# 第十章 构造数据类型及其应用

本章学习目标：

* 掌握枚举类型、结构体类型、联合体类型的定义
* 了解枚举类型、结构体类型、联合体类型及其变量的区别
* 了解枚举类型、结构体类型、联合体类型的使用背景
* 掌握枚举成员的编号规则
* 掌握结构体、联合体在内存中的存储规则
* 掌握结构体数组的使用。
* 掌握链表的操作细节。

## 11.1 实践题

**一、链表的构造及遍历**

**实验目的**

1. 理解链表的概念。
2. 掌握链表的创建。
3. 掌握链表的遍历及其元素访问方法。

**实验步骤**

步骤1：新建工程，命名listDemo;

步骤2：创建一头文件，命名list.h;在该文件中设计结点结构：

#include<stdio.h>

struct Node

{

int data;

struct Node \* next;

};

步骤3：创建一源文件，命名list.c;在该文件中编写测试代码：

#include "list.h"

int main()

{

printf("test\n");

return 0;

}

步骤4：在测试代码中创建结点、链接结点之后，链表创建成功。代码如下：

struct Node x,y,z,m,n;

struct Node \*head;

x.data = 5;

y.data = 4;

z.data = 3;

m.data = 2;

n.data = 1;

x.next = &y;

y.next = &z;

z.next = &m;

m.next = &n;

n.next = 0;//0为结束标识

head = &x; //链表创建成功，head为头指针，该链表的第一个结点为x（值

//为5），然后依次为y，z，m，n。

步骤4：对链表结点进行输出式访问，代码如下：

while(head!=0)//将链表结点数据输出

{

printf("%3d",head->data);

head = head->next;

}

**实验结果/结论**

1. **实验结果**



2. **实验结论**

* 链表的本质就是结点通过指针（链）来维系其前后关系，通过一个指针即可遍历整个链表
* 头指针保存的是第一元素结点的地址（指向第一个元素结点），其可以作为整个链表的标识，一般在操作链表时会重新定义指针，而不应该直接使用头指针。
* 该例中的链表结点是在栈区创建的，每个结点都有名字，如果链表结点是在堆区创建，则每个结点是没有名字的，此时，每个结点都是通过其前一个结点来定位，头指针是链表的唯一标识。
* 本例链表中的结点均是在栈区创建的,读者可以在试着在堆去创建链表:编写一函数,该函数有一个整形参数n，用以创建n个结点的链表，并将第一个结点的地址作为返回值返回。

**二、链表结点的插入、删除**

**实验目的**

1.掌握链表结点的插入过程。

2.掌握链表结点的删除过程。

3.理解链表结点（数据元素）之间关系的维系依据。

**实验步骤**

步骤1：在实验一代码中，创建一结点：struct Node o;

步骤2：若将该结点的数据赋值为7，将之插在4、3之间，则需要如下代码：

o.data = 7;

o.next =&z;

y.next = &o;

步骤3：运行测试代码，可以得出输出结果，如图所示：



步骤4：在遍历代码之前加代码：y.next = &z;

步骤5：运行测试代码，可以得出输出结果，如图所示：



**实验结果/结论**

1. **实验结论**

* 在进行链表结点插入时，若是结点没有名字（在堆区创建的结点），切记指针的修改顺序。
* 在进行链表结点删除时，若是结点没有名字（在堆区创建的结点），有两点特别要注意，第一，要从头开始找，知道找到被删除结点的前一个结点；第二，要用一个新指针将被删除结点的地址保存起来，这样可以在删除后释放该结点。

**三、 结构体数组使用**

**实验目的**

1. 掌握结构体数组的定义方法
2. 掌握结构体数组的使用
3. 理解结构体数组与普通数组的相同本质

**实验步骤**

问题：有十个学生信息存储在计算机中，找出考试成绩最高的学生将其信息输出出来，假设学生信息包括整型的学号、字符数组型姓名、浮点型成绩。

步骤1：新建工程，命名structArrDemo;

步骤2：创建一头文件，命名structArr.h;在该文件中设计结点结构：

#include<stdio.h>

struct student

{

int NO;

char name[10];

float score;

};

步骤3：创建一源文件，命名structArr.c;在该文件中编写测试代码：

#include "structArr.h"

int main()

{

printf("test\n");

return 0;

}

步骤4：在测试代码中创建结构体数组并附初值，然后查找分数最高的学生。代码如下：

struct student stuArr[10]={{1,"赵一",93},{1,"钱二",52},

{1,"孙三",86},{1,"李四",78},{1,"周五",93},{1,"郑六",39},

{1,"冯七",88},{1,"陈八",99},{1,"楚九",77},{1,"魏十",69}};

int i = 0,index=0;

while(i<10)

{

if(stuArr[index].score<stuArr[i].score)

index = i;

i++;

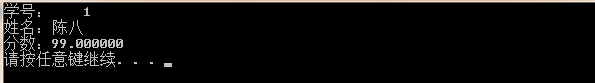
}

printf("学号：%5d\n姓名：%s\n分数：%f\n",

stuArr[index].NO,stuArr[index].name,stuArr[index].score);

**实验结果/结论**

1. **实验结果**



2. **实验结论**

* 结构体数组与普通数组没有本质区别，仅仅是数组的每个元素是结构体而已，当然操作数组元素就相当于操作结构体变量。
* 重做实验：对这十个学生信息按成绩进行排序（从低到高）。

## 11.2 理论题

### A类

**一、填空题**

1. 已知sizeof(int)的返回值为4, sizeof(int)的返回值为1。设有struct Test{char x; int y; char z;};，则sizeof(struct Test)的返回值为 。

**二、选择题**

**三、综合题**

### B类

**一、填空题**

**二、选择题**

**三、综合题**

## 本章答案

### A类

**一、填空题**

1.12，2.略

**二、选择题**

**三、综合题**

### B类

**一、填空题**

**二、选择题**

**三、综合题**