

## C++

笔记本： 我的第一个笔记本

创建时间： 2020/9/14 19:01

更新时间： 2020/9/14 22:08

作者： 1272209351@qq.com

---

## C++

C语言之父-----里奇

C++98 里边有63个关键字

asm	do	if	return	try	continue
auto	double	inline	short	typedef	for
bool	dynamic_cast	int	signed	typeid	public
break	else	long	sizeof	typename	throw
case	enum	mutable	static	union	wchar_t
catch	explicit	namespace	static_cast	unsigned	default
char	export	new	struct	using	friend
class	extern	operator	switch	virtual	register
const	false	private	template	void	true
const_cast	float	protected	this	volatile	while
delete	goto	reinterpret_cast			

### 一、命名空间的三种方式：

1、常用方式； 2、命名空间可以嵌套； 3、 同一个工程中可以定义多个名字相同的命名空间

```

16 namespace N1
17 {
18     int a = 10;
19     int b = 20;
20
21     int Add(int left, int right)
22     {
23         return left + right;
24     }
25 }
26
27 // 2. 命名空间可以嵌套
28 namespace N2
29 {
30     int a = 10;
31     int b = 20;
32
33     int Sub(int left, int right)
34     {
35         return left - right;
36     }
37
38     namespace N3
39     {
40         int c = 10;
41         int d = 20;
42
43         int Mul(int left, int right)
44         {
45             return left*right;
46         }
47     }
48 }
49
50 // 3. 在同一工程中，可以定义多个名字相同的命名空间
51 // 不会冲突
52 // 编译器会将多个相同名称的命名空间合并成一个
53 namespace N1
54 {
55     int Div(int left, int right)
56     {
57         return left / right;
58     }
59 }

```

相同作用域中不能出现相同的变量名字

相同名称的多个命名空间中：也不能出现相同的名字 因为编译器会将多个相同名称的命名空间最终合并成一个

```

1 namespace N
2 {
3     int a = 10;
4
5     // 嵌套的方式----该种方式不会冲突
6     // 相当于外层N命名空间中又包含了一个N的命名空间
7     namespace N1
8     {
9         int b = 10;
10        int a = 10;
11    }
12 }
13
14 int main()
15 {
16
17     return 0;
18 }

```

命名空间的访问方式：

## ：： 作用域运算符

```
9 namespace N
10 {
11     int a = 10;
12     int b = 20;
13
14     int Add(int left, int right)
15     {
16         return left + right;
17     }
18 }
19
20 int a = 20;
21
22 int main()
23 {
24     int a = 30;
25
26     printf("%d\n", a);
27     return 0;
28 }
29
30 int main()
31 {
32     int a = 30;
33
34     // 就近原则
35     printf("%d\n", a);
36
37     // 如果访问全局作用域中的a
38     // ::作用域运算符
39     // ::a 明确说明要访问全局作用域中的a
40     printf("%d\n", ::a);
41     return 0;
42 }
43
44 // 访问N命名空间中的a
45 printf("%d\n", N::a);
46 return 0;
```

```

36 // 该场景：对N命名命名空间中某些成员访问的非常频繁
37
38 using N::a;
39
40 int main()
41 {
42     // 访问N命名空间中的a
43     printf("%d\n", N::a);
44     printf("%d\n", N::a);
45     printf("%d\n", N::a);
46     printf("%d\n", N::a);
47     printf("%d\n", N::a);
48     printf("%d\n", N::a);
49     printf("%d\n", N::a);
50
51     // 为了写代码简单，想要直接访问N命名空间中的a
52     printf("%d\n", a);
53     return 0;
54 }

```

using N::a

该语句加上之后相当于将N 命名空间之内的 a 当成当前文件的一个全局变量来使用

```

// 该场景：对N命名命名空间中某些成员访问的非常频繁
// 优点：写代码简单了
// 缺点：如果该文件中有相同名称的变量或者函数，就会产生冲突
// 如果产生冲突，怎么办？ ----按照方式1使用即可

// 该条语句加上之后，相当于将N命名空间中的a当成当前文件的一个全局变量来使用
using N::a;

// 如果该文件中也有一个a，必然会产生冲突，只能按照方式1来进行使用
//int a = 10;

int main()
{
    int a = 10;
}

```

C++中的输入和输出：

C语言的输入输出方式在C++中依旧可以使用-----兼容

**#pragma warning (disable:4996)**

**#include<iostream>**

**using namespace std ;**

**cin >> endl;**

**cout << endl;**