技术文档

目录

[1.链表 1](#_Toc504287180)

[1.1在后增添题目函数 1](#_Toc504287181)

[1.2删除题目函数 2](#_Toc504287182)

[1.3修改题目函数 3](#_Toc504287183)

[1.4查看题目函数 4](#_Toc504287184)

[1.5保存题目函数 4](#_Toc504287185)

[2.结构体数组，结构体指针 5](#_Toc504287186)

[3.对文件的使用 6](#_Toc504287187)

[4.使用switch 7](#_Toc504287188)

[5.循环的使用 8](#_Toc504287189)

# 1.链表

1.1在后增添题目函数(使用链表)

在建立链表时经常使用动态的方式建立链表，在程序执行过程中从无到有地建立起一个链表，一个个开辟结点并连接起来。

1.当链表不存在时

1）首先为链表分配一个结点大小的区间，作为链表的头结点，头结点数据区为系统默认值，next指针为NULL.

2）添加第一个结点时，动态分配一个结点内存，将数据存到结点数据区，结点的next指针为NULL,将头结点地next指针指向添加的结点。

quest

NULL

head

next

2.当链表已经存在

1）动态分配一个结点内存，将数据保存到该结点的数据区，指针遍历到最后一个结点，在后追加新结点。类似如图：

quest2

NULL

quest1

4569

head

1249

quest3

NULL

quest2

1472

quest1

4569

P

1.2删除题目函数（使用链表）

删除链表结点有两种情况

1.删除头指针

1）定义指针p指向链表头指针，将p->next赋给指针q

2）再释放p所指向的区域

quest2

NULL

quest1

4569

head

1249

P q=p->next

quest2

NULL

quest1

4569

head

1249

P q=p->next

2.删除结点

1）遍历到要删除的结点quest2

2）将quest2->next赋值给quest2前面的一个结点quest1->next

3）释放quest2所指向的区域

quest3

4568

quest2

1234

quest1

4569

quest3

4568

quest2

1234

quest1

4569

quest3

4568

quest2

1234

quest1

4569

注意：无论删除哪个结点都必须先连接后断开

1.3修改题目函数(使用链表)

1）遍历到要修改的结点quest2

quest3

4568

quest2

1234

quest1

4569

2）将要修改的内容放入新节点pTemp中，使得quest1->next指向pTemp，pTemp->next指向quest3.

quest2

1234

quest3

4568

quest1

4569

PTemp

1234

1.4查看题目函数(使用链表)

1.遍历链表

首先知道链表头指针地址，然后设一个指针变量，先指向第一个结点，然后依次后移，直到指向结尾处。

p p

quest2

NULL

quest1

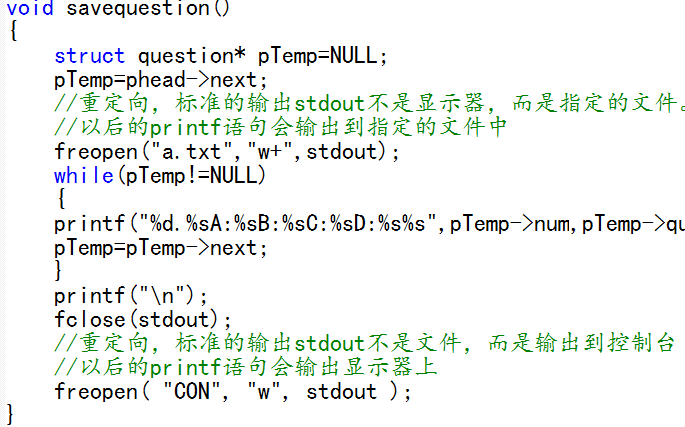
4569

head

1249

1.5保存题目函数(使用文件知识)

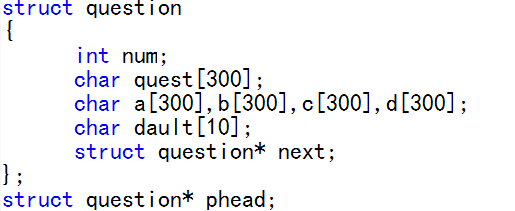
建立新文件，将要保存的内容放入新文件中。



# 2.结构体数组，结构体指针

使用结构体数组便于分配内存而且结构体相当于变量打包，当处理的东西多的时候，容易查找。

定义结构体指针用于构建链表,该结构体类型定义出来是为是表示链表中的每一个结点的。



# 3.对文件的使用

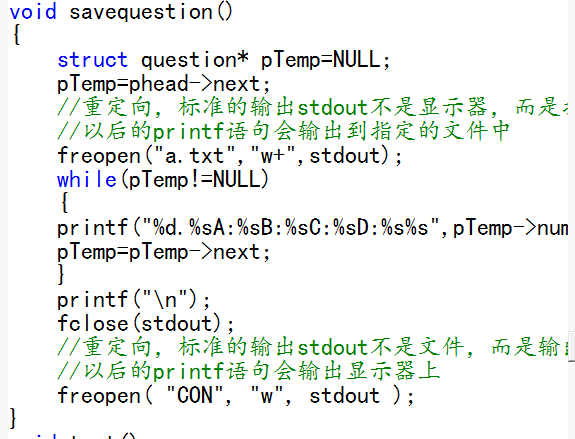
1）打开文件（用fopen函数实现）

fopen函数的调用方式为：fopen(文件名，使用文件方式);

2）对文件进行读或写

3）关闭文件（用fclose函数实现）

fopen函数的调用方式为：fclose(文件指针)；



# 4.使用switch

一般形式如下：

switch(表达式)

{

case 常量1：语句1

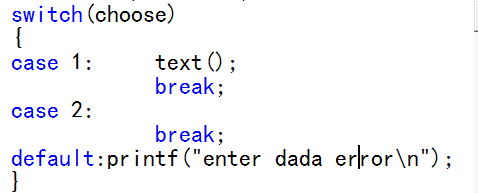
case 常量2：语句2

……

case 常量n：语句n

default ： 语句n+1

}



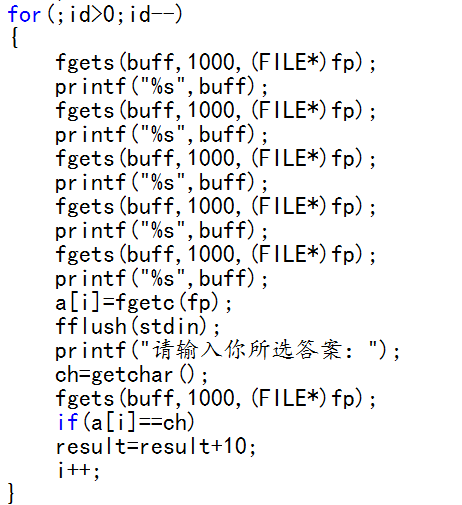
# 5.循环的使用

1）for循环

一般形式如下：

for(表达式1；表达式2；表达式3)

语句



2）while循环

While 表达式2

{

语句

表达式3

}

3）do…while循环

do

{

……

}while()；