C++与C的主要差异
– 带默认形参值的函数

- 在调用函数时，需要针对函数中的每一个形参给出对应的实参。C++中也允许在函数定义或函数声明时给出默认的形参值。在调用函数时，对于有默认值的形参，如果没有给出相应的实参，则函数会自动使用默认形参值；如果给出相应的实参，则函数会优先使用传入的实参值。
- 指定默认形参值的位置：默认形参值可以在两个位置指定：如果有函数声明，则应在函数声明处指定；否则，直接在函数定义中指定。

【例1-5】默认形参值使用方法示例。

```
#include <iostream>
using namespace std;
void f(char *str="abc");            //默认值在函数声明处指定
int main()
{
    f();        // 没有传入实参，此时形参str使用默认值"abc"，执行后输出“abc”
    f("def");// 传入实参，此时形参str的值为"def"，执行后输出“def”
    return 0;
}
void f(char *str)                   //此处不再给出默认值
{
    cout<<str<<endl;
}
```

- 提示：
- 对于有默认值的形参，如果在调用函数时给出了相应的实参，则会优先使用传入的实参值。如执行“f("def");”会输出def。
- 如果有函数声明，则应在函数声明中给出默认形参值，函数定义中不要重复。比如：
- void f(char *str = "abc");　　// 函数声明部分
- void f(char *str = "abc")　　// 函数定义部分。错误：默认形参值被重复指定
- { … }
- 默认形参值可以是全局常量、全局变量，甚至是可以通过函数调用给出，但不能是局部变量。因为形参默认值或其获取方式需在编译时确定，而局部变量在内存中的位置在编译时无法确定。

- 默认形参值的指定顺序
- 默认形参值必须严格按照从右至左的顺序进行指定。比如：
- void f(int a=1, int b, int c=3, int d=4);
- 这种指定默认形参值的写法有误。这是由于在第2个参数b未指定默认形参值的情况下，给出了第1个参数a的默认形参值，不符合从右至左的指定顺序。

【例1-6】默认形参值的指定顺序示例。

```
#include <iostream>
using namespace std;
void f(int a, int b=2, int c=3, int d=4)          // 带默认形参值的函数定义
{
    cout<<a<<" "<<b<<" "<<c<<" "<<d<<endl;
}
int main()
{
    f(1);                    // 输出1 2 3 4
    f(5, 10);                // 输出5 10 3 4
    f(11, 12, 13);           // 输出11 12 13 4
    f(20, 30, 40, 50);       // 输出20 30 40 50
    // f();                  // 错误
    return 0;
}
```