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：学生成绩分析与管理系统

授课教师：吴佳

序号	学号	姓名
1（组长）	2019091605012	钟镇阳
2	201909160509	徐洋
3	2019091605005	陈思阳
4	2019091605011	苏若洋

目录

1 项目简介.....2
1.1 考核方式.....2
1.2 项目题目及内容简介.....3
1.3 项目组成员与分工.....3
2 需求分析.....4
2.1 选题的依据.....4
2.2 功能需求.....4
3 系统设计.....6
3.1 总体设计（设计框图）.....6
3.2 模块设计.....6
4 系统实现（包含模块流程图）.....8
4.1 所有相关函数以及全局变量声明.....8
4.2 数据载入模块.....10
4.3 窗台美化及欢迎页模块.....11
4.4 登陆系统模块.....13
4.5 教师菜单界面.....17
4.6 学生菜单界面.....18
4.7 链表功能模块.....20
4.8 数据分析功能模块.....25
4.9 安全加密模块.....54
4.10 main 函数总体实现.....56
5 关键代码分析.....59
安全模块.....59
对链表的优化.....61
数据分析功能.....65
6 功能测试.....65
7 总结.....69
学到了什么？.....69
痛点和难点：.....70
如何与他人合作？.....70

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：学生成绩分析与管理系统

内容介绍：

对不同的登录角色分配不同的操作界面与权限，根据友好的交互，通过简单的操作，对学生成绩进行管理和分析。

1.3 项目组成员与分工

成员：钟镇阳 徐洋 陈思阳 苏若洋

分工：

钟镇阳：讨论项目、负责链表、数据分析模块代码，调试、写报告

徐洋：讨论项目、数据载入、数据分析模块代码、交互美化模块代码、调试、写报告、PPT

苏若洋：讨论项目、负责安全加密算法模块代码，调试，PPT、写报告

陈思阳：讨论项目、负责程序构架、调试代码、模块衔接、提供思路、写报告

2 需求分析

2.1 选题的依据

1、借助信息技术手段帮助管理和分析学生成绩。

2、现有的学生成绩管理系统功能过于单一，不能满足现实需求。

3、面向学生和老师，成绩管理系统应该具有不同的功能和权限，提供更加实际，更加多样化的服务。

4、成绩是一个需要相对较高安全保密的数据，而现有的成绩管理方式大多忽略了安全性这一特点，本程序为此提供了解决方案。

2.2 功能需求

学生端

1、 考试成绩查询。

2、 GPA 的查询。

3、 成绩数据的分析，以提供引导性的改进方案。

1） 班级排名查询。

2） 多次考试平均分查询。

3） 多次考试波动情况查询。

4） 一键生成考试分析报告。

4、 登录密码的更改

教师端

1、学生成绩查询。
2、添加新学生信息。
3、修改学生信息和成绩。
4、删除学生数据。
5、查看成绩分析。
1）班级平均分。
2）班级最高最低分。

3）班级成绩波动情况。

6、登录密码的修改。

数据安全

1、账户登录系统。

1）账户类型区分。

2）账户密码验证。

3）多次登录失败保护。

2、关联文件加密存储。

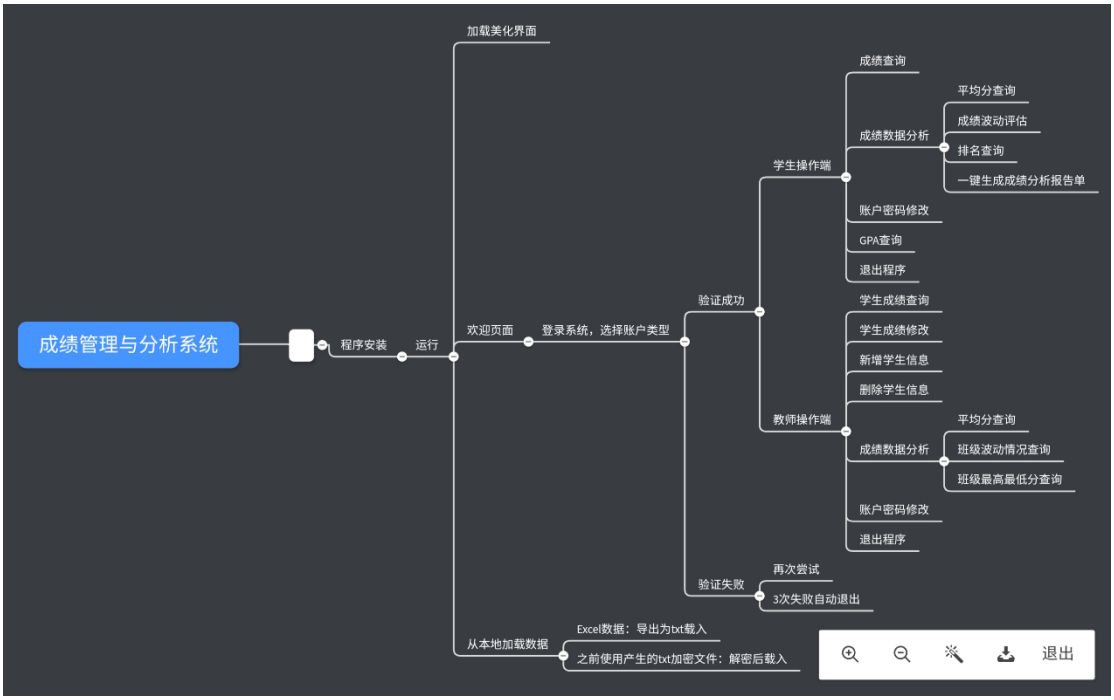
UI 设计

1、界面简洁美观。

2、交互操作友好。

3 系统设计

3.1 总体设计（设计框图）



3.2 模块设计

1、数据载入

实现同一电脑多次使用，和不同电脑间的转移。

1）第一次使用：

通过 Excel 导出为 txt 文件，读取导出后的文本文件实现数据载入。

2）多次使用：

对使用后生成的加密文件进行解密操作载入数据。

2、界面美化

1）使操作界面简洁自然，调用 window.h 中的窗台控制函数对窗台进行自定义美化。

2）对每一个用户界面进行美化适配。

3、登录系统

1）用于选择账户类型，分配不同的操作界面和权限。
2）验证账号密码，识别登录用户。

4、教师端
1）学生成绩的查看。
2）学生成绩的修改。
3）新增学生信息。
4）删除学生信息。
5）学生信息分析。
6）账户密码修改。
7）退出保存、加密操作。

5、学生端
1）个人成绩查看。
2）个人成绩分析。
3）个人 GPA 查看。
4）账户密码修改。
5）退出程序。

6、成绩分析
1）班级、个人平均成绩统计。
2）班级最低分最高分统计。
3）班级、个人成绩波动评估。
4）个人成绩分析报告。
5）个人班级位次查询。

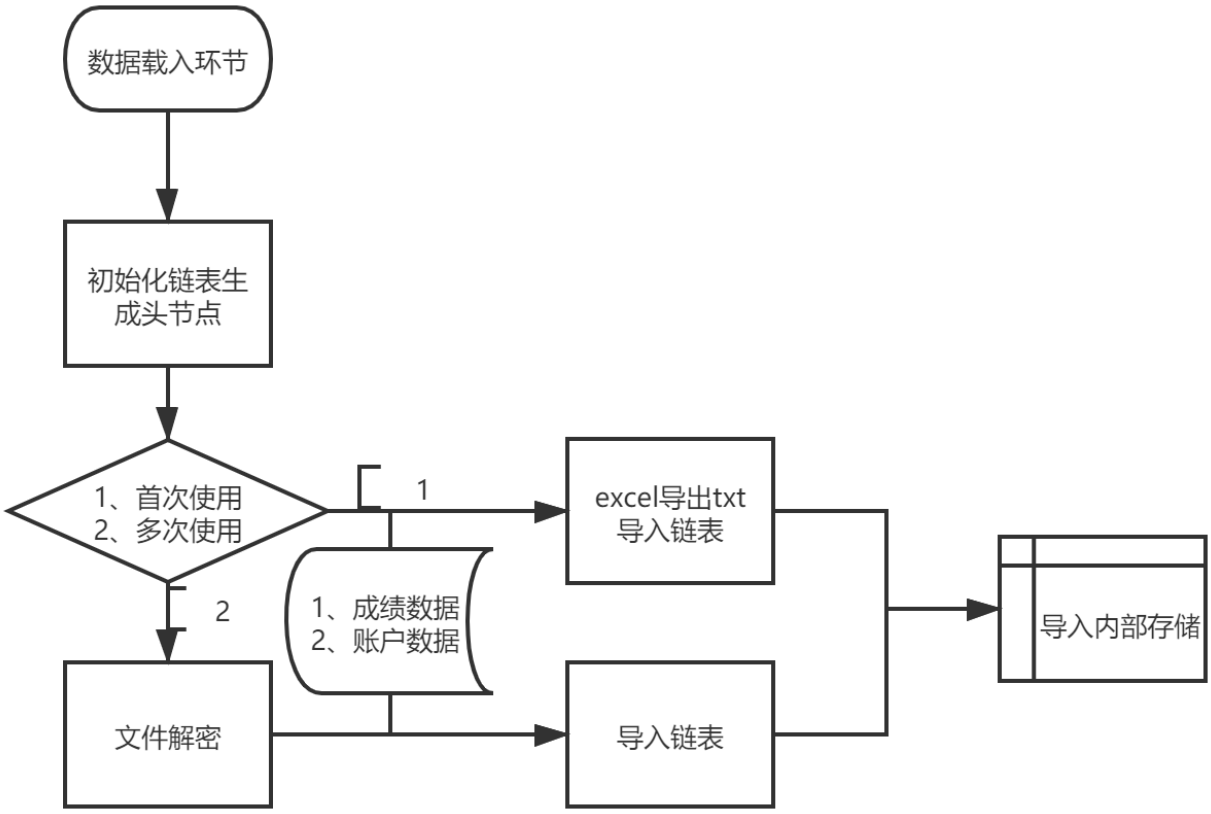
4 系统实现（包含模块流程图）

4.1 所有相关函数以及全局变量声明

```
/*全局数据*/
#define MAX 30
#define length 1000 //密码个数
int error;//报错检测

typedef struct {
    int math;
    int english;
    int programming;
}Score;

typedef struct {
    char name[30];
    char id[30];
    Score score[30];
}Student;

typedef struct node {
    Student data;
    struct node* next;
    struct node* prior;
}Node;

Node* head,*Tail;//头节点
int type, member, account;//账户类型(1-教师，0-学生),人数，账户
char user[MAX] = { 0 }, pass[MAX] = { 0 };//登录用户
char* USER[2*MAX] = { 0 }, * PASS[2*MAX] = { 0 };//账户信息
double avr_c, avr_m, avr_e, avr_a, s_c, s_m, s_e, s_a, s;//平均分,方差
int ranking_c, ranking_m, ranking_e;//排名


/*函数声明*/
void headview(void);//顶部图标
SMALL_RECT SizeOfWindow(HANDLE hConsoleOutput);//窗口
void modeset(int w, int h);//窗口
void surface(void);//窗口初始化
```

```
bool login(void);//登录
int awelcome(void);//管理员主菜单，返回选项
int swelcome(void);//学生主菜单，返回选项
void init_node(void);//初始化链表
void load(void);//载入成绩数据
void jump(void);//跳转界面
void save(void);//保存写入
void add_student(void);//添加成员
void write(void);//控制台写入信息
Node* search(char id[MAX]);//链表搜索
void print_one(Node* list);//打印输出信息
void  delet(Node** list, char id[MAX]);//删除信息
void put_in_order(Node* phead, int code_data, int code_num);//排序函数
void f_rand(int a[], int* t); //随机数
void encrypt(char* location_1, char* code_location); //加密。加密前的文本位置，密码位置
void decode(char* location_2, char* code_location); //解密。After_Encrypt 的位置，密码位置
int statistics(void);//人数统计
int rank(char id[MAX]);//排序
int extreme_value(Node* list, int choice, int code_num);//班级最值统计
double average_all(Node* list, int choice, int code_num);//班级均分统计
double average_one(Node* list, int choice);//个人均分统计
double variance_all(Node* phead);//班级波动指数
double variance_one(Node* p,int choice);//个人波动指数
void report(int code_num);//生成成绩报告
void GPA(Node* phead);//GPA
void insert_sort(Node* phead);//插入排序
int analyze(void);//数据分析界面
void f_analyze(int change);//数据分析实现
```

4.2 数据载入模块



1、双向链表初始化

```
1.  //初始化链表
2.  void init_node(void)
3.  {
4.      head = (Node*)malloc(sizeof(Node));
5.      head->next = NULL;
6.      head->prior = NULL;
7.      load();
8.  }
```

2、链表数据载入

```
1.  //载入成绩数据
2.  void load(void)
3.  {
4.      int i = 0;
5.      Node* list;
6.      Node* flag;
```

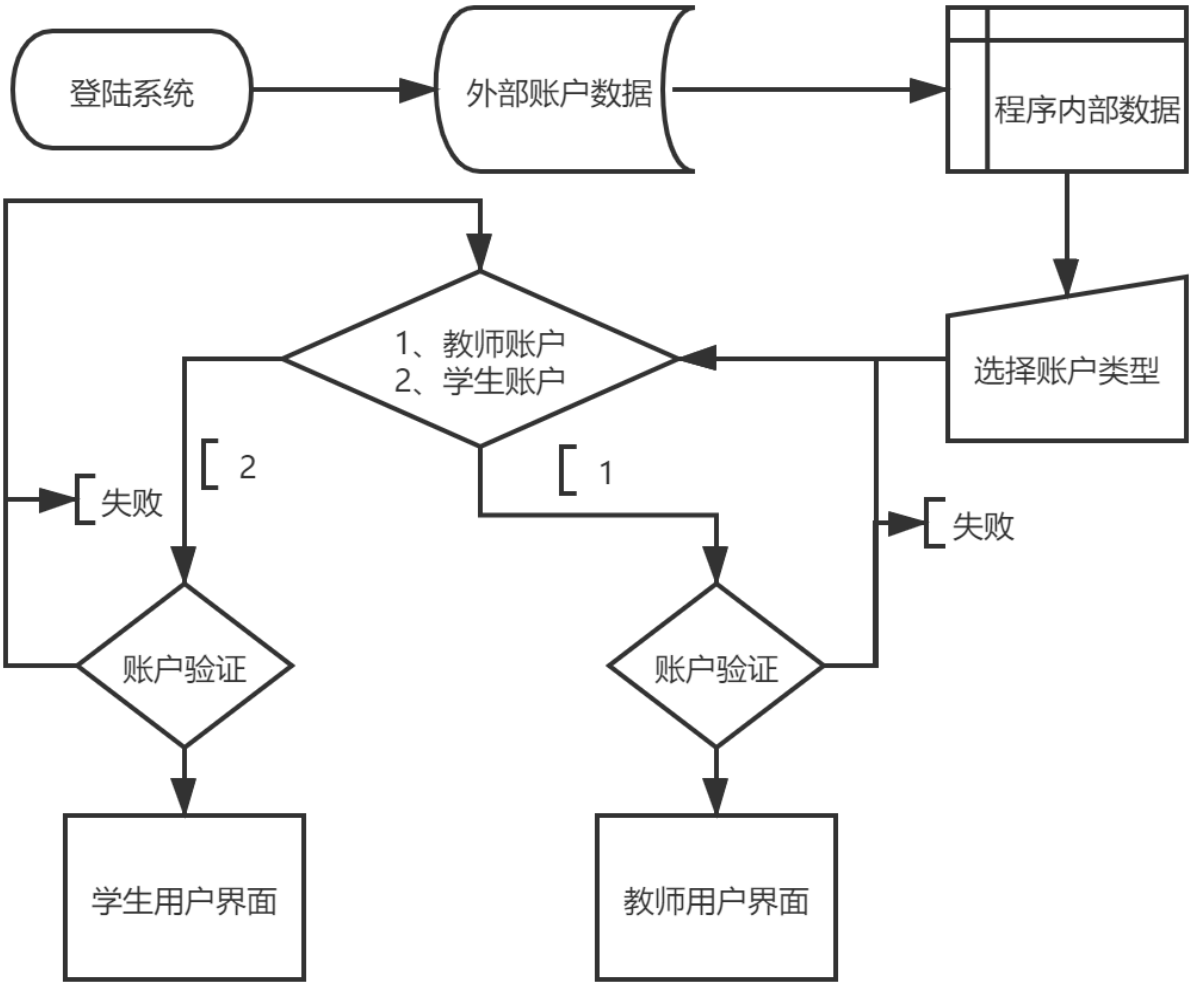
```
    FILE* txt;
    decode("std.txt", "a5.txt");
    txt = fopen("std.txt", "r+");
    for (list = head; \
        fscanf(txt, "%s", list->data.name) != EOF;
        list = list->next, list->prior = flag)//判断是否读到结尾的同时读入 name
    {
        list->data.score->english = list->data.score->math = list->data.score->programming
= 0;
        fscanf(txt, "%s", list->data.id);
        for (i = 0; i < 30; i++)
        {
            fscanf(txt, "%d %d %d", &(list->data.score[i].programming), \
                & (list->data.score[i].math), &(list->data.score[i].english));
        }
        (list->next) = (Node*)malloc(sizeof(Node));
        flag = list;
    }
    fclose(txt);
    encrypt("std.txt", "a5.txt");
    list->next = NULL;
}
```

4.3 窗台美化及欢迎页模块

```
//窗口初始化
void surface(void)
{
    SetConsoleTitle("UESTC_1605 班成绩管理系统");
    system("color 70");
    modeset(1200, 675);
    headview();
    printf("\n\n\t\t\t\t\t 这是一个看似单调，却拥有无数创新的成绩系统！\n");
    printf("\n\n\n\n\n\t\t\t\t\t  ||——Made by -Csy- -Sry- -Xy- -Zzy-||\n");
    int i;
    printf("\n\n\n\n\n\t\t\t\t\t\t\t");
    char heihei[] = { "按 Enter 键进入学生成绩系统..." };
    for (i = 0; i < strlen(heihei); i++)
    {
        printf("%c", heihei[i]);
        Sleep(50);
    }
```

```
    getchar();
    system("CLS");

}
void headview(void)
{
    printf("\n\n\n");
    printf("\t\t\t\t\t     ╭ ╹╱% ╮                    ╭ ``` ╮  \n");
    printf("\t\t\t\t\t     (@^o^@) 学生成绩管理系统 (╭:╮)\n");
    printf("\t\t\t\t\t     (~):(~)                  (~)v(~) \n");
    printf("\n");
}

//窗口调整
SMALL_RECT SizeOfWindow(HANDLE hConsoleOutput)
{
    CONSOLE_SCREEN_BUFFER_INFO info;
    GetConsoleScreenBufferInfo(hConsoleOutput, &info);
    return info.srWindow;
}
void modeset(int w, int h)
{
    //  此函数设置窗口大小为 w*h
    HANDLE hOut = GetStdHandle(STD_OUTPUT_HANDLE);
    CONSOLE_SCREEN_BUFFER_INFO info;
    SMALL_RECT rect = SizeOfWindow(hOut);
    COORD size = { rect.Right + 1,rect.Bottom + 1 };    //定义缓冲区大小
    //COORD size = { w, h };
    SetConsoleScreenBufferSize(hOut, size);
    GetConsoleScreenBufferInfo(hOut, &info);//隐藏滑动条
    SMALL_RECT rc = { 1,1, w, h };
    SetConsoleWindowInfo(hOut, 1, &rc);
    SetWindowLongPtrA(
        GetConsoleWindow(),
        GWL_STYLE,
        GetWindowLongPtrA(GetConsoleWindow(), GWL_STYLE)
        & ~WS_SIZEBOX & ~WS_MAXIMIZEBOX & ~WS_MINIMIZEBOX
    );
}
```

4.4 登陆系统模块




```
1.  //登录界面
2.  bool login(void)
3.  {
4.      FILE* apas,*spas,*ause,*suse;
5.      //apas = "apass.txt";
6.      //spas = "spass.txt";
7.      //ause = "auser.txt";
8.      //suse = "suser.txt";
9.
10.     int times=0,j = 0,a=1,choice=10, i = 0;//a 用于依次验证密码
11.     again:system("cls");
12.     headview();
13.     printf("\t\t\t\t      ┌ ==============================○●○●======┐\n");
```

```
14.     printf("\t\t\t\t     |       电子科技大学 1605 班学生成绩管理系统       |\n");
15.     printf("\t\t\t\t     └ ======o●o●==============================┘\n");
16.     printf("\n\n\n\n\n");
17.     printf("\t\t\t\t\t 你的账户类型？    1—>学生 | 2—>教师\n\n");
18.     printf("\t\t\t\t\t\t\t\t 你的选择：");
19.     scanf("%d", &choice);
20.     system("cls");
21.     begain:headview();
22.     printf("\t\t\t\t     ┌ ==============================o●o●======┐\n");
23.     printf("\t\t\t\t     |       电子科技大学 1605 班学生成绩管理系统       |\n");
24.     printf("\t\t\t\t     └ ======o●o●==============================┘\n");
25.     printf("\n\n\n\n\n");
26.     printf("\t\t\t\t\t   用户名:   ");
27.     scanf("%s", user);
28.     printf("\n\t\t\t\t\t\t   密码:  ");
29.     scanf("%s",pass);
30.     //printf("%s\n%s", user, pass);输入验证
31.     switch (choice)
32.     {
33.     case 1:
34.         decode("spass.txt", "a1.txt");
35.         spas=fopen("spass.txt", "r+");
36.         if (spas == NULL)
37.         {
38.             printf("Error: read file failure.\n");
39.             exit(-1);
40.         }
41.             for ( i = 0; i < 60; i++)
42.             {
43.                 PASS[i] = malloc(sizeof(char*));
44.                 if(fscanf(spas, "%s", PASS[i])==EOF)break;
45.             }
46.             PASS[i] = NULL;
47.         fclose(spas);
48.         encrypt("spass.txt", "a1.txt");
49.         decode("suser.txt", "a2.txt");
50.         suse=fopen("suser.txt", "r+");
51.         if (suse == NULL)
52.         {
53.             printf("Error: read file failure.\n");
54.             exit(-1);
55.         }
```

```
56.             for ( i = 0; i < 60; i++)
57.             {
58.                 USER[i] = malloc(sizeof(char*));
59.                 if (fscanf(suse, "%s", USER[i]) == EOF)break;
60.             }
61.             USER[i] = NULL;
62.         fclose(suse);
63.         encrypt("suser.txt", "a2.txt");
64.
65.         //printf("%s\n%s", PASS[1], USER[1]);//文件写入验证
66.         getchar();
67.         for ( i = 0; a; i++)
68.         {
69.             a *= (!(strcmp(user, USER[i]) == 0 && strcmp(pass, PASS[i]) == 0));//有对则 0，
    全错则 1
70.             if (a == 0)account = i;
71.         }
72.             while (a)
73.             {
74.                 times++;    //密码输入错误 times++
75.                 if (times > 3)
76.                 {
77.                     printf("\n\n\n\t\t\t\t 账号或密码输入错误累计达到%d 次，系统将于 3 秒后关闭
    ", times);
78.                     Sleep(1000);
79.                     system("cls");
80.                     char shutdown[] = { "系统将于%d 秒后关闭..." };
81.                     for (int i = 0; i < 3; i++)
82.                     {
83.                         printf(shutdown, 3 - i);
84.                         Sleep(1000);
85.                         system("cls");
86.                     }
87.                     exit(0);
88.                 }
89.                 printf("\n\n\n\t\t\t\t 账号或密码输入错误,你还有%d 次登录机会，按任意键重新登
    录...", 4 - times);
90.                 getchar();
91.                 system("cls");
92.                 goto begain;
93.             }
94.             /*for (int i = 0; i < 30; i++) {
```

```
95.
96.                 free(USER[i]);
97.                 free(PASS[i]);
98.             }*/
99.             //内存释放报错
100.             return 0;
101.     case 2:
102.         decode("apass.txt", "a3.txt");
103.         apas=fopen( "apass.txt", "r+");
104.         if (apas == NULL)
105.         {
106.             printf("Error: read file failure.\n");
107.             exit(-1);
108.         }
109.         for ( i = 0; i < 60; i++) {
110.             PASS[i] = malloc(sizeof(char*));
111.             if (fscanf(apas, "%s", PASS[i]) == EOF)break;
112.         }
113.         PASS[i] = NULL;
114.         fclose(apas);
115.         encrypt("apass.txt", "a3.txt");
116.         decode("auser.txt", "a4.txt");
117.         ause=fopen( "auser.txt", "r+");
118.         if (ause == NULL)
119.         {
120.             printf("Error: read file failure.\n");
121.             exit(-1);
122.         }
123.         for ( i = 0; i < 60; i++) {
124.             USER[i] = malloc(sizeof(char*));
125.             if (fscanf(ause, "%s", USER[i]) == EOF)break;
126.         }
127.         USER[i] = NULL;
128.         fclose(ause);
129.         encrypt("auser.txt", "a4.txt");
130.         for (int i = 0; a; i++)
131.         {
132.             a *= (!(strcmp(user, USER[i]) == 0 && strcmp(pass, PASS[i]) == 0));//有对则 0，
    全错则 1
133.             if (a == 0)account = i;
134.         }
135.             while (a)
```

```
136.             {
137.                 times++;    //密码输入错误 times++
138.                 if (times > 3)
139.                 {
140.                     printf("\n\n\n\t\t\t\t 账号或密码输入错误累计达到%d 次，系统将于 3 秒后关闭
    ", times);
141.                     Sleep(1000);
142.                     system("cls");
143.                     char shutdown[] = { "系统将于%d 秒后关闭..." };
144.                     for (int i = 0; i < 3; i++)
145.                     {
146.                         printf(shutdown, 3 - i);
147.                         Sleep(1000);
148.                         system("cls");
149.                     }
150.                     exit(0);
151.                 }
152.                 printf("\n\n\n\t\t\t\t 账号或密码输入错误,你还有%d 次登录机会，按任意键重新登
    录...", 4 - times);
153.                 getchar();
154.                 system("cls");
155.                 goto begain;
156.             }
157.             /*for (int i = 0; i < 30; i++) {
158.
159.                 free(USER[i]);
160.                 free(PASS[i]);
161.             }*/
162.             //内存释放报错
163.             return 1;
164.     default:
165.         printf("\n\n\t\t\t\t\t    请重新输入账户类型！\n");
166.         goto again;
167.     }
168.     getchar();
169. }
```

4.5 教师菜单界面

```
1.  //管理员菜单界面
2.  int awelcome(void)
3.  {
```

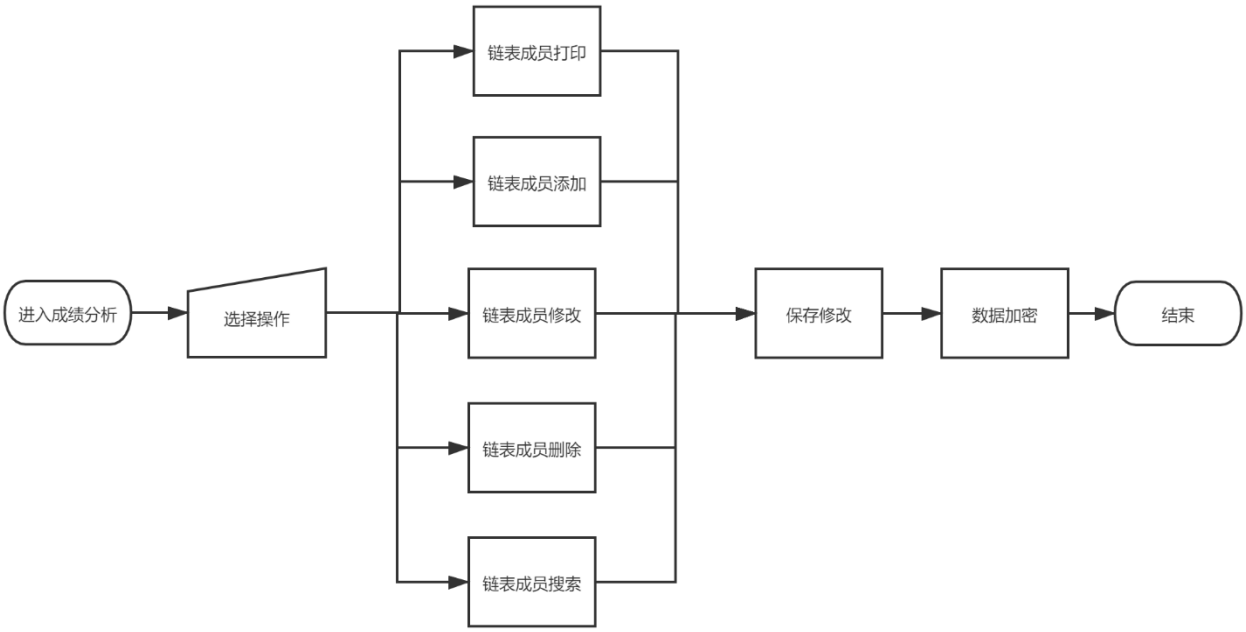
```
    int choice=7;
    system("cls");
    headview();
    printf("\t\t\t\t     ┌ ==============================○●○●======┐\n");
    printf("\t\t\t\t     |       电子科技大学 1605 班学生成绩管理系统      |\n");
    printf("\t\t\t\t     └ ======○●○●==============================┘\n");
    printf("\n");
    printf("\t\t\t\t\t\t\t    MENU\n\n");
    //printf("\t\t\t\t\t\t 0—>查看全班成绩\n\n");
    printf("\t\t\t\t\t\t     1—>查看学生成绩\n\n");
    printf("\t\t\t\t\t\t     2—>新增学生成绩\n\n");
    printf("\t\t\t\t\t\t     3—>修改学生成绩\n\n");
    printf("\t\t\t\t\t\t     4—>删除学生信息\n\n");
    printf("\t\t\t\t\t\t     5—>成绩数据分析\n\n");
    printf("\t\t\t\t\t\t     6—>账户密码修改\n\n");
    printf("\t\t\t\t\t\t     7—>退出 并 保存\n\n");
    printf("\t\t\t\t\t\t        输入指令编号：\n");
    printf("\t\t\t\t\t\t              ");
    scanf("%d", &choice);
    return choice;
}
```

4.6 学生菜单界面

```
//学生端菜单界面
int swelcome(void)
{
    int change,choice=8;//修改传出指令
    system("cls");
    headview();
    printf("\t\t\t\t     ┌ ==============================○●○●======┐\n");
    printf("\t\t\t\t     |       电子科技大学 1605 班学生成绩管理系统      |\n");
    printf("\t\t\t\t     └ ======○●○●==============================┘\n");
    printf("\n\n");
    printf("\t\t\t\t\t\t\t   MENU\n\n");
    printf("\t\t\t\t\t\t    1—>查看你的成绩\n\n");
    printf("\t\t\t\t\t\t    2—>成绩数据分析\n\n");
    printf("\t\t\t\t\t\t    3—>账户密码修改\n\n");
    printf("\t\t\t\t\t\t    4—>  GPA 查询\n\n");
    printf("\t\t\t\t\t\t    5—>     退出\n\n");
    printf("\t\t\t\t\t\t       输入指令编号：\n");
    printf("\t\t\t\t\t\t             ");
```

```
    scanf("%d", &change);
    switch (change)
    {
    case 1:
        system("cls");
        headview();
        print_one(search(user));
        jump();
        break;
    case 2:
        choice = 5;
        break;
    case 3:
        choice = 6;
        break;
    case 4:
        choice = 8;
        break;
    case 5:
        choice = 7;
        break;
    default:
        break;
    }
    return choice;
}
```

4.7 链表功能模块



4.7.1 成绩显示模块

```
1.  //打印成绩
2.  void print_one(Node* list)
3.  {
4.      //list = list->next;测试节点转换
5.      int n = 1, i = 0;
6.      printf("\n\n");
7.      printf("\n\n");
8.      printf("\t\t\t\t 姓名：%s", list->data.name);
9.      printf("\t\t\t\t 学号：%s", list->data.id);
10.     printf("\n\n");
11.     printf("\t\t\t\t\t\t    查看第几次考试：");
12.     scanf("%d", &n);
13.     printf("\n\t\t\t\t\t\t\t  C 语言\t  数学\t  英语\t\n");
14.     if (n)
15.     {
16.         if (abs(list->data.score[n - 1].programming) >= 10000 && abs(list->data.score[n - 1
    ].english) >= 10000 && abs(list->data.score[n - 1].math) >= 10000)
17.             list->data.score[n - 1].programming = list->data.score[n - 1].math = list->data
    .score[n - 1].english = 0;
18.         printf("\t\t\t\t\t\t  %d\t  %d\t  %d\t\n\n\n\n\n\n", list->data.score[n - 1].programm
    ing, \
```

```
            list->data.score[n - 1].math, list->data.score[n - 1].english);
    }
    else
    {
        for (i; i < 30; i++)
        {
            if (abs(list->data.score[n - 1].programming) >= 10000 && abs(list->data.score[n - 1].english) >= 10000 && abs(list->data.score[n - 1].math) >= 10000)
                list->data.score[n - 1].programming = list->data.score[n - 1].math = list->data.score[n - 1].english = 0;
            printf("\t\t\t\t\t\t  %d\t  %d\t  %d\t\n\n\n\n\n", list->data.score[i].programming, \
                list->data.score[i].math, list->data.score[i].english);
            printf("\n");
        }
    }
    getchar();
}
```

4.7.2 成员添加模块

```
//添加成员
void add_student(void)
{
    Node* list;
    int times;
    list = (Node*)malloc(sizeof(Node));
    list->data.score->english = list->data.score->math = list->data.score->programming = 0;

    //for (list = head; list->next; list = list->next);
    list->next = head;
    head = list;
    printf("\n\n");
        printf("\t\t\t\t\t\t 请输入姓名: ");
        scanf("%s", list->data.name);
        printf("\n\t\t\t\t\t\t 请输入学号: ");
        scanf("%s", list->data.id);
        printf("\n\t\t\t\t\t\t 第几次考试: ");
        scanf("%d", ×);
        printf("\n\t\t\t\t\t\t 请输入分数(C 语言 数学 英语): ");
        scanf("%d %d %d", &(list->data.score[times - 1].programming), \
            & (list->data.score[times - 1].math), &(list->data.score[times - 1].english));
```

```
}
```

4.7.3 成绩及信息修改模块

```
//控制台写入信息
void write(void)
{
    Node* list;
    char id[MAX];
    printf("\n\n\n\t\t\t\t\t\t\t\t请输入学号：");
    scanf("%s", id);
    list = search(id);
    printf("\n\t\t\t\t\t\t\t姓名：%s", list->data.name);
    int n1, n2, n3, times;
    printf("\n\n");
    printf("\n\t\t\t\t\t\t\t    修改单项信息输入 0\n");
    printf("\n\t\t\t\t\t\t\t    填入全部信息输入 1\n");
    printf("\n\t\t\t\t\t\t\t        请输入：");
    scanf("%d", &n1);
    while (1)
    {
        switch (n1)
        {
        case 0:
            while (1)
            {
                system("cls");
                headview();
                printf("\n\n");
                printf("\t\t\t\t    ┌ ==============================○●○======┐\n");
                printf("\t\t\t\t    |   电子科技大学 1605 班学生成绩管理系统-教师   |\n");
                printf("\t\t\t\t    └ ======○●○==============================┘\n");
                printf("\n\n");
                printf("\t\t\t\t\t\t     0-修改姓名\n\n");
                printf("\t\t\t\t\t\t     1-修改学号\n\n");
                printf("\t\t\t\t\t\t     2-修改 C 语言成绩\n\n");
                printf("\t\t\t\t\t\t     3-修改数学成绩\n\n");
                printf("\t\t\t\t\t\t     4-修改英语成绩\n\n");
                printf("\t\t\t\t\t\t        请输入：");
                scanf("%d", &n2);
                if (n2 != 0 && n2 != 1)
                {
                    system("cls");
```

```
40.                     headview();
41.                     printf("\n\n");
42.                     printf("\t\t\t\t      ┌ ==============================○●○●======┐\n");
43.                     printf("\t\t\t\t      |    电子科技大学 1605 班学生成绩管理系统-教
   师    |\n");
44.                     printf("\t\t\t\t      └ ======○●○●==============================┘\n");
45.                     printf("\n\n");
46.                     printf("\n\t\t\t\t\t\t\t       修改第几次成绩：");
47.                     scanf("%d", &n3);
48.                 }
49.                 switch (n2)
50.                 {
51.                 case 0:
52.                     printf("\n\t\t\t\t\t\t\t 请输入姓名：");
53.                     scanf("%s", list->data.name);
54.                     break;
55.                 case 1:
56.                     printf("\n\t\t\t\t\t\t\t 请输入学号：");
57.                     scanf("%s", list->data.id);
58.                     break;
59.                 case 2:
60.                     printf("\n\t\t\t\t\t\t\t    请输入 C 语言成绩：");
61.                     scanf("%d", &(list->data.score[n3 - 1].programming));
62.                     break;
63.                 case 3:
64.                     printf("\n\t\t\t\t\t\t\t    请输入数学成绩：");
65.                     scanf("%d", &(list->data.score[n3 - 1].math));
66.                     break;
67.                 case 4:
68.                     printf("\n\t\t\t\t\t\t    请输入英语成绩：");
69.                     scanf("%d", &(list->data.score[n3 - 1].english));
70.                     break;
71.                 default:
72.                     printf("\n\t\t\t\t\t\t\t 无效指令");
73.                     getchar();
74.                     break;
75.                 }
76.                 if (n2 == 0 || n2 == 1 || n2 == 2 || n2 == 3 || n2 == 4)
77.                     break;
78.             }
79.             break;
80.         case 1:
```

```
81.             printf("\n\t\t\t\t\t\t\t 请输入姓名： ");
82.             scanf("%s", list->data.name);
83.             printf("\n\t\t\t\t\t\t\t 请输入学号： ");
84.             scanf("%s", list->data.id);
85.             printf("\n\t\t\t\t\t\t\t 第几次考试： ");
86.             scanf("%d", x);
87.             printf("\n\t\t\t\t\t\t\t 请输入分数(C 语言 数学 英语)： ");
88.             scanf("%d %d %d", &(list->data.score[times - 1].programming), \
89.                 & (list->data.score[times - 1].math), &(list->data.score[times - 1].english
    ));
90.             break;
91.         default:
92.             printf("\n\t\t\t\t\t\t\t 请重新输入");
93.             system("pause");
94.             break;
95.         }
96.         if (n1 == 0 || n1 == 1)
97.             break;
98.     }
99. }
```

4.7.4 成员搜索模块

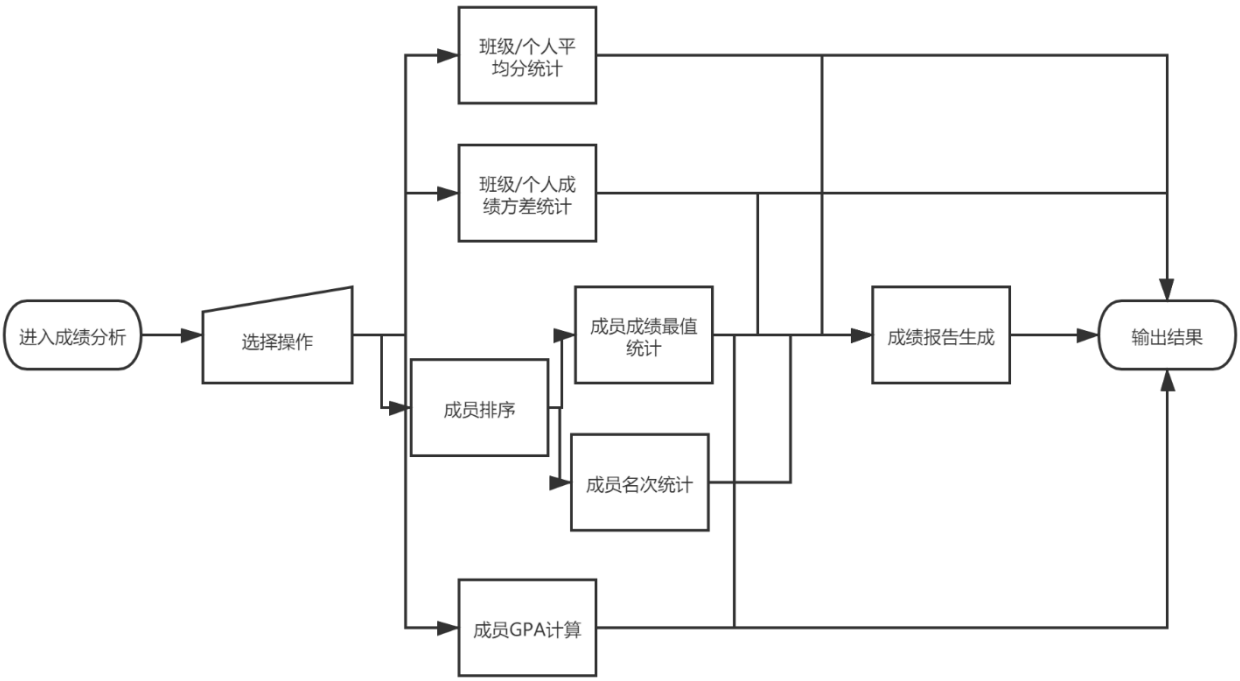
```
1.  //搜索链表
2.  Node* search(char id[MAX])//已知 BUG：学号不存在，到达链表尾崩溃
3.  {
4.      Node* list;
5.      list = head;
6.      for (; list && strcmp(list->data.id, id); list = list->next);//ID search
7.      return list;
8.  }
```

4.7.5 成员删除模块

```
1.  //删除链表节点
2.  void  delet(Node** list, char id[MAX])              //void 函数可以减少赋值一步，避免出错；
3.  {
4.      Node* p = (*list);
5.      Node* q=NULL; //跟随指针
6.      if (!strcmp(p->data.id, id)) {        //第一个数据就是所找数据时的特殊情况
7.          (*list) = (*list)->next;    //删除第一个结点
8.      }
9.      else {
```

```
        for (; p && strcmp(p->data.id, id); q = p, p = p->next);
        if (!p)
            printf("unknown id\n"); else {
            q->next = p->next;    //将跟随指针跳过所删除的结点，指向下一个
        }
    }

}
```

4.8 数据分析功能模块



4.8.1 链表排序算法模块

1）冒泡排序：写了 2 种不同的排列方式，4 种不同排列对象。

```
//排序函数 默认降序排列
void put_in_order(Node* phead,int code_data, int code_num)
{          //排序，不输出
    int code_order;
    Node* p = phead;
    Node* tail = NULL;
    Student temp;
    code_order = 0;
    code_num--;
    while (1)
```

```
    {
        switch (code_order)
        {
        case 0:
            while (1)
            {
                switch (code_data)
                {
                case 0:
                    while (p != tail)
                    {
                        while (p->next != tail)
                        {
                            if (p->data.score[code_num].math < p->next->data.score[code_num].math)
                            {
                                temp = p->data;
                                p->data = p->next->data;
                                p->next->data = temp;
                            }
                            p = p->next;
                        }
                        tail = p;
                        p = phead;
                    }break;
                case 1:
                    while (p != tail)
                    {
                        while (p->next != tail)
                        {
                            if (p->data.score[code_num].english < p->next->data.score[code_num].english)
                            {
                                temp = p->data;
                                p->data = p->next->data;
                                p->next->data = temp;
                            }
                            p = p->next;
                        }
                        tail = p;
                        p = phead;
                    }break;
```

```
51.                 case 2:
52.                     while (p != tail)
53.                     {
54.                         while (p->next != tail)
55.                         {
56.                             if (p->data.score[code_num].programming < p->next->data.score[c
    ode_num].programming)
57.                             {
58.                                 temp = p->data;
59.                                 p->data = p->next->data;
60.                                 p->next->data = temp;
61.                             }
62.                             p = p->next;
63.                         }
64.                         tail = p;
65.                         p = phead;
66.                     }break;
67.                 default:printf("\t\t\t\t\t\t 请重新输入");
68.                 }break;
69.                 if (code_data = 0 || code_data == 1 || code_data == 2)
70.                     break;
71.             }
72.             break;
73.         case 1:
74.             while (1)
75.             {
76.                 switch (code_data)
77.                 {
78.                 case 0://while 中的 p->next->next != NULL 是为了避免读取最后一个未初始化的尾节点
    而将其传到链表头节点
79.                     while (p != tail )
80.                     {
81.                         while (p->next != tail && p->next->next != NULL)
82.                         {
83.                             if (p->data.score[code_num].math > p->next->data.score[code_num
    ].math)
84.                             {
85.                                 temp = p->data;
86.                                 p->data = p->next->data;
87.                                 p->next->data = temp;
88.                             }
89.                             p = p->next;
```

```
                        }
                        tail = p;
                        p = phead;
                    }break;
                case 1:
                    while (p != tail)
                    {
                        while (p->next != tail && p->next->next != NULL)
                        {
                            if (p->data.score[code_num].english > p->next->data.score[code_num].english)
                            {
                                temp = p->data;
                                p->data = p->next->data;
                                p->next->data = temp;
                            }
                            p = p->next;
                        }
                        tail = p;
                        p = phead;
                    }break;
                case 2:
                    while (p != tail)
                    {
                        while (p->next != tail && p->next->next != NULL)
                        {
                            if (p->data.score[code_num].programming > p->next->data.score[code_num].programming)
                            {
                                temp = p->data;
                                p->data = p->next->data;
                                p->next->data = temp;
                            }
                            p = p->next;
                        }
                        tail = p;
                        p = phead;
                    }break;
                default:printf("\t\t\t\t\t\t 请重新输入");
                        break;
                }
            if (code_data = 0 || code_data == 1 || code_data == 2)
```

```
                break;
            }
            break;
        default:printf("\t\t\t\t\t\t 请重新输入");
            break;
        }
        if (code_order == 0 || code_order == 1)
            break;
    }
}
```

2)链表插入排序：只写了 1 种排序对象，排序原理如下。

```
//插入排序
void insert_sort(Node* phead) {
    Node* p = phead;//插入排序法
    Student temp;
    Node* in_order, * out_order, * temp_order;
    for (out_order = p->next; out_order; out_order = out_order->next) {
        for (in_order = out_order->prior; in_order; in_order = in_order->prior) { //默认第一个数为有序数
            if (in_order->data.score[0].math > out_order->data.score[0].math)
                break;
        }   //找到分界点
        if (!in_order) {
            in_order = (Node*)malloc(sizeof(Node)); //此类为特殊情况，当无序值比所有有序值都大时，在链表会导致链表指向空指针，因此需要分配内存空间
            in_order->next = p;
            p->prior = in_order;
        }
        if (in_order != out_order->prior) {  //可以减少复杂度，当无序比有序数都小时直接结束
            temp = out_order->data;
            for (temp_order = out_order->prior; temp_order != in_order; temp_order = temp_order->prior)
                temp_order->next->data = temp_order->data;//数据往后移动一位
            in_order->next->data = temp; //正式将无序数插入 有序数组
        }
        p->prior = NULL;//需要让头结点恢复原样
    }
}
```

3)希尔排序（不常用）等排序法。

```
Node *getMiddleNode(Node *pList)  //归并排序 对第一次数学成绩由高到低地排序
{
    if (pList == NULL)
    {
        return NULL;
    }
    Node *pAhead = pList->next;
    Node *pBehind = pList;
    while (pAhead != NULL)
    {
         pAhead = pAhead->next;
         if (pAhead != NULL)
         {
             pAhead = pAhead->next;
             pBehind = pBehind->next;
         }

    }

    return pBehind;
}

Node *MergeList(Node *p1, Node *p2) //合并有序链表，合并之后升序排列
{
    if (NULL == p1)
    {
        return p2;
    }
    if (NULL == p2)
    {
        return p1;
    }

    Node *pLinkA = p1;
    Node *pLinkB = p2;
    Node *pTemp = NULL;
    if (pLinkA->data.score[0].math <= pLinkB->data->data.score[0].math)
    {
        pTemp = pLinkA;
        pLinkA = pLinkA->next;
    }
    else
```

```
67.     {
68.         pTemp = pLinkB;
69.         pLinkB = pLinkB->next;
70.     }
71.
72.     Node *pHead = pTemp; //初始化头结点，即头结点指向不为空的结点
73.     while (pLinkA && pLinkB)
74.     {
75.         if (pLinkA->data.score[0].math <= pLinkB->data.score[0].math)
76.         {
77.             pTemp->next = pLinkA;
78.             pTemp = pLinkA;
79.             pLinkA = pLinkA->next;
80.         }
81.         else
82.         {
83.             pTemp->next = pLinkB;
84.             pTemp = pLinkB;
85.             pLinkB = pLinkB->next;
86.         }
87.
88.     }
89.
90.     pTemp->next = pLinkA ? pLinkA:pLinkB; //插入多余的链表部分
91.
92.     return pHead;
93. }
94.
95. Node *MergeSort(Node *pList)
96. {
97.     if (pList == NULL || pList->next == NULL)
98.     {
99.         return pList;
100.     }
101.
102.     Node *pMiddle = getMiddleNode(pList); //获取中间结点
103.     Node *pBegin = pList; //链表前半部分，包括中间结点
104.     Node *pEnd = pMiddle->next;//链表后半部分
105.     pMiddle->next = NULL; //必须赋值为空
106.     pBegin = MergeSort(pBegin); //排序前半部分数据
107.     pEnd = MergeSort(pEnd); //排序后半部分数据
108.     return MergeList(pBegin, pEnd); //合并有序链表
```

```
109. }
110.
111. }
```

4.8.2 班级人数统计模块

```
1.  //人数统计
2.  int statistics(void)
3.  {
4.      int total=0;
5.      Node* list;
6.      list = head;
7.      for (; list; total++, list = list->next);
8.      return total-1;
9.  }
```

4.8.3 位次输出模块

```
1.  //排名
2.  int rank(char id[MAX])
3.  {
4.      Node* list;
5.      int ranking = 1;
6.      list = head;
7.      for (; list && strcmp(list->data.id, id);ranking++, list = list->next);//ID search
8.      return ranking;
9.  }
```

4.8.4 最值统计模块（包含了排序算法）

```
1.  //最值统计
2.  int extreme_value(Node* list, int choice, int code_num)
3.  {
4.      int extreme = 0 ,mode;
5.      Node* tail = NULL, *p=list;
6.      Student temp;
7.      system("cls");
8.      printf("\n\n");
9.      printf("\t\t\t\t  ┌ ==============================○●○●======┐\n");
10.     printf("\t\t\t\t  |    电子科技大学 1605 班学生成绩管理系统-教师    |\n");
11.     printf("\t\t\t\t  └ ======○●○●==============================┘\n");
12.     printf("\n\n");
```

```
13.     printf("\n\n");
14.     printf("\t\t\t\t\t\t 0——>    最高分\n\n");
15.     printf("\t\t\t\t\t\t 1——>    最低分\n\n");
16.     printf("\t\t\t\t\t\t    输入指令编号：\n");
17.     printf("\t\t\t\t\t\t          ");
18.     scanf("%d", &mode);
19.     code_num--;
20.     if (mode == 0)
21.     {
22.         do {
23.             switch (choice)
24.             {
25.             case 0://MAX
26.                 while (p != tail)
27.                 {
28.                     while (p->next != tail)
29.                     {
30.                         if (p->data.score[code_num].math < p->next->data.score[code_num].ma
    th)
31.                         {
32.                             temp = p->data;
33.                             p->data = p->next->data;
34.                             p->next->data = temp;
35.                         }
36.                         p = p->next;
37.                     }
38.                     tail = p;
39.                     p = list;
40.                 }
41.                 extreme = list->data.score[code_num].math;
42.                 break;
43.             case 1:
44.                 while (p != tail)
45.                 {
46.                     while (p->next != tail)
47.                     {
48.                         if (p->data.score[code_num].english < p->next->data.score[code_num]
    .english)
49.                         {
50.                             temp = p->data;
51.                             p->data = p->next->data;
52.                             p->next->data = temp;
```

```
                        }
                        p = p->next;
                    }
                    tail = p;
                    p = list;
                }
                extreme = list->data.score[code_num].english;
                break;
            case 2:
                while (p != tail)
                {
                    while (p->next != tail)
                    {
                        if (p->data.score[code_num].programming < p->next->data.score[code_
num].programming)
                        {
                            temp = p->data;
                            p->data = p->next->data;
                            p->next->data = temp;
                        }
                        p = p->next;
                    }
                    tail = p;
                    p = list;
                }
                extreme = list->data.score[code_num].programming;
                break;
            default:
                printf("\n\t\t\t\t\t\t      无效指令!");
                getchar();
                break;
            }
        } while (choice != 0 && choice != 1 && choice != 2);
        return extreme;
    }
    else if (mode == 1)//MIN
    {
        for (; list; list = list->next);
        do {
            switch (choice)
            {
            case 0:
```

```
94.                 while (p != tail)
95.                 {
96.                     while (p->next != tail)
97.                     {
98.                         if (p->data.score[code_num].math < p->next->data.score[code_num].ma
    th)
99.                         {
100.                             temp = p->data;
101.                             p->data = p->next->data;
102.                             p->next->data = temp;
103.                         }
104.                         p = p->next;
105.                     }
106.                     tail = p;
107.                     p = list;
108.                 }
109.                 extreme = list->data.score[code_num].math;
110.                 break;
111.             case 1:
112.                 while (p != tail)
113.                 {
114.                     while (p->next != tail)
115.                     {
116.                         if (p->data.score[code_num].english < p->next->data.score[code_nu
    m].english)
117.                         {
118.                             temp = p->data;
119.                             p->data = p->next->data;
120.                             p->next->data = temp;
121.                         }
122.                         p = p->next;
123.                     }
124.                     tail = p;
125.                     p = list;
126.                 }
127.                 extreme = list->data.score[code_num].english;
128.                 break;
129.             case 2:
130.                 while (p != tail)
131.                 {
132.                     while (p->next != tail)
133.                     {
```

```
                        if (p->data.score[code_num].programming < p->next->data.score[cod
    e_num].programming)
                        {
                            temp = p->data;
                            p->data = p->next->data;
                            p->next->data = temp;
                        }
                        p = p->next;
                    }
                    tail = p;
                    p = list;
                }
                extreme = list->data.score[code_num].programming;
                break;
            default:
                printf("\n\t\t\t\t\t\t      无效指令!");
                getchar();
                break;
            }
        } while (choice != 0 && choice != 1 && choice != 2);
            return extreme;
    }
    return extreme;
}
```

4.8.5 班级均分统计模块

```
//班级均分统计
double average_all(Node* list, int choice, int code_num)//mode==0 是仅仅用于计算 avr,mode==1
用于完整版本输出 average
{
    double avr = 0;
    int sum = 0, cnt = 0;
    do {
        switch (choice)
        {
            case 0:
                for (; list->next; list = list->next)
                {
                    if (list->data.score[code_num].math != 0)
                    {
                        sum += list->data.score[code_num].math;
                        cnt++;
```

```
                    }
                }
                avr = sum / cnt;
                break;
            case 1:
                for (; list->next; list = list->next)
                {
                    if (list->data.score[code_num].english != 0)
                    {
                        sum += list->data.score[code_num].english;
                        cnt++;
                    }
                }
                avr = sum / cnt;
                break;
            case 2:
                for (; list->next; list = list->next)
                {
                    if (list->data.score[code_num].programming != 0)
                    {
                        sum += list->data.score[code_num].programming;
                        cnt++;
                    }
                }
                avr = sum / cnt;
                break;
            case 3:
                for (; list->next; list = list->next)
                {
                    if ((list->data.score[code_num].math != 0) && (list->data.score[code_num].english != 0) && (list->data.score[code_num].programming != 0))
                    {
                        sum += (list->data.score[code_num].math + list->data.score[code_num].english + list->data.score[code_num].programming);
                        cnt++;
                    }
                }
                avr = sum / cnt;
                break;
            default:
                printf("\n\t\t\t\t\t\t\t      无效指令!");
                getchar();
```

```
        }
    } while (choice != 0 && choice != 1 && choice != 2 && choice != 3);
    return avr;
}
```

4.8.6 个人均分统计

```
//个人均分统计
double average_one(Node* list, int choice)
{
    float sum = 0, avr = 0, cnt = 0;
    do{
        switch (choice)
        {
        case 0:
            for (int i = 0; i < 30; i++)
            {
                if (list->data.score[i].math != 0)
                {
                    sum += list->data.score[i].math;
                    cnt++;
                }
            }
            avr = sum / cnt;
            break;
        case 1:
            for (int i = 0; i < 30; i++)
            {
                if (list->data.score[i].english != 0)
                {
                    sum += list->data.score[i].english;
                    cnt++;
                }
            }
            avr = sum / cnt;
            break;
        case 2:
            for (int i = 0; i < 30; i++)
            {
                if (list->data.score[i].programming != 0)
                {
                    sum += list->data.score[i].programming;
                    cnt++;
```

```
                }
            }
            avr = sum / cnt;
            break;
        case 3:
            for (int i = 0; i < 30; i++)
            {
                if ((list->data.score[i].math != 0) && (list->data.score[i].english != 0) &
& (list->data.score[i].programming != 0))
                {
                    sum += (list->data.score[i].math + list->data.score[i].english + list->
data.score[i].programming);
                    cnt++;
                }
            }
            avr = sum / cnt;
            break;
        default:
            printf("\n\t\t\t\t\t\t      无效指令!");
            getchar();
            break;
        }
    }while (choice != 0 && choice != 1 && choice != 2 && choice != 3);
    return avr;
}
```

4.8.7 班级方差统计

```
//班级方差统计
double variance_all(Node* phead)
{
    double temp = 0;
    int cnt = 0,code_num,choice;
    Node* p = phead;
    double sum_variance_all = 0;
    system("cls");
    printf("\n\n");
    printf("\t\t\t\t  ┌ ==============================○●○======┐\n");
    printf("\t\t\t\t  |    电子科技大学 1605 班学生成绩管理系统-教师    |\n");
    printf("\t\t\t\t  └ ======○●○==============================┘\n");
    printf("\n\n");
    printf("\n\n");
    printf("\t\t\t\t\t\t 您想算第几次成绩的方差:");
```

```
    scanf("%d", &code_num);
    double avr;
    code_num--;
    do {
        system("cls");
        printf("\n\n");
        printf("\t\t\t\t  ┌ ==============================○●○●======┐\n");
        printf("\t\t\t\t  │    电子科技大学 1605 班学生成绩管理系统-教师    │\n");
        printf("\t\t\t\t  └ ======○●○●==============================┘\n");
        printf("\n\n");
        printf("\n\n");
        printf("\t\t\t\t\t\t\tMENU\n\n");
        printf("\t\t\t\t\t\t 0——>数学波动指数\n\n");
        printf("\t\t\t\t\t\t 1——>英语波动指数\n\n");
        printf("\t\t\t\t\t\t 2——>C 语言波动指数\n\n");
        printf("\t\t\t\t\t\t 3——>总分波动指数\n\n");
        printf("\t\t\t\t\t\t    输入指令编号：\n");
        printf("\t\t\t\t\t\t          ");
        scanf("%d", &choice);
        avr = average_all(phead, choice, code_num);
        //printf("%.2lf", avr);
        switch (choice)
        {
            case 0:
                for (; p->next; p = p->next)
                {
                    if (p->data.score[code_num].math != 0)
                    {
                        temp = pow(p->data.score[code_num].math - avr, 2);
                        sum_variance_all += temp;
                        cnt++;
                        //printf("%.2lf\n", temp);
                        //printf("%.2lf\n", sum_variance_all);
                    }
                }
                s_m = sum_variance_all / cnt;
                    return sum_variance_all / cnt;
                break;
            case 1:
                for (; p->next; p = p->next)
                {
                    if (p->data.score[code_num].english != 0)
```

```
                    {
                        temp = pow(p->data.score[code_num].english - avr, 2);
                        sum_variance_all += temp;
                        cnt++;
                    }
                }
                s_e = sum_variance_all / cnt;
                    return sum_variance_all / cnt;
                break;
            case 2:
                for (; p->next; p = p->next)
                {
                    if (p->data.score[code_num].programming != 0)
                    {
                        temp = pow(p->data.score[code_num].programming - avr, 2);
                        sum_variance_all += temp;
                        cnt++;
                    }
                }
                s_c = sum_variance_all / cnt;
                    return sum_variance_all / cnt;
                break;
            case 3:
                for (; p->next; p = p->next)
                {
                    if ((p->data.score[code_num].math != 0) && (p->data.score[code_num].eng
lish != 0) && (p->data.score[code_num].programming != 0))
                    {
                        temp = pow(p->data.score[code_num].programming + p->data.score[code
_num].math + p->data.score[code_num].english - avr, 2);
                        sum_variance_all += temp;
                        cnt++;
                    }
                }
                s_a = sum_variance_all / cnt;
                    return sum_variance_all / cnt;
                break;
            default:
                printf("\n\t\t\t\t\t\t\t      无效指令!");
                getchar();
                break;
        }
```

```
    } while (choice != 0 && choice != 1 && choice != 2 && choice != 3);
    return sum_variance_all / cnt;
}
```

4.8.8 个人方差统计

```
//学生成绩方差
double variance_one(Node* p,int choice)
{
    double temp = 0;
    int cnt = 0;
    double sum_variance_all = 0;
    double avr;
    avr = average_one(p, choice);
    do{
        switch (choice)
        {
        case 0:
            for (int i=0; i<30; i++)
            {
                if (p->data.score[i].math != 0)
                {
                    temp = pow(p->data.score[i].math - avr, 2);
                    sum_variance_all += temp;
                    cnt++;
                }
            }
            s_m = sum_variance_all / cnt;
            return sum_variance_all / cnt;
            break;
        case 1:
            for (int i=0; i<30; i++)
            {
                if (p->data.score[i].english != 0)
                {
                    temp = pow(p->data.score[i].english - avr, 2);
                    sum_variance_all += temp;
                    cnt++;
                }
            }
            s_e = sum_variance_all / cnt;
            return sum_variance_all / cnt;
            break;
```

```
        case 2:
            for (int i=0; i<30; i++)
            {
                if (p->data.score[i].programming != 0)
                {
                    temp = pow(p->data.score[i].programming - avr, 2);
                    sum_variance_all += temp;
                    cnt++;
                }
            }
            s_c = sum_variance_all / cnt;
            return sum_variance_all / cnt;
            break;
        default:
            printf("\n\t\t\t\t\t\t\t      无效指令!");
            getchar();
            break;
        }
    } while (choice != 0 && choice != 1 && choice != 2);
    return sum_variance_all / cnt;
}
```

4.8.9 生成成绩报告

```
//生成成绩报告
void report(int code_num)
{
    Node* list;
    list = search(user);
    avr_m = average_one(list, 0);
    avr_e = average_one(list, 1);
    avr_c = average_one(list, 2);
    s_m = variance_one(list, 0);
    s_e = variance_one(list, 1);
    s_c = variance_one(list, 2);
    member = statistics();
    //printf("%d\n", member);
    put_in_order(head, 0, code_num);
    ranking_m = rank(user);
    put_in_order(head, 1, code_num);
    ranking_e = rank(user);
    put_in_order(head, 2, code_num);
    ranking_c = rank(user);
```

```
    printf("\n\t\t\t\t\t\t\t    你在第%d 次考试中：\n", code_num);
    printf("\t\t\t 数学得分%d，班级排名第%d,", list->data.score[code_num - 1].math, ranking_m);
    if (((float)ranking_m / (float)member) <= 0.2)
        printf("表现优秀，希望继续保持，尽可能努力在难题上有所突破！\n");
    else  if (((float)ranking_m / (float)member) > 0.2 && ((float)ranking_m / (float)member) < 0.6)
        printf("表现不错，希望做得更好，争取下一次考试能再上一层楼！\n");
    else if (((float)ranking_m / (float)member) >= 0.6)
        printf("不太理想，需要好好找下原因，不要急于求成，要多巩固基础哦！\n");

    printf("\t\t\t 英语得分%d，班级排名第%d,", list->data.score[code_num - 1].english, ranking_e);
    if (((float)ranking_e / (float)member) <= 0.2)
        printf("表现优秀，希望继续保持，尽可能努力在难题上有所突破！\n");
    else  if (((float)ranking_e / (float)member) > 0.2 && ((float)ranking_e / (float)member) < 0.6)
        printf("表现不错，希望做得更好，争取下一次考试能再上一层楼！\n");
    else if (((float)ranking_e / (float)member) >= 0.6)
        printf("不太理想，需要好好找下原因，不要急于求成，要多巩固基础哦！\n");

    printf("\t\t\tC 语言得分%d，班级排名第%d,", list->data.score[code_num - 1].programming, ranking_c);
    if (((float)ranking_c / (float)member) <= 0.2)
        printf("表现优秀，希望继续保持，尽可能努力在难题上有所突破！\n");
    else  if (((float)ranking_c / (float)member) > 0.2 && ((float)ranking_c / (float)member) < 0.6)
        printf("表现不错，希望做得更好，争取下一次考试能再上一层楼！\n");
    else if (((float)ranking_c / (float)member) >= 0.6)
        printf("不太理想，需要好好找下原因，不要急于求成，要多巩固基础哦！\n");

}
```

4.8.10 数据分析菜单界面

```
//分析界面
int analyze(void)
{
    int choice,change = 0;
    headview();
    if (!type)
    {
        Node* temp;
```

```
9.          temp = search(user);
10.         while (1)
11.         {
12.             system("cls");
13.             printf("\n\n");
14.             printf("\t\t\t\t  ┌ ==============================○●○●======┐\n");
15.             printf("\t\t\t\t  |    电子科技大学 1605 班学生成绩管理系统-学生    |\n");
16.             printf("\t\t\t\t  └ ======○●○●==============================┘\n");
17.             printf("\n\n");
18.             printf("\t\t\t\t 姓名：%s", temp->data.name);
19.             printf("\t\t\t\t 学号：%s", temp->data.id);
20.             printf("\n\n");
21.             printf("\t\t\t\t\t\t\tMENU\n\n");
22.             printf("\t\t\t\t\t\t 1—>你的平均成绩\n\n");
23.             printf("\t\t\t\t\t\t 2—>成绩波动指数\n\n");
24.             printf("\t\t\t\t\t\t 3—>你的成绩报告\n\n");
25.             printf("\t\t\t\t\t\t 4—>  你的位次\n\n");
26.             printf("\t\t\t\t\t\t    输入指令编号：\n");
27.             printf("\t\t\t\t\t\t          ");
28.             scanf("%d", &choice);
29.             switch (choice)
30.             {
31.             case 1:change = 1;
32.                 break;
33.             case 2:change = 2;
34.                 break;
35.             case 3:change = 3;
36.                 break;
37.             case 4:change = 4;
38.                 break;
39.             default:
40.                 printf("\n\t\t\t\t\t\t      无效指令!");
41.                 getchar();
42.                 break;
43.             }
44.             if (choice == 1 || choice == 2 || choice == 3 || choice == 4)
45.                 break;
46.         }
47.     }
48.     else
49.     {
50.         while (1)
```

```
        {
            system("cls");
            printf("\n\n");
            printf("\t\t\t\t  ┌ ==============================○●○●======┐\n");
            printf("\t\t\t\t  |    电子科技大学 1605 班学生成绩管理系统-教师    |\n");
            printf("\t\t\t\t  └ ======○●○●==============================┘\n");
            printf("\n\n");
            printf("\n\n");
            printf("\t\t\t\t\t\t\tMENU\n\n");
            printf("\t\t\t\t\t\t 1—>全班平均成绩\n\n");
            printf("\t\t\t\t\t\t 2—>成绩波动指数\n\n");
            printf("\t\t\t\t\t\t 3—> 最高最低分\n\n");
            printf("\t\t\t\t\t\t    输入指令编号: \n");
            printf("\t\t\t\t\t\t          ");
            scanf("%d", &choice);
            switch (choice)
            {
            case 1:change = 5;
                break;
            case 2:change = 6;
                break;
            case 3:change = 7;
                break;
            default:
                printf("\t\t\t\t\t\t      无效指令!");
                getchar();
                break;
            }
            if (choice == 1 || choice == 2 || choice == 3 )
                break;
        }
    }
    return change;
}
```

4.8.11 数据分析实现函数

```
//数据分析实现
void f_analyze(int change)
{
    Node* temp;
    int choice, code_num,r_num,extreme, code_data;
    temp = search(user);
```

```
    system("cls");
    printf("\n\n");
    if (type)
    {
        printf("\t\t\t\t     ┌ ==============================○●○●======┐\n");
        printf("\t\t\t\t     |   电子科技大学 1605 班学生成绩管理系统-教师   |\n");
        printf("\t\t\t\t     └ ======○●○●==============================┘\n");
        printf("\n\n");
    }
    else
    {
        printf("\t\t\t\t     ┌ ==============================○●○●======┐\n");
        printf("\t\t\t\t     |   电子科技大学 1605 班学生成绩管理系统-学生   |\n");
        printf("\t\t\t\t     └ ======○●○●==============================┘\n");
        printf("\n\n");
        printf("\t\t\t\t   姓名：%s", temp->data.name);
        printf("\t\t\t\t   学号：%s", temp->data.id);
        printf("\n\n");
    }
    while (1)
    {
        switch (change)
        {
        case 1:
            printf("\n\n");
            printf("\t\t\t\t\t\t    0—>数学平均成绩\n\n");
            printf("\t\t\t\t\t\t    1—>英语平均成绩\n\n");
            printf("\t\t\t\t\t\t    2—>C 语言平均成绩\n\n");
            printf("\t\t\t\t\t\t    3—>总分平均成绩\n\n");
            printf("\t\t\t\t\t\t       输入指令编号：\n");
            printf("\t\t\t\t\t\t             ");
            scanf("%d", &choice);
            while (1)
            {
                switch (choice)
                {
                case 0:
                    avr_m = average_one(temp, choice);
                    system("cls");
                    printf("\n\n");
                    printf("\t\t\t\t     ┌ ==============================○●○●======┐\n");
```

```
48.                     printf("\t\t\t\t     |    电子科技大学 1605 班学生成绩管理系统-学
    生   |\n");
49.                     printf("\t\t\t\t     └ ======o●o●==============================┘\n");
50.                     printf("\n\n");
51.                     printf("\t\t\t\t   姓名：%s", temp->data.name);
52.                     printf("\t\t\t\t   学号：%s", temp->data.id);
53.                     printf("\n\n");
54.                     printf("\n\n");
55.                     printf("\t\t\t\t\t\t      数学平均分：%.2lf\n\n\n\n\n", avr_m);
56.                     break;
57.                 case 1:
58.                     avr_e = average_one(temp, choice);
59.                     system("cls");
60.                     printf("\n\n");
61.                     printf("\t\t\t\t     ┌ ==============================o●o●======┐\n");
62.                     printf("\t\t\t\t     |    电子科技大学 1605 班学生成绩管理系统-学
    生   |\n");
63.                     printf("\t\t\t\t     └ ======o●o●==============================┘\n");
64.                     printf("\n\n");
65.                     printf("\t\t\t\t   姓名：%s", temp->data.name);
66.                     printf("\t\t\t\t   学号：%s", temp->data.id);
67.                     printf("\n\n");
68.                     printf("\n\n");
69.                     printf("\t\t\t\t\t\t      C 语言平均分：%.2lf\n\n\n\n\n", avr_e);
70.                     break;
71.                 case 2:
72.                     avr_c = average_one(temp, choice);
73.                     system("cls");
74.                     printf("\n\n");
75.                     printf("\t\t\t\t     ┌ ==============================o●o●======┐\n");
76.                     printf("\t\t\t\t     |    电子科技大学 1605 班学生成绩管理系统-学
    生   |\n");
77.                     printf("\t\t\t\t     └ ======o●o●==============================┘\n");
78.                     printf("\n\n");
79.                     printf("\t\t\t\t   姓名：%s", temp->data.name);
80.                     printf("\t\t\t\t   学号：%s", temp->data.id);
81.                     printf("\n\n");
82.                     printf("\n\n");
83.                     printf("\t\t\t\t\t\t      英语平均分：%.2lf\n\n\n\n\n", avr_c);
84.                     break;
85.                 case 3:
86.                     avr_a = average_one(temp, choice);
```

```
87.                     system("cls");
88.                     printf("\n\n");
89.                     printf("\t\t\t\t     ┌ ==============================○●○●======┐\n");
90.                     printf("\t\t\t\t     |    电子科技大学 1605 班学生成绩管理系统-学
    生    |\n");
91.                     printf("\t\t\t\t     └ ======○●○●==============================┘\n");
92.                     printf("\n\n");
93.                     printf("\t\t\t\t    姓名：%s", temp->data.name);
94.                     printf("\t\t\t\t    学号：%s", temp->data.id);
95.                     printf("\n\n");
96.                     printf("\n\n");
97.                     printf("\t\t\t\t\t\t       总分平均分：%.2lf\n\n\n\n\n", avr_a);
98.                     break;
99.                 default:
100.                     printf("\n\t\t\t\t\t\t          无效指令!\n");
101.                    getchar();
102.                    break;
103.                }
104.                if (choice == 0 || choice == 1 || choice == 2 || choice == 3)
105.                    break;
106.            }
107.                break;
108.        case 2://波动指数
109.            system("cls");
110.            printf("\n\n");
111.            printf("\t\t\t\t     ┌ ==============================○●○●======┐\n");
112.            printf("\t\t\t\t     |    电子科技大学 1605 班学生成绩管理系统-学生    |\n");
113.            printf("\t\t\t\t     └ ======○●○●==============================┘\n");
114.            printf("\n\n");
115.            printf("\t\t\t\t    姓名：%s", temp->data.name);
116.            printf("\t\t\t\t    学号：%s", temp->data.id);
117.            printf("\n\n");
118.            printf("\t\t\t\t\t\t\t    MENU\n\n");
119.            printf("\t\t\t\t\t\t     0—>数学波动指数\n\n");
120.            printf("\t\t\t\t\t\t     1—>英语波动指数\n\n");
121.            printf("\t\t\t\t\t\t     2—>C 语言波动指数\n\n");
122.            printf("\t\t\t\t\t\t        输入指令编号：\n");
123.            printf("\t\t\t\t\t\t              ");
124.            scanf("%d", &choice);
125.            s = variance_one(search(user),choice);
126.            printf("\t\t\t\t\t\t        波动指数：%.2lf\n\n\n\n\n", s);
127.            break;
```

```
128.        case 3://成绩报告
129.            system("cls");
130.            printf("\n\n");
131.            printf("\t\t\t\t     ┌ ==============================○●○●======┐\n");
132.            printf("\t\t\t\t     |    电子科技大学 1605 班学生成绩管理系统-学生    |\n");
133.            printf("\t\t\t\t     └ ======○●○●==============================┘\n");
134.            printf("\n\n");
135.            printf("\t\t\t\t   姓名：%s", temp->data.name);
136.            printf("\t\t\t\t   学号：%s", temp->data.id);
137.            printf("\n\n");
138.            printf("\n\n\n\n\t\t\t\t\t\t   您想生成第几次成绩报告:");
139.            scanf("%d", &code_num);
140.            report(code_num);
141.            break;
142.        case 4://位次排名
143.            printf("\t\t\t\t\t\t   0-->按数学成绩排序输出\n\n");
144.            printf("\t\t\t\t\t\t   1-->按英语成绩排序输出\n\n");
145.            printf("\t\t\t\t\t\t   2-->按 c 语言成绩排序输出\n\n");
146.            printf("\t\t\t\t\t\t         请输入：");
147.            scanf("%d", &code_data);
148.            printf("\n\t\t\t\t\t\t   您想按第几次成绩排序:");
149.            scanf("%d", &code_num);
150.            put_in_order(head,code_data,code_num);
151.            r_num = rank(user);
152.            printf("\n\n\t\t\t\t\t\t       本次排名：%d\n\n\n\n\n",r_num);
153.            break;
154.        case 5:
155.            printf("\t\t\t\t\t\t    0—->数学平均成绩\n\n");
156.            printf("\t\t\t\t\t\t    1—->英语平均成绩\n\n");
157.            printf("\t\t\t\t\t\t    2—->C 语言平均成绩\n\n");
158.            printf("\t\t\t\t\t\t    3—->总分平均成绩\n\n");
159.            printf("\t\t\t\t\t\t       输入指令编号：\n");
160.            printf("\t\t\t\t\t\t            ");
161.            scanf("%d", &choice);
162.            printf("\t\t\t\t\t\t   您想算第几次成绩：");
163.            scanf("%d", &code_num);
164.            code_num--;
165.            while (1)
166.            {
167.                switch (choice)
168.                {
169.                case 0:
```

```
170.                    avr_m = average_all(head, choice, code_num);
171.                    system("cls");
172.                    printf("\n\n");
173.                    printf("\t\t\t\t      ┌ ==============================○●○●======┐\n");
174.                    printf("\t\t\t\t      |    电子科技大学 1605 班学生成绩管理系统-教
    师    |\n");
175.                    printf("\t\t\t\t      └ ======○●○●==============================┘\n");
176.                    printf("\n\n");
177.                    printf("\n\n");
178.                    printf("\t\t\t\t\t\t       数学平均分：%.2lf\n\n\n\n\n", avr_m);
179.                    break;
180.                case 1:
181.                    avr_e = average_all(head, choice, code_num);
182.                    system("cls");
183.                    printf("\n\n");
184.                    printf("\t\t\t\t      ┌ ==============================○●○●======┐\n");
185.                    printf("\t\t\t\t      |    电子科技大学 1605 班学生成绩管理系统-教
    师    |\n");
186.                    printf("\t\t\t\t      └ ======○●○●==============================┘\n");
187.                    printf("\n\n");
188.                    printf("\n\n");
189.                    printf("\t\t\t\t\t\t       C 语言平均分：%.2lf\n\n\n\n\n", avr_e);
190.                    break;
191.                case 2:
192.                    avr_c = average_all(head, choice, code_num);
193.                    system("cls");
194.                    printf("\n\n");
195.                    printf("\t\t\t\t      ┌ ==============================○●○●======┐\n");
196.                    printf("\t\t\t\t      |    电子科技大学 1605 班学生成绩管理系统-教
    师    |\n");
197.                    printf("\t\t\t\t      └ ======○●○●==============================┘\n");
198.                    printf("\n\n");
199.                    printf("\n\n");
200.                    printf("\t\t\t\t\t\t       英语平均分：%.2lf\n\n\n\n\n", avr_c);
201.                    break;
202.                case 3:
```

```
203.                     avr_a = average_all(head, choice, code_num);
204.                     system("cls");
205.                     printf("\n\n");
206.                     printf("\t\t\t\t     ┌ ==============================○●○●======┐\n");
207.                     printf("\t\t\t\t     |    电子科技大学 1605 班学生成绩管理系统-教
    师    |\n");
208.                     printf("\t\t\t\t     └ ======○●○●==============================┘\n");
209.                     printf("\n\n");
210.                     printf("\n\n");
211.                     printf("\t\t\t\t\t\t      总分平均分：%.2lf\n\n\n\n\n", avr_a);
212.                     break;
213.                 default:
214.                     printf("\n\t\t\t\t\t\t          无效指令!");
215.                     getchar();
216.                     break;
217.                 }
218.                 if (choice == 0 || choice == 1 || choice == 2 || choice == 3)
219.                     break;
220.             }
221.             break;
222.         case 6:
223.             s = variance_all(head);
224.             printf("\t\t\t\t\t\t    波动指数：%.2lf\n\n\n\n\n", s);
225.             break;
226.         case 7://最值
227.             system("cls");
228.             printf("\n\n");
229.             printf("\t\t\t\t  ┌ ==============================○●○●======┐\n");
230.             printf("\t\t\t\t  |    电子科技大学 1605 班学生成绩管理系统-教师    |\n");
231.             printf("\t\t\t\t  └ ======○●○●==============================┘\n");
232.             printf("\n\n");
233.             printf("\n\n");
234.             printf("\t\t\t\t\t\t 0——>数学成绩最值\n\n");
235.             printf("\t\t\t\t\t\t 1——>英语成绩最值\n\n");
236.             printf("\t\t\t\t\t\t 2——>C 语言成绩最值\n\n");
237.             printf("\t\t\t\t\t\t    输入指令编号：\n");
238.             printf("\t\t\t\t\t\t          ");
239.             scanf("%d", &choice);
240.             printf("\t\t\t\t\t\t 您想算第几次成绩：");
241.             scanf("%d", &code_num);
```

```
            extreme = extreme_value(head, choice, code_num);
            printf("\t\t\t\t\t\t            %d\n\n\n\n\n\n",extreme);
            break;
        default:
            printf("\n\t\t\t\t\t\t      无效指令!");
            getchar();
            break;
        }
        if (change == 7 || change == 1 || change == 2 || change == 3 || change == 4 || ch
    ange == 5 || change == 6)
            break;
    }
    jump();
}
```

4.8.12 GPA 计算函数

```
//GPA
void GPA(Node* phead)
{
    system("cls");
    headview();
    printf("\n\n");
    printf("\t\t\t\t  ┌ ==============================○●○●======┐\n");
    printf("\t\t\t\t  |    电子科技大学 1605 班学生成绩管理系统-学生    |\n");
    printf("\t\t\t\t  └ ======○●○●==============================┘\n");
    printf("\n\n");
    Node* p;
    int choice;
    float avr;
    p = search(user);
    printf("\t\t\t\t\t\t 您想查询第几次成绩:");
    scanf("%d", &choice);
    choice--;
    avr = (p->data.score[choice].math + p->data.score[choice].english + p->data.score[choic
e].programming) / 3;
    if (avr >= 85)
    {
        printf("\n\t\t\t\t\t\t\t  你太棒了，绩点 4.0");
    }
    else if (avr >= 60)
    {
        printf("\n\t\t\t\t\t\t\t  你的绩点为：%.1f", 4.0 - (85.0 - avr) * 0.1);
```

```
    }
    else
    {
        printf("\n\t\t\t\t\t\t\t      你挂科了");
    }
}
```

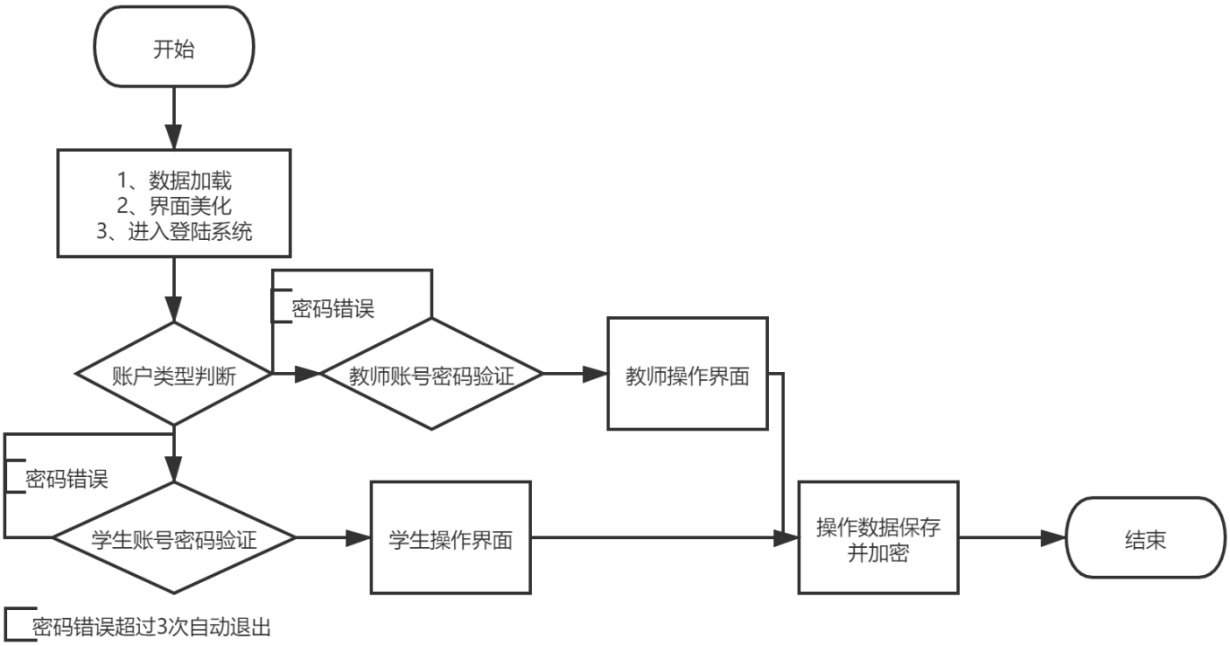
4.9 安全加密模块

```
//下面的函数用于将数组随机化。
void f_rand(int a[], int* t) { //随机数
    srand((unsigned)time(NULL));
    for (int i = 0; i < length; i = i + 1) {
        a[i] = rand();
        *t = rand() % 400 - 200;
    }
}

//下面的函数用于加密。
void encrypt(char* location_1, char* code_location) { //加密。加密前的文本位置，密码位置
    FILE* before_encrypt, * after_encrypt, * encrypt; //加密前，After_Encrypt，加密
    char* location_2 = "After_Encrypt.txt"; //After_Encrypt 的文本位置
    int ch;
    int a[length]; //存储密码的数组
    int t; //再次加密
    f_rand(a, &t);

    before_encrypt = fopen(location_1, "r+");
    if (before_encrypt == NULL) return;
    after_encrypt = fopen(location_2, "w");
    for (int j = 0;; j = j + 1) {
        ch = fgetc(before_encrypt);
        if (ch == EOF) break;
        ch = ch + a[j % length];
        fprintf(after_encrypt, "%d ", ch);
    }
    fclose(before_encrypt);
    fclose(after_encrypt);

    encrypt = fopen(code_location, "w");
    fprintf(encrypt, "%d ", t);
    for (int j = length; j > 0; j = j - 1) {
```

```
34.         fprintf(encrypt, "%d ", a[j - 1] + t);
35.     }
36.     fclose(encrypt);
37.     remove(location_1);
38.     rename(location_2, location_1);
39. }
40.
41. //下面的函数用于解密。
42. void decode(char* location_2, char* code_location) { //解密。After_Encrypt 的位置，密码位置
43.     FILE* before_decode, * after_decode, * code; //解密前，After_Decode，密码
44.     char* location_3 = "After_Decode.txt"; //After_Decode 的位置
45.     int b[length]; //存储已有密码的数组
46.     int ch;
47.     int t; //再次加密
48.
49.     code = fopen(code_location, "r");
50.     if (code == NULL) return;
51.     fscanf(code, " %d", &t);
52.     for (int j = length; j > 0; j = j - 1)
53.         fscanf(code, " %d", &b[j - 1]);
54.     fclose(code);
55.
56.     before_decode = fopen(location_2, "r");
57.     if (before_decode == NULL) return;
58.     after_decode = fopen(location_3, "w");
59.     for (int j = 0; fscanf(before_decode, " %d", &ch) != EOF; j = j + 1) {
60.         ch = ch - (b[j % length] - t);
61.         fputc(ch, after_decode);
62.     }
63.     fclose(before_decode);
64.     fclose(after_decode);
65.     remove(code_location);
66.     remove(location_2);
67.     rename(location_3, location_2);
68. }
```

4.10　main 函数总体实现

用程序流程图说明实现思路



具体函数如下：

```
1.  int main(void)
2.  {
3.      //变量定义
4.      int choice,change;
5.      char id[MAX];
6.      //窗台美化
7.      surface();
8.
9.      //程序初始化
10.     init_node();
11.
12.     //用户登录|菜单选择
13.     type = login();
14.     //type = 1;
15.
16.     while (1)
17.     {
18.     if (type)
```

```
        choice = awelcome();
    else
        choice = swelcome();


        //功能实现
        switch (choice)
        {
        case 1:
            system("cls");
            headview();
            printf("\n\n\t\t\t\t\t\t\t 请输入学号:  ");
            error=scanf("%s", id);
            print_one(search(id));
            jump();
            break;
        case 2:
            system("cls");
            headview();
            printf("\n\n");
            printf("\t\t\t\t      ┌ ==============================○●○●======┐\n");
            printf("\t\t\t\t      |    电子科技大学 1605 班学生成绩管理系统-教师    |\n");
            printf("\t\t\t\t      └ ======○●○●==============================┘\n");
            printf("\n\n");
            add_student();
            jump();
            break;
        case 3:
            system("cls");
            headview();
            printf("\n\n");
            printf("\t\t\t\t      ┌ ==============================○●○●======┐\n");
            printf("\t\t\t\t      |    电子科技大学 1605 班学生成绩管理系统-教师    |\n");
            printf("\t\t\t\t      └ ======○●○●==============================┘\n");
            printf("\n\n");
            write();
            jump();
            break;
        case 4:
            system("cls");
            headview();
            printf("\n\n");
```

```
61.             printf("\t\t\t\t      ┌ ==============================○●○●======┐\n");
62.             printf("\t\t\t\t      |    电子科技大学 1605 班学生成绩管理系统-教师    |\n");
63.             printf("\t\t\t\t      └ ======○●○●==============================┘\n");
64.             printf("\n\n");
65.             printf("\n\n\n\t\t\t\t\t\t\t 请输入学号：");
66.             error=scanf("%s", id);
67.             delet(&head, id);
68.             jump();
69.             break;
70.         case 5:
71.             change=analyze();
72.             f_analyze(change);
73.             break;
74.         case 7:
75.             system("cls");
76.             save();
77.             getchar();
78.             headview();
79.             printf("\n\n\n\n\n\n");
80.             printf("\t\t\t\t\t\t\t 保存成功\n\n\n\n\n");
81.             return 0;
82.             break;
83.         case 8:
84.             GPA(head);
85.             getchar();
86.             getchar();
87.             break;
88.         case 6:
89.             system("cls");
90.             headview();
91.             printf("\n\n");
92.             printf("\t\t\t\t      ┌ ==============================○●○●======┐\n");
93.             printf("\t\t\t\t      |      电子科技大学 1605 班学生成绩管理系统       |\n");
94.             printf("\t\t\t\t      └ ======○●○●==============================┘\n");
95.             printf("\n\n");
96.             printf("\n\n\n\t\t\t\t\t\t\t 请输入新密码：");
97.             scanf("%s", PASS[account]);
98.             jump();
99.             break;
100.         default:
101.             printf("\t\t\t\t\t 请重新输入：");
102.             error=scanf("%d", &choice);
```

```
            break;
        }
    }
    return 0;
}
```

5 关键代码分析

安全模块

```
void f_rand(int a[], int* t) { //随机数
    srand((unsigned)time(NULL));
    for (int i = 0; i < length; i = i + 1) {
        a[i] = rand();
        *t = rand() % 400 - 200;
    }
}
```

这个函数可以生成伪随机数，并存储在数组a中。同时，也生成一个-200~200的随机数，并存储在变量t中。

```
void encrypt(char* location_1, char* code_location) { //加密。加密前的文本位置，密码位置
    FILE* before_encrypt, * after_encrypt, * encrypt; //加密前，After_Encrypt，加密
    char* location_2 = "After_Encrypt.txt"; //After_Encrypt 的文本位置
    int ch;
    int a[length]; //存储密码的数组
    int t; //再次加密
    f_rand(a, &t);

    before_encrypt = fopen(location_1, "r+");
    if (before_encrypt == NULL) return;
    after_encrypt = fopen(location_2, "w");
    for (int j = 0;; j = j + 1) {
        ch = fgetc(before_encrypt);
        if (ch == EOF) break;
        ch = ch + a[j % length];
        fprintf(after_encrypt, "%d ", ch);
    }
    fclose(before_encrypt);
    fclose(after_encrypt);

```

```
    encrypt = fopen(code_location, "w");
    fprintf(encrypt, "%d ", t);
    for (int j = length; j > 0; j = j - 1) {
        fprintf(encrypt, "%d ", a[j - 1] + t);
    }
    fclose(encrypt);
    remove(location_1);
    rename(location_2, location_1);
}
```

首先，打开需要加密的文件。接着，将明文中的每个字符的ASCII码与随机数进行运算，得到加密后的随机数，并将它保存到密文中。然后，将密码加密，并保存到单独的文件中。最后，为了减少文件改变带来的麻烦，删掉明文，同时，把密文的文件名改为明文。

```
void decode(char* location_2, char* code_location) { //解密。After_Encrypt 的位置，密码位置
    FILE* before_decode, * after_decode, * code; //解密前，After_Decode，密码
    char* location_3 = "After_Decode.txt"; //After_Decode 的位置
    int b[length]; //存储已有密码的数组
    int ch;
    int t; //再次加密

    code = fopen(code_location, "r");
    if (code == NULL) return;
    fscanf(code, " %d", &t);
    for (int j = length; j > 0; j = j - 1)
        fscanf(code, " %d", &b[j - 1]);
    fclose(code);

    before_decode = fopen(location_2, "r");
    if (before_decode == NULL) return;
    after_decode = fopen(location_3, "w");
    for (int j = 0; fscanf(before_decode, " %d", &ch) != EOF; j = j + 1) {
        ch = ch - (b[j % length] - t);
        fputc(ch, after_decode);
    }
    fclose(before_decode);
    fclose(after_decode);
    remove(code_location);
    remove(location_2);
    rename(location_3, location_2);
}
```

首先打开密码和密文。然后，将密文与密码进行加密时的逆运算，得到明文的 ASCII 码，进而得到明文。由于每次加密使用的随机数都不同，所以，在最后删掉密码和密文，并把明文的文件名改为之前密文的文件名。这样，在完成加密

和解密工作后，文件的位置和名称都不会变化。

对链表的优化

1、尽可能地简化链表的操作。例如对链表函数的定义尽量采取 void 型，而非返回指针型，让其他操作者能够简易地利用函数，不会因考虑返回值的问题而出错。
2、双向链表的引入使链表操作变得更加灵活。
3、对于排序而言，因为可以反向输出，只需一次从高到低的排序便可以解决次序问题，反向输出便是由低到高；双向链表可以实现更多的功能。
4、对于排序而言，单向链表排序一般由冒泡排序实现，而双向链表可以实现插入排序，希尔排序等排序法。

```
1.  void insert_sort(Node* phead) {
2.  Node* p = phead;//插入排序法
3.  Student temp;//对第一次数学成绩由高到低排序
4.  Node* in_order, * out_order, * temp_order;
5.  for (out_order = p->next; out_order; out_order = out_order->next) {
6.      for (in_order = out_order->prior; in_order; in_order = in_order->prior) { //默认第一个数为有序数
7.          if (in_order->data.score[0].math > out_order->data.score[0].math)
8.              break;
9.      }  //找到分界点
10.     if (!in_order) {
11.         in_order = (Node*)malloc(sizeof(Node)); //此类为特殊情况，当无序值比所有有序值都大时，在链表会导致链表指向空指针，因此需要分配内存空间
12.         in_order->next = p;
13.         p->prior = in_order;
14.     }
15.     if (in_order != out_order->prior) {  //可以减少复杂度，当无序比有序数都小时直接结束
16.         temp = out_order->data;
17.         for (temp_order = out_order->prior; temp_order != in_order; temp_order = temp_order->prior)
18.             temp_order->next->data = temp_order->data;//数据往后移动一位
19.         in_order->next->data = temp; //正式将无序数插入 有序数组
20.     }
21.     p->prior = NULL;//需要让头结点恢复原样
22.  }
23. }
```

对于删除结点等功能，如果是单向链表，需要一个跟随指针记录位置，并且如果删除的是头结点还要特殊处理。

```
1.  void  delet(Node** list, char id[MAX])              //这是单向链表的删除结点法
2.  {
3.    Node* p = (*list);
4.    Node* q=NULL; //跟随指针
5.   if (!strcmp(p->data.id, id)) {       //第一个数据就是所找数据时的特殊情况
6.              (*list) = (*list)->next;    //删除第一个结点
7.          }
8.          else {
9.              for (; p && strcmp(p->data.id, id); q = p, p = p->next);
10.             if (!p)
11.                 printf("unknown id\n"); else {
12.                 q->next = p->next;    //将跟随指针跳过所删除的结点，指向下一个
13.             }
14.         }
15.     }
```

更新后：不需要跟随指针，直接搞定；

```
1.  void  delet(Node* list, char id[MAX])
2.  {
3.      Node *p=list;
4.      while(1){
5.      for(;p&&strcmp(p->data.id,id);p=p->next); //找到相应信息的结点
6.      if(!p){
7.          printf("\t\t\t\t\t\t学号错误!请重新输入学号");//未找到信息时的报错功能
8.          scanf("%s",id);
9.      } else{
10.         p->prior->next=p->next;//链接结点
11.         free(p);//释放结点占据的空间
12.         break;
13.     }
14.   }
15. }
16. }
```

5、排序，希尔排序（不常用）等排序法。

```
1.  Node *getMiddleNode(Node *pList)  //归并排序 对第一次数学成绩由高到低地排序
2.  {
3.      if (pList == NULL)
4.      {
5.          return NULL;
```

```
    }
    Node *pAhead = pList->next;
    Node *pBehind = pList;
    while (pAhead != NULL)
    {
         pAhead = pAhead->next;
         if (pAhead != NULL)
         {
             pAhead = pAhead->next;
             pBehind = pBehind->next;
         }

    }

    return pBehind;
}

Node *MergeList(Node *p1, Node *p2) //合并有序链表，合并之后升序排列
{
    if (NULL == p1)
    {
        return p2;
    }
    if (NULL == p2)
    {
        return p1;
    }

    Node *pLinkA = p1;
    Node *pLinkB = p2;
    Node *pTemp = NULL;
    if (pLinkA->data.score[0].math <= pLinkB->data->data.score[0].math)
    {
        pTemp = pLinkA;
        pLinkA = pLinkA->next;
    }
    else
    {
        pTemp = pLinkB;
        pLinkB = pLinkB->next;
    }

```

```
48.     Node *pHead = pTemp; //初始化头结点，即头结点指向不为空的结点
49.     while (pLinkA && pLinkB)
50.     {
51.         if (pLinkA->data.score[0].math <= pLinkB->data.score[0].math)
52.         {
53.             pTemp->next = pLinkA;
54.             pTemp = pLinkA;
55.             pLinkA = pLinkA->next;
56.         }
57.         else
58.         {
59.             pTemp->next = pLinkB;
60.             pTemp = pLinkB;
61.             pLinkB = pLinkB->next;
62.         }
63.
64.     }
65.
66.     pTemp->next = pLinkA ? pLinkA:pLinkB; //插入多余的链表部分
67.
68.     return pHead;
69. }
70.
71. Node *MergeSort(Node *pList)
72. {
73.     if (pList == NULL || pList->next == NULL)
74.     {
75.         return pList;
76.     }
77.
78.     Node *pMiddle = getMiddleNode(pList); //获取中间结点
79.     Node *pBegin = pList; //链表前半部分，包括中间结点
80.     Node *pEnd = pMiddle->next;//链表后半部分
81.     pMiddle->next = NULL; //必须赋值为空
82.     pBegin = MergeSort(pBegin); //排序前半部分数据
83.     pEnd = MergeSort(pEnd); //排序后半部分数据
84.     return MergeList(pBegin, pEnd); //合并有序链表
85. }
86.
87. }
```

数据分析功能

一. 功能完备齐全，多达十二个成绩信息处理功能

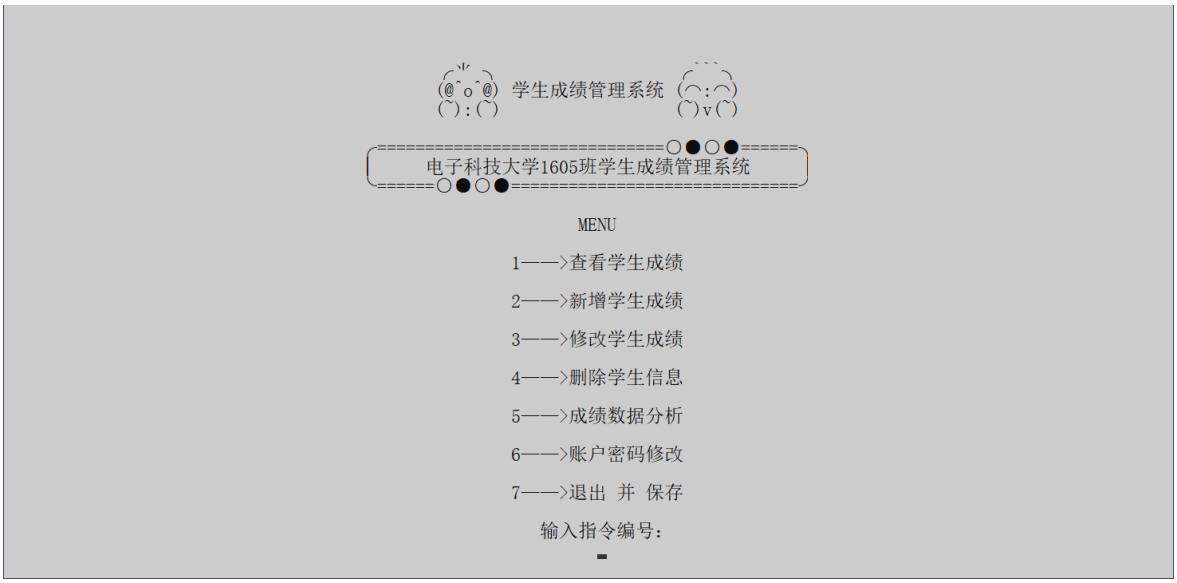
1. 班级平均分，个人平均分，不同次平均分一键生成。3 门科目，30 位成员，30 次成绩统计完备
2. 利用数学公式计算方差，对成绩的波动分析。
3. 成绩管理安全，root 权限和用户权限分离，普通用户没有修改成绩的权限。

二. 人性化分析，从数据中提取信息

1. GPA 绩点计算器，查询每一次考试的折算绩点。
2. 一键生成成绩分析报告，根据排名计算您的成绩档位，提出相应建议。

6 功能测试

教师端界面：


学生成绩管理系统
电子科技大学1605班学生成绩管理系统
MENU
1——>查看学生成绩
2——>新增学生成绩
3——>修改学生成绩
4——>删除学生信息
5——>成绩数据分析
6——>账户密码修改
7——>退出 并 保存
输入指令编号：

查看成绩：


UESTC_1605班成绩管理系统
学生成绩管理系统
请输入学号： 2019091605012
姓名：钟镇阳
学号：2019091605012
查看第几次考试：6
C语言 数学 英语
31 47 37
h-->Home

添加学生信息：

修改成绩：

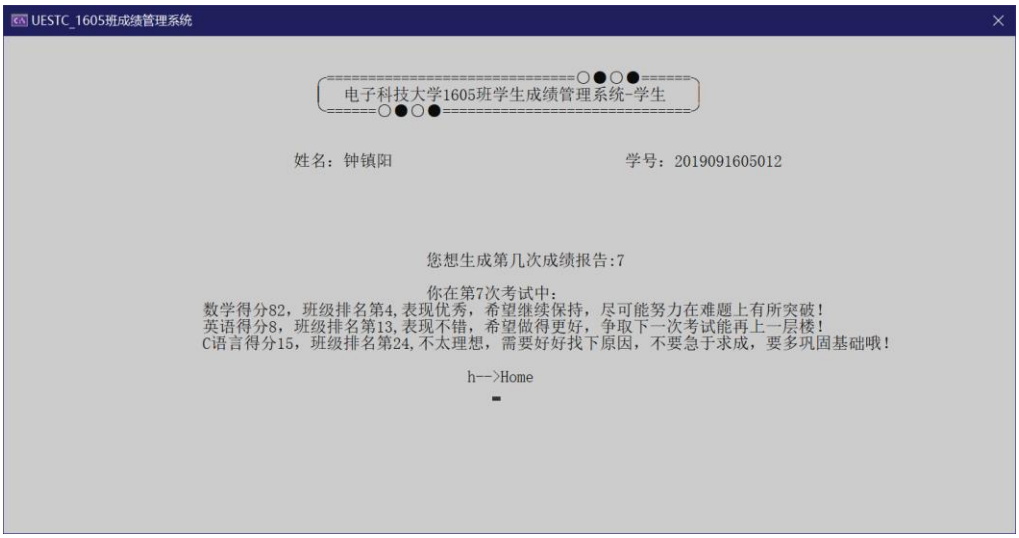
教师端成绩分析：

学生端界面：

成绩查看：

学生成绩分析：

成绩报告生成：



成绩绩点查询：

7 总结

学到了什么？

1、 在技术实践上更加熟悉了 C 语言的操作和实际应用。

2、 学会了比较熟练地使用 C 语言对文件进行相关操作。

3、 通过查阅资料学习了很多库函数的正确使用方法。

4、 了解了基于控制台的美化方法，学会了调用 Windows 库中的函数来实现对窗台的控制。

5、 学习了几种不同的排序算法的具体实现过程，以及应用在数组和链表上的差异。

6、 学会了基于 Git 的代码托管，便于团队的协作开发。

7、 实践了程序设计的基本流程，以及开发方式。

8、 学会了更加合理利用搜索引擎查找资料，查找手册和文档，加快了解决问题的速度。

9、 通过 debug 的过程也学会了许多代码查错的实用方法技巧。

10、 能够比较熟练地使用快捷的 debug 工具，加快问题解决速度。

痛点和难点：

1、 由于知识积累太少，遇到问题花费了大量时间查找文档和资料来解决一个又一个问题。

2、 由于经验欠缺，在设计程序的过程中给自己挖了许多坑。

3、 由于程序最初的整体构架不够优化，导致后面版本迭代时修改起来十分困难。

4、 由于最初设计欠妥，导致代码量过大。

5、 由于庞大的体系和众多模块，有时遇到问题时不能很快地定位错误源头。

如何与他人合作？

1、 结合使用场景首先讨论项目需求，明确目标。

2、 根据需求进行模块化设计，划分具体模块。

3、 在开发时先商量好模块间的连接方案，以便预留接口。

4、 各取所长进行任务分配，将模块分配到相应负责人，提高开发效率。

5、 各自模块预留测试单元，以便分模块测试。

6、 完成后独立测试，寻找漏洞，以便及时解决。