1. 输入/输出

程序的输入指的是从输入文件将数据传送给程序，程序的输出指的是从程序将数据传送给输出文件。

1. C++的输入输出：

（一）标准I/O：

对系统指定的标准设备的输入和输出。即从键盘输入数据，输出到显示器屏幕。这种输入输出称为标准的输入输出

（二）文件I/O：

以外存磁盘文件为对象进行输入和输出，即从磁盘文件输入数据，数据输出到磁盘文件。以外存文件为对象的输入输出称为文件的输入输出

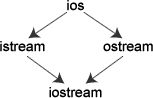
（三）串I/O：

对内存中指定的空间进行输入和输出。通常指定一个字符数组作为存储空间(实际上可以利用该空间存储任何信息)。这种输入输出称为字符串输入输出。

三． Iostream类库

（一）类库的继承关系

系统的iostream类库可以理解为i-stream和o-stream类库。且iostream类库是通过菱形继承得来的，如下图所示：





（二）与iostream类库有关的头文件

* iostream  包含了对输入输出流进行操作所需的基本信息。
* fstream  用于用户管理的文件的I/O操作。
* strstream  用于字符串流I/O。
* stdiostream  用于混合使用C和C + +的I/O机制时，例如想将C程序转变为C++程序。
* iomanip  在使用格式化I/O时应包含此头文件。

1. 标准I/O流：

（一）流对象

在iostream头文件中不仅定义了有关的类，还定义了4种流对象，

|  |  |  |
| --- | --- | --- |
| cin | 标准输入流 | 键盘 |
| cout | 标准输出流 | 屏幕 |
| cerr | 标准错误流 | 屏幕 |
| clog | 标准错误流 | 屏幕 |

如：

Cout的使用：

Ostream cout(stdout)

Cout为ostream流类对象，把标准输出设备stdout作为参数，这样cout就和标准输出设备联系起来。

（二）cout流对象

cout是console output的缩写，即在控制台输出，但是也可以被重定向输出到磁盘文件。Cout是ostream流类对象，并不是一个运算符，而是一个容纳数据的载体。

Cout流在内存中开辟了一个缓冲区，用来存放流中的数据。当向cout流中插入一个endl时，立即输出缓冲区中的数据并插入一个换行符，之后刷新缓冲区。

若只是插入一个”\n”，那么只输出和换行，并不刷新缓冲区。（并不是所有编译系统都是这样）。

（三）cerr流对象

cerr流对象是标准错误流，cerr流已被指定为与显示器关联。cerr的 作用是向标准错误设备(standard error device)输出有关出错信息。cerr流中的信息只能在显示器输出。

当调试程序时，我们希望程序运行时的错误信息输出到屏幕上。

（四）clog流对象

clog流对象也是标准错误流，它是console log的缩写。它的作用和cerr相同，都是在终端显示器上显示出错信息。区别：cerr是不经过缓冲区，直接向显示器上输出有关信息，而clog中的信息存放在缓冲区中，缓冲区满后或遇endl时向显示器输出。

五．标准输入流cin

标准输入流是一个对象，归属于一个类，下面使用的都是类中的成员函数。

（一）cin.get()

一次只从输入的缓冲区中读取一个字符，包括换行符，也是可以读取的。

（二）cin.get(两个参数)，可以读取字符串

char buf[1024];

cin.get(buf, 1024);

将输入的字符串读入buf数组中，但是换行符还是遗留在缓冲区中。

（三）cin.getline() ，读取字符串

读取了字符串，并且读取了换行符，但是它把换行符给扔掉了，既没有读到输出中，也没有遗留在缓冲区中。

（四）cin.ignore(N) 忽略

忽略掉N个字符。

（五）cin.peek() 偷窥

只从缓冲区中拿出一个字符，看一眼，然后又还了回去。

（六）cin.putback() 放回

将刚刚从缓冲区拿出的一个字符放回了缓冲区。

六．标准输出流cout:  
（一） Cout.put()

向缓冲区输出流中写字符

（二） cout.write(buf, num)

从buf中写num个字节到当前输出流中

（三）cout.flush()

刷新缓冲区

七．格式化输出

在输出数据时，通常会指定输出的格式，C++中有两种方式控制输出格式：

1. 使用控制符
2. 使用cout流对象的有关成员函数

（一）使用控制符的格式输出

程序开头需要添加 iomanip头文件和 iostream 头文件

Cout << setw(20) << setfill(“~”) << endl;

控制符的格式输出就是直接在 cout 之后添加控制符即可。

setiosflags中还需要添加格式状态，他是通过格式标志来指定的。

格式标志在类ios中被定义为枚举值。

格式标志如下所示：



控制符如下所示：



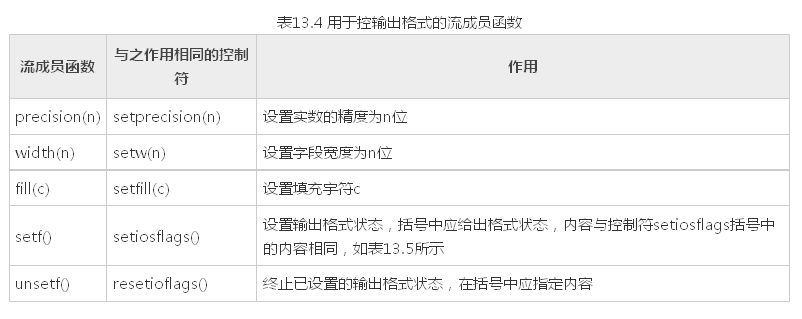
（二）使用cout流对象的成员函数

添加 iostream头文件

Cout.width(20);

Cout.fill(“~”);

下面是成员函数：



八．输入输出的注意事项：

1.成员函数width(n)和控制符setw(n)只对其后的第一个输出项有效。如：

cout. width(6);  
    cout <<20 <<3.14<<endl;  
 输出结果为 203.14

在输出第一个输出项20时，域宽为6，因此在20前面有4个空格，在输出3.14时，width (6)已不起作用，此时按系统默认的域宽输出（按数据实际长度输出）。如果要求在输出数据时都按指定的同一域宽n输出，不能只调用一次width(n)， 而必须在输出每一项前都调用一次width(n)

2.在格式标志中，同类型的只允许存在一种，如dec,hex,oct只能选其一。因此，在设置输出格式状态后，如果想改变同类型的一个格式标志，那么应当调用成员函数unsetf（对应控制符self）或 resetiosflags（对应于控制符setiosflags），先终止原来设置的状态。然后再设置其他状态，

3.可以包含两个或多个格式标志，由于这些格式标志在ios类中被定义为枚举值，每一个格式标志以一个二进位代表，因此可以用位或运算符“|”组合多个格式标志。

九．文件读写

（一）介绍

从磁盘文件读取数据，将数据输入到磁盘文件。和文件输入输出相关的头文件为fstream.h，在这个头文件中主要被定义了三个类，由这三个类控制对文件的各种输入输出操作，他们分别是ifstream、ofstream、fstream。

由于文件设备并不像显示器屏幕与键盘那样是标准默认设备，所以它在fstream头文件中是没有像cout那样预先定义的全局对象，所以我们必须自己定义一个该类的对象。ifstream类，它是从istream类派生的，用来支持从磁盘文件的输入。ofstream类，它是从ostream类派生的，用来支持向磁盘文件的输出。

（二）如何记忆文件输入输出类

如何记忆两种类的不同用法，可以通过类比cin,cout来记忆。如cin是将输入从控制台输入到cin流对象中，那么ifstream也是从文件输入到ifstream流对象中。

Cout是从cout流对象输出到控制台中，那么ofstream也是从ofstream流对象输出到文件中。

（三）打开文件

打开文件是指在文件读写之前做必要的准备工作，包括：

1）为文件流对象和指定的磁盘文件建立关联，以便使文件流流向指定的磁盘文件。

2）指定文件的工作方式，如：该文件是作为输入文件还是输出文件，是ASCII文件还是二进制文件等。

有两种方式为文件指定工作方式：

1. 调用文件流的成员函数open

Ofstream outfille; //定义ofstream类对象

Outfile.open(“1.txt”, ios::out); //是ofstream输出流与1.txt文件建立联系

Ios::out是I/O模式的一种，表示以输出方式打开一个文件。

2. 在定义文件流对象时指定参数

在声明文件流类时定义了带参数的构造函数，其中包含了打开磁盘文件的功能。因此， 可以在定义文件流对象时指定参数，调用文件流类的构造函数来实现打开文件的功能。

文件的I/O模式如下图所示：  


1. 每一个打开的文件都有一个文件指针，该指针的初始位置由I/O方式指定，每次读写都从文件指针的当前位置开始。每读入一个字节，指针就后移一个字节。当文件指针移到最后，就会遇到文件结束EOF（文件结束符也占一个字节，其值为-1)，此时流对象的成员函数eof的值为非0值(一般设为1)，表示文件结束 了。
2. 可以用“位或”运算符“|”对输入输出方式进行组合

如：

  ios::app | ios::nocreate

1. 如果打开操作失败，open函数的返回值为0(假)，如果是用调用构造函数的方式打开文件的，则流对象的值为0。可以据此测试打开是否成功。

  if(outfile.open("1.txt", ios::app) ==0)  
        cout <<"open error";  
或  
    if( !outfile.open("1.txt", ios::app) )  
        cout <<"open error";

（四）关闭文件

在对已打开的磁盘文件的读写操作完成后，应关闭该文件。关闭文件用成员函数close。

所谓关闭，实际上是解除该磁盘文件与文件流的关联，原来设置的工作方式也失效，这样，就不能再通过文件流对该文件进行输入或输出。此时可以将文件流与其他磁盘文件建立关联，通过文件流对新的文件进行输入或输出。

（五）对ASCII文件的读写操作

如果文件的每一个字节中均以ASCII代码形式存放数据,即一个字节存放一个字符,这个文件就是ASCII文件(或称字符文件)。程序可以从ASCII文件中读入若干个字符,也可以向它输出一些字符。

（六）对二进制文件的读写操作

二进制文件不是以ASCII代码存放数据的，它将内存中数据存储形式不加转换地传送到磁盘文件，因此它又称为内存数据的映像文件。因为文件中的信息不是字符数据，而是字节中的二进制形式的信息，因此它又称为字节文件。

在打开时要用ios::binary指定为以二进制形式传送和存储。二进制文件除了可以作为输入文件或输出文件外,还可以是既能输入又能输出的文件。这是和ASCII文件不同的地方。

对二进制文件的读写主要用istream类的成员函数read和write来实现。这两个成员函数的原型为  
    istream& read(char \*buffer,int len);  
    ostream& write(const char \* buffer,int len);  
字符指针buffer指向内存中一段存储空间。len是读写的字节数。调用的方式为：  
    a. write(p1,50);  
    b. read(p2,30);  
 上面第一行中的a是输出文件流对象，write函数将字符指针p1所给出的地址开始的50个字节的内容不加转换地写到磁盘文件中。在第二行中，b是输入文件流对象，read 函数从b所关联的磁盘文件中，读入30个字节(或遇EOF结束），存放在字符指针p2所指的一段空间内。