

1.3 引用_物联网 / 嵌入式工程师 - 慕课网

“ 慕课网慕课教程 1.3 引用涵盖海量编程基础技术教程，以图文图表的形式，把晦涩难懂的编程专业用语，以通俗易懂的方式呈现给用户。

3. 引用

一、介绍

在 C 语言中我们经常需要通过一个指针变量指向一个普通变量，然后通过指针变量修改普通变量的值，特别是函数间传参的时候用的特别多。

** 在 C++ 中提供了一种不使用指针变量同样可以达到同样效果的机制，并且使用起来更方便，这种机制叫做 "引用", 引用就是对某一个变量或对象的别名，** 对引用的操作与对其所绑定的变量或对象的操作完全等价。

二、引用的定义及使用

格式：类型 & 引用名 = 目标变量名

```
int main(void)
{
    int data = 100;

    int &e = data;

    quote = 200;

    cout << "data : " << data << endl;

    return 0;
}
```

注意以下几点:

- & 不是求地址运算符，而是起标志作用
- 引用的类型必须和其所绑定的变量的类型相同
- 定义引用的同时必须对其初始化，否则编译会报错
- 引用相当于变量或对象的别名，因此不能再将已有的引用名作为其他变量或对象的名字或别名

下面代码中，错误的原因是什么？

```
#include <iostream>
```

```
using namespace std;

int main(void)
{
    double a = 10.3;

    int &b = a;
}
```

```
#include <iostream>

using namespace std;

int main(void)
{
    int &b;
}
```

三、引用本质剖析

```
1 int main(int argc, const char *argv[])
2 {
3     int data;
4     int &quote_data = data;
5
6     data++;
7     quote_data++;
8     sp
9     return 0;
10 }
```

fp
-4
0x30012
-8
data
-12
argc
-16
argv
-20
sp

r3 → -12
sp → -20

```
1 main:
2 @ args = 0, pretend = 0, frame = 16
3 @ frame_needed = 1, uses_anonymous_args = 0
4 @ link_register_save eliminated.
5 str fp, [sp, #-4]!
6 add fp, sp, #0
7 sub sp, sp, #20
8 str r0, [fp, #-16]
9 str r1, [fp, #-20]
10 sub r3, fp, #12
11 str r3, [fp, #-8]
12 ldr r3, [fp, #-12]
13 add r3, r3, #1
14 str r3, [fp, #-12]
15 ldr r3, [fp, #-8]
16 ldr r3, [r3, #0]
17 add r2, r3, #1
18 ldr r3, [fp, #-8]
19 str r2, [r3, #0]
20 mov r3, #0
21 mov r0, r3
22 add sp, fp, #0
23 ldmfd sp!, {fp}
24 bx lr
```

不要在骗我了，引用的本质就是指针，把“&”去掉了，我就不认识你了吗？
此时我在想C++的设计者是不是考虑到程序员天生对“&”有恐惧感，而使用引用来代替指针。

结论:

引用操作最终编译器会将它翻译成指针的操作, 所以引用的本质就是指针。

四、对指针和数组的引用

1. 对指针进行引用

** 格式:** 类型 * & 引用名 = 指针名

```
#include <iostream>

using namespace std;
```

```
int main(void)

{

    int    data = 10;

    int * ptr = &data;

    int * &new_ptr = ptr;


    cout << &ptr << " " << &new_ptr << endl;


    return 0;

}
```

2. 对数组进行引用

格式：类型 (& 引用名)[数组中元素个数] = 数组名

```
#include <iostream>


using namespace std;


int main(void)

{

    int a[3] = {1,2,3};

    int (&b)[3] = a;

    cout << "sizeof(b) : " << sizeof(b) << endl;

    return 0;

}
```

五、引用作为函数参数

```
#include <iostream>


using namespace std;


void change_value(int &data)

{

    data = 200;

}


int main(void)

{

    int data = 100;
```

```
        change_value(data);

        cout << "data : " << data << endl;

        return 0;
    }
}
```

六、引用作为函数返回值

```
#include <iostream>

using namespace std;

string &function(void)
{
    string str = "hello world";

    return str;
}

int main(void)
{
    string str = function();

    cout << str << endl;

    return 0;
}
```

思考:

程序有什么问题，如何解决？

七、常引用

** 在定义引用的时候用 const 进行修饰，** 这样引用就变成了常引用，不允许通过该引用对其所绑定的变量或对象进行修改。

格式: const 类型 & 引用名 = 目标变量名

```
#include <iostream>

using namespace std;

int main(void)
{
    int a = 10;

    const int &new_a = a; //常引用
}
```

```
new_a = 100;

return 0;
}

#include <iostream>

using namespace std;

int main(void)
{
    const int &c = 15; //ok
    int &a = 20; //error

    return 0;
}
```

八、对比

1. 普通引用和常引用

无法复制加载中的内容

2. 引用与指针的区别

- 指针在程序运行的时候，可以改变它的值，而引用和一个变量绑定之后就不能在引用其他变量
- 引用不可以为空，当被创建的时候，必须初始化，而指针可以是空值，可以在任何时候被初始化
- `**sizeof(引用)` 得到的是所指向的变量（对象）的大小，而 `sizeof(指针)**` 得到的是指针本身的大小；
- 理论上，对于指针的级数没有限制，但是引用只能是一级

九、任务

1. 设计一个函数 `my_swap`，实现两个数的交换，请分别使用指针和引用进行实现
2. 请修复下面程序中的错误

```
#include <iostream>
```

```
using namespace std;

void change(const int &a)
{
    a ++;
}

void printf_string(string str)
{
    cout << "str:" << str << endl;
}

int main(void)
{
    int a = 10;

    change(a);

    cout << "a = " << a << endl;

    string str = "hello";

    printf_string(str);

    printf_string("world");

    return 0;
}
```

- 划线
- 写笔记

学习要认真，笔记应当先



公开笔记 0/1000 提交



让我再睡五分钟 Sunny_SunshineX

删除 编辑

全文完

本文由 简悦 SimpRead 优化，用以提升阅读体验

使用了 全新的简悦词法分析引擎 beta，点击查看详细说明

