函数的默认值

2023年9月18日 16:30

```cpp
int func(int a, int b = 10, int c = 10) {
    return a + b + c;
}

//1. 如果某个位置参数有默认值，那么从这个位置往后，从左向右，必须都要有默认值
//2. 如果函数声明有默认值，函数实现的时候就不能有默认参数
int func2(int a = 10, int b = 10);
int func2(int a, int b) {
    return a + b;
}

int main() {

    cout << "ret = " << func(20, 20) << endl;
    cout << "ret = " << func(100) << endl;

    system("pause");

    return 0;
}
```

1、如果某个位置已经有了默认参数，那么从这个位置往后都必须有默认参数

2、函数的声明和定义只能有一个有默认参数

函数占位参数

2023年9月19日 19:48

3.2 函数占位参数

C++中函数的形参列表里可以有占位参数，用来做占位，调用函数时必须填补该位置

语法： `返回值类型 函数名 (数据类型){}`

在现阶段函数的占位参数存在意义不大，但是后面的课程中会用到该技术

示例：

```
//函数占位参数 ，占位参数也可以有默认参数
void func(int a, int) {
    cout << "this is func" << endl;
}

int main() {

    func(10,10); //占位参数必须填补

    system("pause");

```

占位参数也可以有默认值

函数重载和注意事项

2023年9月19日 19:48

3.3 函数重载

3.3.1 函数重载概述

作用： 函数名可以相同，提高复用性

函数重载满足条件：

- 同一个作用域下
- 函数名称相同
- 函数参数**类型不同** 或者 **个数不同** 或者 **顺序不同**

注意： 函数的返回值不可以作为函数重载的条件

```
//函数重载需要函数都在同一个作用域下
void func()
{
    cout << "func 的调用！ " << endl;
}
void func(int a)
{
    cout << "func (int a) 的调用！ " << endl;
}
void func(double a)
{
    cout << "func (double a)的调用！ " << endl;
}
void func(int a ,double b)
{
    cout << "func (int a ,double b) 的调用！ " << endl;
}
void func(double a ,int b)
{
    cout << "func (double a ,int b)的调用！ " << endl;
}

```

```cpp
//函数返回值不可以作为函数重载条件
//int func(double a, int b)
//{
//   cout << "func (double a ,int b)的调用! " << endl;
//}


int main() {

    func();
    func(10);
    func(3.14);
    func(10,3.14);
    func(3.14 , 10);

    system("pause");

    return 0;
```

注意：当函数重载碰到默认参数容易产生歧义，需要避免