

§ . 基础知识题 - cin与cout的基本使用



要求:

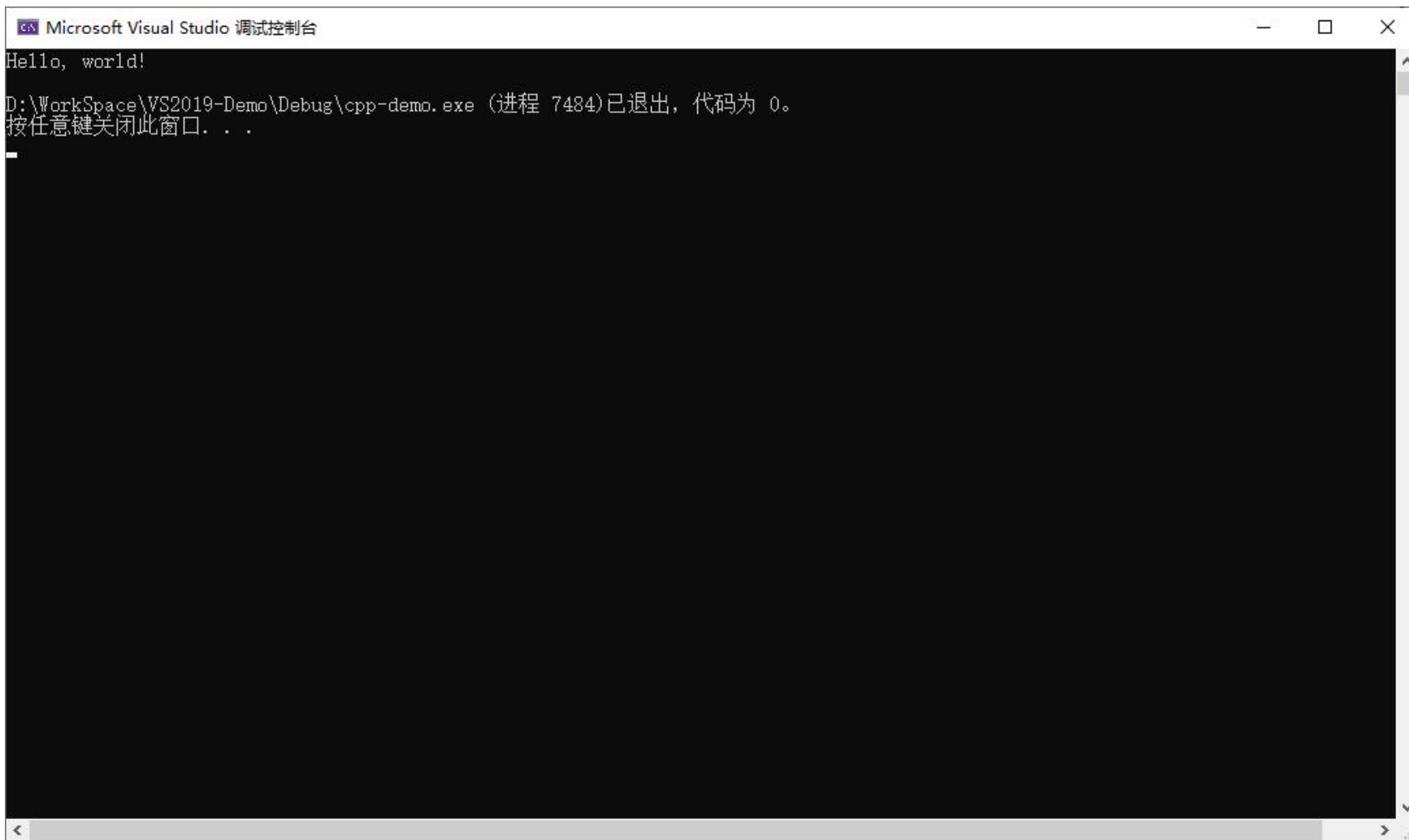
- 1、完成本文档中所有的题目并写出分析、运行结果
- 2、无特殊说明，均使用VS2022编译即可
- 3、直接在本文件上作答，**写出答案/截图（不允许手写、手写拍照截图）**即可；填写答案时，为适应所填内容或贴图，**允许调整**页面的字体大小、颜色、文本框的位置等
 - ★ 贴图要有效部分即可，不需要全部内容
 - ★ 在保证一页一题的前提下，具体页面布局可以自行发挥，简单易读即可
 - ★ **不允许**手写在纸上，再拍照贴图
 - ★ **允许**在各种软件工具上完成（不含手写），再截图贴图
 - ★ 如果某题要求VS+Dev的，则如果两个编译器运行结果一致，贴VS的一张图即可，如果不一致，则两个图都要贴
- 4、转换为pdf后提交
- 5、**9月22日前**网上提交本次作业（在“文档作业”中提交）

§. 基础知识题 - cin与cout的基本使用

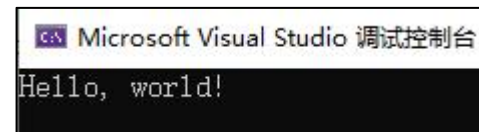


贴图要求：只需要截取输出窗口中的有效部分即可，如果全部截取/截取过大，则视为无效贴图

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

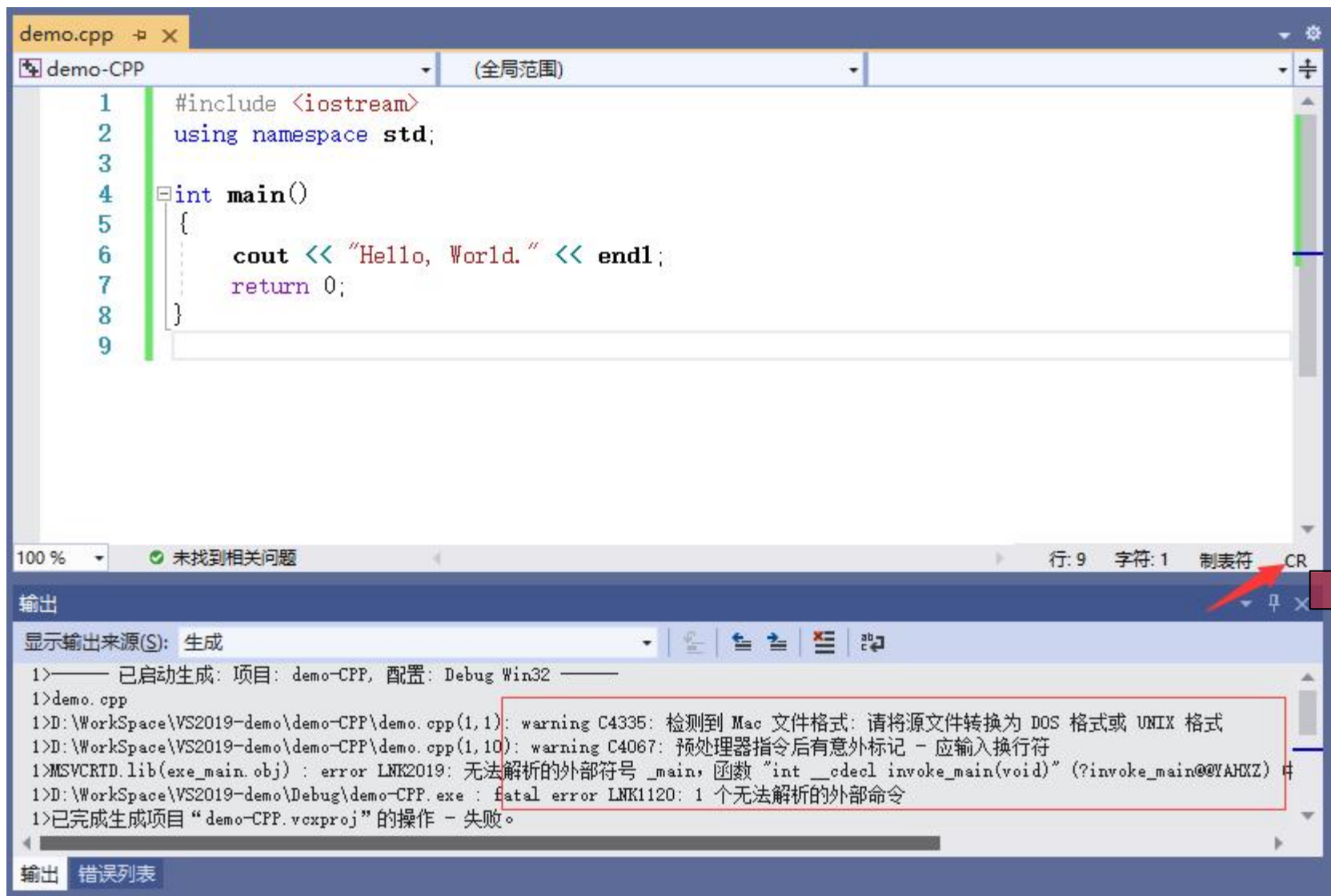
A screenshot of the Microsoft Visual Studio debug console window, cropped to show only the "Hello, world!" text. The window title "Microsoft Visual Studio 调试控制台" is visible at the top.



§. 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

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." ;

    /* 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;
}
```

2
3
4
5
6
7

Microsoft Visual Studio 输出窗口截图显示程序运行结果：

```
This is a C++ program.
This is a C++ program.
This is a C++ program.
This is a C++ program.
```

第3组和第4组在语句上的区别是：

第三句是一个语句，只不过用多行的形式进行书写

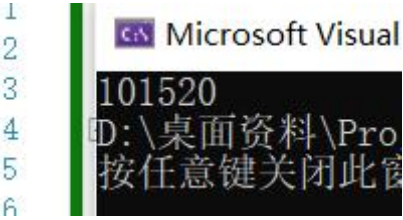
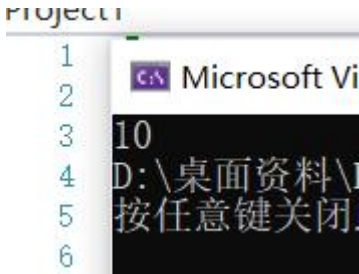
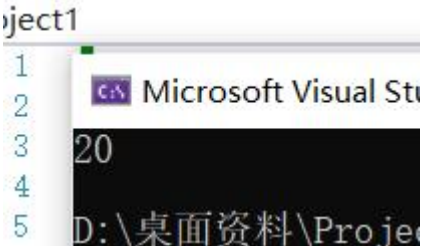
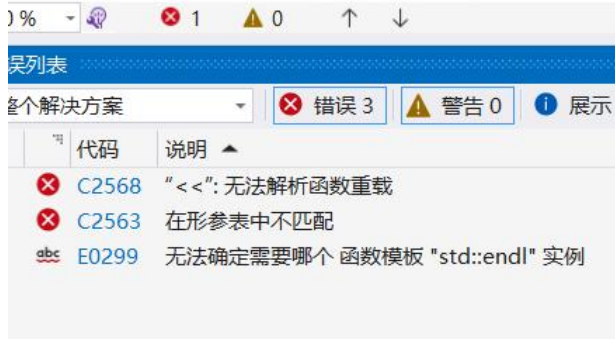
第四句是多个语句，在多行进行书写。两者起到的作用是相同的



§. 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>  | <pre>#include <iostream> using namespace std; int main() { int a=10, b=15, c=20; cout << a, b, c; return 0; }</pre>  | <pre>#include <iostream> using namespace std; int main() { int a=10, b=15, c=20; cout << (a, b, c) << endl; return 0; }</pre>  | <pre>#include <iostream> using namespace std; int main() { int a=10, b=15, c=20; cout << a, b, c << endl; return 0; }</pre>  |
| 解释这3个程序输出不同的原因： (1) 三个数据依次输出，所以是101520 (2) 由优先级的知识，先输出a的值，然后是一个逗号表达式，没有输出，所以是10。 (3) 先计算括号内的，然后输出括号的值，即为c的值，为20 | | | 解释错误原因：左移运算符优先级比逗号高，与c结合，流对象不对。 |
| 结论：一个流插入运算符 << 只能输出____1 ____个数据. | | | |

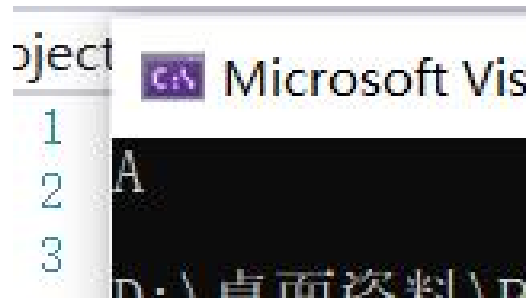


§. 基础知识题 - cin与cout的基本使用

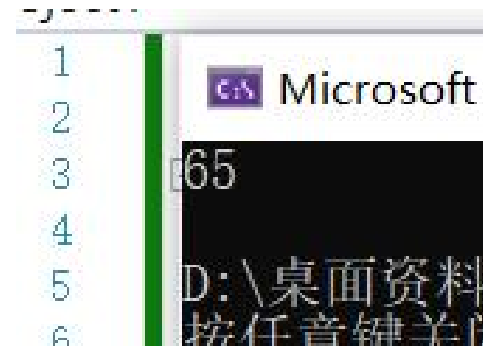
1、cout的基本理解

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：

在定义变量的时候两者不同，前一个是字符型的变量，后面的是整型变量，cout会根据变量的类型，用相应的形式进行输出。所以前一个是字符型的，后一个是整型的。

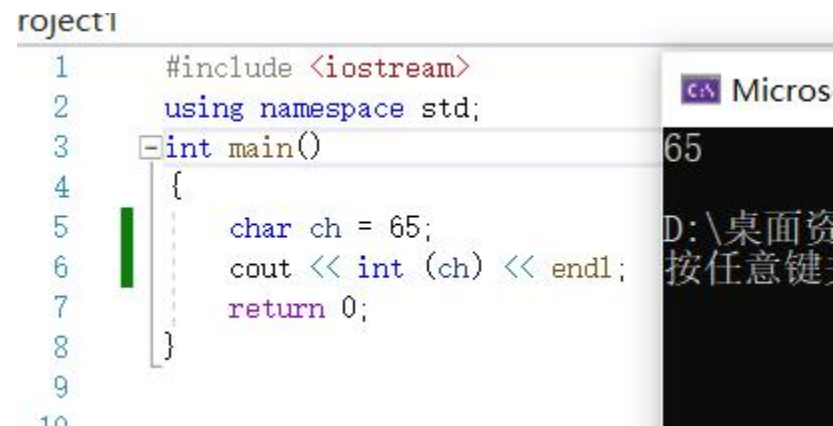


§. 基础知识题 - cin与cout的基本使用

1、cout的基本理解

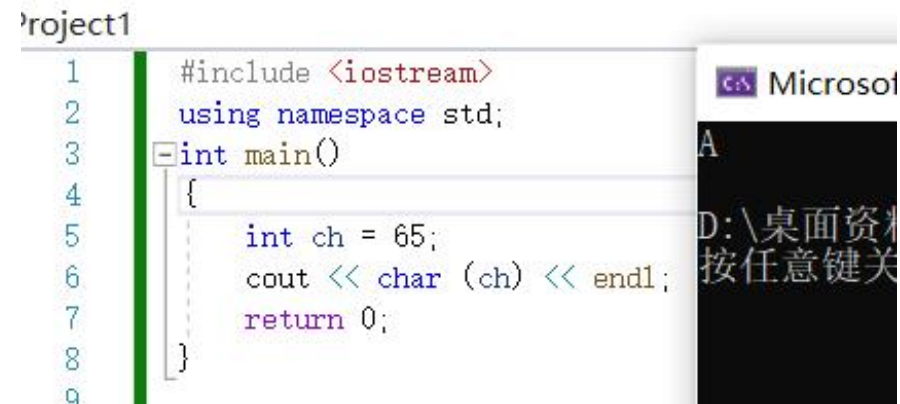
D. 程序同C，将修改后符合要求的程序及运行结果贴上

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



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)
将输出改为 cout<<int (ch)<<endl;

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



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)
将输出改为cout<<char(ch)<<endl;



§. 基础知识题 - cin与cout的基本使用

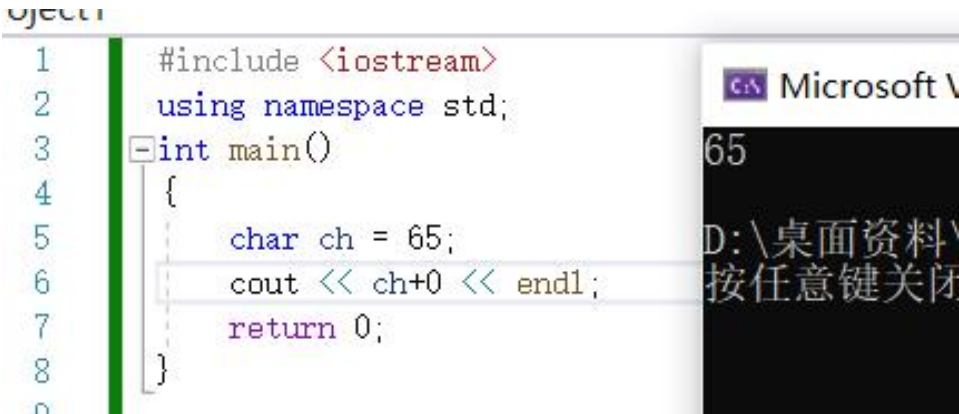
1、cout的基本理解

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 << endl;
    return 0;
}

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

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）

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

    cout << "k=" << k << endl;

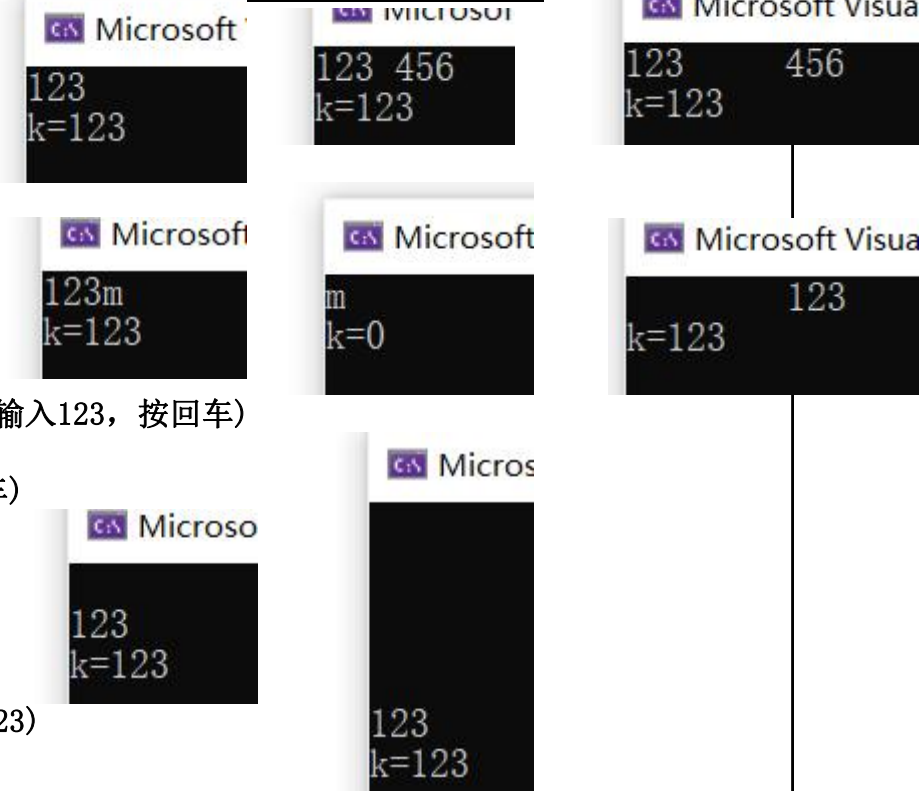
    return 0;
}
```

基础知识:

short的最小值是: -32768

short的最大值是: 32767

- 1、输入: 123✓ (✓代表回车键, 下同)
- 2、输入: 123 456✓ (一个空格)
- 3、输入: 123 456✓ (多个空格)
- 4、输入: 123m✓
- 5、输入: m✓
- 6、输入: 123✓ (持续多个空格后, 再输入123, 按回车)
- 7、输入: ✓ (持续多个空格后, 按回车)
 123✓ (再输入123, 按回车)
- 8、输入: ✓
 ...
 ✓
 123✓ (持续多个空回车后, 输入123)



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?
输入中止条件, 并不读取

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?
能得到正确的值

3、直接输入(对int型而言是)非法的数据m, 输出是?
输出是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

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值是可信的

贴图即可，不需要写分析结果

1、输入: 123 (正确+回车)

```
1 123
2 k=123
3 cin.good()=1
4 cin.fail()=0
5
```

```
D:\桌面资料\未命名2.exe
123
k=123
cin.good()=1
cin.fail()=0
```

2 输入: 123 456 (正确+空格)

```
Microsoft Visual S
123 456
k=123
cin.good()=1
cin.fail()=0
```

```
D:\桌面资料\未命名2.exe
123 456
k=123
cin.good()=1
cin.fail()=0
```

3、输入: -123m (正确+非法字符)

```
Microsoft Visual S
-123m
k=-123
cin.good()=1
cin.fail()=0
```

```
D:\桌面资料\未命名2.exe
-123m
k=-123
cin.good()=1
cin.fail()=0
```

4、输入: m (直接非法)

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

```
D:\桌面资料\未命名2.exe
m
k=0
cin.good()=0
cin.fail()=1
```

5、输入: 54321 (超上限)

```
Microsoft Visual S
54321
k=32767
cin.good()=0
cin.fail()=1
```

```
D:\桌面资料\未命名2.exe
54321
k=32767
cin.good()=0
cin.fail()=1
```

本题要求VS+Dev

```
Microsoft Visual S
-40000
k=-32768
cin.good()=0
cin.fail()=1
```

```
D:\桌面资料\未命名2.exe
-40000
k=-32768
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

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;
}
```

12345
-11215
4464
-12345
11215

B的输入:

1、输入: 12345✓ (合理范围)
对应本例的k1= 12345

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= 32767

54321

3、输入: 70000✓ (超上限且超过同类型的u_short上限)
对应本例的k3= 32767

70000

4、输入: -12345✓ (合理范围)
对应本例的k4= -12345

-12345

5、输入: -54321✓ (超下限)
对应本例的k5= -32768

-54321

u_short=unsigned short



§. 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> | <div>贴图即可，不需要写分析结果</div> <div>u_int=unsigned int</div> <div>1、输入：__40000__ ✓ (合理范围)</div> <div>2、输入：__2200000000__ ✓ (超过int上限但u_int上限)</div> <div>3、输入：__4500000000__ (u_int上限)</div> <div>4、输入：__-40000__ ✓ (合理范围)</div> <div>5、输入：__-2200000000__ ✓ (超过int下限但u_int下限)</div> |
| <p>结论：</p> <p>多个输入中，编号__2, 3, 5__输入的k值是不可信的</p> | <div>本题要求VS+Dev</div> |



§. 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

注: 具体对比程序及输出结果等不要再贴图, 自行完成即可

需要回答下列问题 (回答问题不是完成作业, 而是自己真的弄懂了概念后的总结) :

1、输入/赋值超int上限但未超同类型的u_int上限, 两者是否一致? 如果有区别, 区别是?

有区别, 输入时超过范围, 会在系统中标记一个错误值, 并将上限给变量。如果是赋值时超过范围时, 会发生截断。两者的值并不相同。

2、输入/赋值超int上限且超同类型的u_int上限, 两者是否一致? 如果有区别, 区别是?

有区别, 输入时超过范围, 会在系统中标记一个错误值, 并将上限给变量。如果是赋值时超过范围时, 也会发生截断。比如4500000000会截断为205032704, 所以两者的值并不相同。

3、输入/赋值超int下限, 两者是否一致? 如果有区别, 区别是?

有区别, 输入时超过范围, 会在系统中标记一个错误值, 并将下限给变量。如果是赋值时超过范围时, 也会发生截断。比如-2200000000会截断为2094967296, 所以两者的值并不相同。



§. 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

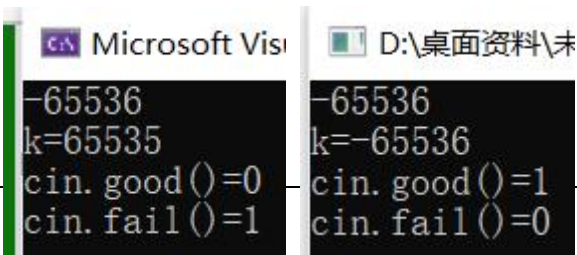
D. 运行下面的程序，观察不同输入下的运行结果（贴图在清晰可辨的情况下尽可能小）

```
#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;
}
```

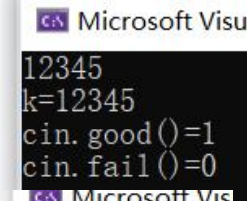
结论:

多个输入中，编号__2, 6__输入的k值是可信的

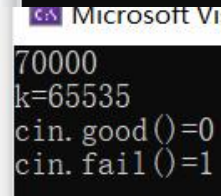


贴图即可，不需要写分析结果

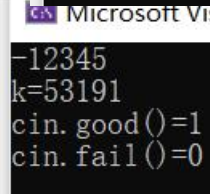
1、输入：12345✓（合



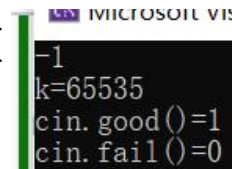
2、输入：70000✓（超



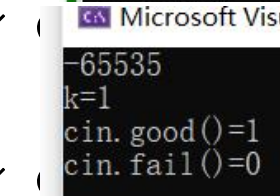
3、输入：-12345✓（负数但未超过sh



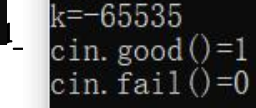
4、输入：-1✓（



5、输入：-65535✓（



6、输入：-65536✓（



u_short=unsigned short

加负号后的下限)

加负号后的下限)

本题要求VS+Dev

D-Compare. 仿B-Compare构造的**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

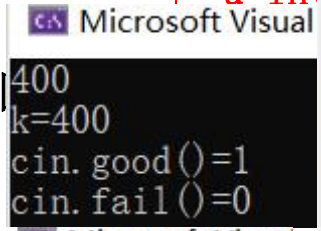
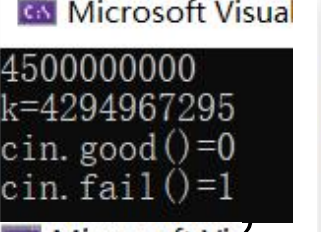

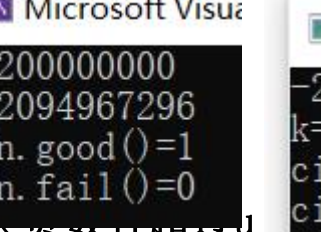
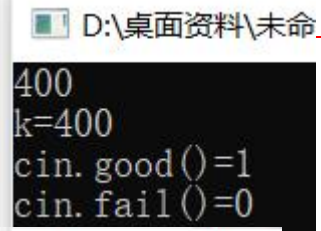
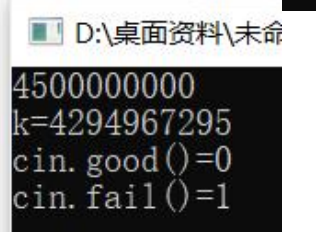
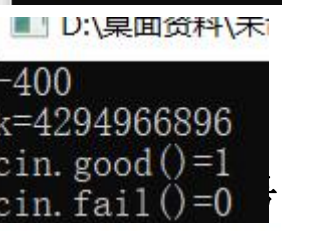
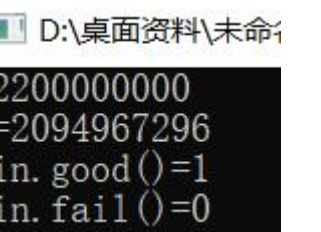

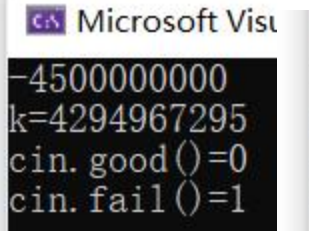
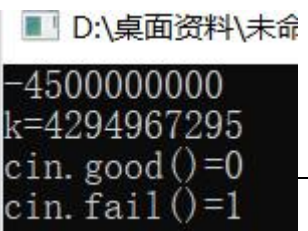
要求VS+Dev



§. 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. 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）

| | | |
|--|---|---|
| <pre>#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; }</pre> | <p>贴图即可，不需要写分析结果</p> <p>1、输入：__400__✓ （合理范围）</p> <p>2、输入：__4500000000__✓</p> <p>3、输入：__ -400__✓ （负数）</p> <p>4、输入：__ -2200000000__✓ （超过后的下限）</p> <p>5、输入：__ -4500000000__✓ （负数且超过下限）</p> | <p>u int-unsigned int</p>          |
| <p>结论：</p> <p>多个输入中，编号_2,6_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

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

注: 具体对比程序及输出结果等不要再贴图, 自行完成即可

需要回答下列问题 (回答问题不是完成作业, 而是自己真的弄懂了概念后的总结) :

1、输入/赋值超u_int上限, 两者是否一致? 如果有区别, 区别是?

有区别, 输入时超过范围, 会在系统中标记一个错误值, 并将上限给变量。如果是赋值时超过范围时, 也会发生截断。比如4500000000会截断为205032704, 所以两者的值并不相同。

2、输入/赋值为负数但未超int下限, 两者是否一致? 如果有区别, 区别是?

并没有区别, 两者都是将一个unsigned int型的数据转化为int型, 补码相同, 转化后的值也是相同

3、输入/赋值为负数且未超过u_int上限加负号后的下限, 两者是否一致? 如果有区别, 区别是?

并没有区别, 两者都是将一个unsigned int型的数据转化为int型, 补码相同, 转化后的值也是相同。比如-2200000000会截断为2094967296, 所以两者的值并不相同。

4、输入/赋值为负数负数且超过u_int上限加负号后的下限? 如果有区别, 区别是?

有区别, 输入时超过范围, 会在系统中标记一个错误值, 并将上限给变量。如果是赋值时超过范围时, 也会发生截断。比如-4500000000会截断为4089934592, 所以两者的值并不相同。



§. 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

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 **两者相同**

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

有区别, 输入时超过范围, 会在系统中标记一个错误值, 并将上限给变量。如果是赋值时超过范围时, 会发生截断。两者的值并不相同。

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

有区别, 输入时超过范围, 会在系统中标记一个错误值, 并将上限给变量。如果是赋值时超过范围时, 也会发生截断。比如4500000000会截断为205032704, 所以两者的值并不相同。

4、signed数据在输入正确但超下限范围的情况下

有区别, 输入时超过范围, 会在系统中标记一个错误值, 并将下限给变量。如果是赋值时超过范围时, 也会发生截断。比如-2200000000会截断为2094967296, 所以两者的值并不相同。

5、unsigned数据在输入正确且范围合理的情况下

两者相同

6、unsigned数据在输入正确且超上限的情况下

有区别, 输入时超过范围, 会在系统中标记一个错误值, 并将上限给变量。如果是赋值时超过范围时, 也会发生截断。比如4500000000会截断为205032704, 所以两者的值并不相同。

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

并没有区别, 两者都是将一个unsigned int型的数据转化为int型, 补码相同, 转化后的值也是相同

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

并没有区别, 两者都是将一个unsigned int型的数据转化为int型, 补码相同, 转化后的值也是相同。比如-2200000000会截断为2094967296, 所以两者的值并不相同。

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

有区别, 输入时超过范围, 会在系统中标记一个错误值, 并将上限给变量。如果是赋值时超过范围时, 也会发生截断。比如-4500000000会截断为4089934592, 所以两者的值并不相同。

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

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



§. 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. 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()
{
    char ch;
    cin >> ch;

    cout << "ch=" << int(c) << endl;
    cout << "ch=" << c << endl;

    return 0;
}
```

1、键盘输入A（单个图形字符）

C:\ Micros

A
ch=65
ch=A

2、键盘输入\b（退格键的转义符）

C:\ 选择 Micro

\b
ch=92
ch=\\

3、键盘输入\101（A的ASCII码的8进制表示）

C:\ Micros

\101
ch=92
ch=\\

4、键盘输入\x41（A的ASCII码的16进制表示）

C:\ Micros

\x41
ch=92
ch=\\

5、键盘输入65（A的ASCII码的十进制表示）

C:\ Micros

65
ch=54
ch=6

6、键盘输入Ctrl+C（注意：是Ctrl+C组合键，注意不要有输入法栏）

C:\ Microsoft Vi

ch=

7、键盘输入Ctrl+z（注意：是Ctrl+z组合键，注意不要有输入法栏）

C:\ Microsoft

Z
ch=-52
ch=



§. 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. 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>
#include <iomanip>
using namespace std;
int main()
{
    float f;
    cin >> f;

    cout << f << endl;
    cout << setprecision(20) << f << endl;

    return 0;
}
```

//注：20已超float和double的有效位数

- | 输入 | cout << f << endl | cout << setprecision(20) << f << endl |
|-------------------------|---------------------------|---------------------------------------|
| 1、键盘输入123.456 (符合精度) | 123.456 | 1.23456e2 |
| 2、键盘输入1.23456e2 (符合精度) | 123.456 | 123.456 |
| 3、键盘输入-123.456 (符合精度) | -123.456 | -1.23456e2 |
| 4、键盘输入-1.23456e2 (符合精度) | -123.456 | -123.456 |
| 5、键盘输入123.456789 (符合精度) | 123.456789 | 123.456789109375 |
| 6、键盘输入6.7e38 (超上限) | 0 | 0 |
| 7、键盘输入1.7e39 (超上限) | 0 | 0 |
| 8、键盘输入-2.3e39 (超上限) | 0 | 0 |
| 9、键盘输入1.23e-30 (符合精度) | 1.23e-30 | 1.23e-30 |
| 10、键盘输入-1.23e-30 (符合精度) | 1.2299999549998595325e-30 | -1.2299999549998595325e-30 |

§. 基础知识题 - cin与cout的基本使用





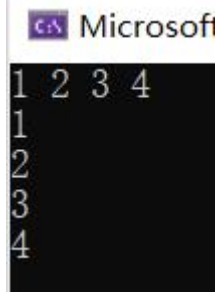
此页不要删除，也没有意义，仅仅为了分隔题目



§. 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>  |
| <p>1、程序运行后，输入：1 2 3 4✓，观察输出结果</p> <p>2、解释第2个和第3个程序的cin语句的使用区别：</p> <p>第2个是一个语句，分多行写。第3个是多个语句，多行写。</p> | | |



§. 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）

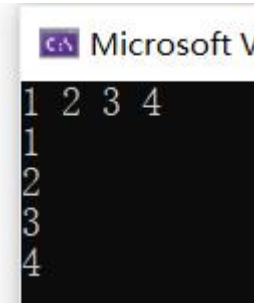
```
#include <iostream>
using namespace std;

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

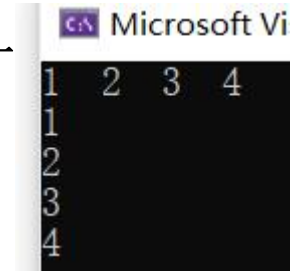
    cout << a << endl;
    cout << b << endl;
    cout << c << endl;
    cout << d << endl;

    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✓

✓

✓

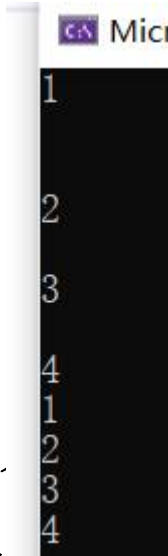
2✓

✓

3✓

✓

4✓（每个数字后立即加回车 + 多一个空格）



结论：在输入正确的情况下，回车和空格的作用？

输入正确的情况下，回车和空格是输入终止的条件，并不读取。



§. 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）

```
#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 4m✓

1 2 3 4m

2、输入: 1 2 3m 4✓

1 2 3m 4

3、输入: 1 2m 3 4✓

1 2m 3 4

4、输入: 1m 2 3 4✓

1m 2 3 4

5、输入: 1 2 3 m✓

1 2 3 m

6、输入: 1 2 m 4✓

1 2 m 4

7、输入: 1 m 3 4✓

1 m 3 4

8、输入: m 2 3 4✓

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

要求: 综合观察运行结果, 加上自己的思考, 给出总结性的结论, 这个结论要能对多个输入情况下不同位置的错误情况有普遍适应性, 而不仅仅是简单的根据结论说错在1/2/3/4位置

(提示: 从什么位置开始值不可信?)

当开始出现非法字符时就开始不可信, 出现m时会赋值0, 同时后面的变量均不可信



§. 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. 观察不同输入下的运行结果（贴图在清晰可辨的情况下尽量）

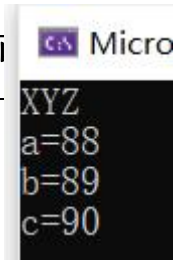
```
#include <iostream>
using namespace std;

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

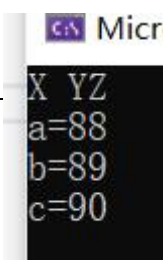
    cout << "a=" << int(a) << endl;
    cout << "b=" << int(b) << endl;
    cout << "c=" << int(c) << endl;

    return 0;
}
```

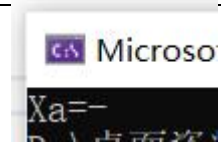
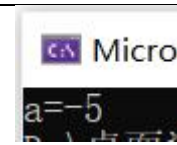
1、输入：XYZ✓



2、输入：X YZ✓



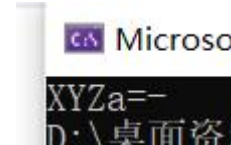
3、输入：Ctrl+C✓ （表示按Ctrl+C组合键，注意不要有输入法栏，下同）



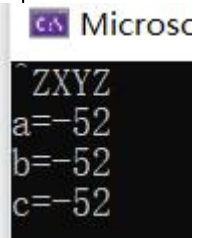
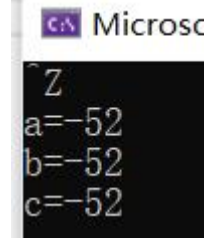
4、输入：XCtrl+C✓



5、输入：XYCtrl+C✓



6、输入：XYZCtrl+C✓



7、输入：Ctrl+z✓ （若未出结果则继续输入，可以按回车后多行输入，打印后观察结果）

8、输入：Ctrl+zXYZ✓ （若未出结果则继续输入，可以按回车后多行输入，打印后观察结果）

总结：多个cin输入时char型数据时

1、能否输入空格

可以输入空格

2、Ctrl+C在输入中表示什么？（可自行查阅资料，若资料与表现不符，信哪个？）

强制中断程序的执行，杀死程序的进程

3、Ctrl+z在输入中表示什么？（可自行查阅资料，若资料与表现不符，信哪个？）


将任务中断，挂起的状态，进程还存在，任务还没有结束

4、Ctrl+z后不按回车而继续输入的其他字符，能否被读入？

不能够被读入

§. 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. 自行构造测试数据，观察不同:  Microsoft Visual S (贴图在清晰可辨的情况下尽可能小)

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

```
int main()
{
```

```
    float a, b, c;
    cin >> a >> b >> c;
```

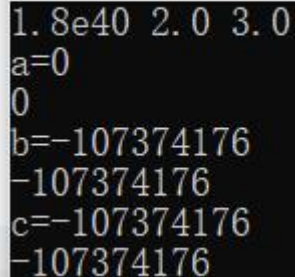
```
    cout << "a=" << a << endl;
    cout << setprecision(20)
```

```
    cout << "b=" << b << endl;
    cout << setprecision(20)
```

```
    cout << "c=" << c << endl;
    cout << setprecision(20)
```

```
    return 0;
```

```
}
```



```
1.8e40 2.0 3.0
a=0
0
b=-107374176
-107374176
c=-107374176
-107374176
```

1、输入: `_1.8e40, 2.0, 3.0_` ✓ (第1个超上限, 2/3正常)


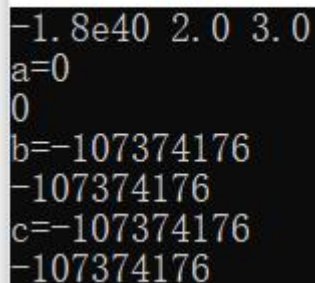
2、输入: `_-1.8e40, 2.0, 3.0_` ✓

3、输入: `_2.0, 1.8e40, 3.0_` ✓


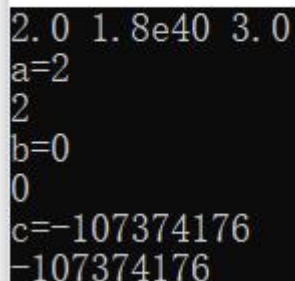
4、输入: `_2.0, -1.8e40, 3.0_` ✓

5、输入: `_2.0, 3.0, 1.8e40_` ✓ (1/2正常, 第3个超上限)


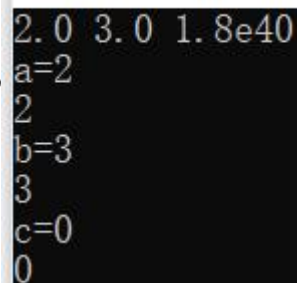
6、输入: `_2.0, 3.0, -1.8e40_` ✓ (1/2正常, 第3个超下限)

 Microsoft Visual Stu

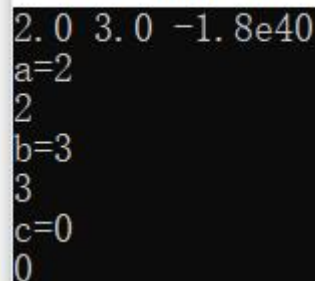
```
-1.8e40 2.0 3.0
a=0
0
b=-107374176
-107374176
c=-107374176
-107374176
```

 Microsoft Visual Stu

```
2.0 1.8e40 3.0
a=2
2
b=0
0
c=-107374176
-107374176
```

 Microsoft Visual Stu

```
2.0 3.0 1.8e40
a=2
2
b=3
3
c=0
0
```

 Microsoft Visual Stu

```
2.0 3.0 -1.8e40
a=2
2
b=3
3
c=0
0
```

总结:

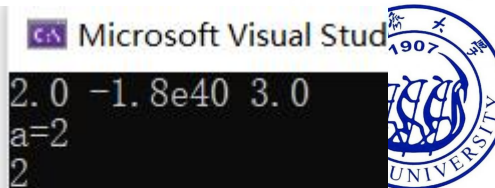
1、多个cin输入时, 错误输入出现在不同位置对输入正确性的影响

要求: 综合观察运行结果, 加上自己的思考, 给出总结性的结论, 这个结论要能对多个输入情况下不同位置的错误情况有普遍适应性, 而不仅仅是简单的根据结论说错在1/2/3位置

(提示: 从什么位置开始值不可信?)

超过范围的值是赋值为0, 后面的全为不可信值。

2、将float替换为double, 上述结论是否仍然成立?
依旧是成立的



§. 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

VS有error无法运行，Dev都没有，可以正常运行

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

输出为右图

3、分析为什么只有某个变量的结果是正确的

第一句初始化了三个变量。

输入语句中，由优先级的知识知，>>的优先级比，高所以先输入a的值，后是一个逗号表达式，b和c并没有输入值，所以没有初始化。输出也自然不正确。

D:\桌面资料\未命名2.exe
1 2 3
1
4242432
7929704

| | 代码 | 说明 |
|---|-------|-----------------|
| ✖ | 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" |
| ⚠ | C6001 | 使用未初始化的内存"b"。 |
| ⚠ | C6001 | 使用未初始化的内存"c"。 |

编译结果...

- 错误: 0
- 警告: 0
- 输出文件名: D:\桌面资料\未命名2.exe
- 输出大小: 2.33345127105713 MiB
- 编译时间: 1.69s

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;
}
```

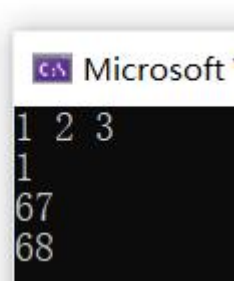
1、运行后，输入三个正确的int型数据(例 :1 2 3✓，注意不要是预置值)，观察输出

2、通过观察三个变量的输出，你得到了什么结论？

`cin>>a,b,c;`

这句话中，>>的优先级比，高

所以先输入了a的值，然后是一个逗号表达式，并未对b, c的值进行修改，所以输出的结果是 1 67 68





§. 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)
- 2、分析为什么编译有错
因为在两句的输入语句中，>>右侧的都是常量或者表达式，如第一句为5，第二句由于+的优先级高为表达式a+10，未找到正确的流对象
- 3、结论：流提取运算符后面必须跟__b__，不能是__a, c__
a) 常量 b) 变量 c) 表达式

| | | 代码 | 说明 |
|-----|----|---|---|
| | | C2678 | 二进制">>": 没有找到接受"std::istream"类型的左操作数的运算符(或没有可接受的转换) |
| | | C2678 | 二进制">>": 没有找到接受"std::istream"类型的左操作数的运算符(或没有可接受的转换) |
| | | E0349 | 没有与这些操作数匹配的 ">>" 运算符 |
| | | E0349 | 没有与这些操作数匹配的 ">>" 运算符 |
| 行 | 列 | 单元 | 信息 |
| | | D:\桌面资料\未命名2.cpp | In function 'int main()': |
| 6 | 9 | D:\桌面资料\未命名2.cpp | [Error] no match for 'operator>>' (operand types are 'std::istream' {aka 'std::basic_istream<char>'} and 'int') |
| 40 | | C:\Program Files (x86)\Dev-Cpp\MinGW64\lib\gcc\x86... | In file included from C:/Program Files (x86)/Dev-Cpp/MinGW64/lib/gcc/x86_64-w64-mingw32/9.2.0/include/c++/iostream |
| 1 | | D:\桌面资料\未命名2.cpp | from D:\桌面资料\未命名2.cpp |
| 120 | 7 | C:\Program Files (x86)\Dev-Cpp\MinGW64\lib\gcc\x86... | [Note] candidate: 'std::basic_istream<_CharT, _Traits>::_istream_type& std::basic_istream<_CharT, _Traits>::operator>>(std::basic_is |
| 120 | 7 | C:\Program Files (x86)\Dev-Cpp\MinGW64\lib\gcc\x86... | [Note] conversion of argument 1 would be ill-formed: |
| 6 | 12 | D:\桌面资料\未命名2.cpp | [Error] invalid conversion from 'int' to 'std::basic_istream<char>::_istream_type& (*)(&std::basic_istream<char>::_istream_type&)' {a |

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

2、通过观察三个变量的输出，你得到了什么结论？

逗号表达式的值是最后一个语句。

该题只给c输入了值，只有c变化了。

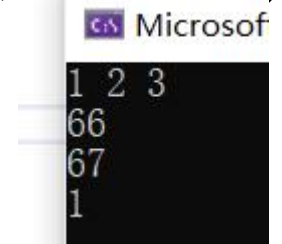
3、和B进行比较，分析为什么结果有差异

我们看到在输入语句中，先计算括号里的值，为c，再输入值给c，所以输出中我们看到只有c的值变了。

而在上一题中，>>的优先级高于，所以先输入值给a，再计算逗号表达式

4、和C进行比较，与C得出的结论矛盾吗？

并不矛盾，该题中流提取运算符后面跟的仍然是变量c
即为逗号表达式的值，与B中的并不矛盾。





§. 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：

Microsoft Visual Studio
1234 56.78
1 2 34 56.78

2、输入：1┐2┐34┐56.78✓

输出：

Microsoft Visual Studio
1 2 34 56.78
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）

第一个中cin会严格根据变量的类型读取，所以在字符c1，c2中会分别读取1和2。而空格是输入中止的条件，所以两题会出现相同的结果。



§. 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)
- 2、结论：在cin中不能跟_____endl_____

整个解决方案

错误 2警告 0展示 20 个消息中的 0 个生成 + IntelliSense

| | 代码 | 说明 |
|---|-----------|---|
| ✖ | 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) |
| ▶ | abc E0349 | 没有与这些操作数匹配的 ">>" 运算符 |

| 行 | 列 | 单元 | 信息 |
|-----|----|--|---|
| | | D:\桌面资料\未命名2.cpp | In function 'int main()': |
| 6 | 14 | D:\桌面资料\未命名2.cpp | [Error] no match for 'operator>>' (operand types are 'std::basic_istream<char>::_istream_type' (aka 'std::basic_istream<char>') and '<unresolved overloa |
| 40 | | C:\Program Files (x86)\Dev-Cpp\MinGW64\lib\gcc\x86... | In file included from C:/Program Files (x86)/Dev-Cpp/MinGW64/lib/gcc/x86_64-w64-mingw32/9.2.0/include/c++/iostream |
| 1 | | D:\桌面资料\未命名2.cpp | from D:\桌面资料\未命名2.cpp |
| 120 | 7 | C:\Program Files (x86)\Dev-Cpp\MinGW64\lib\gcc\x86_... | [Note] candidate: 'std::basic_istream<_CharT, _Traits>::_istream_type& std::basic_istream<_CharT, _Traits>::operator>>(std::basic_istream<_CharT, _Traits |
| 120 | 36 | C:\Program Files (x86)\Dev-Cpp\MinGW64\lib\gcc\x86_... | [Note] no known conversion for argument 1 from '<unresolved overloaded function type>' to 'std::basic_istream<char>::_istream_type& (*)(&std::basic_is |
| 124 | 7 | C:\Program Files (x86)\Dev-Cpp\MinGW64\lib\gcc\x86 ... | [Note] candidate: 'std::basic_istream<_CharT, _Traits>::_istream_type& std::basic_istream<_CharT, _Traits>::operator>>(std::basic_istream<_CharT, _Traits |

§. 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