集训课-17：自定义函数

课程目标

01 函数定义

02 函数调用

03 函数分类

04 变量作用域

05 知识回顾

Part 1

函数定义

函数定义

函 数

指可以实现某个功能，可以重复使用的一段代码。不同的函数之间相互独立，即函数之间的功能互不影响(相互的代码)。

函数定义

返回值类型 函数名(形参1, 形参2, …形参n)

```
{
        执行语句;
        return 值;
}
```

函数定义

```
int 函数名(形参1,形参2, …形参n)
{
        执行语句;
        return sum;
}
```

返回值的数据类型

向函数外输出的函数的运行结果

函数定义

- 标识符命名规则：
  - ✔ 字母、数字、下划线的组合
  - ✔ 不能以数字作为开头
  - ✔ 不能是C++关键字
  - ✔ 区分大小写
  - ✔ 见名知意
  - ✔ 不能冲突和重复

函数定义

★ 有参数有返回值

```
int add(int a, int b)
{
        int sum = a + b;
        return sum;
}
```

规定传入数据的类型并接受该数据

规定该函数正常工作所需要数据的种类和数量

函数定义

直接定义

```
#include<iostream>
using namespace std;

//直接定义区

int main()
{
        return 0;
}
```

声明定义

```
#include<iostream>
using namespace std;

//声明区

int main()
{
        return 0;
}
//定义区
```

Part 2

函数调用

函数调用

- 函数名(实参1, 实参2, … 实参3);

```
add(a, b);
```

函数调用

参数表中的参数为传入函数所需的必要的数据,一般来说函数在调用时需要向函数传递若干个参数,传参的个数=参数个数,极特殊情况下传参<参数。

- 实际参数: 调用函数时传入的参数
- 形式参数: 函数定义时写在参数表中的参数

函数调用

```
#include <iostream>
using namespace std;
double add(double n, double m)
{
        double sum = n + m;
        return sum;
}

int main()
{
        double a, b;
        cin >> a >> b;
        cout << add(a, b);
        return 0;
}
```

把a赋值给n，把b赋值给m，并执行add函数

将sum值返回并输入

Part 3

函数分类

函数分类

- 有参数有返回值 int add(int a, int b)
- 有参数无返回值 void print(int n)

函数分类

- 当一个函数使用了void返回类型,此时函数没有返回值,也不能获取其返回值,void类型的函数全部为操作型函数。

- 虽然void作为返回类型无法提供返回值,但是void类型函数仍然可以使用return语句来结束函数,只不过无法使用返回值。

- 当一个函数具有返回值时,就可以对其调用直接获取其返回值。

函数分类

- 有参数有返回值 int add(int a, int b)
- 有参数无返回值 void add(int a)
- 无参数有返回值 int main()
- 无参数无返回值 void name()

函数分类

参数传递

参数的传递是函数之间沟通的重要渠道，通过实参向形参传递数据，将数据传入函数，通过获取函数的返回值得到结果，就构成了函数之间的数据沟通。

函数分类

传递的分类

传值参数：

从函数外部将数据复制到函数内部，形参会复制实参的值并传入函数中，此时形参在函数中的操作不会影响到外部的实参。

传引用参数

传指针参数

函数分类

```
#include <iostream>
using namespace std;
void sp(int a, int b)
{
		int temp = a;
		a = b;
		b = temp;
}
int main()
{
		int a = 1, b = 2;
		sp(a, b);
		cout << "a=" << a << endl;
		cout << "b=" << b << endl;
		return 0;
}
```

```
a=1
b=2
```

函数分类

传递的分类

传值参数：

从函数外部将数据复制到函数内部，形参会复制实参的值并传入函数中，此时形参在函数中的操作不会影响到外部的实参。

传引用参数:

从函数外部获取数据的引用，此时形参就变成了对应实参的别名，此时形参和实参拥有同一个地址，当函数中形参的值发生改变时，会同时影响到函数外部的实参，这种方式也是函数向外沟通的另一种途径。

传指针参数

函数分类

引　用

是一种修饰手法，通过修饰形参就是可以使形参获得对应实参的地址，此时形参就变成了对应实参的别名，此时形参的地址和实参的地址是相同的。数组作为参数，默认传引用。

地址:是数据存储在计算机中的唯一标识。

写法:返回类型 函数名(参数类型 &参数名....)

&:取地址符，是一个修饰运算符，功能是获取被修饰变量的地址。

函数分类

```
#include <iostream>
using namespace std;
void sp(int a, int b)
{
        int temp = a;
        a = b;
        b = temp;
}
int main()
{
        int a = 1, b = 2;
        sp(a, b);
        cout << "a=" << a << endl;
        cout << "b=" << b << endl;
        return 0;
}
```

```
a=1
b=2
```

```
#include <iostream>
using namespace std;
void sp(int &a, int &b)
{
        int temp = a;
        a = b;
        b = temp;
}
int main()
{
        int a = 1, b = 2;
        sp(a, b);
        cout << "a=" << a << endl;
        cout << "b=" << b << endl;
        return 0;
}
```

```
a=2
b=1
```

函数分类

数组作为参数

返回值类型　函数名(数据类型 数组名[])
{
　　执行语句;
　　return 值;
}

函数名(数组名)

Part 4

变量作用域

变量作用域

局部变量

定义在函数内部或者代码块的变量，作用域是定义该变量的函数或者代码块。

变量作用域

- 局部变量属于定义其的代码块，当代码块执行完毕时，会被系统删除节约内存。

- 局部变量的作用范围只限于定义其的大括号以及大括号内。

- 局部变量不会自动初始化且严禁重名。

- 局部变量若和全局变量发生重名,该代码块会直接屏蔽外部的全局变量使用内部的局部变量

变量作用域

全局变量

定义在函数外的变量，作用域是定义该变量的位置到程序结束。

变量作用域

- 全局变量可以在程序的任意一个位置进行调用
- 全局变量允许和局部变量重名，但是遵循就近原则以局部变量为准。
- 全局变量默认初始化为0。
- 全局变量是直接定义在程序的底层中的，越多，程序运行越慢。

Part 5

知识回顾

知识回顾

一、基本概念

函数：指可以实现某个功能，可以重复使用的一段代码。

函数的定义：

```
返回值类型  函数名(形参1, 形参2, …形参n)
{
    执行语句;
    return 值;
}
```

函数的调用：

```
函数名(实参1, 实参2, … 实参3);
```

知识回顾

二、函数分类

有参数有返回值　　有参数无返回值

无参数无返回值　　无参数有返回值

三、变量作用域

局部变量：定义在函数内部或者代码块的变量，作用域是定义该变量的函数或者代码块。

全局变量：定义在函数外的变量，作用域是定义该变量的位置到程序结束

Class is over