File Operations

File Operations Topics

- Stream
- File Level Operations
- fopen Mode
- Binary and ASCII Streams
- Binary File Layout
- Example less4.c

Stream

- FILE
  - data-type, hold information about a stream
  - an object of type FILE * is created by calling fopen
  - used as argument to MOST I/O facilities
  - contains among such as current stream position, end-of-file, etc.

Stream

- ASCII streams
  - text stream divided into lines
  - each line consists of ASCII characters only
  - each line is separated by a newline character
  - each line has a maximum length of 254 characters
  - generally, these files Portable

Stream

- Binary streams
  - sequences of data values of type char
  - used to transparently record internal data
  - implementation does not have to distinguish

File Level Operations

- FILE *fopen(const char *filename, const char *mode);
  - opens the file with the specified mode
  - returns NULL if open failure
  - otherwise, returns pointer to stream information
  - maximum number of stream defined in FOPEN_MAX (8 min.)

File Level Operations

- FILE *freopen(const char *fname, const char *mode, FILE *stream);
  - closes stream as if a call to fclose
  - ignores any error doing so
  - new file is opened, uses existing stream
  - returns NULL if open failure

File Level Operations

- int fclose(FILE *stream);
  - empties any internal buffers and closes the stream
  - returns 0 or EOF if error detected
  - exit() flushes and closes all files

File Level Operations

- int fflush(FILE *stream);
  - empties buffers
  - returns 0 or EOF if error detected
  - *** not normally used **

File Level Operations

- long int ftell(FILE *stream);
  - returns stream position in open file OR -1 on failure.
  - for binary files, the value returned is the number of bytes preceding the current position (i.e. offset from head of file).
  - for ASCII files, the value returned is implementation dependant.
  - the value is always suitable as the second parameter to fseek.

File Level Operations

- int fseek(FILE *stream , long int offset, int wherefrom);
  - *offset* specifies number of bytes from *wherefrom* position.
  - *wherefrom* indicates at what point *offset* should be measured:
  - **SEEK_SET** - *offset* bytes from beginning of file
  - **SEEK_CUR** - *offset* bytes from current position of file
  - **SEEK_END** - *offset* bytes from end of file (negative values specify positions before the end; positive values extend the file with unspecified contents).
  - returns 0 on success; otherwise non-Zero

File Level Operations

- void rewind(FILE *stream);
  - resets stream position to the beginning of the file
  - per ANSI, equivalent to (void) fseek(stream, 0L, SEEK_SET);

File Level Operations

- int rename(char *oldname, char * newname);
  - returns 0 is operation succeeds; otherwise non-zero
- int remove(char *filename);
  - returns 0 if operation succeeds; otherwise non-zero.
  - remove is implementation specific – at minimum file is not available to be subsequently opened.
  - if file does not exist or is open, remove action is implementation specific.

fopen Mode

- "r" open an existing file for input
- "w" create a new file, or truncate an existing one, for output
- "a" create a new file, or append to an existing one, for output
- "r+" open an existing file for update (reading and writing)
- "w+" create a new file, or truncate an existing one, for update
- "a+" create a new file, or append to an existing one, for update

fopen Mode

- When accessing **binary** files on **Windows** platforms, "b" must be included with the file mode. On **Unix** platforms, the "b" is ignored.
- Examples
  "wb", "rb", "bw", "bw+", "b+w", "w+b". etc.
- Mode Table

	mode					
property	r	w	a	r+	w+	a+
Named file must already exist	yes	no	no	yes	no	no
Named file must already exist	yes	no	no	yes	no	no
Existing file content lost	no	yes	no	no	yes	no
Read from stream permitted	yes	no	no	yes	yes	yes
Write to stream permitted	no	yes	yes	yes	yes	yes
Write begins at end of stream	no	no	yes	no	no	no

Binary and ASCII Streams

- Binary Streams
  - size_t fread(void *ptr, size_t elm_size, size_t count, FILE *stream);
    - reads count elements of size elm_size from input stream
    - returns actual number of items read (may be less than count)
    - OR, returns 0 if an error encountered
  - size_t fwrite(void *ptr, size_t elm_size, size_t count, FILE *stream);
    - writes count element of size elm_size to stream
    - the actual number of items written is returned
    - if (fwrite != count) an error has been encountered

Binary and ASCII Streams

- ASCII Streams
  - most other I/O functions!!
- Example
  - http://faculty.washington.edu/sproedp/advc/csamples/less4.c.html

Binary File Layout

- File Header
  - file level info
  - time, date, etc
  - bytes to first record
  - etc
- Binary Record
  - Header
    - bytes to end of header/start of record
    - type of record
    - bytes to end of record
    - etc
  - Content
    - binary content of the record
    - “the bits”
- *-- More Binary Records --*