Chapter 17

File Processing

OBJECTIVES

- To create, read, write and update files.
- Sequential file processing.
- Random-access file processing.
- To use high-performance unformatted I/O operations.
- The differences between formatted-data and raw-data file processing.
- To build a transaction-processing
- program using random-access file processing.

Topics

- □ **17.1 Introduction**
- □ 17.2 The Data Hierarchy
- □ 17.3 Files and Streams
- □ 17.4 Creating a Sequential File
- □ 17.5 Reading Data from a Sequential File

17.1 Introduction

- ❑ 程序中的变量、数组、常量等— 临时存储

- ❑ data persistence: File(文件)
- ❑ • permanent retention of large amounts of data
- ❑ • secondary storage devices

- ❑ sequential files and random-access files

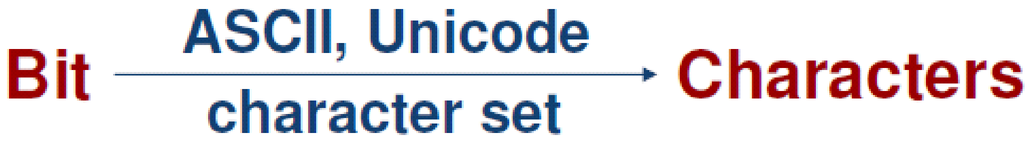
Topics

- □ 17.1 Introduction
- □ **17.2 The Data Hierarchy**
- □ 17.3 Files and Streams
- □ 17.4 Creating a Sequential File
- □ 17.5 Reading Data from a Sequential File

17.2 The Data Hierarchy

- □ Bit (binary digit) 二进制位: a digit that can assume one of two values

- □ Characters 字符: decimal, letters and special symbols (i.e., $, @ and many others). C++ provides data type char (occupies one byte of memory) and wchar_t (occupy more than one byte)



Bit

ASCII, Unicode

character set

Characters

17.2 The Data Hierarchy

- □ **Fields 字段: a group of characters that conveys some meaning (called data members in C++)**

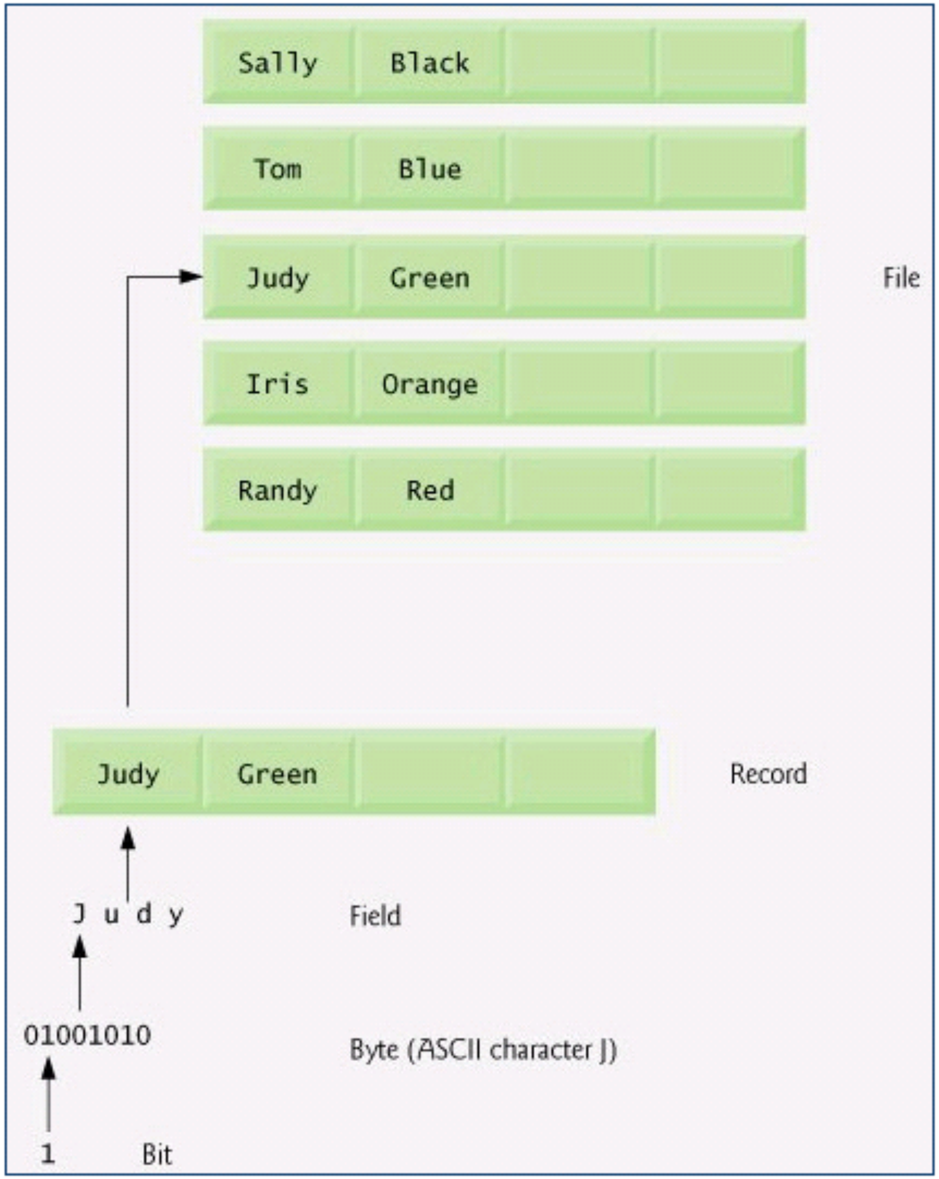
- □ **Record 记录: composed of several related fields (represented as a class in C++); A record key (键, 关键字) is a field unique to each record**

17.2 The Data Hierarchy

- □ **Sequential file 顺序文件: records typically are stored in order by a record-key field**

- □ **Database 数据库: a group of related files. A collection of programs designed to create and manage databases is called a database management system (DBMS)**

17.2 The Data Hierarchy



Topics

- □ 17.1 Introduction
- □ 17.2 The Data Hierarchy
- □ **17.3 Files and Streams**
- □ 17.4 Creating a Sequential File
- □ 17.5 Reading Data from a Sequential File

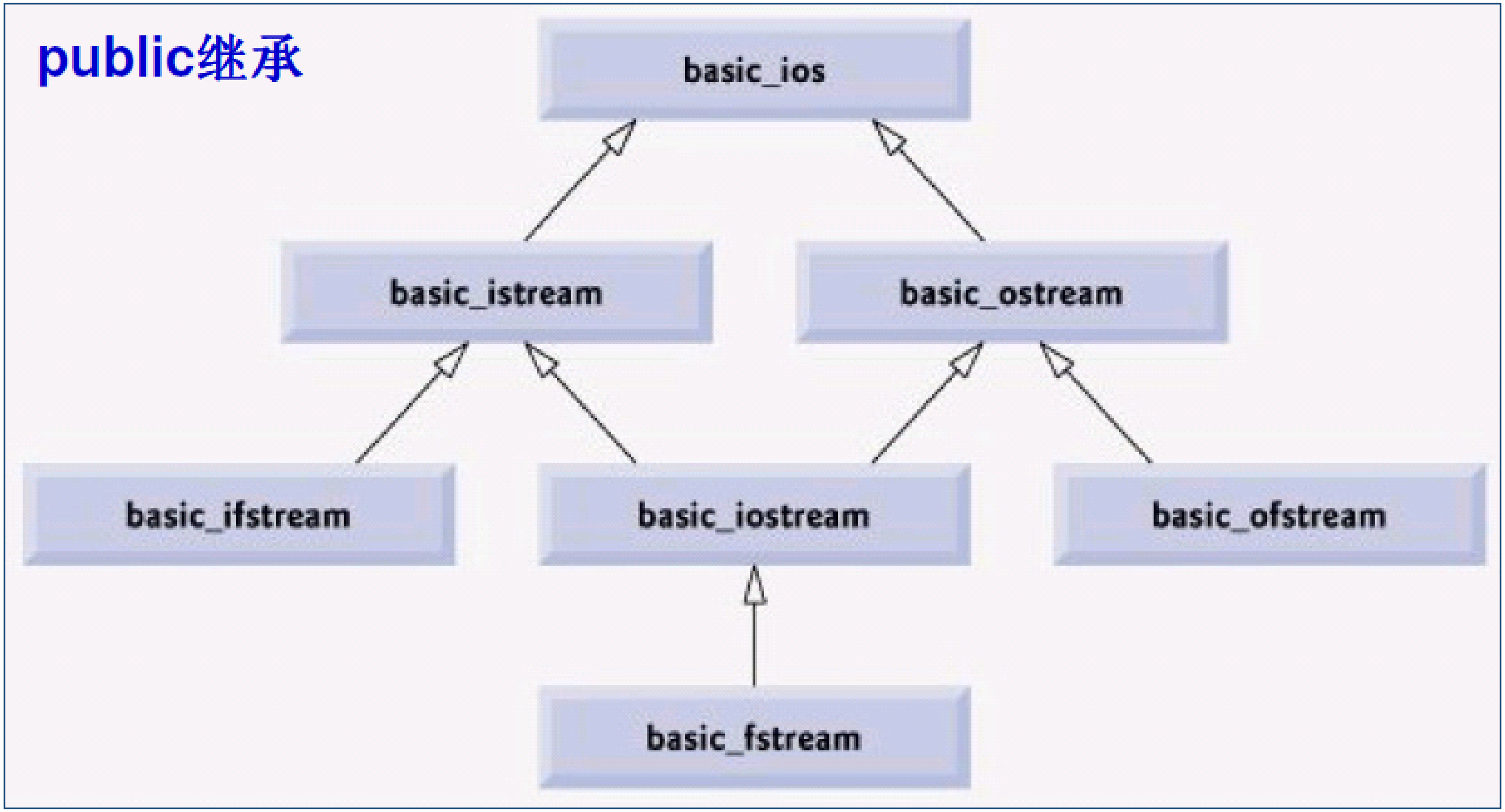
17.3 Files and Streams

- C++ views each file as a sequence of bytes and imposes no structure on a file. Each file ends either with an end-of-file marker or at a specific byte number recorded in a system maintained, administrative data structure. (文件结束标志或指定数量字节数)

- 所谓stream流就是一个字节序列: 当进行输入操作时, 字节从设备(键盘、磁盘等)流向内存; 当进行输出操作时, 字节从内存流向外部设备(键盘、磁盘等).

17.3 Files and Streams

❑ Stream I/O template hierarchy



public继承

basic_ios

basic_istream

basic_ostream

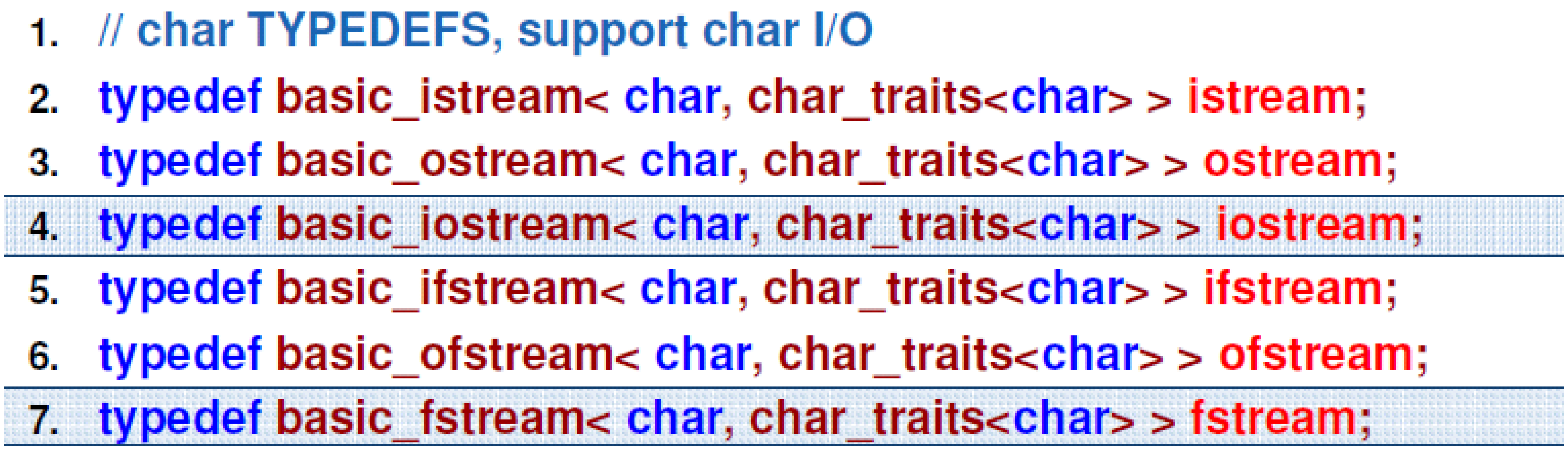
basic_ifstream

basic_iostream

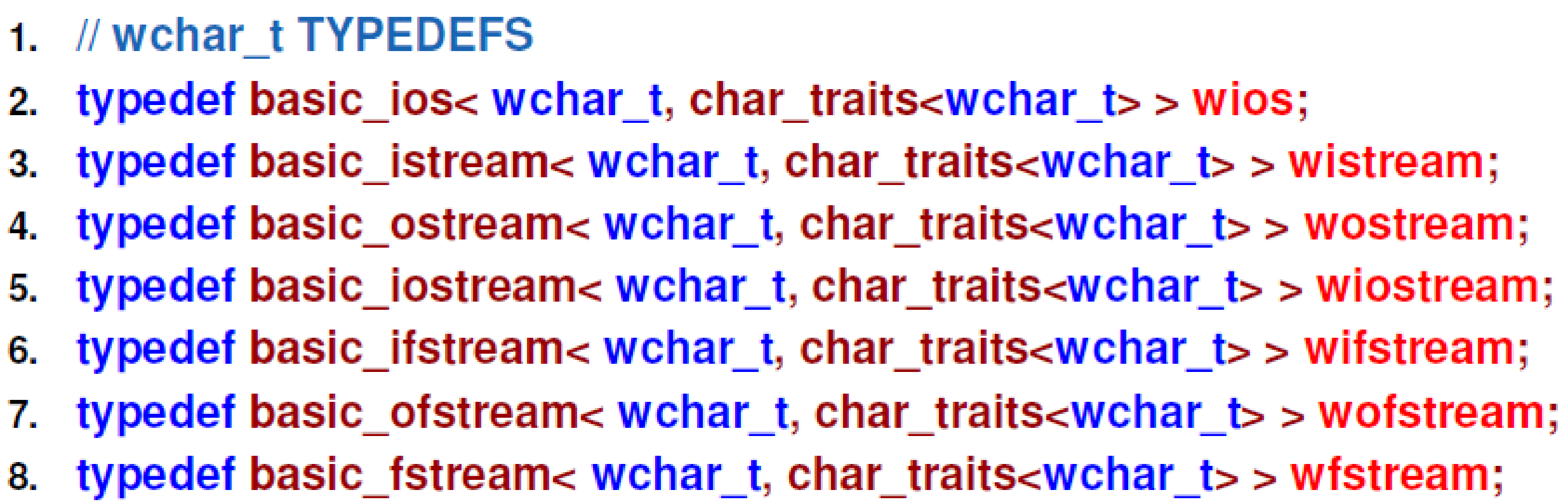
basic_ofstream

basic_fstream

17.3 Files and Streams

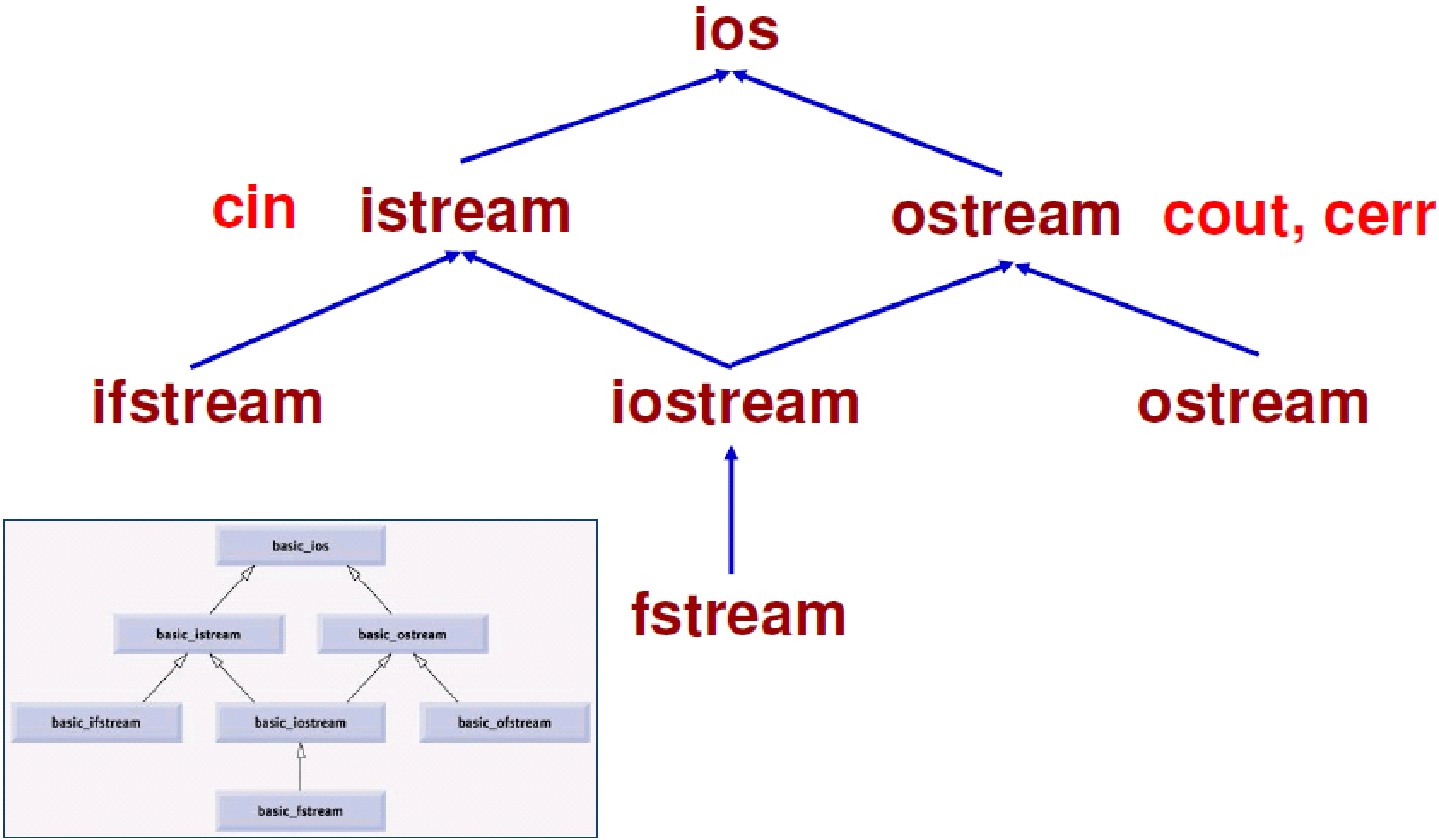


```
// char TYPEDEFS, support char I/O
typedef basic_istream< char, char_traits<char> > istream;
typedef basic_ostream< char, char_traits<char> > ostream;
typedef basic_iostream< char, char_traits<char> > iostream;
typedef basic_ifstream< char, char_traits<char> > ifstream;
typedef basic_ofstream< char, char_traits<char> > ofstream;
typedef basic_fstream< char, char_traits<char> > fstream;
```



```
// wchar_t TYPEDEFS
typedef basic_ios< wchar_t, char_traits<wchar_t> > wios;
typedef basic_istream< wchar_t, char_traits<wchar_t> > wistream;
typedef basic_ostream< wchar_t, char_traits<wchar_t> > wostream;
typedef basic_iostream< wchar_t, char_traits<wchar_t> > wiostream;
typedef basic_ifstream< wchar_t, char_traits<wchar_t> > wifstream;
typedef basic_ofstream< wchar_t, char_traits<wchar_t> > wofstream;
typedef basic_fstream< wchar_t, char_traits<wchar_t> > wfstream;
```

17.3 Files and Streams



ios

cin istream

ostream cout, cerr

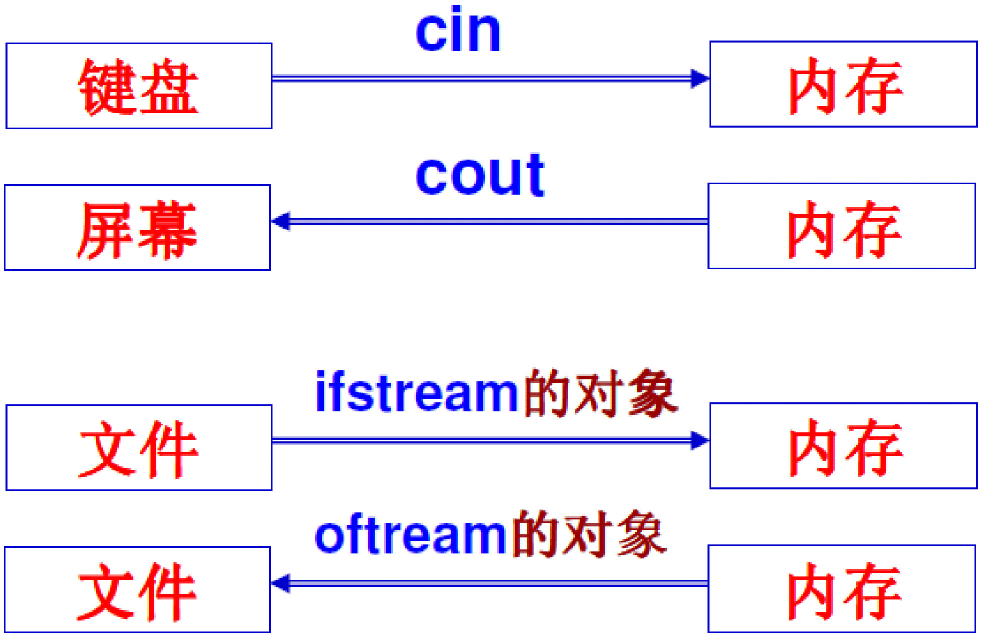
ifstream

iostream

ostream

fstream

17.3 Files and Streams



cin

键盘

内存

cout

屏幕

内存

ifstream的对象

文件

内存

oftream的对象

文件

内存

- 主要差异: 文件操作时需定义**ifstream / ofstream**对象, 以指定所具体操作的文件和操作相关的参数

17.3 Files and Streams

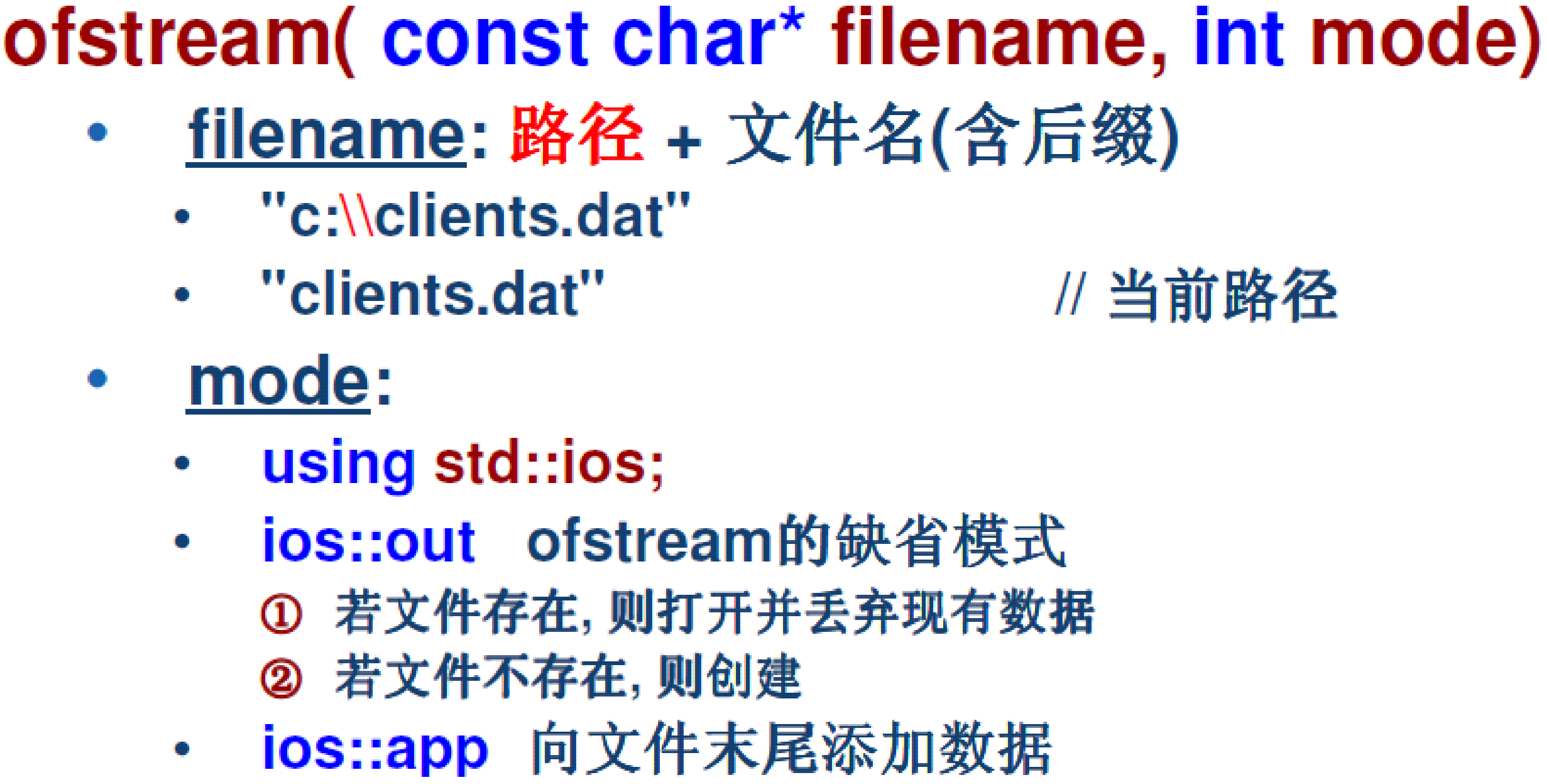
- 头文件
  - #include <iostream>
  - #include <fstream>
- <fstream>
- 包括三种类模板的定义
  - basic_ifstream (for file input)
  - basic_ofstream (for file output)
  - basic_fstream (for file input and output)
- 提供了处理char字符流的类模板特化定义
  - ifstream: 从文件中输入字符(读文件)
  - ofstream: 向文件输出字符(写文件)
  - fstream: 支持文件中字符的输入和输出

Topics

- □ 17.1 Introduction
- □ 17.2 The Data Hierarchy
- □ 17.3 Files and Streams
- □ **17.4 Creating a Sequential File**
- □ 17.5 Reading Data from a Sequential File

17.4 Creating a Sequential File

- ❑ 创建ofstream对象
- ❑ (1) 创建流类对象的同时打开文件



ofstream(const char* filename, int mode)

- filename: 路径 + 文件名(含后缀)
  - "c:\\clients.dat"
  - "clients.dat"   // 当前路径
- mode:
  - using std::ios;
  - ios::out   ofstream的缺省模式

    ① 若文件存在，则打开并丢弃现有数据

    ② 若文件不存在，则创建
  - ios::app   向文件末尾添加数据

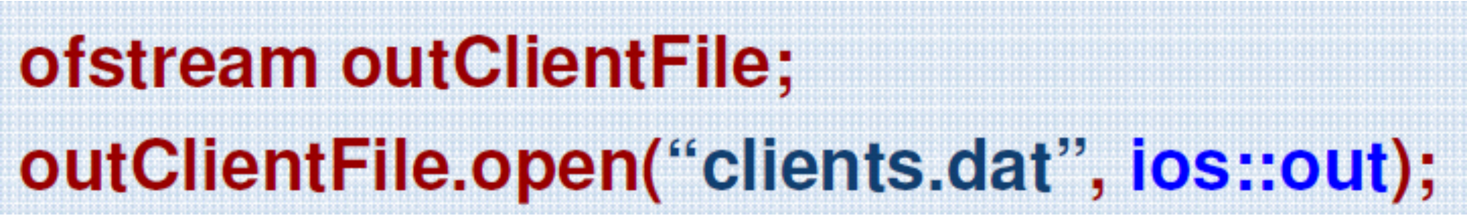
17.4 Creating a Sequential File

□ 创建ofstream对象

□ (2) 先创建对象, 后打开文件

□ • 缺省构造函数+ open成员函数

□ • open与前述构造函数的参数相同

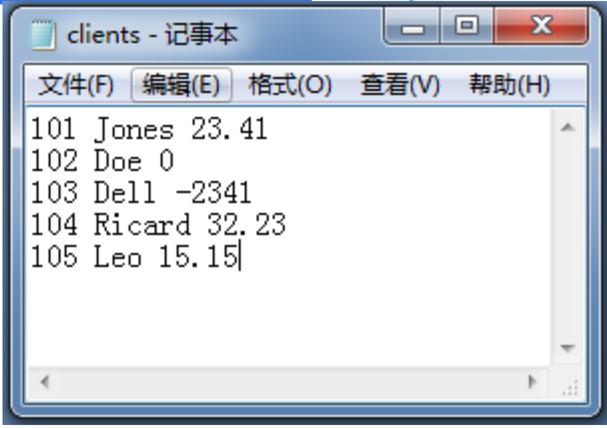


```
ofstream outClientFile;
outClientFile.open("clients.dat", ios::out);
```

17.4 Creating a Sequential File

- □ 文件的写操作(与cout相似)

```
outClientFile << account << ‘ ‘
                         << name << ‘ ‘
                         << balance << endl;
```



```
101 Jones 23.41
102 Doe 0
103 Dell -2341
104 Ricard 32.23
105 Leo 15.15
```

- □ 为何写数据的时候需要空格分割?

程序解读（P510）

17.4 Creating a Sequential File

- P510.17.3.29
- 当流引用作为condition使用, 会自动隐含调用void* 重载运算符, 以将其转换成指针
- 根据上一次流操作是否成功, 得到:
  - • null指针: 操作失败, 则0, 即False
  - • Non-null指针: 操作成功, 则非0, 即True
- 一种常见的流读取失败是读到了EOF标记,此时condition即为False

17.4 Creating a Sequential File

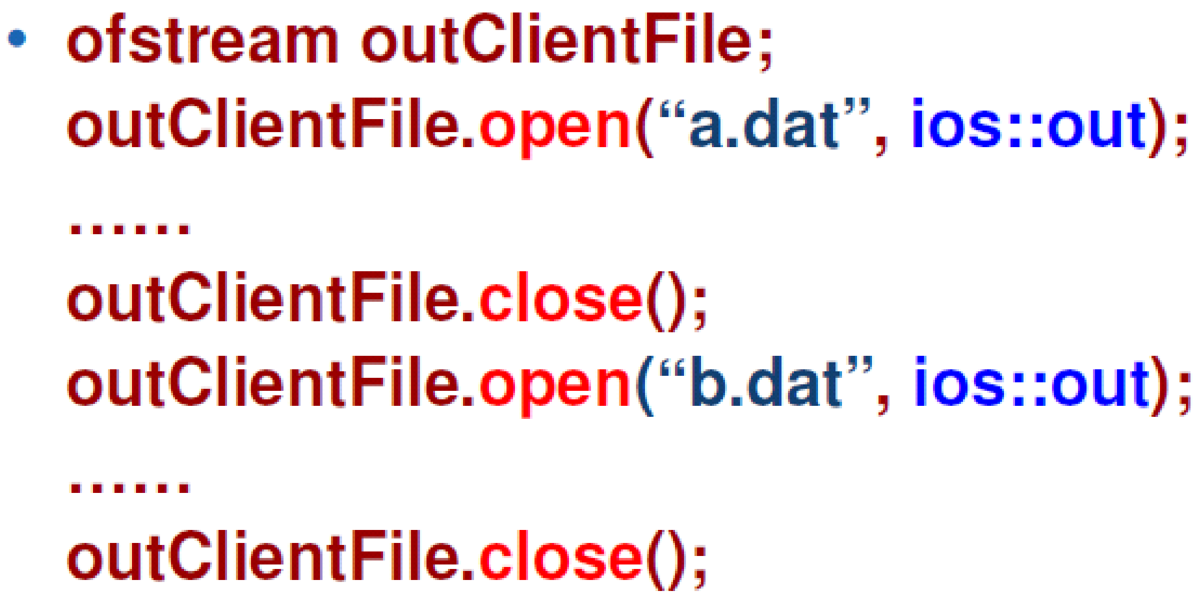
- If that reference is used as a condition (e.g., in a while statement's loop continuation condition), the stream's overloaded void * cast operator function is implicitly invoked to convert the reference into a non-null pointer value or the null pointer based on the success or failure of the last input operation. A non-null pointer converts to the bool value True to indicate success and the null pointer converts to the bool value false to indicate failure. When an attempt is made to read past the end of a stream, the stream's overloaded void * cast operator returns the null pointer to indicate end-of-file.

17.4 Creating a Sequential File

❑文件的关闭

❑• ofstream析构时会自动关闭文件

❑• 建议当文件不再需要使用时, 显式调用close成员函数关闭



- ofstream outClientFile;

```
ofstream outClientFile;
outClientFile.open(“a.dat”, ios::out);
......
outClientFile.close();
outClientFile.open(“b.dat”, ios::out);
......
outClientFile.close();
```

Topics

- ❑ 17.1 Introduction
- ❑ 17.2 The Data Hierarchy
- ❑ 17.3 Files and Streams
- ❑ 17.4 Creating a Sequential File
- ❑ **17.5 Reading Data from a Sequential File**

17.5 Reading Data from a Sequential File

- □ 创建ifstream 对象
- □ (1) 创建流类对象的同时打开文件

```
ifstream inClientFile( "clients.dat", ios::in );
```

- □ • ios::in – 缺省模式, 仅能从文件读取数据(最小权限原则)

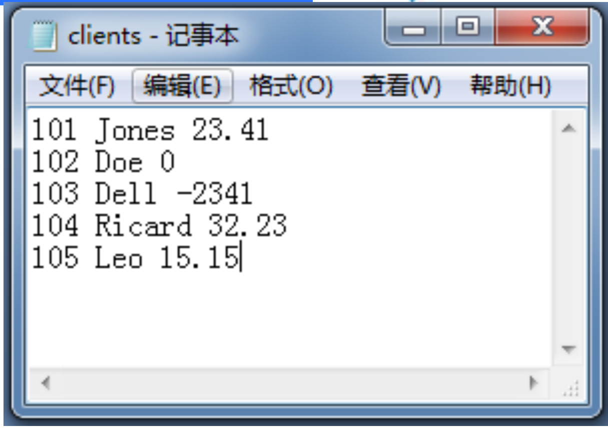
- □ (2) 创建对象, 后打开文件

```
ifstream inClientFile;
inClientFile.open("clients.dat", ios::in );
```

17.5 Reading Data from a Sequential File

- ❑ 文件的读操作(与cin相似)

```
inClientFile >> account
             >> name
             >> balance;
```



```
101 Jones 23.41
102 Doe 0
103 Dell -2341
104 Ricard 32.23
105 Leo 15.15
```

- ❑ 为何写文件的时候需要空格分割?

  读文件时, 需要空白符分割数据!

程序解读（P512）

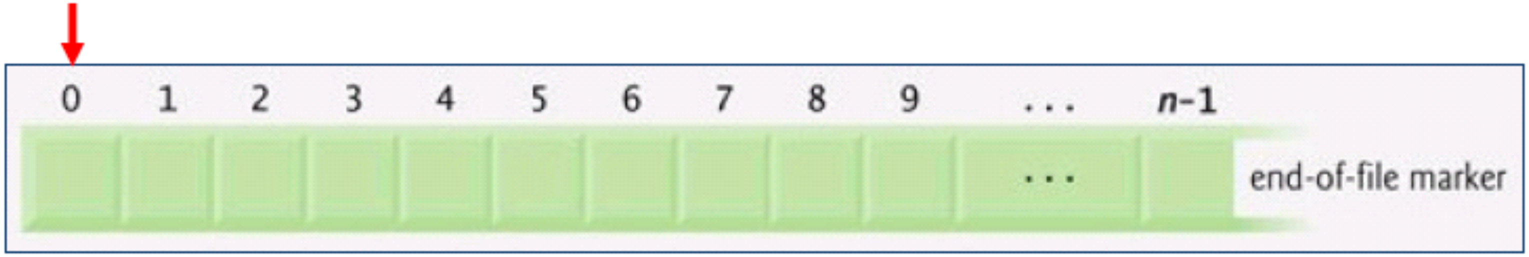
17.5 Reading Data from a Sequential File

- ❑ 已完成功能:

  打开并顺序读取文件内容, 直到文件结束

- ❑ 文件位置指针: 指向下一个将要读(get指针)或写(put指针)的字节位置

- ❑ 问题: 如何重新定位文件位置指针?



17.5 Reading Data from a Sequential File

- istream成员函数

```
seekg( streamoff, ios::seek_dir );
tellg();// 返回当前get指针位置
```

- ostream成员函数

```
seekp( streamoff, ios::seek_dir );
tellp();// 返回当前put指针位置
```

17.5 Reading Data from a Sequential File

- 文件位置指针的偏移量和Seek Direction
  - ios::beg – the default
    - Positioning relative to the beginning
  - ios::cur
    - Positioning relative to the current position
  - ios::end
    - Positioning relative to the end

17.5 Reading Data from a Sequential File

- position to the nth byte of fileObject (assumes ios::beg)

  ```
  fileObject.seekg( n );
  ```

- position n bytes forward in fileObject

  ```
  fileObject.seekg( n, ios::cur );
  ```

- position n bytes back from end of fileObject

  ```
  fileObject.seekg( n, ios::end );
  ```

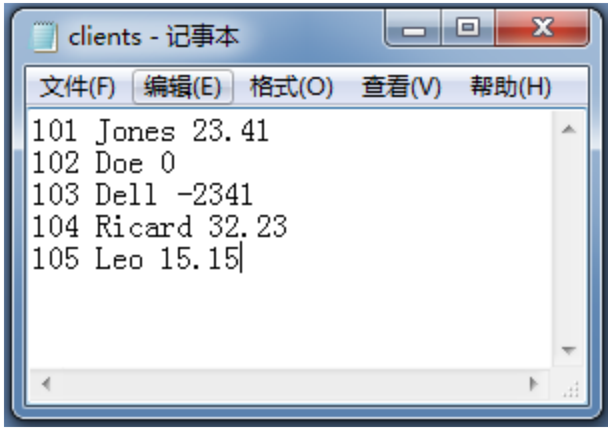
- position at end of fileObject

  ```
  fileObject.seekg( 0, ios::end );
  ```

- assigns the "get" file-position pointer value to variable location of type long:

  ```
  location = fileObject.tellg();
  ```

17.5 Reading Data from a Sequential File



```
101 Jones 23.41
102 Doe 0
103 Dell -2341
104 Ricard 32.23
105 Leo 15.15
```

❑ 信用卡账户管理

- Zero balance: 没有消费，没有存款
  - 1 balance == 0
- Credit balance: 有存款
  - 2 balance < 0
- Debit balance: 有欠款
  - 3 balance > 0

❖ inClientFile.clear(); // reset eof for next input

inClientFile.seekg(0); // reposition to beginning of file

程序解读 P642 17.8

Summary

- 三种文件流
  - • 文件输入流(ifstream)
  - • 文件输出流(ofstream)
  - • 文件输入/输出流(fstream)
- 文件处理步骤
  - • 定义文件流对象
  - • 打开文件：open
  - • 读写文件
  - • 关闭文件：close

Summary

❑ 文件读写

❑ • 顺序文件操作: 从文件的开始处依次顺序读写文件内容, 不能任意读写文件内容.

❑ • 读: 文件流类的get、getline、read成员函数以及抽取符 “>>”

❑ • 写: put、write函数以及插入符 “<<”

Homework

- ❑ 实验必选题目：

  17.14

- ❑ 实验任选题目：

- ❑ 作业题目：