**程序总设计和算法分析**

目录

[1.设计过程 1](#_Toc504291584)

[1.1设想供选择的方案 1](#_Toc504291585)

[1.2选取合理的方案 2](#_Toc504291586)

[1.3制定编程计划 2](#_Toc504291587)

[1.3.1 时间计划和小组成员任务分配 2](#_Toc504291588)

[1.3.2整个程序的菜单部分 2](#_Toc504291589)

[1.3.3设计整个程序的大纲 3](#_Toc504291590)

[1.4小组编程 5](#_Toc504291591)

[1.5 小组审核整个代码和优化代码 5](#_Toc504291592)

[2.设计原理 5](#_Toc504291593)

[2.1模块化设计 5](#_Toc504291594)

[2.2 设计原理结构图 6](#_Toc504291595)

[3.算法分析 7](#_Toc504291596)

[3.1 选择结构的使用 7](#_Toc504291597)

[3.2 循环结构的使用 9](#_Toc504291598)

[3.3 函数的调用 11](#_Toc504291599)

[3.4 指针的定义和使用 12](#_Toc504291600)

[3.5 链表 13](#_Toc504291601)

[3.6 文件的读取 15](#_Toc504291602)

# 1.设计过程

## 1.1设想供选择的方案

在总体设计阶段应该考虑各种可能实现的方案，并且力求选出最佳的方案。在本程序的整体设计中可以通过数组来实现添加题目也可以通过链表的方式添加题目。

## 1.2选取合理的方案

充分比较不同的方案，一旦选出最佳方案，将大大提高效率。本程序用数组的方式添加题目，它限定了题目的个数且实现题目删除和修改较困难，而链表的方式添加题目，删体 ，改题方便很多。最终以设置结构体的方式，用链表的形式实现添题，删题，改题，保题，以及后面的保存题目，查看分数等部分都可以使用链表，方便，效率高。

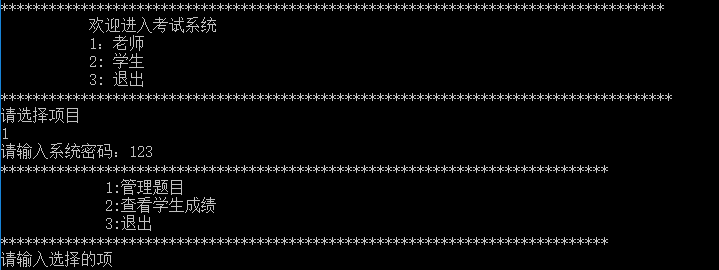
## 1.3制定编程计划

### 1.3.1 时间计划和小组成员任务分配

在小组规定的时间内，对每个成员分配每天的任务，详细说明见小组项目开发计划表。

### 1.3.2整个程序的菜单部分

本程序中的菜单部分由原来的2个菜单现细分为3个菜单





### 1.3.3设计整个程序的大纲

1.头文件

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

2.结构体的全局变量

struct question

{ …….

struct question\* next;

};

struct question\* phead;

3.主函数

int main()

{

void initquestion();

void showmainmenu();

………

initquestion();

showmainmenu();

return 0;

}

4.老师管理菜单中函数调用

void show1menu()

{

void addquestion();

……….. //函数声明

int choose=0;

while(choose!=6)

{

printf(" 1:新增题目\n");

……

printf("请输入选择的项\n");

scanf("%d",&choose);

switch(choose)

{

case 1: addquestion()

break;

………….

case 6:

break;

default:printf("enter dada error\n");

}

}

}

## 1.4小组编程

## 1.5 小组审核整个代码和优化代码

小组完成整个程序后，进行了代码的调试和优化，在优化的过程中，1.将原来的2个菜单细化为3个菜单，分为主界面，老师界面和老师管理题目3个界面，其中在老师界面中添加了查看学生成绩功能。 2在保存题目函数中将a+改为w+，这样的好处是避免第二次添加题目时，第一次的题目被第二次的题目覆盖。3.如果编题的数目很大，这样编题很费力，后采用了文件初始化到链表中，这样可以在网上拷贝题目到记事本中，如果想自己添题，改题，删题，还是照样可以完成。

# 2.设计原理

## 2.1模块化设计

每个函数占用一个模块，每个函数又可以相互调用，就是每个模块可以被其他模块调用。

注意：函数可以嵌套调用，绝对不可以嵌套定义。每个函数都是平等的。

利用函数可以相互调用的优势，这样层层调用，嵌套调用后就是整个程序连续在一起。形成一个可以实现一个特定功能的具有整体性的代码。下一节从结构图可以看出函数是怎样相互调用的。

## 2.2 设计原理结构图

此代码主要涉及到如下函数：

主函数:void main()

初始化文件题目到链表中的函数：void initquestion()

显示主菜单函数：void showmainmenu()

显示二级主菜单函数：void showmenu()

显示二级菜单第二个菜单函数：void show2menu()

显示二级主菜单下的三级主菜单函数：void show1menu()

显示二级菜单第二个菜单下的三级第二个菜单函数：void findscore()

增加题目函数：void addquestion()

删除题目函数：void deletquestion()

修改题目函数：void changequestion()

查找题目函数：void findquestion()

保存题目函数：void savequestion()

开始考试函数：void text()

**设计原理结构图**

initquestion() addquestion()

deletequestion()

changequestion()

show1menu() findquestion()

Main() savequestion()

showmenu()

showmainmenu() findscore()

show2menu() text()

# 3.算法分析

## 3.1 选择结构的使用

C语言有两种选择语句：（1）if语句用来实现两个分支大的选择结构；（2）switch语句用来实现多分支的选择结构。

If语句有三中基本形式，在本程序中用到if语句的两种形式，

第一种if(表达式) 语句1（没有else的部分）；

if(fp==NULL)

{

printf("文件不存在！");

return ;

}

第二种if(表达式) 语句1 else 语句2；

if(p->next->num==i)

{

ptemp=phead->next;

p->next=ptemp->next;

}

else

{

……

}

Switch语句实现多分支选择结构，它的一般形式是

Switch（表达式）

{

Case 常量1：语句1;break；

………

Case 常量n: 语句n;break;

default: 语句n+1;

}

说明：(1)switch后面的表达式其值的类型为整型（包括字符型）

(2)switch下面花括号内的复合语句包涵多个以关键字case开头的语句行和最多一个default开头的行。

(3)一般情况下，在执行一个case语句后，应当用break语句使流程跳出 switch结构。

在本程序中，在菜单部分多次用到switch语句

switch(choose)

{

case 1: addquestion();

break;

case 2: deletquestion();

break;

…………..

case 5: savequestion();

break;

case 6:

break;

default:printf("enter dada error\n");

}

## 3.2 循环结构的使用

循环结构语句有三种：包括while语句，do{..}while();语句，for语句。

While语句的一般形式是：while（表达式）语句；注意：while循环的特点是：先判断条件表达式，后执行循环语句。

在本程序中在循环输出菜单时 使用：

while(choose!=3)

{

………..

printf(" 1:管理题目\n");

printf(" 2:查看学生成绩\n");

printf(" 3:退出\n");

………..

}

do{…}while();语句的执行过程是：先执行循环体，然后在检查条件是否成立，若成立，再执行循环体。这一点与while语句循环不同，

在本程序中的添加题目函数中用到这类循环语句：

do{

……

printf("\n是否继续添加题目y/n\n");

……..

scanf("%c",&ch);

}while(ch=='y');

Fo循环语句的一般表达式：

for(表达式;表达式2；表达式3) 语句

表达式1：设置初始条件，只执行一次。可以为零个，一个或多个变量设置初值。

表达式2：是循环条件表达式，用来判断是否继续循环。在每次执行循环体前先执行此表达式，决定是否继续执行循环。

表达式3：作为循环变量增值，它是在执行完循环体后才执行的

For语句与while语句无条件等价。

在本程序中在学生考试函数中，判断题目个数时用到：

for(;id>0;id--)

{

…….

}

## 3.3 函数的调用

调用函数，每一个函数用来实现一个特定的功能，这就是模块化设计。函数调用一定要先定义和声明再调用，从用户使用的角度，函数可分为两种，一种是库函数（系统提供），第二种是用户自己定义的函数。从函数的形式上来看，也分为两种：第一种是无参函数，无参函数可以带回或不带回函数值，但一般以不带回函数值居多。第二种是有参函数，一般情况下会得到一个返回值。

函数的实参可以是 整型，字符，数组元素，数组名（多维数组名），指针，函数。函数的形参可以是整型，字符，数组 ，数组名（多维数组名），指针，函数。

在本程序中void addquestion();

void deletquestion();

void changequestion();

void findquestion();

void savequestion();

这些都是函数，每一个函数都有自己功能。

## 3.4 指针的定义和使用

什么是指针？指针就是一个地址，

指针的使用使程序变得实用，效率高，在本程序中用到了结构体指针和文件指针。

struct question

{

……..

struct question\* next;

};

struct question\* phead;

结构体的成员包括一个指针类型的成员struct question\* next来存放下一个节点的地址。一个指针类型的成员既可以指向其他类型的结构体数据，也可以指向自己所在的结构体类型的数据 。struct question\* phead; 是一个指向结构体的头指针，它存放一个地址，该地址指向一个元素。

3.4 结构体的定义和使用

C语言允许用户自己建立由不同类型数据组成的组合型的数据结构，称为结构体。声明一个结构体类型的一般形式是

Struct 结构体名

{成员列表}；（后面的；一定不要忘记。）

本程序中的结构体如下，定义一个question类型的结构体,成员有

题目序号整型num,题干字符串quest[300],选项字符串a[300],答案字符串dault[10],还有一个成员是指针类型的成员，用来存放下一个结点的地址。

struct question

{ int num;

char quest[300];

char a[300],b[300],c[300],d[300];

char dault[10];

struct question\* phead;

};

## 3.5 链表

1：链表是一种常见的重要的数据结构。它是动态的进行存储分配的一种 结构。链表有一个“头指针”变量，在本程序中struct question\* phead是一个头指针，它指向的是一个结构体，它存放一个地址指向一个元素，链表中的每一个元素称为“结点”，每个结点包括两个部分:1)用户需要的实际数据，2)下一个结点的地址 。head指向第一个元素，第一个元素有指向第二元素….直到最后一个元素，该元素不在指向其他元素，它称为“表尾”，它的地址部分放在一个“null”（表示“空地址”），链表到此结束。

2：链表中的各元素在内存中的地址可以是不连续的。要找到某一元素，必须先找到上一个元素，根据它提供的下一个元素地址才能找到下一个元素。如果不提供头指针，则整个链表无法访问。

3：链表这种结构，必须利用指针变量才能实现，即一个结点应包涵一个指针变量 ，用它存放下一个结点的地址，在本程序中struct question\*next是一个结构体类型的指针变量 ，用来存放下一个结点的地址。Next是struct question类型的一个成员，它又指向 struct question类型数据。用这种方法就可以建立链表。

4.链表的形式有两种，一种是静态链表，一种是动态链表。本程序中使用的是动态链表，

struct question\* p=phead;

struct question\* pNew;

do{

while(p->next!=NULL)

{

p=p->next;

}

pNew=(struct question\*)malloc(sizeof(struct question));

printf("请输入题目编号：");

scanf("%d",&pNew->num);

……

pNew->next=NULL;

p->next=pNew;

…..

} while(ch=='y');

## 3.6 文件的读取

文件的类型包括两种，一种是**程序文件**，一种是**数据文件**。

**C语言中主要讲的是数据文件。**

文件要有一个唯一的文件标示，以便用户识别和引用。文件标识包括三个部分：**文件路径，文件名主干，文件后缀。**

文件的分类包括：ASCII文件和二进制文件。

**打开和关闭文件**：用到fopen函数打开数据文件，形式如fopen（”a1”,”r”），表示要打开名字为“a1”的文件，使用文件方式为读入。在使用文件的方式中有很多种，如“r”“w”“a”“rb”“wb”“ab”“r+”“w+”“a+”“rd+”等，不同的方式，有不同的含义，即打开文件的方式不一样.

在本程序中，在学生考试的函数和老师查询成绩的函数中用到fp=fopen("a.txt","r")，是以读的方式打开文件。在文件初始化到链表时用到 FILE \*fp=fopen("a.txt","r+")，是读写的方式打开一个文件 。

用fclose函数关闭数据文件，形式是fclose（文件指针）。在本程序中用到fclose(fp)，关闭文件。

**顺序读写数据文件：**用fgets和fputs读对文本文件读入或输出一个字符，fgetcfp)是从fp指向的文件读入一个字符，fput（ch,fp）,是把字符ch写到文件指针变量fp所指向的文件中。用fgets和fputs函数向文件读写一个字符串，fgets(str,n,fp)是从fp所指的文件读入一个长度为（n-1）的字符串，存放到字符数组str中。fputs(str,fp)是把str所指的字符串写到文件指针变量fp所指向的文件中。在本程序中在文件初始化到链表的函数和学生开始考试的函数中用到

fgets(pnew->quest,300,(FILE\*)fp); fgets(buff,1000,(FILE\*)fp);

本程序中未用到格式化的方式读写文件和用二进制的方式向文件读写一组数据，格式化的方式是fprintf(文件指针，格式字符串，输出列表) 和fscanf(文件指针，格式字符串，输入列表)；二进制方式向文件读写一组数据是

fread(buffer,size,count,f和fwriite(buffer,size,count,fp) , buffer是地址，size是要读多少字节数，count是要读写多少个数据项，fp是FILE类型指针。

随机读写数据文件：

文件位置标记及其定位：1）用rewind函数式文件位置标记指向文件开头，rewind函数的作用是使文件位置标记指向文件开头，此函数没有返回值。2）用fseek函数改变文件位置标记fseek（文件类型指针，移位量）3）用ftell函数测定标记的当前位置ftell(fp).

文件读写出错检测：用ferror函数检查，形式是ferror(fp).