

§. 基础知识题 - cin与cout的基本使用



要求:

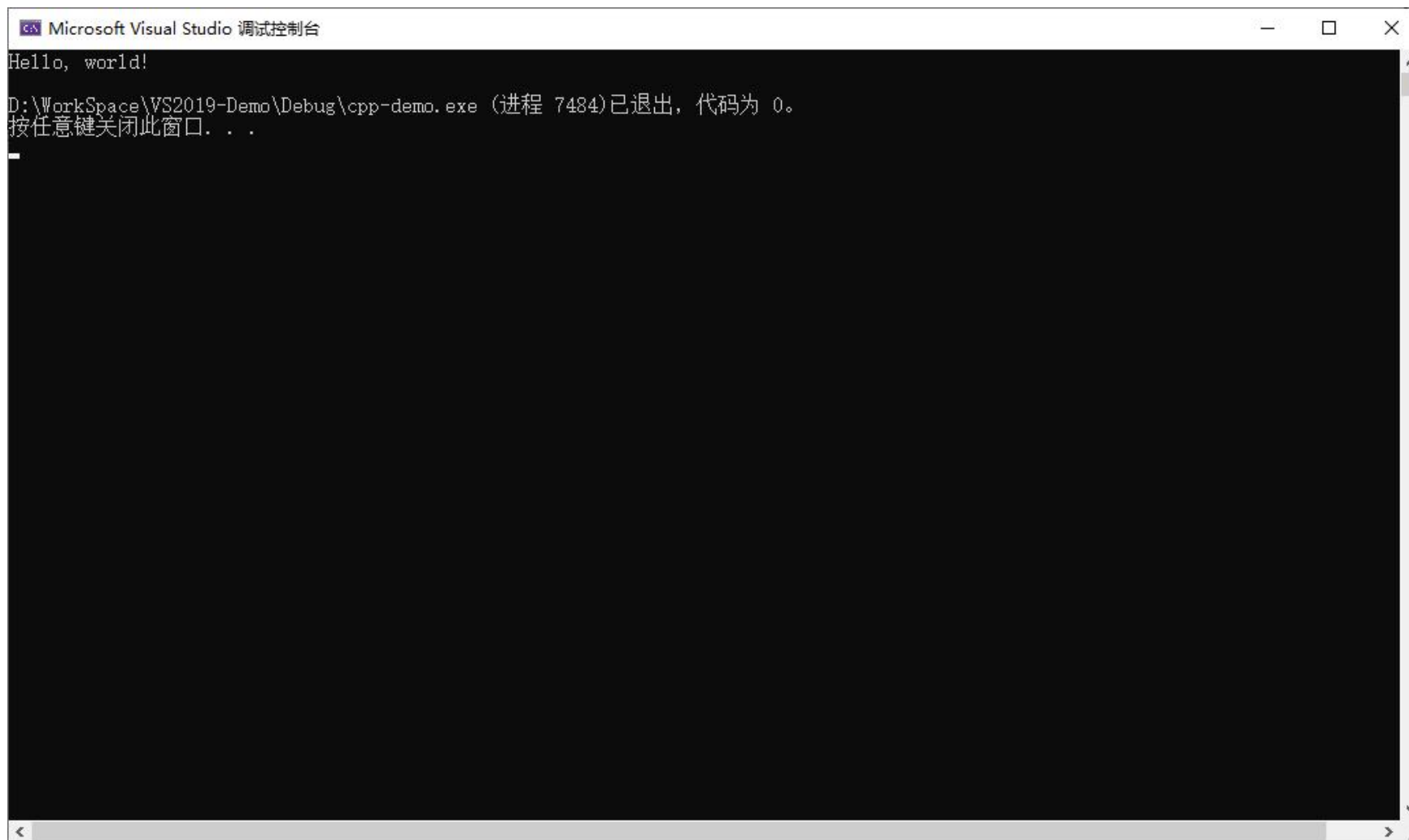
- 1、完成本文档中所有的题目并写出分析、运行结果
- 2、无特殊说明，均使用VS2019编译即可
- 3、直接在本文件上作答，**写出答案/截图（不允许手写、手写拍照截图）**即可；填写答案时，为适应所填内容或贴图，**允许调整**页面的字体大小、颜色、文本框的位置等
 - ★ 贴图要有效部分即可，不需要全部内容
 - ★ 在保证一页一题的前提下，具体页面布局可以自行发挥，简单易读即可
 - ★ **不允许**手写在纸上，再拍照贴图
 - ★ **允许**在各种软件工具上完成（不含手写），再截图贴图
- 4、转换为pdf后提交
- 5、**9月30日前**网上提交本次作业（在“实验报告”中提交）
=> 注：因为课时问题，本次作业10069206/10071706班级的同学放宽到10.4提交

§. 基础知识题 - cin与cout的基本使用

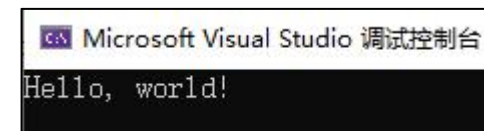


贴图要求：只需要截取输出窗口中的有效部分即可，如果全部截取/截取过大，则视为无效贴图

例：无效贴图

A screenshot of the Microsoft Visual Studio debug console window. The window is titled "Microsoft Visual Studio 调试控制台". It contains the text "Hello, world!" followed by a newline, and then "D:\Workspace\VS2019-Demo\Debug\cpp-demo.exe (进程 7484)已退出, 代码为 0." followed by another newline and "按任意键关闭此窗口. . .". The window is large, showing a significant portion of the screen.

例：有效贴图

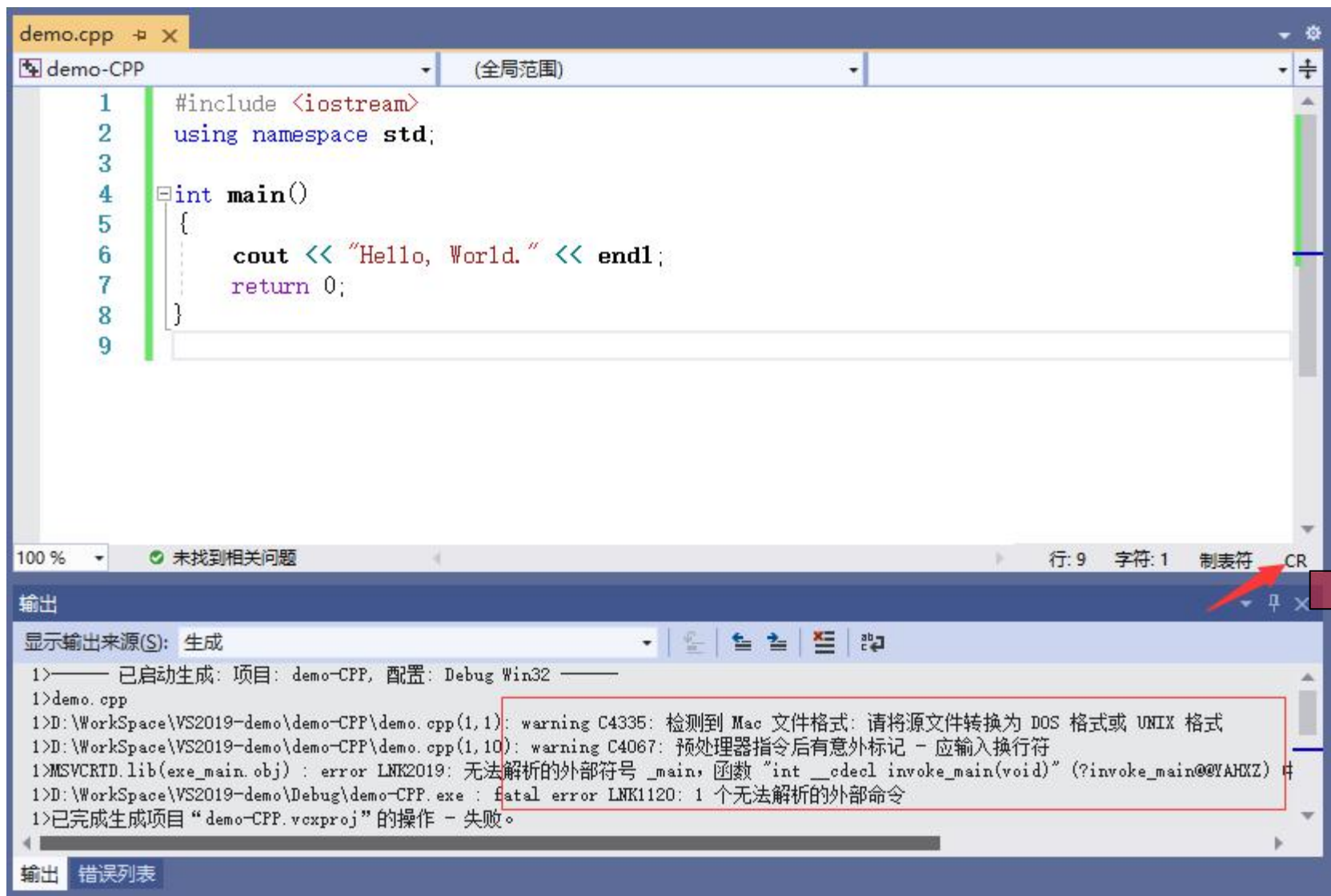
A screenshot of the Microsoft Visual Studio debug console window, cropped to show only the "Hello, world!" text. The window title "Microsoft Visual Studio 调试控制台" is visible at the top.



§. 基础知识题 - cin与cout的基本使用

附：用WPS等其他第三方软件打开PPT，将代码复制到VS2019中后，如果出现类似下面的**编译报错**，则观察源程序编辑窗

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 - cin与cout的基本使用



特别提示:

- 1、做题过程中, 先按要求输入, 如果想替换数据, 也要先做完指定输入
- 2、如果替换数据后出现某些问题, 先记录下来, 不要问, 等全部完成后, 还想不通再问 (也许你的问题在后面的题目中有答案)
- 3、不要偷懒、不要自以为是的脑补结论!!!
- 4、先得到题目要求的小结论, 再综合考虑上下题目间关系, 得到综合结论
- 5、这些结论, 是让你记住的, 不是让你完成作业后就忘掉了
- 6、换位思考(从老师角度出发), 这些题的目的是希望掌握什么学习方法?



§ . 基础知识题 - cin与cout的基本使用

基本知识点:

- 1、cin是按格式读入，到空格、回车、非法为止
- 2、cin的输入必须以回车结束，输入的内容放在输入缓冲区中，从输入缓冲区去取得所需要的内容后，多余的内容还放在输入缓冲区中，等待下次读入（如果程序结束，则操作系统会清空输入缓冲区）
- 3、系统会自动根据cin后变量的类型按**最长原则**来读取合理数据
- 4、变量读取后，系统会判断输入数据是否超过变量的范围，若超过则**置内部的错误标记**并返回一个**不可信**的值（不同编译器处理不同）
 - 4.1、cin输入完成后，通过cin.good()/cin.fail()可判断本次输入是否正确
 - 4.2、cin碰到非法字符后会置错误标记位，后面会一直错（**如何恢复还未学到，先放着**）
 - 4.3、cin连续输入多个int时，碰到非法字符，下一个是0，再下面才是随机值
 - 4.4、cin超范围后，不同类型的数据处理不同，如果细节记不清，问题不大，但一定要知道有这回事，别奇怪
 - 4.5、cin超范围和赋值超范围是不同的

5、cout根据数据类型决定输出形式

输入	cin.good() 返回	cin.fail() 返回
正确范围 +回车/空格/非法输入	1	0
错误范围 +回车/空格/非法输入	0	1
非法输入	0	1

6、先认真看课件!!!



§. 基础知识题 - cin与cout的基本使用

1、cout的基本理解

A. 观察下列程序的运行结果，回答问题并将程序的运行结果截图贴上(如果有错则贴错误信息截图)

```
#include <iostream>
using namespace std;

int main()
{
    /* 第1组 */
    cout << "This is a C++ program." << endl;

    /* 第2组 */
    cout << "This is " << "a C++ " << "program." << endl;

    /* 第3组 */
    cout << "This is "
         << "a C++ "
         << "program."
         << endl;

    /* 第4组 */
    cout << "This is ";
    cout << "a C++ ";
    cout << "program.";
    cout << endl;

    return 0;
}
```

第2组中，如果删除is及C++后面的空格，如何才能保持输出不变？
将你准备替换的行(仅允许1行)写在下面

```
cout << "This is" << " " << "a C++" << " " << "program." << endl;
```













第3组和第4组在语句上的区别是：
第三组是一个语句，总共有一个分号；第四组是四个语句，总共有四个分号。



§. 基础知识题 - cin与cout的基本使用

1、cout的基本理解

B. 观察下列4个程序的运行结果，回答问题并将各程序的运行结果截图贴上(如果有错则贴错误信息截图)

<pre>#include <iostream> using namespace std; int main() { int a=10, b=15, c=20; cout << a << b << c; return 0; }</pre> <div> Microsoft Visual Studio 调试控制台 101520</div>	<pre>#include <iostream> using namespace std; int main() { int a=10, b=15, c=20; cout << a, b, c; return 0; }</pre> <div> Microsoft Visual Studio 调试控制台 10</div>	<pre>#include <iostream> using namespace std; int main() { int a=10, b=15, c=20; cout << (a, b, c) << endl; return 0; }</pre> <div> Microsoft Visual Studio 调试控制台 20</div>	<pre>#include <iostream> using namespace std; int main() { int a=10, b=15, c=20; cout << a, b, c << endl; return 0; }</pre> <div><table><tr><th></th><th>代码</th><th>说明</th></tr><tr><td></td><td>E0299</td><td>无法确定需要哪个 函数模板 "std::endl" 实例</td></tr><tr><td></td><td>C2563</td><td>在形参表中不匹配</td></tr><tr><td></td><td>C2568</td><td>"<<": 无法解析函数重载</td></tr></table></div>		代码	说明		E0299	无法确定需要哪个 函数模板 "std::endl" 实例		C2563	在形参表中不匹配		C2568	"<<": 无法解析函数重载
	代码	说明													
	E0299	无法确定需要哪个 函数模板 "std::endl" 实例													
	C2563	在形参表中不匹配													
	C2568	"<<": 无法解析函数重载													
<p>解释这3个程序输出不同的原因：第一个程序是依次输出a, b, c的值；第二个程序的流插入运算符只能输出a的值，第一个逗号并列了前面的输出语句与后面的逗号表达式(b, c)；第三个程序输出了逗号表达式(a, b, c)的值。</p>			<p>解释错误原因：b, c<<endl是逗号表达式，c<<endl语法错误</p>												
<p>结论：一个流插入运算符 << 只能输出<u>1</u>个数据.</p>															

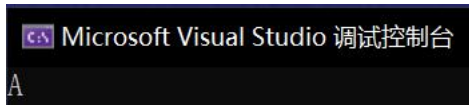


§. 基础知识题 - cin与cout的基本使用

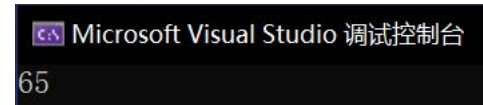
1、cout的基本理解

C. 观察下列2个程序的运行结果，回答问题并将各程序的运行结果截图贴上(如果有错则贴错误信息截图)

```
#include <iostream>
using namespace std;
int main()
{
    char ch = 65;
    cout << ch << endl;
    return 0;
}
```



```
#include <iostream>
using namespace std;
int main()
{
    int ch = 65;
    cout << ch << endl;
    return 0;
}
```



解释这两个程序输出不同的原因：第一个程序定义ch为字符型变量，并将ASCII码值赋予它，故输出对应ASCII码的字符；第二个程序定义ch为整型变量，故输出整型数65。



§. 基础知识题 - cin与cout的基本使用

1、cout的基本理解

D. 程序同C，将修改后符合要求的程序及运行结果贴上

```
#include <iostream>
using namespace std;
int main()
{
    char ch = 65;
    cout << ch << endl;
    return 0;
}
```

```
#include <iostream>
using namespace std;
int main()
{
    int ch = 65;
    cout << ch << endl;
    return 0;
}
```

```
test.cpp 1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     char ch = 65;
6     cout << int(ch) << endl;
7     return 0;
8 }
```

Microsoft Visual Studio 调试控制台
65

在char类型不变的情况下，要求输出为65
(不允许添加其它变量)

```
test.cpp 1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int ch = 65;
6     cout << char(ch) << endl;
7     return 0;
8 }
```

Microsoft Visual Studio 调试控制台
A

在int类型不变的情况下，要求输出为A
(不允许添加其它变量)



§. 基础知识题 - cin与cout的基本使用

1、cout的基本理解

E. 程序同C，将修改后符合要求的程序及运行结果贴上

```
#include <iostream>
using namespace std;
int main()
{
    char ch = 65;
    cout << ch + 0 << endl;
    return 0;
}
```

The screenshot shows the Visual Studio IDE with a file named 'test.cpp' open. The code is as follows:

```
1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     char ch = 65;
6     cout << ch + 0 << endl;
7     return 0;
8 }
```

The 'test' window shows the code with line numbers. The 'Debug Console' window at the bottom right shows the output '65'.

在char类型不变的情况下，要求输出为65
(不允许添加其它变量，
不允许使用任何方式的强制类型转换)



§. 基础知识题 - cin与cout的基本使用

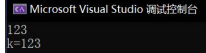
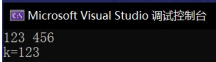
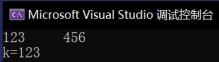
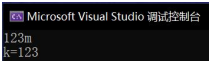
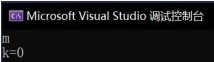
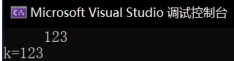
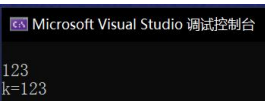

此页不要删除，也没有意义，仅仅为了分隔题目



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

A. 运行下面的程序，观察不同输入下的运行结果（贴图在清晰可辨的情况下尽可能小）

<pre>#include <iostream> using namespace std; int main() { short k; cin >> k; cout << "k=" << k << endl; return 0; }</pre>	<div>1、输入：123✓（✓代表回车键，下同）</div> <div>2、输入：123 456✓（一个空格）</div> <div>3、输入：123 456✓（多个空格）</div> <div>4、输入：123m✓</div> <div>5、输入：m✓</div> <div>6、输入：123✓（持续多个空格后，再输入123，按回车）</div> <div>7、输入：123✓（持续多个空格后，按回车） 123✓（再输入123，按回车）</div> <div>8、输入：✓ ... ✓ 123✓（持续多个空回车后，输入123）</div> <div>分析结果： 1、在前面有正确输入的情况下，回车、空格、（对int型而言是非法的字符）m的作用是？ 作为输入终止条件使输入停止，缓冲区不再存储字节 2、直接输入若干空格和回车后，再输入正确，变量是否能得到正确的值？ 是 3、直接输入（对int型而言是）非法的数据m，输出是？ 0</div>
<p>基础知识：</p> <p>short的最小值是：-32768</p> <p>short的最大值是：32767</p>	



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

B. 运行下面的程序，观察不同输入下的运行结果（贴图在清晰可辨的情况下尽可能小）

```
#include <iostream>
using namespace std;

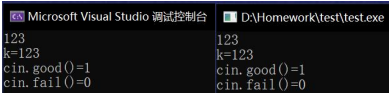
int main()
{
    short k;
    cin >> k;
    cout << "k=" << k << endl;
    cout << "cin.good()=" << cin.good() << endl;
    cout << "cin.fail()=" << cin.fail() << endl;
    return 0;
}
```

结论:

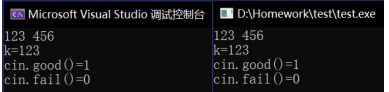
多个输入中，编号4、5、6输入的k值是不可信的

贴图即可，不需要写分析结果

1、输入：123✓（正确+回车）



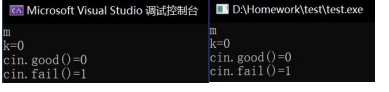
2、输入：123 456✓（正确+空格）



3、输入：-123m✓（正确+非法字符）



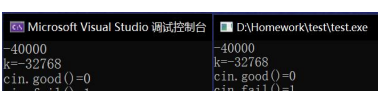
4、输入：m✓（直接非法字符）



5、输入：54321✓（超上限）



6、输入：-40000✓（超下限）



本题要求VS+Dev



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

B-Compare. 运行下面的**对比**程序（cin输入与赋值），观察运行结果并与B的输出结果进行对比分析

```
#include <iostream>
using namespace std;
int main()
{
    short k1, k2, k3, k4, k5;

    k1 = 12345;
    k2 = 54321;
    k3 = 70000;
    k4 = -12345;
    k5 = -54321;

    cout << k1 << endl;
    cout << k2 << endl;
    cout << k3 << endl;
    cout << k4 << endl;
    cout << k5 << endl;

    return 0;
}
```

B的输入:

1、输入: 12345✓ （合理范围）
对应本例的k1=12345

2、输入: 54321✓ （超上限但未超同类型的u_short上限）
对应本例的k2=-11215

3、输入: 70000✓ （超上限且超过同类型的u_short上限）
对应本例的k3=4464

4、输入: -12345✓ （合理范围）
对应本例的k4=-12345

5、输入: -54321✓ （超下限）
对应本例的k5=11215

u_short=unsigned short

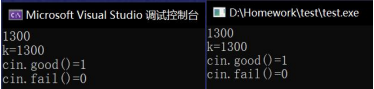
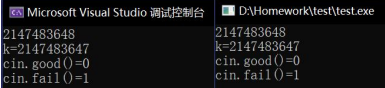

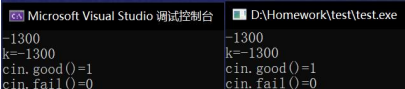
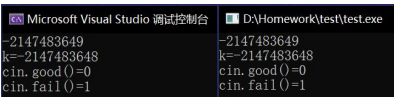
对比分析: cin输入时, 如果超出上限（或下限），会返回一个不可信值【VS与Dev是该数据类型上限（或下限）】；赋值时, 如果超出上限（或下限），会以二进制补码的形式存储, 再进行赋值、输出



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

C. 仿B，自行构造不同测试数据，观察不同输入下的运行结果（贴图在清晰可辨的情况下尽可能小）

<pre>#include <iostream> using namespace std; int main() { int k; cin >> k; cout << "k=" << k << endl; cout << "cin.good()=" << cin.good() << endl; cout << "cin.fail()=" << cin.fail() << endl; return 0; }</pre>	<p>贴图即可，不需要写分析结果</p> <p>1、输入：1300✓（合理范围）</p>  <p>2、输入：2147483648✓（超上限但未超同类型的u_int上限）</p>  <p>3、输入：4294967297✓（超上限且超过同类型的u_int上限）</p>  <p>4、输入：-1300✓（合理范围）</p>  <p>5、输入：-2147483649✓（超下限）</p> 	<p>u_int=unsigned int</p>
<p>结论：</p> <p>多个输入中，编号2、3、5输入的k值是可信的</p>		<p>本题要求VS+Dev</p>



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

C-Compare. 仿B-Compare, 构造**对比**程序 (cin输入与赋值, int型), 观察运行结果并与C的输出结果进行对比分析

```
#include <iostream>
using namespace std;
int main()
{
    int k1, k2, k3, k4, k5;

    k1 = 1300;
    k2 = 2147483648;
    k3 = 4294967297;
    k4 = -1300;
    k5 = -2147483649;

    cout << k1 << endl;
    cout << k2 << endl;
    cout << k3 << endl;
    cout << k4 << endl;
    cout << k5 << endl;

    return 0;
}
```

B的输入:

- 1、输入: 1300✓ (合理范围)
对应本例的k1=1300
- 2、输入: 2147483648✓ (超上限但未超同类型的u_short上限)
对应本例的k2=-2147483648
- 3、输入: 4294967297✓ (超上限且超过同类型的u_short上限)
对应本例的k3=1
- 4、输入: -1300✓ (合理范围)
对应本例的k4=-1300
- 5、输入: -2147483649✓ (超下限)
对应本例的k5=2147483647

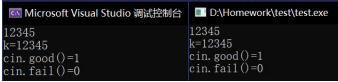
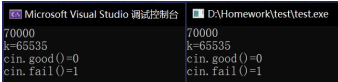
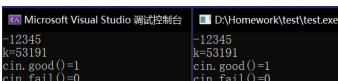
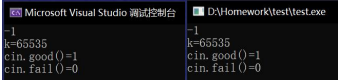
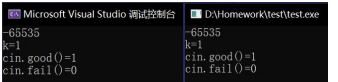
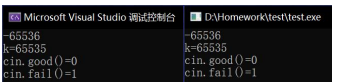
对比分析: cin输入时, 如果超出上限 (或下限), 会返回一个不可信值【VS与Dev是该数据类型上限 (或下限)】; 赋值时, 如果超出上限 (或下限), 会以二进制补码的形式存储, 再进行赋值、输出



§ . 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

D. 运行下面的程序，观察不同输入下的运行结果（贴图在清晰可辨的情况下尽可能小）

<pre>#include <iostream> using namespace std; int main() { unsigned short k; cin >> k; cout << "k=" << k << endl; cout << "cin.good()=" << cin.good() << endl; cout << "cin.fail()=" << cin.fail() << endl; return 0; }</pre>	<p>贴图即可，不需要写分析结果</p> <ol style="list-style-type: none">1、输入：12345✓（合理范围） 2、输入：70000✓（超上限） 3、输入：-12345✓（负数但未超过short下限） 4、输入：-1✓（负数且未超过short下限） 	<p>u_short=unsigned short</p>
<p>结论：</p> <p>多个输入中，编号<u>2、6</u>输入的k值是不可信的</p>	<ol style="list-style-type: none">5、输入：-65535✓（负数且未超过u_short上限加负号后的下限） 6、输入：-65536✓（负数且超过u_short上限加负号后的下限） 	<p>本题要求VS+Dev</p>



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

D-Compare. 仿B-Compare, 构造对比程序 (cin输入与赋值, u_short型), 观察运行结果并与D的输出结果进行对比分析

<pre>#include <iostream> using namespace std; int main() { unsigned short k1, k2, k3, k4, k5, k6; k1 = 12345; k2 = 70000; k3 = -12345; k4 = -1; k5 = -65535; k6 = -65536; cout << k1 << endl; cout << k2 << endl; cout << k3 << endl; cout << k4 << endl; cout << k5 << endl; cout << k6 << endl; return 0; }</pre>	<p>B的输入:</p> <ul style="list-style-type: none">1、输入: 12345✓ (合理范围) 对应本例的k1=123452、输入: 70000✓ (超上限) 对应本例的k2=44643、输入: -12345✓ (负数但未超过short下限) 对应本例的k3=531914、输入: -1✓ (负数且未超过short下限) 对应本例的k4=655355、输入: -65535✓ (负数且未超过u_short上限加负号后的下限) 对应本例的k5=16、输入: -65536✓ (负数且超过u_short上限加负号后的下限) 对应本例的k6=0 <p>对比分析: 1、cin输入时, 如果超出上限 (或u_short上限加负号后的下限), 会返回一个不可信值【VS与Dev是该数据类型上限 (或u_short上限)】; 赋值时, 如果超出上限 (或u_short上限加负号后的下限), 会以二进制补码的形式存储, 再进行赋值、输出</p> <p>2、如果为负数但未超过u_short上限加负号后的下限, 两种情况下都是以二进制补码的形式存储, 再进行赋值、输出 (cin输入的值显示可信)</p>
--	---



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

E. 仿D，自行构造不同测试数据，观察不同输入下的运行结果（贴图在清晰可辨的情况下尽可能小）

```
#include <iostream>
using namespace std;

int main()
{
    unsigned int k;
    cin >> k;
    cout << "k=" << k << endl;
    cout << "cin.good()=" << cin.good() << endl;
    cout << "cin.fail()=" << cin.fail() << endl;
    return 0;
}
```

结论：
多个输入中，编号2、5输入的k值是不可信的

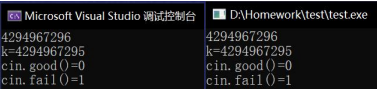
贴图即可，不需要写分析结果

u_int=unsigned int

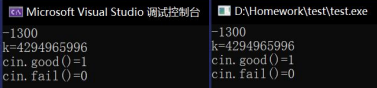
1、输入：1300✓（合理范围）



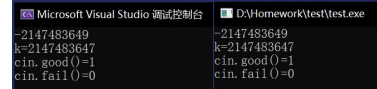
2、输入：4294967296✓（超上限）



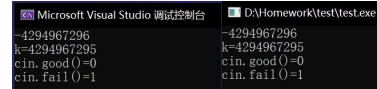
3、输入：-1300✓（负数但未超int下限）



5、输入：-2147483649✓（负数且未超过u_int上限加负号后的下限）



6、输入：-4294967296✓（负数且超过u_int上限加负号后的下限）



本题要求VS+Dev



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

E-Compare. 仿B-Compare, 构造对比程序 (cin输入与赋值, u_int型), 观察运行结果并与E的输出结果进行对比分析

<pre>#include <iostream> using namespace std; int main() { unsigned int k1, k2, k3, k4, k5; k1 = 1300; k2 = 4294967296; k3 = -1300; k4 = -2147483649; k5 = -4294967296; cout << k1 << endl; cout << k2 << endl; cout << k3 << endl; cout << k4 << endl; cout << k5 << endl; return 0; }</pre>	<p>B的输入:</p> <ul style="list-style-type: none">1、输入: 1300✓ (合理范围) 对应本例的k1=13002、输入: 4294967296✓ (超上限) 对应本例的k2=03、输入: -1300✓ (负数但未超int下限) 对应本例的k3=42949659964、输入: -2147483649✓ (负数且未超过u_int上限加负号后的下限) 对应本例的k4=21474836475、输入: -4294967296✓ (负数且超过u_short上限加负号后的下限) 对应本例的k5=0 <p>对比分析: 1、cin输入时, 如果超出上限 (或u_int上限加负号后的下限), 会返回一个不可信值【VS与Dev是该数据类型上限 (或u_int上限)】; 赋值时, 如果超出上限 (或u_int上限加负号后的下限), 会以二进制补码的形式存储, 再进行赋值、输出</p> <p>2、如果为负数但未超过u_int上限加负号后的下限, 两种情况下都是以二进制补码的形式存储, 再进行赋值、输出 (cin输入的值显示可信)</p>
--	---



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

B-E. 总结

名词解释:

输入正确 - 指数学上合法的数，但不代表一定在C/C++的某类型数据的数据范围内（下同）

综合2.B~2.E, 给出下列问题的分析及结论:

- 1、signed数据在输入正确且范围合理的情况下
正常输出
- 2、signed数据在输入正确但超上限（未超同类型unsigned上限）的情况下
以二进制补码的形式存储，再进行赋值、输出
- 3、signed数据在输入正确且超上限（超过同类型unsigned上限）的情况下
以二进制补码的形式存储，再进行赋值、输出
- 4、signed数据在输入正确但超下限范围的情况下
输出下限
- 5、unsigned数据在输入正确且范围合理的情况下
正常输出
- 6、unsigned数据在输入正确且超上限的情况下
输出上限
- 7、unsigned数据在输入正确但为负数（未超同类型signed下限）的情况下
以二进制补码的形式存储，再进行赋值、输出
- 8、unsigned数据在输入正确且为负数（超过同类型signed下限）的情况下
以二进制补码的形式存储，再进行赋值、输出
- 9、unsigned数据在输入正确且为负数（超过同类型unsigned上限加负号后的下限）的情况下
输出同类型unsigned上限

对比: cin输入与变量赋值，在输入/右值超范围的情况下，表现是否相同？总结规律

signed数据: 超上限相同，超下限不同；unsigned数据: 超下限相同，超上限不同。

cin输入与变量赋值，在输入/右值合理范围的情况下，表现是否相同？总结规律

两者表现相同

§. 基础知识题 - cin与cout的基本使用



2、cin的基本理解 - 单数据情况

F. 仿照short/int型，自行构造测试程序和测试数据，来验证char/unsigned char型数据的输入，至少要涉及如下知识点：

- a) 单个图形字符的输入（例如：char c; 能否键盘输入使c得到'A'）
- b) 能否输入\r\n\b等转义符（例如：char c; 能否键盘输入\b使c得到退格键的ASCII，想想，怎么得到c的ASCII码值？）
- c) 能否输入***及\x**形式的转义符（例如：char c; 能否键盘输入\101使c得到'A'）
- d) 能否以数字形式输入ascii，使char型变量得到对应ascii字符（例如：char c; 能否键盘输入65使c得到'A'）
- e) 和char/unsigned char型变量赋值的情况进行对比

注：测试程序及测试数据的构造要求（下同）

- 1) 每组测试，为了涵盖所有情况，允许多页
- 2) 每页一个完整的测试程序（字体最小12，超过则将测试程序分开）
- 3) 每个测试程序要配合多组测试数据
- 4) 在涵盖所有测试可能的情况下，尽量使页数少
- 5) 测试结论，可以写在每页上，也可以多页后进行单独的总结

本页不要再贴测试程序了，下页开始



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

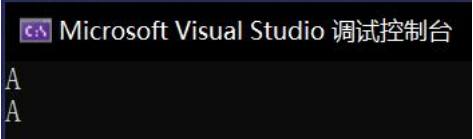
F. 仿照short/int型，自行构造测试程序和测试数据，来验证char/unsigned char型数据的输入，至少要涉及如下知识点：

a) 单个图形字符的输入（例如：char c; 能否键盘输入使c得到'A'）

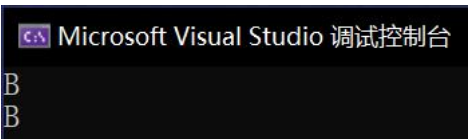
```
#include <iostream>
using namespace std;

int main()
{
    char c;
    cin >> c;
    cout << c << endl;
    return 0;
}
```

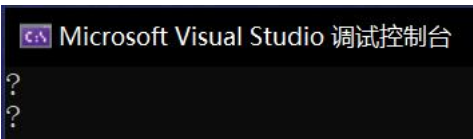
1、输入：A



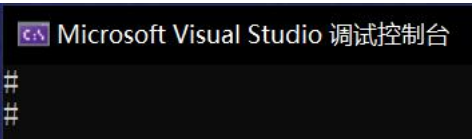
2、输入：B



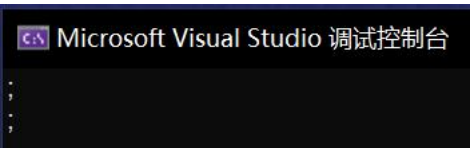
3、输入：?



4、输入：#



5、输入：;



结论：可以键盘输入使c的值输出为'A'



§. 基础知识题 - cin与cout的基本使用

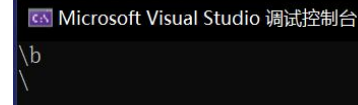
2、cin的基本理解 - 单数据情况

F. 仿照short/int型，自行构造测试程序和测试数据，来验证char/unsigned char型数据的输入，至少要涉及如下知识点：

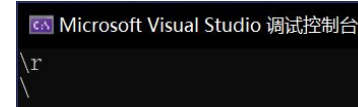
b) 能否输入\r\n\b等转义符（例如：char c; 能否键盘输入\b使c得到退格键的ASCII，想想，怎么得到c的ASCII码值？）

```
#include <iostream>
using namespace std;
int main()
{
    char c;
    cin >> c;
    cout << c << endl;
    return 0;
}
```

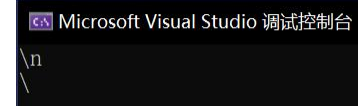
1、输入：\b



2、输入：\r



3、输入：\n



结论：无法输入转义符使c得到对应的ASCII，且这种情况下只能得到符号\的ASCII。因为输入\时，\被认为是图形字符，缓冲区已满，流提取运算符只提取了\，未提取后边的字母，将其对应的ASCII码值赋给c，故只能输出符号\。



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

F. 仿照short/int型，自行构造测试程序和测试数据，来验证char/unsigned char型数据的输入，至少要涉及如下知识点：

c) 能否输入***及\x**形式的转义符（例如：char c; 能否键盘输入\101使c得到'A'）

```
#include <iostream>
using namespace std;

int main()
{
    char c;
    cin >> c;
    cout << c << endl;
    return 0;
}
```

程序1:

1、输入：\101

2、输入：\x41

3、输入：\134

4、输入：\x5c

5、输入：\077

6、输入：\x3f

结论：无法输入转义符使c得到对应的ASCII，且这种情况下只能得到符号\的ASCII。因为输入\时，\被认为是图形字符，缓冲区已满，流提取运算符只提取了\，未提取后边的字母，将其对应的ASCII码值赋给c，故只能输出符号\。



§. 基础知识题 - cin与cout的基本使用

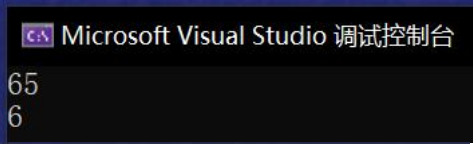
2、cin的基本理解 - 单数据情况

F. 仿照short/int型，自行构造测试程序和测试数据，来验证char/unsigned char型数据的输入，至少要涉及如下知识点：

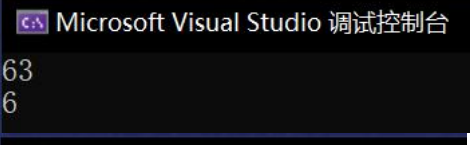
d) 能否以数字形式输入ascii，使char型变量得到对应ascii字符（例如：char c；能否键盘输入65使c得到'A'）

```
#include <iostream>
using namespace std;
int main()
{
    char c;
    cin >> c;
    cout << c << endl;
    return 0;
}
```

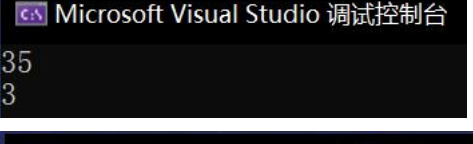
1、输入：65



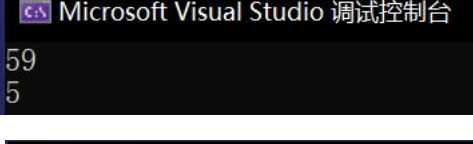
2、输入：63



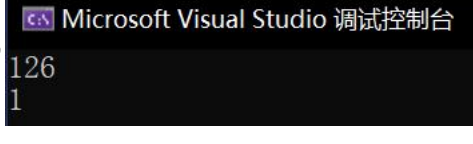
3、输入：35



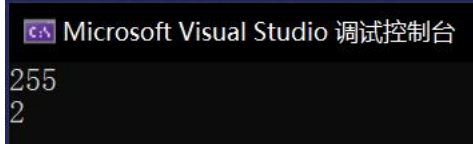
4、输入：59



5、输入：126



6、输入：255



结论：无法输入ASCII使c得到对应的ASCII，且这种情况下只能得到ASCII码第一位数字的ASCII。因为输入ASCII码时，这串数字被认为是一个个图形字符，缓冲区已满，流提取运算符只提取了第一位数字，未提取后边的数字，将其对应的ASCII码值赋给c，故只能输出ASCII码第一位数字。



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

F. 仿照short/int型，自行构造测试程序和测试数据，来验证char/unsigned char型数据的输入，至少要涉及如下知识点：

e) 和char/unsigned char型变量赋值的情况进行对比

```
#include <iostream>
using namespace std;

int main()
{
    char c1, c2, c3, c4, c5;

    c1 = 'A';
    c2 = '\n';
    c3 = '\101';
    c4 = '\x41';
    c5 = 65;

    cout << c1 << endl;
    cout << c2 << endl;
    cout << c3 << endl;
    cout << c4 << endl;
    cout << c5 << endl;

    return 0;
}
```

B的输入：

- 1、输入：'A' ✓ （图形字符）
对应本例的k1=A
- 2、输入：'\n' ✓ （转义字符）
对应本例的k2=换行符
- 3、输入：'\101' ✓ （八进制转义字符）
对应本例的k3=A
- 4、输入：'\x41' ✓ （十六进制转义字符）
对应本例的k4=A
- 5、输入：65 ✓ （数字形式的ASCII）
对应本例的k5=A

对比分析：cin输入时，只能通过键盘输入使c得到目标图形字符，无法通过键盘输入任何形式的转义字符或以数字形式输入ASCII，使变量得到目标图形字符。赋值时，通过上述方法进行赋值都可行。



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

F. 仿照short/int型，自行构造测试程序和测试数据，来验证char/unsigned char型数据的输入，至少要涉及如下知识点：

e) 和char/unsigned char型变量赋值的情况进行对比

```
#include <iostream>
using namespace std;

int main()
{
    unsigned char c1, c2, c3, c4,
    c5;

    c1 = '?';
    c2 = '\n';
    c3 = '\077';
    c4 = '\x3f';
    c5 = 63;

    cout << c1 << endl;
    cout << c2 << endl;
    cout << c3 << endl;
    cout << c4 << endl;
    cout << c5 << endl;

    return 0;
}
```

B的输入：

- 1、输入：'?' ✓ （图形字符）
对应本例的k1=?
- 2、输入：'\n' ✓ （转义字符）
对应本例的k2=换行符
- 3、输入：'\077' ✓ （八进制转义字符）
对应本例的k3=?
- 4、输入：'\x3f' ✓ （十六进制转义字符）
对应本例的k4=?
- 5、输入：63 ✓ （数字形式的ASCII）
对应本例的k5=?

对比分析：cin输入时，只能通过键盘输入使c得到目标图形字符，无法通过键盘输入任何形式的转义字符或以数字形式输入ASCII，使变量得到目标图形字符。赋值时，通过上述方法进行赋值都可行。

§. 基础知识题 - cin与cout的基本使用



2、cin的基本理解 - 单数据情况

G. 仿照short/int型，自行构造测试程序和测试数据，来验证float型和double型数据的输入，至少要涉及如下知识点：

- a) 小数形式：输入正确且范围合理、输入正确但超上下限范围(超上下限只做float即可)、输入错误
- b) 指数形式：输入正确且范围合理、输入正确但超上下限范围、输入错误
- c) 输入的有效位数超过该类型数据有效位数的情况下是如何处理的
- d) 和float/double型变量赋值的情况进行对比



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

G. 仿照short/int型，自行构造测试程序和测试数据，来验证float型和double型数据的输入，至少要涉及如下知识点：

a) 小数形式：输入正确且范围合理、输入正确但超上下限范围(超上下限只做float即可)、输入错误

```
#include <iostream>
using namespace std;
int main()
{
    float k;
    cin >> k;
    cout << "k=" << k << endl;
    cout << "cin.good()=" << cin.good() << endl;
    cout << "cin.fail()=" << cin.fail() << endl;
    return 0;
}
```

结论:

多个输入中，编号2、3、4输入的k值是不可信的

1、输入：12345.6✓（合理范围）

```
12345.6
k=12345.6
cin.good()=1
cin.fail()=0
```

2、输入：341000000000000000000000000000000000000.0 ✓ （超上限）

[illegible][illegible][illegible]

4、输入：m↙ (输入错误)

```
m
k=0
cin.good()=0
cin.fail()=1
```



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

G. 仿照short/int型，自行构造测试程序和测试数据，来验证float型和double型数据的输入，至少要涉及如下知识点：

a) 小数形式：输入正确且范围合理、输入正确但超上下限范围(超上下限只做float即可)、输入错误

```
#include <iostream>
using namespace std;
int main()
{
    double k;
    cin >> k;
    cout << "k=" << k << endl;
    cout << "cin.good()=" << cin.good() << endl;
    cout << "cin.fail()=" << cin.fail() << endl;
    return 0;
}
```

结论：

多个输入中，编号2输入的k值是不可信的

1、输入：12345.6✓ （合理范围）

```
Microsoft Visual Studio 调试控制台
12345.6
k=12345.6
cin.good()=1
cin.fail()=0
```

2、输入：m✓ （输入错误）

```
Microsoft Visual Studio 调试控制台
m
k=0
cin.good()=0
cin.fail()=1
```



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

G. 仿照short/int型，自行构造测试程序和测试数据，来验证float型和double型数据的输入，至少要涉及如下知识点：

b) 指数形式：输入正确且范围合理、输入正确但超上下限范围、输入错误

```
#include <iostream>
using namespace std;
int main()
{
    float k;
    cin >> k;
    cout << "k=" << k << endl;
    cout << "cin.good()=" << cin.good() << endl;
    cout << "cin.fail()=" << cin.fail() << endl;
    return 0;
}
```

结论：

多个输入中，编号2、3、4输入的k值是不可信的

1、输入：1.23456e+10✓ （合理范围）

```
Microsoft Visual Studio 调试控制台
1.23456e+10
k=1.23456e+10
cin.good()=1
cin.fail()=0
```

2、输入：3.41e+38（超上限）

```
Microsoft Visual Studio 调试控制台
3.41e+38
k=0
cin.good()=0
cin.fail()=1
```

3、输入：-3.41e+38（超下限）

```
Microsoft Visual Studio 调试控制台
-3.41e+38
k=0
cin.good()=0
cin.fail()=1
```

4、输入：m✓ （输入错误）

```
Microsoft Visual Studio 调试控制台
m
k=0
cin.good()=0
cin.fail()=1
```



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

G. 仿照short/int型，自行构造测试程序和测试数据，来验证float型和double型数据的输入，至少要涉及如下知识点：

b) 指数形式：输入正确且范围合理、输入正确但超上下限范围、输入错误

```
#include <iostream>
using namespace std;
int main()
{
    double k;
    cin >> k;
    cout << "k=" << k << endl;
    cout << "cin.good()=" << cin.good() << endl;
    cout << "cin.fail()=" << cin.fail() << endl;
    return 0;
}
```

结论：

多个输入中，编号2、3、4输入的k值是不可信的

1、输入：1.23456e+10✓（合理范围）

```
Microsoft Visual Studio 调试控制台
1.23456e+10
k=1.23456e+10
cin.good()=1
cin.fail()=0
```

2、输入：1.8e+308（超上限）

```
Microsoft Visual Studio 调试控制台
1.8e+308
k=0
cin.good()=0
cin.fail()=1
```

3、输入：-1.8e+308（超下限）

```
Microsoft Visual Studio 调试控制台
-1.8e+308
k=0
cin.good()=0
cin.fail()=1
```

4、输入：m✓（输入错误）

```
Microsoft Visual Studio 调试控制台
m
k=0
cin.good()=0
cin.fail()=1
```



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

G. 仿照short/int型，自行构造测试程序和测试数据，来验证float型和double型数据的输入，至少要涉及如下知识点：

c) 输入的有效位数超过该类型数据有效位数的情况下是如何处理的

```
#include <iostream>
using namespace std;
int main()
{
    float k;
    cin >> k;
    cout << "k=" << k << endl;
    cout << "cin.good()=" << cin.good() << endl;
    cout << "cin.fail()=" << cin.fail() << endl;
    return 0;
}
```

结论：

输入的有效位数超过float型数有效位数的情况下，无论是小数或指数型，都按四舍五入的原则去掉多余位数再输出。

1、输入：1.23456e+10✓（合理范围）

```
Microsoft Visual Studio 调试控制台
1.23456e+10
k=1.23456e+10
cin.good()=1
cin.fail()=0
```

2、输入：1.234565（超过float有效位数且为小数型）

```
Microsoft Visual Studio 调试控制台
1.234565
k=1.23457
cin.good()=1
cin.fail()=0
```

3、输入：1.234564（超过float有效位数且为小数型）

```
Microsoft Visual Studio 调试控制台
1.234564
k=1.23456
cin.good()=1
cin.fail()=0
```

4、输入：1.234565e+10（超过float有效位数且为指数型）

```
Microsoft Visual Studio 调试控制台
1.234565e+10
k=1.23457e+10
cin.good()=1
cin.fail()=0
```

5、输入：1.234564e+10（超过float有效位数且为指数型）

```
Microsoft Visual Studio 调试控制台
1.234564e+10
k=1.23456e+10
cin.good()=1
cin.fail()=0
```



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

G. 仿照short/int型，自行构造测试程序和测试数据，来验证float型和double型数据的输入，至少要涉及如下知识点：

c) 输入的有效位数超过该类型数据有效位数的情况下是如何处理的

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
    double k;
    cin >> k;
    cout << "k=" << k << endl;
    cout << "cin.good()=" << cin.good() << endl;
    cout << "cin.fail()=" << cin.fail() << endl;
    return 0;
}
```

结论：

cout默认输出6位有效数字。尽管double有效位数大于6位，但无论是小数或指数型，只要输入的有效数字大于6位，都按四舍五入的原则去掉多余位数再输出。本例中与是否超出double有效位数无关。

1、输入：1.23456e+10✓（合理范围）

```
Microsoft Visual Studio 调试控制台
1.23456e+10
k=1.23456e+10
cin.good()=1
cin.fail()=0
```

2、输入：1.234567890123456（超过double有效位数且为小数型）

```
Microsoft Visual Studio 调试控制台
1.234567890123456
k=1.23457
cin.good()=1
cin.fail()=0
```

3、输入：1.234567890123456e+10（超过double有效位数且为指数型）

```
Microsoft Visual Studio 调试控制台
1.234567890123456e+10
k=1.23457e+10
cin.good()=1
cin.fail()=0
```

4、输入：1.234567（超过cin默认输出有效位数且为小数型）

```
Microsoft Visual Studio 调试控制台
1.234567
k=1.23457
cin.good()=1
cin.fail()=0
```

5、输入：1.234567e+10（超过cin默认输出有效位数且为指数型）

```
Microsoft Visual Studio 调试控制台
1.234567e+10
k=1.23457e+10
cin.good()=1
cin.fail()=0
```




§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

G. 仿照short/int型，自行构造测试程序和测试数据，来验证float/double型数据的输入，至少要涉及如下知识点：

d) 和float/double型变量赋值的情况进行对比

```
#include <iostream>
using namespace std;
int main()
{
    double c1, c2, c3, c4, c5;

    c1 = 12345.6;
    c2 = 1.8e+308;
    c3 = -1.8e+308;
    c4 = 1.234567890123456;
    c5 = 1.234567890123456e+10;

    cout << c1 << endl;
    cout << c2 << endl;
    cout << c3 << endl;
    cout << c4 << endl;
    cout << c5 << endl;

    return 0;
}
```

B的输入：

1、输入：12345.6✓ （合理范围）
对应本例的k1=12345.6

2、输入：1.8e+308✓ （超上限）
对应本例的k2=（无）

3、输入：-1.8e+308✓ （超下限）
对应本例的k3=（无）

4、输入：1.234567890123456✓ （超过有效位数小数型）
对应本例的k4=1.23457

5、输入：1.234567890123456e+10✓ （超过有效位数指数型）
对应本例的k5=1.23457e+10

对比：1、cin输入时，如果超出上限（或下限），会返回一个不可信值0；赋值时，如果超出上限（或下限），程序会报error，甚至无法给变量赋值。
2、两种情况下，cout都默认输出6位有效数字。无论输入的有效位数是否超过double型数有效位数，只要超过6位有效数字，都按四舍五入的原则去掉多余位数再输出。



§. 基础知识题 - cin与cout的基本使用

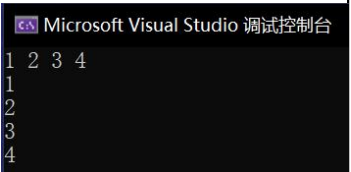
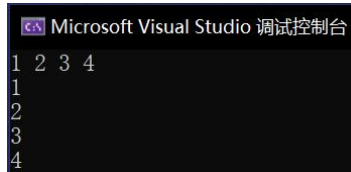
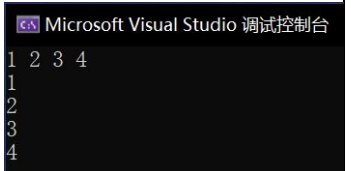
此页不要删除，也没有意义，仅仅为了分隔题目



§. 基础知识题 - cin与cout的基本使用

3、cin的基本理解 - 多个同类型数据的情况

A. 观察下列3个程序的运行结果，回答问题并将各程序的运行结果截图贴上(如果有错则贴错误信息截图)

<pre>#include <iostream> using namespace std; int main() { int a, b, c, d; cin >> a >> b >> c >> d; cout << a << endl; cout << b << endl; cout << c << endl; cout << d << endl; return 0; }</pre> 	<pre>#include <iostream> using namespace std; int main() { int a, b, c, d; cin >> a >> b >> c >> d; cout << a << endl; cout << b << endl; cout << c << endl; cout << d << endl; return 0; }</pre> 	<pre>#include <iostream> using namespace std; int main() { int a, b, c, d; cin >> a; cin >> b; cin >> c; cin >> d; cout << a << endl; cout << b << endl; cout << c << endl; cout << d << endl; return 0; }</pre> 
--	--	---

1、程序运行后，输入：1 2 3 4↵，观察输出结果

2、解释第2个和第3个程序的cin语句的使用区别：第2个程序使用了一个cin语句，总共有一个分号；第3个程序使用了四个cin语句，总共四个分号。



§. 基础知识题 - cin与cout的基本使用

3、cin的基本理解 - 多个同类型数据的情况

B. 程序同A，观察不同输入下的运行结果（贴图在清晰可辨的情况下尽可能小）

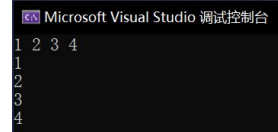
```
#include <iostream>
using namespace std;

int main()
{
    int a, b, c, d;
    cin >> a >> b >> c >> d;

    cout << a << endl;
    cout << b << endl;
    cout << c << endl;
    cout << d << endl;

    return 0;
}
```

1、输入：1 2 3 4✓



2、输入：1 2 3 4✓（每个数字间多于一个空格）



3、输入：1✓

2✓

3✓

4✓（每个数字后立即加回车）



4、输入：1✓

✓

✓

2✓

✓

3✓

✓

4✓（每个数字后立即加回车 + 多个空回车）



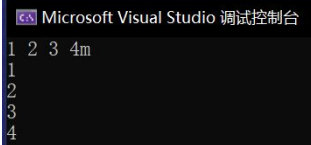



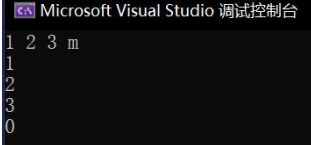


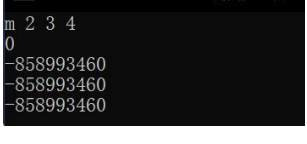
结论：在输入正确的情况下，回车和空格的作用？
作为输入终止条件使输入停止



§. 基础知识题 - cin与cout的基本使用

3、cin的基本理解 - 多个同类型数据的情况

C. 程序同A，观察不同输入下的运行结果（贴图在清晰可辨的情况下尽可能小）

<pre>#include <iostream> using namespace std; int main() { int a, b, c, d; cin >> a >> b >> c >> d; cout << a << endl; cout << b << endl; cout << c << endl; cout << d << endl; return 0; }</pre>	1、输入：1 2 3 4m✓		2、输入：1 2 3m 4✓	
	3、输入：1 2m 3 4✓		4、输入：1m 2 3 4✓	
	5、输入：1 2 3 m✓		6、输入：1 2 m 4✓	
	7、输入：1 m 3 4✓		8、输入：m 2 3 4✓	
	总结：多个cin输入时，错误输入出现在不同位置对输入正确性的影响			
	1、错误输入量以正确输入部分开头：若错误输入前有正确输入，则这些正确输入照常输出；错误输入量的正确输入部分照常输出；若错误输入后有正确输入，第一个正确输入输出为0，其他正确输入输出为一个不可信值。			
	2、错误输入量以错误输入部分开头：若错误输入前有正确输入，则这些正确输入照常输出；错误输入输出为0；若错误输入后有正确输入，全部输出为一个不可信值			



§ . 基础知识题 - cin与cout的基本使用

3、cin的基本理解 - 多个同类型数据的情况

D. 仿BC，自己构造测试程序及测试数据，观察多个char型数据在不同输入下的运行结果（贴图在清晰可辨的情况下尽可能小）

<pre>#include <iostream> using namespace std; int main() { char a, b, c, d; cin >> a >> b >> c >> d; cout << a << endl; cout << b << endl; cout << c << endl; cout << d << endl; return 0; }</pre>	1、输入：A B C DEF✓ 	2、输入：A B CEF D✓
	3、输入：A BEF C D✓ 	4、输入：AEF B C D✓
	5、输入：A B C ,. ✓ 	6、输入：A B ,. D✓
	7、输入：A ,. C D✓ 	8、输入：,. B C D✓
	总结：多个cin输入时，错误输入出现在不同位置对输入正确性的影响	
	1、对于错误输入，会将构成其的每一个字符依次提取，赋给不同的字符型变量。	
	2、若错误输入前有正确输入，则这些正确输入照常输出。	
	3、若错误输入后有正确输入，且错误输入被提取完成后，仍有变量未赋值的，跟在后面正常输出，到最后一个被赋值的变量为止；若错误输入后有正确输入，且错误输入未完成（或刚好完成）时所有变量都被赋值，则只输出到最后一个被赋值的变量。	



§. 基础知识题 - cin与cout的基本使用

3、cin的基本理解 - 多个同类型数据的情况

E. 仿BC，自己构造测试程序及测试数据，观察多个float/double型数据在不同输入下的运行结果(含正确/错误情况)
(贴图在清晰可辨的情况下尽可能小)

```
#include <iostream>
using namespace std;
int main()
{
    float a, b, c, d;
    cin >> a >> b >> c >> d;

    cout << a << endl;
    cout << b << endl;
    cout << c << endl;
    cout << d << endl;

    return 0;
}

#include <iostream>
using namespace std;
int main()
{
    double a, b, c, d;
    cin >> a >> b >> c >> d;

    cout << a << endl;
    cout << b << endl;
    cout << c << endl;
    cout << d << endl;

    return 0;
}
```

1、输入: 1.1 2.2 3.3 4.4m✓

3、输入: 1.1 2.2m 3.3 4.4✓

5、输入: 1.1 2.2 3.3 m✓

7、输入: 1.1 m 3.3 4.4✓

2、输入: 1.1 2.2 3.3m 4.4✓

4、输入: 1.1m 2.2 3.3 4.4✓

6、输入: 1.1 2.2 m 4.4✓

8、输入: m 2.2 3.3 4.4✓

总结: 多个cin输入时, 错误输入出现在不同位置对输入正确性的影响

1、错误输入量以正确输入部分开头: 若错误输入前有正确输入, 则这些正确输入照常输出; 错误输入量的正确输入部分照常输出; 若错误输入后有正确输入, 第一个正确输入输出为0, 其他正确输入输出为一个不可信值。

2、错误输入量以错误输入部分开头: 若错误输入前有正确输入, 则这些正确输入照常输出; 错误输入输出为0; 若错误输入后有正确输入, 全部输出为一个不可信值



§. 基础知识题 - cin与cout的基本使用

此页不要删除，也没有意义，仅仅为了分隔题目



§. 基础知识题 - cin与cout的基本使用

4、cin的基本理解 - 其他情况

A. 程序如下，观察编译及运行结果（贴图在清晰可辨的情况下尽可能小）

```
#include <iostream>
using namespace std;
int main()
{
    int a, b, c;
    cin >> a,b,c;

    cout << a << endl;
    cout << b << endl;
    cout << c << endl;
    return 0;
}
```

- 1、如果编译有error或warning，则贴相应信息的截图
- 2、如果能运行(包括有warning)，则输入三个正确的int型数据（例 :1 2 3✓），观察输出
- 3、分析为什么只有某个变量的结果是正确的

代码	说明
⚠ C6001	使用未初始化的内存“b”。
⚠ C6001	使用未初始化的内存“c”。
✖ C4700	使用了未初始化的局部变量“b”
✖ C4700	使用了未初始化的局部变量“c”



分析：流提取运算符只提取了变量a的值，输入语句后是逗号表达式，b, c 的值并未输入，故输出一个不可信值

本题要求VS+Dev



§. 基础知识题 - cin与cout的基本使用

4、cin的基本理解 - 其他情况

B. 程序如下，观察编译及运行结果（贴图在清晰可辨的情况下尽可能小）

```
#include <iostream>
using namespace std;
int main()
{
    int a=66, b=67, c=68;
    cin >> a,b,c;

    cout << a << endl;
    cout << b << endl;
    cout << c << endl;
    return 0;
}
```

1、运行后，输入三个正确的int型数据(例 :1 2 3✓，注意不要是预置值)，观察输出

Microsoft Visual Studio 调试控制台

```
1 2 3
1
67
68
```

2、通过观察三个变量的输出，你得到了什么结论？

结论：cin输入可以覆盖变量的预置值，流提取运算符只提取了变量a的cin输入值，输入语句后是逗号表达式，b,c的值并未输入，故只有变量a的值被覆盖，其余变量依然输出为预置值。



§. 基础知识题 - cin与cout的基本使用

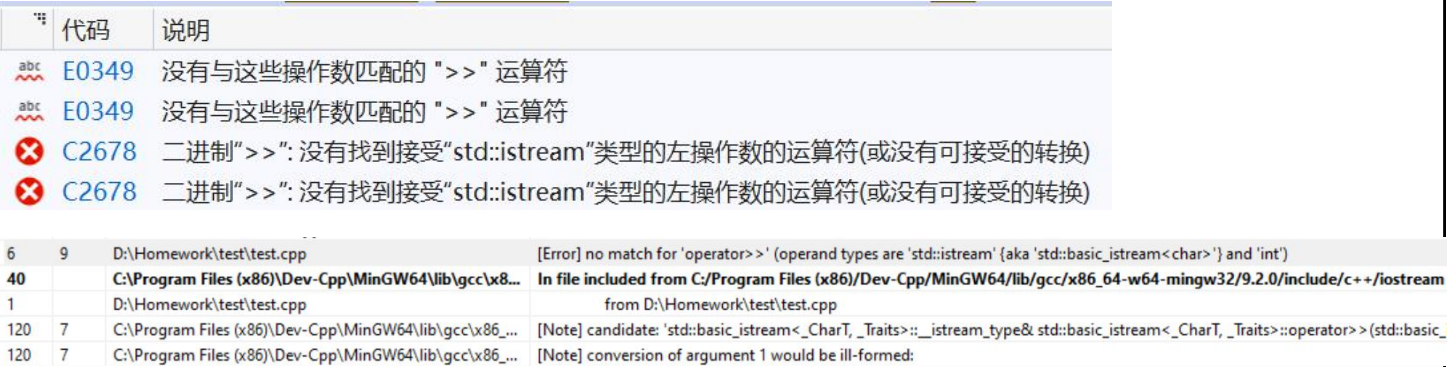
4、cin的基本理解 - 其他情况

C. 程序如下，观察编译及运行结果（贴图在清晰可辨的情况下尽可能小）

```
#include <iostream>
using namespace std;
int main()
{
    int a;
    cin >> 5;
    cin >> a+10;

    cout << a << endl;
    return 0;
}
```

1、如果编译有error或warning，则贴相应信息的截图(信息太多则前五五行)



2、分析为什么编译有错

分析：流提取运算符后面只能跟变量，跟常量与表达式会造成语法错误。

3、结论：流提取运算符后面必须跟b，不能是ac
a) 常量 b) 变量 c) 表达式

本题要求VS+Dev



§. 基础知识题 - cin与cout的基本使用

4、cin的基本理解 - 其他情况

D. 程序如下，观察编译及运行结果（贴图在清晰可辨的情况下尽可能小）

```
#include <iostream>
using namespace std;
int main()
{
    int a=66, b=67, c=68;
    cin >> (a,b,c);

    cout << a << endl;
    cout << b << endl;
    cout << c << endl;
    return 0;
}
```

1、运行后，输入三个正确的int型数据(例 :1 2 3✓，注意不要是预置值)，观察输出

Microsoft Visual Studio 调试控制台

```
1 2 3
66
67
1
```

2、通过观察三个变量的输出，你得到了什么结论？

结论：cin输入可以覆盖变量的预置值，只要被流提取运算符提取cin输入值的变量都可被覆盖，输出为输入值而不是预置值

3、和B进行比较，分析为什么结果有差异

分析：该语句先进行逗号表达式(a, b, c)的运算，得到结果为c，流提取运算符再提取输入的第一个数1（其余数由于有空格存在未被存入缓冲区）赋值给c，则c输出为输入值而不是预置值，其余变量输出为预置值。

4、和C进行比较，与C得出的结论矛盾吗？

答：不矛盾。虽然本例中流提取运算符后跟的是表达式，但表达式的结果依旧是单个变量，符合流提取运算符的语法要求，流提取运算符依然可提取该变量的输入值。



§. 基础知识题 - cin与cout的基本使用

4、cin的基本理解 - 其他情况

E. 程序如下，观察编译及运行结果（贴图在清晰可辨的情况下尽可能小）

```
#include <iostream>
using namespace std;
int main()
{
    char c1, c2;
    int a;
    float b;
    cin >> c1 >> c2 >> a >> b;

    cout << c1 << ' ' << c2 << ' ' << a << ' ' << b << endl;
    return 0;
}
```

注：┐表示空格

1、输入：1234┐56.78✓
输出：1 2 34 56.78

2、输入：1┐2┐34┐56.78✓
输出：1 2 34 56.78

3、分析在以上两种不同输入的情况下，
为什么输出相同（提示：空格的作用）

分析：输入过程中，缓冲区满或遇到空格的时候，过程都会终止。本例中，第二种情况下遇到了空格，故输入终止，1赋值给c1，2赋值给c2；第一种情况下1已经占满了字符型变量的缓冲区，故输入也终止，1赋值给c1，2赋值给c2。综上，两种情况输出相同。



§. 基础知识题 - cin与cout的基本使用

4、cin的基本理解 - 其他情况

F. 程序如下，观察编译及运行结果（贴图在清晰可辨的情况下尽可能小）

```
#include <iostream>
using namespace std;
int main()
{
    int a;
    cin >> a >> endl;

    return 0;
}
```

1、如果编译有error或warning, 则贴相应信息的截图(信息太多则前五)

代码	说明
E0349	没有与这些操作数匹配的 ">>" 运算符
C2679	二元 ">>": 没有找到接受 "overloaded-function" 类型的右操作数的运算符(或没有可接受的转换)

6	14	D:\Homework\test\test.cpp	[Error] no match for 'operator>>' (operand types are 'std::basic_istream<char>::__istream_type' {aka 'std::basic_istream<char>'} and '<unresolved overloaded function type>')
40		C:\Program Files (x86)\Dev-Cpp\MinGW64\lib\gcc\x86_64-w64-mingw32\9.2.0\include\c++\iosstream	In file included from C:\Program Files (x86)\Dev-Cpp\MinGW64\lib\gcc\x86_64-w64-mingw32\9.2.0\include\c++\iosstream
		D:\Homework\test\test.cpp	from D:\Homework\test\test.cpp
120	7	C:\Program Files (x86)\Dev-Cpp\MinGW64\lib\gcc\x86_64-w64-mingw32\9.2.0\include\c++\iosstream	[Note] candidate: 'std::basic_istream<Char_T, Traits>::__istream_type& std::basic_istream<Char_T, Traits>::operator>>()' (std::basic_istream<Char_T, Traits>::__istream_type& (&)) (std::basic_istream<Char_T, Traits>::__istream_type& (&)) [with Char_T = char, Traits = std::basic_istream_traits<char>]
120	36	C:\Program Files (x86)\Dev-Cpp\MinGW64\lib\gcc\x86_64-w64-mingw32\9.2.0\include\c++\iosstream	[Note] no known conversion for argument 1 from '<unresolved overloaded function type>' to 'std::basic_istream<char>::__istream_type& (&)' (std::basic_istream<char>::__istream_type& (&)) (std::basic_istream<char>::__istream_type& (&)) [with Char_T = char, Traits = std::basic_istream_traits<char>]

2、结论：在cin中不能跟换行符

本题要求VS+Dev



§. 基础知识题 - cin与cout的基本使用

此页不要删除，也没有意义，仅仅为了分隔题目