

# 【C++】C++中函数重载的理解

原创

忽晴忽雨江湖

2018-06-19

版权

17:37:05



41971

★ 收藏 115

我们在平时写代码中会用到几个函数但是他们的实现功能相同，但是有些细节却不同。例如：交换两个数的值其中包括（int, float, char, double)这些个类型。在C语言中我们是利用不同的函数名来加以区分。

```
1 void Swap1(int* a, int* b);
2 void Swap2(float* a, float* b);
3 void Swap3(char* a, char* b);
4
void Swap4(double* a, double* b);
```

我们可以看出这样的代码不美观而且给程序猿也带来了很多的不便。于是在C++中人们提出了用一个函数名定义多个函数，也就是所谓的函数重载。

## 一.函数重载定义

**函数重载**是一种特殊情况，C++允许在**同一作用域中声明几个类似的同名函数**，这些同名函数的形参列表（参数个数，类型，顺序）必须不同，常用来处理实现功能类似数据类型不同的问题。

在C++中不仅函数可以重载，运算符也可以重载。例如：

运算符<<,>>。既可以做移位运算符，也可以做输出，输入运算符。

**注意：重载函数的参数个数，参数类型或参数顺序三者中必须有一个不同**

```
1 #include<Windows.h>
2 #include<iostream>
3 using namespace std;
4
5 int Add(int a, int b)
6 {
7
8     return a + b;
9
10 }
11
12 double Add(double a, double b)
13 {
14
15     return a + b;
16 }
17
18 float Add(float a, float b)
```

```
19 | {
    20 |
21 |     return a + b;
22 |
23 | }
24 | int main()
25 | {
26 |
27 |     cout<<Add(1,2)<<endl;
28 |
29 |     cout<<Add(3.5, 4.5)<<endl;
30 |
31 |
    cout << Add(2.22, 3.33) << endl;
32 | 33 |
34 |     system("pause");
35 |     return 0;
36 | }
```

我们可以看到定义了一个Add函数来求三个不同类型数的和，在调用过程中系统会自动根据其实际参的类型不同来实现准确调用。

```
1 | #include<iostream>
2 | #include<Windows.h>
3 | using namespace std;
4 |
5 | int main()
6 | {
7 |
    int max(int a, int b, int c);
8 |     int max(int a, int b); 9 |
    int a = 10;10 |     int b = 20;
11 |     int c = 30;
12 |
13 |
    cout << max(a, b, c) << endl;
14 |     cout << max(a, b) << endl;
15 |     system("pause");16 |
    return 0;17 | }
18 |
19 |
20 | int max(int a, int b, int c)
21 | {
22 |     if (b > a)
23 |         a = b;
24 |     if (c > a)
25 |         a = c;
26 |     return a;
27 |
28 | }
29 |
30 | int max(int a, int b)
31 | {
32 |
33 |     return (a > b) ? a : b;
34 | }
```

从上边代码可以看出函数重载除了允许函数类型不同以外，还允许参数个数不同。

函数重载的规则：

- 函数名称必须相同。
- 参数列表必须不同（个数不同、类型不同、参数排列顺序不同等）。
- 函数的返回类型可以相同也可以不相同。
- 仅仅返回类型不同不足以成为函数的重载。

二、函数重载的作用：

重载函数通常用来在同一个作用域内 用同一个函数名 命名一组功能相似的函数，这样做减少了函数名的数量，避免了名字空间的污染，对于程序的可读性有很大的好处。

三、函数重载是一种静态多态：

- (1) 多态：用同一个东西表示不同的形态；
- (2) 多态分为：
  - 静态多态（编译时的多态）；
  - 动态多态（运行时的多态）；
- (3) 函数重载是一种静态多态；

四.面试题

1.C语言中为什么不能支持函数重载？



从上图可知编译器在编译.c文件时，只会给函数进行简单的重命名；具体的方法是给函数名之前加上”\_”；所以加入两个函数名相同的函数在编译之后的函数名也照样相同；调用者会因为不知道到底调用那个而出错；

2.C++中函数重载底层是如何处理的？



在.cpp文件中，虽然两个函数的函数名一样，但是他们在符号表中生成的名称不一样。

‘?’ 表示名称开始， ‘?’ 后边是函数名 “@@YA” 表示参数表开始，后边的3个字符分别表示返回值类型，两个参数类型。 “@Z” 表示名称结束。

由于在.cpp文件中，两个函数生成的符号表中的名称不一样，所以是可以编译通过的。

### 3.C++中能否将一个函数按照C的风格来编译？

```
1  #include<iostream>
2  #include<windows.h>
3  using namespace std;
4
5
6  extern "C" int Add(int a, int b)
7  {
8
9      return a + b;
10 }
11 int main()
12 {
13     cout << Add(10, 20) << endl;
14     system("pause");
15     return 0;
16 }
17
```

可以按照C风格来编译，只需在函数名前加**extern "C"** 就可以完成按照C风格来编译