LECTURE 3 结构和指针

工学院18-19学年秋季学期计算概论（邓习峰班）课后辅导

讲师：陈婉雯

日期：2018/10/20

目录

- 课堂讲义讲解：
- 什么是结构
- 什么是指针

- 上周部分作业讲解与示例

结构体

- 自定义的数据格式，能够更加灵活、高效地表示数据
- 定义结构：描述对象由什么组成（没有创建实际对象）

```
struct stuff {
    int number;
    char code[4];
    float cost;
};
```

struct: 关键字
stuff: 标记（可选）

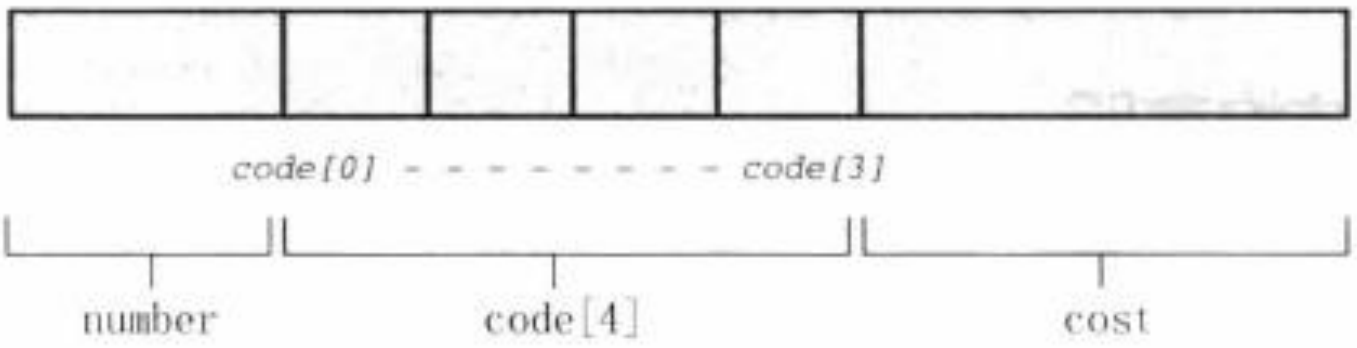

图 14.1 一个结构的内存分配

结构体

- 定义结构变量：创建结构体，分配存储空间
- struct stuff Alex; ← Alex 才是我们定义的结构变量的名称

- 初始化结构：
- 除了直接访问结构成员，还能够这样初始化：

```
struct stuff Alex = {
    20,
    abcd,
    4.5
};
```

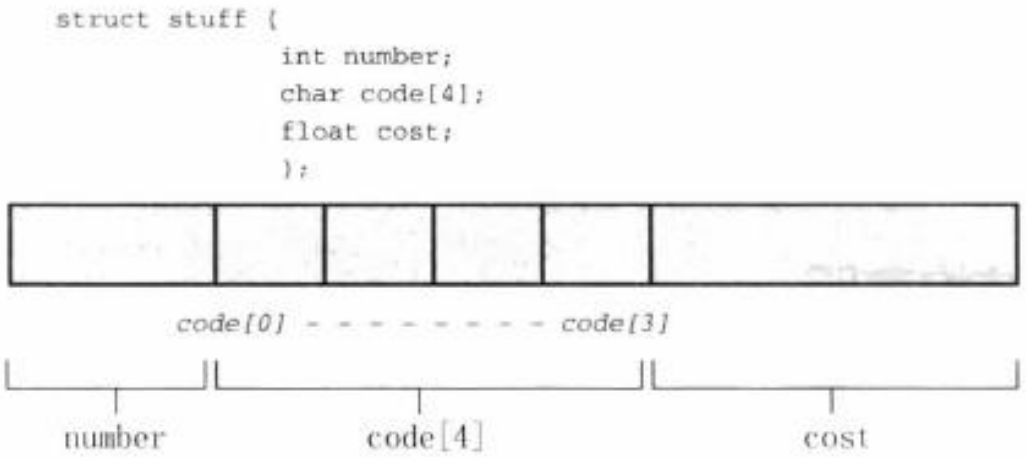

图 14.1 一个结构的内存分配

结构体

- 访问结构成员：
  - Alex.number
  - Alex.code

```
struct stuff {
        int number;
        char code[4];
        float cost;
        };
```

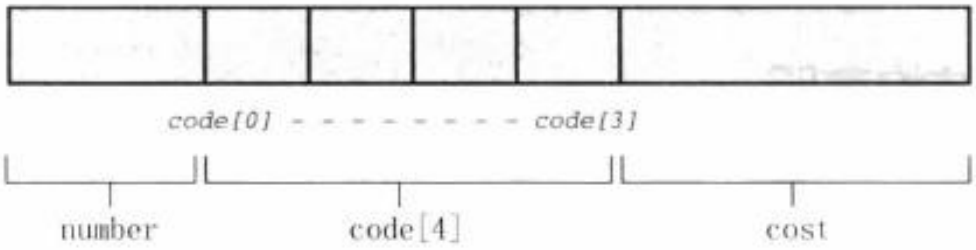

图 14.1 一个结构的内存分配

- 注意！假如创建的是结构体指针：
  - struct stuff *Alex;
  - Alex = (struct stuff *) malloc(sizeof(struct stuff));
- 访问方式变为：
  - (*Alex).number
  - Alex->number

(struct stuff *)：类型转换
malloc：分配内存空间
sizeof(struct stuff)：内存空间的大小和struct stuff相同

指针

- 指针：值为内存地址的变量
- 取地址符：& 求变量的内存地址，后面跟一个变量名
- 间接运算符：* 取内存地址内的值，后面跟一个地址/指针名

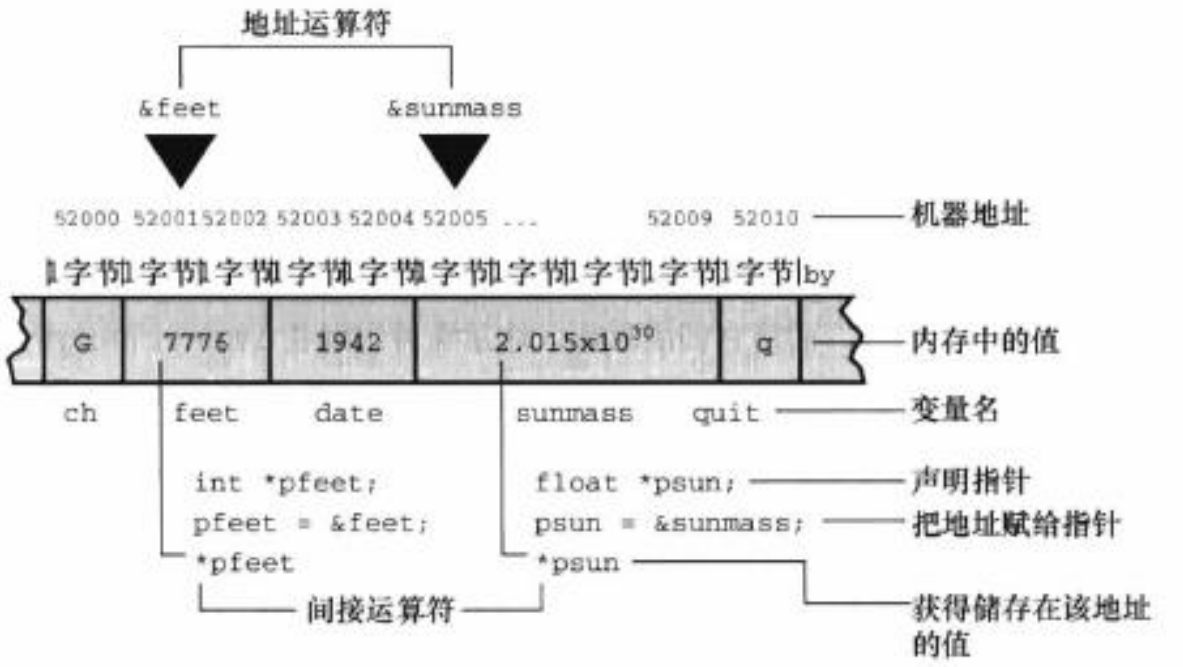

图 9.5 声明并使用指针

- 变量：名称和值
- 计算机编译和加载程序后：地址和值
- 地址是对我们隐藏的，需要用&才可访问

指针与数组

- 数组名：数组首元素的地址
- 指针加1：递增至它所指向类型的大小
- 数组的两种表示法：
  - int days[MONTHS];
  - 指针表示法：*(days+index)　　和days+index的区别？
  - 数组表示法：days[index]　　和&(days[index])的区别？

- 注意运算优先级：

```
*(dates + 2)   // dates 第 3 个元素的值
*dates + 2     // dates 第 1 个元素的值加 2
```

指针与函数参数

- 函数的参数：
- 形式参数：

```
void show_n_char(char ch, int num)
```

  - ch,num: 形式参数（形参），为局部变量
- 实际参数：

```
show_n_char(SPACE, 12);
```

  - SPACE, 12: 实际参数（实参）
  - 调用时会把实际参数的值拷贝到被调函数对应的形式参数中

- 为什么函数不能直接对实际参数进行操作？
- 为什么用指针就可以改变实际参数的值？

使用指针在函数间通信

```
void swapA(int a,int b){
    int tmp=a;
    a=b;
    b=tmp;
}
```

```
void swapB(int *a,int *b){
    int *tmp=a;
    a=b;
    b=tmp;
}
```

```
void swapC(int *a,int *b){
    int tmp=*a;
    *a=*b;
    *b=tmp;
}
```

参数到底是如何传递的？
如何理解：不管函数内外，内存地址统一编码，只要是操作相同的地址，则该地址处的值都将发生变化

使用数组作为函数参数

- 函数定义：
  - int sum(int *ar, int n)
  - int sum(int ar[], int n)

指针与内存分配

- int value;
  - 系统自动预留相应的内存空间存储value
- int *add;
  - 我们只知道add会是一个地址（无符号整型），但系统不会自动预留内存空间，add也没有指向特定的内存位置
  - 所以我们需要给add分配内存！

- malloc(size):
  - 找到合适的、大小为size字节的空闲内存块
  - 返回动态分配内存块的首字节地址
  - 要把这个地址赋给指针变量，通过指针访问
  - 要记得强制类型转换（为什么？）

```
double * ptd;
ptd = (double *) malloc(30 * sizeof(double));
```

malloc()和free()

- malloc()：分配内存给程序
- free()：把内存归还到内存池中，使得内存可以重复使用
  - free()里面的参数是指向由malloc()分配的内存的指针，不能释放其他方式分配的内存
  - 为什么要free()：内存池数量有限

- 另一个分配内存的函数calloc()：

```
long * newmem;
newmem = (long *)calloc(100, sizeof (long));
```

const和指针

- 主要作用：保护原始数据

```
double rates[5] = {88.99, 100.12, 59.45, 183.11, 340.5};
const double * pd = rates;        // pd指向数组的首元素
```

```
*pd = 29.89;        // 不允许
pd[2] = 222.22;     //不允许
rates[0] = 99.99; // 允许，因为rates未被const限定
```

const和指针

- 把const数据或者非const数据的地址初始化为const指针 √

```
double rates[5] = {88.99, 100.12, 59.45, 183.11, 340.5};
const double locked[4] = {0.08, 0.075, 0.0725, 0.07};
const double * pc = rates; // 有效
pc = locked;               //有效
pc = &rates[3];            //有效
```

- 把非const数据地址赋给普通指针 √

```
double rates[5] = {88.99, 100.12, 59.45, 183.11, 340.5};
const double locked[4] = {0.08, 0.075, 0.0725, 0.07};
double * pnc = rates;  // 有效
pnc = locked;          // 无效
pnc = &rates[3];       // 有效
```

函数指针

- 指向函数的指针：通常用作另外一个函数的参数

- 声明：函数返回类型和形参类型

```
void (*pf)(char *);     // pf 是一个指向函数的指针
```

```
//函数指针在数组中应用示例
#include<stdio.h>

int isPrime(int N);//判断是否质数
int isEven(int N); //判断是否偶数
```

函数声明

```
void printIF(int *beg,int *end,int (*pFun)(int)){
    while(beg!=end){
        if(pFun(*beg)==1)printf("%d\t",*beg);
        ++beg;
    }
}
unsigned int countIF(int *beg,int *end,int (*pFun)(int)){
    unsigned int count=0;
    while(beg!=end){
        if(pFun(*beg)==1)count++;
        ++beg;
    }
    return count;
}
```

```
int main(){
    int a[7]={1,23,23,12,34,67,99};

    printIF(a,a+7,isPrime);printf("\n");
    printIF(a,a+7,isEven);printf("\n");

    printf("%d",countIF(a,a+7,isPrime));printf("\n");
    printf("%d",countIF(a,a+7,isEven));printf("\n");

    return 0;
}
```

```
int isEven(int N){
    return N%2==0;
}
int isPrime(int N){
    if(N==0||N==1||N<0)return 0;//0、1和负数直接判为非质数
    int lastNum=N-1;
    for(int i=2;i<=lastNum;i++){
        if(N%i==0)return 0;
    }
    return 1;
}
```

上周作业

- 猜价格：

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int randNum(int Bottom,int Top){
    srand((unsigned int)(time(0)));
    return Bottom+rand()%(Top-Bottom+1);
}

int main(){
    int a=randNum(1,10),b;
    int i=0;
    scanf("%d",&b);
    while(a!=b){
        if(b>a){
            printf("Greater!\n");
            scanf("%d",&b);
            i++;
            continue;   }
        else if(b<a){
            printf("Less!\n");
            scanf("%d",&b);
            i++;
            continue;   }
        else if(i>a){
            printf("oh!My God!\n");
            break;  }
    }

    if(b==a){
        printf("OK!\n");
    }

    return 0;
}
```

上周作业

- 最大10个距离：