



§. 基础知识题 – 输入输出部分

要求:

- 1、完成本文件中所有的题目并按要求写出分析、运行结果（包括截图）
- 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(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)
- 4、如果是WPS打开PPT，则在某些版本的WPS中直接复制代码到VS2019中后，**编译会报错**，解决方式是先复制到Dev中，再次复制到VS2019中即可
- 5、转换为pdf后提交
- 6、**10月8日前**网上提交本次作业（在“实验报告”中提交）
- 7、作业部分细节内容可能会有调整，随时注意Update!!!

预告：本次作业未完，尚余如下内容：

- 8、在cout中使用格式化控制符
- 9、在cin中使用格式化控制符
- 10、C语言的scanf和printf的使用

**实在来不及准备了，这三部分的作业，最迟在10.5发，截止时间仍为10.8日（同本文件截止时间）
为了留出充足时间，建议本文件上的作业在10.5前完成**

§. 基础知识题 - 输入输出部分



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 – 输入输出部分

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;

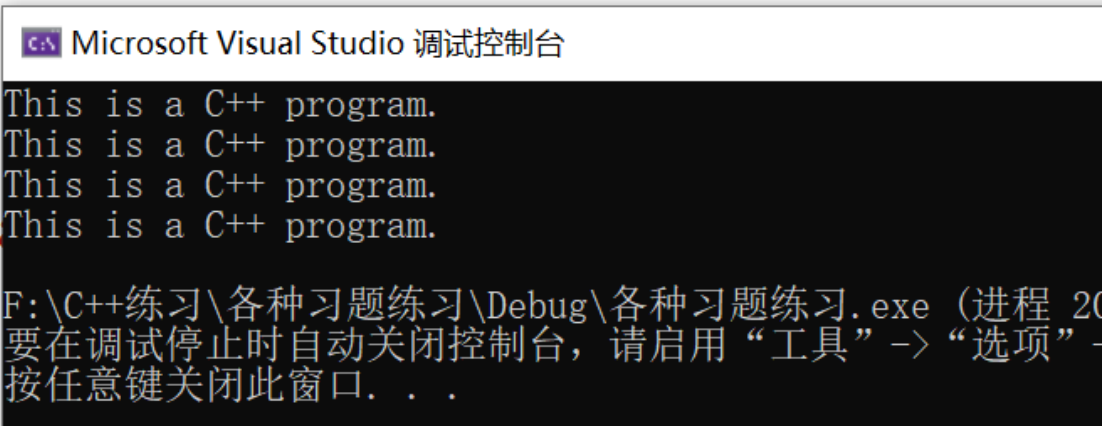
int main()
{
    /* 第1组 */
    cout << "This is a C++ program." << endl;

    /* 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;

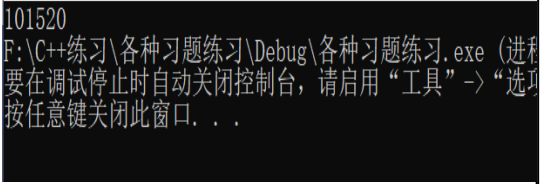
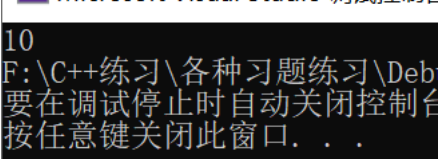
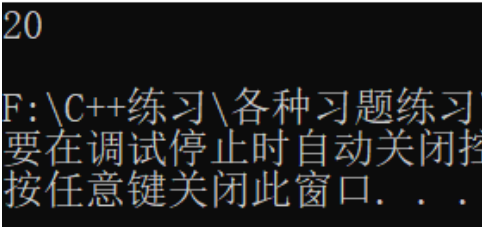
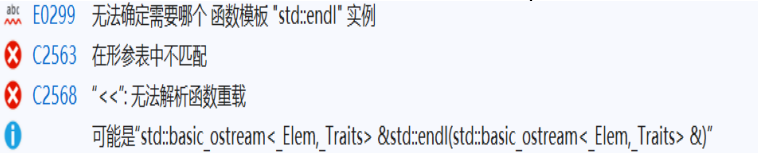
第3组和第4组在语句上的区别是：
第3组是一条语句，分了4行，但是整体是一个语句，第4组是4条语句，分了4行，整体是4条语句，但是输出显示它们是一样的。



§. 基础知识题 – 输入输出部分

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：第一个是输出a，b，c三个数，得到它们的值；第二组是因为<<优先级大于逗号运算符，所以只输出a的值，b和c并没有输出；第三组将a，b，c，括在一起，那么逗号表达式的值为右值，则为c的值。			解释错误原因： 逗号运算符优先级低于<<，结合时c<<endl;这个就是错误
结论：一个流插入运算符 << 只能输出__1__个数据.			



§ . 基础知识题 – 输入输出部分

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)

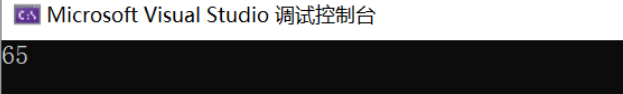
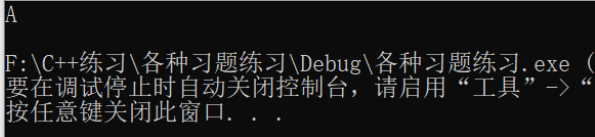
<pre>#include <iostream> using namespace std; int main() { char ch = 65; cout << ch << endl; return 0; }</pre> 	<pre>#include <iostream> using namespace std; int main() { int ch = 65; cout << ch << endl; return 0; }</pre> 
<p>解释这两个程序输出不同的原因：第一个程序将变量ch定义成char类型，那么输出时输出的是字符型，输出ASCII码65对应的字符，第二个程序将变量ch定义成int型，输出的是整型数字65.</p>	



§. 基础知识题 – 输入输出部分

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

<pre>#include <iostream> using namespace std; int main() { char ch = 65; cout << ch << endl; return 0; }</pre> <pre>#include <iostream> using namespace std; int main() { char ch = 65; cout << static_cast<int>(ch) << endl; return 0; }</pre> 	<pre>#include <iostream> using namespace std; int main() { int ch = 65; cout << ch << endl; return 0; }</pre> <pre>#include <iostream> using namespace std; int main() { int ch = 65; cout << static_cast<char>(ch) << endl; return 0; }</pre> 
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)	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 – 输入输出部分

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

<pre>#include <iostream> using namespace std; int main() { char ch = 65; cout << ch << endl; return 0; }</pre>	<pre>#include <iostream> using namespace std; int main() { char ch = 65; cout << ch+0 << endl; return 0; }</pre>
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 - 输入输出部分

此页不要删除，也没有意义，仅仅为了分隔题目



§. 基础知识题 - 输入输出部分

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）

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

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

    return 0;
}
```

1、输入: 123✓ (✓代表回车键)

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: 123✓ (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)

Microsoft Visual Studio 调试控制台

123
k=123

123 456
k=123

123 456
k=123

123m
k=123

m
k=0

123
k=123

123
k=123

123
k=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?
回车是代表输完了, 完成输入, 并运行程序进行结果显示
空格是分隔, cin中无法读取字符空格, 所以是分隔两个数据
m的作用就是与空格一样, 分隔数据, 但是都不读取它们。

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

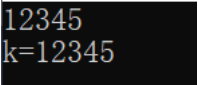
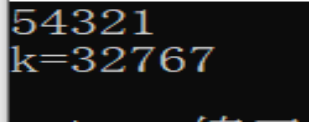
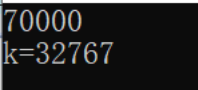
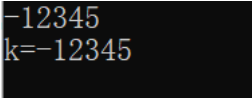
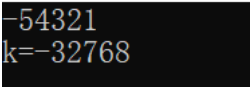
3、直接输入(对int型而言是)非法的数据m, 输出是? 输出是0



§. 基础知识题 - 输入输出部分

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; return 0; }</pre>	<div>1、输入：12345✓（合理范围） </div> <div>2、输入：54321✓（超上限但未超同类型的unsigned上限） </div> <div>3、输入：70000✓（超上限且超过同类型的unsigned上限） </div> <div>4、输入：-12345✓（合理范围） </div> <div>5、输入：-54321✓（超下限） </div>	<div>12345 k=12345 ----- Process exited after 请按任意键继续. . .</div> <div>54321 k=32767 -----</div> <div>70000 k=32767 -----</div> <div>-12345 k=-12345 -----</div> <div>-54321 k=-32768 -----</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>	<p>贴图即可，不需要写分析结果</p>	<p>本题要求VS+Dev</p>



§ . 基础知识题 - 输入输出部分

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

<pre>#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; }</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：54321✓（超上限但未超同类型的unsigned上限） 对应本例的k2=-112153、输入：70000✓（超上限且超过同类型的unsigned上限） 对应本例的k3=44644、输入：-12345✓（合理范围） 对应本例的k4=-123455、输入：-54321✓（超下限） 对应本例的k5=11215 <p>没有输入cin，直接给这些变量赋值的话，超范围输出时就会自动转化为short里对应的数， 但是有cin，cin里输入超范围的数就直接输出为short类型最大的数32767， 负数那边超范围的话就是-32768</p>
---	--



§. 基础知识题 - 输入输出部分

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; return 0; }</pre>	<div>1、输入：_2000_✓ （合理范围） </div> <div>2、输入：_2150000000_✓ （超上限但未超同类型的unsigned上限） </div> <div>3、输入：_5000000000_✓ （超上限且超过同类型的unsigned上限） </div> <div>4、输入：_-2000_✓ （合理范围） </div> <div>5、输入：_-5000000000_✓ （超下限） </div>
<p>基础知识：</p> <p>int的最小值是：_-2147483648_</p> <p>int的最大值是：_2147483647_</p>	<div>贴图即可，不需要写分析结果</div> <div>本题要求VS+Dev</div>



§. 基础知识题 - 输入输出部分

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), 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 = 2000;
    k2 = 21500000000;
    k3 = 50000000000;
    k4 = -2000;
    k5 = -50000000000;
    cout << k1 << endl;
    cout << k2 << endl;
    cout << k3 << endl;
    cout << k4 << endl;
    cout << k5 << endl;
    return 0;
}
```

Microsoft Visual Studio 调试控制台

```
2000
-2144967296
705032704
-2000
-705032704
```

F:\C++练习\各种习题练习\Debug\各种习题练习.exe (进程 18944) 已退出, 代码为 0。
要在调试停止时自动关闭控制台, 请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
按任意键关闭此窗口. . .

与short的那个类似, 没有输入cin, 直接给这些变量赋值的话, 超范围输出时就会自动转化为int里对应的数, 但是有cin, cin里输入超范围的数就直接输出为int类型最大的数2147483647, 负数那边超范围的话就是-2147483648



§. 基础知识题 - 输入输出部分

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; return 0; }</pre>	<div>1、输入：12345✓（合理范围）</div> <div></div> <div>2、输入：70000✓（超上限）</div> <div></div> <div>3、输入：-12345✓（负数但未超同类型的signed下限）</div> <div></div> <div>4、输入：-54321✓（负数且超过同类型的signed下限）</div> <div></div>
<p>基础知识：</p> <p>u_short的最小值是：__0__</p> <p>u_short的最大值是：_65535_</p>	<div>贴图即可，不需要写分析结果</div> <div>本题要求VS+Dev</div>



§. 基础知识题 - 输入输出部分

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), 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;
    k1 = 12345;
    k2 = 70000;
    k3 = -12345;
    k4 = -54321;
    cout << k1 << endl;
    cout << k2 << endl;
    cout << k3 << endl;
    cout << k4 << endl;
    return 0;
}
```

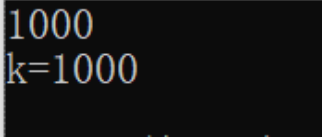
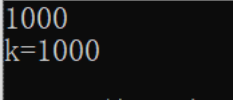
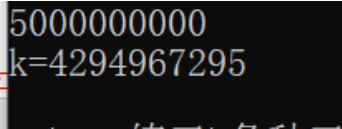
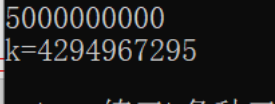
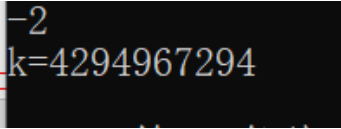
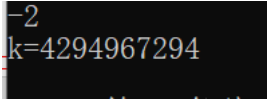
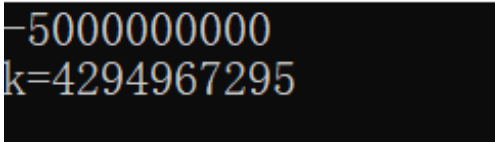
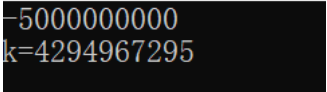
输入正数与short的那个类似, 没有输入cin, 直接给这些变量赋值的话, 超范围输出时就会自动转化为unsigned short里对应的数, 但是有cin, cin里输入超范围的数就直接输出为unsigned short类型最大的数65535。
但是负数就不一样了, 负数不管有无cin, 统一都是转化成unsigned short里面对应的数。



§. 基础知识题 - 输入输出部分

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; return 0; }</pre>	<div>1、输入：_1000_✓ （合理范围）</div> <div></div> <div>2、输入：_5000000000____✓ （超上限）</div> <div></div> <div>3、输入：_-2____✓ （负数但未超同类型的signed下限）</div> <div></div> <div>4、输入：_-5000000000____✓ （负数且超过同类型的signed下限）</div> <div></div>
<p>基础知识：</p> <p>u_int的最小值是：__0__</p> <p>u_int的最大值是：_4294967295_</p>	<div>贴图即可，不需要写分析结果</div> <div>本题要求VS+Dev</div>

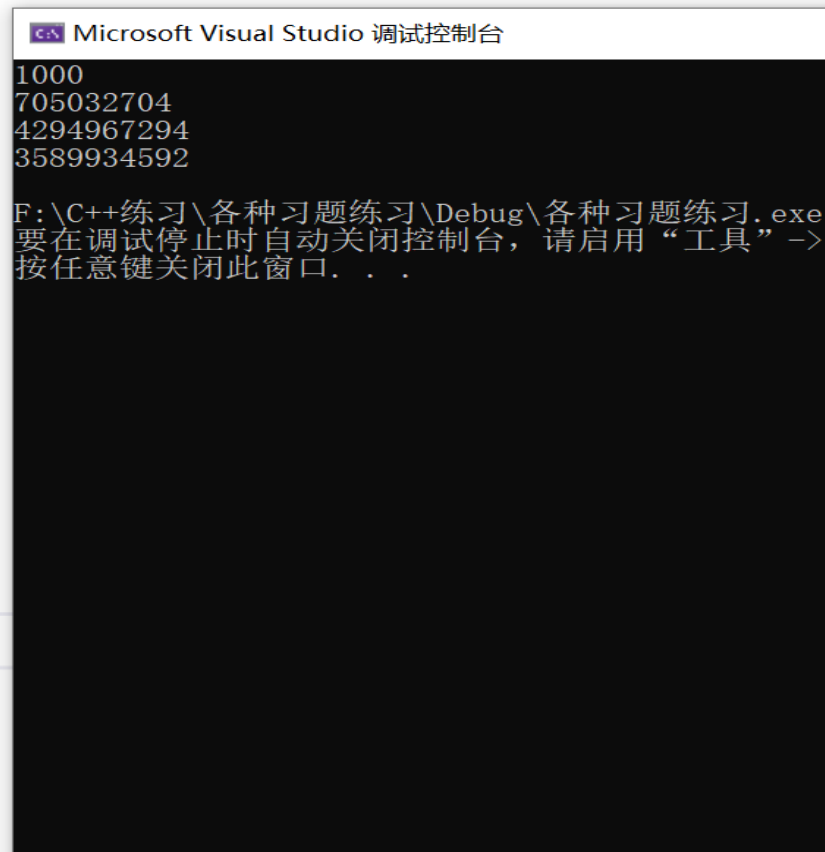


§. 基础知识题 - 输入输出部分

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), 观察运行结果并与E的输出结果进行对比分析

```
#include <iostream>
using namespace std;
int main()
{
    unsigned int k1, k2, k3, k4;
    k1 = 1000;
    k2 = 5000000000;
    k3 = -2;
    k4 = -5000000000;
    cout << k1 << endl;
    cout << k2 << endl;
    cout << k3 << endl;
    cout << k4 << endl;
    return 0;
}
```



输入正数与short的那个类似, 没有输入cin, 直接给这些变量赋值的话, 超范围输出时就会自动转化为unsigned里对应的数, 但是有cin, cin里输入超范围的数就直接输出为unsigned类型最大的数4294967295。

但是负数就不一样了, 负数不管有无cin, 统一都是转化成unsigned里面对应的数。



§ . 基础知识题 – 输入输出部分

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	输入什么就输出什么
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	输出时为该类型最大的数
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	输出时为该类型最大的数
4、signed数据在输入正确但超下限范围的情况下	输出时为该类型最小的负数
5、unsigned数据在输入正确且范围合理的情况下	输入什么就输出什么
6、unsigned数据在输入正确且超上限的情况下	输出时为该类型最大的数
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	输出时负数转化为对应的二进制然后输出时转化为unsigned数据
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	输出时负数转化为对应的二进制然后输出时转化为unsigned数据

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

不同，cin输入最后输出的是该类型右值

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

相同



§. 基础知识题 – 输入输出部分

2、cin的基本理解 – 单数据情况

F. 仿照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：

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

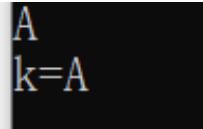
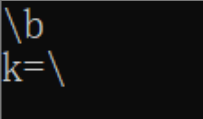
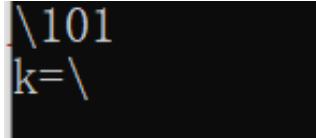
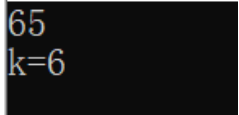
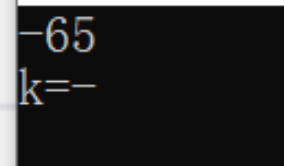
本页不要再贴测试程序了，下页开始



§. 基础知识题 - 输入输出部分

2、cin的基本理解 - 单数据情况

F1. 运行下面的程序，观察不同输入下的运行结果

<pre>#include <iostream> using namespace std; int main() { char k; cin >> k; cout << "k=" << k << endl; return 0; }</pre>	<p>1、输入：A✓ （单个字符）</p>  <p>2、输入：\b✓ （反义字符）</p>  <p>4、输入：\101✓ （8进制数）</p> 
<p>基础知识：</p> <p>char的最小值是：__-128__</p> <p>char最大值是：__127__</p>	<p>5、输入：65✓ （数字输入ASCII码值）</p>  <p>5、输入：-65✓ （数字输入ASCII码值）</p> 



§. 基础知识题 - 输入输出部分

2、cin的基本理解 - 单数据情况

F1-Compare. 运行下面的**对比**程序（cin输入与赋值），观察运行结果

```
#include <iostream>
using namespace std;
int main()
{
    char k1, k2, k3, k4, k5;
    k1 = 'A';
    k2 = '\b';
    k3 = '\101';
    k4 = 65;
    k5 = -65;
    cout << k1 << endl;
    cout << k2 << endl;
    cout << k3 << endl;
    cout << k4 << endl;
    cout << k5 << endl;
    return 0;
}
```

Microsoft Visual Studio 调试控制台

A
A
A
A

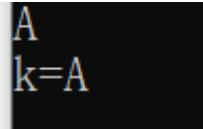
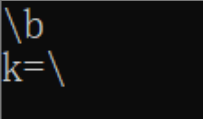
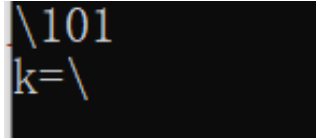
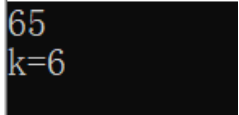
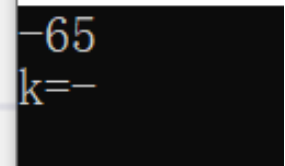
F:\C++练习\各种习题练习\Debug\各种习题练习.exe (进
要在调试停止时自动关闭控制台，请启用“工具”->“选
按任意键关闭此窗口. . .



§ . 基础知识题 – 输入输出部分

2、cin的基本理解 – 单数据情况

F2. 运行下面的程序，观察不同输入下的运行结果

<pre>#include <iostream> using namespace std; int main() { unsigned char k; cin >> k; cout << "k=" << k << endl; return 0; }</pre>	<p>1、输入：A✓ （单个字符）</p>  <p>2、输入：\b✓ （反义字符）</p>  <p>4、输入：\101✓ （8进制数）</p> 
<p>基础知识：</p> <p>Unsigned char的最小值是： __0__</p> <p>Unsigned char最大值是： __255__</p>	<p>5、输入：65✓ （数字输入ASCII码值）</p>  <p>5、输入：-65✓ （数字输入ASCII码值）</p> 



§. 基础知识题 – 输入输出部分

2、cin的基本理解 – 单数据情况

F2-Compare. 运行下面的**对比**程序（cin输入与赋值），观察运行结果

```
#include <iostream>
using namespace std;
int main()
{
    unsigned char k1, k2, k3, k4, k5;
    k1 = 'A';
    k2 = '\b';
    k3 = '\101';
    k4 = 65;
    k5 = -65;

    cout << k1 << endl;
    cout << k2 << endl;
    cout << k3 << endl;
    cout << k4 << endl;
    cout << k5 << endl;
    return 0;
}
```

Microsoft Visual Studio 调试控制台

A
A
A
A

F:\C++练习\各种习题练习\Debug\各种习题练习.exe (进程 17916) 已退出，
要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->
按任意键关闭此窗口。...



总结：cin输入时不能输入转义字符和八进制数对应的ASCII码值的字符，可以输入单个字符，输入多个字符只读取第一个字符。signed char和unsigned char在ASCII为负值时都可以输出，但都是不可见字符。



§. 基础知识题 – 输入输出部分

2、cin的基本理解 – 单数据情况

G. 仿照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：

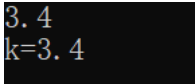
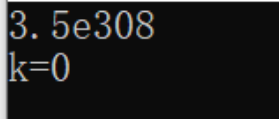
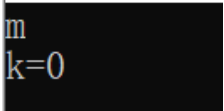

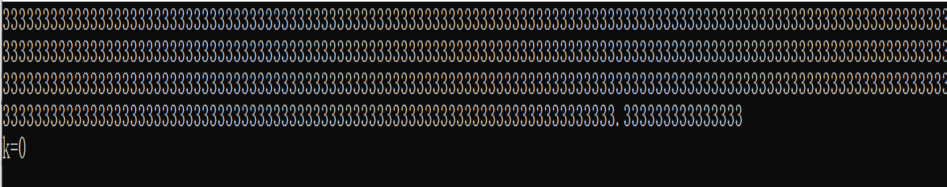
- a) 小数形式：输入正确且范围合理、输入正确但超上下限范围、输入错误
- b) 指数形式：输入正确且范围合理、输入正确但超上下限范围、输入错误
- c) 输入的有效位数超过该类型数据有效位数的情况下是如何处理的
- d) 和float/double型变量赋值的情况进行对比



§. 基础知识题 – 输入输出部分

2、cin的基本理解 – 单数据情况

G2. 运行下面的程序，观察不同输入下的运行结果

<pre>#include <iostream> using namespace std; int main() { double k; cin >> k; cout << "k=" << k << endl; return 0; }</pre>	<div>1、输入：3.4✓（单个字符） </div> <div>2、输入：3.5e308✓（反义字符） </div> <div>3、输入：m✓（8进制数） </div>
<p>基础知识：</p> <p>double的最小值是：1.7e-308_____</p> <p>double最大值是：_1.7e308__</p>	<div>4、输入：3.5e40✓（数字输入ASCII码值） </div> <div>5、输入：333...33333.333333333333✓（数字输入ASCII码值） <div>350个</div></div>



§. 基础知识题 - 输入输出部分

2、cin的基本理解 - 单数据情况

G2-Compare. 运行下面的**对比**程序（cin输入与赋值），观察运行结果

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      doule k1, k2, k3, k4;
6      k1 = 3.4;
7      k2 = 3.5e308;
8      k3 = 3.5e40;
9      k4 = 33333333.333333;
10
11      cout << "k=" << << endl;
12
13      return 0;
14 }
```

% 3 0

列表

个解决方案 错误 5 警告 0 消息 0 生成 + IntelliSense

代码	说明
E0020	未定义标识符 "doule"
E0030	浮点常量超出范围
E0029	应输入表达式
C2177	常量太大
C2059	语法错误: "<<"



总结：

float和double用cin输入时当输入越界的数时统一都是输出0。

float直接输出越界的数时正数显示inf，负数显示-inf。

double直接输出越界的数时显示错误越界，不能进行输入。



§. 基础知识题 - 输入输出部分

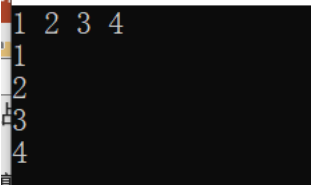
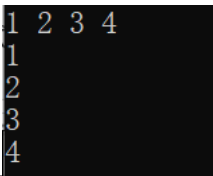
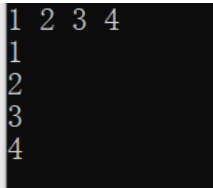
此页不要删除，也没有意义，仅仅为了分隔题目



§ . 基础知识题 – 输入输出部分

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：
第二个cin语句是一条语句，分为4行；第三个cin语句是4句，一行一句，最后的结果都是一样的。



§. 基础知识题 - 输入输出部分

3、cin的基本理解 - 多个同类型数据的情况

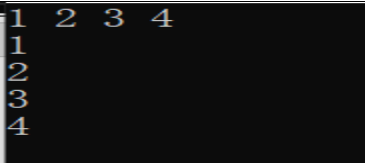
B. 程序同A，观察不同输入下的运行结果（贴图在清晰可辨的情况下尽可能小）

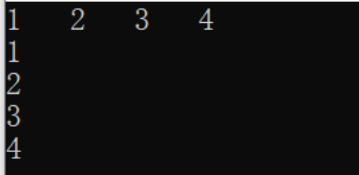
```
#include <iostream>
using namespace std;

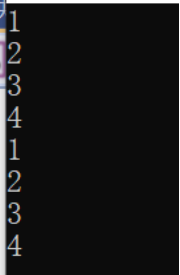
int main()
{
    int a, b, c, d;
    cin >> a >> b >> c >> d;

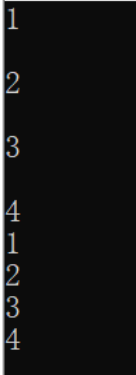
    cout << a << endl;
    cout << b << endl;
    cout << c << endl;
    cout << d << endl;

    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)


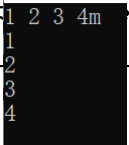
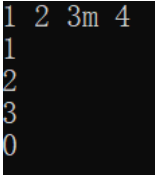
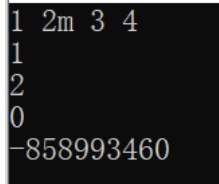
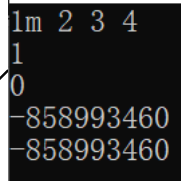
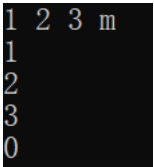
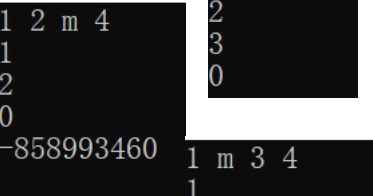
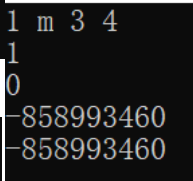
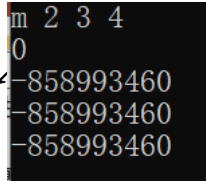
结论：在输入正确的情况下，回车和空格的作用？
分隔数据，把不同的数据放到不同的变量中。



§. 基础知识题 - 输入输出部分

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，能小）

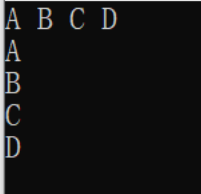
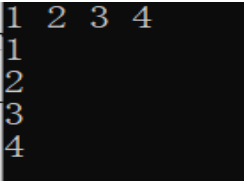
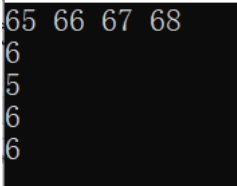
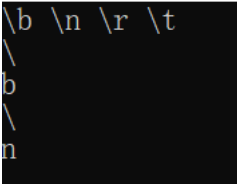
<pre>#include <iostream> using namespace std; int main() { int a, b, c, d; cin >> a >> b >> c >> d; cout << a << endl; cout << b << endl; cout << c << endl; cout << d << endl; return 0; }</pre>	<div>1、输入: 1 2 3 4m✓ </div> <div>2、输入: 1 2 3m 4✓ </div> <div>3、输入: 1 2m 3 4✓ </div> <div>4、输入: 1m 2 3 4✓ </div> <div>5、输入: 1 2 3 m✓ </div> <div>6、输入: 1 2 m 4✓ </div> <div>7、输入: 1 m 3 4✓ </div> <div>8、输入: m 2 3 4✓ </div> <div><p>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</p><p>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</p></div>
--	---



§. 基础知识题 – 输入输出部分

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>	<div>1、输入：A B C D✓ </div> <div>2、输入：1 2 3 4✓ </div> <div>3、输入：65 66 67 68✓ </div> <div>4、输入：\b \n \r \t </div> <div>总结： 多个char输入时，char读取是一个字符一个字符读取，不能读取一整个转义字符，也不能读取字符对应的ASCII码值。</div>
--	---



§. 基础知识题 - 输入输出部分

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
(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;
}
```

1、输入: 3.4 1.2 0.87 4.563✓

```
3.4 1.2 0.87 4.563
3.4
1.2
0.87
4.563
```

2、输入: 4e38 -4e38 50e300 -50e300✓

```
4e38 -4e38 50e300 -50e300
0
-1.07374e+08
-1.07374e+08
-1.07374e+08
```

3、输入: 4e38 2.3 4.8 23 ✓

```
4e38 2.3 4.8 23
0
-1.07374e+08
-1.07374e+08
-1.07374e+08
```

4、输入: 2.3 m 4.7 32

```
2.3 m 4.7 32
2.3
0
-1.07374e+08
-1.07374e+08
```

总结:

多个float输入时，如果输入的数据超范围则该数据为0，并且后面的数据数值与输入无关，均是系统的随机值，如果输入错误值，结果也是系统给的随机值



§. 基础知识题 – 输入输出部分

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
(贴图在清晰可辨的情况下尽可能小)

```
#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: 3.4 1.2 0.87 4.563✓

```
3.4 1.2 0.87 4.563
3.4
1.2
0.87
4.563
```

2、输入: 4e38 -4e38 50e300 -50e300✓

```
4e38 -4e38 50e300 -50e300
4e+38
-4e+38
5e+301
-5e+301
```

3、输入: 5e400 2.3 4.8 23 ✓

```
5e400 2.3 4.8 23
0
-9.25596e+61
-9.25596e+61
-9.25596e+61
```

4、输入: 2.3 m 4.7 32

```
2.3 m 4.7 32
2.3
0
-9.25596e+61
-9.25596e+61
```

总结:
多个double输入时，如果输入的数据超范围则该数据为0，并且后面的数据数值与输入无关，均是系统的随机值，如果输入错误值，结果也是系统给的随机值



§. 基础知识题 - 输入输出部分

此页不要删除，也没有意义，仅仅为了分隔题目



§. 基础知识题 – 输入输出部分

4、cin的基本理解 – 其他情况

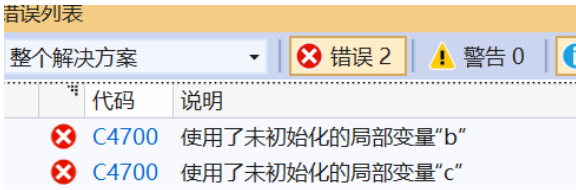
A. 程序如下，观察编译及运行结果（贴图在清晰可辨的情况下尽可能小）

```
#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;
}
```

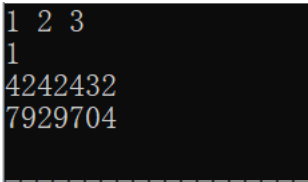
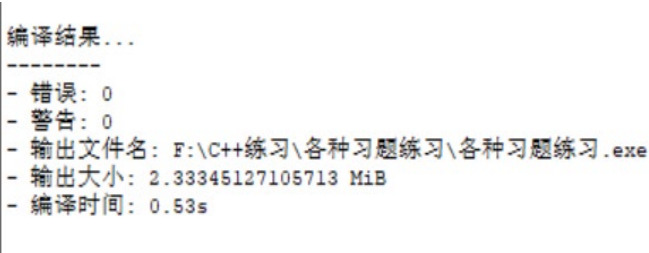
1、如果编译有error或warning，则贴相应信息的截图

VS 错误，运行不了



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

DEV
可以运行



3、分析为什么只有某个变量的结果是正确的

优先级问题，按位运算符优先与逗号运算符，则只有cin>>a结合，后面的b，c 并无输入，并且也没有赋数值，所以是系统给的随机值。

本题要求VS+Dev



§. 基础知识题 - 输入输出部分

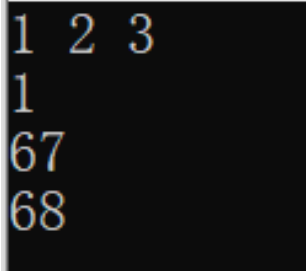
4、cin的基本理解 - 其他情况

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？

优先级问题，按位运算符优先与逗号运算符，则只有cin>>a结合，所以输入时给a赋值为1生效，但是后面的b，c输入时无效，所以输出时还是输出原来定义的数值。



§. 基础知识题 – 输入输出部分

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）

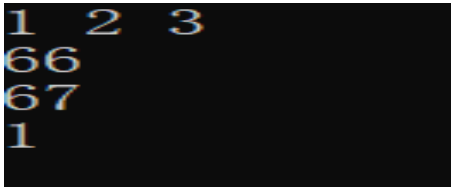
<pre>#include <iostream> using namespace std; int main() { int a; cin >> 5; cin >> a+10; cout << a << endl; return 0; }</pre>	<div>1、如果编译有error或warning，则贴相应信息的截图(信息太多则前五五行)</div> <div></div> <div>2、分析为什么编译有错 输入时后面接常量5，常量不需要输入，后面的表达式也是看成常量，都是不需要输入的，输入的话，流提取运算符应该是变量。</div> <div>3、结论：流提取运算符后面必须跟__b__，不能是__ac__ a) 常量 b) 变量 c) 表达式</div> <div>本题要求VS+Dev</div>
--	---



§. 基础知识题 - 输入输出部分

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）

<pre>#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; }</pre>	<div>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</div> <div></div> <div>2、通过观察三个变量的输出，你得到了什么结论？ 输入的是括号中的逗号表达式，这是一个值也就是c的值，所以输入的值改变了c的值，而a和b的值并没有改变。</div> <div>3、和B进行比较，分析为什么结果有差异 这个有括号，是一个值，这个值等于c的值，所以输入时只改变了c的值，而且是一个值，也就是1，b和c的值并没有改变。所以仍然是66和67。</div> <div>4、和C进行比较，与C得出的结论矛盾吗？ 并不矛盾，因为这个括号，将整体变成变量，而变量的值就是c的值，所以相当于重新输入，改变c的值。</div>
--	--



§ . 基础知识题 - 输入输出部分

4、cin的基本理解 - 其他情况

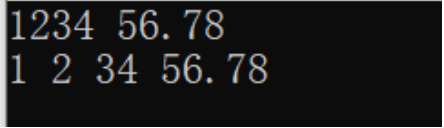
E. 程序如下，观察编译及运行结果（贴图在清晰可辨的情况下尽可能小）

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

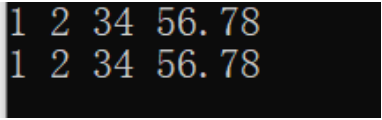
    cout << c1 << ' ' << c2 << ' ' << a << ' ' << b << endl;
    return 0;
}
```

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)
第一种情况下，char型只能取一个字符，则取的是1和2分别给c1和c2，34后面有空格分开两个数据给a和b
第二种情况下，每一个空格把数据分开，给不同的变量赋值，则两种输入得到的输出结果相同。



§. 基础知识题 – 输入输出部分

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（信息太多则前五五行）



2、结论：在cin中不能跟__endl_____

本题要求VS+Dev



§. 基础知识题 - 输入输出部分

此页不要删除，也没有意义，仅仅为了分隔题目



§. 基础知识题 – 输入输出部分

5、putchar的基本使用

字符输出函数putchar的基本知识:

形式: putchar(字符变量/常量)

功能: 输出一个字符

```
char a='A';  
putchar(a);  
putchar('A');  
putchar('\x41');  
putchar('\101');
```

} 均表示输出'A'

★ 某些编译器需要 #include <cstdio> 或 #include <stdio.h> (目前所用的双编译器均不需要)

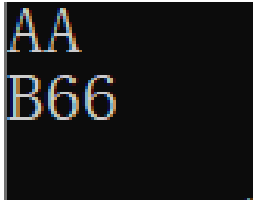
★ 返回值是int型, 是输出字符的ASCII码, 可赋值给字符型/整型变量



§. 基础知识题 – 输入输出部分

5、putchar的基本使用

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）

<pre>#include <iostream> #include <cstdio> using namespace std; int main() { char ret1; cout << (ret1 = putchar('A')) << endl; int ret2; cout << (ret2 = putchar('B')) << endl; return 0; }</pre>	<div>1、观察运行结果</div>  <div>2、分析运行结果中各输出是哪个语句/函数造成的 (可选: cout/putchar)</div> <p>先算括号里面的，则先输出的是A和B，然后因为 <code>putchar('A')</code> 的值为整型，所以把A对应的ASCII码值65赋给了ret1，但是ret1是字符型，所以输出的又是A 第二个ret2是整型，<code>putchar('B')</code> 的值66赋给了ret2，输出的也是66</p> <div>3、这个例子能确认上个Page的基本知识中的说法： “返回值是int型，是输出字符的ASCII码” 完全正确/部分正确吗？ 不能证明完全正确，因为如果说putchar的返回值是字符型的话，第一种情况和此题的一样，第二种情况因为整型提升，所以输出也是一样的，所以没办法排除返回结果为字符型的情况。</div>
---	---



§. 基础知识题 - 输入输出部分

5、putchar的基本使用

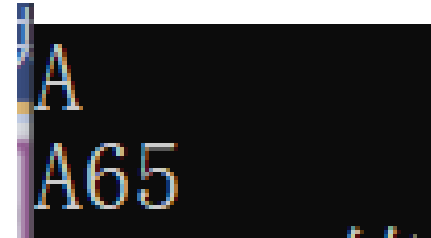
B. 自行构造测试程序，证明putchar的返回值是int型(要求两种方法)

```
//方法一
#include <iostream>
#include <stdio>
using namespace std;
int main()
{
    cout << sizeof(putchar('A')) << endl;
    cout << sizeof(int) << endl;
    return 0;
}
```



putchar的值的字节与int都是4，证明不是char型。

```
//方法2
#include <iostream>
#include <stdio>
using namespace std;
int main()
{
    cout << 'A' << endl;
    cout << putchar('A');
    return 0;
}
```



单独输出字符时就是输出A，而输出putchar时就是输出65，证明返回值不是char而是int。



§. 基础知识题 - 输入输出部分

此页不要删除，也没有意义，仅仅为了分隔题目



§. 基础知识题 – 输入输出部分

6、getchar的基本使用

字符输入函数getchar的基本知识:

形式: `getchar()`

功能: 输入一个字符 (给指定的变量)

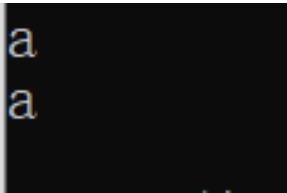

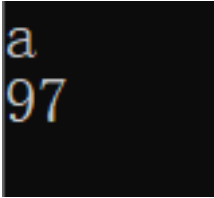
- ★ 某些编译器需要 `#include <cstdio>` 或 `#include <stdio.h>` (目前所用的双编译器均不需要)
- ★ 返回值是int型, 是输入字符的ASCII码, 可赋值给字符型/整型变量
- ★ 输入有回显, 而且不是键盘输入一个字符后立即执行`getchar`, 必须要等按回车后才执行
(弄清楚上课课件中的输入缓冲区的概念)
- ★ 可以输入空格, 回车等`cin`无法处理的非图形字符, 但仍不能处理转义符
- ★ `getchar/cin`等每次仅从输入缓冲区中取需要的字节, 多余的字节仍保留在输入缓冲区中供下次读取



§ . 基础知识题 - 输入输出部分

6、getchar的基本使用

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）

<pre>#include <iostream> #include <cstdio> using namespace std; int main() { char ch; ch = getchar(); cout << ch << endl; return 0; }</pre> 	<pre>#include <iostream> #include <cstdio> using namespace std; int main() { char ch; cout << (ch = getchar()) << endl; return 0; }</pre> 	<pre>#include <iostream> #include <cstdio> using namespace std; int main() { int ch; ch = getchar(); cout << ch << endl; return 0; }</pre> 
输入：a✓ 输出：__a__ 输出的是：_ch的值____ (ch的值/赋值表达式值)	输入：a✓ 输出：__a__ 输出的是：__赋值表达式值____ (ch的值/赋值表达式值)	输入：a✓ 输出：__97__

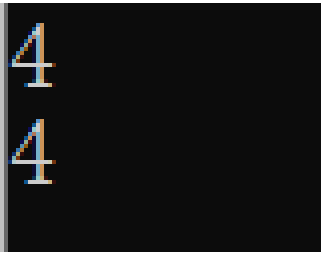


§. 基础知识题 - 输入输出部分

6、getchar的基本使用

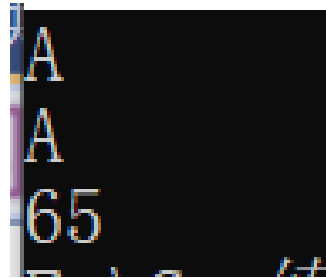
B. 自行构造测试程序，证明getchar的返回值是int型(要求两种方法)

```
//方法一
#include <iostream>
#include <cstdio>
using namespace std;
int main()
{
    cout << sizeof(getchar()) << endl;
    cout << sizeof(int) << endl;
    return 0;
}
```



getchar的值的字节与int都是4，证明不是char型。

```
//方法2
#include <iostream>
#include <cstdio>
using namespace std;
int main()
{
    cout << 'A' << endl;
    cout << getchar();
    return 0;
}
```



输出字符的话显示的就是字符，而输出getchar，输出的是65，65是int型，所以可以证明。



§ . 基础知识题 - 输入输出部分

6、getchar的基本使用

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> #include <cstdio> using namespace std; int main() { char ch; ch = getchar(); cout << int(ch) << endl; return 0; }</pre>	<div><div>1、键盘输入：Hello↵（5个字母+回车）</div><div>72</div></div> <div><div>2、键盘输入：↵（空回车）</div><div>10</div></div> <div><div>3、键盘输入：␣↵（空格+回车）</div><div>32</div></div> <div><div>4、键盘输入：␣↵（空格+回车）</div><div>32</div></div> <div><div>5、键盘输入：\n↵（2个字符+回车）</div><div>92</div></div> <div><div>6、键盘输入：\101↵（4个字符+回车）</div><div>92</div></div> <div><div>结论：可以输入__a__、__c__等cin无法处理的非图形字符，但仍不能处理__b__</div><div>a) 空格 b) 转义符 c) 回车</div></div>
---	--



§. 基础知识题 - 输入输出部分

6、getchar的基本使用

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>
#include <cstdio>
using namespace std;
int main()
{
    cout << "--Step1--" << endl;
    cout << getchar() << endl;

    cout << "--Step2--" << endl;
    cout << getchar() << endl;

    cout << "--Step3--" << endl;
    cout << getchar() << endl;

    cout << "--Step4--" << endl;
    cout << getchar() << endl;
    return 0;
}
```

本次要求仔细观察运行现象及结果，特别是Step1~4出现的时机!!!

1、每次输入一个回车

程序从开始执行到结束，共停顿了__4__次来等待输入

第1次停顿时，屏幕上输出的最后一行是Step__1__ ?

第2次停顿时，屏幕上输出的最后一行是Step__2__ ? (没有则不填)

第3次停顿时，屏幕上输出的最后一行是Step__3__ ? (没有则不填)

第4次停顿时，屏幕上输出的最后一行是Step__4__ ? (没有则不填)

2、第一次输入一个字母+回车，以后每次停顿，均输入一个字母+回车

程序从开始执行到结束，共停顿了__2__次来等待输入

第1次停顿时，屏幕上输出的最后一行是Step__1__ ?

第2次停顿时，屏幕上输出的最后一行是Step__3__ ? (没有则不填)

第3次停顿时，屏幕上输出的最后一行是Step_____ ? (没有则不填)

第4次停顿时，屏幕上输出的最后一行是Step_____ ? (没有则不填)

3、第一次即输入4个以上的字母+回车

程序从开始执行到结束，共停顿了__1__次来等待输入

第1次停顿时，屏幕上输出的最后一行是Step__1__ ?

第2次停顿时，屏幕上输出的最后一行是Step_____ ? (没有则不填)

第3次停顿时，屏幕上输出的最后一行是Step_____ ? (没有则不填)

第4次停顿时，屏幕上输出的最后一行是Step_____ ? (没有则不填)

结论：getchar每次仅从输入缓冲区中取需要的字节，多余的字节仍保留在输入缓冲区中供下次读取

思考：结合3.c的例子，考虑一下3.c中非法m对int的影响(错在第几个数)与输入缓冲区的关系，为什么?)



我对于上面问题的思考：

`cin`函数依次从输入流中提取数据为变量一一赋值，当提取到为一个变量赋了一个值`m`时，我们这是在尝试为整型变量赋字符类型值`m`，所以C++会自动设置输入失效位，并关闭输入，此后所有的`cin`语句都不会被执行，错误的输入`m`和后面的正确输入的值会保留在输入流中，但程序不会终止，照常执行其他语句。这就会出现系统随机值。

改善方法：用`cin.fail`等函数 这些都是我花了很长时间去研究的，我也给sj老师发过，请老师过目。

```
#include <iostream>
using namespace std;
int main()
{
    int a,b,c;
    cin >> a>>b>>c;
    while (cin.fail()) /*判断cin的状态，如果输入错误数据将会自动进入循环*/
    {
        cin.clear();    //清除cin的错误状态
        cin.ignore();    //忽略掉缓冲区的内容
        cout << "no";
        cin >> a>>b>>c;
    }cout << a<<b<<c << endl;
    return 0;
}
```



§ . 基础知识题 – 输入输出部分

6、getchar的基本使用

E. 自行构造证明D结论的使用cin读入的测试程序

```
#include <iostream>
using namespace std;
int main()
{
    char a, b, c, d;
    cout << "--Step1--" << endl;
    cin >> a;
    cout << "--Step2--" << endl;
    cin >> b;
    cout << "--Step3--" << endl;
    cin >> c;
    cout << "--Step4--" << endl;
    cin >> d;
    return 0;
}
```

本次要求仔细观察运行现象及结果，特别是Stepx出现的时机!!!

因为cin不能读取空格、回车（有特殊方法可读，先忽略），因此测试有所不同

- 1、第一次输入两个字母+回车，以后每次停顿，均输入两个字母+回车
程序从开始执行到结束，共停顿了下2次来等待输入
第1次停顿，屏幕上输出的最后一行是Step_1_ ?
第2次停顿，屏幕上输出的最后一行是Step_3_ ?（没有则不填）
第3次停顿，屏幕上输出的最后一行是Step_____ ?（没有则不填）
第4次停顿，屏幕上输出的最后一行是Step_____ ?（没有则不填）
- 2、第一次即输入4个以上的字母+回车
程序从开始执行到结束，共停顿了下1次来等待输入
第1次停顿，屏幕上输出的最后一行是Step____1_ ?
第2次停顿，屏幕上输出的最后一行是Step_____ ?（没有则不填）
第3次停顿，屏幕上输出的最后一行是Step_____ ?（没有则不填）
第4次停顿，屏幕上输出的最后一行是Step_____ ?（没有则不填）

结论：cin每次仅从输入缓冲区中取需要的字节，多余的字节仍保留在输入缓冲区中供下次读取



§. 基础知识题 - 输入输出部分

此页不要删除，也没有意义，仅仅为了分隔题目



§. 基础知识题 - 输入输出部分

7、getchar、_getch与_getche的基本使用

- 1、测试时cmd窗口下面不能是中文输入法
- 2、<conio.h>是_getch()/_getche()需要的头文件

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）

<pre>#include <iostream> using namespace std; int main() { char ch; ch = getchar(); cout << (int)ch << endl; return 0; }</pre>	<pre>#include <iostream> #include <conio.h> using namespace std; int main() { char ch; ch = _getch(); cout << (int)ch << endl; return 0; }</pre>	<pre>#include <iostream> #include <conio.h> using namespace std; int main() { char ch; ch = _getche(); cout << (int)ch << endl; return 0; }</pre>
<div>1、输入：a✓ 输出： 97 输入回显： 有 (有/无) 按回车生效： 是 (是/否)</div> <div>2、输入： ✓ (直接回车) 输出： 10</div>	<div>1、输入：a✓ 输出： 97 输入回显： 无 (有/无) 按回车生效： 否 (是/否)</div> <div>2、输入： ✓ (直接回车) 输出： 13</div>	<div>1、输入：a✓ 输出： a97 输入回显： 有 (有/无) 按回车生效： 否 (是/否)</div> <div>2、输入： ✓ (直接回车) 输出： 13</div>
<div>注：直接按回车时的差异可，具体原因有兴趣自己课外提供技术支持</div>		

本题要求
VS+Dev



我对于作业最后的理解：

`getchar()`是从键盘的文件缓冲区读取字符，当从键盘输入字符时可以输入若干个字符（但是只读取一个），并且最后一定要回车确认。

而`getch()`和`getche()`是直接从键盘读取（不经过文件缓冲区），不需要回车确认，比如`getch()`输入一个a的话直接屏幕显示97，不需要你按回车等其他键。

针对回显问题，其实可以理解成你键盘输入的数据在屏幕上能否显示出来，那么就会发现`getchar`和`getche`是回显的。

针对直接输出回车问题，前面说到`getchar`是从文件缓冲区读取的，所以输入回车的时候，文件缓冲区自动将回车键转化成换行符'`\n`'，所以大家会看到ASCII码为10，但是`getch`和`getche`不需要转化，直接读的是回车键'`\r`'，它的ASCII码值是13，这样问题就迎刃而解了。



§. 基础知识题 - 输入输出部分

7、getchar、_getch与_getche的基本使用

- 1、测试时cmd窗口下面不能是中文输入法
- 2、<conio.h>是_getch()/_getche()需要的头文件

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）

<div>本题要求VS+Dev</div> <div>哪个编译器报错？ 哪个编译器下结果同A？</div> <div>VS报错 DEV结果同A</div>	<pre>#include <iostream> #include <conio.h> using namespace std; int main() { char ch; ch = getch(); cout << (int)ch << endl; return 0; }</pre> <div>C4996 'getch': The POSIX name for this symbol is _getch. See the documentation for details. 错误</div> <div>97</div>	<pre>#include <iostream> #include <conio.h> using namespace std; int main() { char ch; ch = getche(); cout << (int)ch << endl; return 0; }</pre> <div>C4996 'getche': The POSIX name for this symbol is _getche. See the documentation for details.</div> <div>a97</div>
	<div>1、输入：a✓ 输出：__97__ 输入回显：__无__（有/无） 按回车生效：__否__（是/否）</div> <div>2、输入：✓（直接回车） 输出：__13__</div>	<div>1、输入：a✓ 输出：__a97__ 输入回显：__有__（有/无） 按回车生效：__否__（是/否）</div> <div>2、输入：✓（直接回车） 输出：__13__</div>



§. 基础知识题 - 输入输出部分

此页不要删除，也没有意义，仅仅为了分隔题目