西北工业大学
NORTHWESTERN POLYTECHNICAL UNIVERSITY

C程序设计

Programming in C

1011014

主讲：姜学锋，计算机学院

调用函数 - 原型与声明

1、函数声明和原型

2、库函数调用

4.3 函数原型与调用

- 当要调用函数时，C语言规定在调用一个函数之前必须有该函数的声明。

- 最好的函数声明形式就是函数原型。

4.3.1 函数声明和函数原型

- 1. 函数声明
- 编译器在编译函数调用时，需要检查函数接口，即返回类型、参数类型、参数次序、参数数目是否正确，这样就能避免参数类型、参数数目不一致而引发的错误。保证正确的函数调用栈。而编译器之所以能够发现这些错误，原因就在于它事先有了该函数的声明，进而知道函数接口是如何规定的。

4.3.1 函数声明和函数原型

- 一个函数只能定义一次，但是可以声明多次。
- 定义是函数实现，函数代码一经实现，就不能再来一次。但声明的作用是程序向编译器提供函数的接口信息，因而多次提供接口信息是允许的，但不能提供相互矛盾、语义不一致的接口信息。

4.3.1 函数声明和函数原型

- C语言规定函数定义语法既是函数定义，也是函数声明。换言之，只要函数调用是写在函数定义的后面，就自然有了函数声明。
- 但这种方式与C语言允许函数定义可放在任意位置的规定矛盾了，而且使用起来也不方便。显然，将函数调用均写在函数定义的后面不是现实的方法。

4.3.1 函数声明和函数原型

- 一般情况下，将函数声明放在头文件（.h）中，将函数实现放在源程序文件中。凡是要调用这个函数的地方，通过#include将头文件包含即可。

4.3.1 函数声明和函数原型

- 另一方面，C语言允许调用库函数，所谓库函数是指事先由程序员编制好的函数。

- 多数情况下，基于各种理由，如保护知识产权，这些库函数仅提供二进制形式的目标代码给调用者链接，却没有提供源码形式的函数定义。

- 这种情况下，又如何让调用者有函数声明呢？方法是使用函数原型。

4.3.1 函数声明和函数原型

- 2．函数原型
- 函数原型（function prototype）的作用是提供函数调用所必须的接口信息，使编译器能够检查函数调用中可能存在的问题，有两种形式：
- ①第一种形式：

```
返回类型 函数名(类型1 参数名1, 类型2 参数名2, ......);
```

- ②第二种形式：

```
返回类型 函数名(类型1, 类型2, ......);
```

4.3.1 函数声明和函数原型

- 显然第二种形式是第一种形式的简写，之所以在函数原型中可以不写参数名称，是因为参数名称不是形参与实参对应的依据，因而参数名称不是重要的接口信息，可以省略。
- 语法后面的分号（；）必须要写。

4.3.1 函数声明和函数原型

- 例如：

```
#include <math.h>
double sqrt(double x);
```

- 是标准库求平方根的函数原型，表示调用它需要：
- ①包含头文件math.h，因为sqrt函数原型在math.h中。
- ②sqrt函数须提供一个double型的实参，返回值也是double型。

4.3.1 函数声明和函数原型

【例4.3】

编写求两个数的最大公约数的函数。

4.3.1 函数声明和函数原型

例4.3

```
#include <stdio.h>
int gcd(int m, int n); //gcd函数原型，gcd函数声明在前
int main()
{
  int m,n;
  scanf("%d%d",&m,&n);
  printf("%d\n",gcd(m,n)); //调用时已有gcd函数声明
  return 0;
}
int gcd(int m, int n) //求最大公约数，gcd函数实现在后
{
  int r;
  while (n!=0) { //欧几里德算法(Euclidean algorithm)，原理是：
    r = m % n ; //r为m/n的余数
    m = n ; //则gcd(m,n)=gcd(n,r)=...
```

4.3.1 函数声明和函数原型

例4.3

```
16        n = r ; //r=0时n即是gcd
17    }
18    return m;
19 }
```

4.3.1 函数声明和函数原型

```
10 int gcd(int m, int n) //求最大公约数，gcd函数实现在后
11 {
12   int r;
13   while (n!=0) { //欧几里德算法(Euclidean algorithm)，原理是:
14     r = m % n ; //r为m/n的余数
15     m = n ; //则gcd(m,n)=gcd(n,r)=...
16     n = r ; //r=0时n即是gcd
17   }
18   return m;
19 }
```

第2行即是gcd函数的函数原型，第10行～第19行是gcd函数定义（函数实现）。

4.3.1 函数声明和函数原型

- 函数原型属于C语言的声明部分，因此，必须放在函数或语句块中所有执行语句的前面，或者函数外的全局范围内。

- 函数原型几乎就是函数定义中的函数头，但函数头后面不能有分号，而函数原型没有函数体。函数定义与函数原型是有区别的，函数定义具有函数原型的声明作用，但它还是函数功能的具体实现，所有函数定义是主体，函数原型像是它的“说明书”。

4.3.1 函数声明和函数原型

- 函数原型通常出现在函数定义的前面，也允许在函数定义的后面，只不过意义不大。

- 编译器在编译时，无论它们哪个在前，均以第一次“看到”的函数接口为准,，如果后面的与这个函数接口不一致，就会出现编译错误，所以函数原型要与函数定义匹配。

4.3.1 函数声明和函数原型

- 3．函数调用
- 有了函数声明，就可以调用函数，有参数函数调用的形式为：

```
函数名(实参列表);
```

- 实参可以是常量、变量、表达式和函数调用，各实参之间用逗号（，）分隔。实参的类型、次序、个数应与形参一致。

4.3.1 函数声明和函数原型

- 无参数函数调用的形式为：

```
函数名();
```

- 函数名后面的括号（）必须有，括号内不能有任何参数。

4.3.1 函数声明和函数原型

- 在C语言中，可以用以下几种方式调用函数。
- （1）函数表达式。
- 函数调用作为其中的一项出现在表达式中，以函数返回值参与表达式的运算，这种方式要求函数必须是有返回值的。例如：

```
z = max(x,y)
```

4.3.1 函数声明和函数原型

- （2）函数调用语句。
- 函数调用的语法形式加上分号就构成函数调用语句。例如：

```
printf("area=%lf",s);
```

- 如果函数没有返回值，则只能使用函数语句的方式调用，而有返回值的函数允许使用函数语句的方式调用，只不过函数的返回值被舍弃不用了。

4.3.1 函数声明和函数原型

- （3）函数实参
- 函数可以作为另一个函数调用的实参出现。这种情况是把该函数的返回值作为实参进行传递，因此要求该函数必须是有返回值的。例如：

```
printf("%d",max(x,y));
```

- 即是把max调用的返回值又作为printf函数的实参来使用的。

4.3.1 函数声明和函数原型

- 假设max(x,y)返回两个数的最大值，则：

```
max(max(a,b),max(c,d))
```

- 返回四个数的最大值。

4.3.1 函数声明和函数原型

- 前面述及，函数调用时实参的运算是有方向的，即函数调用对实参的计算是有求值顺序的。运算方向由不同的函数调用约定决定。

- C语言默认使用C调用约定，求值顺序是自右向左。

- 与此相反的是PASCAL调用约定，求值顺序是自左向右。C程序需要经过特别的设定才能是PASCAL调用约定。

4.3.1 函数声明和函数原型

- 例如：

```
int i=1,j=2;
printf("%d,%d,%d\n",i=i+j,j=j+i,i=i+j); //从右向左计算实参
```

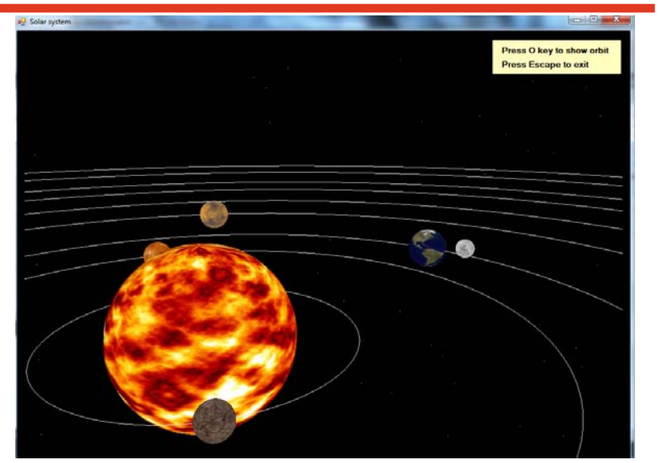
- 程序输出结果为：8,5,3。因为在调用printf函数时，先处理最右边的i=i+j，这个实参值是3，再处理中间的j=j+i，这个实参值是5，最后处理左边的i=i+j，这个实参值是8。

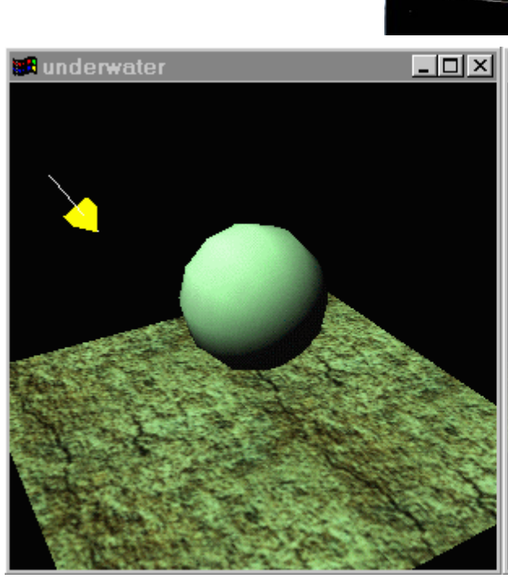
4.3.2 库函数的调用方法

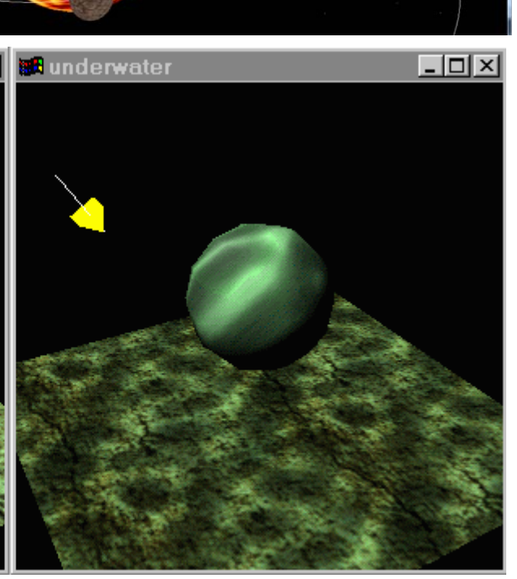
- C语言拥有庞大的系统库函数可以使用，既有标准库函数完成基本功能，又有专业库函数实现特定功能。

4.3.2 库函数的调用方法

- 图形库OpenGL
- https://www.opengl.org/

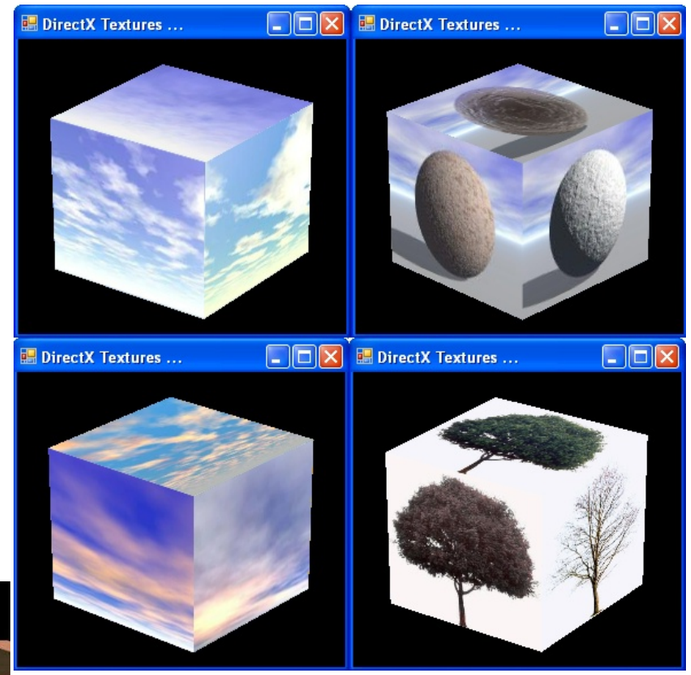


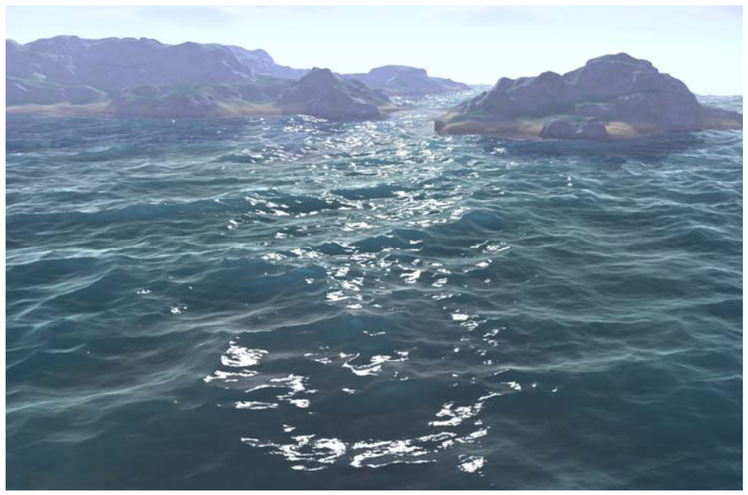




4.3.2 库函数的调用方法

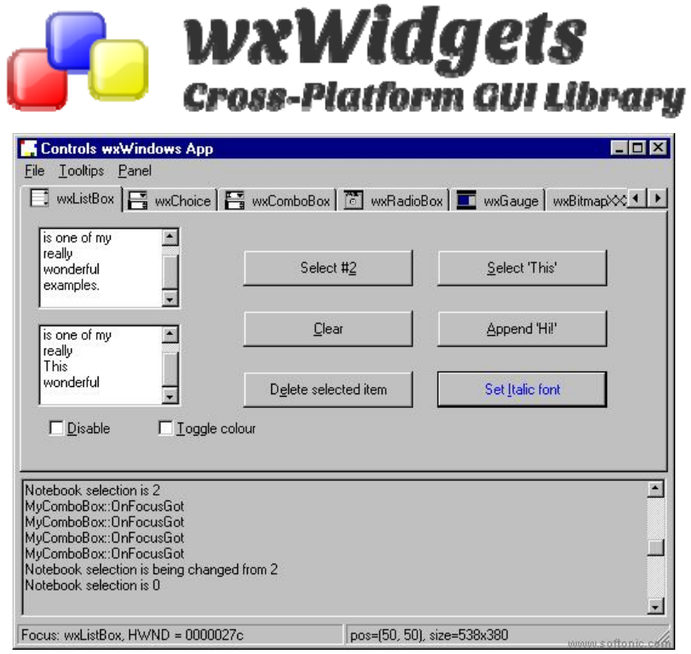
- 图形库DirectX

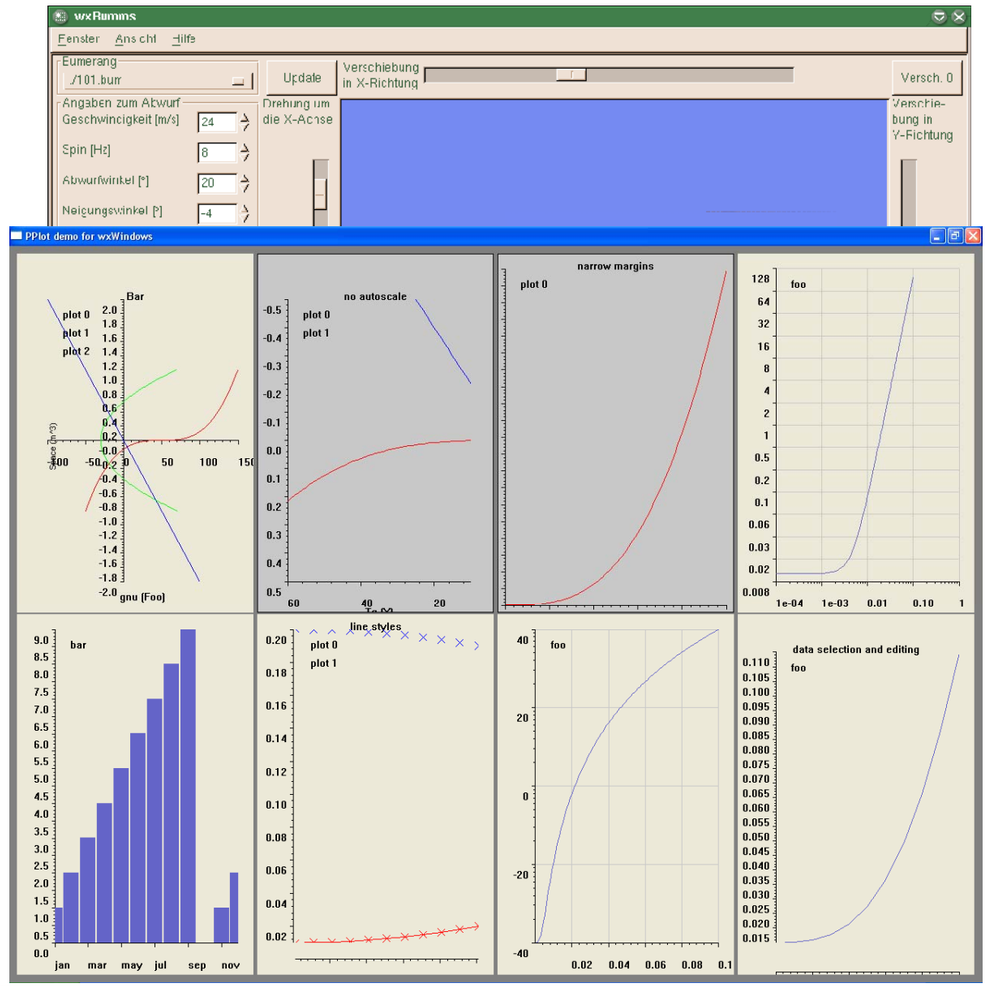




4.3.2 库函数的调用方法

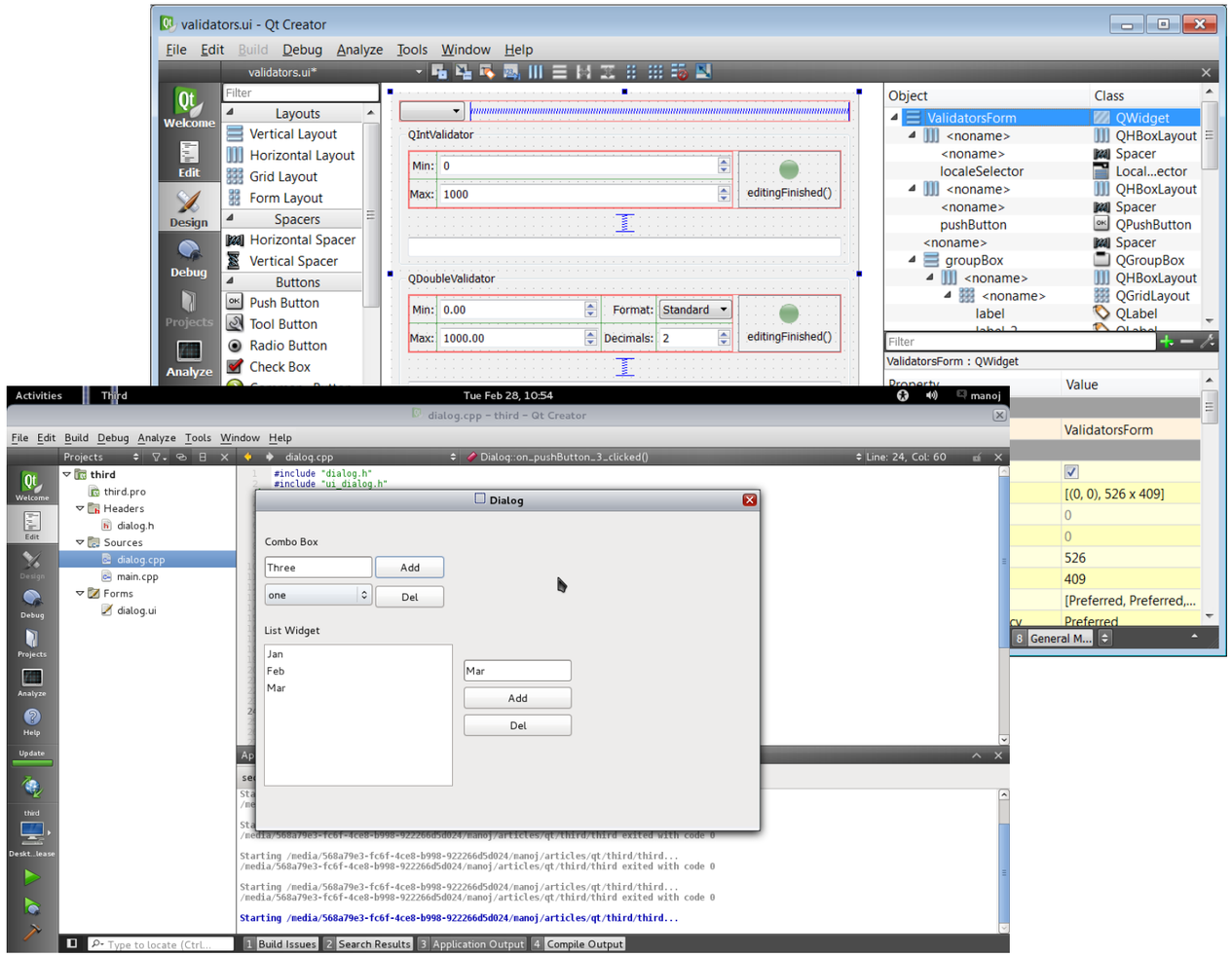
- 图形界面库wxWindows
- https://www.wxwidgets.org/





4.3.2 库函数的调用方法

- 图形界面库Qt
- http://www.qt.io/

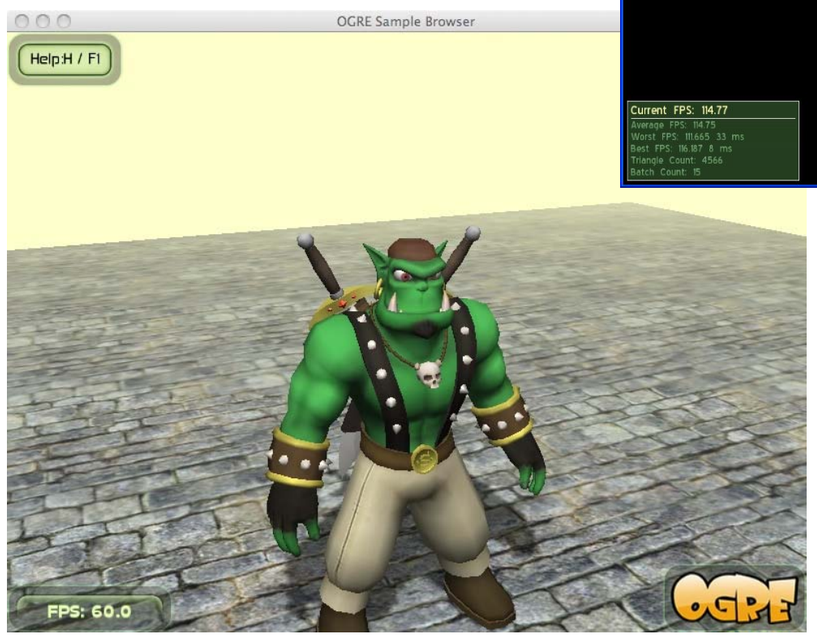


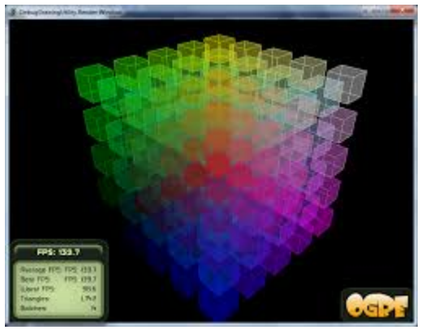
4.3.2 库函数的调用方法

- 多媒体库OpenAL
- https://www.openal.org/

4.3.2 库函数的调用方法

- 游戏开发库OGRE
- http://www.ogre3d.org/





4.3.2 库函数的调用方法

- 游戏开发库Allegro
- http://liballeg.org/

Allegro

4.3.2 库函数的调用方法

- 网络开发库Winsock，数据库开发库ODBC API，科学计算函数库GSL等。同时多数应用软件，例如Office、MATLAB、AutoCAD等均提供了C语言接口，使C/C++通过混合编程用到这些软件的特色功能。

4.3.2 库函数的调用方法

- 无论使用库函数或是混合编程，对于C/C++程序来说本质上就是在使用函数。这里给出库函数调用的一般方法。

4.3.2 库函数的调用方法

- （1）在程序中添加库函数声明
- 多数库函数将自己的函数原型和特殊数据等放在头文件（.h）中，所以应首先使用文件包含命令将这些头文件包含到程序中。例如欲使用数学库函数，文件包含命令为：

```
#include <math.h>
```

4.3.2 库函数的调用方法

- 从而使得程序有函数声明，例如：

```
y=sin(x); //求x（弧度）的正弦
```

- 调用就能够通过编译。

4.3.2 库函数的调用方法

- （2）将库函数目标代码连接到程序中。
- 在连接时，例如使用了sin函数，就必须要有sin函数的实现代码才能生成可执行文件，否则连接出错。要将库函数的目标代码能够连接到程序中，主要是配置好开发环境的相关参数，然后由连接器处理。

4.3.2 库函数的调用方法

- 标准库函数的连接在开发环境中是默认的，一般可以不用特别设置。
- 经过上述两个步骤，可以让程序调用库函数了。但要让库函数发挥作用，实现期望的功能，还需要通过库函数详尽的使用手册了解：
- ①函数的作用、功能、调用参数要求等，例如sin函数要求调用参数是弧度值。
- ②函数的调用约定，确保正确地参数传递和函数返回。


程序设计