

it studio

学生成绩管理系统 详细设计说明

it studio
2013/1/21

目 录

1.	班级管理程序	1
1.1.	数据文件结构说明.....	1
1.2.	数据结构说明	1
1.3.	源文件说明	2
1.3.1.	ban_data.c 源文件	2
1.3.2.	ban_process.c 源文件	3
1.3.3.	ban_data.h 头文件	3
1.3.4.	ban_process.h 头文件	3
1.4.	数据处理函数	3
1.4.1.	int read_ban(struct banji_list *list).....	3
1.4.2.	int write_ban(struct banji_list *list).....	3
1.4.3.	int sort_ban_bh(struct banji_list *list).....	4
1.4.4.	int sort_ban_zy(struct banji_list *list);	4
1.4.5.	int sort_ban_nj(struct banji_list *list).....	4
1.4.6.	int disp_ban_list(struct banji_list *list)	4
1.4.7.	struct __banji * get_ban_by_bh(struct banji_list *list,char *bianhao)	4
1.4.8.	void disp_ban(struct __banji * p)	5
1.4.9.	struct banji_list * add_ban_to_list(struct banji_list *list,struct __banji *banji)	5
1.4.10.	struct __banji * getout_ban_by_bh(struct banji_list *list,char *bianhao).....	5
1.4.11.	int empty_ban_list(struct banji_list *list)	5
1.5.	业务处理函数	6
1.5.1.	void __disp_all_ban();.....	6
1.5.2.	void __disp_ban_bh(char ban_bh[]);.....	6
1.5.3.	void __zengjia_ban();.....	6
1.5.4.	void __shanchu_ban_bh(char ban_bh[]);	6
1.5.5.	void __paixu_ban_bh();	6
1.5.6.	void __paixu_ban_zy();	6
1.5.7.	void __paixu_ban_nj();	7
2.	学生管理程序	8

2.1.	文件结构说明	8
2.2.	数据结构说明	8
2.3.	源文件说明	9
2.3.1.	xuesheng_data.c	9
2.3.2.	xuesheng_process.c	10
2.3.3.	xuesheng_data.h	10
2.3.4.	xuesheng_process.h	10
2.4.	数据处理函数	10
2.4.1.	int read_xuesheng(struct xuesheng_list *list,char bianhao[])	10
2.4.2.	int write_xuesheng(struct xuesheng_list *list,char ban_bh[])	11
2.4.3.	int sort_xuesheng_bh(struct xuesheng_list *list)	11
2.4.4.	int sort_xuesheng_zy(struct xuesheng_list *list)	11
2.4.5.	int disp_xuesheng_list(struct xuesheng_list *list)	11
2.4.6.	struct __xuesheng * get_xuesheng_by_bh(struct xuesheng_list *list,char *xuehao)	12
2.4.7.	struct __xuesheng * getout_xuesheng_by_bh(struct xuesheng_list *list,char *xuehao)	12
2.4.8.	void disp_xuesheng(struct __xuesheng * p)	12
2.4.9.	struct xuesheng_list * add_xuesheng_to_list(struct xuesheng_list *list,struct __xuesheng *xuesheng)	12
2.4.10.	int empty_xuesheng_list(struct xuesheng_list *list)	13
2.4.11.	char *get_xs_filename(char *filename,char *ban_bh)	13
2.4.12.	disp_xs(char ban_bh[],char xs_bh[])	13
2.5.	业务处理函数	13
2.5.1.	void __disp_xs_ban_bh(char bianhao[])	13
2.5.2.	void __zengjia_ban_xs(char ban_bh[])	13
2.5.3.	void __shanchu_ban_xs(char ban_bh[],char xs_bh[])	14
2.5.4.	void __xiugai_ban_xs(char ban_bh[],char xs_bh[])	14
3.	成绩管理程序	14
3.1.	数据文件结构说明	14
3.2.	数据结构说明	14
3.3.	源文件说明	16

3.3.1.	chengji_data.c.....	16
3.3.2.	chengji_process.c.....	16
3.3.3.	cengji_data.h	16
3.3.4.	cengji_process.h	17
3.4.	数据处理函数	17
3.4.1.	int read_chengji(struct chengji_list *list,char ban_bh[],char ke_bh[]).....	17
3.4.2.	int write_chengji(struct chengji_list *list,char ban_bh[],char ke_bh[]).....	17
3.4.3.	int disp_chengji_list(struct chengji_list *list)	17
3.4.4.	struct __chengji * get_chengji_by_xh(struct chengji_list *list,char *xuehao).....	18
3.4.5.	void disp_chengji_list(struct chengji_list *list chengji_list *list)	18
3.4.6.	struct chengji_list * add_chengji_to_list(struct chengji_list *list,struct __chengji *chengji)	18
3.4.7.	int empty_chengji_list(struct chengji_list *list).....	18
3.5.	业务处理函数	18
3.5.1.	void __disp_ban_ke_cj(char ban_bh[],char ke_bh[])	18
3.5.2.	void __zengjia_ban_ke_cj(char ban_bh[],char ke_bh[]).....	19
3.5.3.	void __xiugai_ban_ke_cj(char ban_bh[],char ke_bh[],char xs_bh[])	19
3.5.4.	void __shanchu_ban_ke_cj(char ban_bh[],char ke_bh[],char xs_bh[]).....	19
3.5.5.	void __disp_ban_xs_cj(char ban_bh[],char xs_bh[])	19
4.	课程管理系统	21
4.1.	文件结构说明	21
4.2.	数据结构说明	21
4.3.	源文件说明	22
4.3.1.	kecheng_data.c 源文件	22
4.3.2.	kecheng_process.c 源文件	23
4.3.3.	kecheng_data.h 头文件	23
4.3.4.	kecheng_process.h 头文件	23
4.4.	数据处理函数	24
4.4.1.	int read_ke(struct kecheng_list *list);.....	24
4.4.2.	int write_ke(struct kecheng_list *list);	24

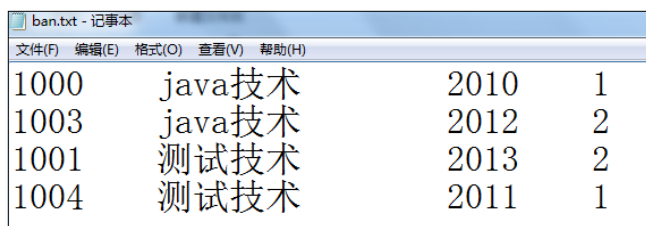
4.4.3.	int sort_ke_bh(struct kecheng_list *list);.....	24
4.4.4.	int sort_ke_xuefen(struct kecheng_list *list);	24
4.4.5.	int sort_ke_mingcheng(struct kecheng_list *list);.....	24
4.4.6.	int disp_ke_list(struct kecheng_list *list);	25
4.4.7.	struct __kecheng * get_ke_by_bh(struct kecheng_list *list,char *bianhao);	25
4.4.8.	void disp_ke(struct __kecheng * p);	25
4.4.9.	struct kecheng_list * add_ke_to_list(struct kecheng_list *list,str__kecheng*kecheng); 25	
4.4.10.	int empty_ke_list(struct kecheng_list *list);	25
4.4.11.	struct __kecheng * getout_ke_by_bh(struct kecheng_list *list,char *bianhao).....	26
4.5.	业务处理函数	26
4.5.1.	void __disp_all_ke();	26
4.5.2.	void __disp_ke_bh();.....	26
4.5.3.	void __shanchu_ke_bh(char ke_bh[]);	26
4.5.4.	void __zengjia_ke();	26
4.5.5.	void __paixu_ke_bh();.....	26
4.5.6.	void __paixu_ke_mc();	27
4.5.7.	void __paixu_ke_xf();	27
5.	班课管理程序	28
5.1.	数据文件结构说明.....	28
5.2.	数据结构说明	28
5.3.	源文件说明	29
5.3.1.	ban_ke_data.c 源文件.....	29
5.3.2.	ban_ke_process.c 源文件.....	30
5.3.3.	ban_ke_data.h 头文件	30
5.3.4.	ban_ke_process.h 头文件.....	30
5.4.	数据处理函数	31
5.4.1.	int read_ban(struct banke_list *list).....	31
5.4.2.	int write_banke(struct banke_list *list)	31
5.4.3.	struct banke_list *add_banke_to_list(struct banck_list *list,struct__banke *banke) 31	

5.4.4.	struct banke_list *get_banke_by_bh(struct banke_list *dest_list,struct banke_list* src_list,char *bianhao)	31
5.4.5.	struct banke_list * get_banke_by_kh(struct banke_list *dest_list,struct banke_list *src_list,char *ke_bh)	31
5.4.6.	int disp_banke_list(struct banke_list *banke,struct banji_list*banji,struct kecheng_list *kecheng).....	32
5.4.7.	struct __banke *getout_banke_by_kh(struct banke_list *list,char *ban_bh,char *ke_bh)	32
5.4.8.	int empty_banke_list(struct banke_list *list)	32
5.5.	业务处理函数	32
5.5.1.	void __disp_ke_ban_bh(char ban_bh[]);	32
5.5.2.	void __disp_ban_ke_bh(char ke_bh[]);	33
5.5.3.	void __zengjia_ban_ke(char ban_bh[]);.....	33
5.5.4.	void __quxiao_ban_ke(char ban_bh[],char ke_bh[])......	33
5.5.5.	void __zengjia_ke_ban(char ke_bh[]).....	33

1. 班级管理程序

1.1. 数据文件结构说明

班级数据都存放在 D:\根目录中, 文件名为 ban.txt, 访问时使用文件全名: “d:\ban.txt”。ban.txt 以文本方式存放数据, 文件中每一行代表一个班级的数据, 每一行中第一个字符串是班级编号, 第二个字符串是专业, 第三个字符串是年纪, 第四个字符串是班号, 每个字符串之间使用制表符 (\t) 分开。格式如图 1-1 所示。



1000	java技术	2010	1
1003	java技术	2012	2
1001	测试技术	2013	2
1004	测试技术	2011	1

图 1-1 ban.txt 文件内容

编写代码时, 请注意使用 fopen、fclose、feof 函数打开、关闭、检测是否读完, 使用 fscanf 和 fprintf 函数完成对文件内容的读与写。

1.2. 数据结构说明

数据结构已经定义, 编写代码时不得修改数据结构。

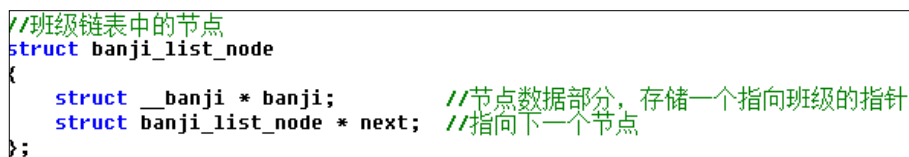
结构体 __banji 代表班级, 有 bianhao、zhuanye、nianji、banhao 四个数据成员, 均为 32 个字符长度的一维字符数组, 以字符串的形式存储班级编号、专业、年级、班号。见图 1-2。代码内容在 ban_data.h 中。如果要使用该数据结构, 请包含 ban_data.h。



```
//自定义结构体: 代表班级
struct __banji
{
    char bianhao[32]; //班级编号
    char zhuanye[32]; //专业
    char nianji[32]; //年级
    char banhao[32]; //班号
};
```

图 1-2 代表班级的结构体

结构体 banj_list_node 代表链表中的节点。这个节点的 banji 属性是 struct __banji *类型的指针变量, 是节点的数据成员; next 属性是指向下一节点的指针变量, 当节点处于链表尾端时, next 的值为 null。结构体定义见图 1-3。代码内容在 ban_data.h 中。如果要使用该数据结构, 请包含 ban_data.h。



```
//班级链表中的节点
struct banj_list_node
{
    struct __banji * banji; //节点数据部分, 存储一个指向班级的指针
    struct banj_list_node * next; //指向下一个节点
};
```

图 1-3 包含班级结构体指针的链表结点

结构体 `banji_list` 代表班级链表。这个节点有 `header` 和 `tail` 两个属性，均是 `struct banji_node` 类型的指针变量,其中 `header` 指向链表的第一个元素,`tail` 指向链表的最后一个元素。

当链表中没有元素时，`header` 和 `tail` 的值都为 `null`。

当链表中仅有一个元素时，`header` 和 `tail` 都指向这一个元素。

通过 `banji_list` 结构体的变量，可以完成所有班级信息在内存中的存储。

`banji_list` 结构体的定义见图 1-4。代码内容在 `ban_data.h` 中。如果要使用该数据结构，请包涵 `ban_data.h`。

```
//班级链表
struct banji_list
{
    struct banji_list_node * header,* tail;//指向链表的第一个元素和最后一个元素。
};
```

图 1-4 用来存储班级信息的链表

1.3. 源文件说明

1.3.1.ban_data.c 源文件

`ban_data.c` 源文件中的代码主要负责班级的文件读写和数据结构组织，`ban_data.h` 中给出了 `ban_data.c` 源文件中所有函数的预定义以及相关的数据结构定义。数据结构的说明请参考 1.2，每个函数的详细信息请见下文。

表格 1-1 `ban_data.c` 中的函数列表

序号	函数名	说明
1	<code>read_ban</code>	从 <code>ban.txt</code> 文件中读出所有班级,并保存到 <code>list</code> 所指的链表中
2	<code>write_ban</code>	将 <code>list</code> 所指的链表中的班级信息都覆盖式地存放到 <code>ban.txt</code> 文件中
3	<code>sort_ban_bh</code>	按照班级编号对链表中的元素予以排序
4	<code>sort_ban_zy</code>	按照班级专业对链表中的元素予以排序
5	<code>sort_ban_nj</code>	按照班级年级对链表中的元素予以排序
6	<code>disp_ban_list</code>	在屏幕上显示链表中班级信息
7	<code>get_ban_by_bh</code>	根据 <code>bianhao</code> 中所存储的班级编号,在 <code>list</code> 所指向的链表中查找班级信息
8	<code>disp_ban</code>	在屏幕上显示指针 <code>p</code> 所指向的班级信息
9	<code>add_ban_to_list</code>	向班级列表尾部增加一个班级
10	<code>getout_ban_by_bh</code>	根据 <code>bianhao</code> 中所存储的班级编号,在 <code>list</code> 所指向的链表中查找班级信息,并移除链表中该节点,同时释放节点内存,
11	<code>empty_ban_list</code>	释放链表中的节点,清空链表

1.3.2.ban_process.c 源文件

ban_process.c 源文件中的代码主要负责班级的文件的业务逻辑，ban_process.h 中给出了 ban_process.c 源文件中所有函数的预定义以及相关的数据结构定义。数据结构的说明请参考 1.2, 每个函数的详细信息请见下文。

表格 1-2 ban_process.c 中函数列表

序号	函数名	说明
1	disp_all_ban	显示所有班级信息
2	disp_ban_bh	根据班级编号显示该班的信息
3	zengjia_ban	让用户在屏幕上输入信息，增加一个班
4	shanchu_ban_bh	根据班级号删除指定班级的信息
5	paixu_ban_bh	根据编号对班级进行排序
6	paixu_ban_zy	根据专业对班级进行排序
7	paixu_ban_nj	根据年级对班级进行排序

1.3.3.ban_data.h 头文件

ban_data.h 头文件给出了 ban_data.c 源文件中的函数定义信息和数据结构定义，见本章前面的内容。

1.3.4.ban_process.h 头文件

ban_process.h 头文件给出了 ban_process.c 源文件中的函数定义信息和数据结构定义，见本章前面内容。

1.4. 数据处理函数

1.4.1.int read_ban(struct banji_list *list)

作用：从 ban.txt 文件中读出所有班级，并保存到 list 所指的链表中。

参数说明：参数 list 类型为 struct banji_list *，指向一个链表，值不能为 null。

返回值：整型，当函数运行成功时，返回从 ban.txt 读取到的班级信息数量，当函数运行不成功时，返回-1。

1.4.2.int write_ban(struct banji_list *list)

作用：将 list 所指链表中包含的班级信息都覆盖式地存放到 ban.txt 文件中，ban.txt 原有的内容都将被改写为 list 中所指链表中包含的班级信息

参数说明：参数 list 类型为 struct banji_list *，指向一个链表，值不能为 null，且链表中

应该存放了需要写入文件的班级信息。

返回值：当写文件成功时，返回所写入的班级的数量，当写文件不成功时，返回-1。

1.4.3.int sort_ban_bh(struct banji_list *list)

作用：按照班级编号对 list 所指班级链表中的元素予以排序。

参数说明：参数 list 类型为 struct banji_list *，指向一个链表，值不能为 null，且链表中应该存放了需要排序的班级信息。

返回值：当排序成功时，返回按班级编号所排序的班级的数量，当排序不成功时，返回-1。

1.4.4.int sort_ban_zy(struct banji_list *list);

作用：按照班级专业对链表中的元素予以排序

参数说明：参数 list 类型为 struct banji_list *，指向一个链表，值不能为 null，且链表中应该存放了需要排序的班级信息。

返回值：当排序成功时，返回按班级编号所排序的班级的数量，当排序不成功时，返回-1。

1.4.5.int sort_ban_nj(struct banji_list *list)

作用：按照班级年级对链表中的元素予以排序

参数说明：参数 list 类型为 struct banji_list *，指向一个链表，值不能为 null，且链表中应该存放了需要排序的班级信息。

返回值：当排序成功时，返回按班级编号所排序的班级的数量，当排序不成功时，返回-1。

1.4.6.int disp_ban_list(struct banji_list *list)

作用：在屏幕上显示链表中班级信息

参数说明：参数 list 类型为 struct banji_list *，指向一个链表，值不能为 null，且链表中应该存放了需要显示的班级信息。

返回值：显示成功时返回所显示的班级信息的个数，否则返回-1。

1.4.7.struct __banji * get_ban_by_bh(struct banji_list *list,char *bianhao)

作用：根据 bianhao 中所存储的班级编号，在 list 所指向的班级链表中查找班级信息

参数说明：参数 list 类型为 struct banji_list *，指向一个链表，值不能为 null，且链表中应该存放了班级信息；参数 bianhao 类型为 char *，指向一个字符串，代表待查找的班级编

号。

返回结果：类型为 `struct __banji *` 类型，当查找成功时返回一个指针，指向查到的班级信息。当查找不成功时，返回 `null`。

1.4.8. `void disp_ban(struct __banji * p)`

作用：在屏幕上显示指针 `p` 所指向的班级信息。

参数说明：参数 `p` 类型为 `struct __banji *`，指向一个班级信息，值不能为空。

返回结果：无。

1.4.9. `struct banji_list * add_ban_to_list(struct banji_list *list, struct __banji *banji)`

作用：向班级列表尾部增加一个班级

参数：参数 `list` 类型为 `struct banji_list *`，指向一个链表，值不能为 `null`，代表将要增加班级的链表。参数 `banji` 类型为 `struct __banji *`，指向一个待增加的班级，值不能为 `null`。

返回结果：当函数成功时，本函数返回所传入的 `list` 指针，当函数运行不成功，本函数返回 `null`。

1.4.10. `struct __banji * getout_ban_by_bh(struct banji_list *list, char *bianhao)`

作用：根据 `bianhao` 中所存储的班级编号，在 `list` 所指向的班级链表中查找班级信息，并将该班级信息从 `list` 所指向的链表中移出。

参数说明：参数 `list` 类型为 `struct banji_list *`，指向一个链表，值不能为 `null`，且链表中应该存放了班级信息；参数 `bianhao` 类型为 `char *`，指向一个字符串，代表待查找的班级编号。

返回结果：类型为 `struct __banji *` 类型，当查找成功时返回一个指针，指向查到的班级信息。当查找不成功时，返回 `null`。

1.4.11. `int empty_ban_list(struct banji_list *list)`

作用：释放链表中的所有节点，清空链表

参数说明：参数 `list` 类型为 `struct banji_list *`，指向一个链表，值不能为 `null`。

返回值：如果成功则返回释放的节点数目，如果不成功则返回-1。

1.5. 业务处理函数

1.5.1. void __disp_all_ban();

作用：调用相关的数据处理函数实现显示 ban.txt 中所有班级信息

参数说明：本函数无参数

返回值：无

1.5.2. void __disp_ban_bh(char ban_bh[]);

作用：调用相关的数据处理函数实现根据班级编号显示该班的信息

参数说明：参数 ban_bh[] 的参数类型为字符串数组，该参数用于存放用户输入的班级编号信息，值不可以为空

返回值：无

1.5.3. void __zengjia_ban();

作用：调用相关的数据处理函数实现让用户在屏幕上输入信息，增加一个班

参数说明：本函数无参数

返回值：无

1.5.4. void __shanchu_ban_bh(char ban_bh[]);

作用：调用相关的数据处理函数实现根据班级号删除指定班级的信息

参数说明：参数 ban_bh[] 的参数类型为字符串数组，该参数用于存放用户输入的班级编号信息，值不可以为空

返回值：无

1.5.5. void __paixu_ban_bh();

作用：调用相关的数据处理函数实现根据编号对班级进行排序

参数说明：本函数无参数

返回值：无

1.5.6. void __paixu_ban_zy();

作用：调用相关的数据处理函数实现根据专业对班级进行排序

参数说明：本函数无参数

返回值：无

1.5.7.void __paxu_ban_nj();

作用：调用相关的数据处理函数实现根据年级对班级进行排序

参数说明：本函数无参数

返回值：无

2. 学生管理程序

2.1. 文件结构说明

学生数据都存放在 C:\根目录中，文件名为学 xuesheng_班级编号.txt，访问时使用文件全名：“c:\xuesheng_班级编号.txt”。xuesheng_班级编号.txt 以文本方式存放数据，文件中每一行代表一个学生的数据，每一行中第一个字符串是学号，第二个字符串是姓名，第三个字符串是年纪性别，第四个字符串是籍贯，第五个字符串是联系方式，每个字符串之间使用制表符（\t）分开。格式如图 2-1 代表班级所对应的学生所示。



文件(F)	编辑(E)	格式(O)	查看(V)	帮助(H)
201202	嘿嘿	男	河南	12354
201203	哼哼	男	河南	12354
201201	呵呵	男	河南	12354
a	a	a	a	a
2	2	2	2	2

图 2-1 代表班级所对应的学生

编写代码时，请注意使用 fopen、fclose、feof 函数打开、关闭、检测是否读完，使用 fscanf 和 fprintf 函数完成对文件内容的读与写。

2.2. 数据结构说明

数据结构已经定义，编写代码时不得修改数据结构。

结构体 __xuesheng 代表学生，有 xuehao、xingming、xingbie、jiguan、lianxifangshi 五个数据成员，均为 32 个字符长度的一维字符数组，以字符串的形式存储学号、姓名、性别、籍贯，联系方式。见[错误!未找到引用源。](#)。代码内容在 xuehseng_data.h 中。如果要使用该数据结构，请包含 xuesheng_data.h。

```
//自定义结构体：代表学生
struct __xuesheng
{
    char xuehao[32];    //学号
    char xingming[32];  //姓名
    char xingbie[32];   //性别
    char jiguan[32];    //籍贯
    char lianxifangshi[32]; //联系方式
};
```

图 2-2 代表学生的结构体

结构体 xuesheng_list_node 代表链表中的节点。这个节点的 xuesheng 属性是 struct __xuesheng *类型的指针变量，是节点的数据成员；next 属性是指向下一节点的指针变量，

当节点处于链表尾端时，next 的值为 null。结构体定义见错误!未找到引用源。。代码内容在 xuesheng_data.h 中。如果要使用该数据结构，请包含 xuesheng_data.h。

```
//学生链表中的节点
struct xuesheng_list_node
{
    struct __xuesheng * xuesheng;
    struct xuesheng_list_node * next;
};
```

图 2-3 包含学生结构体的指针的链表结点

结构体 xuesheng_list 代表学生链表。这个节点有 header 和 tail 两个属性，均是 struct xuesheng_node 类型的指针变量,其中 header 指向链表的第一个元素,tail 指向链表的最后一个元素。

当链表中没有元素时，header 和 tail 的值都为 null。

当链表中仅有一个元素时，header 和 tail 都指向这一个元素。

通过 xuesheng_list 结构体的变量，可以完成所有学生信息在内存中的存储。

xuesheng_list 结构体的定义见错误!未找到引用源。。代码内容在 xuesheng_data.h 中。如果要使用该数据结构，请包含 xuesheng_data.h。

```
//学生链表
struct xuesheng_list
{
    //指向链表的第一个元素和最后一个元素。
    struct xuesheng_list_node * header,* tail;
};
```

图 2-4 用来存储学生信息的

2.3. 源文件说明

2.3.1.xuesheng_data.c

xuesheng_data.c 源文件中的代码主要负责学生的文件读写和数据结构组织，xuesheng_data.h 中给出了 xuesheng_data.c 源文件中所有函数的预定义以及相关的数据结构定义。数据结构的说明请参考 1.2，每个函数的详细信息请见下文。

表格 2-1 xuesheng_data.c 中的函数

序号	函数名	说明
1	read_xuesheng	从 ban.txt 文件中读出所有班级，并保存到 list 所指的链表中
2	write_xuesheng	将 list 所指的链表中的班级信息都覆盖式地存放到 ban.txt 文件中
3	sort_xuesheng_bh	按照班级编号对链表中的元素予以排序
4	sort_xuesheng_zy	按照班级专业对链表中的元素予以排序
5	add_xuesheng_to_list	向班级列表尾部增加一个班级

6	empty_xuesheng_list	释放链表中的节点，清空链表
7	disp_xuesheng_list	在屏幕上显示链表中班级信息
8	get_xuesheng_by_bh	根据 xuehao 中所存储的学生学号，在 list 所指向的链表中查找学生信息
9	getout_xuesheng_by_bh	根据 xuehao 中所存储的学生学号，在 list 所指向的链表中查找学生信息，并移除链表中该节点，同时释放节点内存，
10	disp_xuesheng	在屏幕上显示指针 p 所指向的学生信息
11	*get_xs_filename	根据班级编号，获取存储该班级学生名单的那个文件的文件名
12	disp_xs	通过学号找到某班的学生信息

2.3.2. xuesheng_process.c

xuesheng_process.c 源文件中的代码主要负责学生的文件的业务逻辑，xuesheng_process.h 中给出了 xuesheng_process.c 源文件中所有函数的预定义以及相关的数据结构定义。数据结构的说明请参考 1.2，每个函数的详细信息请见下文。

表格 2-2 xuesheng_process.c 中函数列表

序号	函数名	说明
1	__disp_xs_ban_bh	根据班级编号在屏幕上显示该班所有学生
2	__zengjia_ban_xs	在相应班级编号中增加成绩
3	__shanchu_ban_xs	删除班级编号对应的学生学号
4	__xiugai_ban_xs	修改班级编号对应的学生学号

2.3.3.xuesheng_data.h

xuesheng_data.h 头文件给出了 xuesheng_data.c 源文件中的函数定义信息和数据结构定义，见本章前面的内容。

2.3.4.xuesheng_process.h

xuesheng_process.h 头文件给出了 xuesheng_process.c 源文件中的函数定义信息和数据结构定义，见本章前面内容。

2.4. 数据处理函数

2.4.1.int read_xuesheng(struct xuesheng_list *list,char bianhao[])

作用：根据班级编号从 xuesheng_班级编号.txt 文件中读出所有学生，并保存到 list 所指

的链表中。

参数说明：参数 list 类型为 struct xuesheng_list *, 指向一个链表，值不能为 null。

返回值：整型，当函数运行成功时，返回从 xuesheng_班级编号.txt 读取到的学生信息数量，当函数运行不成功时，返回-1

2.4.2.int write_xuesheng(struct xuesheng_list *list,char ban_bh[])

作用：根据班级编号将 list 所指链表中包含的学生信息都覆盖式地存放到 xuesheng_学生编号.txt 文件中，xuesheng_学生编号.txt 原有的内容都将被改写为 list 中所指链表中包含的学生信息

参数说明：参数 list 类型为 struct xuesheng_list, 指向一个链表，值不能为 null，且链表中应该存放了需要写入文件的学生信息。

返回值：当写文件成功时，返回所写入的学生的数量，当写文件不成功时，返回-1。

2.4.3.int sort_xuesheng_bh(struct xuesheng_list *list)

作用：按照班级编号对链表中的元素予以排序。

参数说明：参数 list 类型为 struct xuesheng_list *, 指向一个链表，值不能为 null，且链表中应该存放了需要排序的班级信息。

返回值：当排序成功时，返回按学生学号所排序的学生的数量，当排序不成功时，返回-1。

2.4.4.int sort_xuesheng_zy(struct xuesheng_list *list)

作用：按照班级专业对链表中的元素予以排序

参数说明：参数 list 类型为 struct xuesheng_list *, 指向一个链表，值不能为 null，且链表中应该存放了需要排序的班级信息。

返回值：当排序成功时，返回按班级编号所排序的班级的数量，当排序不成功时，返回-1。

2.4.5.int disp_xuesheng_list(struct xuesheng_list *list)

作用：在屏幕上显示链表中学生信息

参数说明：参数 list 类型为 struct xuesheng_list *, 指向一个链表，值不能为 null，且链表中应该存放了需要显示的学生信息。

返回值：显示成功时返回所显示的学生信息的个数，否则返回-1。

2.4.6. struct __xuesheng * get_xuesheng_by_bh(struct xuesheng_list *list, char *xuehao)

作用：根据 xuehao 中所存储的学生学号，在 list 所指向的学生链表中查找学生信息

参数说明：参数 list 类型为 struct xuesheng_list *，指向一个链表，值不能为 null，且链表中应该存放了学生信息；参数 xuehao 类型为 char *，指向一个字符串，代表待查找的学生学号。

返回结果：类型为 struct __xuesheng 类型，当查找成功时返回一个指针，指向查到的学生信息。当查找不成功时，返回 null。

2.4.7. struct __xuesheng * getout_xuesheng_by_bh(struct xuesheng_list *list, char *xuehao)

作用：根据 xuehao 中所存储的学生，在 list 所指向的学生链表中查找学生信息，并将该学生信息从 list 所指向的链表中移出。

参数说明：参数 list 类型为 struct xuesheng_list *，指向一个链表，值不能为 null，且链表中应该存放了学生信息；参数 xuehao 类型为 char*，指向一个字符串，代表待查找的学生学号。

返回结果：类型为 struct xuesheng_list *类型，当查找成功时返回一个指针，指向查到的学生信息。当查找不成功时，返回 null。

2.4.8. void disp_xuesheng(struct __xuesheng * p)

作用：在屏幕上显示指针 p 所指向的学生信息。

参数说明：参数 p 类型为 struct __xuesheng *，指向一个学生信息，值不能为空。

返回结果：无。

2.4.9. struct xuesheng_list * add_xuesheng_to_list(struct xuesheng_list *list, struct __xuesheng *xuesheng)

作用：向学生列表尾部增加一个学生

参数：参数 list 类型为 struct xuesheng_list *，指向一个链表，值不能为 null，代表将要增加学生的链表。参数 xuesheng 类型为 struct __xuesheng *，指向一个待增加的学生，值不能为 null。

返回结果：当函数成功时，本函数返回所传入的 list 指针，当函数运行不成功，本函数返回 null。

2.4.10. **int empty_xuesheng_list(struct xuesheng_list *list)**

作用：释放链表中的所有节点，清空链表

参数说明：参数 list 类型为 struct xuesheng_list *，指向一个链表，值不能为 null。

返回值：如果成功则返回释放的节点数目，如果不成功则返回-1。

2.4.11. **char *get_xs_filename(char *filename,char *ban_bh)**

作用：根据班级编号，获取存储该班级学生名单的那个文件的文件名

参数说明：参数 filename：必须是一个足够长的字符串缓冲区，防止溢出，参数 ban_bh：是一个字符串，里面存储班级的编号信息，指向一个链表，值不能为 null。

返回值：如果函数运行成功，则 filename 里面存储着生成的文件名，返回 filename 否则返回 null。

2.4.12. **disp_xs(char ban_bh[],char xs_bh[])**

作用：根据班级编号和学生编号查询学生信息

参数说明：参数 ban_bh[]是字符串数组参数 ban_bh[]：里面存储班级的编号信息，指向一个链表，值不能为 null。参数 xs_bh[]：是一个字符串，里面存储学号信息，指向一个链表，值不能为 null。

返回值：如果函数运行成功，则 ban_bh[]里面存储着生成的文件名，返回 ban_bh[] 否则返回 null。

2.5. 业务处理函数

2.5.1. **void __disp_xs_ban_bh(char bianhao[])**

作用：在学生列表中显示班级中所有学生的信息

参数说明：参数 char bianhao[] 类型为 char,用于处理班级编号

返回值：无返回值

2.5.2. **void __zengjia_ban_xs(char ban_bh[])**

作用：在学生列表中添加学生信息

参数说明：参数：char ban_bh[] 类型：char 用于为增加学生信息提供班级编号信息

返回值：无返回值

2.5.3. void __shanchu_ban_xs(char ban_bh[],char xs_bh[])

作用：调用相关的函数实现对学生的信息进行删除

参数说明： 参数 char ban_bh[] ， char xs_bh[]用于存放班级编号及学生编号

返回值： 无返回值

2.5.4. void __xiugai_ban_xs(char ban_bh[],char xs_bh[])

作用：调用相关的数据处理函数对学生的相关信息进行修改

参数说明： 参数： char ban_bh[],char xs_bh[] 用于存放班级编号及学生编号信息

返回值： 无返回值

3. 成绩管理程序

3.1. 数据文件结构说明

成绩数据都存放在 D:\根目录中，文件名为 chengji_班级编号_课程编号.txt，访问时使用文件全名：“d:\ chengji_班级编号_课程编号.txt”。chengji_班级编号_课程编号.txt 以文本方式存放数据，文件中每一行代表一个成绩的数据，每一行中第一个字符串是学号，第二个字符串是姓名，第三个字符串是平时成绩，第四个字符串是考试成绩，第五个字符串是综合成绩，每个字符串之间使用制表符（\t）分开。格式如图 3-1 所示。

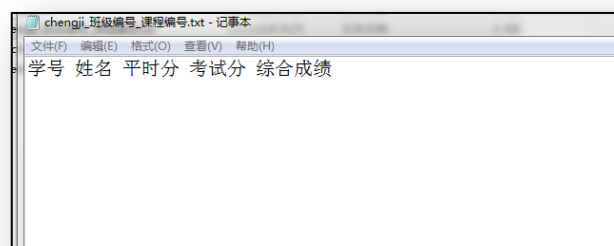


图 3-1 chengji_班级编号_课程编号.txt 文件内容

编写代码时，请注意使用 fopen、fclose、feof 函数打开、关闭、检测是否读完，使用 fscanf 和 fprintf 函数完成对文件内容的读与写。

3.2. 数据结构说明

数据结构已经定义，编写代码时不得修改数据结构。

结构体 __chengji 代表班级，有 xuehao、name、pingshi、kaoshi、zonghe 四个数据成员，均为 32 个字符长度的一维字符数组，以字符串的形式存储学生学号、名字、平时信息、考试信息、综合成绩。见图 3-2。代码内容在 chengji_data.h 中。如果要使用该数据结构，请

包含 chengji_data.h。

```
struct __chengji
{
    char xuehao[32];    //学号
    char name[32];     //姓名
    char pingshi[32];   //平时分
    char kaoshi[32];    //考试分
    char zonghe[32];    //综合成绩
};
```

图 3-2 代表成绩的结构体

结构体 chengji_list_node 代表链表中的节点。这个节点的 chengji 属性是 struct __chengji *类型的指针变量，是节点的数据成员；next 属性是指向下一节点的指针变量，当节点处于链表尾端时，next 的值为 null。结构体定义见图 3-3。代码内容在 chengji_data.h 中。如果要使用该数据结构，请包含 chengji_data.h

```
struct chengji_list_node
{
    struct __chengji * chengji;    //节点数据部分，存储一个指向班级的指针
    struct chengji_list_node * next; //指向下一个节点
};
```

图 3-3 包含成绩结构体指针的链表结点

结构体 chengji_list 代表成绩链表。这个节点有 header 和 tail 两个属性，均是 struct chengji_node 类型的指针变量,其中 header 指向链表的第一个元素,tail 指向链表的最后一个元素。

当链表中没有元素时，header 和 tail 的值都为 null。

当链表中仅有一个元素时，header 和 tail 都指向这一个元素。

通过 chengji_list 结构体的变量，可以完成所有成绩信息在内存中的存储。

chengji_list 结构体的定义见图 3-4。代码内容在 chengji_data.h 中。如果要使用该数据结构，请包含 chengji_data.h

```
struct chengji_list
{
    struct chengji_list_node * header,* tail;//指向链表的第一个元素和最后一个元素。
};
```

图 3-4 用来存储成绩信息的链表

3.3. 源文件说明

3.3.1.chengji_data.c

chengji_data.c 源文件中的代码主要负责成绩的文件读写和数据结构组织，chengji_data.h 中给出了 chengji_data.c 源文件中所有函数的预定义以及相关的数据结构定义。数据结构的说明请参考 1.2，每个函数的详细信息请见下文。

表格 3-1 chengji_data.c 中的函数列表

序号	函数名	说明
1	read_chengji	从 chengji_班级编号_课程编号.txt 文件中读出所有学生，并保存到 list 所指的链表中
2	write_chengji	将 list 所指的链表中的学生成绩信息都覆盖式地存放到 chengji_班级编号_课程编号.txt 文件中
3	disp_chengji_list	在屏幕上显示链表中学生成绩的信息
4	get_chengji_by_bh	根据 xuehao 中所存储的班级编号，在 list 所指向的链表中查找学生成绩信息
5	disp_chengji	在屏幕上显示指针 p 所指向的成绩信息
6	add_chengji_to_list	向班级列表尾部增加一个成绩
7	empty_chengji_list	释放链表中的节点，清空链表

chengji_process.c 源文件中的代码主要负责班级的文件的业务逻辑，

chengji_process.h 中给出了 chengji_process.c 源文件中所有函数的预定义以及相关的数据结构定义。数据结构的说明请参考 1.2，每个函数的详细信息请见下文。

3.3.2.chengji_process.c

表格 3-2 chengji_process.c 中函数列表

序号	函数名	说明
1	disp_ban_ke_cj	在屏幕上显示该班所有学生的一门课程的成绩
2	disp_ban_xs_cj	在屏幕上显示某班某学生的详细成绩信息
3	zengjia_ban_ke_cj	让用户在屏幕上输入信息，增加一个学生成绩
4	shanchu_ban_ke_cj	根据班级号、课程号、学号删除指定班级的信息
5	xiugai_ban_ke_cj	根据编号对学生成绩进行排序

3.3.3.cengji_data.h

cengji_data.h 头文件给出了 cengji_data.c 源文件中的函数定义信息和数据结构定义，见

本章前面的内容。

3.3.4.cengji _process.h

cengji _process.h 头文件给出了 cengji_process.c 源文件中的函数定义信息和数据结构定义，见本章前面内容。

3.4. 数据处理函数

3.4.1.int read_chengji(struct chengji_list *list,char ban_bh[],char ke_bh[])

作用：从 chengji_班级编号_课程编号.txt 文件中读出所有 成绩，并保存到 list 所指的链表中。

参数说明：参数 list 类型为 struct chengji_list *，指向一个链表，值不能为 null。

返回值：整型，当函数运行成功时，返回从 chengji_班级编号_课程编号.txt 读取到的成绩信息数量，当函数运行不成功时，返回-1。

3.4.2.int write_chengji(struct chengji_list *list,char ban_bh[],char ke_bh[])

作用：将 list 所指链表中包含的成绩信息都覆盖式地存放到 chengji_班级编号_课程编号.txt 文件中，chengji_班级编号_课程编号.txt 原有的内容都将被改写为 list 中所指链表中包含的成绩信息

参数说明：参数 list 类型为 struct chengji_list *，指向一个链表，值不能为 null，且链表中应该存放了需要写入文件的成绩信息。

返回值：当写文件成功时，返回所写入的成绩的数量，当写文件不成功时，返回-1。

3.4.3.int disp_chengji_list(struct chengji_list *list)

作用：在屏幕上显示链表中成绩信息

参数说明：参数 list 类型为 struct chengji_list *，指向一个链表，值不能为 null，且链表中应该存放了需要显示的成绩信息。

返回值：显示成功时返回所显示的成绩信息的个数，否则返回-1。

3.4.4. struct __chengji * get_chengji_by_xh(struct chengji_list *list, char *xuehao)

作用：根据 chengji_班级编号_课程编号.txt 中所存储的学号，在 list 所指向的成绩链表中查找成绩信息

参数说明：参数 list 类型为 struct chengji_list *，指向一个链表，值不能为 null，且链表中应该存放了成绩信息；参数 xuehao 类型为 char *，指向一个字符串，代表待查找的学号。

返回结果：类型为 struct __chengji * 类型，当查找成功时返回一个指针，指向查到的成绩信息。当查找不成功时，返回 null。

3.4.5. void disp_chengji_list(struct chengji_list *list chengji_list *list)

作用：在屏幕上显示指针 p 所指向的成绩信息。

参数说明：参数 p 类型为 struct __chengji *，指向一个成绩信息，值不能为空。

返回结果：无。

3.4.6. struct chengji_list * add_chengji_to_list(struct chengji_list *list, struct __chengji *chengji)

作用：向成绩列表尾部增加一个成绩

参数：参数 list 类型为 struct chengji_list *，指向一个链表，值不能为 null，代表将要增加成绩的链表。参数 chengji 类型为 struct __chengji *，指向一个待增加的成绩，值不能为 null。

返回结果：当函数成功时，本函数返回所传入的 list 指针，当函数运行不成功，本函数返回 null。

3.4.7. int empty_chengji_list(struct chengji_list *list)

作用：释放链表中的所有节点，清空链表

参数说明：参数 list 类型为 struct chengji_list *，指向一个链表，值不能为 null。

返回值：如果成功则返回释放的节点数目，如果不成功则返回 -1。

3.5. 业务处理函数

3.5.1. void __disp_ban_ke_cj(char ban_bh[], char ke_bh[])

作用：调用相关的数据处理函数实现显示 chengji_班级编号_课程编号.txt 中所有成绩信息

参数说明：参数 `ban_bh` 类型为字符串数组，指向一个字符串，代表待查找成绩的班级编号；
参数 `ke_bh` 类型为 `char`，指向一个字符串，代表待查找成绩的课程编号。
返回值：无

3.5.2.void __zengjia_ban_ke_cj(char ban_bh[],char ke_bh[])

作用：调用相关的数据处理函数实现让用户在屏幕上输入信息，增加一个成绩
参数说明：参数 `ban_bh[]` 的参数类型为字符串数组，该参数用于存放用户输入的班级编号信息，值不可以为空；参数 `ke_bh[]` 的参数类型为字符串数组，该参数用于存放用户输入的课程编号信息，值不可以为空
返回值：无

3.5.3.void __xiugai_ban_ke_cj(char ban_bh[],char ke_bh[],char xs_bh[])

作用：调用相关的数据处理函数实现让用户在屏幕上输入信息，修改一个成绩
参数说明：参数 `ban_bh[]` 的参数类型为字符串数组，该参数用于存放用户输入的班级编号信息，值不可以为空；参数 `ke_bh[]` 的参数类型为字符串数组，该参数用于存放用户输入的课程编号信息，值不可以为空；参数 `xs_bh[]` 的参数类型为字符串数组，该参数用于存放用户输入的学生编号信息，值不可以为空
返回值：无

3.5.4.void __shanchu_ban_ke_cj(char ban_bh[],char ke_bh[],char xs_bh[])

作用：调用相关的数据处理函数实现根据班级编号，课程编号，学生编号删除指定成绩的信息
参数说明：参数 `ban_bh[]` 的参数类型为字符串数组，该参数用于存放用户输入的班级编号信息，值不可以为空；参数 `ke_bh[]` 的参数类型为字符串数组，该参数用于存放用户输入的课程编号信息，值不可以为空；参数 `xs_bh[]` 的参数类型为字符串数组，该参数用于存放用户输入的学生编号信息，值不可以为空
返回值：无

3.5.5.void __disp_ban_xs_cj(char ban_bh[],char xs_bh[])

作用：调用相关的数据处理函数实现根据班级编号，学生编号对成绩进行排序
参数说明：参数 `ban_bh[]` 的参数类型为字符串数组，该参数用于存放用户输入的班级编号信息，值不可以为空；参数 `xs_bh[]` 的参数类型为字符串数组，该参数用于存放用户输入的

学生编号信息，值不可以为空

返回值：无

4. 课程管理系统

4.1. 文件结构说明

课程数据都存放在 D:\根目录中，文件名为 kecheng.txt，访问时使用文件全名：“d:\kecheng.txt”。kecheng.txt 以文本方式存放数据，文件中每一行代表一个课程的数据，每一行中第一个字符串是课程编号，第二个字符串是课程名称，第三个字符串是学分，第四个字符串是简介，每个字符串之间使用制表符（\t）分开。格式如图 4-1 所示。

图 4-1 kecheng.txt 文件

编写代码时，请注意使用 fopen、fclose、feof 函数打开、关闭、检测是否读完，使用 fscanf 和 fprintf 函数完成对文件内容的读与写。

4.2. 数据结构说明

数据结构已经定义，编写代码时不得修改数据结构。

结构体 __kecheng 代表班级，有 bianhao、mingcheng、xuefen、jianjie 四个数据成员，均为 32 个字符长度的一维字符数组，以字符串的形式存储课程编号、课程名称、学分、简介。见图 4-2。代码内容在 kecheng_data.h 中。如果要使用该数据结构，请包含 kecheng_data.h。

```
//自定义结构体：代表课程
struct __kecheng
{
    char bianhao[32]; //课程编号
    char mingcheng[32]; //课程名称
    char xuefen[32]; //学分
    char jianjie[32]; //简介
};
```

图 4-2 代表课程的结构体

结构体 kecheng_list_node 代表链表中的节点。这个节点的 kecheng 属性是 struct __kecheng *类型的指针变量，是节点的数据成员；next 属性是指向下一节点的指针变量，当节点处于链表尾端时，next 的值为 null。结构体定义见图 4-3。代码内容在 kecheng_data.h 中。如果要使用该数据结构，请包含 kecheng_data.h。

```
//课程链表中的节点
struct kecheng_list_node
{
    struct __kecheng * kecheng;    //节点数据部分，存储一个指向课程的指针
    struct kecheng_list_node * next; //指向下一个节点
};
```

图 4-3 包含课程结构体指针的链表结点

结构体 kecheng_list 代表课程链表。这个节点有 header 和 tail 两个属性，均是 struct kecheng_node 类型的指针变量,其中 header 指向链表的第一个元素,tail 指向链表的最后一个元素。

当链表中没有元素时，header 和 tail 的值都为 null。

当链表中仅有一个元素时，header 和 tail 都指向这一个元素。

通过 kecheng_list 结构体的变量，可以完成所有课程信息在内存中的存储。

kecheng_list 结构体的定义见图 4-4。代码内容在 kecheng_data.h 中。如果要使用该数据结构，请包含 kecheng_data.h。

```
//课程链表
struct kecheng_list
{
    struct kecheng_list_node * header,* tail;//指向链表的第一个元素和最后一个元素。
};
```

图 4-4 用来存储课程信息的链表

4.3. 源文件说明

4.3.1.kecheng_data.c 源文件

kecheng_data.c 源文件中的代码主要负责课程的文件读写和数据结构组织，kecheng_data.h 中给出了 kecheng_data.c 源文件中所有函数的预定义以及相关的数据结构定义。数据结构的说明请参考 1.2，每个函数的详细信息请见下文。

表格 4-1 ban_data.c 中的函数列表

序号	函数名	说明
1	read_ke	从 kecheng.txt 文件中读出所有课程，并保存到 list 所指的链表中
2	write_ke	将 list 所指的链表中的课程信息都覆盖式地存放到 kecheng.txt 文件中
3	sort_ke_bh	按照课程编号对链表中的元素予以排序
4	sort_ke_xuefen	按照课程学分对链表中的元素予以排序
5	sort_ke_mingcheng	按照课程名称对链表中的元素予以排序
6	disp_ke_list	在屏幕上显示链表中课程信息

7	get_ke_by_bh	根据 bianhao 中所存储的课程编号，在 list 所指向的链表中查找课程信息
8	disp_ke	在屏幕上显示指针 p 所指向的课程信息
9	add_ke_to_list	向课程列表尾部增加一个课程
10	empty_ke_list	释放链表中的节点，清空链表
11	getout_ke_by_bh	根据 bianhao 中所存储的课程编号，在 list 所指向的链表中查找课程信息，并移除链表中该节点，同时释放节点内存

4.3.2.kecheng _process.c 源文件

kecheng_process.c 源文件中的代码主要负责班级的文件的业务逻辑，kecheng_process.h 中给出了 kecheng _process.c 源文件中所有函数的预定义以及相关的数据结构定义。数据结构的说明请参考 1.2，每个函数的详细信息请见下文。

表格 4-2 kecheng_process.c 中函数列表

序号	函数名	说明
1	__disp_all_ke	显示所有课程
2	__disp_ke_bh	显示一门课程详细情况
3	__shanchu_ke_bh	根据课程编号删除一门课
4	__zengjia_ke	让用户完成增加一门课程的操作
5	__paixu_ke_bh	根据课程编号排序
6	__paixu_ke_mc	根据课程名称排序
7	__paixu_ke_xf	根据课程学分排序

4.3.3.kecheng _data.h 头文件

kecheng_data.h 头文件给出了 kecheng _data.c 源文件中的函数定义信息和数据结构定义，见本章前面的内容。

4.3.4.kecheng _process.h 头文件

kecheng_process.h 头文件给出了 kecheng _process.c 源文件中的函数定义信息和数据结构定义，见本章前面内容。

4.4. 数据处理函数

4.4.1.int read_ke(struct kecheng_list *list);

作用：从 kecheng.txt 文件中读出所有课程，并保存到 list 所指的链表中业务逻辑函数

参数说明：参数 list 类型为 struct kecheng_list *，指向一个链表，值不能为空

返回值：整型，当函数运行成功时，返回从 kecheng.txt 中读取的课程信息数量，当函数运行不成功时，返回-1

4.4.2.int write_ke(struct kecheng_list *list);

作用：将 list 所指链表中包含的课程信息都覆盖式地存放到 kecheng.txt 文件中，kecheng.txt 原有的内容都将被改写为 list 中所指链表中包含的课程信息

参数说明：参数 list 类型为 struct kecheng_list *，指向一个链表，值不能为 null，且链表中应该存放了需要写入文件的课程信息。

返回值：当写文件成功时，返回所写入的课程的数量，当写文件不成功时，返回-1。

4.4.3.int sort_ke_bh(struct kecheng_list *list);

作用：按照课程编号对 list 所指课程链表中的元素予以排序。

参数说明：参数 list 类型为 struct kecheng_list *，指向一个链表，值不能为 null，且链表中应该存放了需要排序的课程信息。

返回值：当排序成功时，返回按课程编号所排序的课程的数量，当排序不成功时，返回-1。

4.4.4.int sort_ke_xuefen(struct kecheng_list *list);

作用：按照课程学分对 list 所指课程链表中的元素予以排序。

参数说明：参数 list 类型为 struct kecheng_list *，指向一个链表，值不能为 null，且链表中应该存放了需要排序的课程信息。

返回值：当排序成功时，返回按课程学分所排序的课程的数量，当排序不成功时，返回-1。

4.4.5.int sort_ke_mingcheng(struct kecheng_list *list);

作用：按照课程名字对 list 所指课程链表中的元素予以排序。

参数说明：参数 list 类型为 struct kecheng_list *，指向一个链表，值不能为 null，且链表中应该存放了需要排序的课程信息。

返回值：当排序成功时，返回按课程名字所排序的课程的数量，当排序不成功时，返回-1。

4.4.6.int disp_ke_list(struct kecheng_list *list);

作用：在屏幕上显示链表中课程信息

参数说明：参数 list 类型为 struct kecheng_list *,指向一个链表，值不能为空

返回值：整型，当函数运行成功时，返回从 kecheng.txt 中读取的课程信息数量，当函数运行不成功时，返回-1

4.4.7.struct __kecheng * get_ke_by_bh(struct kecheng_list *list,char *bianhao);

作用：根据 bianhao 中所存储的课程编号，在 list 所指向的链表中查找课程信息，并移除链表中该节点，同时释放节点内存

参数说明：参数 list 类型为 struct kecheng_list *, 指向一个链表，值不能为空

参数 bianhao 类型为 char *, 指向一个课程的编号，值不能为空

返回值：指向课程信息的指针，当查找成功时返回该课程信息的指针，当查找不成功时返回 null

4.4.8.void disp_ke(struct __kecheng * p);

作用：在屏幕上显示指针 p 所指向的课程信息

参数说明：参数 list 类型为 struct __kecheng * p

返回值：无

4.4.9.struct kecheng_list * add_ke_to_list(struct kecheng_list *list,str__kecheng*kecheng);

作用：向课程列表尾部增加一个课程

参数说明：参数 list 类型为 struct kecheng_list*,指向一个链表，值不能为 null，参数 list 代表将要增加课程的链表

参数 kecheng 类型为 struct __kecheng *, 指向一个链表，值不能为 null，参数 kecheng 指向一个待增加的课程

返回值：当成功时返回课程的 list, 当函数运行不成功，本函数返回 null。

4.4.10. int empty_ke_list(struct kecheng_list *list);

作用：释放链表中的节点，清空链表

参数说明：参数 list 类型为 struct kecheng_list *, 指向一个链表，不能为空值

返回值：整型，当函数运行成功时，则返回释放的节点数目，当函数运行不成功时，返回-1

4.4.11. struct __kecheng * getout_ke_by_bh(struct kecheng_list *list,char *bianhao)

作用：根据 bianhao 中所存储的课程编号，在 list 所指向的链表中查找课程信息，并移除链表中该节点，同时释放节点内存

参数说明：参数 list 类型为 struct kecheng_list *，指向一个链表，不能为空值

返回值：整型，当函数运行成功时，则返回释放的节点数目，当函数运行不成功时，返回-1

4.5. 业务处理函数

4.5.1.void __disp_all_ke();

作用：将 kecheng.txt 文件中的所有课程显示在界面上

参数说明：无参数

返回值：无

4.5.2.void __disp_ke_bh();

作用：显示一门课程详细情况

参数说明：本函数无参数

返回值：无

4.5.3.void __shanchu_ke_bh(char ke_bh[]);

作用：根据课程编号删除一门课

参数说明：字符数组（即课程编号），用户界面上输入要删除的课程编号

返回值：无

4.5.4.void __zengjia_ke();

作用：让用户完成增加一门课程的操作

参数说明：本函数无参数

返回值：无

4.5.5.void __paixu_ke_bh();

作用：根据课程编号排序

参数说明：本函数无参数

返回值：无

4.5.6.void __paixu_ke_mc();

作用：根据课程名称排序

参数说明：本函数无参数

返回值：无

4.5.7.void __paixu_ke_xf();

作用：根据课程学分排序

参数说明：本函数无参数

返回值：无

5. 班课管理程序

5.1. 数据文件结构说明

班级授课信息数据都存放在 D:\根目录中，文件名为 ban_ke.txt，访问时使用文件全名：“d:\ban_ke.txt”。ban_ke.txt 以文本方式存放数据，文件中每一行代表一个班级授课信息的数据，每一行中第一个字符串是学期，第二个字符串是班级编号，第三个字符串是课程编号，第四个字符串是授课老师，每个字符串之间使用制表符（\t）分开。格式如图 5-1 所示。

学期	班级编号	课程编号	授课老师
2011	1001	0019	刘德华
2012	1001	0010	张昭
2012	1002	0199	刘秀
2009	1003	0192	王雪

图 5-1 ban_ke.txt 文件内容

编写代码时，请注意使用 fopen、fclose、feof 函数打开、关闭、检测是否读完，使用 fscanf 和 fprintf 函数完成对文件内容的读与写。

5.2. 数据结构说明

数据结构已经定义，编写代码时不得修改数据结构。

结构体 __banke 代表班级，有 xueqi、bbianhao、kbianhao、jiaoshi 四个数据成员，均为 32 个字符长度的一维字符数组，以字符串的形式存储学期、班级编号、课程编号、教师。见图 5-2。代码内容在 ban_ke_data.h 中。如果要使用该数据结构，请包含 ban_ke_data.h。

```
struct __banke
{
    char xueqi[32]; //学期
    char bbianhao[32]; //班级编号
    char kbianhao[32]; //课程编号
    char jiaoshi[32]; //教师
};
```

图 5-2 代表班课的结构体

结构体 banke_list_node 代表链表中的节点。这个节点的 banke 属性是 struct __banke *类型的指针变量，是节点的数据成员；next 属性是指向下一节点的指针变量，当节点处于链表尾端时，next 的值为 null。结构体定义见图 1-3。代码内容在 ban_ke_data.h 中。如果

要使用该数据结构，请包含 ban_ke_data.h。

```
//班课链表中的节点
struct banke_list_node
{
    struct __banke * banke;           //节点数据部分，存储一个指向班级的指针
    struct banke_list_node * next;    //指向下一个节点
};
```

图 5-3 包含班课结构体指针的链表结点

结构体 banke_list 代表班级链表。这个节点有 header 和 tail 两个属性，均是 struct banke_node 类型的指针变量,其中 header 指向链表的第一个元素,tail 指向链表的最后一个元素。

当链表中没有元素时，header 和 tail 的值都为 null。

当链表中仅有一个元素时，header 和 tail 都指向这一个元素。

通过 banke_list 结构体的变量，可以完成所有班级信息在内存中的存储。

banke_list 结构体的定义见图 1-4。代码内容在 ban_ke_data.h 中。如果要使用该数据结构，请包涵 ban_ke_data.h。

```
//班级链表
struct banke_list
{
    struct banke_list_node * header,* tail;//指向链表的第一个元素和最后一个元素。
};
```

图 5-4 用来存储班课信息的链表

5.3. 源文件说明

5.3.1.ban_ke_data.c 源文件

ban_ke_data.c 源文件中的代码主要负责班级的文件读写和数据结构组织，ban_ke_data.h 中给出了 ban_ke_data.c 源文件中所有函数的预定义以及相关的数据结构定义。数据结构的说明请参考 1.2，每个函数的详细信息请见下文。

表格 5-1 ban_ke_data.c 中的函数列表

序号	函数名	说明
1	read_banke	从 ban_ke.txt 文件中读出所有班级，并保存到 list 所指的链表中
2	write_banke	将 list 所指的链表中的班级信息都覆盖式地存放到 ban_ke.txt 文件中
3	add_banke_to_list	向班级授课信息列表尾部增加一个班级授课信息

4	get_banke_by_bh	根据班级编号查找课程信息
5	disp_banke_list	在屏幕上显示指针 p 所指向的班级信息
6	get_banke_by_kh	根据 bianhao 中所存储的课程编号, 在 list 所指向的链表中查找班级授课信息
7	getout_ban_by_kh	根据班级编号和课程编号号中, 在 list 所指向的链表中查找班级授课信息, 并移除链表中该节点, 同时释放节点内存
8	empty_banke_list	清空链表

5.3.2.ban_ke_process.c 源文件

ban_ke_process.c 源文件中的代码主要负责班级的文件的业务逻辑, ban_ke_process.h 中给出了 ban_ke_process.c 源文件中所有函数的预定义以及相关的数据结构定义。数据结构的说明请参考 1.2, 每个函数的详细信息请见下文。

表格 5-2 ban_process.c 中函数列表

序号	函数名	说明
1	__disp_ke_ban_bh	调用相关的数据处理函数实现根据班级编号显示该班的课程信息
2	__disp_ban_ke_bh	调用相关的数据处理函数实现根据课程编号显示班级的信息
3	__zengjia_ban_ke	调用相关的数据处理函数实现让用户在屏幕上输入班级编号信息, 增加课程
4	__quxiao_ban_ke	调用相关的数据处理函数实现根据课程编号增加班级授课信息
5	__zengjia_ke_ban	调用相关的数据处理函数实现根据课程编号增加班级授课信息

5.3.3.ban_ke_data.h 头文件

ban_ke_data.h 头文件给出了 ban_ke_data.c 源文件中的函数定义信息和数据结构定义, 见本章前面的内容。

5.3.4.ban_ke_process.h 头文件

ban_ke_process.h 头文件给出了 ban_ke_process.c 源文件中的函数定义信息和数据结构定义, 见本章前面内容。

5.4. 数据处理函数

5.4.1.int read_ban(struct banke_list *list)

作用：从 ban_ke.txt 文件中读出所有班级，并保存到 list 所指的链表中

参数说明：参数 list 类型为 struct banke_list *，指向一个链表，值不能为 null

返回值：整型，当函数运行成功时，返回从 ban_ke.txt 读取到的班级授课信息数量，当函数运行不成功时，返回-1。

5.4.2.int write_banke(struct banke_list *list)

作用：将 list 所指的链表中的班级授课信息都覆盖式地存放到 ban_ke.txt 文件中。

参数说明：参数 list 类型为 struct banke_list *，指向一个链表，值不能为 null。

返回值：整型，当写文件成功时，返回所写入的班级的数量，当写文件不成功时，返回-1

5.4.3.struct banke_list *add_banke_to_list(struct banck_list *list,struct__banke *banke)

作用：向班级列表尾部增加一条班级授课信息

参数说明：参数 list 代表将要增加班级授课信息的链表，值不能为 null，参数 banke 指向一个待增加的班级授课信息，值不能为 null

返回值：当函数成功时，本函数返回 list 指针。当函数运行不成功，本函数返回 null。

5.4.4.struct banke_list *get_banke_by_bh(struct banke_list *dest_list,struct banke_list* src_list,char *bianhao)

作用：根据班级编号查找课程信息

参数说明：dest_list 代表符合信息的班课信息列表，值为空，src_list 代表包含全部班课信息的列表，值不能为空,bianhao 指向待查找的班级编号

返回值：当函数成功时，本函数返回 dest_list 指针。当函数运行不成功，本函数返回 null。

5.4.5.struct banke_list * get_banke_by_kh(struct banke_list *dest_list,struct banke_list *src_list,char *ke_bh)

作用：根据班级编号查找课程信息

参数说明：dest_list 代表符合信息的班课信息列表，值为空，src_list 代表包含全部班课信

息的列表，值不能为空.ke_bh 指向待查找的课程编号.

返回结果：当函数成功时，本函数返回 dest_list 指针。当函数运行不成功，本函数返回 null。

5.4.6.int disp_banke_list(struct banke_list *banke,struct banji_list*banji,struct kecheng_list *kecheng)

作用：在屏幕上显示指针 p 所指向的班级信息。

参数说明：参数 banke 类型为 struct __banke *，指向一个班级授课信息，值不能为空 参数 banji 类型为 struct __banji *，指向一个班级信息，值不能为空。参数 kecheng 类型为 struct kecheng_list *,指向一个课程信息，值不能为空

返回结果：无。

5.4.7.struct __banke *getout_banke_by_kh(struct banke_list *list,char *ban_bh,char *ke_bh)

作用：根据班级编号和课程编号号中，在 list 所指向的链表中查找班级信息，并移除链表中该节点，同时释放节点内存。

参数说明：list 代表班课信息列表，值不能为空，ban_bh 代表待删除的班级编号，ke_bh 代表待删除的课程编号

返回结果：当查找成功时，返回该班课信息的指针，当查找不成功时，返回 null

5.4.8.int empty_banke_list(struct banke_list *list)

作用：释放链表中的所有节点，清空链表

参数说明：参数 list 类型为 struct banke_list *，指向一个链表，值不能为 null。

返回值：如果成功则返释放的节点数目，如果不成功则返回-1。

5.5. 业务处理函数

5.5.1.void __disp_ke_ban_bh(char ban_bh[]);

作用：调用相关的数据处理函数实现根据课程编号显示课程信息

参数说明：参数 ban_bh[]的参数类型为字符串数组，该参数用于存放用户输入的班级编号信息，值不可以为空

返回值：无

5.5.2.void __disp_ban_ke_bh(char ke_bh[]);

作用：调用相关的数据处理函数实现根据课程编号显示班级信息

参数说明：参数 ke_bh[]的参数类型为字符串数组，该参数用于存放用户输入的班级编号信息，值不可以为空

返回值：无

5.5.3.void __zengjia_ban_ke(char ban_bh[]);

作用：调用相关的数据处理函数实现让用户在屏幕上输入班级编号信息，增加班级授课信息

参数说明：参数 ban_bh[]的参数类型为字符串数组，该参数用于存放用户输入的班级编号信息，值不可以为空

返回值：无

5.5.4.void __quxiao_ban_ke(char ban_bh[],char ke_bh[])

作用：调用相关的数据处理函数实现根据班级编号和课程编号删除班级课程信息

参数说明：参数 ban_bh[]的参数类型为字符串数组，该参数用于存放用户输入的班级编号信息，值不可以为空，参数 ke_bh[]的参数类型为字符串数组，该参数用于存放用户输入的课程编号信息，值不可以为空

返回值：无

5.5.5.void __zengjia_ke_ban(char ke_bh[])

作用：调用相关的数据处理函数实现根据课程编号增加班级

参数说明：参数 ke_bh[]的参数类型为字符串数组，该参数用于存放用户输入的课程编号信息，值不可以为空

返回值：无