第八章-函数

函数的概述

概述：一个C程序由一个或多个函数组成，但是每一个C程序都有且仅有一个main函数。main函数是一个程序的组织者，所有其他的函数都是由main函数直接或间接的调用的，因此，一个C程序，总是从main函数开始，以main函数结束。

函数的定义

无参函数的定义

其一般格式为：

```
[类型说明符] 函数名(void){     // 函数头

    定义说明语句;
    执行语句;                  // 函数体

    [ruturn 语句];
}
```

说明：

（1） 类型说明符：限定返回值类型

（2） 函数名：用户定义的一个表示符

（3） 函数名后的括号：括号里面写传入函数的参数，无参函数写void

有参函数的定义

其一般格式为：

```
[类型说明符] 函数名(参数列表){     // 函数头

    定义说明语句;
    执行语句;                  // 函数体

    [ruturn 语句];
}
```

说明：相比无参函数，有参函数比其多了一个参数列表，参数列表就是定义这个函数时，刻的一个模板。以后调用这个参数的时候都要，给定这个模板对应的参数。模板就是，形式参数，简称形参。调用时传入的叫做实际参数，简称形参。

函数的调用

（1） 函数无返回值的调用语句

函数名（[实际参数表]）；

（2） 函数有返回值的调用语句

函数名（[实际参数表]）；

函数的声明

在C语言中，要使用自定义函数都要先进性声明，这就类似于变量的声明一样。其一般格式为：

类型说明符 函数名（形式参数表）；

函数的传值方式

在函数调用时，需要给定确定的参数以满足函数的需求，在这个过程中，只能是实参向形参传递，形参不能向实参传递参数。这样做，无论在函数中参数做如何变化吧，都不会影响到其本来的值，这就是值传递。

在函数的调用中，实参需要与形参相对应。顺序不能乱。

函数的嵌套调用和递归调用

函数的嵌套

说明：说白了，函数的嵌套就是，当前正在调用的函数又跑去调用了另外的函数，妈妈让你买点菜做饭，你去买了菜发现自己不会做饭，然后喊你爸爸做饭，这就是函数的嵌套。

函数的递归

说明：递归就是函数直接或间接的调用自身的情况。

比较难，暂且用一个例子填个坑吧：

```
#include <stdio.h>
```

```c
int main(void){
    int fac(int n);
    int n, f;

    printf("Please enter n:");
    scanf("%d", &n);
    if(n < 0){
        printf("Sorry you enter a wrong number!\n");
    }else{
        f = fac(n);
        printf("%d! = %dn", n, f);
    }
}

int fac(int n){
    int m;
    if(n == 0 || n == 1){
        m = 1;
    }else{
        m = n * fac(n - 1)
    }

    return m;
}
```

函数参数

严格意义来说，函数的参数可以是各种类型的变量，可以是基本数据类型的，也可以是数组，指针等类型的数据。

局部变量和全局变量

局部变量：定义在函数内部和符合语句内部的变量是局部变量

全局变量：定义在函数外部的变量叫做全局变量

以下内容初次学习，要认真哦

变量的存储类型

内存中可以供用户使用的存储空间分为：**程序区**和**数据区**，变量存储存储在数据区

内存中可以供用户使用的存储空间分为：**程序区**和**数据区**，变量存储存储在数据区内，数据区可以分为**动态存储区**和**静态存储区**。

在程序运行期间也不是所有的变量都会一直占用着内存，有些变量从始至终一直占用着内存，比如全局变量；但也有像局部变量并不会从始至终占用内存。

（1） 动态存储：是指在程序运行期间，根据实际需要动态的分配空间的方式。

（2） 静态存储：是指在程序运行期间给变量分配固定存储空间的方式。

所以，定义一个变量的完整形式是：

[存储类型] 数据类型 变量名；

举例：static float x,y;

变量的存储类型分为：**自动型、寄存器型、静态型和外部型**四种类型。

自动型变量存储在动态存储区内，寄存器型的自然是存在寄存器中，而静态型和外部型都存在静态存储区内。

自动型变量

说明：自动型变量用于定义局部变量，但是局部变量的默认形式就是自动型，因此自动型类型说明“auto”一般不常用。其定义变量的格式为：

auto int m, n;

寄存器型变量

寄存器型变量用register进行声明。它只能用于局部变量，如果局部变量没有赋值，那初值就是随机数，寄存器存取效率要高于内存，因此使用寄存器型变量就是为了加快效率的。

静态型变量

用关键字static声明。它既可以定义局部变量也可以定义全局变量，默认初始值为零或空字符。

静态局部变量：在整个程序运行期间，静态局部变量在内存中占据着永久的内存单元。即使退出函数，下次在进入函数的时候静态局部变量仍然使用原来的存储单元。静态局部变量不能被其他函数访问。

静态全局变量：便是该变量不能被本文件意外的程序调用。

外部型变量

外部型变量只能定义全局变量，且是全局变量的默认属性。

注：声明和定义的区别

int a; //这是定义变量，并分配了内存单元，可以使用

extern int a; 、、这是声明变量，没有分配内存单元，不可以使用。

extern的功能：

（1） 在同一源文件里使用extern来扩大全局变量的引用。

全局变量定义在后，变引用该变量的函数在前面，则应该使用extern在引用它的函数中进行声明。

代码示例：

```
#include <stdio.h>
int main(void){
    int max(int x,int y);
    extern int a, b;     //声明全局变量
    printf("max = %d\n", max(a,b));
}
int a = 12, b = 5;
int max(int x, int y){
    int z;
    z = x > y ? x : y;
    return z;
}
```

（2） 在不同的源文件中使用extern来扩大全局变量的作用域。

原则：一次定义，多次声明。

例如，在11.c文件中定义了一个变量 `int x = 10` ，这个时候想要在22.c文件中进行使用这个变量，那么就可以在22.c文件中使用 `extern int x;` 对其进行声明，然后使用。

值得注意的是，不能对extern声明的变量赋初值。

内部函数和外部函数

内部函数

用static表示： `static 类型说明符 函数名（参数列表）;` ，由于用static表示，所以内部函数也可以叫做静态函数。

内部函数的特点：只能被本文件中的其他函数所调用，作用域仅限于本文件。

外部函数

用extern表示： `extern 类型说明符 函数名（参数列表）;` 。函数的默认形式就是外

部函数。

外部函数的特点：不但可以被本文件中的函数调用，还可以被此文件以外的文件调用，只是调用之前需要先对该函数进行声明。