

# C/C++基础学习笔记

## 1.static的三种用法

### 1.1用法1:

在同一个.c文件里，static一个变量，相当于是一个本文件的全局变量，其他文件不能调用该变量。

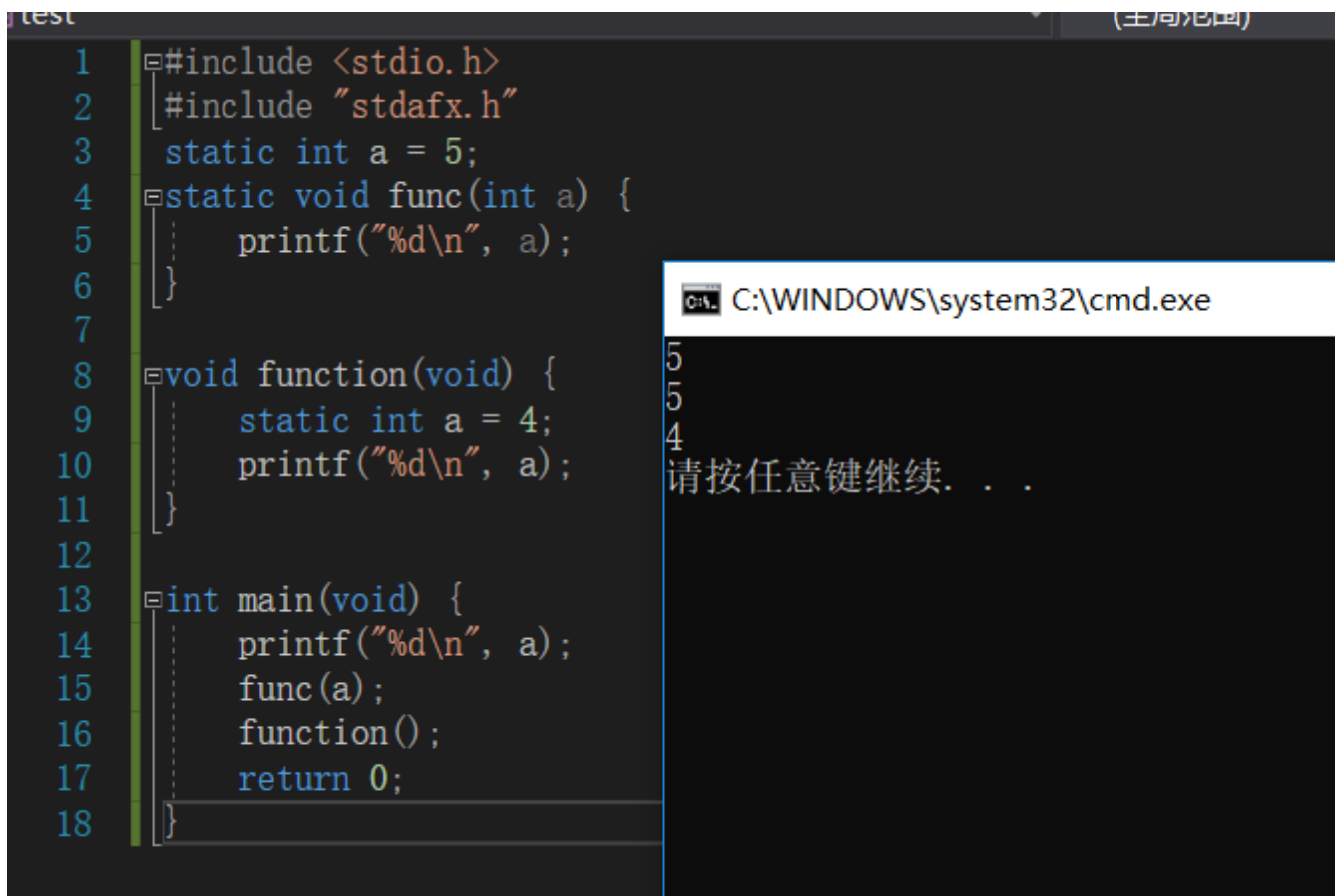
### 1.2用法2:

声明成静态函数时，这个函数只能被本文件的其他函数调用，其他文件不能调用。

### 1.3用法3:

修饰一个局部变量时，作用域是局部的，但是生命域是全局的。这个变量被其他函数调用时，变量值不变。

### 代码示例及运行结果:



The screenshot shows a C++ IDE with a code editor on the left and a command prompt window on the right. The code in the editor is as follows:

```
1  #include <stdio.h>
2  #include "stdafx.h"
3  static int a = 5;
4  static void func(int a) {
5      printf("%d\n", a);
6  }
7
8  void function(void) {
9      static int a = 4;
10     printf("%d\n", a);
11 }
12
13 int main(void) {
14     printf("%d\n", a);
15     func(a);
16     function();
17     return 0;
18 }
```

The command prompt window on the right shows the output of the program:

```
C:\WINDOWS\system32\cmd.exe
5
5
4
请按任意键继续. . .
```

## 2.const——只读

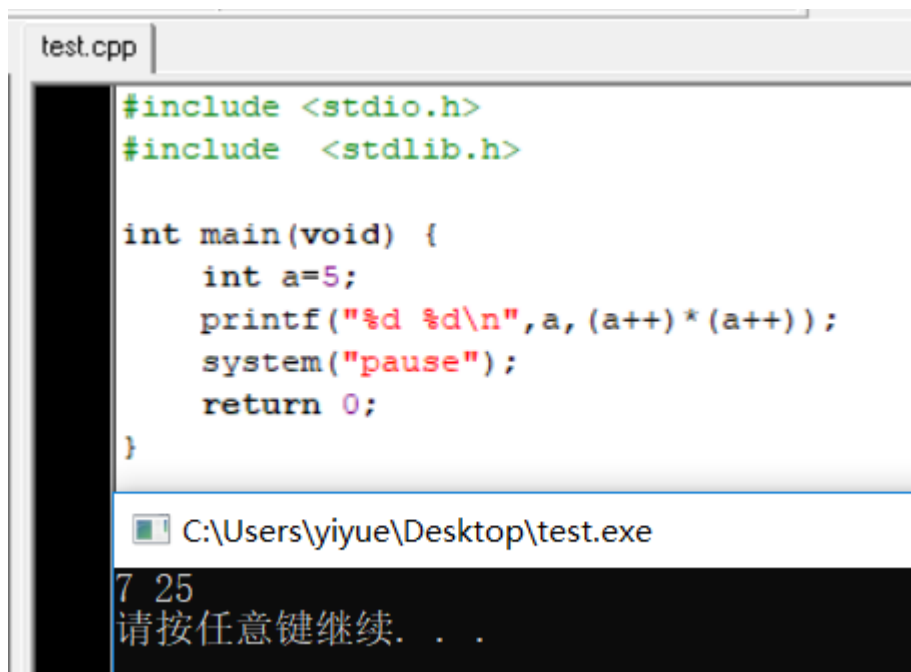
不能改变。权限不够，想要修改，就会报错。

const修饰的数据类型是指常类型，常类型的变量或对象的值是不能被更新的。

### 3.C/C++函数参数读取顺序

有的版本运行结果是 7 30.但是重点还是在7.

在Dev-C++上编译的结果：



The screenshot shows a code editor window titled 'test.cpp' containing the following C++ code:

```
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    int a=5;
    printf("%d %d\n",a, (a++)*(a++));
    system("pause");
    return 0;
}
```

Below the code editor, the output window shows the execution result:

```
C:\Users\yiyue\Desktop\test.exe
7 25
请按任意键继续. . .
```

<https://www.cnblogs.com/easonliu/p/4224120.html>

<https://github.com/Losfish/Blog/issues/1> by王博大佬

(Dev-c++和VS2017编译结果不一样，好麻烦不管了)

### 4.extern

可置于变量或函数前，以表示变量或者函数的定义在别的文件中。提示编译器遇到此变量或函数时，在其它模块中寻找其定义。

#### 4.1用法：

```
//正确用法：
extern int a; //'声明'一个全局变量a
int a;        //'定义'一个全局变量a
//定义只能在一处，但是声明可以出现多次。

//错误用法：
extern int a=0; //一旦给了初值，便为定义。这样定义变量不能通过编译（vs2013可以，但还是别皮）
```

#### 示例：

```
//在A.cpp里
extern int counter;
int main(void){
    printf("%d\n",counter);
    return 0;
}
```

```
//在B.cpp里
int counter=0;
```

## 5.动态数组申请与malloc不得不说的秘密

### 又名：数组与malloc与指针的修罗场

#### 5.1一维数组

```
//写法1
#include <stdio.h>
#include <malloc.h>
int main()
{
    int n, i, *array = NULL;
    scanf("%d", &n);
    array = (int*)malloc(n*sizeof(int)); //动态分配空间
    /*
    for(i = 0; i < n; ++i)
        scanf("%d", &array[i]); //赋值 等价于scanf("%d", array+i);
    for(i = 0; i < n; ++i)
        printf("%d\t", array[i]); //输出数组
    */
    free(array); //释放分配的内存空间
    return 0;
}

//写法2
int n;
scanf("%d",&n);
int *p=(int *)malloc(sizeof(int)*n);
```

代码示例及运行结果：

```

#include <stdio.h>
#include <stdlib.h>
int main()
{
    int n;
    while (scanf("%d",&n)) {
        int *p=(int *)malloc(sizeof(int)*n);
        for (int i=0;i<n;i++){
            p[i]=i+1; //等价于*(p+i)=i+1;
            printf("%d ",p[i]); //等价于 printf("%d ",*(p+i));
        }
        printf("\n");
        free(p);
    }
}

```

C:\Users\yiyue\Desktop\test.exe

```

1
1
2
1 2
4
1 2 3 4
5
1 2 3 4 5

```

## 5.2 二维数组

```

int row=0;    //行数
int col=0;    //列数
int i;
int ** arr=NULL;    //下面假设存储的数据类型为int
printf("请输入二维数组的行数和列数: ");
scanf("%d%d",&row,&col);
//要不要加判断输入是否合法你自己决定, 这里就不加,
arr = (int **)malloc(sizeof(int*)*row);    //arr在这里可以看出成数组, 数组的每个成员都是指向int
类型的指针, 这样每个指针指向的代表一行, 共row行
for(i=0; i<row; i++)    //为每行申请空间
{
    arr[i]=(int*)malloc(sizeof(int)*col);    //每一行有col列
}

```

代码示例及运行结果：

```

#include <stdio.h>
#include <stdlib.h>
int main()
{
    int row=0;    //函数
    int col=0;    //列数
    int **arr=NULL;    //下面假设存储的数据类型为int
    printf("请输入二维数组的行数和列数: ");
    while(scanf("%d %d",&row,&col)){
        arr = (int **)malloc(sizeof(int*)*row);
        //arr在这里可以看出成数组, 数组的每个成员都是指向int类型的指针, 这样每个指针指向的代表一行, 共row行
        for(int i=0; i<row; i++)//为每行申请空间
        {
            arr[i]=(int*)malloc(sizeof(int)*col);//每一行有col列
        }

        for(int i=0;i<row;i++){
            for(int j=0;j<col;j++){
                arr[i][j]=j; //赋值
                printf("%d ",arr[i][j]);//输出
                if(j == col-1)
                    printf("\n");
            }
        }
        free(arr);
        printf("请输入二维数组的行数和列数: ");
    }
    system("pause");
}

```

C:\Users\yiyue\Desktop\test.exe

```

请输入二维数组的行数和列数: 1 3
0 1 2
请输入二维数组的行数和列数: 4 4
0 1 2 3
0 1 2 3
0 1 2 3
0 1 2 3
请输入二维数组的行数和列数:

```