

# §. 基础知识题 - sscanf与sprintf的理解与体会



要求:

- 1、完成本文档中所有的测试程序并填写运行结果，从而体会这些cin的流成员函数的用法及区别
- 2、题目明确指定编译器外，缺省使用VS2022即可
  - ★ 如果要换成其他编译器，可能需要自行修改头文件适配
  - ★ 部分代码编译时有warning，不影响概念理解，可以忽略
- 3、直接在本文件上作答，**写出答案/截图（不允许手写、手写拍照截图）**即可；填写答案时，为适应所填内容或贴图，**允许调整**页面的字体大小、颜色、文本框的位置等
  - ★ 贴图要有效部分即可，不需要全部内容
  - ★ 在保证一页一题的前提下，具体页面布局可以自行发挥，简单易读即可
  - ★ **不允许**手写在纸上，再拍照贴图
  - ★ **允许**在各种软件工具上完成（不含手写），再截图贴图
  - ★ 如果某题要求VS+Dev的，则如果两个编译器运行结果一致，贴VS的一张图即可，如果不一致，则两个图都要贴
- 4、转换为pdf后提交
- 5、**5月9日前**网上提交本次作业（在“文档作业”中提交）

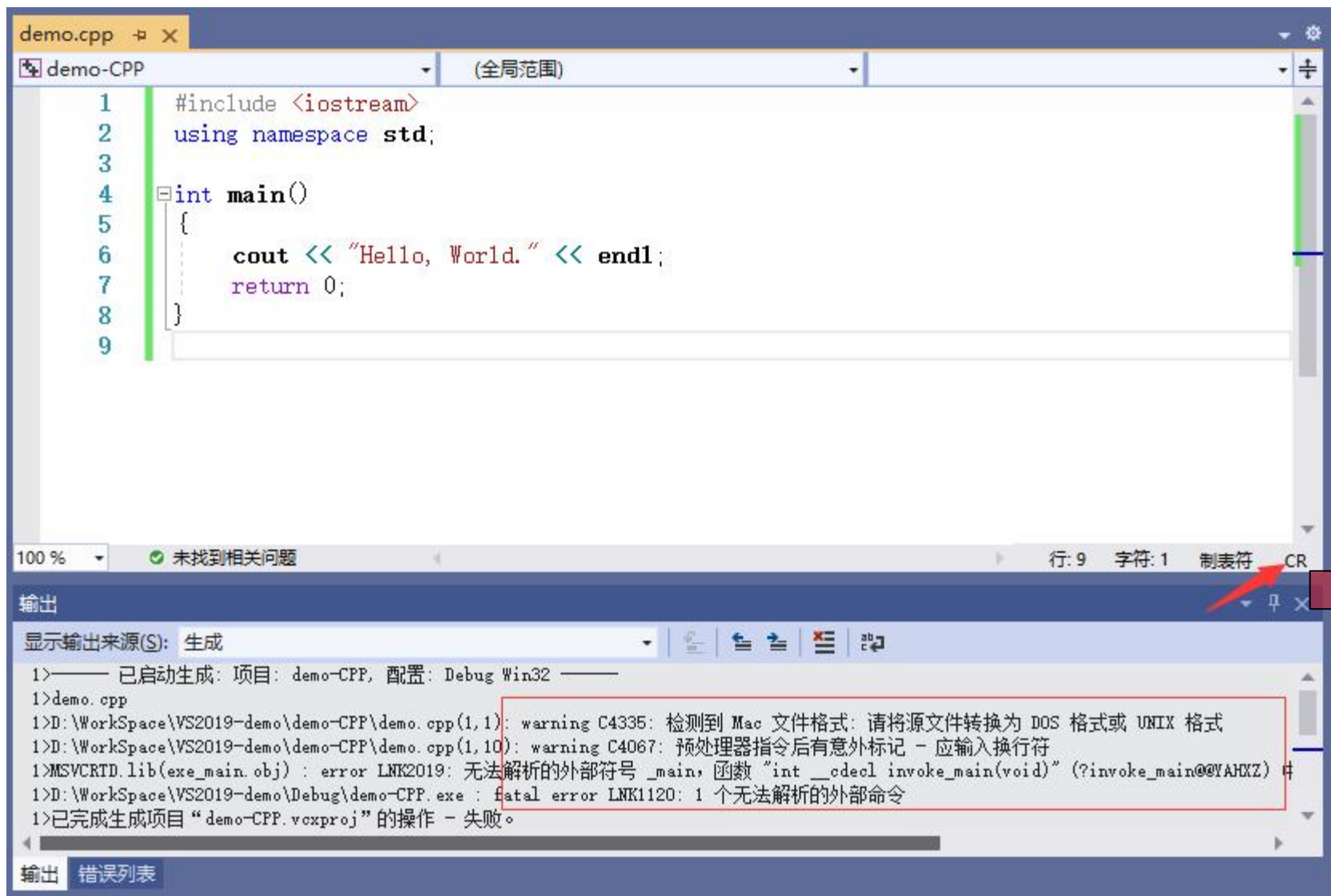


## §. 基础知识题 - sscanf与sprintf的理解与体会

注意:

用WPS等其他第三方软件打开PPT, 将代码复制到VS2022中后, 如果出现类似下面的**编译报错**, 则观察源程序编辑窗口的

的右下角是否为CR, 如果是, 单击CR, 在弹出中选择CRLF, 再次CTRL+F5运行即可



# § . 基础知识题 - sscanf与sprintf的理解与体会



基本概念:

★ 将格式化输出的内容放入字符串中

int sprintf(**字符数组**, "格式串", 输出表列);

- 返回值是输出字符的个数 (**同printf**)
- 字符数组要有足够空间容纳输出的数据 (**否则越界错**)
- 格式串同printf
- VS下需加 #define \_CRT\_SECURE\_NO\_WARNINGS

★ 从字符串中进行格式化输入

int sscanf(**字符数组**, "格式串", 输入地址表列);

- 返回值是正确读入的输入数据的个数 (**同scanf**)
- 格式串同scanf
- VS下需加 #define \_CRT\_SECURE\_NO\_WARNINGS

## §. 基础知识题 - sscanf与sprintf的理解与体会



1. 将格式化输出的内容放入字符串中

例1:

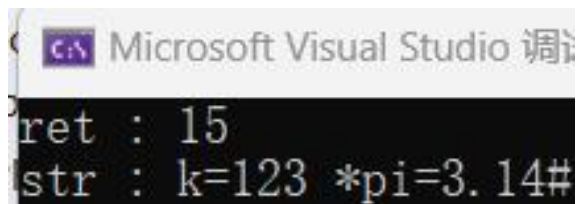
```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main()
{
    char str[80];
    int k=123, ret;
    double pi=3.1415925;

    ret = sprintf(str, "k=%-4d*pi=%.2f#", k, pi);
    printf("ret : %d\n", ret);
    printf("str : %s\n", str);

    return 0;
}
```

输出结果:



```
ret : 15
str : k=123 *pi=3.14#
```

- 1、本作业的所有程序，均在.c方式下运行，后续不再提示
- 2、认真阅读第5章课件!!!

本页需填写答案



## §. 基础知识题 - sscanf与sprintf的理解与体会

1. 将格式化输出的内容放入字符串中

例2:

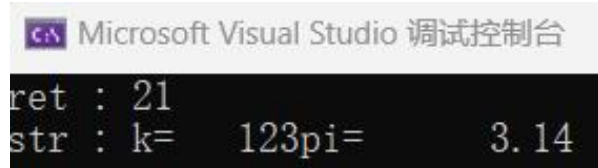
```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main()
{
    char str[80];
    int k=123, ret;
    double pi=3.1415925;

    ret = sprintf(str, "k=%6dpi=%10.2f", k, pi);
    printf("ret : %d\n", ret);
    printf("str : %s\n", str);

    return 0;
}
```

输出结果:



```
Microsoft Visual Studio 调试控制台
ret : 21
str : k=   123pi=   3.14
```

结合例1和例2，sprintf的返回值是：返回写入的字符总数，不包括追加在字符串末尾的空字符

本页需填写答案



## §. 基础知识题 - sscanf与sprintf的理解与体会

1. 将格式化输出的内容放入字符串中

例3:

VS+Dev

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
```

```
int main()
{
```

```
    char str[15];
    int k=123, ret;
    double pi=3.1415925;
```

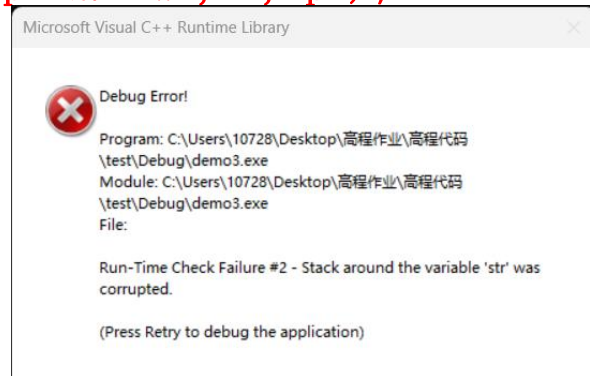
```
    ret = sprintf(str, "k=%-4d*pi=%.2f#", k, pi);
```

```
    printf("ret : %d\n", ret);
```

```
    printf("str : %s\n", str);
```

```
    return 0;
```

```
}
```



```
C:\Users\10728\Desktop\高程
ret : 15
str : k=123 *pi=3.14#
```

输出结果:

结合例1/2/3, sprintf使用时对字符数组的要求是: 输出位数要可靠

本页需填写答案



## §. 基础知识题 - sscanf与sprintf的理解与体会

### 2. 从字符串中进行格式化输入

例4:

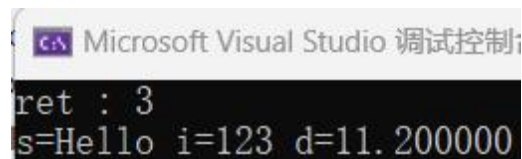
```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main()
{
    char str[80] = "Hello 123 11.2", s[10];
    int i, ret;
    double d;

    ret = sscanf(str, "%s %d %lf", s, &i, &d);
    printf("ret : %d\n", ret);
    printf("s=%s i=%d d=%f\n", s, i, d);

    return 0;
}
```

输出结果:



```
Microsoft Visual Studio 调试控制台
ret : 3
s=Hello i=123 d=11.200000
```

本页需填写答案



## §. 基础知识题 - sscanf与sprintf的理解与体会

### 2. 从字符串中进行格式化输入

例5:

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main()
{
    char str[80] = "123Hello";
    int i, j, ret;

    ret = sscanf(str, "%d%d", &i, &j);
    printf("ret : %d\n", ret);
    printf("i=%d j=%d\n", i, j);

    return 0;
}
```

输出结果:

```
ret : 1
i=123 j=-858993460
```

结合4例和例5，sscanf的返回值是：成功返回参数数目，失败返回-1

本页需填写答案





## §. 基础知识题 - sscanf与sprintf的理解与体会

### 2. 从字符串中进行格式化输入

例6:

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main()
{
    char str[80] = "123 456";
    int i, j, ret;

    ret = sscanf(str, "%d%d", &i, &j);
    printf("ret : %d\n", ret);
    printf("i=%d j=%d\n", i, j);

    ret = sscanf(str, "%d%d", &j, &i); //顺序反
    printf("ret : %d\n", ret);
    printf("i=%d j=%d\n", i, j);

    return 0;
}
```

输出结果:

```
ret : 2
i=123 j=456
ret : 2
i=456 j=123
```

本例说明，str中的内容\_可以\_（可以/不可以）被重复读取

本页需填写答案



## §. 基础知识题 - sscanf与sprintf的理解与体会

### 3. 综合应用

例7:

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

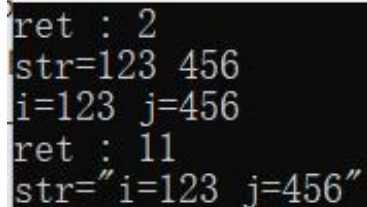
int main()
{
    char str[80] = "123 456";
    int i, j, ret;

    ret = sscanf(str, "%d%d", &i, &j);
    printf("ret : %d\n", ret);
    printf("str=%s\ni=%d j=%d\n", str, i, j);

    ret = sprintf(str, "i=%d j=%d", i, j);
    printf("ret : %d\n", ret);
    printf("str=\"%s\"\n", str);

    return 0;
}
```

输出结果:



```
ret : 2
str=123 456
i=123 j=456
ret : 11
str="i=123 j=456"
```

本例说明，str中的内容\_可以\_（可以/不可以）被替换

本页需填写答案

# §. 基础知识题 - sscanf与sprintf的理解与体会



## 3. 综合应用 例8:

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main()
{
    int x, w;
    printf("请输入[1..99999]间的整数及显示宽度[6..10]\n");
    scanf("%d %d", &x, &w); //不考虑输入错误
    printf("01234567890123456789\n"); //标尺

    char fmt[16];
    sprintf(fmt, "%%dd*\n", w);
    printf(fmt, x);

    return 0;
}
```



1、输入3 6，输出：

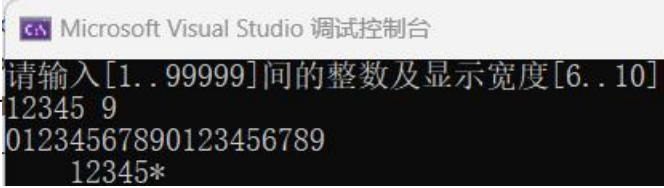
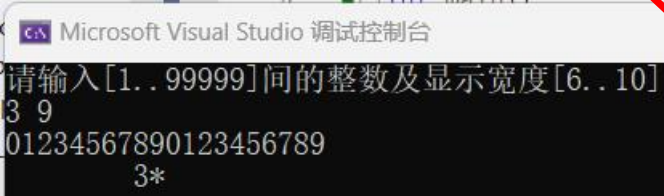
2、输入123 6，输出：

3、输入12345 6，输出：

4、输入3 9，输出：

5、输入123 9，输出：

6、输入12345 9，输出：



别去网上瞎查，认真阅读第5章课件

本页需填写答案



## §. 基础知识题 - sscanf与sprintf的理解与体会

### 3. 综合应用

例9：键盘输入一个长度[3..12]间字符串，再输入显示宽度[长度+1..20]，左对齐输出这个字符串（最后加\*分辨空格）

注：输入宽度小于等于串长则置为串长+1，不考虑其它输入错误

//给出相应的代码，字体为宋体，字号根据代码量调整，不小于9号

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
```

```
int main()
{
    int i = 0, w;
    char arr[20] = { 0 };
    printf("请输入[3..12]间的字符串及显示宽度[1en+1..20]\n");
    scanf("%s %d", &arr, &w); //不考虑输入错误
    printf("01234567890123456789\n"); //标尺
```

```
    while (1)
    {
        if (arr[i] > 0)
            i++;
        else
            break;
    }
    if (w <= i + 1)
        w = i + 1;
    char fmt[16];
    sprintf(fmt, "%%-%ds*\n", w);
    printf(fmt, arr);
```

```
    return 0;
}
```

1、输入abc 12，输出：

```
请输入长度[3..12]间的字符串及显示宽度[1en+1..20]
abc 12
01234567890123456789
abc *
```

2、输入abc 2，输出：

```
请输入长度[3..12]间的字符串及显示宽度[1en+1..20]
abc 2
01234567890123456789
abc *
```

3、自己构造的测试样本1



```
Microsoft Visual Studio 调试控制台
请输入[3..12]间的字符串及显示宽度[1en+1..20]
abc 8
01234567890123456789
abc *
```

4、自己构造的测试样本2



```
Microsoft Visual Studio 调试控制台
请输入[3..12]间的字符串及显示宽度[1en+1..20]
a12c 9
01234567890123456789
a12c *
```

页需填写答案

## §. 基础知识题 - sscanf与sprintf的理解与体会

本页需填写答案



### 3. 综合应用

例10: 键盘输入一个double型数据, 再输入总显示宽度及小数点后的位数, 右对齐输出这个字符串 (最后加\*分辨空格)

注:

//给出相应的代码, 字体为宋体, 字号根据代码量调整, 不小于9号

```
#define _CRT_SECURE_NO_WARNINGS
```

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

```
int main()
```

```
{
```

```
    int w, i = 0, y;
```

```
    double x;
```

```
    printf("请输入[3..12]间的字符串及显示宽度  
[len+1..20]\n");
```

```
    scanf("%lf %d %d", &x, &w, &y); //不考虑输入错误
```

```
    printf("01234567890123456789\n"); //标尺
```

```
    char fmt[16];
```

```
    sprintf(fmt, "%%d.%df*\n", w, y);
```

```
    printf(fmt, x);
```

```
    return 0;
```

```
}
```

1、输入12.34 9 5, 输出:

```
请输入double型数据及显示总宽度、小数点后位数  
12.34 9 5  
01234567890123456789  
12.34000*
```

2、输入123.456789 12 2, 输出:

```
请输入double型数据及显示总宽度、小数点后位数  
123.456789 12 2  
01234567890123456789  
123.46*
```

3、输入12345678.9 5 2, 输出:

```
请输入double型数据及显示总宽度、小数点后位数  
12345678.9 5 2  
01234567890123456789  
12345678.90*
```

4、输入12345678.9 5 0, 输出:

```
请输入double型数据及显示总宽度、小数点后位数  
12345678.9 5 0  
01234567890123456789  
12345679*
```

(3/4的答案没问题, 想不通去看第3章作业)