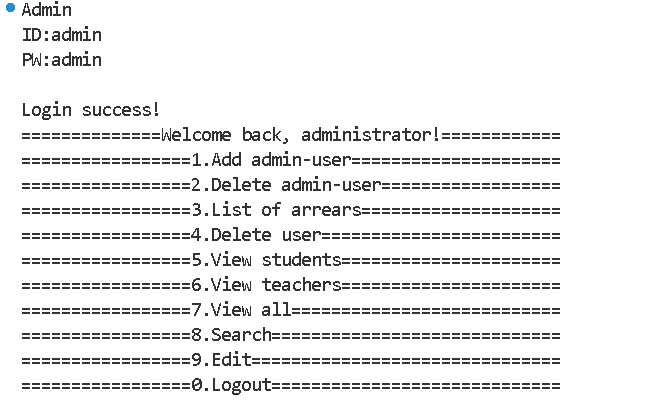


注册界面：



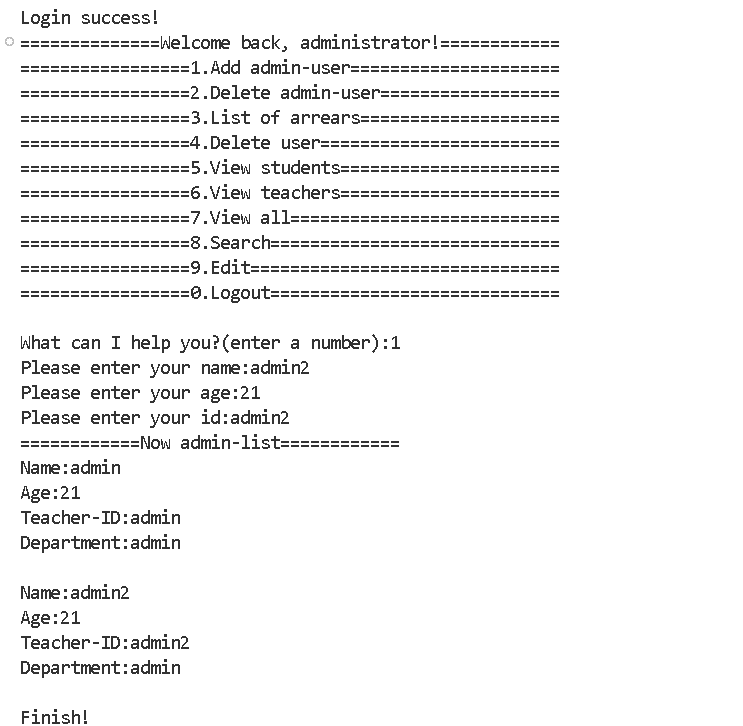


管理员界面：

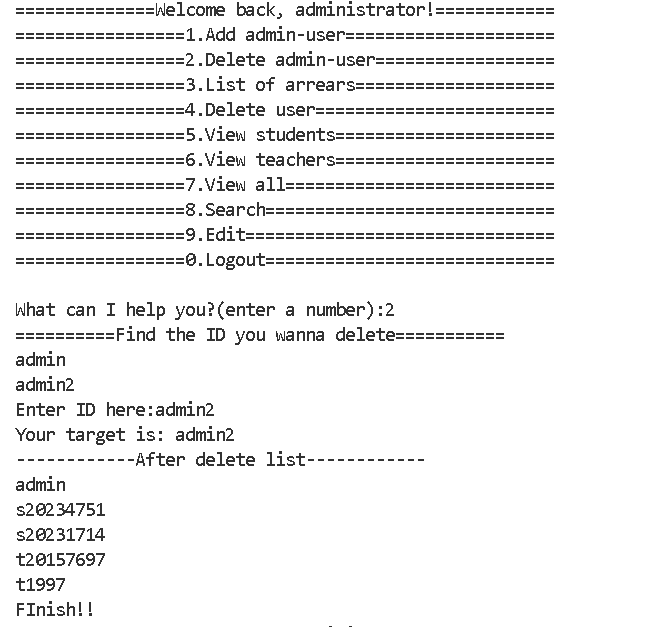


管理员功能实现：

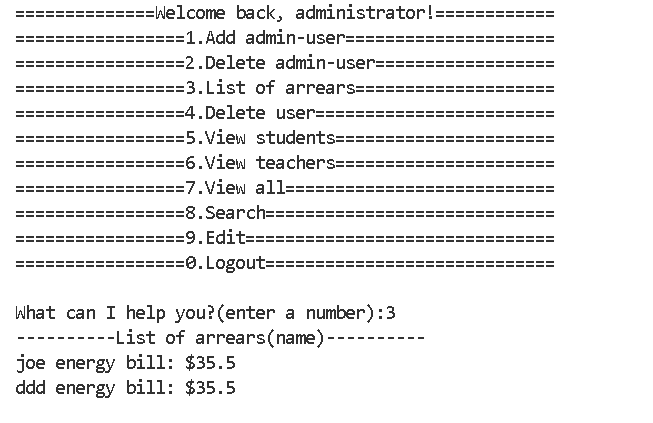
1.



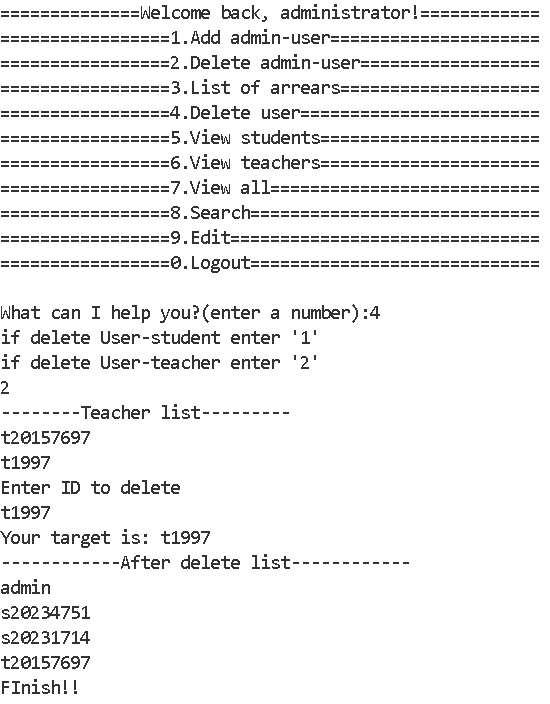
2.



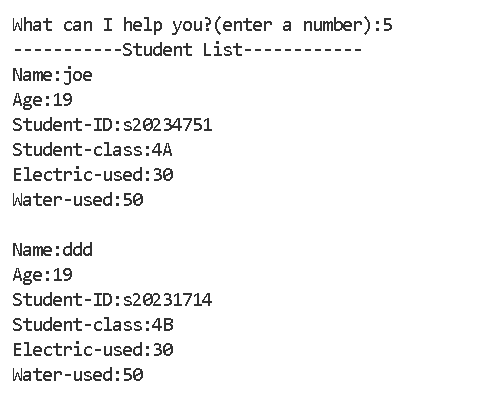
3.



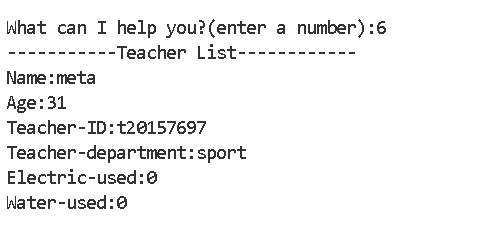
4.



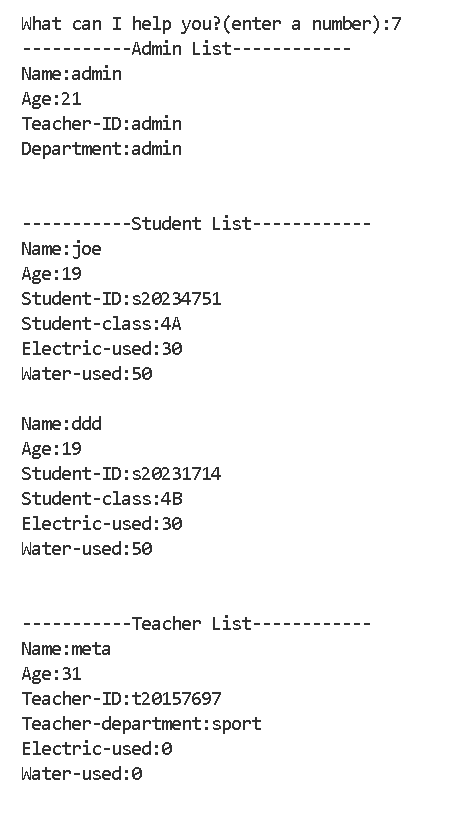
5.



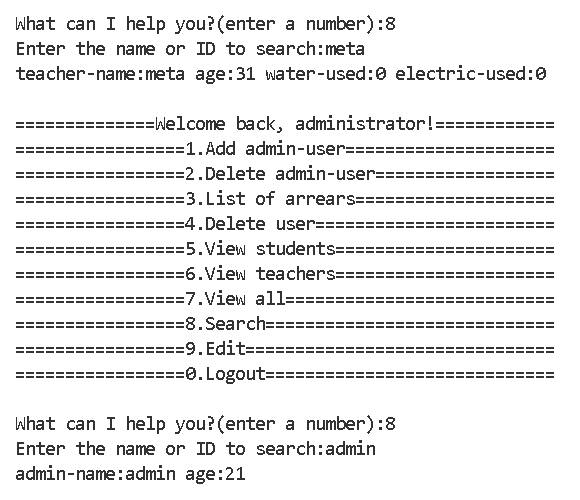
6.



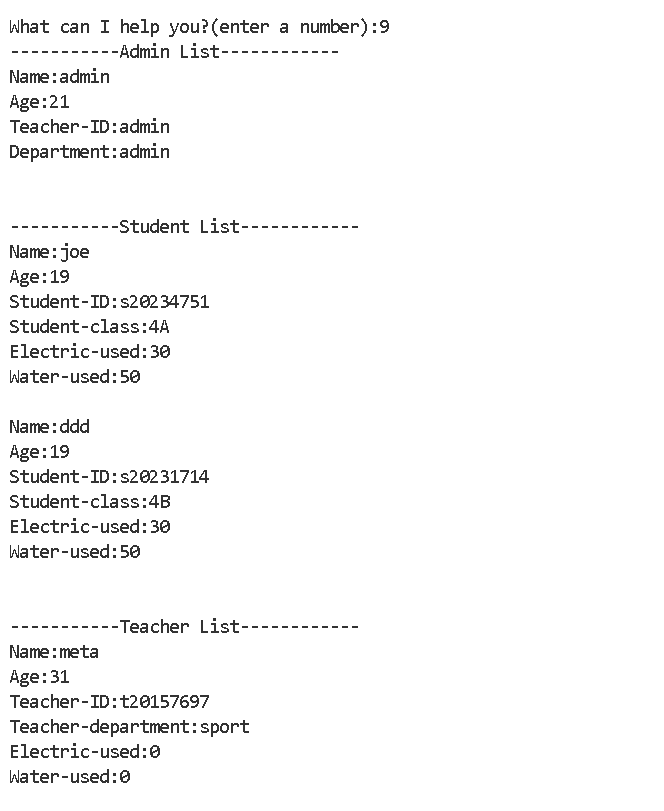
7.

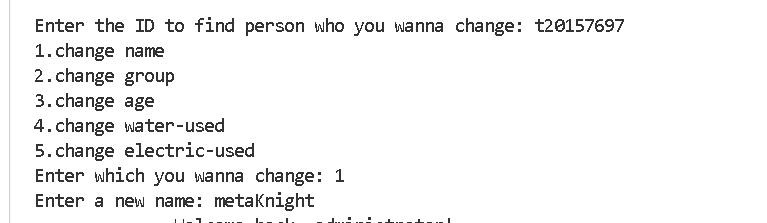


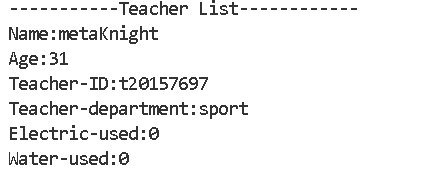
8.



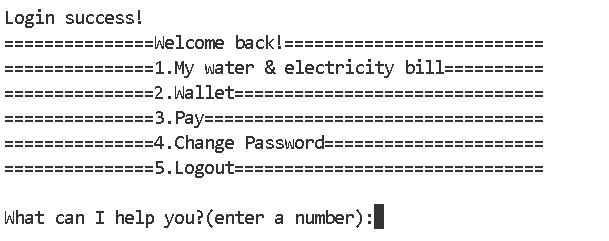
9.







用户界面：

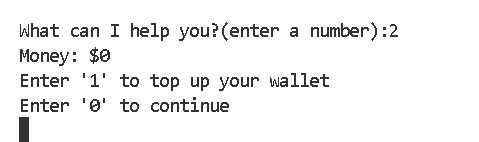


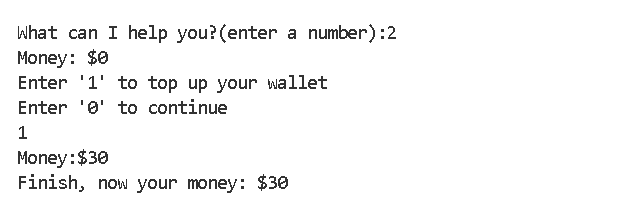
用户界面功能：

1.

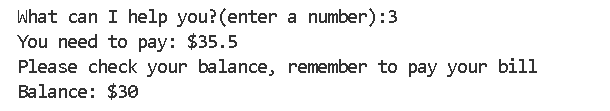


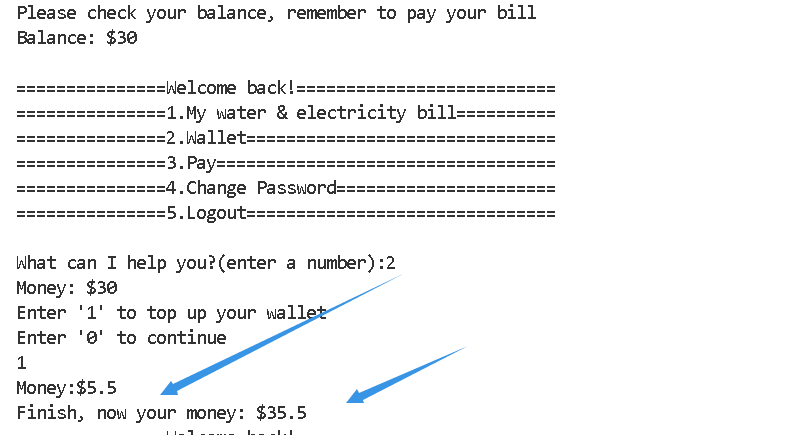
2.

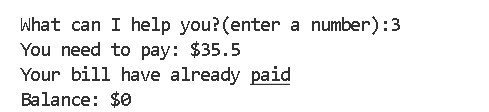




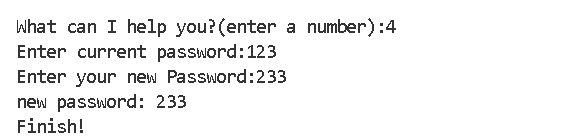
3.

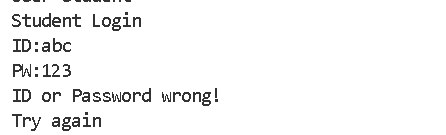


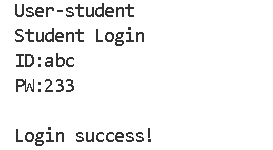




4.







2. 系统问题及其解决方案。

最常出现的问题就是输入输出时候的输入流未清理干净的问题，比如要求用户输入账户密码是采用getline来获取，但是需要提前知道调入这个getline之前有没有留在流里的字符没被舍弃，常见的例如cin捕获到回车符会停止输入捕捉，但是并不会丢弃回车符，会撤回会仍然留在输出流里面，导致下一个输入被跳过，最开始用的是cin.ignore()来无视一个字符，但是这样做会导致当流是空的时候，就会忽略掉用户输入的第一个字符，也就是最开始getline的时候用户输入abc，’a’字符会被忽视不被读取，造成严重错误。之后查找办法说可以利用cin.peek()来检测是否未空，当cin.peek()!=EOF时执行cin.ignore()来忽视掉流里的回车符，但是这样做依然没有显著效果。最后决定针对每次的cin单字符捕获后用getchar（）吃掉回车符。

除此以外就是switch里的变量被跳过处理问题，c++不允许在case里单独实现功能，需要改用函数来实现，因此每个case的单独功能实现都被改成调用函数实现。

对于数据的存储问题，用了C++的vector容器来存取数据，用一个类专门存放用户的账号密码信息，作为暂存的信息类，之后再把这些存放好的信息类导入进对应的成员类当中。

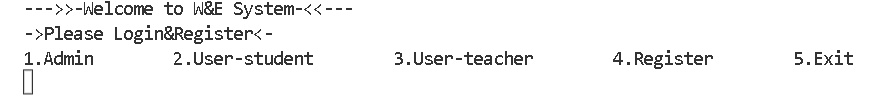
1. 可扩展功能及设计实现构想。

针对自动缴费方面，当用户的钱包余额满足需要扣除的费用时，会自动扣除功能

针对管理员可以单独创建新超级用户。

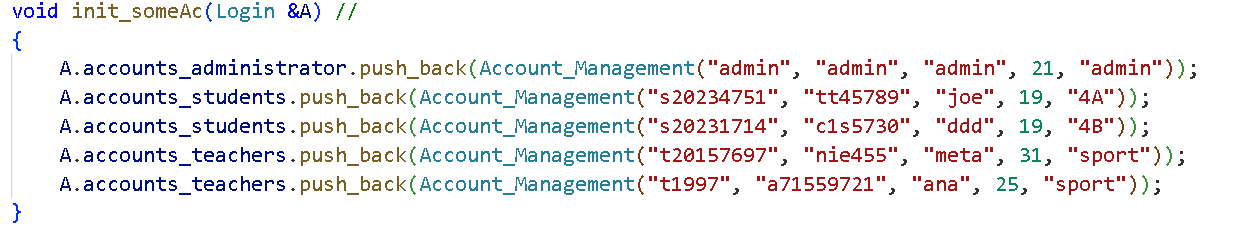
四、设计总结

在最开始的主页面设计，因为没完成函数的逐步退出功能，因此做成了while循环反复执行index的进入，模拟退出到主菜单的功能。



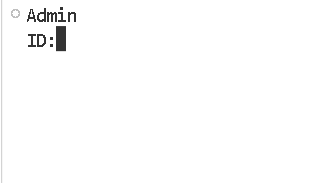
进入主菜单的页面，有五个功能，User的登陆选项为了减少分支带来的不断函数调用深入问题，以尽量减少深度调用改为平面化处理，对功能进行展开，因此有五个选项，分别对应管理员、教师、学生和第一次登录用的注册以及退出系统功能。

为了方便测试，在系统运行时先完成了对成员的预载入，作为NPC存在。

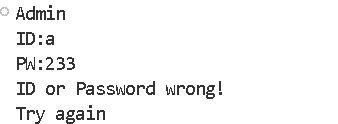


这些NPC的存在也可以作为检测之后登陆测试代码的选择，他们对应了不同的成员，有管理员、学生和老师。

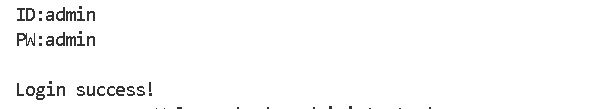
一号功能管理员登录



进入界面会看到提示输入ID和密码，同时这部分也完成了对数据的检测，也就是登录的密码是否正确。

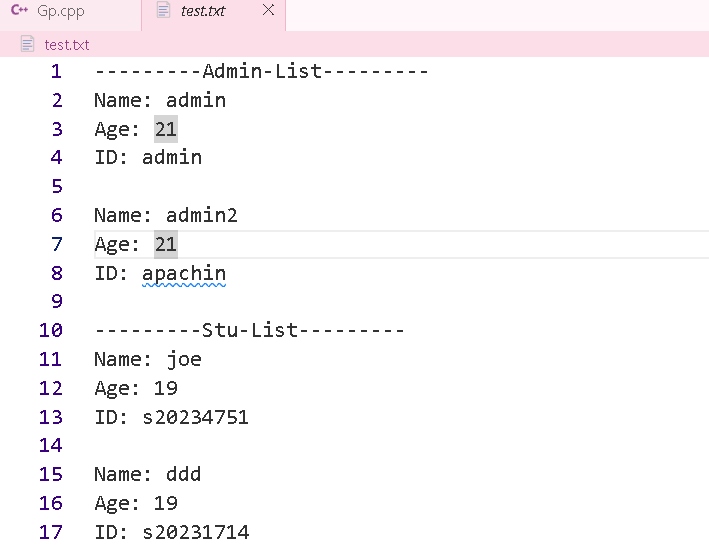


回顾init函数会发现最开始的管理员仅有admin一位，当输入不正确的信息，则会输出错误信息，提示登陆失败，完成了对登录的检测。



当输入正确的账号密码后，则会正确登录。

另外对每次的成员变动都会把记录更新在txt文档当中。

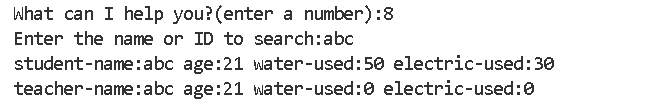


透过文档可以查看到最新的用户信息。

这里的ID是作为独立的成员识别码存在，ID是不允许重复的，每当新用户注册时，第一个填写的ID需要具有唯一性，当出现重复时就会出现错误，而名字是允许重名的，密码也是。

在管理员界面实现了九个功能，由于管理员是不需要交付水电费这些功能的，因此不在管理员里增设缴付功能，但是管理员可以透过编辑功能调整用户的水电费用量和成员信息。

对于八号功能查找上，因为是关键查找，所以不提供用户信息，仅留下一个输入栏让用户输入，用户有两个选择，如果输入id关键查找只会出来一个人，但是输入名字可以找出对应名字重复的人，提供了指定查找和范围查找。



可以看到他们来自学生和老师，但是因为同名都会被查找出来。

由于程序很多都是与查找相关，因此对于for迭代器的使用非常频繁，C++的auto变量可以很轻松的解决类型问题，不用再去翻看用了什么类型声明的变量。

在界面选择上，尽量用系统调用cls来完成对屏幕的清理，但是有些罗列出来的数据需要停留，因此有限地使用屏幕清理功能。

比较麻烦的是对成员变量的生成和导入，最开始想的是用成员类生成一个new对象建立起来后再把他push进去准备好的成员vector容器，但是实际操作的时候发现了成员并没有push进去，也没有报错，访问的时候vector为空，后来改成在push里直接生成构造函数反而可以成功push进去vector向量中。

对密码的检测方面，因为每一次的用户进入都需要经过login，也就是在login里的成员变量可以捕获到用户的登录信息，因此可以对用户进行身份验证，这个身份信息会一直保存直到下一次新的登录信息覆盖掉他，于是乎就可以透过访问变量来获取成员的账号密码，省去了跳去对成员的访问。

1. 参考文献

<https://cplusplus.com/reference/vector/vector/>

<https://cplusplus.com/reference/vector/vector/erase/>

<https://blog.csdn.net/u013654125/article/details/77321081>

<http://c.biancheng.net/view/338.html>

<https://www.runoob.com/cplusplus/cpp-files-streams.html>

<https://zhuanlan.zhihu.com/p/352961501>

<https://blog.csdn.net/chongzi_daima/article/details/103508585>

<https://blog.csdn.net/wangkai_123456/article/details/76598917>

六、附录

给出系统的核心代码

#include <iostream>

#include <string.h>

#include <algorithm>

#include <vector>

#include <string>

#include <fstream>

#include <stdlib.h>

using namespace std;

class Member

{

public:

    string name;

    int age;

    double used\_electric;

    double used\_water;

    void displayInfo()

    {

        cout << "name:" << name << endl;

        cout << "age:" << age << endl;

        cout << "electric:" << used\_electric << endl;

        cout << "water:" << used\_water << endl;

    }

};

class Student : public Member

{

public:

    Student(){}; // default

    Student(string name, int age, string cls, string id, double electric = 30, double water = 50, double money = 0)

    {

        this->name = name;

        this->age = age;

        this->student\_class = cls;

        this->student\_ID = id;

        this->used\_electric = electric;

        this->used\_water = water;

        this->money = money;

    }

    void displayInfo()

    {

        cout << "Name:" << name << endl;

        cout << "Age:" << age << endl;

        cout << "Student-ID:" << student\_ID << endl;

        cout << "Student-class:" << student\_class << endl;

        cout << "Electric-used:" << used\_electric << endl;

        cout << "Water-used:" << used\_water << endl;

    }

    string getStuID() const

    {

        return student\_ID;

    }

    string getStuCls() const

    {

        return student\_class;

    }

    double getMoney() const

    {

        return money;

    }

    void addMoney(double cash)

    {

        this->money += cash;

    }

    void changeCds(string newCds)

    {

        this->student\_class=newCds;

    }

private:

    string student\_class;

    string student\_ID;

    double money;

};

class Teacher : public Member

{

public:

    Teacher(){}; // default

    Teacher(string name, int age, string dept, string id, double electric = 0, double water = 0, double money = 30)

    {

        this->name = name;

        this->age = age;

        this->teacher\_department = dept;

        this->teacher\_ID = id;

        this->used\_electric = electric;

        this->used\_water = water;

        this->money = money;

    }

    void displayInfo()

    {

        cout << "Name:" << name << endl;

        cout << "Age:" << age << endl;

        cout << "Teacher-ID:" << teacher\_ID << endl;

        cout << "Teacher-department:" << teacher\_department << endl;

        cout << "Electric-used:" << used\_electric << endl;

        cout << "Water-used:" << used\_water << endl;

    }

    string getTecID() const

    {

        return teacher\_ID;

    }

    string getTecCls() const

    {

        return teacher\_department;

    }

    double getMoney() const

    {

        return money;

    }

    void addMoney(double cash)

    {

        this->money += cash;

    }

    void changeCds(string newCds)

    {

        this->teacher\_department=newCds;

    }

private:

    string teacher\_department;

    string teacher\_ID;

    double money;

};

class Administrator : public Member

{

public:

    Administrator(){}; // default

    Administrator(string name, int age, string dept, string id, double electric = 30, double water = 10, double money = 9999)

    {

        this->name = name;

        this->age = age;

        this->admin\_department = dept;

        this->admin\_ID = id;

        this->used\_electric = electric;

        this->used\_water = water;

        this->money = money;

    }

    void displayInfo()

    {

        cout << "Name:" << name << endl;

        cout << "Age:" << age << endl;

        cout << "Teacher-ID:" << admin\_ID << endl;

        cout << "Department:" << admin\_department << endl;

    }

    string getAdminID() const

    {

        return admin\_ID;

    }

    string getAdminCls() const

    {

        return admin\_department;

    }

private:

    string admin\_department;

    string admin\_ID;

    double money;

};

class Account\_Management

{

public:

    Account\_Management(const string &username, const string &password, const string &nickname, const int &age, const string &cds) // cds=class or department

    {

        this->username = username;

        this->password = password;

        this->nickname = nickname;

        this->age = age;

        this->cds = cds;

    }

    bool check(const string &username, const string &password) const

    {

        return this->username == username && this->password == password;

    }

    string getUsername() const

    {

        return username;

    }

    string getPassword() const

    {

        return password;

    }

    string getNickname() const

    {

        return nickname;

    }

    int getAge() const

    {

        return age;

    }

    string getCds() const

    {

        return cds;

    }

    void changePwd(string newPWD)

    {

        this->password = newPWD;

    }

private:

    string username;

    string password;

    int age;

    string nickname;

    string cds;

};

class Login

{

public:

    int login\_check;

    char user\_id[13];

    char user\_pw[13];

    int len\_userID;

    int len\_userPW;

    vector<Account\_Management> accounts\_students; // DATA SAVE

    vector<Account\_Management> accounts\_teachers;

    vector<Account\_Management> accounts\_administrator;

    void index()

    {

        // Index part

        cout << "--->>-Welcome to W&E System-<<---" << endl;

        while (1)

        {

            cout << "->Please Login&Register<-" << endl;

            cout << "1.Admin        2.User-student        3.User-teacher        4.Register        5.Exit" << endl;

            cin >> login\_check;

            if (login\_check == 1 || login\_check == 2 || login\_check == 3 || login\_check == 4)

            {

                system("cls");

                break;

            }

            if (login\_check == 5)

            {

                cout << "See you next time" << endl;

                exit(1);

            }

        }

        login();

    }

    void displayAccounts() const

    {

        cout << "-------------------------" << endl;

        cout << "Administrator List" << endl;

        cout << "-------------------------" << endl;

        for (const auto &account : accounts\_administrator)

        {

            cout << "Username: " << account.getUsername() << endl;

            cout << "Password: " << account.getPassword() << endl;

            cout << "Nickname:" << account.getNickname() << endl;

            cout << "Age:" << account.getAge() << endl;

            cout << "Class/Department:" << account.getCds() << endl;

            cout << endl;

        }

        cout << endl;

        cout << "-------------------------" << endl;

        cout << "Student List" << endl;

        cout << "-------------------------" << endl;

        for (const auto &account : accounts\_students)

        {

            cout << "Username: " << account.getUsername() << endl;

            cout << "Password: " << account.getPassword() << endl;

            cout << "Nickname:" << account.getNickname() << endl;

            cout << "Age:" << account.getAge() << endl;

            cout << "Class/Department:" << account.getCds() << endl;

            cout << endl;

        }

        cout << endl;

        cout << "-------------------------" << endl;

        cout << "Teacher List" << endl;

        cout << "-------------------------" << endl;

        for (const auto &account : accounts\_teachers)

        {

            cout << "Username: " << account.getUsername() << endl;

            cout << "Password: " << account.getPassword() << endl;

            cout << "Nickname:" << account.getNickname() << endl;

            cout << "Age:" << account.getAge() << endl;

            cout << "Class/Department:" << account.getCds() << endl;

            cout << endl;

        }

    }

    void Build() // data->member

    {

        for (auto it = accounts\_administrator.begin(); it < accounts\_administrator.end(); ++it) // ok

        {

            Mana\_Adm.push\_back(Administrator((\*it).getNickname(), (\*it).getAge(), (\*it).getCds(), (\*it).getUsername()));

        }

        // if(!Mana\_Adm.empty())

        // cout<<"success"<<endl;

        for (auto it = accounts\_students.begin(); it < accounts\_students.end(); ++it) // ok

        {

            Mana\_Stu.push\_back(Student((\*it).getNickname(), (\*it).getAge(), (\*it).getCds(), (\*it).getUsername()));

        }

        for (auto it = accounts\_teachers.begin(); it < accounts\_teachers.end(); ++it) // ok

        {

            Mana\_Tec.push\_back(Teacher((\*it).getNickname(), (\*it).getAge(), (\*it).getCds(), (\*it).getUsername()));

        }

    }

    void index\_login()

    {

        int n;

        if (card\_login == 0) // admin part

        {

            cout << "==============Welcome back, administrator!============" << endl;

            cout << "=================1.Add admin-user=====================" << endl;

            cout << "=================2.Delete admin-user==================" << endl;

            cout << "=================3.List of arrears====================" << endl;

            cout << "=================4.Delete user========================" << endl;

            cout << "=================5.View students======================" << endl;

            cout << "=================6.View teachers======================" << endl;

            cout << "=================7.View all===========================" << endl;

            cout << "=================8.Search=============================" << endl;

            cout << "=================9.Edit===============================" << endl;

            cout << "=================0.Logout=============================" << endl;

            cout << endl;

            cout << "What can I help you?(enter a number):";

            cin >> n;

            getchar();

            switch (n)

            {

            case 1:

                ADM\_f1();

                break;

            case 2:

                ADM\_f2();

                break;

            case 3:

                ADM\_f3();

                break;

            case 4:

                ADM\_f4();

                break;

            case 5:

                ADM\_f5();

                break;

            case 6:

                ADM\_f6();

                break;

            case 7:

                ADM\_f7();

                break;

            case 8:

                ADM\_f8();

                break;

            case 9:

                ADM\_f9();

                break;

            default:

                ADM\_f10();

                break;

            }

            if (n != 0)

            {

                index\_login();

            }

        }

        if (card\_login == 1 || card\_login == 2)

        {

            cout << "===============Welcome back!==========================" << endl;

            cout << "===============1.My water & electricity bill==========" << endl;

            cout << "===============2.Wallet===============================" << endl;

            cout << "===============3.Pay==================================" << endl;

            cout << "===============4.Change Password======================" << endl;

            cout << "===============5.Logout===============================" << endl;

            cout << endl;

            cout << "What can I help you?(enter a number):";

            cin >> n;

            getchar();

            switch (n)

            {

            case 1:

                USR\_f1();

                break;

            case 2:

                USR\_f2();

                break;

            case 3:

                USR\_f3();

                break;

            case 4:

                USR\_f4();

                break;

            case 5: // out exec

                break;

            default:

                cout << "Error input" << endl;

                index\_login();

            }

            if (n != 5)

            {

                index\_login();

            }

            USR\_f5();

        }

        index();

    }

    void transTxt() // save to txt

    {

        ofstream f("test.txt");

        f << "---------Admin-List---------" << endl;

        for (auto &it : Mana\_Adm)

        {

            f << "Name: " << it.name << endl;

            f << "Age: " << it.age << endl;

            f << "ID: " << it.getAdminID() << endl;

            f << endl;

        }

        f << "---------Stu-List---------" << endl;

        for (auto &it : Mana\_Stu)

        {

            f << "Name: " << it.name << endl;

            f << "Age: " << it.age << endl;

            f << "ID: " << it.getStuID() << endl;

            f << endl;

        }

        f << "---------Tec-List---------" << endl;

        for (auto &it : Mana\_Tec)

        {

            f << "Name: " << it.name << endl;

            f << "Age: " << it.age << endl;

            f << "ID: " << it.getTecID() << endl;

            f << endl;

        }

    }

private:

    vector<Administrator> Mana\_Adm;

    vector<Student> Mana\_Stu;

    vector<Teacher> Mana\_Tec;

    int card\_login = 0; // identity 0admin 1stu 2teach

    void login()

    {

        switch (login\_check)

        {

        case 1:

            cout << "Admin" << endl;

            Login\_Branch(login\_check);

            break;

        case 2:

            cout << "User-student" << endl;

            Login\_Branch(login\_check);

            break;

        case 3:

            cout << "User-teacher" << endl;

            Login\_Branch(login\_check);

            break;

        case 4:

            cout << "Register" << endl;

            Add\_Account(login\_check);

        default:

            index();

            break;

        }

    }

    void Login\_Branch(int login\_check)

    {

        char \*id\_login = new char[13]();

        int len\_id;

        char \*pwd\_login = new char[13]();

        int len\_pwd;

        if (login\_check == 2)

            cout << "Student Login" << endl;

        if (login\_check == 3)

            cout << "Teacher Login" << endl;

        // id

        cout << "ID:";

        cin.ignore();

        Login\_Info\_get(id\_login, len\_id);

        strcpy(user\_id, id\_login); // send to public

        len\_userID = len\_id;

        // pw

        cout << "PW:";

        Login\_Info\_get(pwd\_login, len\_pwd);

        strcpy(user\_pw, pwd\_login);

        len\_userPW = len\_pwd;

        // check is\_valid

        if (login\_check == 1) // admin part

        {

            bool login\_is\_valid = false;

            for (const auto &account : accounts\_administrator)

            {

                if (account.check(user\_id, user\_pw))

                {

                    login\_is\_valid = true;

                    break;

                }

            }

            if (login\_is\_valid)

            {

                cout << endl;

                cout << "Login success!" << endl;

                card\_login = 0;

                index\_login();

                cout << endl;

            }

            else

            {

                cout << "ID or Password wrong!" << endl;

                cout << "Try again" << endl;

                index();

            }

        }

        if (login\_check == 2) // student part

        {

            bool login\_is\_valid = false;

            for (const auto &account : accounts\_students)

            {

                if (account.check(user\_id, user\_pw))

                {

                    login\_is\_valid = true;

                    break;

                }

            }

            if (login\_is\_valid)

            {

                cout << endl;

                cout << "Login success!" << endl;

                card\_login = 1;

                index\_login();

                cout << endl;

            }

            else

            {

                cout << "ID or Password wrong!" << endl;

                cout << "Try again" << endl;

                index();

            }

        }

        if (login\_check == 3) // teacher part

        {

            bool login\_is\_valid = false;

            for (const auto &account : accounts\_teachers)

            {

                if (account.check(user\_id, user\_pw))

                {

                    login\_is\_valid = true;

                    break;

                }

            }

            if (login\_is\_valid)

            {

                cout << endl;

                cout << "Login success!" << endl;

                card\_login = 2;

                index\_login();

                cout << endl;

            }

            else

            {

                cout << "ID or Password wrong!" << endl;

                cout << "Try again" << endl;

                index();

            }

        }

        delete[] id\_login;

        delete[] pwd\_login;

        id\_login = pwd\_login = nullptr;

    }

    void Add\_Account(int login\_check)

    {

        char \*new\_id\_login = new char[13]();

        char \*tmp\_pwd = new char[13]();

        char \*new\_pwd\_login = new char[13]();

        int flag;

        cout << "Input '1' to create User-Student or input '2' to create User-Teacher" << endl; // branch

        while (1)

        {

            cin >> flag;

            getchar();

            if (flag > 2 || flag < 1)

            {

                cout << "invalid input" << endl;

                continue;

            }

            if (flag == 2 || flag == 1)

                break;

        }

        cout << "newID(Limit:13):";

        cin.getline(new\_id\_login, 13, '\n');

        for(auto&it:accounts\_administrator)

        {

            if(it.getUsername()==new\_id\_login)

            {

                cout << endl;

                cout << "Account already exist" << endl;

                cout << endl;

                exit(1);

            }

        }

        for(auto&it:accounts\_students)

        {

            if(it.getUsername()==new\_id\_login)

            {

                cout << endl;

                cout << "Account already exist" << endl;

                cout << endl;

                exit(1);

            }

        }

         for(auto&it:accounts\_teachers)

        {

            if(it.getUsername()==new\_id\_login)

            {

                cout << endl;

                cout << "Account already exist" << endl;

                cout << endl;

                exit(1);

            }

        }

        cout << "newPassword(Limit:13):";

        cin.getline(tmp\_pwd, 13, '\n');

        cout << "Password confirm:";

        cin.getline(new\_pwd\_login, 13, '\n');

        // check\_is\_valid?

        // send to AM

        // success register, addition materials ↓

        string temp\_nickname;

        int temp\_age;

        string temp\_cds; // class or department

        // id -> login username

        if (flag == 1)

        {

            if (strcmp(tmp\_pwd, new\_pwd\_login) == 0)

            {

                cout << "-----------Please enter the personal information------------" << endl;

                cout << "Your name:";

                cin >> temp\_nickname;

                cout << "Your age:";

                cin >> temp\_age;

                cout << "Your class/department:";

                cin >> temp\_cds;

                accounts\_students.push\_back(Account\_Management(new\_id\_login, new\_pwd\_login, temp\_nickname, temp\_age, temp\_cds));

                Mana\_Stu.push\_back(Student(temp\_nickname, temp\_age, temp\_cds, new\_id\_login));

            }

            else

            {

                cout << "Not the same password,please try again" << endl;

                index();

            }

        }

        if (flag == 2)

        {

            // send to AM

            if (strcmp(tmp\_pwd, new\_pwd\_login) == 0)

            {

                cout << "-----------Please enter the personal information------------" << endl;

                cout << "Your name:";

                cin >> temp\_nickname;

                cout << "Your age:";

                cin >> temp\_age;

                cout << "Your class/department:";

                cin >> temp\_cds;

                accounts\_teachers.push\_back(Account\_Management(new\_id\_login, new\_pwd\_login, temp\_nickname, temp\_age, temp\_cds));

                Mana\_Tec.push\_back(Teacher(temp\_nickname, temp\_age, temp\_cds, new\_id\_login));

            }

            else

            {

                cout << "Not the same password,please try again" << endl;

                index();

            }

        }

        transTxt();

        delete[] tmp\_pwd;

        delete[] new\_id\_login;

        delete[] new\_pwd\_login;

        tmp\_pwd = new\_id\_login = new\_pwd\_login = nullptr;

    }

    void Login\_Info\_get(char \*arr, int &len)

    {

        cin.getline(arr, 13, '\n');

        len = strlen(arr);

    }

    // switch part

    void ADM\_f1() // Add admin-user ok

    {

        string add\_temp\_nickname;

        int add\_temp\_age;

        string add\_temp\_id;

        string add\_temp\_pwd;

        cout << "Please enter your name:";

        getline(cin, add\_temp\_nickname);

        cout << "Please enter your age:";

        cin >> add\_temp\_age;

        getchar();

        cout << "Please enter your id:";

        getline(cin, add\_temp\_id);

        for(auto&it:Mana\_Adm)

        {

            if(it.getAdminID()==add\_temp\_id)

                {

                    cout<<"Error: ID already exist"<<endl;

                    cout<<endl;

                    index\_login();

                }

        }

        cout<<"Please enter your pwd: ";

        getline(cin,add\_temp\_pwd);

        Mana\_Adm.push\_back(Administrator(add\_temp\_nickname, add\_temp\_age, "admin", add\_temp\_id));

        accounts\_administrator.push\_back(Account\_Management(add\_temp\_id,add\_temp\_pwd,add\_temp\_nickname,add\_temp\_age,"admin"));

        cout << "============Now admin-list============" << endl;

        for (auto &it : Mana\_Adm)

        {

            it.displayInfo();

            cout << endl;

        }

        transTxt();

        cout << "Finish!" << endl;

    }

    void ADM\_f2() // delete admin ok

    {

        string d\_id;

        cout << "==========Find the ID you wanna delete===========" << endl;

        for (auto &admin : Mana\_Adm)

            cout << admin.getAdminID() << endl;

        cout << "Enter ID here:";

        getline(cin, d\_id);

        cout << "Your target is: " << d\_id << endl;

        for (auto it = Mana\_Adm.begin(); it < Mana\_Adm.end();)

        {

            if ((\*it).getAdminID() == d\_id)

            {

                it = Mana\_Adm.erase(it);

            }

            else

                ++it;

        }

        cout << "------------After delete list------------" << endl;

        for (auto &admin : Mana\_Adm)

            cout << admin.getAdminID() << endl;

        for (auto &stu : Mana\_Stu)

            cout << stu.getStuID() << endl;

        for (auto &tec : Mana\_Tec)

            cout << tec.getTecID() << endl;

        transTxt();

        cout << "FInish!!" << endl;

    }

    void ADM\_f3() // ListArrears ok

    {

        cout << "----------List of arrears(name)----------" << endl;

        for (auto it = Mana\_Stu.begin(); it < Mana\_Stu.end();)

        {

            if ((((\*it).used\_electric - 15) \* 0.5 + ((\*it).used\_water - 10) \* 0.7) > 0)

            {

                cout << (\*it).name << " energy bill: $" << ((\*it).used\_electric - 15) \* 0.5 + ((\*it).used\_water - 10) \* 0.7 << endl;

                ++it;

            }

            else

                ++it;

        }

        for (auto it = Mana\_Tec.begin(); it < Mana\_Tec.end();)

        {

            if ((((\*it).used\_electric - 15) \* 0.5 + ((\*it).used\_water - 10) \* 0.7) > 0)

            {

                cout << (\*it).name << " energy bill: $" << ((\*it).used\_electric - 15) \* 0.5 + ((\*it).used\_water - 10) \* 0.7 << endl;

                ++it;

            }

            else

                ++it;

        }

        cout << endl;

    }

    void ADM\_f4() // delete User ok

    {

        int n;

        string d\_id;

        cout << "if delete User-student enter '1' " << endl;

        cout << "if delete User-teacher enter '2' " << endl;

        cin >> n;

        getchar();

        if (n < 1 || n > 2)

        {

            cout << "Error input" << endl;

            return;

        }

        if (n == 1)

        {

            cout << "--------Student list---------" << endl;

            for (auto &stu : Mana\_Stu)

                cout << stu.getStuID() << endl;

            cout << "Enter ID to delete" << endl;

            getline(cin, d\_id);

            cout << "Your target is: " << d\_id << endl;

            for (auto it = Mana\_Stu.begin(); it < Mana\_Stu.end();)

            {

                if ((\*it).getStuID() == d\_id)

                {

                    it = Mana\_Stu.erase(it);

                }

                else

                    ++it;

            }

            cout << "------------After delete list------------" << endl;

            for (auto &admin : Mana\_Adm)

                cout << admin.getAdminID() << endl;

            for (auto &stu : Mana\_Stu)

                cout << stu.getStuID() << endl;

            for (auto &tec : Mana\_Tec)

                cout << tec.getTecID() << endl;

            transTxt();

            cout << "FInish!!" << endl;

        }

        if (n == 2)

        {

            cout << "--------Teacher list---------" << endl;

            for (auto &tec : Mana\_Tec)

                cout << tec.getTecID() << endl;

            cout << "Enter ID to delete" << endl;

            getline(cin, d\_id);

            cout << "Your target is: " << d\_id << endl;

            for (auto it = Mana\_Tec.begin(); it < Mana\_Tec.end();)

            {

                if ((\*it).getTecID() == d\_id)

                {

                    it = Mana\_Tec.erase(it);

                }

                else

                    ++it;

            }

            cout << "------------After delete list------------" << endl;

            for (auto &admin : Mana\_Adm)

                cout << admin.getAdminID() << endl;

            for (auto &stu : Mana\_Stu)

                cout << stu.getStuID() << endl;

            for (auto &tec : Mana\_Tec)

                cout << tec.getTecID() << endl;

            transTxt();

            cout << "FInish!!" << endl;

        }

    }

    void ADM\_f5() // view stu ok

    {

        cout << "-----------Student List------------" << endl;

        for (auto &stu : Mana\_Stu)

        {

            stu.displayInfo();

            cout << endl;

        }

        cout << endl;

    }

    void ADM\_f6() // view Tec ok

    {

        cout << "-----------Teacher List------------" << endl;

        for (auto &tec : Mana\_Tec)

        {

            tec.displayInfo();

            cout << endl;

        }

        cout << endl;

    }

    void ADM\_f7() // viewAll ok

    {

        cout << "-----------Admin List------------" << endl;

        for (auto &it : Mana\_Adm)

        {

            it.displayInfo();

            cout << endl;

        }

        cout << endl;

        ADM\_f5();

        ADM\_f6();

    }

    void ADM\_f8() // searchKey ok

    {

        string key;

        cout << "Enter the name or ID to search:";

        getline(cin, key);

        for (const auto &admin : Mana\_Adm)

        {

            if (admin.name == key)

                cout << "admin-name:" << admin.name << " age:" << admin.age << endl;

            else if (admin.getAdminID() == key)

                cout << "admin-name:" << admin.name << " age:" << admin.age << endl;

        }

        // stu list

        for (const auto &stu : Mana\_Stu)

        {

            if (stu.name == key)

                cout << "student-name:" << stu.name << " age:" << stu.age << " water-used:" << stu.used\_water << " electric-used:" << stu.used\_electric << endl;

            else if (stu.getStuID() == key)

                cout << "student-name:" << stu.name << " age:" << stu.age << " water-used:" << stu.used\_water << " electric-used:" << stu.used\_electric << endl;

        }

        // tec list

        for (const auto &tec : Mana\_Tec)

        {

            if (tec.name == key)

                cout << "teacher-name:" << tec.name << " age:" << tec.age << " water-used:" << tec.used\_water << " electric-used:" << tec.used\_electric << endl;

            else if (tec.getTecID() == key)

                cout << "teacher-name:" << tec.name << " age:" << tec.age << " water-used:" << tec.used\_water << " electric-used:" << tec.used\_electric << endl;

        }

        cout<<endl;

    }

    void ADM\_f9() //Edit

    {

        string key;

        string change\_name;

        string change\_cds;

        int change\_age;

        double change\_water;

        double change\_elec;

        int n;

        ADM\_f7();

        cout<<"Enter the ID to find person who you wanna change: ";

        getline(cin,key);

        cout<<"1.change name"<<endl;

        cout<<"2.change group"<<endl;

        cout<<"3.change age"<<endl;

        cout<<"4.change water-used"<<endl;

        cout<<"5.change electric-used"<<endl;

        cout<<"Enter which you wanna change: ";

        cin>>n;

        getchar();

        for(auto& it:Mana\_Stu)

        {

            if(key==it.getStuID())

            {

                switch(n)

                {

                    case 1: //change name

                        cout<<"Enter a new name: ";

                        getline(cin,change\_name);

                        it.name=change\_name;

                        break;

                    case 2: //change group

                        cout<<"Enter a new group: ";

                        getline(cin,change\_cds);

                        it.changeCds(change\_cds);

                        break;

                    case 3://change age

                        cout<<"Enter a new age: ";

                        cin>>change\_age;

                        getchar();

                        it.age=change\_age;

                        break;

                    case 4://change water

                        cout<<"Change water-used: ";

                        cin>>change\_water;

                        getchar();

                        it.used\_water=change\_water;

                        break;

                    case 5://change elec

                        cout<<"Change elec-used: ";

                        cin>>change\_elec;

                        getchar();

                        it.used\_electric=change\_elec;

                        break;

                    default:

                        break;

                }

                transTxt();

                cout<<"Finish!"<<endl;

            }

        }

        for(auto& it:Mana\_Tec)

        {

            if(key==it.getTecID())

            {

                 switch(n)

                {

                    case 1: //change name

                        cout<<"Enter a new name: ";

                        getline(cin,change\_name);

                        it.name=change\_name;

                        break;

                    case 2: //change group

                        cout<<"Enter a new group: ";

                        getline(cin,change\_cds);

                        it.changeCds(change\_cds);

                        break;

                    case 3://change age

                        cout<<"Enter a new age: ";

                        cin>>change\_age;

                        getchar();

                        it.age=change\_age;

                        break;

                    case 4://change water

                        cout<<"Change water-used: ";

                        cin>>change\_water;

                        getchar();

                        it.used\_water=change\_water;

                        break;

                    case 5://change elec

                        cout<<"Change elec-used: ";

                        cin>>change\_elec;

                        getchar();

                        it.used\_electric=change\_elec;

                        break;

                    default:

                        break;

                }

            }

        }

    }

    void ADM\_f10() // exit ok

    {

        cout << "Logout" << endl;

        return;

    }

    void USR\_f1() // energy bill ok

    {

        double sum;

        for (auto &it : Mana\_Stu)

        {

            if (it.getStuID() == user\_id)

            {

                sum = it.used\_water + it.used\_electric;

                cout << "You need to pay: $" << sum << " in energy fee" << endl;

            }

        }

        for (auto &it : Mana\_Tec)

        {

            if (it.getTecID() == user\_id)

            {

                sum = it.used\_water + it.used\_electric;

                cout << "You need to pay: $" << sum << " in energy fee" << endl;

            }

        }

    }

    void USR\_f2() // money ok

    {

        int t;

        double cash = 0;

        for (auto &it : Mana\_Stu)

        {

            if (it.getStuID() == user\_id)

            {

                cout << "Money: $" << it.getMoney() << endl;

                cout << "Enter '1' to top up your wallet" << endl;

                cout << "Enter '0' to continue" << endl;

                cin >> t;

                getchar();

                if (t == 1)

                {

                    cout << "Money:$";

                    cin >> cash;

                    getchar();

                    it.addMoney(cash);

                    cout << "Finish, now your money: $" << it.getMoney() << endl;

                }

                else

                    return;

            }

        }

        for (auto &it : Mana\_Tec)

        {

            if (it.getTecID() == user\_id)

            {

                cout << "Money: $" << it.getMoney() << endl;

                cout << "Enter '1' to top up your wallet" << endl;

                cout << "Enter '0' to continue" << endl;

                cin >> t;

                getchar();

                if (t == 1)

                {

                    cout << "Money:$";

                    cin >> cash;

                    getchar();

                    it.addMoney(cash);

                    cout << "Finish, now your money: $" << it.getMoney() << endl;

                }

                else

                    return;

            }

        }

    }

    void USR\_f3() // pay

    {

        double need\_pay;

        for (auto &it : Mana\_Stu)

        {

            if (it.getStuID() == user\_id)

            {

                need\_pay = (it.used\_electric - 15) \* 0.5 + (it.used\_water - 10) \* 0.7;

                if (need\_pay > 0)

                {

                    cout << "You need to pay: $" << need\_pay << endl;

                    if (it.getMoney() >= need\_pay)

                    {

                        cout << "Your bill have already paid" << endl;

                        it.addMoney((-1) \* need\_pay);

                        cout << "Balance: $" << it.getMoney() << endl;

                        // do clear energy bill

                        it.used\_electric = 0;

                        it.used\_water = 0;

                    }

                    if (it.getMoney() < need\_pay)

                    {

                        cout << "Please check your balance, remember to pay your bill" << endl;

                        cout << "Balance: $" << it.getMoney() << endl;

                        cout<<endl;

                    }

                }

            }

        }

        for (auto &it : Mana\_Tec)

        {

            if (it.getTecID() == user\_id)

            {

                need\_pay = (it.used\_electric - 15) \* 0.5 + (it.used\_water - 10) \* 0.7;

                if (need\_pay > 0)

                {

                    cout << "You need to pay: $" << need\_pay << endl;

                    if (it.getMoney() >= need\_pay)

                    {

                        cout << "Your bill have already paid" << endl;

                        it.addMoney((-1) \* need\_pay);

                        cout << "Balance: $" << it.getMoney() << endl;

                        it.used\_electric = 0;

                        it.used\_water = 0;

                    }

                    if (it.getMoney() < need\_pay)

                    {

                        cout << "Please check your balance, remember to pay your bill" << endl;

                        cout << "Balance: $" << it.getMoney() << endl;

                    }

                }

            }

        }

    }

    void USR\_f4() // change shadow

    {

        string temp\_pwd;

        for (auto &it : Mana\_Stu) // stu

        {

            if (user\_id == it.getStuID())

            {

                cout << "Enter current password:";

                getline(cin, temp\_pwd);

                if (user\_pw == temp\_pwd)

                {

                    cout << "Enter your new Password:";

                    getline(cin, temp\_pwd);

                    for (auto &it : accounts\_students)

                    {

                        if (it.getUsername() == user\_id) // primary key

                        {

                            it.changePwd(temp\_pwd);

                            cout << "new password: " << it.getPassword() << endl;

                            cout << "Finish!" << endl;

                        }

                    }

                }

            }

        }

        for (auto &it : Mana\_Tec) // stu

        {

            if (user\_id == it.getTecID())

            {

                cout << "Enter current password:";

                getline(cin, temp\_pwd);

                for (auto &it : accounts\_teachers)

                {

                    if (it.getUsername() == user\_id)

                    {

                        it.changePwd(temp\_pwd);

                        cout << "new password: " << it.getPassword() << endl;

                        cout << "Finish!" << endl;

                    }

                }

            }

        }

        transTxt();

    }

    void USR\_f5() // exit ok

    {

        cout << "Logout" << endl;

        return;

    }

};

void init\_someAc(Login &A) //

{

    A.accounts\_administrator.push\_back(Account\_Management("admin", "admin", "admin", 21, "admin"));

    A.accounts\_students.push\_back(Account\_Management("s20234751", "tt45789", "joe", 19, "4A"));

    A.accounts\_students.push\_back(Account\_Management("s20231714", "c1s5730", "ddd", 19, "4B"));

    A.accounts\_teachers.push\_back(Account\_Management("t20157697", "nie455", "meta", 31, "sport"));

    A.accounts\_teachers.push\_back(Account\_Management("t1997", "a71559721", "ana", 25, "sport"));

}

int main()

{

    int identity;

    Login login\_used;

    init\_someAc(login\_used);

    login\_used.Build(); // build member info

    login\_used.transTxt();

    login\_used.index();

    return 0;

}