首先关于输入的几个基本概念：

1. 我们从键盘输入的值会先存储到内存中一个部分：键盘缓冲区。系统在执行到 `cin>>` 时，会检测键盘缓冲区中有无内容，有则读入，没有则等待我们从终端输入。
2. `cin` 有一个函数叫做 `fail()` ，如果 `cin` 读取的过程中出现错误， `cin.fail()` 就会返回true。而且，在此之后的 `cin>>` 都会因为 `cin.fail()==true` 而不被执行。除非使用 `cin.clear()` 来清除这个错误标志。
3. 对于代码 `int a;cin>>a;` ，其中 `cin` 在对整数输入错误时，会先将整数变量赋值为0，然后再设置 `cin` 内部的错误标识设置为 `true`

以下代码为例：

```
int main()
{
   int a=-11,n=-22,m=-33;//对变量初始化，如果输出的变量值未变，则说明cin代码未生效
   cin>>a;
   if(cin.fail())
      cout<<"cin fail! a="<<a<<endl;
   else
      cout<<"success! a="<<a<<endl;

   cin>>n;
   if(cin.fail())
      cout<<"cin fail! n="<<n<<endl;
   else
      cout<<"success! n="<<n<<endl;

   //cin.clear();
   cin>>m;
   if(cin.fail())
      cout<<"cin fail! m="<<m<<endl;
   else
      cout<<"success! m="<<m<<endl;
}
```

输入 `2.3` 输出是 `2 0 -33`

执行的步骤如下：

1. 系统执行到 `cin>>a` 时，此时键盘缓冲区啥也没有，会等待我们的输入。然后输入 `2.3` ， `2.3` 进入缓冲区， `cin` 开始读取。由于 `cin` 函数想要获得的是一个整数，所以它会读取到 `2` ，后一个字符是 `.` 不属于整数， `cin` 便停止，但并不属于失败。
2. 现在缓冲区残留的是 `.3` ，执行到 `cin>>n` 的时候， `.3` 并不是一个有效的整数，此时 `cin` 会出错，首先将该变量初始化为0（所以输出 `n=0` ），其次将 `cin` 的错误标识置为 `true` 。
3. 现在执行到 `cin>>m` ，由于 `cin` 的错误标识为 `true` 。所以，这句话并未执行，直接跳过。输出 `m=-33` 表示 `m` 并未被赋值。后面无论有多少个 `cin>>x` ，变量x都不会被赋值

有几个实验可供验证：

1. 输入不变，将 `int n=-22;` 改为 `char n="o"`
2. 去除 `cin.clear()` 的注释

有什么问题可以下次课提问：）