

# 第九章 输入/输出流



# 学习目标

---

- 理解如何使用输入/输出流。
- 掌握格式化输入/输出。
- 掌握如何输入/输出用户自定义类型数据。
- 掌握文件的顺序/随机处理方式。





# C++流类库简介

---

- **cin**, 表示标准输入的**istream**类对象。**cin**从终端读入数据。
- **cout**, 表示标准输出的**ostream**类对象。**cout**向终端写数据。
- **cerr**, 表示标准错误输出(非缓冲方式), 导出程序错误消息。
- **clog**, 表示标准错误输出(缓冲方式), 导出程序错误消息。
- 为了在程序中使用**cin/cout**, 必须在程序中包含**iostream**库的相关头文件

- **#include<iostream>**



# C++流类库简介

- ❁ **iostream**类同时从**istream**(输入流)类和**ostream**(输出流)类派生而来，允许双向输入/输出。输入由重载的操作符**>>**完成，输出由重载的操作符**<<**来完成。

```
cin>>变量名;  
cout<<变量名或常量;
```

- ❁ **[例9-1]** 简单**I/O**举例。



# C++流类库简介

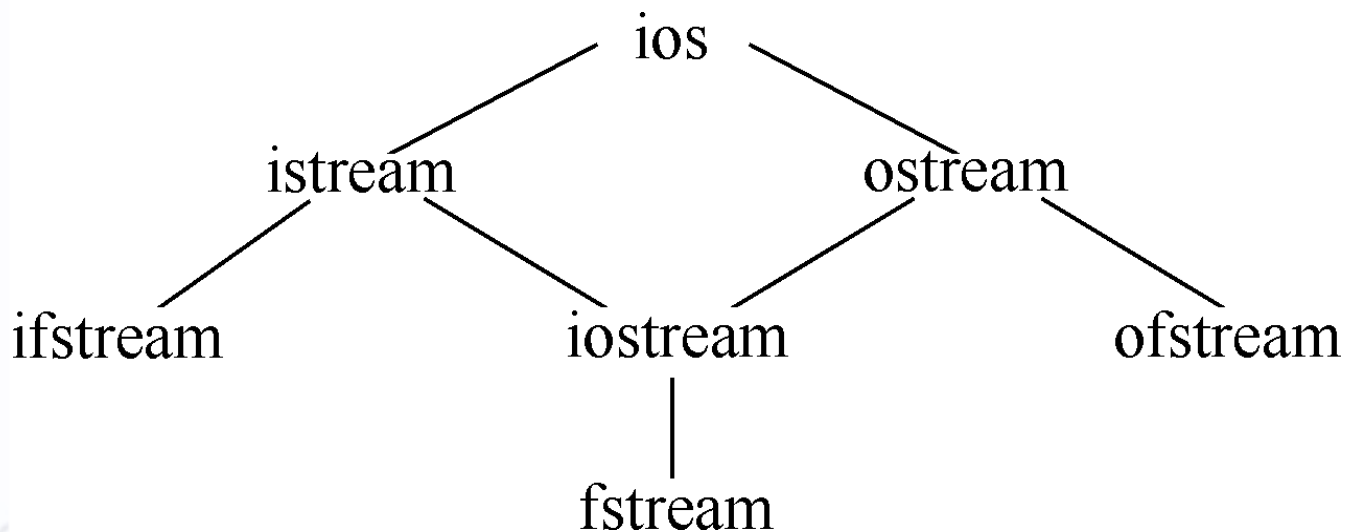
除了用户终端的输入/输出，C++还提供了文件的输入/输出类：

- **ifstream**，从**istream**派生而来，把文件绑定到程序上输入；
- **ofstream**，从**ostream**派生而来，把文件绑定到程序输出；
- **fstream**，从**iostream**派生而来，把文件绑定到程序输入/输出。
- 使用**iostream**库的文件流，必须包含相关的头文件：
  - `#include <fstream>`

 [例9-2] 文件读写举例。

# C++流类库简介

几个常用的流类库层次结构如图



# 输入/输出流

---

- ❁ 流输出可以用流插入运算符——即重载的 $\ll$ (左移位运算符)来完成。运算符 $\ll$ 左边的操作数是`istream`类的一个对象(如`cout`)，右边可以是C++的合法表达式。
- ❁ [例9-3] 用插入运算符实现流输出。
- ❁ [例9-4] 函数调用作输出操作符的实参。
- ❁ [例9-5] 简单输出实例1。
- ❁ [例9-6] 简单输出实例2。





# 基本输入流

❁ 流输入可以用流读取运算符——即重载的>>(右移位运算符)来完成。

❁ 注意:

- 输入运算符>>也支持级联输入。在默认情况下,运算符>>跳过空格,读入后面与变量类型相应的值。因此给一组变量输入值时,用空格或换行将输入的数值间隔开,例如:

```
int i;
```

```
float f;
```

```
cin>>i>>f;
```

```
当从键盘输入 10 12.34
```





# 基本输入流

- 当输入字符串(**char\*** 类型)时, 输入运算符**>>**会跳过空格, 读入后面的非空格符, 直到遇到另外一个空格结束, 并在字符串末尾自动放置字符 '**\0**'作为结束标志, 例如:
  - `char s[20]; cin>>s;` 当输入 `Hello! world!`
- 数据输入时, 不仅检查数据间的空格, 还做类型检查、自动匹配, 例如:
  - `int i; float f; cin>>i>>f;`
  - 如果输入 `12.34 34.56`



# 格式化输入/输出

## ios类成员函数控制格式

- 设置状态标志
  - 设置状态标志可使用 `long ios::setf(long flags)` 函数，使用格式如下：
  - `stream_obj.setf(ios::enum_type);`
    - 例如：`cout.setf(ios::dec);`
- 清除状态标志
  - 清除状态标志可使用 `long ios::unsetf(long flags)` 函数。使用格式如下：
  - `stream_obj.unsetf(ios::enum_type);`
- 取状态标志
  - 取状态标志用 `flags( )` 函数，在 `ios` 类中重载了两个版本：
  - `long ios::flags();`
  - `long ios::flags(long flag);`



# 格式化输入/输出

- 设置域宽

- 域宽用于控制输出格式，在**ios**类中重载了两个函数来控制域宽，原型如下：

- `int ios::width();`

- `int ios::width(int w);`

- 设置填充字符

- 填充字符的作用是，当输出值不满域宽时用设定的字符来填充，默认情况下填充的字符为空格。实际应用中填充字符函数与设置域宽函数**width( )**配合使用，否则无空可填，毫无意义。**ios**类提供了两个重载的成员函数来操作填充字符，原型如下：

- `char ios::fill();`

- `char ios::fill(char c);`

以上内容仅为本文档的试下载部分，为可阅读页数的一半内容。如要下载或阅读全文，请访问：<https://d.book118.com/287013165032006042>