

浙江大学

程序设计专题

大程序报告



2019~2020 春夏学期 2020 年 6 月 5 日

报告撰写注意事项

- 1) 图文并茂。文字通顺，语言流畅，无错别字。
- 2) 书写格式规范，排版良好，内容完整。
- 3) 存在拼凑、剽窃等现象一律认定为抄袭；0分
- 4) 蓝色文字为说明，在最后提交的终稿版本，请删除这些文字。

目 录

1	大程序简介	4
1.1	背景及意义	4
1.2	目标要求	4
1.3	术语说明	4
2	功能需求分析.....	4
3	程序开发设计.....	6
3.1	总体架构设计	6
3.2	功能模块设计	6
3.3	数据结构设计	7
3.4	源代码文件组织设计	12
3.5	函数设计描述	115
4	部署运行和使用说明	25
4.1	编译安装	25
4.2	运行测试	25
4.3	使用操作	226
5	团队合作	38
5.1	分工（略）	
5.2	开发计划	38
5.3	编码规范	39
5.4	合作总结（匿名）	39
5.5	收获感言（匿名）	40
6	参考文献资料.....	41

本科生信息管理系统大程序设计

1 大程序简介

1.1 选题背景及意义

21 世纪以来，信息化为公司和学校的管理提高效率做出了弥足贡献。本科生信息管理系统，更是我们日常接触频繁的信息管理系统。对于学院、班级和同学个人档案的信息化管理，可以大幅度减少检索的耗时和降低保存的难度，从而提升效率。本科生信息管理系统从学院、课程、事务、班级等管理员层面进行查阅、添加、修改和删除信息，也可以让学生得以申请事务和查看各项信息反馈。

1.2 目标要求

- (1) 实现一个简易的本科生信息管理系统。
- (2) 基于 libgraphics 图形库来进行系统界面设计。
- (3) 通过链表来实现将不同的信息整合并且调用。
- (4) 可以从学生、班级和学院层面进行信息修改和查阅。
- (5) 能将相关数据保存在外部数据文件中。
- (6) 使用多文件来储存程序，并且对于密码部分进行加密保护。

1.3 术语说明

MD5 加密：MD5 信息摘要算法是一种被广泛使用的密码函数，可以通过产生散列值进行加密，并且保证信息的正确性。

2 功能需求分析

功能需求：

需要分为学生和管理员两个角色。

管理员管理着学院层面，班级层面、课程层面和事务层面。

学生则可以在学生页面查看信息。

学院层面：管理班级、课程和事项三个内容。班级层面可以创建班级、删除班级；课程层面可以添加课程、删除课程；事务层面可以添加事务、删除事务；高级的事项审批、条件查询和学院层面的汇总统计；

班级层面：可以加入学生、删除学生、初步审核学生提出的申请事项、汇总统计班级同学的信息；

课程层面：可以添加学生、录入成绩和汇总统计课程信息；

事务层面：可以在学院层面审核学生提出的申请事项并且汇总统计信息；

学生层面：可以登录学生界面、修改个人信息，提出事务申请和查看申请结果、查看成绩和修改个人密码；

帮助层面：关于本软件，使用方法，版权信息的内容。

菜单层面：可以保存或者放弃保存文件、返回首页、查看帮助。

数据结构：

将数据以学院为单位保存在 txt 文档里；

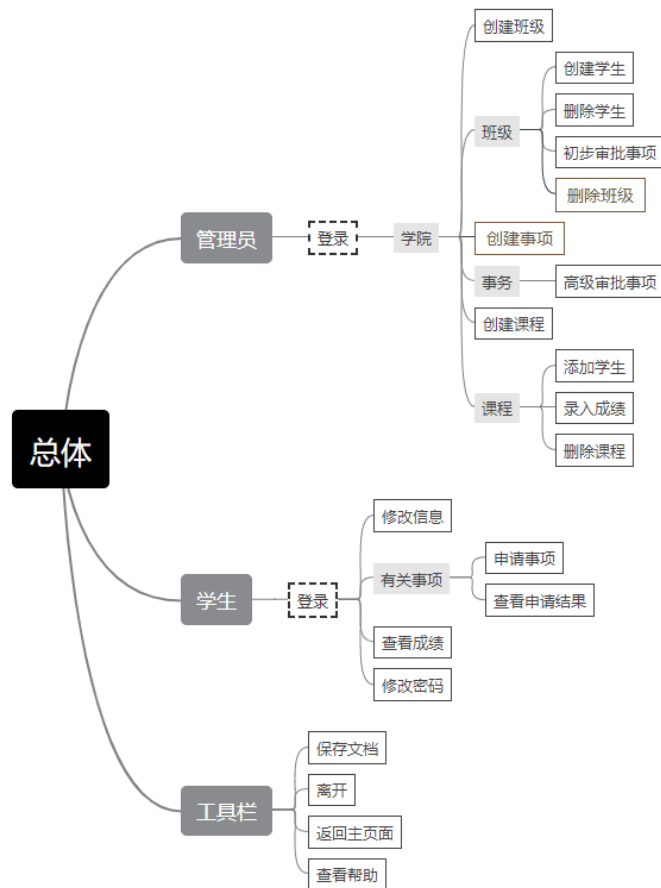
通过链表链接学院、班级、事务、课程和同学，并且储存信息。

性能方面：

可以用鼠标点击操控页面，可以用键盘输入。

3 程序开发设计

3.1 总体架构设计



3.2 功能模块设计

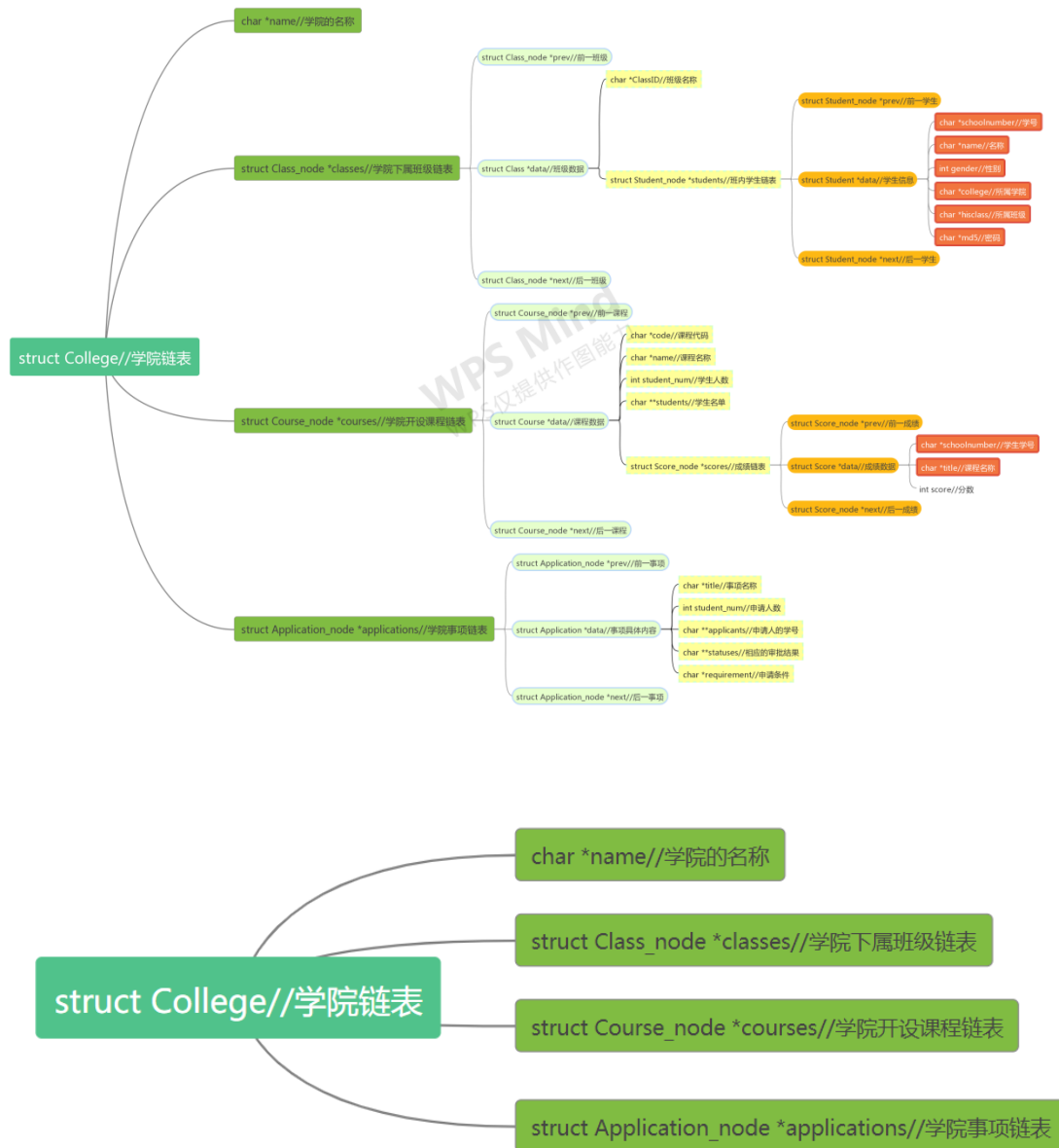
模块名称	功能简介
学院界面	查看班级信息
	添加班级
	进入班级管理界面
	查看课程信息
	添加课程
	进入课程管理界面
	查看事务信息
	添加事务

	进入事务管理界面
班级界面	查看同学信息（姓名和学号）
	创建学生
	删除学生
	查看同学的事务申请信息
	初步审批学生事务
	删除班级
	返回学院界面
课程界面	查看学生列表
	查看学生成绩
	添加学生
	录入学生成绩
	删除课程
	返回学院界面
事务界面	查看申请学生和结果
	高级审批事务
	返回学院界面
学生界面	进入学生信息修改界面
	进入学生事项管理界面
	查看总成绩
	进入修改密码界面
学生信息 修改界面	修改信息
	返回学生管理界面
学生事项 界面	查看事项和申请状态
	申请事项
	返回学生管理界面
修改密码 界面	修改密码
	返回学生管理界面
侧边栏界 面	保存/退出文档
	返回主界面
	查看帮助

3.3 数据结构设计

数据存储采用双向链表结构存储。

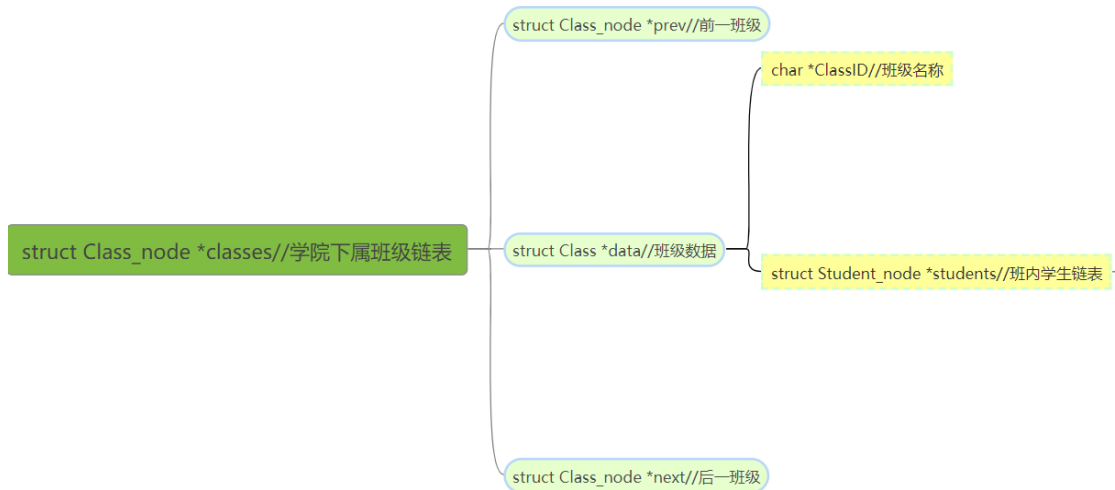
主体架构如下：



```

// College object
struct College {
    // Name of college
    char *name;
    // Linked list of classes in this college
    struct Class_node *classes;
    // Courses of this college
    struct Course_node *courses;
    // Applications of this college
    struct Application_node *applications;
};

```

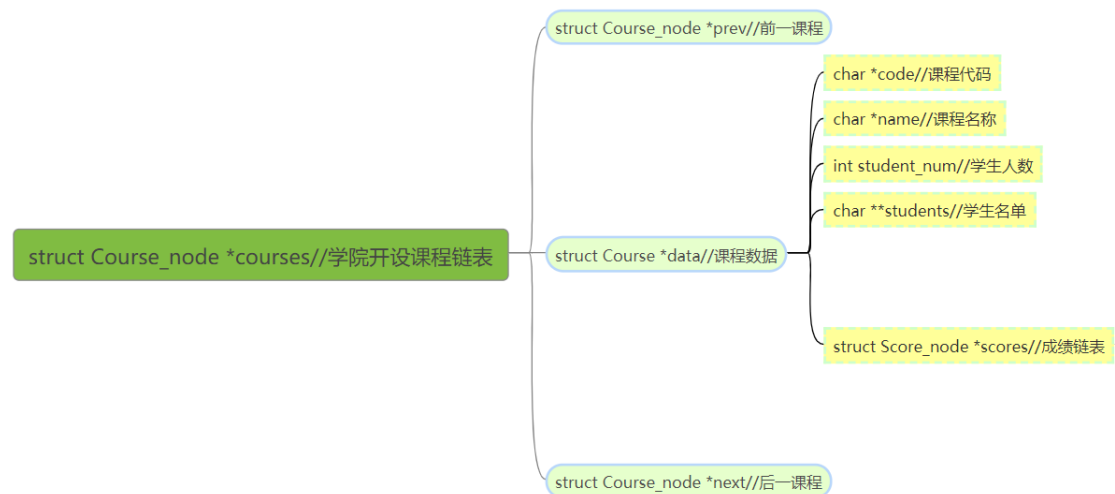



```

// Class Object
struct Class {
    // ID of this class
    char *ClassID;
    // Linked list of students in this class
    struct Student_node *students;
};
  
```

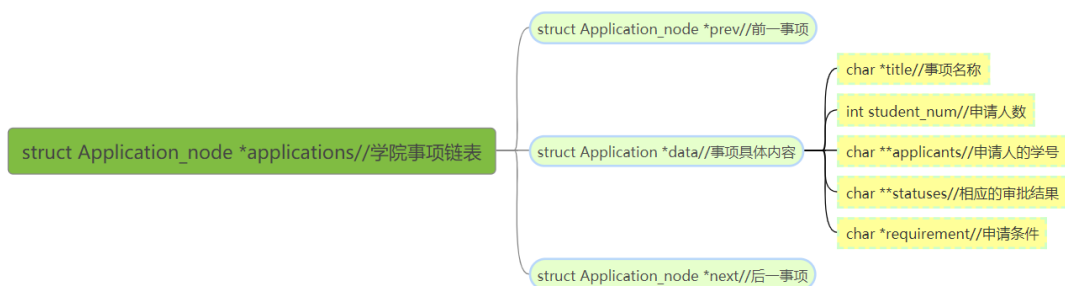
```

// List node of class object
struct Class_node {
    // Previous node in list
    struct Class_node *prev;
    // Pointer to Class Object
    struct Class *data;
    // Next Node in List
    struct Class_node *next;
};
  
```



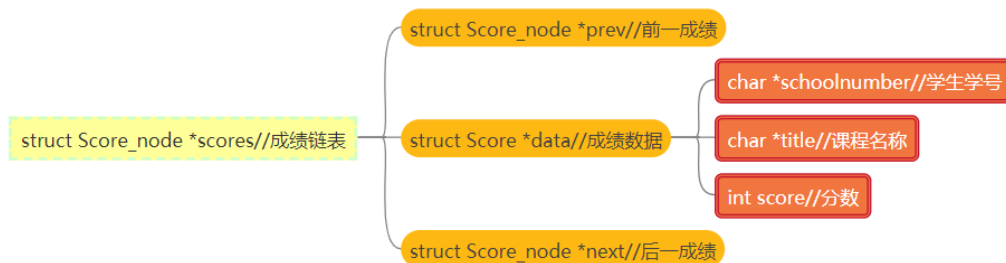
```
// Course Object
struct Course {
    // Course Code
    char *code;
    // Course Name
    char *name;
    // number of students
    int student_num;
    // students
    char **students;
    // scores
    struct Score_node *scores;
};

// List node of course object
struct Course_node {
    // Previous node of course in list
    struct Course_node *prev;
    // Pointer to course object
    struct Course *data;
    // Next node of course in list
    struct Course_node *next;
};
```



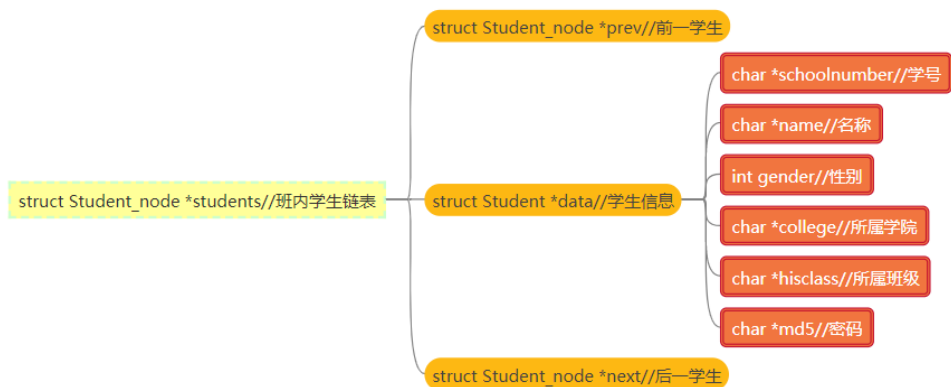
```
// Application Object
struct Application {
    // Title of this application
    char *title;
    // number of students who submit this application
    int student_num;
    // students who submit this application
    char **applicants;
    // statuses of applications
    char **statuses;
    // requierment of this application
    char *requirement;
};

struct Application_node {
    // Previous node in list
    struct Application_node *prev;
    // Pointer to Application
    struct Application *data;
    // Next node in list
    struct Application_node *next;
};
```



```
// Score Object
struct Score {
    // School number of the student this score belongs to
    char *schoolnumber;
    // Score title
    char *title;
    // Score
    int score;
};

// List node of score object
struct Score_node {
    // Previous node of score in list
    struct Score_node *prev;
    // Pointer to score object
    struct Score *data;
    // Next node of score in list
    struct Score_node *next;
};
```



3.4 源代码文件组织设计

<文件目录结构>

1) 文件函数结构

文件名称	clas.c			
包含函数	函数名	返回值	参数	功能
	clas_creat_stud	有	有	添加学生
	clas_delete_stud	有	有	删除学生

	clas_check_stud	有	有	查看是否属于自己班级的同学
	input_score	有	有	录入同学成绩
	clas_event_exam	有	有	班级层面审批同学事项
补充说明	无			

文件名称	clas.h
包含内容	函数声明
补充说明	无

文件名称	coll.c			
包含函数	函数名	返回值	参数	功能
	coll_creat_clas	有	有	创建班级
	coll_delete_clas	有	有	删除班级
	coll_event_creat	有	有	创建事项
	coll_event_delete	有	有	删除事项
	coll_event_exam	有	有	学院层面审批同学的事项
	coll_course_creat	有	有	创建课程
	coll_course_creat_stud	有	有	录入课程名单
	coll_course_delete	有	有	删除课程
补充说明	无			

文件名称	coll.h
包含内容	函数声明
补充说明	无

文件名称	data.c			
包含函数	函数名	返回值	参数	功能
	readData	有	有	读取文件
	getCollegeData	有	有	获得学院的链表数据
	getStudentBySchoolNumber	有	有	用学号找到学生的链表
	getClassByID	有	有	用班级编号找到班级链表
	getCourseByCode	有	有	用课程编号找到课程链表
	checkPassword	有	有	验证密码
	newCollege	有	有	新建学院
	getStringFromNewLine	有	有	从文件中读取字符串

	newCollege	无	有	新建学院
	exportCollegeData	有	有	文件中存储链表数据
补充说明	无			

文件名称	data.h			
包含内容	函数声明 链表结构定义			
补充说明	无			

文件名称	general.h			
包含内容	部分头文件			
补充说明	无			

文件名称	md5.c			
包含函数	函数名	返回值	参数	功能
	MD5Init	无	有	加密
	MD5Update	无	有	加密
	MD5Final	无	有	加密
	MD5Transform	无	有	加密
	MD5Encode	无	有	加密
	MD5Decode	无	有	加密
	md5check	有	有	加密
补充说明	无			

文件名称	md5.h			
包含内容	密码结构定义 函数声明			
补充说明	无			

文件名称	stud.c			
包含函数	函数名	返回值	参数	功能
	stud_input_information	有	有	录入信息
	stud_apply	有	有	申请事项
	stud_view_refults	有	有	查看申请结果
	stud_view_scores	有	有	查看单项成绩
	stud_change_password	有	有	修改密码
	stud_load	有	有	登录
	char **student_check_score(struct Course_node *upn, struct Student *sp)	有	有	查看汇总成绩

补充说明	无
------	---

文件名称	stud.h
包含内容	函数声明
补充说明	无

文件名称	total.c			
包含函数	函数名	返回值	参数	功能
	event_find	有	有	找到事项地址
	view_requ	有	有	查看事项要求
补充说明	无			

文件名称	total.h
包含内容	函数声明
补充说明	无

2) 多文件构成机制

data 文件中包含的是链表以及写入和读取 txt 文档中数据的过程，还有一些，在每一个具体功能文件（clas、coll、stud、total）中都包含了 data.h 的文件。

total 文件中包含一些功能性函数。coll、clas、stud 文件中分别存放了学院、班级和学生层面需要的函数。

在所有的 h 文件中都加入了 #ifndef.....#endif，防止重复编译。

将运行时不变的班级、课程、学生、事务等链表指针和用于标定计时器和页面跳转的变量定义为全局变量。

3.5 函数设计描述

函数原型	int clas_creat_stud (struct College *op, struct Class *cp, char *studentnumber, char *studentname, int studentgender)
功能描述	创建学生
参数描述	struct College *op: 所在学院 struct Class *cp: 所在班级 char *studentnumber: 学生学号 char *studentname: 学生姓名 int studentgender: 学生性别
返回值描述	Int: 标识是否成功添加学生
重要局部变量定义	struct Student_node *thestudents
重要局部变量用途描述	用来找到学生链表的末尾，添加指针

函数算法描述	申请一段内存并且将学生的数据分别存储进去，并且将这段衔接到学生链表的末端。
--------	---------------------------------------

函数原型	Int clas_delete_stud (struct Class * cp , char *studentnumber)
功能描述	删除学生
参数描述	struct Class * cp: 所在班级 char *studentnumber: 学生学号
返回值描述	Int: 标识是否成功删除学生
重要局部变量定义	struct Student_node *spn
重要局部变量用途描述	用于在学生链表中找到相应学生
函数算法描述	用指针在链表中找到该学生并删除节点。

函数原型	int clas_check_stud(struct Class *cp,char *stud_number)
功能描述	查看是否属于自己班级的同学
参数描述	struct Class * cp: 所在班级 char *studentnumber: 学生学号
返回值描述	Int: 标识是否属于本班学生
重要局部变量定义	struct Student_node *spn
重要局部变量用途描述	用于在学生链表中找到相应学生
函数算法描述	通过指针在学生链表中比较学号，找到相应学生，若找不到则说明非本班学员。

函数原型	int input_score(struct Score_node *rpn, char *schoolnumber,int thescore)
功能描述	输入成绩
参数描述	struct Class *cp: 所在班级 struct Score_node *rpn: 成绩链表 char *schoolnumber: 学生学号 int thescore: 学生成绩
返回值描述	Int: 标识是否成功录入成绩
重要局部变量定义	int thescore
重要局部变量用途描述	将字符串转化为数字
函数算法描述	将字符串转化为数字，然后录入相应位置

函数原型	int clas_event_exam(struct Application *ap,char *studentnumber,int thestatues)
------	--

功能描述	批准/拒绝同学
参数描述	struct Application *ap: 事项未知 char *studentnumber: 学生学号 int thestatues: 批准状态
返回值描述	Int: 标识是否成功录入结果
重要局部变量定义	无
重要局部变量用途描述	无
函数算法描述	比较申请人, 在相应位置录入结果

函数原型	int coll_creat_clas(struct College *op, char *classid)
功能描述	创建班级
参数描述	struct College *op: 学院 char *classid: 班级名称
返回值描述	Int: 标识函数是否成功运行
重要局部变量定义	struct Class_node *theclasses;
重要局部变量用途描述	找到班级函数末尾
函数算法描述	申请一段内存并且将班级的数据存储进去, 并且将这段衔接到班级链表的末端。

函数原型	int coll_delete_clas(struct College *op, char *classid)
功能描述	删除班级
参数描述	struct College *op: 学院 char *classid: 班级名称
返回值描述	Int: 标识函数是否成功运行
重要局部变量定义	struct Class_node *cpn;
重要局部变量用途描述	指向班级链表
函数算法描述	用指针在链表中找到该班级并删除节点。

函数原型	int coll_event_creat(struct College *op, char *thetitle, char *therequirement)
功能描述	通过班级编号找到班级链表的地址
参数描述	struct College *op: 学院 char *thetitle: 事件名称 char *therequirement: 要求
返回值描述	Int: 标识函数是否成功运行
重要局部变量定义	struct Application_node *theapplication;

重要局部变量 用途描述	指向事项链表的末尾
函数算法描述	申请一段内存并且将事项的数据存储进去，并且将这段衔接到事项链表的末端。

函数原型	int coll_event_delete(struct College *op, char *thetitle){
功能描述	删除事项
参数描述	struct College *op: 学院 char *thetitle: 标题
返回值描述	Int: 标识函数是否成功运行
重要局部变量 定义	struct Application_node *apn;
重要局部变量 用途描述	指向事项链表
函数算法描述	用指针在链表中找到该事项并删除节点。

函数原型	int coll_event_exam(struct Application *ap, char *studentnumber, int thestatues)
功能描述	初审同学申请
参数描述	struct Application *ap: 事项 char *studentnumber: 学号 int thestatues: 审核结果
返回值描述	Int: 标识函数是否成功运行
重要局部变量 定义	无
重要局部变量 用途描述	无
函数算法描述	找到该同学的地址，将审核结果输入相应位置

函数原型	int coll_course_creat(struct College *op, char *codes, char *names, int studentnumber)
功能描述	创建课程
参数描述	struct College *op: 学院 char *codes: 课程代码 char *names: 课程名称 int studentnumber: 学生人数
返回值描述	Int: 标识函数是否成功运行
重要局部变量 定义	struct Course_node *thecourses;
重要局部变量 用途描述	找到课程链表的末尾
函数算法描述	申请一段内存并且将课程的数据存储进去，并且将这段衔接到

	课程链表的末端。
--	----------

函数原型	int coll_course_creat_stud(struct Course *up, int i, char *studentnumber)
功能描述	录入课程名单
参数描述	struct Course *up: 课程 int i: 学生的序号 char *studentnumber: 学号
返回值描述	Int: 标识函数是否成功运行
重要局部变量定义	struct Score_node *thescore;
重要局部变量用途描述	找到成绩链表末尾
函数算法描述	申请一段内存并且将学生的数据存储进去，并且将这段衔接到成绩链表的末端。

函数原型	int coll_course_delete(struct College *op, char *codes)
功能描述	删除课程
参数描述	cstruct College *op: 学院 char *codes: 课程编号
返回值描述	Int: 标识函数是否成功运行
重要局部变量定义	struct Course_node *upn;
重要局部变量用途描述	指向课程链表
函数算法描述	找到相应的课程并删除该节点

函数原型	char *getStringFromNewLine(FILE *fp)
功能描述	从文件中读取字符串
参数描述	FILE *fp: 文件
返回值描述	char *: 返回读取的字符串
重要局部变量定义	无
重要局部变量用途描述	无
函数算法描述	申请一段内存，并从文件中读取字符串

函数原型	int readData(FILE *fp)
功能描述	从文件中读取链表
参数描述	FILE *fp: 文件
返回值描述	Int: 标识函数是否成功运行
重要局部变量定义	bufline

重要局部变量 用途描述	从文件中读取的字符串，用于写入链表
函数算法描述	从文件中读取字符串，判断类型并且写入链表

函数原型	void newCollege(char *name)
功能描述	建立新的学院
参数描述	char *name: 学院名称
返回值描述	无
重要局部变量 定义	无
重要局部变量 用途描述	无
函数算法描述	申请一段内存并且将学院的数据存储进去。

函数原型	int checkPassword(char *schoolnumber, char *password)
功能描述	char *schoolnumber: 学号 char *password: 密码
参数描述	验证密码是否正确
返回值描述	Int: 标识密码是否正确
重要局部变量 定义	struct Student *stu
重要局部变量 用途描述	找到相应学生链表
函数算法描述	找到相应学生链表，读取密码与输入比较。

函数原型	int exportCollegeData(FILE *fp)
功能描述	在文件中存储链表
参数描述	FILE *fp: 文件
返回值描述	Int: 标识函数是否成功运行
重要局部变量 定义	struct Class_node *curr_cla struct Student_node *curr_stu struct Course_node *curr_cou struct Application_node *curr_app
重要局部变量 用途描述	作为链表节点
函数算法描述	将链表中的内容转化为字符存入文件。

函数原型	struct Student *getStudentBySchoolNumber(char *schoolnumber)
功能描述	由学号得到学生链表位置
参数描述	char *schoolnumber: 学号
返回值描述	struct Student *: 该学生在链表中的地址
重要局部变量	struct Student_node *curr_stu

定义	
重要局部变量用途描述	指向学生链表
函数算法描述	在学生链表中逐个比较，找到与学号相符的一个为止，返回地址。

函数原型	struct Class *getClassByID(char *class_id)
功能描述	有班级编号得到班级链表的位置
参数描述	char *class_id: 班级编号
返回值描述	struct Class *: 该班级在链表中的地址
重要局部变量定义	struct Class_node *curr_cla
重要局部变量用途描述	指向班级链表
函数算法描述	在班级链表中逐个比较，找到与班级编号相符的一个为止，返回地址。

函数原型	struct Course *getCourseByCode(char *course_code)
功能描述	通过课程代码找到课程链表的位置
参数描述	char *course_code: 课程代码
返回值描述	struct Course *: 该课程在链表中的地址
重要局部变量定义	struct Course_node *curr_cou
重要局部变量用途描述	指向课程链表
函数算法描述	在课程链表中逐个比较，找到与课程代码相符的一个为止，返回地址。

函数原型	void MD5Init(MD5_CTX *context) void MD5Update(MD5_CTX *context, unsigned char *input, unsigned int inputlen) void MD5Final(MD5_CTX *context, unsigned char digest[16]) void MD5Encode(unsigned char *output, unsigned int *input, unsigned int len) void MD5Decode(unsigned int *output, unsigned char *input, unsigned int len) void MD5Transform(unsigned int state[4], unsigned char block[64])
功能描述	加密
参数描述	MD5_CTX *context: 密码加密过程
返回值描述	无
重要局部变量定义	无

重要局部变量 用途描述	无
函数算法描述	给密码加密

函数原型	int md5check(char *origin_string, char *md5_string)
功能描述	比较密码是否正确
参数描述	char *origin_string,: 原字符串 char *md5_string: 加密后的密码
返回值描述	Int: 标识密码是否正确
重要局部变量 定义	无
重要局部变量 用途描述	无
函数算法描述	通过将输入密码同样加密, 比较密码是否一致。

函数原型	int stud_input_information(struct Student *sp, char *thename, int thegender)
功能描述	修改自身信息
参数描述	struct Student *sp: 学生链表地址 char *thename: 学生姓名 int thegender: 学生性别
返回值描述	Int: 标识函数是否成功运行
重要局部变量 定义	无
重要局部变量 用途描述	无
函数算法描述	将修改后的信息填入链表

函数原型	int stud_apply(struct Student *sp, struct Application *ap)
功能描述	申请事项
参数描述	struct Student *sp: 学生链表地址 struct Application *ap: 事项
返回值描述	Int: 标识函数是否成功运行
重要局部变量 定义	无
重要局部变量 用途描述	无
函数算法描述	将学生信息添加到事项申请者的链表内。

函数原型	char *stud_view_refults(struct Student *sp, struct Application *ap)
功能描述	查看申请结果
参数描述	struct Student *sp: 学生链表地址

	struct Application *ap: 事项
返回值描述	char *: 返回申请结果的字符串
重要局部变量定义	无
重要局部变量用途描述	无
函数算法描述	找到相应的事项, 并且返回申请结果的字符串

函数原型	int stud_view_scores(struct Student *sp, char *codes)
功能描述	查看单项成绩
参数描述	struct Student *sp: 学生链表地址 char *codes: 课程代码
返回值描述	Int: 返回成绩
重要局部变量定义	struct Course *up struct Score_node
重要局部变量用途描述	指向课程和成绩链表
函数算法描述	找到相应的课程和成绩链表, 并且返回成绩

函数原型	char **student_check_score(struct Course_node *upn, struct Student *sp)
功能描述	查看汇总成绩
参数描述	struct Student *sp: 课程链表 struct Student *sp: 学生链表地址
返回值描述	char **: 返回记录科目和成绩字符串数组
重要局部变量定义	Int hisscore char *scor struct Score_node *rpn
重要局部变量用途描述	将成绩转化为字符串 指向成绩链表
函数算法描述	遍历课程链表, 找到该学生参与的, 输入课程名称和成绩。

函数原型	int stud_change_password(struct Student *sp, char *password1, char *password2)
功能描述	修改密码
参数描述	struct Student *sp: 学生链表地址 char *password1: 原密码 char *password2: 现密码
返回值描述	Int: 标识是否修改密码
重要局部变量定义	无
重要局部变量用途描述	无

函数算法描述	比较输入的原密码是否正确，如果正确则修改为现密码。
--------	---------------------------

函数原型	int stud_load(struct Student *sp, char *password)
功能描述	学生登录
参数描述	struct Student *sp: 学生链表地址 char *password: 密码
返回值描述	Int: 标识是否成功登陆
重要局部变量定义	无
重要局部变量用途描述	无
函数算法描述	引用函数判断密码是否正确，如正确则登录

函数原型	struct Application * event_find(struct College *op, char *thetitle)
功能描述	查找相应事项的地址
参数描述	struct College *op: 学院 char *thetitle: 事项名称
返回值描述	struct Application *: 相应事项的地址
重要局部变量定义	struct Application_node *apn;
重要局部变量用途描述	指向事项链表
函数算法描述	依次与事项链表的名称比较，并返回相应的事项地址

函数原型	char *view_requ(struct Application *ap) {
功能描述	查看事件要求
参数描述	struct Application *ap: 事项
返回值描述	char *: 要求的字符串
重要局部变量定义	无
重要局部变量用途描述	无
函数算法描述	引用函数得到相应事项的位置，并返回其中的要求字符串地址

函数原型	struct College *total(int college)
功能描述	打开文件
参数描述	int college: 表示相应学院
返回值描述	struct College *: 相应学院指针
重要局部变量定义	FILE *fp;
重要局部变量用途描述	打开文件

4.1 编译安装

如要编译工程,请使用 Dec-C++ 打开 cpr.j.dev 项目,然后选择编译运行即可。

1、反复添加班级、事务等内容，尝试输入长度上限，测试数据是否正确录入；

Gender must be "Male" or "Female"

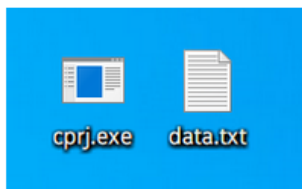
- 2、输入不符合规范的数据，查看反馈；
- 3、保存数据后再次打开，查看文件是否正确保存；

4.3 使用操作

详细可见《操作手册》。

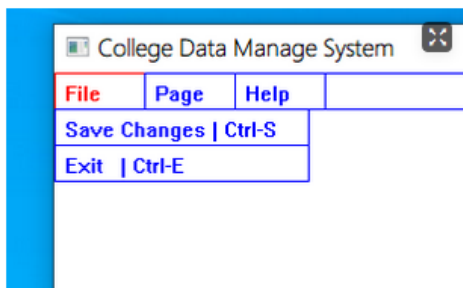
在程序执行的过程中，提示文字会出现在视窗左下角，提醒您目前的状态，例如输入错误等等。若运行过程中发生问题，请查看左下角的提示讯息。

开启文件



将本软件和我们特定格式的数据文件 data.txt 放置在同一个目录下，开启软件将会自动读取文件中的数据。

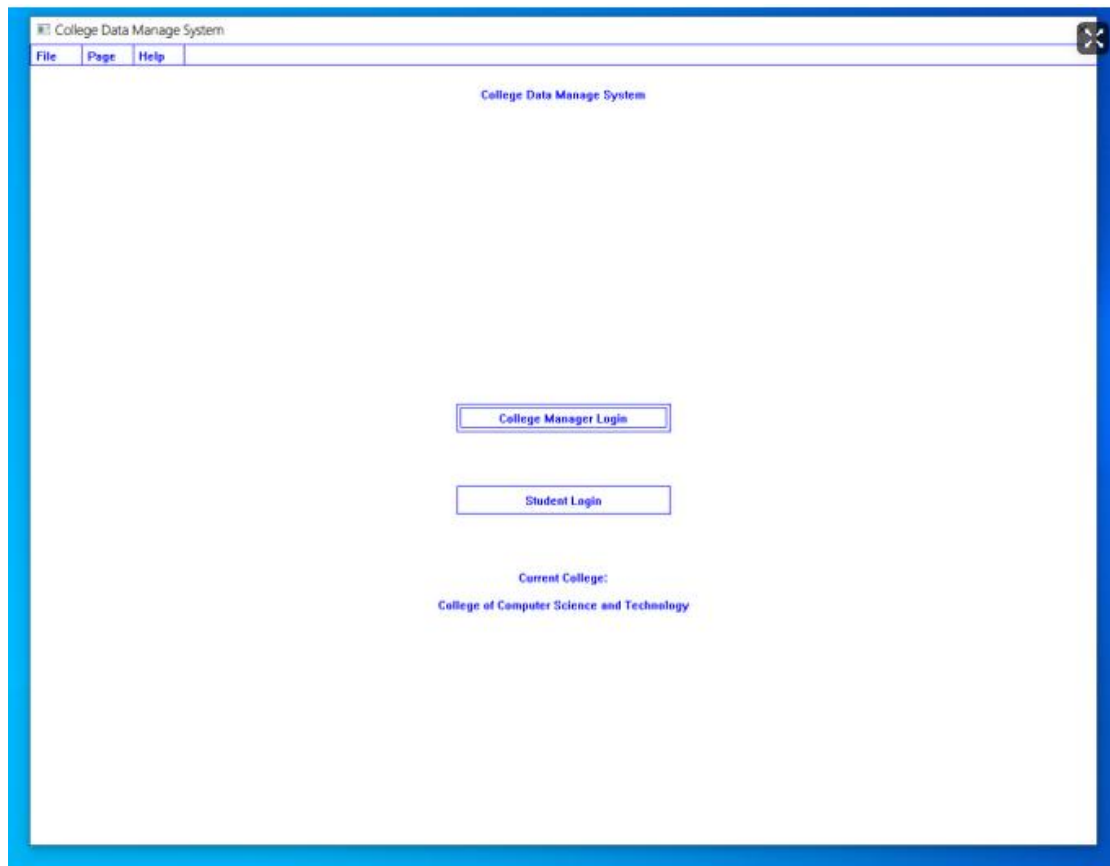
保存文件



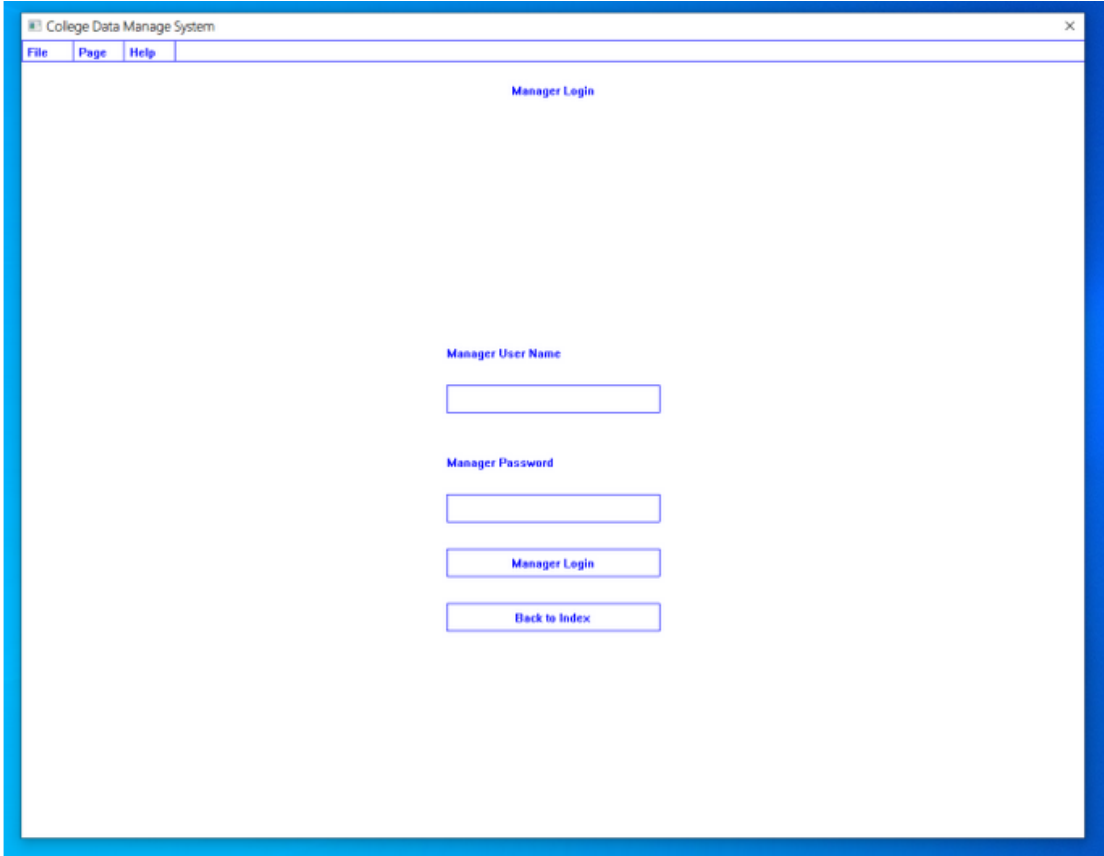
你可以随时使用菜单栏 File 内的 Save Changes 来讲当前的数据保存到文件中。没有保存的数据在程序关闭后都将遗失。

选择模式

本软件包含两种使用模式，「学生使用模式」以及「学院管理模式」，学生可以透过学生管理模式查看自己的信息，并进行操作。学院领导及老师可以透过学院管理模式对学院内的班级信息、课程信息以及申请事项进行管理。



在刚开启的页面中有两个按钮可以选择要用什么模式开启。点击上方的 College Manager Login 按钮进入学院管理模式；点击下方的 Student Login 按钮进入学生使用模式。



The screenshot shows a web browser window titled "College Data Manage System". The browser's menu bar includes "File", "Page", and "Help". The main content area is titled "Manager Login" and contains the following elements:

- A label "Manager User Name" followed by a text input field.
- A label "Manager Password" followed by a text input field.
- A button labeled "Manager Login".
- A button labeled "Back to Index".

选择学院管理模式后需要输入管理员账号及密码,系统预设的管理员账号密码为 admin admin, 输入后点击 Manager Login 就可以以管理员身份进入系统。

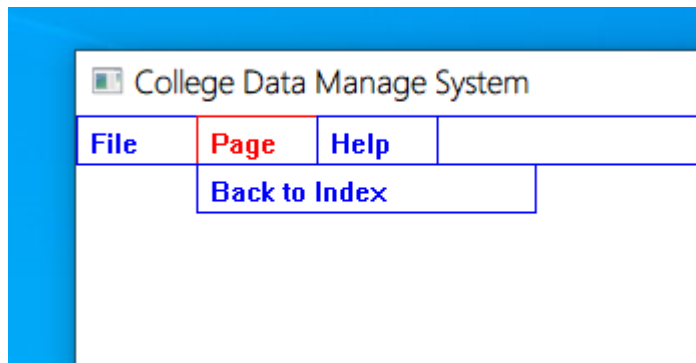
The screenshot shows a web browser window titled "College Data Manage System". The browser's menu bar includes "File", "Page", and "Help". The main content area is titled "Student Login" and contains the following elements:

- A label "School Number" followed by a text input field.
- A label "Password" followed by a text input field.
- A "Student Login" button.
- A "Back to Index" button.

学生使用模式需要输入学号以及密码进行登入，预设的学生及密码配对如下：

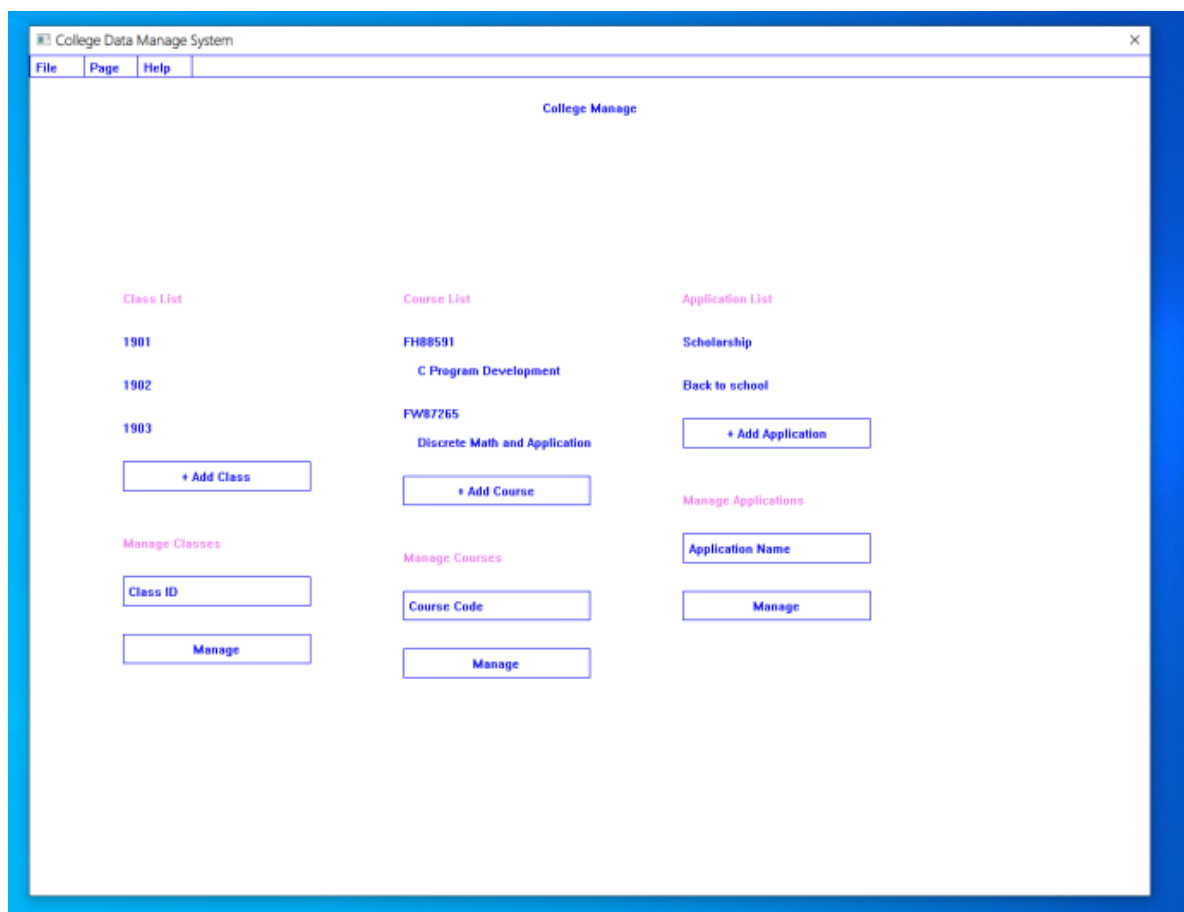
1. 学号 3190000001 密码 1
2. 学号 3190000002 密码 password
3. 学号 3190000003 密码 password
4. 学号 3190000004 密码 password
5. 学号 3190000005 密码 password
6. 学号 3190000006 密码 password
7. 学号 3190000007 密码 password
8. 学号 3190000008 密码 password

若有自行添加过学生数据，或修改过密码，则请输入新的学号及密码即可登入。



你可以随时透过菜单栏 Page 的 Back to Index 回到模式选择页面。

学院管理



进入学院管理模式后会看到以上的画面，整体由三个区块构成，分别是左边的班级（class）区块、中间的课程（course）区块、右边的申请事项（application）区块。

在这个画面，你可以透过点击按钮来添加新的班级、课程或是发布一个新的申请事项供学生申请。也可以透过将班级号码、课程代码或申请事项名称输入到区块下方的输入框，点击按钮来进入对应的信息设置。

班级管理

College Data Manage System

File Page Help

Manage Class

1901

Student List

Student Number	Name
3190000001	Amy Wang
3190000002	Jack Chen
3190000003	Ken Lin

Add new student into class

Student Number

Student Name

Student's Gender

Add Student

Delete a student in class

Student Number

Delete Student

Application List

Scholarship

GPA higher than 4.0

3190000001	Rejected
3190000003	Accepted
3190000004	Rejected
3190000007	Rejected

Back to school

Have done the Nucleic Acid Testing of Covid-19

3190000002	Accepted
3190000003	Accepted
3190000005	Rejected
3190000008	Accepted

Approval Items

Application Title

Student Number

Approval

Reject

Delete Class

Delete the class

Return to College Manage

Return

班级管理页面由四个区块构成，从左边到右边分别是学生列表区块、学生操作区块、申请事项区块以及其他操作区块。

学生列表区块

学生列表区块有目前选择班级的学生列表，显示了学生的学号以及姓名。

学生操作区块

学生操作区块的上方可以新建新的学生数据到目前选择的班级，你需要先输入新学生的学号（Student Number）、姓名（Student Name）以及性别（Student's Gender）才能加入新学生。其中性别必须输入 Male 或是 Female。

学生操作区块的下方则可用来移除学生数据，输入要移除的学生的学号即可移除。

申请事项区块

申请事项区块提供了班级层级的申请事项审批，申请事项必须先经过班级审批许可后才可以进入学院层级审批，在列表中会先显示申请事项的名称、需求以及报名的学生学号列表，列表右方为该学生这一项申请事项的状态。

经过班级审批的申请事项有以下两个状态：

1. class_passed 班级审核通过

2. class_rejected 班级驳回了申请

其他操作区块

其他操作区块可供进行审批事项的操作，只要输入当前审批的事项名称 (Application Title) 和被审批的学生学号后，点击 Approval 按钮通过该申请；点击 reject 按钮驳回该申请。

其他操作区块也可供删除当前所选的班级，该操作将直接删除班级以及班级内的所有学生数据。

回到学院管理页面请点击其他操作区块最下方的返回按钮。

课程管理

课程管理页面由四个区块组成，从左边到右边分别是课程参与者列表区块、成绩记录列表区块、数据操作区块以及其他操作区块。

课程参与者列表区块

课程参与者列表区块显示了所有有上该堂课程的学生学号。

成绩记录列表区块

成绩记录列表区块显示了该课程目前所有的成绩记录，每笔记录以以下格式显示：

3190000001 got 90 points in Quiz 1

1. 3190000001 的位置代表了该成绩数据的拥有者学号
2. 90 的位置代表了该成绩记录的得分
3. Quiz 1 的位置代表了该成绩记录的事项

数据操作区块

数据操作区块上方可以输入学生的学号，将制定学生加入到这门课程中。

下方是成绩操作区块，可以输入学号、事件标题以及成绩来进行课程成绩的录入。透过输入现有记录的学号和事件标题，可以覆盖原有的数据实现成绩的更新。

其他操作区块

其他操作区块包含删除课程以及返回上个页面。

删除课程将一并删除课程中所有成绩数据。

申请事项审批

The screenshot displays a web application window titled "College Data Manage System". The interface is divided into several sections:

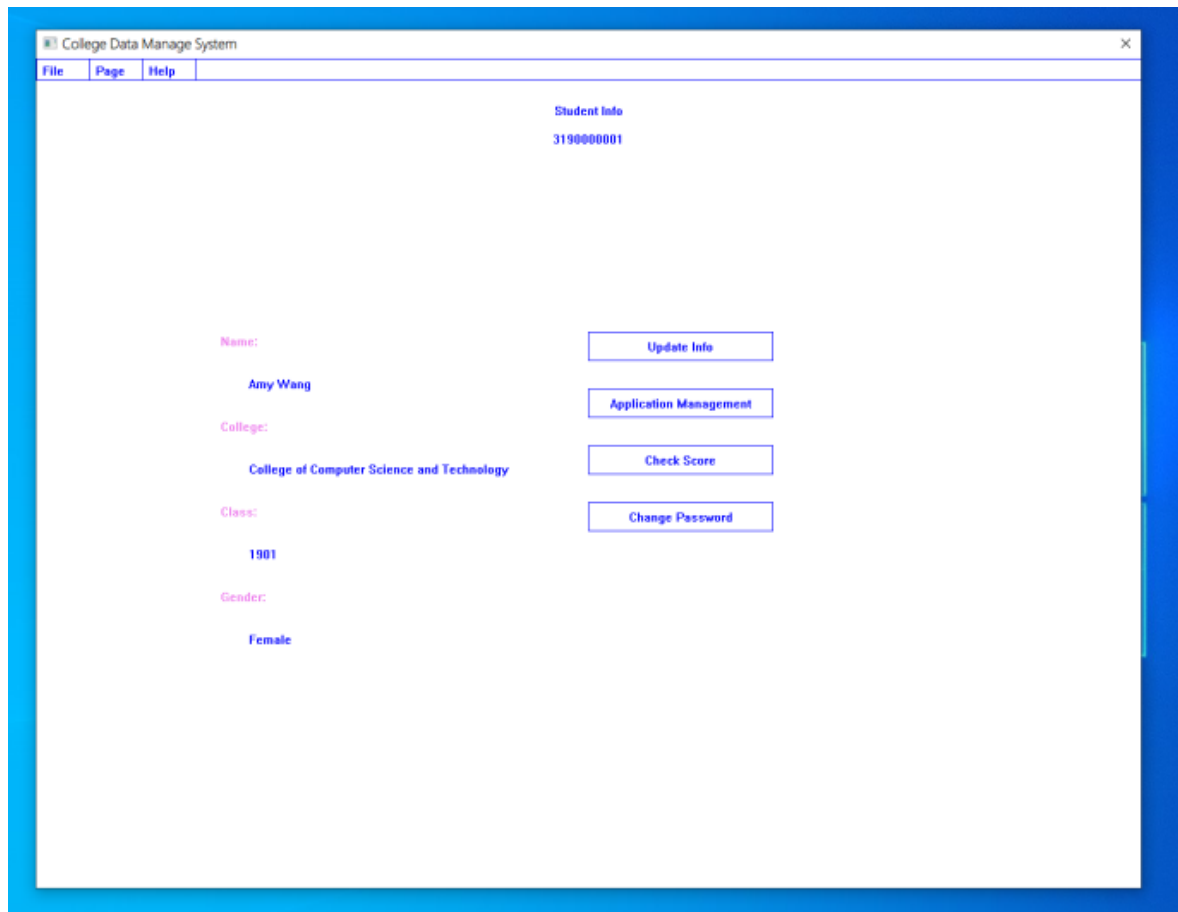
- Manage Application**: The main heading for the section.
- Scholarship**: A sub-heading indicating the type of application being managed.
- Applicants List**: A table listing applicants with their IDs and current status.

Applicant ID	Status
3190000001	Rejected
3190000003	Accepted
3190000004	Rejected
3190000007	Rejected
- Change status of application**: A section for updating the status of an application. It includes a text input field labeled "School Number" and two buttons: "Accept" and "Reject".
- Return to college manage**: A section with a "Return" button to navigate back to the main college management page.

本页面可以实现学院层级的申请事项审批，学生的申请事项状态必须是 class-passed，也就是班级审批通过，才可以进行学院层级的申请事项审批。

左方是当前选择的申请事项的申请者学号列表，下方有每位申请者的当前状态。
右方是操作区块，输入申请者的学号就可以选择要通过审批或是驳回申请。

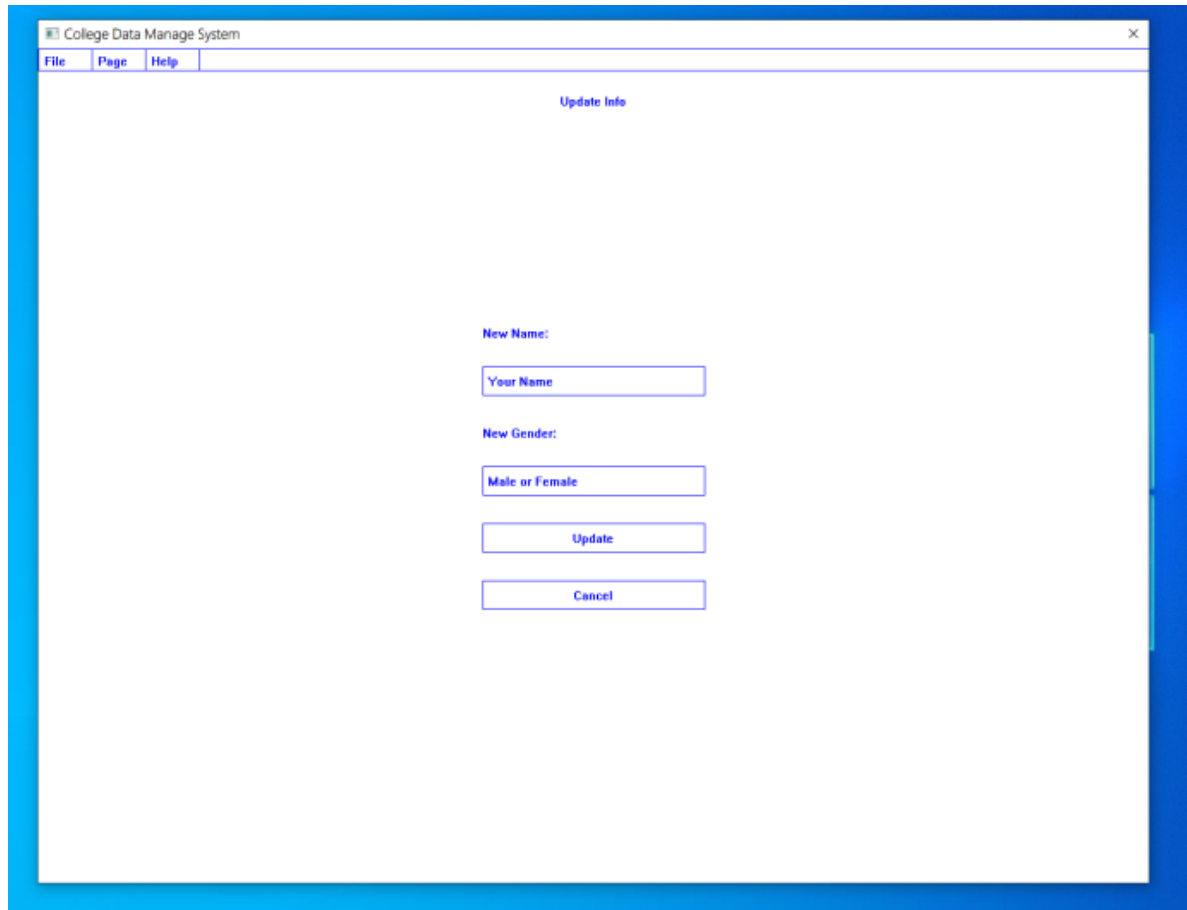
学生使用模式



进入学生使用模式后，你可以在左方看到当前登入的学生信息，也可以从右方选择需要的操作。

1. Update Info 更新学生数据
2. Application Management 管理我的申请事项
3. Check Score 查看我的成绩记录
4. Change Password 更换密码

更新学生数据



The screenshot shows a window titled "College Data Manage System" with a menu bar containing "File", "Page", and "Help". The main content area is titled "Update Info" and contains the following form elements:

- A label "New Name:" followed by a text input field labeled "Your Name".
- A label "New Gender:" followed by a text input field labeled "Male or Female".
- An "Update" button.
- A "Cancel" button.

这个页面可以更改当前学生的姓名以及性别，性别需要输入 Male 或是 Female，按下 Update 按钮来套用变更，或是按下 Cancel 回到上一页。

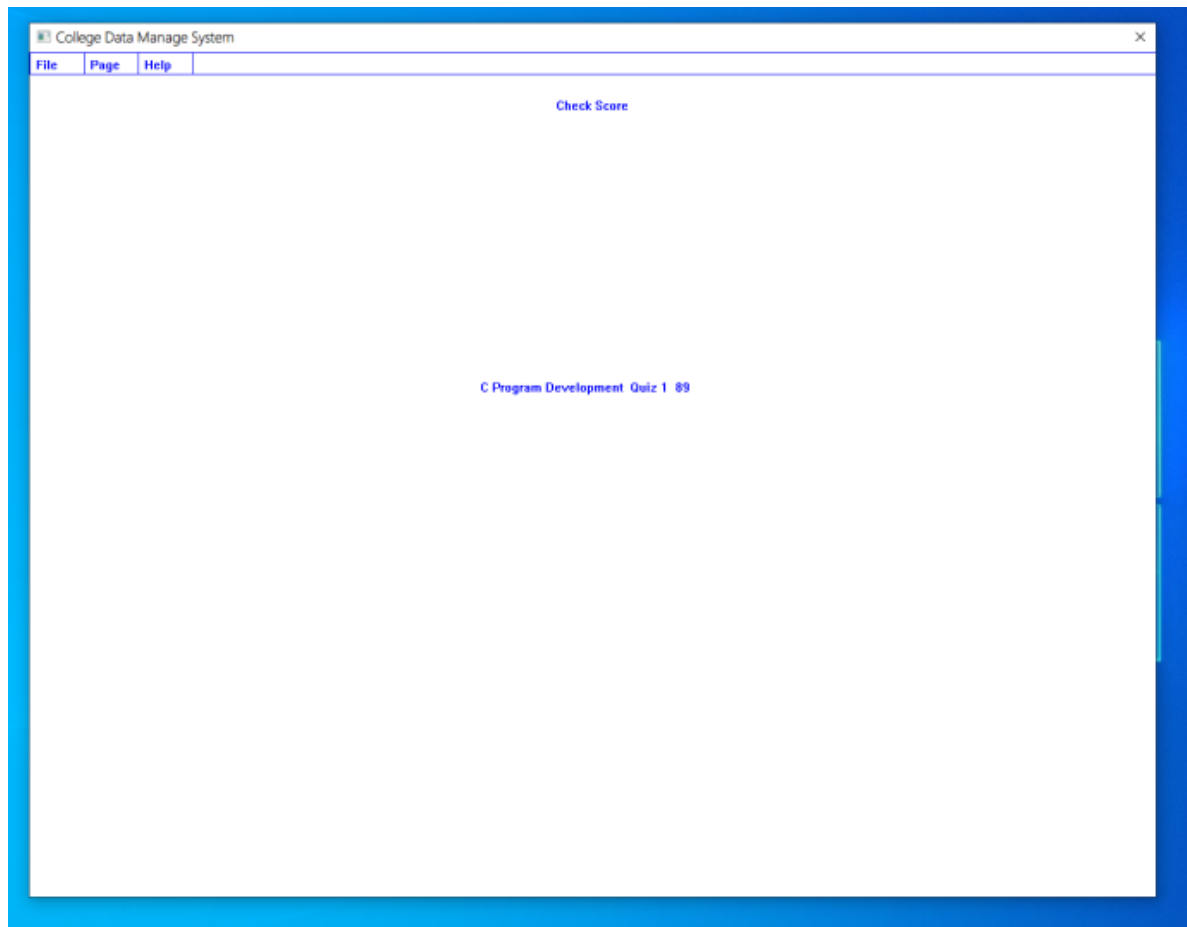
管理我的申请事项

The screenshot shows a web application window titled "College Data Manage System". It has a menu bar with "File", "Page", and "Help". The main content area is titled "Application Management". On the left, under "Applications of College", there is a "Scholarship" section with the criteria "GPA higher than 4.0", a "Rejected" status, and a "Back to school" button. Below this, it says "Have done the Nucleic Acid Testing of Covid-19" and "You has not submit this application.". On the right, under "Submit / Cancel Applications", there is an input field for "Application Name", a "Submit / Cancel" button, and a "Back to Student Info" button.

这个页面可以看到当前登入学生所提交的申请事项，以及当前状态。

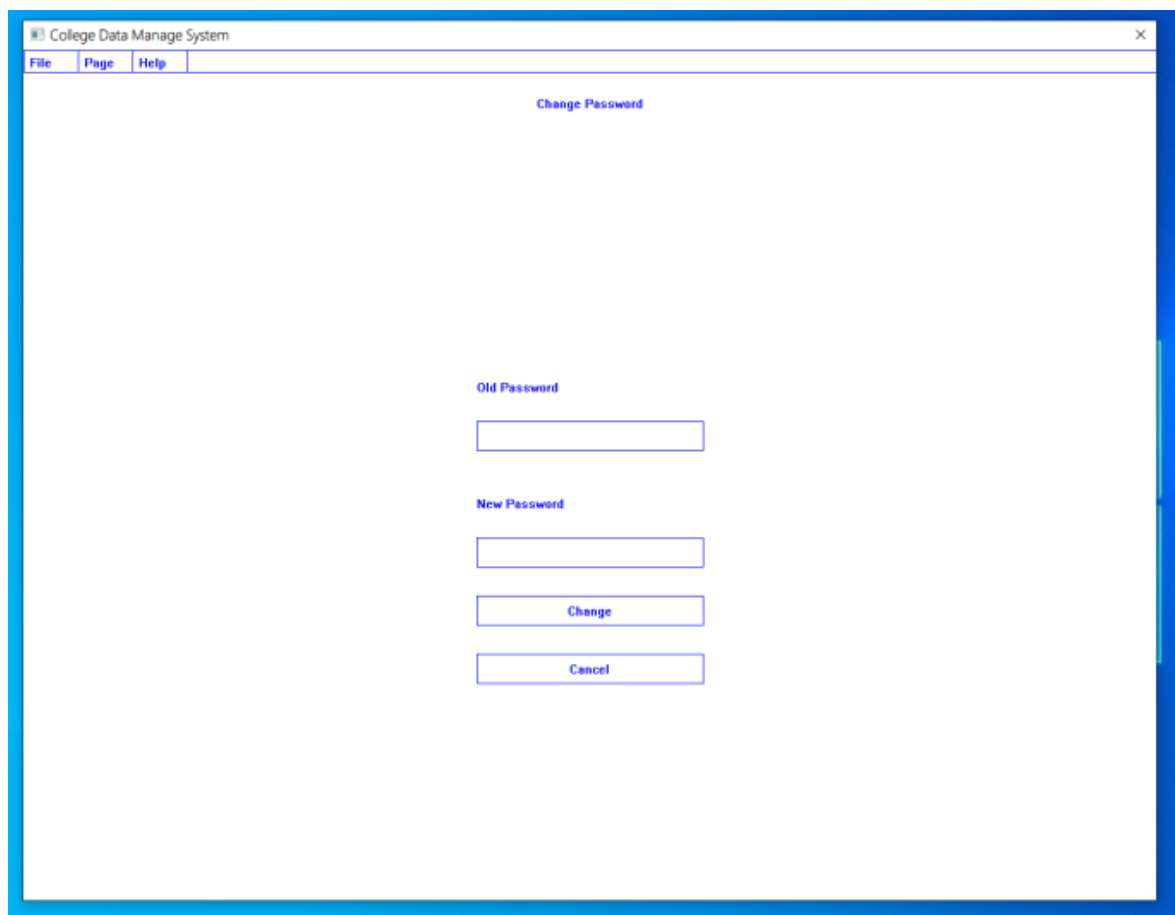
在右方输入框输入申请事项的名称，选择 Submit / Canel 按钮可以自动提交或是收回当前的申请。

查看成绩记录



这个页面显示了该学生在各个课程获得的成绩。

更换密码



在这个页面可以更改当前登入用户的密码，需要输入当前密码进行安全验证。本系统的密码使用 md5 加密算法进行加密，及时数据文件遭窃您的密码也不会被他人得知。

5 团队合作

5.1 任务分工（互评版略）

A：负责制作链表和与文件的读入输出等功能，并且写了部分功能函数。后期因 C 同学关系负责了界面绘制与交互，进行 BUG 测试与修复。撰写操作手册。

B：负责根据链表制作需要的函数完成功能，后期因 C 同学关系负责绘制部分界面，进行 BUG 测试与修复。撰写报告其余部分。

5.2 开发计划

先透过读取文件来在程序中保存数据+透过数据保存新的文件，并制作需要的函数接口；

从第一部分的函数接口获得数据，然后实现程序需要的各种功能；
设计界面的设计和交互行为的处理
调用函数接口来获取需要渲染的数据然后渲染在视窗上。
测试数据，修改 BUG。

5.3 编码规范

- 1、工程通过编译，已经消除相应 error；
- 2、变量、函数的命名方式符合编码的规范，并且有意义；
- 3、有对接口函数的功能、使用方法、参数定义进行注释和说明；
- 4、全局变量在模块的初始化函数中初始化，未在定义时赋初值；
- 5、函数指针变量在使用前检查是否有效；
- 6、遍历数组时，使用 sizeof 来指定遍历的长度；
- 7、在声明指针或参数时，符号 “*” 紧靠变量名。

5.4 合作总结（注意匿名）

〈说明：对合作程序和开发过程进行总结，包括开发亮点、挑战点和应用知识点总结；总结合作过程中的备忘录信息，如小组会议记录等体现良好的沟通能力，如邮件关键内容屏幕截图〉

前期由于时间安排出现问题，并没有进行充分的讨论就开始制作，A 同学直接开始进行链表模块撰写，随后 B 同学依照链表进行函数制作。在制作过程中，每天都有个人的成果交流，并且将如果存在的问题提出，两人讨论并且解答。在此期间，开发时间一直是一个挑战。由于缺乏领导和组织，我们组的工作进度被大幅度拖后，导致了所有任务都需要在短时间内完成。尤其是 C 同学画出部分界面后的接口问题，由于我们与 C 的无法正常交流，导致了组内的矛盾冲突，以至于在 48 小时之内从零开发了一个新的 UI。故而，我认为在程序设计中最大的挑战有两个：时间和交流。

在技术方面，遇到的难点并不太多，大多数 BUG 在发现的时候可以较快地得到解决。有关链表的新建和链接问题一度成为问题，但是在查阅书籍和请教他人后得以解决。同时，因为某个不符合规范的函数，进度一度停滞。

在对程序要求的研究中，我们新增了一些相关的功能，用加密提升了信息的安全性，这属于本系统的亮点。

4 月 26 日，B 同学发起群聊，在讨论中推选 C 作为组长；

5 月 5 日，A 同学提出确定题目，次日三人经过讨论定下选题；

5 月 10 日，A 同学重新提出分工安排，次日确定分工；

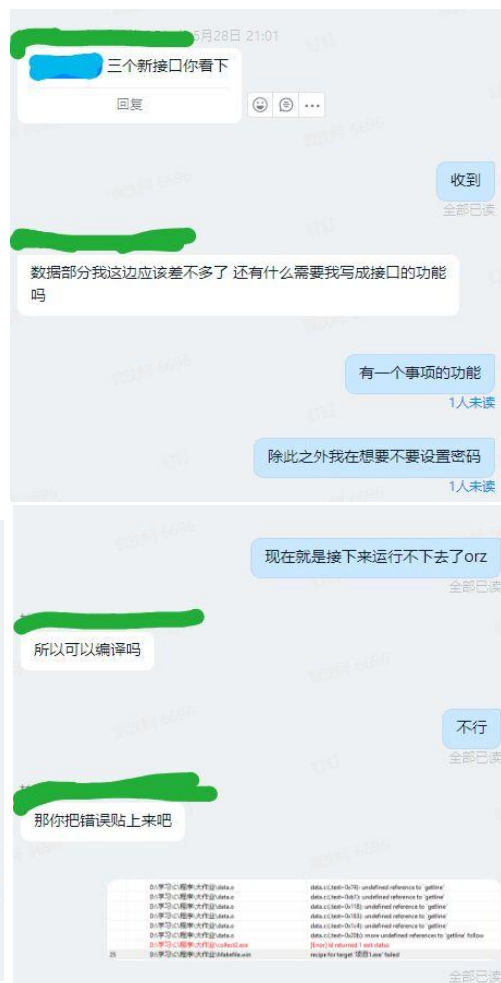
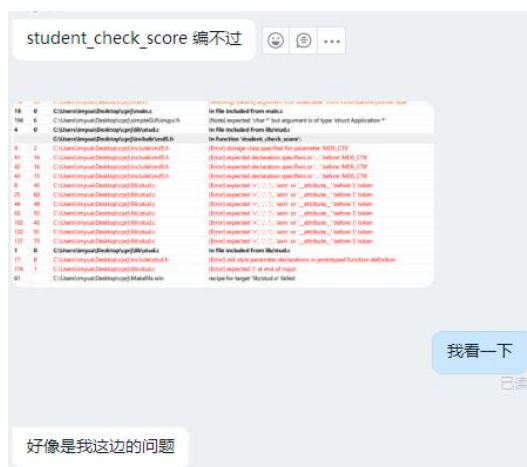
5 月 24 日，AB 开始工作，并与当晚给出初步的链表；

5月25-31日，讨论完成链表与函数部分；

6月2日，组长C出现，给出功能和框架，在对接中出现问题，无法实现功能。

6月3日晚，A同学开始写图形界面；

6月4日，信息系统制作完毕。



5.5 收获感言（注意匿名）

对我来说，本次的大作业是一个不可多得的珍贵经历。在此期间我们综合运用一年之内学到的C语言知识来编制一个简单的信息系统。最开始我有些无从下手，然后在与同学的交流中逐渐理清思路，将原本的大问题拆分成能够运用知识解决的小问题，最终得到了成果。但是同时，大作业给了我很多教训。我们最早时候没有一个合理的时间规划，同时也因此没有进行充分的交换意见，最终导致了与组员的沟通出现问题，甚至引发了争吵并且导致小组分崩离析。除此之外，我们作业的制作顺序还可以优化，避免时间的浪费。在此次大作业的过程中，我认为我的行为中规中矩，根据之前同学的链表完成了相应的函数，制作接口，并且在同学有需要的时候积极修改和帮助。在其中我熟悉了对于链表的应用和多文件等内容，对于我个人的C语言学习来说弥足珍贵。

C语言在我过去的认识中一直都是一个只能用来做作业或是算法比赛的程序语言，毕竟从一开始的基础语法到现在我们学习了指针、链表等等的写法以后，对于如何用C语言写出一个真正拥有生产价值的应用程序或服务后段我仍然

毫无头绪,但透过这次的专题作业,我成功的将过去用其他语言进行开发的经验、逻辑实作进来,让 C 语言对我而言有了新的意义与价值。

我负责了程序的数据结构以及页面渲染的部分。在数据结构的开发过程中第一个遇到的难关就是必须用原始的方式进行数据的读取,相较于 json 这类的物件数据保存格式,透过 `fgets()` 函数一行一行读取来建立整套的数据是相当具有挑战性的。页面渲染的部分也是相当有趣的部分,透过 `libgraphics` 图形库我第一次用 C 语言写出了图形程序,并且可以实际上进行交互、串接接口函数等等。

6 参考文献资料

<https://blog.csdn.net/zhaobinbin2015/article/details/81007737>

https://blog.csdn.net/xiaowang1379214245/article/details/80365996?utm_medium=distribute.pc_relevant.none-task-blog-BlogCommendFromBaidu-3&depth_1-utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromBaidu-3

https://blog.csdn.net/xiaowang1379214245/article/details/80365996?utm_medium=distribute.pc_relevant.none-task-blog-BlogCommendFromBaidu-3&depth_1-utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromBaidu-3

<http://witmax.cn/c-md5-code.html>

C 程序设计语言, Brian.W.Kernighan 著, 徐宝文 译。