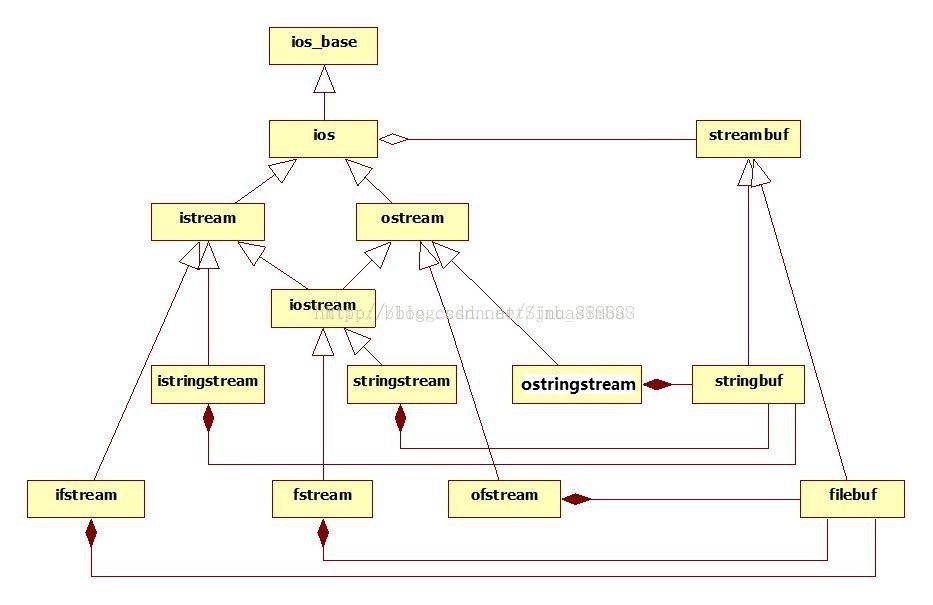
## io库

#### io库介绍

1. io就是input，output的简写，也就是输入输出功能。数据在内存，磁盘，输入输出设备之间移动就是io功能。



1. io库组成部分：
   1. C++定义了ios这个基类来定义输入输出的最基本操作，这个类的具体功能我们无需了解，只需了解C++io库所有的类都继承自这个类即可。
   2. istream，ostream这两个类直接继承自ios类。
      1. ostream类定义了从内存到输出设备（比如显示器）的功能，我们最常使用的cout就是ostream类的对象。
      2. istream类定义了从输入设备比如键盘）到内存的功能，我们最常用的cin就是istream类的对象。
      3. **iostream文件定义了ostream和istream类的对象，就是cout和cin。所以我们只要简单的引入iostream这个头文件，就可以方便的使用这两个对象**

**注意：这个输入，输入时相对于内存来说的，输入到内存，是istream。**

* 1. ifstream，ofstream类分别继承自istream类和ostream类。
     1. ifstream定义了从磁盘到内存的功能。因为istream重载了“<<”运算符，所以ifstream对象也可以用“<<”运算符来将文件数据写入内存。除了“=”的所有重载运算符都是可以被继承的。
     2. ofstream定义了从内存到磁盘的功能。与ifstream同理，也可以用“>>”操作数据流。
     3. **fstream文件引入了ifstream和ofstream，所以我们只要引入ftream这个头文件，就可以使用文件流功能了。**

注意：这个输入输出同样是相对内存来说的。

内存与输入输出设备的数据流动，磁盘与内存的数据流动已经介绍完了。磁盘和输入输出设备直接无法直接交互，必须通过内存。

io库还为我们额外定义了字符串的输入输出类，因为对字符串的操作极为频繁，所以这个库还是很有意义的。

* 1. istringstream，ostringstream分别继承自istream类和ostream类
     1. istringstream定义了从指定字符串到特定内存的功能。与ifstream同理，也可以用“<<”运算符操作数据。
     2. ostringstream定义了从特定内存到指定字符串的功能。可以用“>>”操作数据。
     3. **sstream头文件就引入了istringstream和ostringstream，所以我们只要引入sstream这个头文集，就可以使用字符串与内存直接交互数据的功能。**

所以我们使用io库主要就三个头文件，iostream，fstream，sstream。接下来三节课会对这三个文件的使用依次讲解。

#### io库的注意事项

1. **io对象无法使用拷贝构造函数和赋值运算符**

所以我们使用流对象无法使用值传递，一般使用引用进行传递。

1. **Io对象的状态**
   1. **io操作是非常容易出现错误的操作，一些错误是可以修复的，另一部分则发生在系统更深处，已经超出了应用程序可以修正的范围。**

比如我们使用cin向一个int类型的数中输入一个字符串，会使cin这个对象出现错误。

**所以我们在使用io对象时都应该判断io对象的状态。**

**比如：while(cin >> val) 或if(cin >> val)（不要只用这两个进行控制，最好搭配iostate来使用）**

* 1. 我们需要知道流对象错误的原因，因为不同的错误需要不同的处理方法。

io库定义了iostate类型，可以完整的表示io对象当前的状态。在不同的平台中， iostate实现方法略有区别，在vs中直接用int来代表iostate类型，不同的位置 表示不同的状态。可以与位操作符一起使用来一次检测或设置多个标志位。

可以用**rdstat函数**来获得io对象当前用iostat类型来表示的状态：

* 1. iostata类型有以下状态
     1. badbit状态，系统级错误，一旦表示badbit的位被置为1，流对象就再也无法使用了。
     2. failbit状态，代表可恢复错误，比如想读取一个数字却读取了一个字符，这种错误就是可以恢复的。当badbit位被置1时，failbit位也会被置1。
     3. eofbit状态，当到达文件结束位置时，eofbit和failbit位都会被置1。
     4. goodbit状态，表示流对象没有任何错误。

只要badbit，failbit，eofbit有一位被置为1，则检测流状态的条件就会失败。

* 1. 标准库还定义了一组成员函数来查询这些标志位的状态
     1. good()函数在所有错误位均未置1的情况下返回true。
     2. bad()，fail()，eof()函数在对应位被置1的情况下返回true。因为badbit位被置1或eofbit位被置1时，failbit位也会被置为1。所以用fail()函数可以准确判断出流对象是否出现错误。
     3. 实际上，我们将流对象当做条件使用的代码就等价于“!fail()”
  2. 流对象的管理
     1. rdstate函数，返回一个iostate值，对应当前流状态
     2. setstate(flag) 函数，将流对象设置为想要的状态
     3. clear函数：是一个重载的函数。
        1. clear()，将所有位置0，也就是goodbit状态。
        2. clear(flag)，将对应的条件状态标志位复位。
     4. ignore函数：

作用：提取输入字符并丢弃他们。

函数原型：istream& ignore (streamsize n = 1, int delim = EOF)

              读取到前n个字符或在读这n个字符进程中遇到delim字符就停止，把读 取的这些东西丢掉

#### 内存与输入输出设备的交互（iostream）

1. **getline：**

iostream比较常用的就是这个getline了，getline其实并不复杂，不过是按行接收数据罢了，**因为存储在string对象中，所以不容易出现格式错误，但仍然可能出现系统错误，所以在企业级程序中，还是应当对bad的情况进行处理。**

1. **get:**

还有个不怎么常用的get函数。get函数的用法和getline类似，只不过get是以字符的格式进行接收。**在企业级代码中仍然需要对bad的情况进行处理。**

#### 内存与磁盘的交互（fstream）

1. fstream相对于iostream。多了很多自己独有的操作
   1. i**o库默认没有给ifstream和ofstream类提供对象，需要我们自己去定义。**
   2. fstream对象创建方式有三种
      1. 可以使用默认构造函数进行定义。例如： ifstream fstrm，
      2. 也可以在创建流对象时打开想要打开的文件。例如ifstream fstrm(s)。s可以是字符串，也可以是c风格的字符串指针。文件的mode依赖于流对象的类型。
      3. 也可以在打开文件时就指定文件的mode。例如ifstream fstrm(s, mode)
   3. fstrm.open(s)函数，打开名为s的文件，并将文件与fsrm绑定，s可以是一个string，也可以是一个c风格的字符串指针。
   4. fstrm.close()函数，关闭文件。**注意，一定不要忘了。**
   5. fstrm.is\_open()函数，返回一个bool值，指出与fstrm关联的文件是否成功打开且尚未关闭。
2. 文件模式：
   1. in以读的方式打开
   2. out以写的方式打开
   3. app在进行写操作时定位到文件末尾
   4. ate打开文件后立即定位到文件末尾
   5. trunc截断文件（也就是文件已有的全部删除，重新开始写）
   6. binary以二进制方式打开文件
3. 文件模式需要强调以下几点
   1. 与ifstream关联的文件默认in模式打开。
   2. 与ofstream关联的文件默认out模式打开
   3. 与fstream关联的文件默认in和out模式打开
   4. 默认情况下，即使我们没有指定trunc，以out模式打开的文件也会被截断。为了保持以out模式打开的文件的内容，我们必须同时指定app模式或in模式。
   5. 只可以对ifstream或fstream的对象设定in的模式
   6. 只可以对ofstream或fstream的对象设定out的模式
   7. 只有当out模式被设置时才可以设置trunc模式
   8. ate和binary模式可以应用与任何类型的文件流对象，且可以与任何其它文件模式组合使用。
4. **总结：文件流这部分还是有一些东西的，理解起来可能有些困难，用的多了就好了。其实常用的就那么几点。**

#### 内存之中对于字符串的操作（sstream）

1. string流介绍：string流可以向string对象写入数据，也可以从string对象读取数据。与文件操作类似，只不过数据交互变成了从内存到内存。

string流默认包含一个string对象，当然，我们也可以指定。

1. string流有哪些
   1. istringstream从string对象读取数据
   2. ostringstream向string对象写数据
   3. stringstream既可以从string对象读取数据，也可以向string对象写数据
2. string流对象继承自iostream对象，除了继承得来的操作，string流对象还有自己的成员来管理流相关的string。
   1. 对于string流，io库是没有像cout，cin这样的自定流对象的。流对象需要我们自己去定义
      1. sstream strm：sstream代表一个string流对象的类型，以下同理。strm是一个未绑定的stringstream对象。
      2. Sstream strm(s)：strm是一个绑定了s的拷贝的string流对象。s是一个string对象
   2. strm.str()：返回strm所保存的string的拷贝。
   3. strm.str(s)：将s拷贝到strm中，返回void
3. string流对象的作用
   1. **对数据类型进行转化，也就是string和其它类型的转化，这是string流对象最重要的功能。**
      1. string转化为int等类型。
      2. int等类型转化为string。
   2. 用于对空格分隔的字符串的切分