Section 4.2
Files and Streams

1. Streams
2. Files
3. Error state flags

4.2.1 Streams

- What is a stream?
  - a sequence of bytes
  - the flow from data source to data sink
    - from a source to program memory
    - from program memory to a sink
  - data sources and data sinks
    - keyboard, console
    - files
    - printers
    - network adapters
    - ... etc ...

Streams (cont.)

- **`iostream`** library has generic I/O template specializations
  - **`istream`**
    - important object: **`cin`**
  - **`ostream`**
    - important objects: **`cout, cerr, clog`**

Streams (cont.)

- Characteristics of streams
  - maintain a set of error bits
    - they indicate the state of the stream
      - **`good`** bit, **`fail`** bit, **`bad`** bit
  - offer member functions to test error bits
    - ... more on this later ...

Streams (cont.)

- Characteristics of streams (cont.)
  - overloaded `!` operator
    - returns true if one of the error bits is true
  - cast to `void*` operator
    - invoked implicitly when a stream is tested as a condition
    - converts the stream to a pointer
      - null if one of the error bits is true
      - non-null if no error bit is true
    - tests the pointer
      - null == zero == false
      - non-null == not zero == true
  - **`coding example <p1>`**

Streams (cont.)

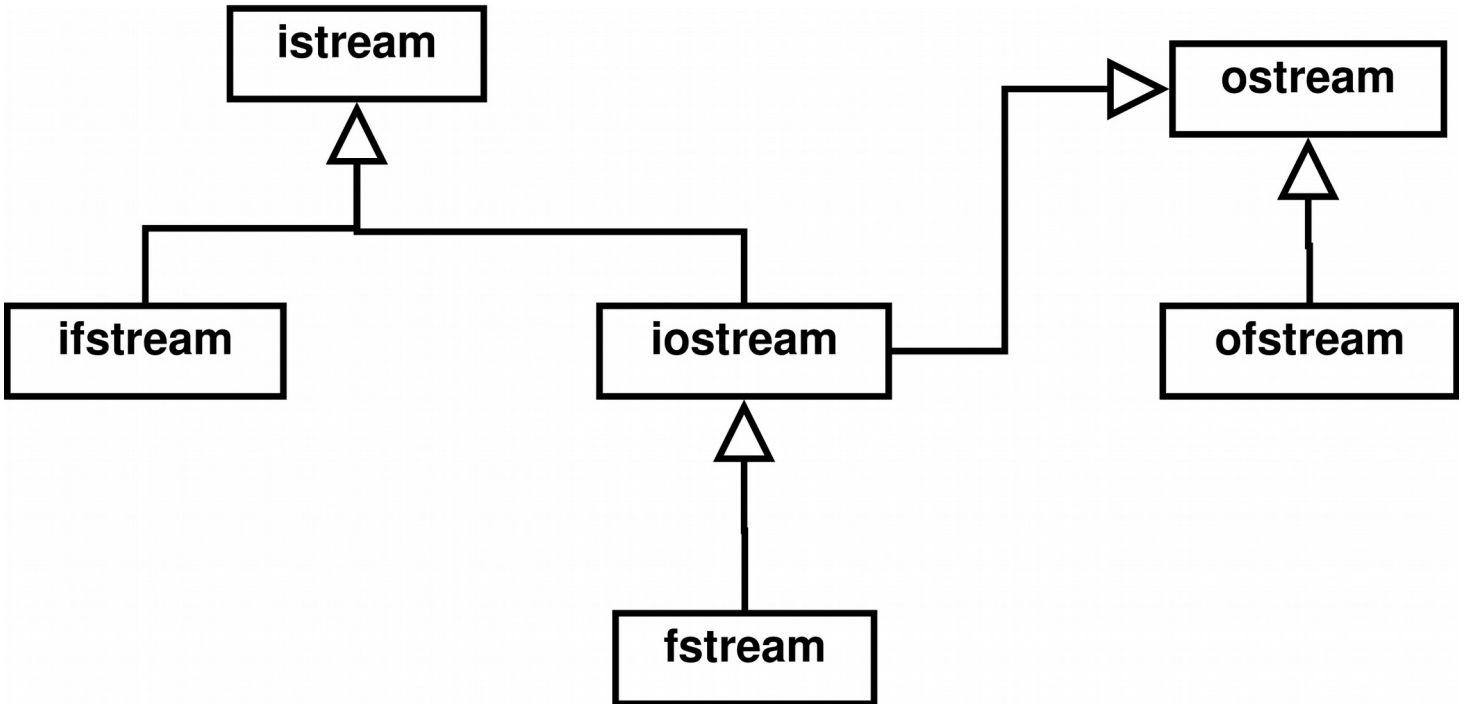
- Characteristics of input streams
  - two types of input
    - formatted data
      - use stream extraction operator **`>>`**
    - unformatted data
      - use member functions **`get()`**, **`getline()`**
  - end-of-file marker
    - value is OS dependent
    - tested using member function **`eof()`**

4.2.2 Files

- What is a file?
  - a stream kept in *persistent storage*
    - also called *non-volatile storage*
- Characteristics of files
  - linear array of bytes
  - terminated by end-of-file marker
  - in C++, files are represented as objects

Files (cont.)

- `iostream` library has file I/O template specializations



Files (cont.)

- **`iostream`** library has file I/O template specializations
  - **`ifstream`**
    - derived from **`istream`**
    - objects represent input files
  - **`ofstream`**
    - derived from **`ostream`**
    - objects represent output files

Files as Objects

- Characteristics of both **`ofstream`** and **`ifstream`**
  - maintain a file buffer object
    - file buffer destructor closes the file
  - can be tested for errors or end-of-file, just like other streams
    - overloaded **`!`** operator
    - cast to **`void*`** operator

Files as Objects (cont.)

- Useful member functions for **`ofstream`** and **`ifstream`**
  - constructor
    - can optionally open the file
    - second parameter indicates mode
      - for input files, file pointer can be re-positioned
  - file management
    - **`open`**
    - **`close`**
- **`coding example <p2>`**

Files as Objects (cont.)

- Useful **`ofstream`** member functions
  - **`<<`**
  - **`put`**
  - **`flush`**

Files as Objects (cont.)

- Useful **`ifstream`** member functions
  - **`>>`**
  - **`get`**
  - **`getline`**

4.2.3 Error State Flags

- Stream objects contain flags to indicate stream state
  - **`good`** bit
    - if true, it indicates that there are no errors
  - **`fail`** bit
    - if true, indicates a formatting error
  - **`bad`** bit
    - if true, indicates an unrecoverable error

- **`istream`** objects also have:
  - **`eof`** bit
    - if true, indicates that the end of file has been reached

Error State Flags (cont.)

- Stream member functions for error state bits
  - testing individual bits
    - **`fail()`** returns true if **`fail`** bit is true
    - **`bad()`** returns true if **`bad`** bit is true
    - **`eof()`** returns true if the end-of-file is true
    - **`good()`** returns true if none of the above are true
  - resetting the stream
    - **`clear()`**
      - clears individual error and end-of-file bits
      - by default, clears all errors and sets the **`good`** bit