C++程序设计第十三节课官方笔记

目录

一、课件下载及重播方法

二、教材节构图

三、本章知识点及考频总结

（一）选择题（共 10 道）

1. 同 C 语言一样，C++中也没有输入/输出语句，但在 C++的标准库中有一个面向对象的输入/输出软件包，即 I/O 流类库，输入和输出均是通过**流**完成的。流是 I/O 流类的核心概念。

C++的输出操作是将一个对象转换成一个字符序列，输出到指定对象。输入操作是从某

个对象接收到一个字符序列，然后将其转换为相应对象所要求的格式。数据输入和输出的过程就是数据传输的过程，数据像“水”一样从一个地方流动到另一个地方，因此，在C++中将此过程称为“流”。

从更一般的意义上说，C++中凡是数据从一个地方传输到另一个地方的操作都是流的操作。因此，一般意义下的**读操作**在流数据抽象中被称为（从流中）“**提取**”，**写操作**被称为（向流中）“**插入**”。

在C++的标准类库中，将与数据输入/输出相关的类统称为“流类”。C++中常用的几个流类及其关系如图7-1所示。

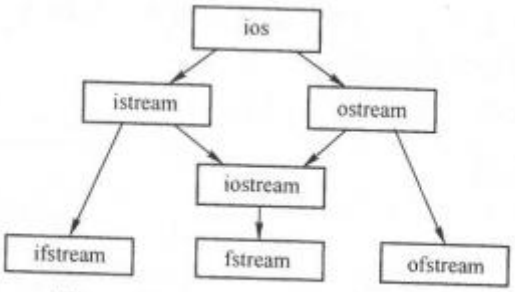

图7-1 iostream流类库的类关系图

图7-1中的箭头代表派生关系。ios是抽象基类，提供输入/输出所需的公共操作，它派生出两个类istream和ostream。为了避免多重继承的二义性，从ios派生istream和ostream时，均使用了virtual关键字（虚继承）。

istream类提供了流的大部分输入操作，对系统预定义的所有输入流重载提取运算符“>>”。ostream类对系统预定义的所有输出流重载插入运算符“<<”。

2. C++的iostream类库提供了数百种I/O功能，iostream类库的接口部分包含在几个头文件中。常见的头文件有以下3个。

(1)**iostream**

头文件iostream包含操作所有输入/输出流所需的基本信息，因此大多数C++程序都应包含这个头文件。该文件含有4个标准流对象，提供了无格式化和格式化的I/O功能。

(2)**iomanip**

头文件iomanip包含格式化I/O的带参数流操纵符，可用于指定数据输入/输出的格式。

(3)**fstream**

头文件fstream包含处理文件的有关信息，提供建立文件、读/写文件的各种操作接口。

3. C++在头文件iostream中为用户预定义了4个标准流对象，分别是：

- cin(标准输入流)
- cout(标准输出流)
- cerr(非缓冲错误输出流)
- clog(缓冲错误输出流)

cin与标准输入设备（键盘）相关联，用于读取数据，可以被重定向为从文件中读取数据。cout与标准输出设备（显示器）相关联，用于输出数据，可以被重定向为向文件里写入数据。cerr与标准错误信息输出设备（显示器）相关联（非缓冲），用于输出出错信息，不能被重定向。clog与标准错误信息输出设备相关联（缓冲），用于输出出错信息，不能被重定向。

在实际中，cin常用于从键盘输入数据，是流类istream的对象。cout常用于向屏幕输出数据，是流类ostream的对象。

4. cerr和clog的区别在于：cerr不使用缓冲区，直接向显示器输出信息；而输出到clog

中的信息会先被存储到缓冲区中，缓冲区满或者刷新时才输出到屏幕。

cout 可以使用重定向函数 freopen 进行重定向，而 cerr 不能。

5. 重定向函数 freopen 的原型如下：

```
FILE *freopen( const char *path, const char *mode, FILE *stream);
```

函数 freopen()的功能是将 stream 按 mode 指定的模式重定向到路径 path 指向的文件。

```
freopen("test.txt","w",stdout);  //将标准输出重定向到文件 test.txt
```

6. iostream 中常用流操纵符

流操纵符	作用	输入/输出
endl	输出一个新行符，并清空流	0
ends	输出字符串结束，并清空流	0
flush	清空流缓冲区	0
dec *	以十进制形式输入或输出整数	I/0
hex	以十六进制形式输入或输出整数	I/0
oct	以八进制形式输入或输出整数	I/0
ws	提取空白字符	0

注：“流操纵符”栏中的星号“*”不是操纵符的一部分，表示是默认设置。

7.在头文件 iomanip 中还定义了一些用于格式控制的流操纵符，见表 7-5。使用这些操纵符时必须包含头文件 iomanip。

常用的用于格式控制的流操纵符

流操纵符	作用
fixed	以普通小数形式输出浮点数
scientific	以科学计数法形式输出浮点数
left	左对齐，即在宽度不足时将填充字符添加到右边
right *	右对齐，即在宽度不足时将填充字符添加到左边
setbase(int b)	设置输出整数时的进制，b 为 8、10 或 16
setw(int w)	指定输出宽度为 w 个字符，或输入字符串时读入 w 个字符。一次有效
setfill(int c)	在指定输出宽度的情况下，输出的宽度不足时用 ASCII 码为 c 的字符填充（默认情况是用空格填充）
setprecision(int n)	设置输出浮点数的精度为 n。在使用非 fixed 且非 scientific 方式输出的情况下，n 即为有效数字最多的位数。如果有效数字位数超过 n，则小数部分四舍五入，或自动变为科学计数法输出并保留一共 n 位有效数字；在使用 fixed 方式和 scientific 方式输出的情况下，n 是小数点后面应保留的位数
setiosflags(fmtfalgs f)	通用操纵符。将格式标志 f 所对应的格式标志位置为 1
resetiosflags(fmtfalgs f)	通用操纵符。将格式标志 f 所对应的格式标志位置为 0(清除)
boolapha	把 true 和 false 输出为字符串
noboolalpha *	把 true 和 false 分别输出为 1 和 0
showbase	输出表示数值进制的前缀
noshowbase *	不输出表示数值进制的前缀
showpoint	总是输出小数点
noshowpoint *	只有当小数部分存在时才显示小数点
showpos	在非负数值中显示+

noshowpos *	在非负数值中不显示+
skipws *	输入时跳过空白字符
noskipws	输入时不跳过空白字符
uppercase	十六进制数中使用’A’~’E’。若输出前缀，则前缀输出“0x”，科学计数法中输出’E’
no uppercase *	十六进制数中使用’a’~’e’。若输出前缀，则前缀输出“0x”，科学计数法中输出’e’
internal	数值的符号（正负号）在指定宽度内左对齐，数值右对齐，中间由填充字符填充

8. 标志字

为满足不同用户对数据输入/输出格式的要求，C++提供了通过 setiosflags()设置标志字进行格式控制的方式。setiosflags()是带参数的操纵符，在头文件 iostream 中，用以设置指定的标志，函数的参数为流的格式标志位。

标志字是一个 long 型的数据，由若干个系统定义的格式控制标志位“组合”而成。常见的格式标志和含义见表 7-6。

表 7-6 常见格式标志常量及含义

标志常量名	值	含义	输入/输出
ios::skipws	0X0001	跳过输入中的空白	I
ios::left	0X0002	按输出域左对齐，用填充字符填充右边	0
ios::right *	0X0004	按输出域右对齐，用填充字符填充左边	0
ios::internal	0X0008	在符号位或基数指示符后填入字符	0
ios::dec *	0X0010	转换为十进制基数形式	I/0
ios::oct	0X0020	转换为八进制基数形式	I/0
ios::hex	0X0040	转换为十六进制基数形式	I/0
ios::showbase	0X0080	在输出中显示基数指示符	0
ios::showpoint	0X0100	在输出浮点数时必须带小数点和尾部的 0	0
ios::uppercase	0X0200	以大写字母表示十六进制数，科学计数法使用大写字母 E	0
ios::showpos	0X0400	正数前加“+”号	0
ios::scientific	0X0800	科学记数法显示浮点数	0
ios::fixed	0X1000	定点形式表示浮点数	0
ios::unitbuf	0X2000	插入操作后立即刷新流	0

例如，程序中可以采用以下方式设置标准输出流为八进制基数形式插入数据项：

```
cout setf(ios::oct, ios::basefield);
```

其中 ios::basefield 表不清除已经设置的各基数的格式位。

9.在 cout 中调用成员函数控制输出格式，其作用和对应的流操纵符相同。

ostream 类的成员函数及与其作用相同的流操纵符

成员函数	作用相同的流操纵符
precision(int np)	setprecision(np)
width(int nw)	setw(nw)
fill(char cFill)	setfill(cFill)
setf(long IFlags)	setiosflags(IFlags)
unsetf(long IFlags)	resetiosflags(IFlags)

10.调用 cin 的成员函数

```
int get();
```

此函数从输入流中读入一个字符（包括空白字符），返回值就是该字符的 ASCII 码。如果碰到输入结束符，则返回值为系统常量 EOF(End Of File，文件结束标记)。

getline()成员函数的原型如下：

```
istream & getline(char * buf, int bufSize);
```

其功能是从输入流中的当前字符开始读取 bufSize-1 个字符到缓冲区 buf，或读到’\n’为止（哪个条件先满足即按哪个执行）。函数会在 buf 中读入数据的结尾自动添加串结束标记’\0’。

```
istream & getline(char * buf, int bufSize, char delim);
```

其功能是从输入流中的当前字符开始读取 bufSize-1 个字符到缓冲区 buf，或读到字符 delim 为止(哪个条件先满足即按哪个执行)。函数会在 buf 中读入数据的结尾自动添加’\0’。

eof()成员函数的原型如下：

```
bool eof();
```

eof()函数用于判断输入流是否已经结束。返回值为 true 表示输入结束。

```
istream & ignore(int n=1, int delim=EOF);
```

此函数的作用是跳过输入流中的 n 个字符，或跳过 delim 及其之前的所有字符（哪个条件先满足就按哪个执行）。两个参数都有默认值。因此 cin.ignore()等效于 cin.ignore(1,EOF)，即跳过一个字符。该函数常用于跳过输入中的无用部分，以便提取有用的部分。

```
int peek();
```

函数 peek()返回输入流中的当前字符，但是并不将该字符从输入流中取走——相当于只是“看了一眼”将要读入的下一个字符，因此叫“窥视”。cin.peek()不会跳过输入流中的空格和回车符。在输入流已经结束的情况下，cin.peek()返回 EOF。

（二）主观题（共 0 道）

四、配套练习题

1. 下列选项中，不能作为输出流的对象是（ ）

A:文件

B:内存

C:键盘

D:显示器

2. 下列选项中，不是 C++中的标准输入输出的是（ ）

A:stdin

B:cout

C:clog

D:cerr

3. 下列选项中，用于清除基数格式位设置以十六进制数输出的语句是（ ）

A:cout<<setf(ios::dec,ios::basefield);

B:cout<<setf(ios::hex,ios::basefield);

C:cout<<setf(ios::oct,ios::basefield);

D:cin>>setf(ios::hex,ios::basefield);

[参考答案] CAB

五、其余课程安排