Folk Chapter 2

Fundamental File Processing Operations

Physical vs. Logical Files

- **Physical** file: a collection of bytes stored on some external medium under a single identifier.
- **Logical** file: a file as seen by a program, i.e. essentially a byte stream.
- An operating system establishes and controls the correspondence and linkage between a logical and physical file.
- In a modern OS, this linkage is performed within a program.

Opening Files

- The process of file *opening* allows a file to be used by a program.
- Two opening options:
  - existing;
  - new – deletes existing contents in the physical file.
- Positioning is at file beginning.
- C++: `fd = open(filename, flags [,pmode]);`

Closing Files

- Closing a file:
  - allows the logical file name or descriptor to be associated with another file;
  - ensures that all has been written to the file (buffer flushed);
  - performed automatically by OS upon normal program termination.
  - Manual closure protects against data-loss in the event of program interruption or abnormal termination
- Buffer: temporary & intermediate storage interposed between a program and a file.
  - (allows for more efficient I/O via blocking)

Reading & Writing
(low level)

- Sequential I/O
- Functions:
  - `Read(Source_file, Dest_addr, Size)`
  - `Write(Dest_file, Source_addr, Size)`
- Streams:
  - `File = fopen(filename, type)`
- End-of-File
- Read/write pointer

Seeking

- Random-access I/O
- View file as if it were an *array*
  - `Seek(Source)file, Offset)`
- C streams:
  - `Pos = fseek(file, byte_offset, origin)`
- C++ streams:
  - `File.seek[gp](byte_offset, origin)`

Special Characters in Files

- Uninvited character modifications that RMS or I/O packages may make to files.
- Usually associated w. EOF & EOL.
- Portability bugbear

Unix Directory Structure

- Set of directories and files organized as a ***tree*** – graph.
- “/” ***root*** at top.
- Directories are special types of files.

Physical Devices and Logical Files

- Regardless of the physical representation and manifestation of a file, e.g. disk, keyboard, A/D card, printer, CRT, etc., Unix views them *identically* as logical files.
- **UNIX** shell I/O conduits:
  - **Pipes** mediate between standard and file I/O.
    - {e.g. `program_1 | program_2`}
  - **Redirection operators** redirect standard input and error
    - *from* programs *to* files {`program > file`}
    - *from* files *to* programs {`program < file`}

File-Related Header Files

- C and C++ offload most I/O operation and configuration details to a set of standard libraries.
- The details of which are encapsulated by the corresponding set of header files {<library>.h}
- Those relating to material in this chapter are
  - `Iostream.h, fstream.h, fcntl.h, file.h, stdio.h, iostream.h, fstream.h, stdio.h`