



§. 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、**3月19日前（两周）**网上提交本次作业（在“文档作业”中提交）

特别说明:

- 1、本次作业是预习作业，在**第三周第一次上课前完成效果更好**
- 2、对于作业过程中不清楚的问题或不会的内容，先不要问（不清楚的位置可以先做个标记，结合听课再去理解）

§. 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 - 浮点数机内存储格式(IEEE 754)理解



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

The screenshot shows the Visual Studio 2022 IDE interface. In the top-left, there's a code editor window titled "demo.cpp" containing the following C++ code:

```
1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     cout << "Hello, 同济!" << endl;
6     return 0;
7 }
```

In the bottom-right corner of the code editor, there is a small dropdown menu with three options: "CRLF", "LF", and "CR". The "CR" option is checked. A red arrow points from the text above to this dropdown menu.

Below the code editor is the "Output" window. It displays the build log:

```
生成开始于 22:23...
1>—— 已启动生成: 项目: demo-cpp, 配置: Debug Win32 ——
1>demo.cpp
1>D:\WorkSpace\VS2022-demo\demo-cpp\demo.cpp(1,1): warning C4335: 检测到 Mac 文件格式: 请将源文件转换为 DOS 格式或 UNIX 格式
1>D:\WorkSpace\VS2022-demo\demo-cpp\demo.cpp(1,10): warning C4067: 预处理器指令后有意外标记 - 应输入换行符
1>MSVCRTD.lib(exe_main.obj) : error LNK2019: 无法解析的外部符号 _main, 函数 "int __cdecl invoke_main(void)" (?invoke_main@@YAHXZ) 中引用了该符号
1>D:\WorkSpace\VS2022-demo\Debug\demo-cpp.exe : fatal error LNK1120: 1 个无法解析的外部命令
1>已完成生成项目“demo-cpp.vcxproj”的操作 - 失败。
===== 生成: 0 成功, 1 失败, 0 最新, 0 已跳过 =====
===== 生成于 22:23 完成, 耗时 01.132 秒 =====
```

A red box highlights the warning message about the file format. Another red arrow points from the text above to the bottom of the output window, near the status bar.

§. 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，下一个也是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 P. 13-23 !!!



§. 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;
}
```

```
选择 Microsoft Visual Studio 调试控制台
This is a C++ program.

D:\XUE\CTF\VS2022\homework\Debug\模块3.exe (进程 12540) 已退出, 代码为 0
按任意键关闭此窗口. . .
```

第3组和第4组在语句上的区别是：第3组是全部存在缓冲区里一起输出，第4组是每个存在缓冲区里立刻输出再存入下一个。



§. 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)

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

int main()
{
    int a=10, b=15, c=20;
    cout << a << b << c;
    return 0;
}
```

101520

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

int main()
{
    int a=10, b=15, c=20;
    cout << a, b, c;
    return 0;
}
```

10

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

int main()
{
    int a=10, b=15, c=20;
    cout << (a, b, c) << endl;
    return 0;
}
```

20

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

int main()
{
    int a=10, b=15, c=20;
    cout << a, b, c << endl;
    return 0;
}
```

error C2568: “<<”：无法解析函数重载
at

解释这3个程序输出不同的原因：
1、cout操作符按顺序输出a, b, c的值。
2、使用了逗号运算符，<<优先级比，高，先输出a，后面没有<<, 不在输出。
3、使用括号和逗号运算符，先进行括号内的运算，运算结果是c的值，输出c的值。

解释错误原因：endl没有赋值给cout，赋值给了c，因此报错。

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

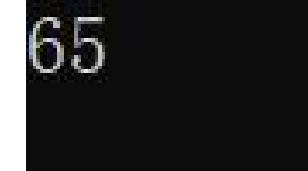
1. cout的基本理解

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：第一个ch的类型是字符，A对应的ASCLL码是65，输出A。第二个ch类型是整型，直接输出65

§. 基础知识题 - cin与cout的基本使用



1. cout的基本理解

D. 程序同C，将修改后符合要求的程序及运行结果贴上

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

Microsoft Visual Studio 调试控制台
65

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

Microsoft Visual Studio 调试控制台
A

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 - 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+0) << endl;
7 }
8
9
```

Microsoft Visual Studio 调试控制台
65

在char类型不变的情况下，要求输出为65
(不允许添加其它变量，
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 << cin.good();
    cout << " k=" << k << endl;

    return 0;
}
```

基础知识：

short的最小值是：-32767

short的最大值是：32767

全部做一遍，任选3题截图即可
(多截不限)

1、输入：456↙ (↙代表回车键，下同) 456
1 k=456

2、输入：456 123↙ (一个空格) 456 123
1 k=456

3、输入：456 123↙ (多个空格) 456 123
1 k=456

4、输入：456m↙ 456m
1 k=456

5、输入：x↙ 0 k=0

6、输入： 456↙ (持续多个空格后，再输入456，按回车) 456
1 k=456

7、输入： ↙ (持续多个空格后，按回车)
456↙ (再输入456，按回车) 456
1 k=456

8、输入： ↙
...
456↙ (持续多个空回车后，输入456) 456
1 k=456

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？
输入终止。

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？
cin.good输出0，代表非法输入，k=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值是不可信的

全部做一遍，任选2题截图即可(多截不限)

贴图即可，不需要写分析结果

1、输入： 456↙ (正确+回车)

```
456
k=456
cin.good()=1
cin.fail()=0
```

2、输入： 456↙123↙ (正确+空格)

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

3、输入： -456m↙ (正确+非法字符)

```
-456m
k=-456
cin.good()=1
cin.fail()=0
```

4、输入： m↙ (直接非法字符)

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

5、输入： 54321↙ (超上限)

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

6、输入： -40000↙ (超下限)

```
-40000
k=-32768
cin.good()=0
cin.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

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

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

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

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

    return 0;
}
```

B的输入:

- 1、输入: 12345↙ (合理范围)
对应本例的k1=12345
- 2、输入: 54321↙ (超上限但未超同类型的u_short上限)
对应本例的k2=-11215
- 3、输入: 70000↙ (超上限且超过同类型的u_short上限)
对应本例的k3=4464
- 4、输入: -12345↙ (合理范围)
对应本例的k4=-12345
- 5、输入: -54321↙ (超下限)
对应本例的k5=11215

u_short=unsigned short

**分析: cin超范围时, 取short型上限或下限, 且cin. good()=0。
赋值超范围时, 会取补数。**



§. 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)

```
#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;
}
```

结论:

多个输入中, 编号2, 3, 5输入的k值是不可信的

贴图即可, 不需要写分析结果

u_int=unsigned int

1、输入: 1234567890 ↗ (合理范围)

```
1234567890  
k=1234567890  
cin.good()=1  
cin.fail()=0
```

2、输入: 3000000000 ↗ (超上限但未超同类型的u_int上限)

```
3000000000  
k=2147483647  
cin.good()=0  
cin.fail()=1
```

3、输入: 5000000000 ↗ (超上限且超过同类型的u_int上限)

```
5000000000  
k=2147483647  
cin.good()=0  
cin.fail()=1
```

4、输入: -1234567890 ↗ (合理范围)

```
-1234567890  
k=-1234567890  
cin.good()=1  
cin.fail()=0
```

5、输入: -3000000000 ↗ (超下限)

```
-3000000000  
k=-2147483648  
cin.good()=0  
cin.fail()=1
```

全部做一遍, 任选2题截图即可(多截不限)

本题要求VS+Dev



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

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。输入超范围直接取int型上限，赋值超范围会取补数。

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。输入超范围直接取int型上限，赋值超范围会取补数，超出部分从-2147483648开始重新往上加。

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。输入超范围直接取int型下限，赋值超范围会取补数，超出部分从2147483647开始重新往下减。



§. 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;
}
```

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

结论：

多个输入中，编号 2, 6 输入的k值是不可信的

全部做一遍，任选2题截图即可(多截不限)

本题要求VS+Dev



§. 基础知识题 - cin与cout的基本使用

2. cin的基本理解 - 单数据情况

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()
{
    u_short=u_short=unsigned short
    Dev
    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;
}
```

贴图即可（有warning还有贴warning），不需要写分析结果

1、输入：12345↙（合理范围） VS
对应本例的k1=12345

warning C4305: “=” : 从“int”到“unsigned short”截断
warning C4309: “=” : 截断常量值
warning C4309: “=” : 截断常量值

2、输入：70000↙（超上限）
对应本例的k2=4464

12345
4464
53191
65535
1
0

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

本题要求VS+Dev



§. 基础知识题 - cin与cout的基本使用

2. cin的基本理解 - 单数据情况

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

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

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

贴图即可, 不需要写分析结果

u_int=unsigned int

1、输入: 1234567890 ↘ (合理范围) 1234567890
k=1234567890 good()=1 fail()=0

2、输入: 5000000000 ↘ (超上限) 5000000000
k=4294967295 good()=0 fail()=1

3、输入: -1234567890 ↘ (负数但未超int下限)
-1234567890
k=3060399406 good()=1 fail()=0

4、输入: -3000000000 ↘ (负数且未超过u_int上限加负号后的下限)
-3000000000
k=1294967296 good()=1 fail()=0

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

结论:

多个输入中, 编号 2, 5 输入的k值是不可信的

unsigned int 基本同 unsigned short, 弄懂即可
本页可以不做, 空着不扣分

本题要求VS+Dev



§. 基础知识题 - cin与cout的基本使用

2、cin的基本理解 - 单数据情况

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

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) :

```
1234567890  
705032704  
3060399406  
1294967296  
3589934592
```

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, 输入超范围取上限, 赋值超范围取补数。

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。

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。

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

不一致, 输入超范围取上限, 复制超范围取补数。

unsigned int 基本同 unsigned short, 弄懂即可
本页可以不做, 空着不扣分



§. 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
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
不相同，输入超范围取signed上限，赋值超范围取补数。

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
不相同，输入超范围取signed上限，赋值超范围取补数。

4、signed数据在输入正确但超下限范围的情况下
不相同，输入超范围取signed下限，赋值超范围取补数。

5、unsigned数据在输入正确且范围合理的情况下
相同。

6、unsigned数据在输入正确且超上限的情况下
不相同，输入超范围取unsigned上限，赋值超范围取补数。

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
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
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
不相同，输入超范围取unsigned上限，赋值超范围取补数。

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(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CtrlL+C (注意：是Ctrl+C组合键，注意不要有输入法栏)

7、键盘输入CtrlL+z (注意：是Ctrl+z组合键，注意不要有输入法栏)

全部做一遍，任选3题截图即可(多截不限)



§. 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 << cin.good() << ', ' << f << endl;
    cout << setprecision(20) << f << endl;

    return 0;
}

//注: setprecision(20)表示输出时保留
//    20位有效位数
//    (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
1 123.456
123.45600128173828125

2、键盘输入1.23456e2 (合理范围正数, 指数形式)

1.23456e2
1 123.456
123.45600128173828125

3、键盘输入-123.456 (合理范围负数, 小数形式)

-123.456
1 -123.456
-123.45600128173828125

4、键盘输入-1.23456e2 (合理范围负数, 指数形式)

-1.23456e2
1 -123.456
-123.45600128173828125

5、键盘输入123.456789 (合理范围, 但超有效位数)

123.456789
1 123.457
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
1.7e39
0 inf
inf

7、键盘输入1.7e39 (超上限且数量级已超 10^{38})

1.7e39
0 inf
inf

8、键盘输入-2.3e39 (超上限且数量级已超 10^{38})

9、键盘输入1.23e-30 (合理范围整数但指数很小)

10、键盘输入-1.23e-30 (合理范围负数但指数很小)

全部做一遍，任选4题截图即可(多截不限)

§. 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)

```
#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 2 3 4  
1  
2  
3  
4
```

```
#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 2 3 4  
1  
2  
3  
4
```

```
#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;
} 1 2 3 4  
1  
2  
3  
4
```

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：第二个程序是a, b, c, d都进入缓冲区再提取；第三个程序是每一个进入缓冲区先提取出来下一个在进入缓冲区。



§. 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;
}
```

全部做一遍, 任选2题截图即可
(多截不限)

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?
终止输入。



§. 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 4↙ | 1、 1 2 3 4↙ | 5、 1 2 3 m |
| 2、输入: 1 2 3m 4↙ | 2
3
4 | 2
3
0 |
| 3、输入: 1 2m 3 4↙ | | |
| 4、输入: 1m 2 3 4↙ | 1 | 1 2 3m 4 |
| 5、输入: 1 2 3 m↙ | 2
3 | 2
3
0 |
| 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?)

错误输入之前的值可信, 错误输入之后的值为0, 不可信。

全部做一遍, 任选3题截图即可
(多截不限)



§. 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;
}
```

全部做一遍，任选3题截图即可
(多截不限)

- 1、输入: XYZ↙ 1、 XYZ
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=-
- 4、输入: XCtrl+C↙ Xa=-
- 5、输入: XYCtrl+C↙ XYa=-
- 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, cin会忽略空格。

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?)

Ctrl+C 是一个中断信号, 用于中断当前运行的程序

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?)

Ctrl+z 是一个终端控制字符, 用于将当前运行的进程挂起

4、Ctrl+z后不按回车而继续输入的其它字符, 能否被读入?

如果程序被 Ctrl+Z 挂起, 继续输入的字符不会被立即读入, 因为程序已经暂停执行。



§. 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;
}
```

全部做一遍，任选2题截图即可(多截不限)

1、输入: 6. 7e39 123. 456 1. 23456e2↙ (第1个超上限, 2/3正常)

2、输入: -6. 7e39 123. 456 1. 23456e2↙ (第1个超下限, 2/3正常)

3、输入: 123. 456 6. 7e39 1. 23456e2↙ (1/3正常, 第2个超上限)

4、输入: 123. 456 -6. 7e39 1. 23456e2↙ (1/3正常, 第2个超下限)

5、输入: 123. 456 1. 23456e2 6. 7e39↙ (1/2正常, 第3个超上限)

6、输入: 123. 456 1. 23456e2 -6. 7e39↙ (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？)

从错误输入开始值都不可信，错误输入之前的值可信。

2、将float替换为double，上述结论是否仍然成立?
成立。

-6. 7e39 123. 456 1. 23456e2
a=-inf
-inf
b=-107374176
-107374176
c=-107374176
-107374176

123. 456 6. 7e39 1. 23456e2
a=123. 456
123. 45600128173828125
b=inf
inf
c=-107374176
-107374176

§. 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

1、
error C4700: 使用了未初始化的局部变量“b”
error C4700: 使用了未初始化的局部变量“c”

2、
D:\XUE\CTF\VS2022\homework\模块3\4.exe
1 2 3
1
4242432
7929704

- 3、一个流提取运算符只能提取1个数据，因此只有a被正常初始化，b和c未被初始化，只有a的输出结果正确。

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？

1、
1
67
68

2、每个流提取符号读入一个数据，其他数据无法提取。因此a会通过输入了赋值，b和c直接输出初始的值



§. 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）

```
#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(信息太多则前五行)

2、分析为什么编译有错

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

1、VS

error 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)

Dev

[Error] no match for 'operator>>' (operand types are 'std::istream' {aka 'std::basic_istream<char>'} and 'int')

2、5是一个常量，a+10是一个表达式，两者本身是不可赋值的，没有存储位置来接收输入的数据，因此编译出错。

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？
`cin >>`流提取符后可以跟逗号表达式，但要求最终运算的结果必须是一个变量，而不是一个表达式

3、和B进行比较，分析为什么结果有差异
此处为(a, b, c)这是一个逗号表达式，此时结果是c，c仍然是一个变量，
可以通过流提取符来输入

4、和C进行比较，与C得出的结论矛盾吗？
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：
1234 56.78
1 2 34 56.78

2、输入：1「2「34「56.78↙

输出：
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)

根据最长且合理原则，c1和c2是char型，只读取1和2，a是int型，读取3和4，空格起到输入终止的作用，b读取56.78。第二个程序都有空格，直接读取，两个程序结果一样。



§. 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(信息太多则前五行)
- 2、结论：在cin中不能跟_____换行符_____

VS error 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)

Dev

[Error] no match for 'operator>>' (operand types are 'std::basic_istream<char>::__istream_type' {aka 'std::basic_istream<char>'} and '<unresolved overload...'

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