

- 1 举例说明int和int&类型不同
- 2 举例说明int\*和int&类型不同
- 3 举例说明type和const type类型不同
- 4 举例说明typedef int MYINT类型
- 5 举例说明重载函数的调用二义性问题
- 6 举例说明const int&可以传常量

具体代码见homework，这里仅记录一些现象。

## 1 举例说明int和int&类型不同

Code snippet:

```

7 {
8     int n = 1;
9     int& an = n;
10
11     std::cout << n << std::endl;
12     std::cout << an << std::endl;
13 }

```

Annotations:

- an 是对n的引用，相当于就是n的一个别名，VS2015中，他们共用一块内存
- an不另外开辟内存

Memory dump (Address: 0x004FFC54):

地址	值	类型
0x004FFC54	01 00 00 00	int
0x004FFC64	63 17 9b 00	int *
0x004FFC74	cc cc cc cc	int &
0x004FFC84	cc cc cc cc	int *
0x004FFC94	cc cc cc cc	

Watch window:

名称	值	类型
n	1	int
&n	0x004ffc54 (1)	int *
an	↑	int &
&an	0x004ffc54 (1)	int *

## 2 举例说明int\*和int&类型不同

Code snippet:

```

4 #include <iostream>
5
6 using namespace std;
7
8 void foo2()
9 {
10     int n = 1;
11     int* p = &n;
12
13     std::cout << n << std::endl;
14     std::cout << &n << std::endl;
15     std::cout << p << std::endl;
16 }

```

Annotations:

- int型指针变量p指向n的地址

Console output:

```

C:\WINDOWS\system32\cmd
010FFB8C
010FFB8C
请按任意键继续. . .

```

### 3 举例说明type和const type类型不同

```
void foo3()
{
    int n = 1;
    int* p1 = &n;
    const int* p2 = &n;

    /*p2 = 3;
    std::cout << n << std::endl;
    std::cout << *p1 << std::endl;
    *p1 = 3;
    std::cout << *p1 << std::endl;
    //std::cout << *p2 << std::endl;
}
```

p1指向的值可以修改

const 修改的值不能修改

C:\> 选择C:\WINDOWS\system32

```
1
3
请按任意键继续...
```

### 4 举例说明typedef int MYINT类型

```
//typedef user-defined type int to MYINT
typedef int MYINT;

void foo4()
{
    int n = 1;
    MYINT m = 2;

    std::cout << n << std::endl;
    std::cout << m << std::endl;
}
```

定义一个新的类型，这个类型可以统称为，用户自定义类型

C:\> C:\WINDOWS\system32

```
1
2
请按任意键继续...
```

### 5 举例说明重载函数的调用二义性问题

```
5
6 using namespace std;
7
8 int getValue()
9 {
10     return 1;
11 }
12
13 float getValue()
14 {
15     return 1.0;
16 }
17
18 void foo5()
19 {
20     getValue();
21 }
```

错误列表

代码	说明
abc	无法重载仅按返回类型区分的函数
abc	无法重载仅按返回类型区分的函数
C2556	"float getValue(void)": 重载函数与"int getValue(void)"只是在返回类型上不同
C2371	"getValue": 重定义; 不同的基类型

想要仅按返回值类型区分, 达到函数重载的目的是不可取的。

## 6 举例说明const int&可以传常量

```
6 using namespace std;
7
8 void foo6()
9 {
10     const int& n = 10;
11
12     std::cout << n << std::endl;
13     std::cout << &n << std::endl;
14 }
```

C:\WINDOWS\system32\cmd.exe

```
10
0115F7F8
请按任意键继续...
```