



§. 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、网上提交本次作业（在“文档作业”中提交），**各班截止日期不同：01班-3月5日，03班-3月6日，其他班-3月7日 !!!**

特别说明:

- 1、本次作业是预习作业，在下周上课前必须完成，因此各班截止时间不同
- 2、对于作业过程中不清楚的问题或不会的内容，各班课程结束前（01班-3月6日，03班-3月7日，其他班-3月8日）先不要问（不清楚的位置可以先做个标记，课程结束后再去理解即可）
- 3、大家根据自己的意愿合理安排时间



§. 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

```
Microsoft Visual Studio 调试控制台
Hello, world!
D:\Workspace\VS2019-Demo\Debug\cpp-demo.exe (进程 7484)已退出, 代码为 0。
按任意键关闭此窗口...
```

例：有效贴图

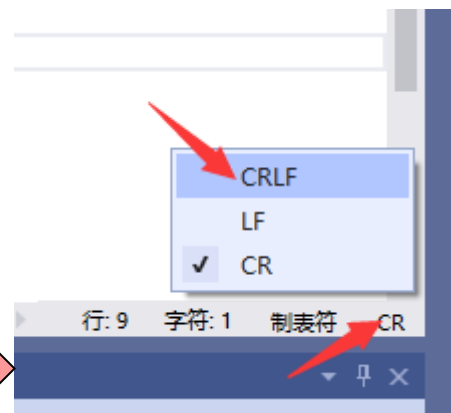
```
Microsoft Visual Studio 调试控制台
Hello, world!
```



§. 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、要求一个程序多次运行的，不要自以为是的修改程序，放在一次去运行
- 4、不要偷懒、不要自以为是的脑补结论!!!
- 5、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
- 6、这些结论，是让你记住的，不是让你完成作业后就忘掉了
- 7、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;

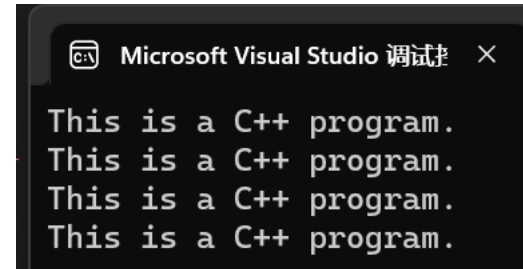
int main()
{
    /* 第1组 */
    cout << "This is a C++ program." << endl;

    /* 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;
}
```



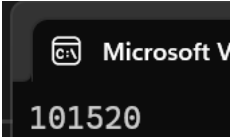
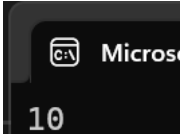
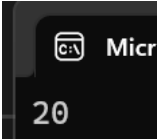
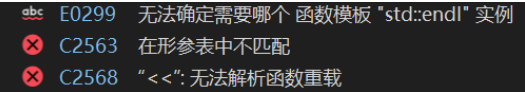
第3组和第4组在语句上的区别是：第三组语句仅一个分号，是一句cout语句，而分成很多行输出；而第四组语句是多个cout语句，有多个分号。



§. 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> 	<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：第一个程序是输出a, b, c的值；第二个程序是输出a的值；第三个程序是输出括号中最后一个变量c的值。			解释错误原因：两个插入运算符中有多个表达式。
结论：一个流插入运算符 << 只能输出__1__个数据.			

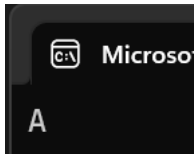


§. 基础知识题 - cin与cout的基本使用

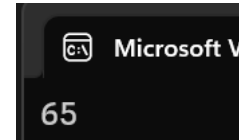
1、cout的基本理解

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：左侧的ch是char型，为字符变量；右侧的ch是int型，为整型变量，系统会自动判断输出数据的格式。



§. 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;
}
```

```
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
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)

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

```
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
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 << endl;
    return 0;
}
```

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      char ch = 65;
6      cout << ch*1 << endl;
7      return 0;
8  }
9
```

Micros
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

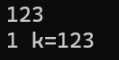
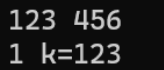
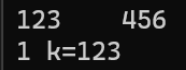
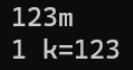
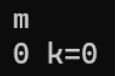
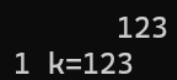
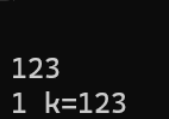
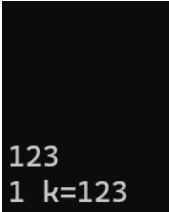
此页不要删除，也没有意义，仅仅为了分隔题目



§. 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 << cin.good(); 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？ 作为非法输入，终止输入，依旧可以正确输入。 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 3、直接输入（对int型而言是）非法的数据m，输出是？ 0 k=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）

<pre>#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; }</pre>	<p>贴图即可，不需要写分析结果</p> <div><p>1、输入：123✓（正确+回车）</p></div> <div><p>2、输入：123 456✓（正确+空格）</p></div> <div><p>3、输入：-123m✓（正确+非法字符）</p></div> <div><p>4、输入：m✓（直接非法字符）</p></div> <div><p>5、输入：54321✓（超上限）</p></div> <div><p>6、输入：-40000✓（超下限）</p></div>
<p>结论：</p> <p>多个输入中，编号_4、5、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

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
- 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：合理范围内的输入直接输出；而超上限或超下限的值会进行运算转换，将其变为short范围内的值再输出。



§. 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>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>1、输入：12345✓ （合理范围）</p> <p>2、输入：4000000000✓ （超上限但未超同类型的u_int上限）</p> <p>3、输入：22538932253893✓ （超上限且超过同类型的u_int上限）</p> <p>4、输入：-12345✓ （合理范围）</p> <p>5、输入：-2147483649✓ （超下限）</p>	<div>12345 k=12345 cin.good()=1 cin.fail()=0</div> <div>4000000000 k=2147483647 cin.good()=0 cin.fail()=1</div> <div>22538932253893 k=2147483647 cin.good()=0 cin.fail()=1</div> <div>-12345 k=-12345 cin.good()=1 cin.fail()=0</div> <div>-2147483649 k=-2147483648 cin.good()=0 cin.fail()=1</div>
		<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

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?

不一致, 区别在于输入超int上限但未超同类型的u_int上限会输出int上限, 而赋值会进行运算转换, 去除最前面的多余项再输出。

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?

不一致, 区别在于输入超int上限且超同类型的u_int上限会输出int上限, 而赋值会进行运算转换, 去除最前面的多余项再输出。

3、输入/赋值超int下限, 两者是否一致? 如果有区别, 区别是?

不一致, 区别在于输入超int下限会输出int下限, 而赋值会进行运算转换, 去除最前面的多余项再输出。



§. 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）

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

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

结论:

多个输入中，编号_2、6_输入的k值是不可信的

贴图即可，不需要写分析结果

u_short=unsigned short

1、输入：12345✓（合理范围）

12345
k=12345 good=1 fail=0

2、输入：70000✓（超上限）

70000
k=65535 good=0 fail=1

3、输入：-12345✓（负数但未超过short下限）

-12345
k=53191 good=1 fail=0

4、输入：-1✓（负数且未超过short下限）

-1
k=65535 good=1 fail=0

5、输入：-65535✓（负数且未超过u_short上限加负号后的下限）

-65535
k=1 good=1 fail=0

6、输入：-65536✓（负数且超过u_short上限加负号后的下限）

-65536
k=65535 good=0 fail=1

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

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

```
#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;
}
```

u_short=unsigned short

贴图即可（有warning还有贴warning），不需要写分析结果

12345
4464
53191
65535
1
0

myinfo.cpp(8,15): warning C4305: “=”: 从“int”到“unsigned short”截断
myinfo.cpp(8,10): warning C4309: “=”: 截断常量值
myinfo.cpp(12,11): warning C4309: “=”: 截断常量值

8 10 D:\同济\学习\大一（下）\高级程序设计\代码\h... [Warning] unsigned conversion from 'int' to 'short unsigned int' changes value from '70000' to '4464' [-Woverflow]
11 10 D:\同济\学习\大一（下）\高级程序设计\代码\h... [Warning] unsigned conversion from 'int' to 'short unsigned int' changes value from '-65535' to '1' [-Woverflow]
12 10 D:\同济\学习\大一（下）\高级程序设计\代码\h... [Warning] unsigned conversion from 'int' to 'short unsigned int' changes value from '-65536' to '0' [-Woverflow]

1、输入：12345✓ （合理范围）
对应本例的k1=12345

2、输入：70000✓ （超上限）
对应本例的k2=4464

3、输入：-12345✓ （负数但未超过short下限）
对应本例的k3=53191

4、输入：-1✓ （负数且未超过short下限）
对应本例的k4=65535

5、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=1

6、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

+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; cout << " good()=" << cin.good(); cout << " 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：123✓ （合理范围） 123 k=123 good()=1 fail()=0</p> <p>2、输入：5000000000✓ （超上限） 5000000000 k=4294967295 good()=0 fail()=1</p> <p>3、输入：-1✓ （负数但未超int下限） -1 k=4294967295 good()=1 fail()=0</p> <p>4、输入：-4294967294✓ （负数且未超过u_int上限加负号后的下限） -4294967294 k=2 good()=1 fail()=0</p> <p>5、输入：-5000000000✓ （负数且超过u_int上限加负号后的下限） -5000000000 k=4294967295 good()=0 fail()=1</p>	<p>u_int=unsigned int</p>
<p>结论：</p> <p>多个输入中，编号_2、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

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?
不一致, 输入超u_int上限则输出u_int上限; 而赋值会进行运算转换, 去除最前面的多余项再输出。
- 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?
一致, 二者都会运算转换, 作为u_int输出。
- 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?
一致, 二者都会运算转换, 作为u_int输出。
- 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?
不一致, 输入时输出u_int上限, 而赋值会进行运算转换, 去除最前面的多余项再输出。



§. 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

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
输出内容与输入内容相同。
- 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
输出signed的上限。
- 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
输出signed的上限。
- 4、signed数据在输入正确但超下限范围的情况下
输出signed的下限。
- 5、unsigned数据在输入正确且范围合理的情况下
输出内容与输入内容相同。
- 6、unsigned数据在输入正确且超上限的情况下
输出unsigned的上限。
- 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
输出为输入加上unsigned上限，再加一。
- 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
输出为输入加上unsigned上限，再加一。
- 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
输出为unsigned的上限。

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

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

超范围表现不同，合理范围表现相同。



§. 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. 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(ch) << endl;
    cout << "ch=" << ch << endl;

    return 0;
}
```

1、键盘输入A（单个图形字符）

```
A
ch=65
ch=A
```

2、键盘输入\b（退格键的转义符）

```
\b
ch=92
ch=\\
```

3、键盘输入\101（A的ASCII码的8进制转义表示）

```
\101
ch=92
ch=\\
```

4、键盘输入\x41（A的ASCII码的16进制转义表示）

```
\x41
ch=92
ch=\\
```

5、键盘输入65（A的ASCII码的十进制整数形式表示）

```
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）

```
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）

```
^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：setprecision(20)表示输出时保留
// 小数点后20位
// （已超float和double的有效位数）

1、键盘输入123.456（合理范围正数，小数形式）

```
123.456
123.456
123.45600128173828125
```

2、键盘输入1.23456e2（合理范围正数，指数形式）

```
1.23456e2
123.456
123.45600128173828125
```

3、键盘输入-123.456（合理范围负数，小数形式）

```
-123.456
-123.456
-123.45600128173828125
```

4、键盘输入-1.23456e2（合理范围负数，指数形式）

```
-1.23456e2
-123.456
-123.45600128173828125
```

5、键盘输入123.456789（合理范围，但超有效位数）

```
123.456789
123.457
123.456787109375
```

6、键盘输入6.7e38（尾数超上限但数量级未超，仍是 10^{38} ）

```
6.7e38
0
0
```

7、键盘输入1.7e39（超上限且数量级已超 10^{38} ）

```
1.7e39
0
0
```

8、键盘输入-2.3e39（超上限且数量级已超 10^{38} ）

```
-2.3e39
0
0
```

9、键盘输入1.23e-30（合理范围整数但指数很小）

```
1.23e-30
1.23e-30
1.2299999549998595325e-30
```

10、键盘输入-1.23e-30（合理范围负数但指数很小）

```
-1.23e-30
-1.23e-30
-1.2299999549998595325e-30
```



§. 基础知识题 - cin与cout的基本使用

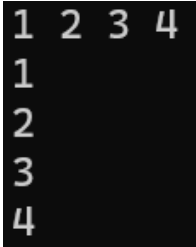
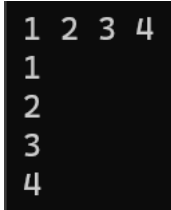
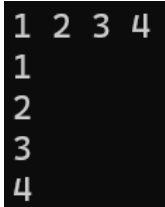
此页不要删除，也没有意义，仅仅为了分隔题目



§. 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：第2个程序是一个cin语句分四行写，而第3个程序是四个cin语句。</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）

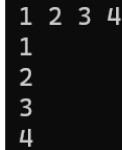
```
#include <iostream>
using namespace std;

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

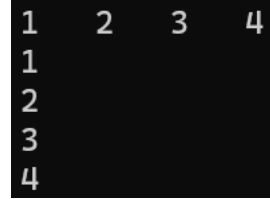
    cout << a << endl;
    cout << b << endl;
    cout << c << endl;
    cout << d << endl;

    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？
结束当前的输入进入下一个数据的输入。



§. 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

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

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

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?)

结论: 当某一位置输入错误, 则输出结果中此位置之前的位置输出与输入内容相同, 此位置为0, 其后位置均错误; 当某一位置输入正确值+非法字符, 此前位置正确输出, 此位置输出为输入的正确值, 其后位置输入错误。



§. 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可能小）

```
#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✓

XYZ
a=88
b=89
c=90

X YZ

a=88

b=89

c=90

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）

a=-52

4、输入：X Ctrl+C✓

Xa=-52

5、输入：XY Ctrl+C✓

XYa=-52

^Z

a=-52

b=-52

c=-52

6、输入：XYZ Ctrl+C✓

XYZa=

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+z XYZ✓ （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）

^Z XYZ
a=-52
b=-52
c=-52

总结：多个cin输入时char型数据时

1、能否输入空格

能

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？）
强制结束。信表现

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？）
当Ctrl+z单独位于一行的最前端时，表示输入结束；其余时候输入不结束。

4、Ctrl+z后不按回车而继续输入的其它字符，能否被读入？
不能。



§. 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 a, b, c;
    cin >> a >> b >> c;
```

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

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

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

```
    return 0;
```

```
}
```

```
-3.4e39 123.456 789
a=0
0
b=-107374176
-107374176
c=-107374176
-107374176
```

```
123.456 -3.4e39 789
a=123.456
123.45600128173828125
b=0
0
c=-107374176
-107374176
```

1、输入：3.4e39 123.456 789✓ （第1个超上限，2/3正常）

2、输入：-3.4e39 123.456 789✓ （第1个超下限，2/3正常）

3、输入：123.456 3.4e39 789✓ （1/3正常，第2个超上限）

4、输入：123.456 -3.4e39 789✓ （1/3正常，第2个超下限）

5、输入：123.456 789 3.4e39✓ （1/2正常，第3个超上限）

6、输入：123.456 789 -3.4e39✓ （1/2正常，第3个超下限）

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？）

多个cin输入时，某位置错误输入，其前位置的数字正确输出，此位置及其后位置均错误输出，且此位置输出为0，其后位置输出为-107374176。

2、将float替换为double，上述结论是否仍然成立？

仍然成立。

```
3.4e39 123.456 789
a=0
0
b=-107374176
-107374176
c=-107374176
-107374176
```

```
123.456 3.4e39 789
a=123.456
123.45600128173828125
b=0
0
c=-107374176
-107374176
```

```
123.456 789 3.4e39
a=123.456
123.45600128173828125
b=789
789
c=0
0
```

```
123.456 789 -3.4e39
a=123.456
123.45600128173828125
b=789
789
c=0
0
```



§. 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

因为一个运算符只能输入一个变量，只有a前有运算符，因此只有a成功输入又输出。

⚠ 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”

```
1 2 3
1
0
16
```

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

```
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？

只有a为输入的值，而b，c均为之前的赋值。得出结论：当变量赋值后，若成功新输入一个值，则会覆盖原先的值；若为成功输入，则不会覆盖，输出原先的值。



§ . 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）

<pre>#include <iostream> using namespace std; int main() { int a; cin >> 5; cin >> a+10; cout << a << endl; return 0; }</pre>	<div>1、如果编译有error或warning，则贴相应信息的截图（信息太多则前五行）</div> <div>2、分析为什么编译有错</div> <div>因为5是常量，a+10是表达式，无法作为变量输入。</div> <div>3、结论：流提取运算符后面必须跟_b_，不能是__a, c__</div> <div>a) 常量 b) 变量 c) 表达式</div>
--	--

```
[Error] no match for 'operator>>' (operand types are 'std::istream' {aka 'std::basic_istream<char>'} and 'int')
In file included from C:/Program Files (x86)/Dev-Cpp/MinGW64/lib/gcc/x86_64-w64-mingw32/9.2.0/include/c++/iostream
from D:\同济\学习\大一（下）\高级程序语言设计\代码\homework\myinfo\myinfo.cpp
[Note] candidate: 'std::basic_istream<_CharT, _Traits>::_istream_type& std::basic_istream<_CharT, _Traits>::operator>>(std::basic_istream<_CharT, _Traits>::_istream_type& (*)(std::basic_istream<_CharT, _Traits>::_istream_type&)) [with _C...
[Note] conversion of argument 1 would be ill-formed:
[Error] invalid conversion from 'int' to 'std::basic_istream<char>::_istream_type& (*)(std::basic_istream<char>::_istream_type&)' {aka 'std::basic_istream<char>& (*)(std::basic_istream<char>&)' } [-fpermissive]
```

本题要求VS+Dev



§. 基础知识题 - cin与cout的基本使用

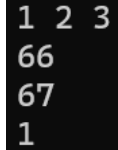
4、cin的基本理解 - 其他情况

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



```
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？

()看作整体来输入，序列运算符只输出括号内的最后一个值。

3、和B进行比较，分析为什么结果有差异

因为B中无()，而>>后跟的是a，而a后b前无>>，因而只输入a.

4、和C进行比较，与C得出的结论矛盾吗？

不矛盾，因为()看作整体，作为变量处理。



§. 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;
}
```

```
1234 56.78
1 2 34 56.78
```

```
1 2 34 56.78
1 2 34 56.78
```

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）
空格是为了结束当前输入进入下一个变量的输入。第一种因为c1, c2都是char型，两者均只取一个数字，由于56.78前有空格，a只能取34，从而两者输出结果相同。



§. 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）

<pre>#include <iostream> using namespace std; int main() { int a; cin >> a >> endl; return 0; }</pre>	<p>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)</p> <p>2、结论：在cin中不能跟____endl____</p>  <p>[Error] no match for 'operator>>' (operand types are 'std::basic_istream<char>::__istream_type' {aka 'std::basic_istream<char>'} and '<unresolved overloaded function type>')</p>
	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