

电子科技大学信息与软件工程学院

实 验 报 告

学 号 2018091619034

姓 名 王省

(实验) 课程名称 程序设计与算法 II

理论教师 刘峤

实验教师 刘峤

电子科技大学

实验报告

学生姓名：王省 学号：2018091619034

指导教师：刘峤

实验地点：基础实验大楼 506

实验时间：2019.4.19、2019.4.26

一、 实验名称：学生课程程序查询

二、 实验学时：10 学时

三、 实验目的：

1. 掌握单链表的定义和使用方法
2. 掌握单链表的建立方法
3. 掌握单链表中节点的查找与删除
4. 掌握输出单链表节点的方法
5. 掌握链表节点排序的一种方法
6. 掌握结构体的定义和使用方法
7. 掌握读写文件的方法
8. 掌握栈的定义和使用方法
9. 掌握链式存储栈的建立方法、基本操作方法
10. 掌握队列的定义和使用方法
11. 掌握链式存储队列的建立方法、基本操作方法

四、 实验原理：

1. 结构体：结构体是由不同数据类型组成的，结构是可能具有相同或不同类型的值(成员)的集合。组成结构的每一个数据称为该结构的成员项。
2. 结构体的定义：结构体的定义是宣布该结构由哪几个成员项组成。用以下方式定义一个结构：

```
struct 结构名{
```

```

数据类型 成员名 1;
数据类型 成员名 2
...
};

```

例如：在书这个结构中，包括了标题，作者，价格三个成员。

```

struct book{
    char title[MAXTITLE];
    char author[MAXAUTL];
    float value;
};

```

3. 结构体变量的声明：以以下形式声明一个结构体变量：

<存储类型> struct 结构名 结构变量

例如 struct book library。

也可以将定义和声明同时进行。例如：

```

struct book{
    char title[MAXTITLE];
    char author[MAXAUTL];
    float value;
} library1, library2;

```

library1 中的成员不会与 library2 中的成员冲突。

每个结构变量都有三个成员:title(书的名称)、auther(书的作者) 和 value(书的价格)。这里的声明格式和 C 语言中其他变量的声明格式一样。struct{...} 指明了类型。而 library1 和 library2 则是具有这种类型的变量。结构的成员在内存中是按照声明的顺序存储的。

4. 计算结构变量的内存大小：通常使用 sizeof 运算求出。
5. 结构体变量的初始化：一般形式为：

struct 结构名 结构变量 = {初始数据}

例如：

```
struct book library = {"to Kill a Mocking Bird"; "Harper Lee";43.5}
```

其实现的效果和对其成员赋值的效果一样。

```

library .title = "to Kill a Mocking Bird";
library .author = "Harper Lee";

```

library .value = 43.5

6. 结构指针：指向结构的指针称为结构指针。其定义格式为：

<存储类型> struct 结构名 *结构指针名

例如 struct library *p;

使用指针对结构成员进行引用的时候，一般有两种形式：

(*结构指针名).成员名

结构指针名->成员名

例如

(*p). book; p->book;

这两种等价

7. 结构类型的定义：形式为 typedef <存储类型> 新命名。

例如：

typedef p* nodeptr;

8. 链表存储结构：链表是一种很常见重要的结构。它是存储分配的一种结构。链表有一个头指针的变量，一般以 head 表示。它没有数据域，只有指针域。存放一个地址，指向第一个元素。链表中每一个元素称为“节点”，每一个节点包含两个部分：(1)用户所需要的实际数据，即数据域；(2)下一个节点指向下一个指针。

9. 建立动态链表：步骤：(1) 使用 malloc()函数为一个结构分配足够的空间。(2). 存储这个结构的地址。(3). 把正确的信息复制到这个结构中。

10. 文件操作实例

FILE *fp; //定义文件操作的指针

//fopen 用于打开文件，接收两个参数，一个是文件的路径，一个是文件打开的方式。例如 xxxxxx.txt 和该项目的可执行文件在同一目录下，则此处只需要所读取内容文件名，r 代表以只读方式打开文件

fp = fopen("xxxxxxx.dat", "r");

//如果以 w 方式代表打开只写文件，若文件存在则长度清为 0，即该文件内容消失，若不存在则创建该文件，其余打开方式请自行查阅文档

fp = fopen("xxxxxxx.dat", "w");

//fscanf 用于从 txt 文件中读取内容，下例以字符串形式读入一段字符并存放在 tempstring 中

fscanf(fp, "%s", tempstring);

```

//或者以格式化的方式读入字符串
fscanf(fp, "%t%s\n", tempstring);
//fprintf 以格式化的方式向 txt 文件中写入内容
fprintf(fp, "%s\t", tempstring);
//检查文件内容是否已经读到文件结束符了
while ( !feof(fp)){.....}
//最后需要使用 fclose 关闭文件指针
fclose(fp);

```

11. 栈的定义及特点：限定仅在表的一端进行插入或删除操作的线性表
允许插入和删除的一端称栈顶，另一端称栈底。含元素的空表称空栈。
特点：后进先出（LIFO）或 先进后出（FILO）。

12. 链式存储栈的节点定义：

```

typedef struct node {
    ElemType data;
    struct node *next;
}TNode, *PStack;

```

13. 链式存储栈的基本操作：初始化：init_stack(S)、进栈：push_stack(S, e)、
退栈：pop_stack(S)、查看栈顶：top_stack(S)、判栈空，判栈满
14. 队列的定义及特点：限定只能在表的一端插入，在另一端删除的线性表队
列的特点：先进先出（First In First Out，FIFO）
15. 链队列的节点定义：

```

typedef struct{
    TNode *front;    // 队头指针
    TNode *rear;     // 队尾指针
    int len;         // 队列长度
} TQue,*PQue;

```

16. 链队列的基本操作：初始化：init_queue(Q)、入队：enqueue(Q, e)、出队：
dequeue(Q)、判队空，判队满（循环队列）

五、实验内容：

用 C 语言、单链表、栈、队列数据结构实现一个学生成绩查询系统。具备

输入学生信息、课程信息、成绩信息、读入以上信息、生成所有信息、生成六十分以下学生名单、用栈完成学生信息的逆序输出、用链式队列进行学生成绩的排序。具体实现步骤如下：

1. 软件界面控制：实现一个数字选项式的启动界面，包含输入学生信息、输入课程信息、输入成绩信息、读入学生信息、读入课程信息、读入成绩信息、生成所有课程信息、查询某单个课程信息、生成成绩低于 60 分的学生课程信息、用链式栈逆序输出学生信息、用队列进行成绩信息的排序 11 个选项。这些功能可以循环调用，如图 5-1 所示；

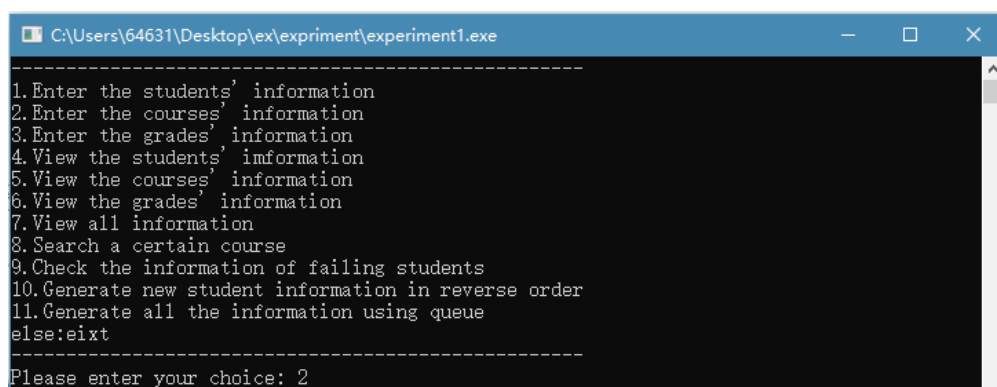


图 5-1

2. 输入学生信息：定义一个函数，完成学生信息的输入。可输入十个学生信息记录，并保存至文件 student.dat。其中软件技术专业 5 人，人工智能专业 5 人，如图 5-2 所示；

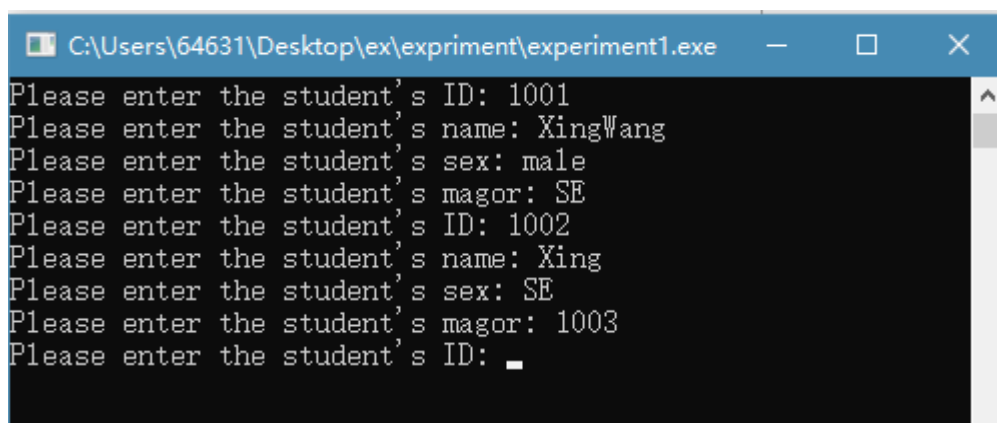


图 5-2

3. 输入课程信息：定义一个函数，完成对课程信息的输入。可输入 3 个课程

(数据库、数据结构、程序设计) 信息记录, 并保存至文件 `course.dat` 文件中, 如图 5-3 所示;

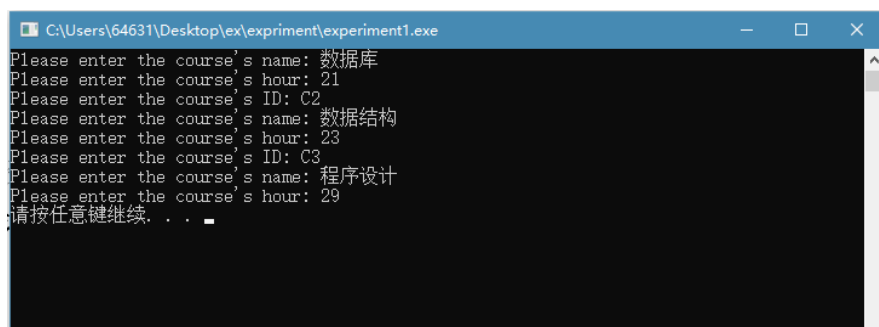


图 5-3

4. 输入学生成绩信息: 定义一个函数, 完成对学生成绩的输入。输入上面 10 位同学的成绩, 并存储至文件 `courseGrade.dat` 中, 如图 5-4 所示。

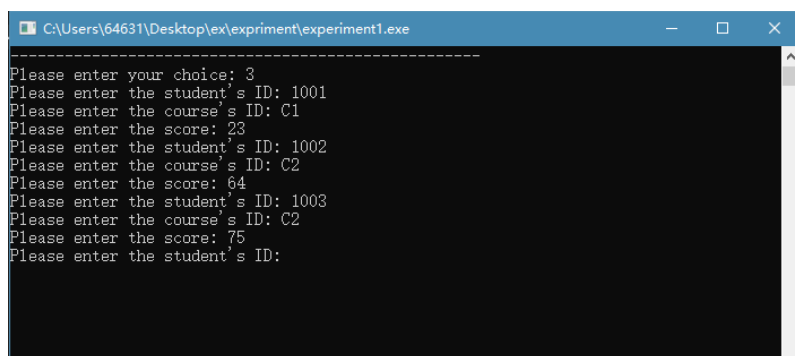


图 5-4

5. 学生信息的读取: 定义一个函数, 完成从文件 `student.dat` 中读出学生信息, 并生成单向链表, 定义一个函数, 完成对学生按照学号的升序排列, 定义一个函数, 完成对学生的信息在屏幕上的显示输出, 如图 5-5 所示;

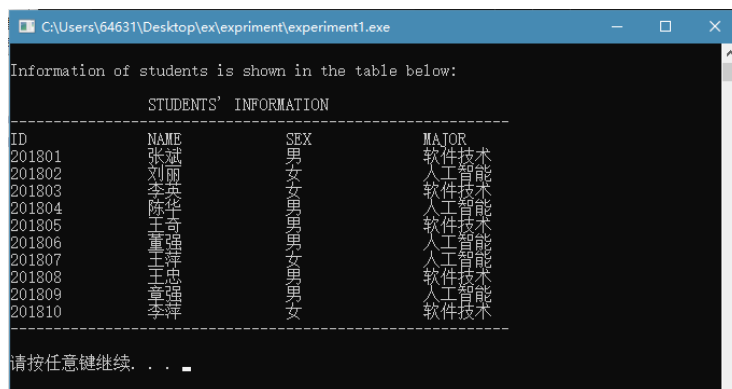
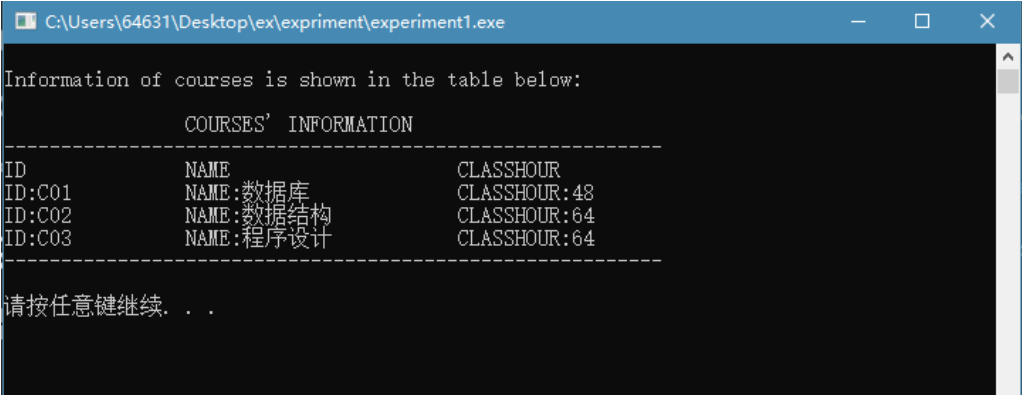


图 5-5

6. 课程信息的读取: 定义一个函数, 完成从文件 `course.dat` 中读出课程信息, 并生成单向链表, 定义一个函数, 完成对课程信息按照课程号的升序排列, 定义一个函数, 完成对课程信息在屏幕上的显示输出, 如图 5-6 所示;



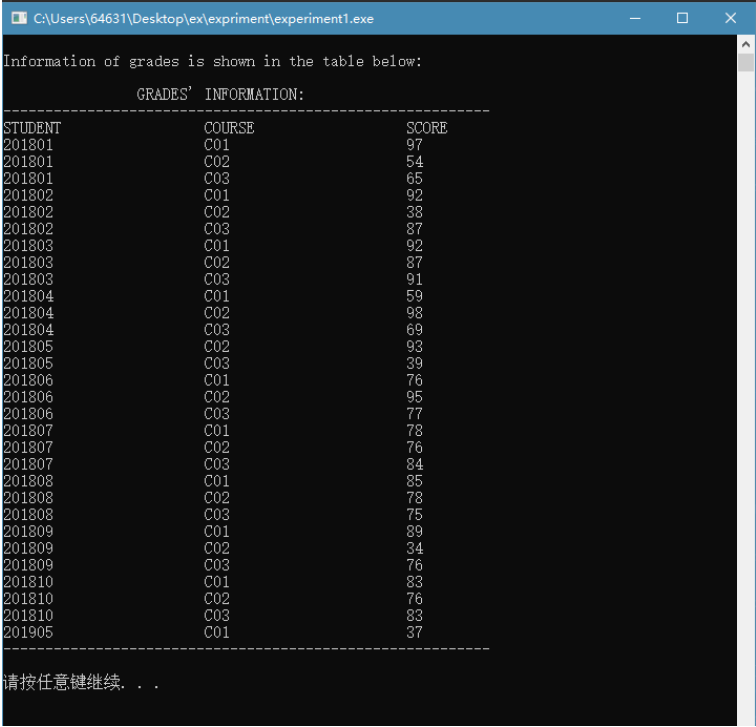
```
Information of courses is shown in the table below:

-----
COURSES' INFORMATION
-----
ID          NAME          CLASSHOUR
ID:C01      NAME:数据库    CLASSHOUR:48
ID:C02      NAME:数据结构    CLASSHOUR:64
ID:C03      NAME:程序设计    CLASSHOUR:64
-----

请按任意键继续. . .
```

图 5-6

7. 成绩信息的读取: 定义一个函数, 完成从文件 `courseGrade.dat` 中读出课程成绩信息, 并生成单向链表, 定义一个函数, 完成对成绩信息按照学号和课程号的升序排列, 定义一个函数, 完成对成绩信息在屏幕上的显示输出, 如图 5-7 所示;



```
Information of grades is shown in the table below:

-----
GRADES' INFORMATION:
-----
STUDENT    COURSE    SCORE
201801     C01       97
201801     C02       54
201801     C03       65
201802     C01       92
201802     C02       38
201802     C03       87
201803     C01       92
201803     C02       87
201803     C03       91
201804     C01       59
201804     C02       98
201804     C03       69
201805     C02       93
201805     C03       39
201806     C01       76
201806     C02       95
201806     C03       77
201807     C01       78
201807     C02       76
201807     C03       84
201808     C01       85
201808     C02       78
201808     C03       75
201809     C01       89
201809     C02       34
201809     C03       76
201810     C01       83
201810     C02       76
201810     C03       83
201905     C01       37
-----

请按任意键继续. . .
```

图 5-7

8. 查询所有学生的所有课程的考试成绩：定义一个函数，查询所有学生所有课程的考试成绩，并生成该课程的成绩单链表，包括学号、学生姓名、专业、课程名、考试成绩等信息。定义一个函数，将该链表按考试成绩降序排列。定义一个函数，将学生的该成绩信息输出到文件 studentGrade.dat 中。定义一个函数在屏幕上显示输出，如图 5-8 所示：

```

All information is shown in the table below:

INFORMATION
-----
COURSE      ID      NAME      MAJOR      SCORE
-----
数据结构    201804    陈华      人工智能    98
编译原理    201801    张斌      软件技术    97
编译原理    201806    董强      人工智能    95
数据结构    201805    王奇      软件技术    93
编译原理    201802    刘丽      人工智能    92
编译原理    201803    李英      软件技术    92
编译原理    201809    章强      人工智能    89
数据结构    201803    李强      软件技术    87
程序设计    201802    刘丽      人工智能    87
编译原理    201808    王忠      软件技术    85
程序设计    201807    王萍      软件技术    84
编译原理    201810    李萍      软件技术    83
程序设计    201810    李萍      软件技术    83
编译原理    201807    王萍      软件技术    78
数据结构    201808    王忠      软件技术    78
程序设计    201806    董强      人工智能    77
编译原理    201806    董强      人工智能    76
数据结构    201807    王萍      软件技术    76
编译原理    201810    李萍      软件技术    76
程序设计    201809    章强      软件技术    76
程序设计    201808    王忠      软件技术    75
程序设计    201804    陈华      软件技术    69
程序设计    201801    陈华      软件技术    65
编译原理    201804    陈华      人工智能    59
数据结构    201801    张斌      软件技术    54
程序设计    201805    王奇      软件技术    39
数据结构    201802    刘丽      人工智能    38
数据结构    201809    章强      人工智能    34

请按任意键继续...

```

图 5-8

9. 查询指定课程号的所有学生的考试成绩：定义一个函数，查询某课程号的信息（包括学号、学生姓名、专业、课程名、考试成绩），生成该课程的成绩单链表，定义一个函数，将该链表按照考试成绩降序排列，定义一个函数，将该链表输出到屏幕上显示，如图 5-9 所示：

```

Enter the course ID: C01

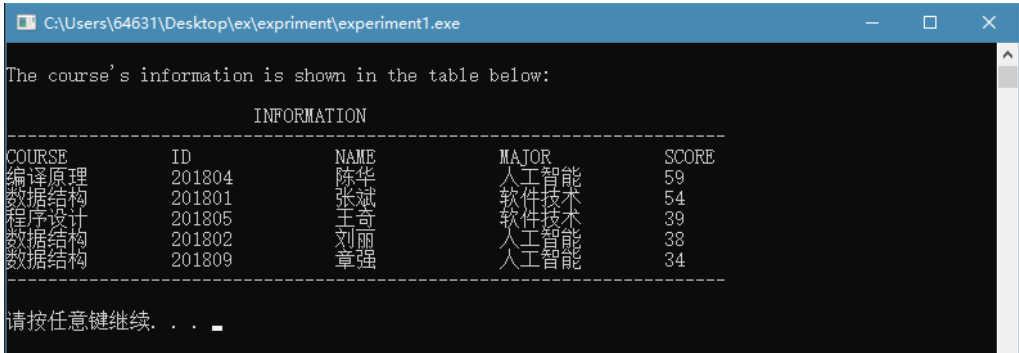
INFORMATION
-----
COURSE      ID      NAME      MAJOR      SCORE
-----
编译原理    201801    张斌      软件技术    97
编译原理    201802    刘丽      人工智能    92
编译原理    201803    李英      软件技术    92
编译原理    201809    章强      人工智能    89
编译原理    201808    王忠      软件技术    85
编译原理    201810    李萍      软件技术    83
编译原理    201807    王萍      软件技术    78
编译原理    201806    董强      人工智能    76
编译原理    201804    陈华      人工智能    59

请按任意键继续...

```

图 5-9

10. 查询考试成绩小于 60 分的学生信息：定义一个函数，查询指定课程号的考试成绩小于 60 分的学生成绩信息（包括学号、学生姓名、专业、课程名、考试成绩），生成该课程的成绩链表，定义一个函数，将该链表按照考试成绩降序排列，定义一个函数，将该链表在屏幕上显示输出，如图 5-10 所示：



```
C:\Users\64631\Desktop\ex\expriment\experiment1.exe
The course's information is shown in the table below:

-----
INFORMATION
-----
COURSE      ID      NAME      MAJOR      SCORE
编译原理    201804    陈华      人工智能    59
数据结构    201801    张斌      软件技术    54
程序设计    201805    王奇      软件技术    39
数据结构    201802    刘丽      人工智能    38
数据结构    201809    章强      人工智能    34
-----

请按任意键继续. . .
```

图 5-10

11. 用栈逆序输出学生信息：定义一个函数，定义栈，实现打印输出，定义函数完成对栈的初始化操作，定义函数完成入栈出栈，定义函数完成栈的销毁，结果如图 5-9 所示。



```
C:\Users\64631\Desktop\ex\expriment\experiment1.exe
Please enter your choice: 10
Information of students in reverse order is shown in the table below:

-----
STUDENTS' INFORMATION
-----
ID      NAME      SEX      MAJOR
201810    李萍      女      软件技术
201809    章强      男      人工智能
201808    王忠      男      软件技术
201807    王萍      女      人工智能
201806    董强      男      人工智能
201805    王奇      男      软件技术
201804    陈华      男      人工智能
201803    李英      女      软件技术
201802    刘丽      女      人工智能
201801    张斌      男      软件技术
-----

请按任意键继续. . .
```

图 5-11

12. 用队列完成查询所有学生的所有课程的考试成绩：定义一个函数，完成对队列的定义，实现打印输出，定义函数完成对队列的初始化操作，定义函数完成入列出列，定义函数完成队列按照考试成绩降序排序，定义函数完

成队列的销毁操作，结果如图 5-12 所示。

图 5-12

六、实验器材（设备、元器件）：

个人电脑一台

七、实验步骤：

1. 完成对学生信息，课程信息，成绩表，学生成绩信息，栈，队列的结构体定义。
2. 定义并完成函数 `void snode_input(Slist **p)`，完成单个节点学生信息的输入。
3. 定义并完成函数 `void stu_info_input()`，输入 10 个学生记录，其中软件技术专业 5 人，人工智能专业 5 人，并存入文件 `student.dat` 中；
4. 定义并完成函数 `void cou_info_input()`，完成单个节点课程信息输入。
5. 定义并完成函数 `void cnode_input()`，输入 3 门课程（数据库、数据结构、程序设计）信息记录，并存入文件 `course.dat` 中；

6. 定义并完成函数 `void gnode_input(Glist **p)`, 完成单个节点成绩表的输入。
7. 定义并完成函数 `void gra_info_input(Stable *pstu, Ctable *pcou)` , 输入上述 10 位同学分别选修上述三门课程的考试成绩到文件 `courseGrade.dat` 中。
8. 定义并完成函数 `void stu_init(Stable **pstu)`, `pstu` 为学生信息链表, 从文件 `student.dat` 中读出学生信息, 单向链表。
9. 定义并完成函数 `void stu_sorted(Stable **pstu)`, `pstu` 为学生信息链表, 将链表按学号升序排序。
10. 定义并完成函数 `void stu_output_all(Stable *pre)`, `pre` 为学生信息链表, 将链表中的信息打印至屏幕上。
11. 定义并完成函数 `void cou_init(Ctable **pcou)`, `pcou` 为课程信息单链表, 从文件 `course.dat` 中读出课程信息, 生成单向链表。
12. 定义并完成函数 `void cou_sorted(Ctable **pcou)`, `pcou` 为课程信息单链表, 将课程信息链表按课程号升序排列。
13. 定义并完成函数 `void cou_output_all(Ctable *pre)`, `pre` 为课程信息单链表, 将课程信息链表中的信息打印至屏幕上。
14. 定义并完成函数 `void gra_init(Gtable **pgra)`, `pgra` 为成绩信息单链表, 从文件 `courseGrade.dat` 中读出成绩信息, 生成单链表。
15. 定义并完成函数 `void gra_sno_sorted(Gtable **pgra)`, `pgra` 为成绩信息单链表, 将该链表按照学号和课程号升序排列。
16. 定义并完成函数 `void gra_output_all(Gtable *pre)`, `pre` 为成绩信息单链表, 将该链表中的信息打印至屏幕上。
17. 定义并完成函数 `SCGtable *scg_init(Stable *pstu, Ctable *pcou, Gtable *pgra)`, `pstu` 为学生信息单链表, `pcou` 为课程信息单链表, `pgra` 为成绩信息单链表, 查询所有学生所有课程的考试成绩, 生成该课程的成绩单链表, 包括学号、学生姓名、专业、课程名、考试成绩等信息, 并将学生的该成绩信息输出到文件 `studentGrade.dat` 中, 并返回该链表。
18. 定义并完成函数 `void gra_score_sorted(SCGtable **pscg)`, `pscg` 为学生成绩信息链表, 将该链表按照考试成绩降序排列
19. 定义并完成函数 `void scg_output_all(SCGtable *pscg)`, `pscg` 为学生成绩信息链表, 将该链表信息在屏幕上显示输出。
20. 定义并完成函数 `void gra_stu_search(Stable *pstu, Ctable *pcou, Gtable *pgra)`, 在 `pscg` 链表中查询指定课程号的所有学生的考试成绩, 生成该课程的成绩

单链表，包括学号、学生姓名、专业、课程名、考试成绩等信息。

21. 定义并完成函数 `void stack(Stack *pstu)`，`pstu` 为学生信息链表，该函数包含了栈的定义以及栈基本函数的接口，包含栈的初始化，入栈出栈，打印输出。
22. 定义并完成函数 `bool isempty_stack(Stack *ps)`，`ps` 为栈，当栈为空时，返回 `true` 值，当栈非空时，返回 `false` 值。
23. 定义并完成函数 `void init_stack(Stack **ps)`，完成栈的初始化操作。
24. 定义并完成函数 `void push_stack(Stack **ps, Stack *pstu)`，其中 `ps` 为栈，`pstu` 为学生信息链表。用 `while` 循环将学生信息入栈。
25. 定义并完成函数 `Stack* pop_stack(Stack **ps)`，将栈中的元素取出，并将其依次保存至新的链表中，并将新链表返回。最后销毁栈。
26. 定义并完成函数 `void que(SCGtable *pscg)`，`pscg` 为学生成绩信息链表，该函数包含了对队列的定义以及队列的基本函数接口，包含队列的初始化，入栈排序，打印输出操作。
27. 定义并完成函数 `bool isempty_queue(QUE *que)`，完成队列的判空操作，当队列为空时返回 `true`，当非空时返回 `false` 值。
28. 定义并完成函数 `void en_queue(QUE **que, SCGtable *pscg)`，完成元素入队操作。将链表 `pscg` 中的值依次入队。
29. 定义并完成函数 `void menu()`，实现一个数字选项式的启动界面，包含输入学生信息、输入课程信息、输入成绩信息、读入学生信息、读入课程信息、读入成绩信息、生成所有课程信息、查询某单个课程信息、生成成绩低于 60 分的学生课程信息、用链式栈逆序输出学生信息、用队列进行成绩信息的排序 11 个选项。这些功能可以循环调用。

八、实验结果与分析（含重要数据结果分析或核心代码流程分析）

1. 单个学生信息输入

```
printf("Please enter the student's ID: ");
read_line(&p->data.sno,12);
printf("Please enter the student's name: ");
read_line(&p->data.sname,10);
printf("Please enter the student's sex: ");
read_line(&p->data.sex,4);
printf("Please enter the student's major: ");
```

```
read_line((*p)->data.major,20);
```

代码功能：实现对单个学生节点的输入。

算法正确性：使用 `read_line()` 函数实现字符串的写入。分别实现 `data` 域四个数据的写入。

2. 学生信息写入文件

```
FILE *fp = fopen("student.dat", "w");           //写文件
for(int i = 0; i < 10; ++i){
    Slist* p = (Slist*)malloc(sizeof(Slist));
    snode_input(&p);                             //单个节点的信息输入
    fprintf(fp, "%s\t%s\t%s\t%s\t", p->data.sno, p->data.sname, p->
data.sex, p->data.major);
    if(i != 9) fprintf(fp, "\n");                //除开最后一行外，空行
}
fclose(fp);
```

代码功能：将 10 个学生信息依次写入文件。

算法正确性：用 `w` 打开文件，在一个 `for` 循环里初始化学生信息节点，输入学生信息并写入文件，最后关闭文件。

3. 从文件中读取学生信息

```
if (check_nullfile("student.dat"))                //判断文件是否存在
{
    FILE *fp = fopen("student.dat", "r");
    while(!feof(fp)){
        fscanf(fp, "%s%s%s%s\n", p->data.sno,
p->data.sname, p->data.sex, p->data.major);
        fclose(fp);
    }
else{
    printf("Please enter the information first!");
    system("pause");
    exit(0);
}
```

代码功能：从文件中读取学生的信息。

算法正确性分析：使用 `check_nullfile()` 函数判断文件是否存在，若存在，则打开该文件，用 `while` 循环读入文件信息直至文件末尾，用 `fscanf()` 函数读取文

件内容并保存至单链表节点的 **data** 域中，最后关闭文件；若文件不存在，会提醒用户首先输入信息，并退出。

4. 将学生信息节点生成单链表。

```
Slist* p = (Slist*)malloc(sizeof(Slist));
if((*pstu)->snode == NULL){
    (*pstu)->snode = p;
}
else{
    pre->next = p;
}
p->next = NULL;
pre = p;
++(*pstu)->rows;
```

代码功能：将学生信息节点链接成单链表。

算法正确性分析：当链表无节点时，将该节点插入至头部，当链表有节点时，使用尾插法，将 **p** 插入至尾部。并使尾指针的 **next** 域至为 **NULL**。

5. 使用冒泡排序法对链表进行排序。

```
Slist *p = (*pstu)->snode;
for(int i = 1; i < (*pstu)->rows; ++i){
    for( int j = 0; j < (*pstu)->rows-i; ++j){
        if(strcmp(p->data.sno,p->next->data.sno) > 0){
            Slist *temp = (Slist*)malloc(sizeof(Slist));
            temp->data = p->data;
            p->data = p->next->data;
            p->next->data = temp->data;
        }
        p = p->next;
    }
    p = (*pstu)->snode;
}
```

代码功能：将学生信息按照学号升序进行排序。

算法正确性分析：使用冒泡排序法，使用 **strcmp()** 比较两个节点中 **data** 域的 **sno** 值，若前一个元素大于后一个元素，则交换两个节点中的 **data** 域。

6. 将学生链表信息整体打印输出。

```
Slist *p = pstu->snode;
printf("\nID\t\tNAME\t\tSEX\t\tMAJOR\t\t\n");
while(p){
    printf("%s\t\t%s\t\t%s\t\t%s\t\t\n",p->data.sno,
        p->data.sname,p->data.sex,p->data.major);
    p = p->next;
}
```

代码功能：实现对学生信息的打印输出。

算法正确性：使用 while 循环，当 p 不为 NULL 时打印输出链表的节点信息。用制表符将信息制表输出。

7. 课程、考试成绩等代码同 1-6 可得。不做详细阐述。

8. 查找所有课程的相关信息。

```
while(q){
    r = pgra->gnode;
    while(r){
        //查询课程表中课程
        if(strcmp(r->data.cno,q->data.cno)==0){
            p = pstu->snode;    //查询成功
            while(p){
                //查询学生信息
                if(strcmp(p->data.sno,r->data.sno) == 0){
                    SCGlist* temp = (SCGlist*)malloc(sizeof(SCGlist));
                    temp->data.score = r->data.score;
                    strcpy(temp->data.sname,p->data.sname);
                    strcpy(temp->data.major,p->data.major);
                    strcpy(temp->data.sno,p->data.sno);
                    strcpy(temp->data.cname,q->data.cname);
                    if(pscg->scgnode == NULL){
                        pscg->scgnode = temp;
                    }
                    else{
                        pre->next = temp;
                    }
                    temp->next = NULL;
                    pre = temp;
                    ++(pscg->rows);
                }
            }
        }
        r = r->next;
    }
    q = q->next;
}
```



```

        p = p->next;
    }
}
r = r->next;
}
q = q->next;
}

```

代码功能：查询所有课程的相关成绩。

算法正确性分析：使用三个 while 循环，最外层查找课程的名称（课程名称的唯一性），中间层查找学生的学号，最内层查找该学生的相关信息。

9. 查找某个课程的代码与上类似，不作过多阐述。

10. 查找成绩低于 60 分的学生信息。

```

while(p){
    if(p->data.score < 60){
        SCGlist* temp = (SCGlist*)malloc(sizeof(SCGlist));
        temp->data = p->data;
        if(belowsixty->scgnode == NULL){
            belowsixty->scgnode = temp;
        }
        else{
            pre->next = temp;
        }
        temp->next = NULL;
        pre = temp;
        ++(belowsixty->rows);

        p = p->next;
    }
}

```

代码功能：实现对成绩低于 60 分的学生信息查找。

算法正确性分析：对 pscg 链表进行遍历，若 data 域的 score 值小于 60，将之赋值到新的链表节点中。

11. 栈的初始化操作。

```

(*ps)->pTop = (STACKNODE*)malloc(sizeof(STACKNODE));
if((*ps)->pTop == NULL){

```

```

        printf("Wrong!");
        system("pause");
        exit(0);
    }
    else{
        (*ps)->pBottom = (*ps)->pTop;
        (*ps)->pTop->next = NULL;
    }
}

```

代码功能：实现对栈的初始化操作。

算法正确性分析：为栈的节点分配内存，若分配失败，则退出。若分配成功，则将该节点置为 top 指针和 bottom 指针，并将 top 指针的 next 域置为 NULL。

12. 栈的判空操作。

```

bool isempty_stack(STACK *ps){
    return ps->pTop == ps->pBottom;
}

```

代码功能：实现栈的判空。

算法正确性分析：当 top 指针和 bottom 指针相等时，即为栈空，返回 true 值，相反则返回 false 值。

13. 压栈操作。

```

for(int i = 0; i < pstu->rows; ++i){
    STACKNODE *pNew = (STACKNODE*)malloc(sizeof(STACKNODE));
    if(pNew){
        pNew->data = pstu->snode->data;
        pNew->next = (*ps)->pTop;
    }
    else{
        printf("Wrong!");
        system("pause");
        exit(0);
    }
    (*ps)->pTop = pNew;
    pstu->snode = pstu->snode->next;
}

```

代码功能：实现将学生信息的节点输入到栈中。

算法正确性分析：首先为节点分配空间，当分配成功时，将 `pstu` 的数据域复制到该节点中，再将该节点插入到栈中。

14. 去除栈顶元素并使用链表返回。

```
while(!isempty_stack(*ps)){
    p = (*ps)->pTop;
    (*ps)->pTop = p->next;

    Slist* temp = (Slist*)malloc(sizeof(Slist));
    temp->data = p->data;
    if(pSTU->snode == NULL){
        pSTU->snode = temp;
    }
    else{
        pre->next = temp;
    }
    temp->next = NULL;
    pre = temp;
    ++(pSTU->rows);
    free(p);
}
*ps = NULL;
return pSTU;
```

功能分析：实现元素出栈操作并以链表返回。

算法正确性分析：将栈顶元素 `p` 取出，`free` 掉，用尾插法将之构成一个新的链表。

15. 队列的初始化操作：

```
QUENODE *p = (QUENODE*)malloc(sizeof(QUENODE));
if(p){
    (*que)->front = p;
    (*que)->rear = p;
    (*que)->len = 0;
    p->next = NULL;
}
else{
    printf("Wrong!");
    system("pause");
}
```

```
        exit(0);
    }
```

功能分析：实现队列的初始化操作。

算法正确性分析：为节点 **p** 分配空间，若分配成功，则将 **que** 指针和 **rear** 指针都等于 **p** 指针，并使这两个指针的 **next** 域为 **NULL**。若未分配成功，则提示用户并退出程序。

16. 入队列操作：

```
for(int i = 0; i < pscg->rows ;++i){
    QUEENODE *p = (QUEENODE*)malloc(sizeof(QUEENODE));
    if(p){
        p->next = NULL;
        p->data = pscg->scgnode->data;
        (*que)->rear->next = p;
        (*que)->rear = p;
    }
    else{
        printf("Wrong!");
        system("pause");
        exit(0);
    }
    pscg->scgnode = pscg->scgnode->next;
}
```

代码功能：将链表的每一个元素取出，并放入队列里。

算法正确性分析：为指针 **p** 分配空间，将 **p** 的 **next** 域设为空，将原链表节点指针的 **data** 域拷贝至 **p** 指针的 **data** 域。并将 **rear** 指针的 **next** 域指向 **p**。最后将 **p** 拷贝至 **rear** 指针。

17. 链表销毁操作。

```
while (p)
{
    q = p;
    p = q->next;
    free(q);
    q = NULL;
}
*L = NULL;
```

代码功能：销毁链表，释放链表内存。

算法正确性分析：让 q 等于 p ，再让 p 指向下一个节点，释放 q 的内存，让 q 为空。循环上述结构，直到 p 为空。最后使头指针为空。

18. 应用结果：

分析：该程序能将学生信息写入文件，读取文件内容并保存至链表中，将链表信息写入文件中，使用冒泡排序法将链表排序，使用字符串匹配查找需要的信息，用链表内容入栈并出栈，实现链表的逆序，使用队列将

正确与否：能够实现一个学生信息管理系统，能够成功运行。

是否完成功能：1. 该系统具备输入学生信息、课程信息、成绩信息、读入以上信息、生成所有信息、生成六十分以下学生名单、用栈完成学生信息的逆序输出、用链式队列进行学生成绩的排序这几个功能。2. 根据需要加入了判断是否成功分配内存，是否写入信息等保护性措施，使程序更加健壮。

九、总结及心得体会：

1. 总结：

- 1) 程序优点：该程序能够完美的运行，实现了要求的功能，并加入了一些保护性措施。
- 2) 程序缺点：有大量逻辑相同，几乎重复的代码，修改无从下手。

2. 心得体会：

- 1) 加深了对链表的理解，对栈的理解，以及对队列的理解。
- 2) 抗压能力有所提高。

十、对本实验过程及方法、手段的改进建议：

1. 希望能够减少一些对链表的重复操作。原因：1. 构建三个链表虽说在某些细节中有微小差异，但其基本逻辑大致相同，有许多重复的操作。2. 上学期已经写过一次大型实验，其中几乎涵盖了链表的所有基本操作，再做一次实验感觉意义不大。

2. 希望能增加一些对栈和队列的基本操作。例如增加评卷环节，判断括号是否正确打印出等。
3. 实验报告格式尚有歧义，希望明确并将模板给得更详细正确。实验报告模板也未按照格式给出：例如标题字体和行间距，正文字体及行间距，页面格式等。

报告评分：

指导教师签字：