CSC161 2011S *Imperative Problem Solving*

Class 42: Files

Held: Tuesday, 19 April 2011

Summary: We explore file input and output.

Related Pages:

- EBoard.
- Lab: Files.
- Reading: K&R 7.5-7.7.

Notes:

- We have an observer today. Ignore it.
- I have responded to your your questions on the files reading.
- I was sorry (and surprised) not to receive questions from DF, DG, JN, EO, KS, and DU.
- Reading for tonight: K&R 6.1-6.5, 6.7.
- Due to prereg and College committee work, I need folks to sign up for appointments. Times at http://www.cs.grinnell.edu/~rebelsky/Schedule/signup.2011S.txt. Signup via email.
- EC for any of the Activism Week activities, e.g., Sarah Smith on Wednesday. [academic]
- EC for Ginsburg convo, Thursday. [academic]
- EC for open discussion with Jane Ginsburg, Thursday, 9:00-10:30 in Mears Cottage. [academic]
- EC for open discussion with Jane Ginsburg, Thursday, 2:30-4:00 in Mears Cottage. [academic]
- EC for Benji's films event Saturday at 4. [peer]
- EC for Spring Bling on Friday. [peer]
- EC for Grinnell Monologues on Friday/Saturday. [peer]
- EC for ISO Cultural evening on Saturday. When else can you watch AG dance? [peer]

Overview:

- Files.
- Files in C.
- Lab.

Files

- Yay! Another basic type.
- Issues:
  - We need persistant storage of data.
  - Most operating systems provide files; it makes sense to be able to write programs that manipulate them.
- C and Unix's contributions:

- Files are just sequences of bytes.
- Treat program input and output identically to system input and output.

Files in C

- What do we need for basic types?
  - The type itself.
  - Functions on the type.
- We use `FILE *` to represent the type.
  - There is underlying information there (just as there are in `int`s, `float`s, and the ilk), but you don't need to know it.
- We use `fopen (`*`filename, mode`*`)` to convert the name of a file to one of these structures.
  - The mode for writing to a file is `"w"`
  - The mode for reading from a file is `"r"`
  - The mode for appending to a file is `"a"`
  - There are other more complicated options.
- We use `fclose` to tell the operating system that we're done using a file.
- We use `fgetc` (char), `fgets` (string), and `fscanf` (general) to read from a file.
- We use `fputc` (char), `fputs` (string), and `fprintf` (general) to write to a file.
- `stdout`, `stdin`, and `stderr` are "pseudo-files" - you can treat them just like files, except you don't use `fopen` and `fclose`.
- `ferror` checks if something is wrong with a file.

Lab

- Do the lab.
- Be prepared to discuss how to do unit testing with the program.
