

**信息与软件工程学院**

**项目报告**

课程名称： 程序设计项目实践（BPLF）

学 期： 2019-2020第1学期

项目名称： Linux专用密码本

授课教师： 白忠建

|  |  |  |
| --- | --- | --- |
| 序号 | 学号 | 姓名 |
| 1（组长） | 2019091601026 | 崔冬辰 |
| 2 | 2019091601021 | 李尚哲 |
| 3 | 2019091601001 | 唐昊哲 |
| 4 |  |  |
| 5 |  |  |

目录

[1 项目简介 3](#_Toc27580295)

[1.1 考核方式 3](#_Toc27580296)

[1.2 项目题目及内容简介 3](#_Toc27580297)

[1.3 项目组成员与分工 3](#_Toc27580298)

[2 需求分析 4](#_Toc27580299)

[2.1 选题的依据 4](#_Toc27580300)

[2.2 功能需求 4](#_Toc27580301)

[3 系统设计 5](#_Toc27580302)

[3.1 总体设计 5](#_Toc27580303)

[3.2 模块设计 5](#_Toc27580304)

[4 系统实现 7](#_Toc27580305)

[4.1 主函数 7](#_Toc27580306)

[4.2 其他函数 7](#_Toc27580307)

[5 功能测试 8](#_Toc27580308)

[6 总结 9](#_Toc27580309)

# 1 项目简介

## 1.1 考核方式

总成绩 = 项目和项目文档成绩(40%) + 汇报幻灯片成绩(20%)

+ 表达能力(20%) + 团队合作(20%)

## 1.2 项目题目及内容简介

项目题目：为Linux用户打造的密码本

内容简介：可以在Linux环境下实现屏幕操作的密码管理本，对比于普通程序，能够为Linux用户带来更好的体验。用户可以进行对账户密码进行查看，存储，删除，以及通过账户名或登录环境来查找密码。

Github仓库地址：<https://github.com/Erewhon-Lucy/BPLP_demo>

## 1.3 项目组成员与分工

崔冬辰：统筹团队合作，程序框架，功能测试，项目文档，项目汇报幻灯片

李尚哲：face函数，searchByName函数，searchByArea函数，print函数，append函数，delete函数

唐昊哲：debug，算法流程图，saveTofile函数，

# 2 需求分析

## 2.1 选题的依据

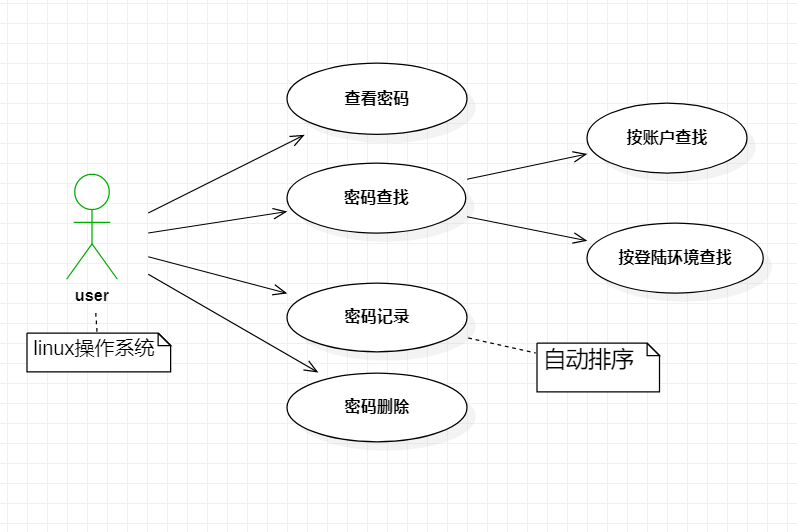
在互联网发达的当下，登录各种各样的网站以及app已经是我们的日常。很多情况下我们可能会忘记自己当初设定的账户和密码，因此我们需要制作一个密码本，用以记录自己的账户和密码并能够对密码本进行修改。

我们的密码本为Linux用户开发。由于Linux用户在运行程序时，命令行界面颜值比较低，为了给用户带来更好的体验，我们将系统制作成类似windows系统下的窗口。

基于上述分析，我们将制作一个能够在Linux系统下实现屏幕操作的密码管理本。

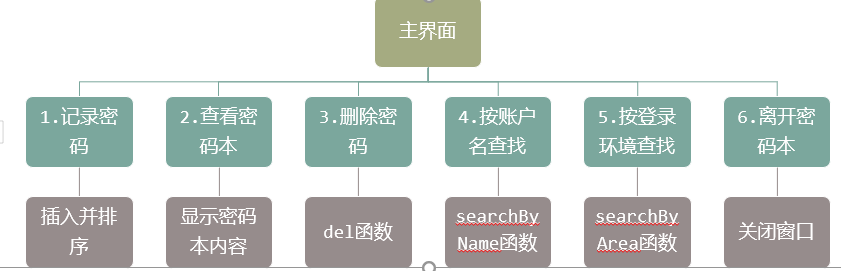
## 2.2 功能需求

使用UML用例图说明系统功能：



# 3 系统设计

## 3.1 总体设计



## 3.2 模块设计

头文件 head.h

#include<stdio.h>

#include<ncurses.h>

#include<signal.h>

#include <stdlib.h>

#include <string.h>

#include <stdbool.h>

#include <locale.h>

#define LEN sizeof(struct student)

struct student

{

    char name[50];  //用户名

    char code[21];   //密码

    char answer[5];   //密保答案

    char area[100]; //登录环境

    struct student \*next;

};

void line()//分割线函数

{

    for (int i = 0; i < (COLS / 5) ; i++)

    {

        addstr(" ^\_^ ");

    }

}

主文件main.c

#include "head.h"

void face();

void print(struct student \*head);

struct student \*append(struct student \*head);

struct student \*del(struct student \*head);

void searchByName(struct student \*head);

void searchByArea(struct student \*head);

void saveToFile(struct student \*head, const char \*filename);

Makefile

sources = main.c

ccflags = -Wall -g -fsanitize=address

CC = gcc

target = b

link = ncursesw

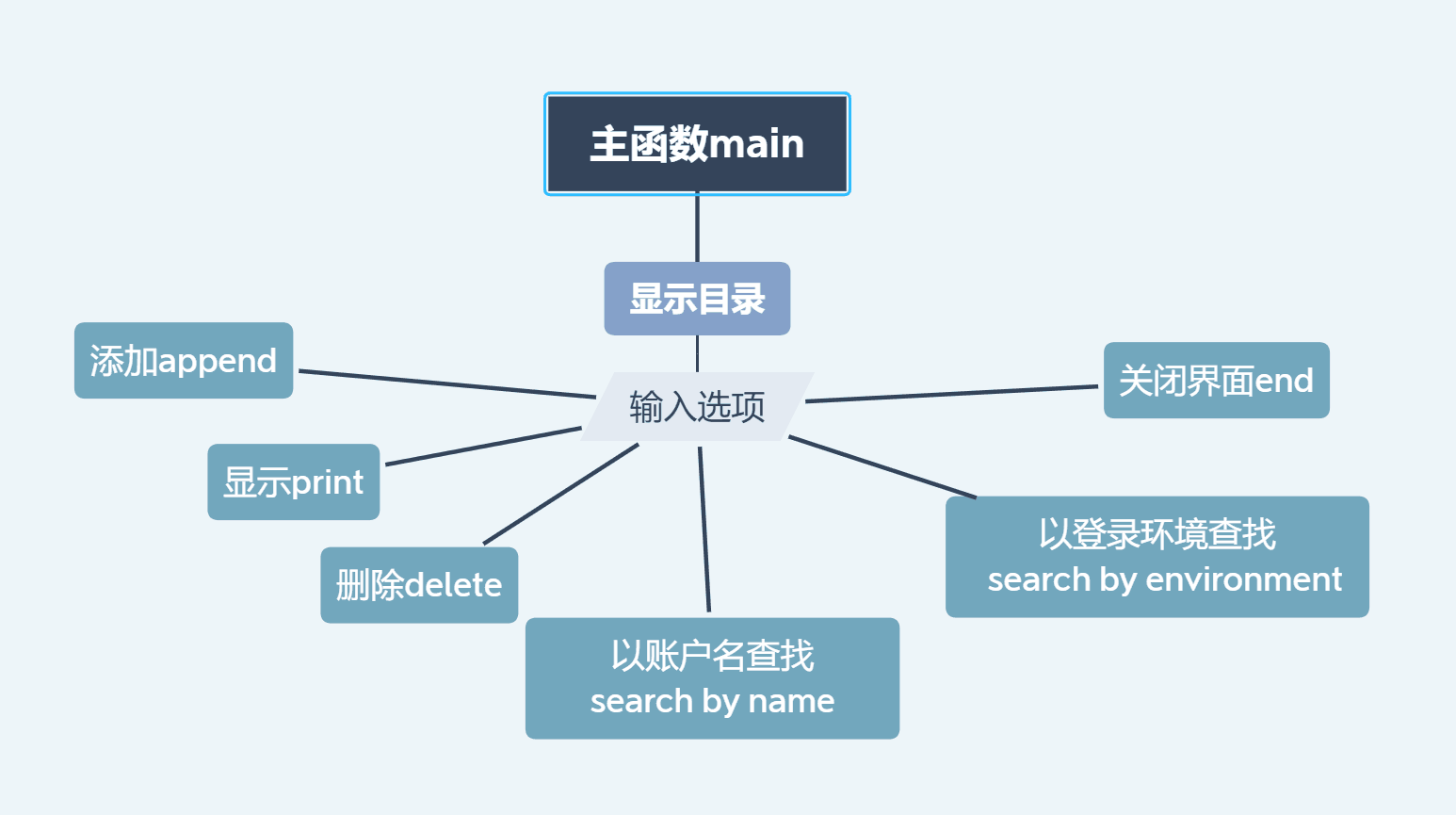
all:

    $(CC) $(sources) $(ccflags) -o $(target) -l$(link)

# 4 系统实现

## 4.1 主函数

用程序流程图说明实现思路



关键代码分析

int main()

{

struct student \*head = NULL;

int input;

setlocale(LC\_ALL, "");

while (1)

{

face();

scanw("%d", &input);

printw("%d", input);

getch();

switch (input)

{

case 1:

head = append(head);

break;

case 2:

print(head);

break;

case 3:

head = del(head);

break;

case 4:

searchByName(head);

break;

case 5:

searchByArea(head);

break;

case 6:

clear();

endwin();

return 0;

break;

}

clear();

}

getch();

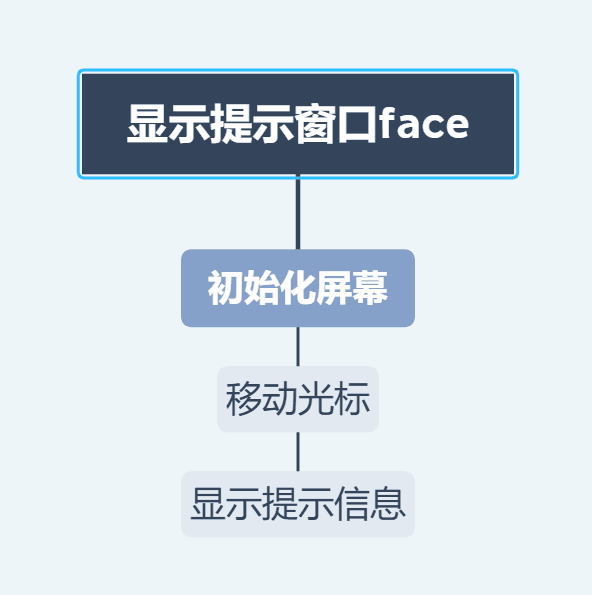
endwin();

return 0;

}

## 4.2 其他函数

Face函数



void face()

{

initscr(); //初始化屏幕

line();

move(2, COLS / 2 - 2); //移动光标

addstr("你好"); //输出字符串

move(3, COLS / 2 - 11);

addstr("我是一个没有感情的密码本\n");

line();

move(5, COLS / 2 - 4);

addstr("请操♂作我\n\n");

addstr("1.记下一个密码\n");

addstr("2.看看我的密码本\n");

addstr("3.删除一组密码\n");

addstr("4.按账户名找到一个密码\n");

addstr("5.按登录环境查找一个密码\n");

addstr("6.离开没有感情的密码本\n\n");

line();

move(15, COLS / 2 - 5);

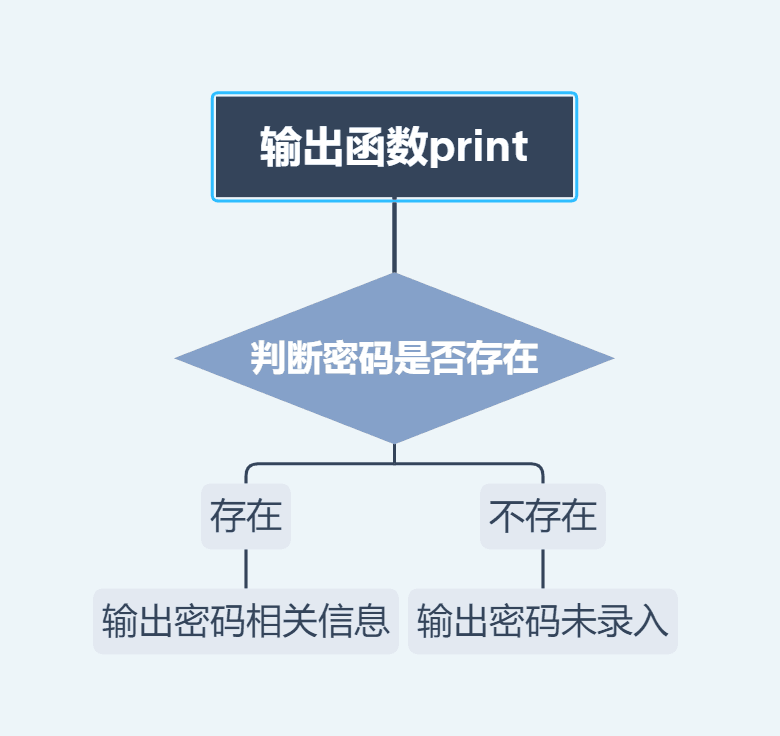
addstr("开始操♂作吧！\n");

addstr("你的操作：");

echo(); /

}

Print函数



void print(struct student \*head)

{

    struct student \*p;

    p = head;

    clear();

    if (head != NULL)

        do

        {

            printw("你的账户名字:%s\n", p->name);

            printw("这个账户的密码:%s\n", p->code);

            printw("密保问题:%s\n", p->answer);

            printw("账户是用来登录这个的:%s\n", p->area);

            line();

            p = p->next;

        } while (p != NULL);

    else

    {

        addstr("你傻了你的密码没保存在这\n\n");

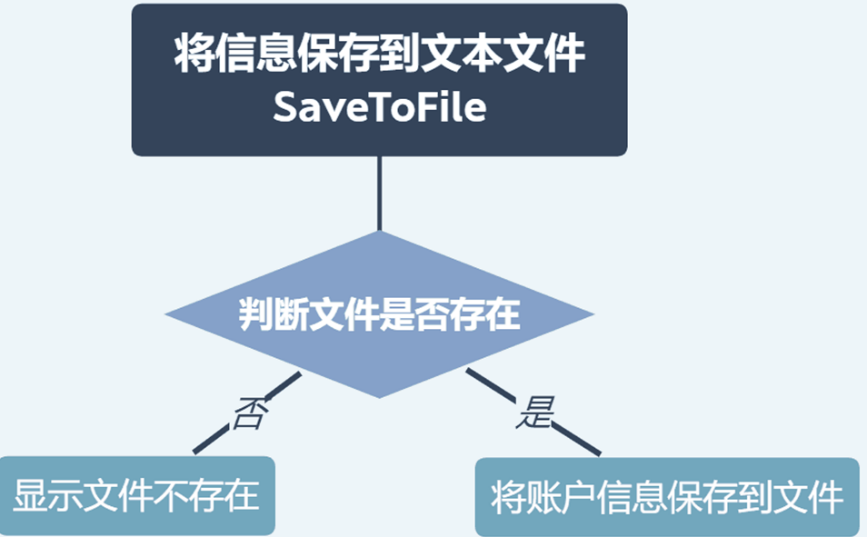
        line();

    }

    getch();

}

SaveToFile函数



void saveToFile(struct student \*head,const char \*filename)

{

    FILE \*fp;

    fp=fopen(filename,"wt");

    if (fp==NULL)

    {

        printf("cannot find %s.",filename);

        exit(1);

    }

    while (head!=NULL)

    {

        fprintf(fp,"%s\n",head->name);

        fprintf(fp,"%s\n",head->code);

        fprintf(fp,"%s\n",head->answer);

        fprintf(fp,"%s\n",head->area);

        fprintf(fp,"\n");

        head=head->next;

    }

    fclose(fp);

}

List\_create\_from\_file函数



void list\_create\_from\_file(struct student \*head,const char \*filename)

{

    FILE \*fp;

    fp=fopen(filename,"rt");

    if (fp==NULL)

    {

        printf("cannot find %s.", filename);

        exit(1);

    }

    while (!feof(fp))

    {

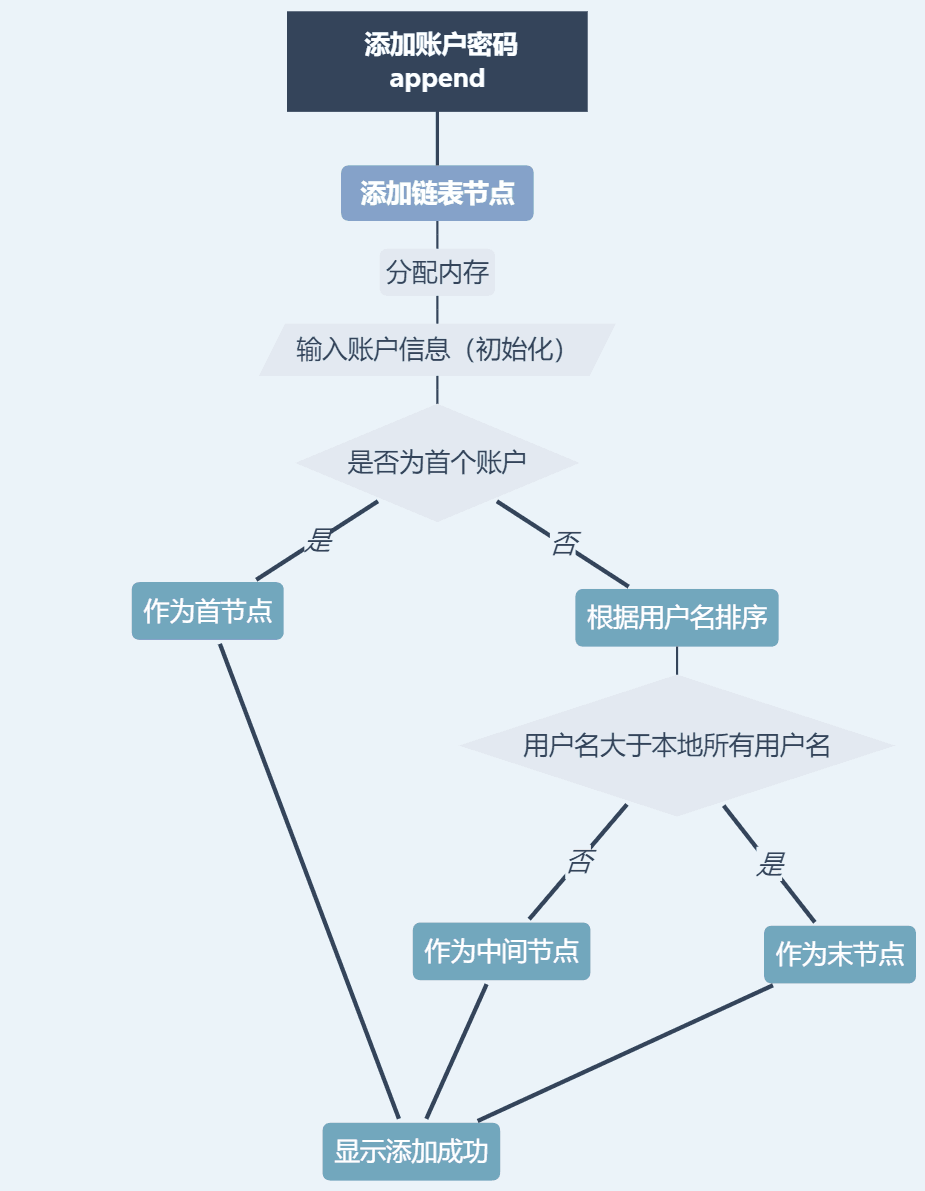
        fscanf(fp," %s %s %s %s",head->name,head->code,head->answer,head->area);

    }

    fclose(fp);

}

Append函数



struct student \*append(struct student \*head)

{

    struct student \*p0 = NULL, \*p1, \*p2;

    p1 = head;

    p2 = head;

    clear();

    addstr(" 你能在这里保存你愚蠢脑瓜记不住的密码\n");

    line();

    p0 = (struct student \*)malloc(LEN);  //p0为暂存新数据的指针

    addstr("你的账户:\n");

    scanw("%s", p0->name);

    addstr("你的密码:\n");

    scanw("%s", p0->code);

    addstr("密保问题:\n");

    scanw("%s", p0->answer);

    addstr("顺便记一下它在哪登录免得你的憨憨脑子忘记:\n");

    scanw("%s", p0->area);

    //根据用户名排序

    if (head == NULL)

    {

        head = p0;

        p0->next = NULL;

    }

    //

    else

    {

        while ((strcmp(p0->name, p1->name) > 0) && (p1->next != NULL))

        {

            p2 = p1;

            p1 = p1->next;

        }

        if ((strcmp(p0->name, p1->name)) <= 0)

        {

            if (head == p1)

            {

                head = p0;

                head->next=p1;

            }

            else

            {

                p2->next = p0;

                p0->next = p1;

            }

        }

        else

        {

            p1->next = p0;

            p0->next = NULL;

        }

    }

    printw("恭喜你!!可以从脑子里忘记你的密码了 \n"); //恭喜你 可以从脑子里忘记你的密码了

    saveToFile(head,"1.txt");

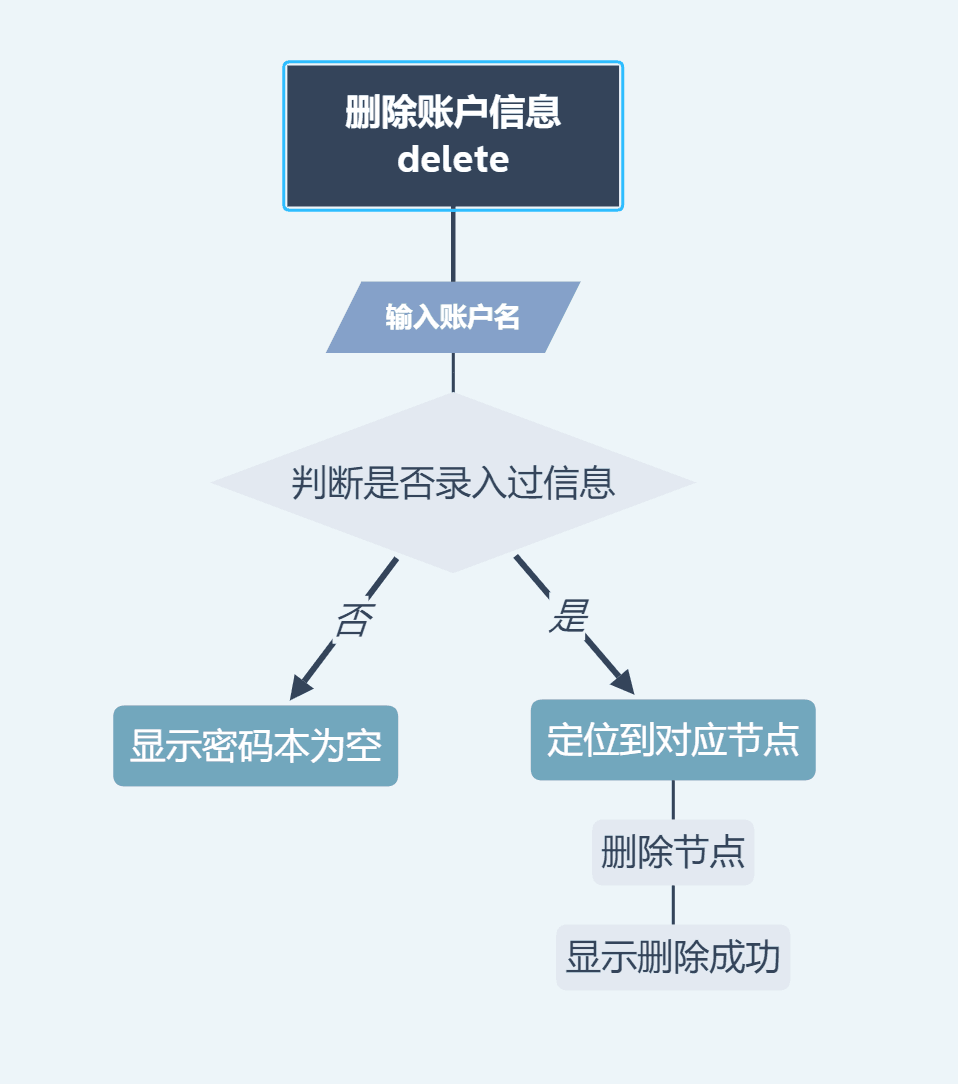
    line();

    getch();

    return (head);

}

Delete函数



struct student \*del(struct student \*head)

{

    struct student \*p1, \*p2;

    char name[100];

    clear();

    printw("你哪个号没了？:"); //你哪个号没了？

    scanw("%s", &name);

    p1 = head;

    if (head == NULL)

    {

        printw("你还没临幸过这个莫得感情的密码本\n"); //你还没临幸过这个莫得感情的密码本

        line();

        return (head);

    }

    while (p1 != NULL)

    {

        if (strcmp(p1->name, name) == 0)

        {

            if (p1 == head)

                head = p1->next;

            else

                p2->next = p1->next;

            free(p1);

            printw("恭喜！你号没了!!\n"); //你号没了

            //save

            getch();

            return (head);

        }

        p2 = p1;

        p1 = p1->next;

    }

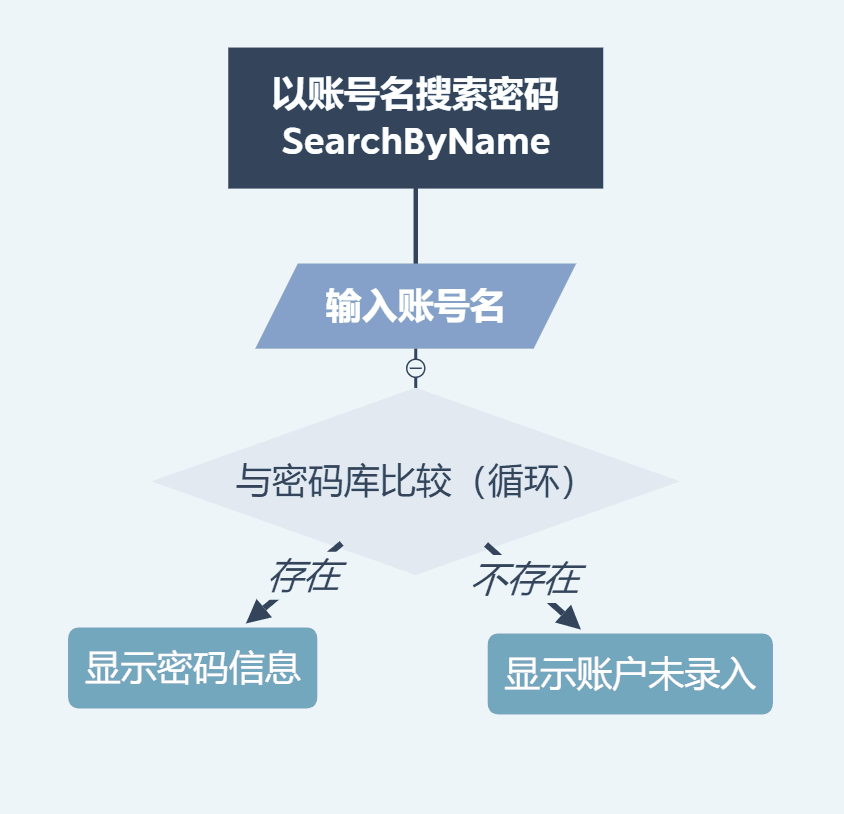
    printw("你记错了你根本就没来登记过它\n"); //你记错了你根本就没来登记过它

    getch();

    return (head);

}

SearchByName函数



void searchByName(struct student \*head)

{

    char name[50];

    struct student \*p1, \*p2;

    p1 = head;

    p2 = p1;

    clear();

    addstr("你要回忆的密码 它的账户叫啥\n");

    line();

    scanw("%s", &name);

    while (p1 != NULL)

    {

        if (strcmp(p1->name, name) == 0)

        {

            printw("它的密码是:%s\n", p1->code);

            printw("密保答案是:%s\n", p1->answer);

            printw("它用来登录这个:%s\n", p1->area);

            getch();

            break;

        }

        p2 = p1;

        p1 = p1->next;

    }

    if (p1 == NULL)

        printw("醒醒你还没登记过这个账户 !!\n");

}

SearchByArea函数



void searchByArea(struct student \*head)

{

    char area[50];

    struct student \*p1, \*p2;

    p1 = head;

    p2 = p1;

    clear();

    addstr("你要回忆的密码 在哪里登录\n");

    line();

    scanw("%s", &area);

    while (p1 != NULL)

    {

        if (strcmp(p1->area, area) == 0)

        {

            printw("它的账户是:%s\n", p1->name);

            printw("它的密码是:%s\n", p1->code);

            printw("密保答案是:%s\n", p1->answer);

            getch();

            break;

        }

        p2 = p1;

        p1 = p1->next;

    }

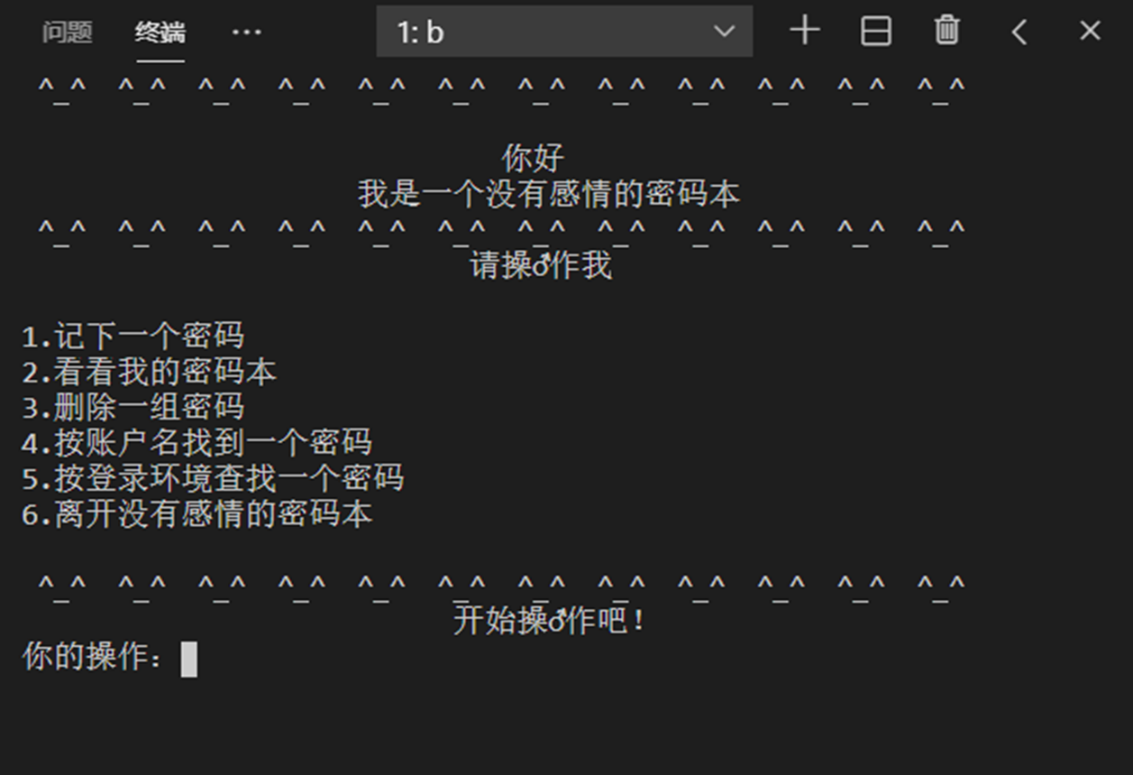
    if (p1 == NULL)

        printw("醒醒你还没登记过这个账户 !!\n");

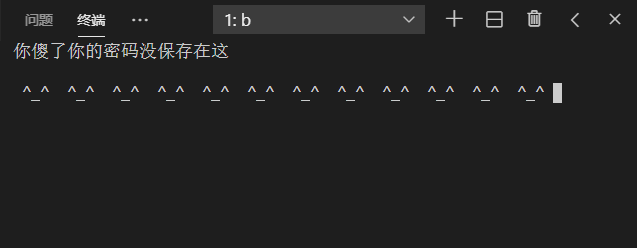
}

# 5 功能测试

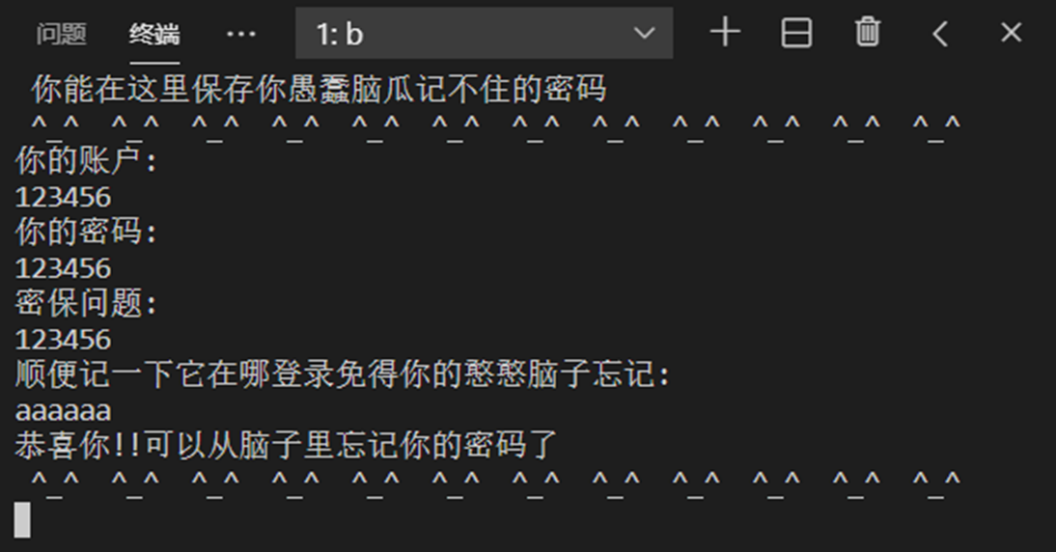
主界面显示



当密码本为空时选2查看密码本

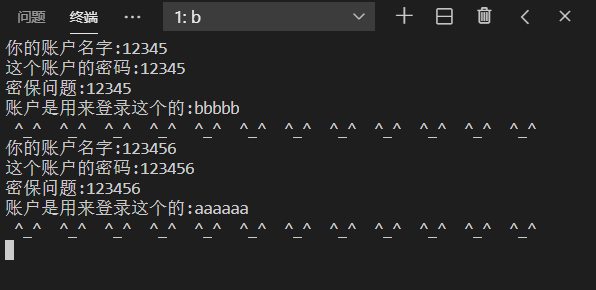


选1添加新的账号123456

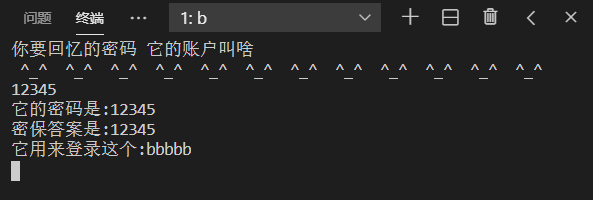


再次添加新账号12345

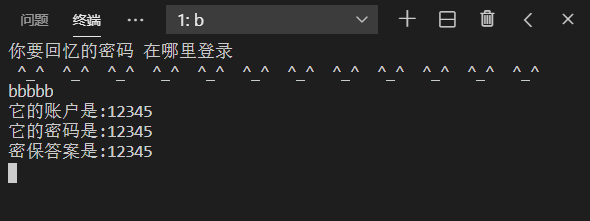
查看密码本，账号已添加并已实现自动排序



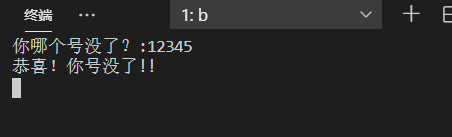
选4，按照账户名查找



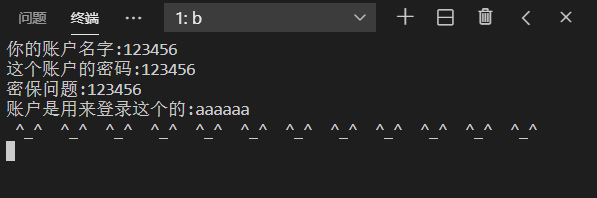
选5，按照登录环境查找



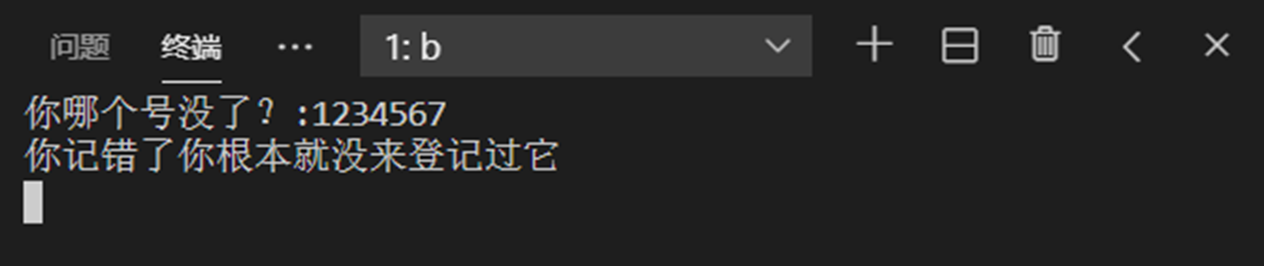
选3删除一个账号



再次查看密码本，账号成功删除



选3删除一个未曾登记过的账号



# 6 总结

崔冬辰：作为组长，在本次BPLP课程中，我学习到了如何统筹规划一个团队的工作，处理团队内部的意见分歧以及分工合作。在团队合作中任何一个人都非常重要，缺少任何一个人这个项目都不可能顺利完成。并且对通过git以及github进行多人协作开发的过程有了更深刻的体会。痛点难点：1.团队如何分工，团队合作的进度如何把握2.在使用git进行团队合作时出现的分支冲突等各种问题。我的贡献：统筹规划团队合作，程序框架，制作项目文档，制作项目汇报幻灯片。

李尚哲：学到了：1.Linux环境下对屏幕指针的操作（ncurses库）2.窗口化应用程序3.单结构链表。痛点：1.窗口化应用后调试要多用getch。 2.指针真的很难用，还是好好套用双结构链表和typedef吧。3.同一个项目最好有不止一个远程仓库，至少也要两个分支。我的贡献：实现除文件读取和保存功能之外的代码，并调试。如何合作？多干活，多交流

唐昊哲：在本次BPLP课程中，我学习到了代码可读性和可维护性的重要性。清爽的代码能让合作者和自己更好理解代码逻辑，完成进一步的编码。我觉得最大的痛点是改bug。我在我们小组负责写算法流程图，一部分代码和改bug。我认为和组员合作的关键一点就是时刻保持好沟通，不懂就问。