第四章 函数

1. 函数定义与函数声明

(1) 函数的四要素

由字母、数字、下划线组成，必须以字母或下划线开头

①函数的标识：函数名，遵循标识语法

②输入参数：形式参数(形参)

③处理结果：函数体

④返回结果：返回值

(2) 函数定义的一般形式

返回值类型 函数名([形式参数表]) ——→ 函数头

{ 语句序列 ——→ 函数体

}

[例] void printmessage( ) (void 无返回值；printmessage 函数名；( ) 无形参数值)

{ cout << "Hello!" << endl; } —— 无 return 语句

(3) 函数的返回值

① 函数的返回值通过函数中的 return 语句获得

· 一个函数中可以有一个以上的 return 语句

· 在函数被调用执行时，只要遇到一个 return 语句，立刻返回到调用程序

② 如果函数类型和 return 语句中表达式的值不一致，以函数类型为准.

③ 当一个函数没有返回值时，返回类型必须用 void 说明，这时函数体中可以没有 return 语句，也可以有.

2. 函数的调用

(1) 函数调用的一般形式：

①函数语句　printstar( ); —— 没有返回值

②函数表达式　int c = 2 * max(a,b);

③函数参数　m = max(a, sqrt(b));

△若用户定义的函数与调用它的函数在同一个程序单位中，且位置在主调函数之后，则必须对被调函数声明.

3. 函数声明

(1) 函数原型的一般形式

①函数类型 函数名(参数类型1,参数类型2...);

float add(float,float);

②函数类型 函数名(参数类型1 参数名1,参数类型2 参数名2...)

float add(float x,float y);

(2) 函数的声明和定义区别

函数定义	函数声明
返回值类型	返回值类型
函数名	函数名
形式参数表	形式参数表
函数体	

定义是对函数功能的确立---说明怎么做
声明通知编译系统函数的信息，以便在对包含函数调用的语句进行编译时，据此对其进行对照检查

函数只能在一个文件夹中定义一次，

却可以声明多次

(3) 函数的传值调用

```
#include <iostream>
using namespace std;
void Interchange(int, int );   //函数原型声明
int main()
{
        int m=10,n=20;
        cout<<"调用函数之前（实参）：  "<<"m="<<m<<",
n="<<n<<endl;
        Interchange (m,n);          //函数调用
        cout<<"调用函数之后（实参）：  "<<"m="<<m<<",
n="<<n<<endl;
        return 0;
}
void Interchange (int x ,int y)
{
        cout<<"交换之前（形参）：  "<<"x="<<x<<", y="<<y<<endl;
        int temp;
        temp=x;
        x=y;
        y=temp;
        cout<<"交换之后（形参）：  "<<"x="<<x<<", y="<<y<<endl;
}
```

调用函数之前(实参)：m=10, n=20

交换之前(形参)：x=10, y=20

交换之后(形参)：x=20, y=10

调用函数之后(实参)：m=10, n=20

(4) 有关形参与实参的说明

①定义函数时，必须在函数首部指定形参类型；

②实参可以是常量、变量或表达式

③实参与形参的类型必相同或赋值兼容.

④资源分配.

· 只有在发生函数调用时，形参才被分配内存单元

· 实参单元与形参单元是不同的单元

· 在调用结束后，形参所占的内存单元也被释放.

· 实参变量对形参变量的数据传递是"值传递"

(5) 默认值参数

① 指定默认值的参数必须放在形参表列中的最右端.

② 默认值可以是常量、全局变量或函数调用，但不能是局部变量

4. 函数重载

(1) 定义：在同一范围中声明几个功能类似的同名函数，这些函数的形式参数(参数的个数、类型)必须不同.

(2) 说明：重载函数形参的个数、类型和顺序(参数表列)三者中必须至少有一种不同.

5. 函数模版

```
#include <iostream>
using namespace std;
template<typename T> //模板声明，其中T为类型参数
T abs(T a)  //定义一个通用函数，用T作虚拟类型名
{
        if(a>=0) return a;
        else return -a;
}
int main( )
{
        int i=185,iResult;
        double d=56.87,dResult;    //dResult;
        long g=67854,gResult;      //gResult;
        iResult= abs (i);  //第13行，调用模板函数，此时T被int取代
        dResult= abs (d);  //第14行，调用模板函数，此时T被double取代
        gResult= abs (g);  //第15行，调用模板函数，此时T被long取代
        cout<<"abs(i)="<<iResult<<endl;
        cout<<"abs(d)="<<dResult<<endl;
        cout<<"abs(g)="<<gResult<<endl;
        return 0;
}
```

{ 返回值类型不同
| 形参类型不同.
{ 函数体相同.

6. 嵌套作用：

C++不允许对函数作嵌套定义，在一个函数中不能完整地包含另一个函数.

在一个程序中每一个函数的定义都是相互平行和独立的.

7. 递归函数：直接调用自己或借助其它函数间接调用自己的函数

要求：①递归形式(算法)

②递归条件(缩小问题规模)

③递归终止条件(基本情况)

[例] int Factorial (int n)

{ if (n==0) (终止条件) return 1;

else return n * Factorial(n-1); } ← 修改递归条件