Lab 4

Modularity and Abstraction
File I/O

September 29th, 2010
James Marshall

Bash Tips

- Useful commands
  - man
  - Files: cp, mv, rm, rm -r (be VERY careful)
  - Navigation: cd, cd .., cd ~, pwd,
  - cat, tac, ls, ls -al, chmod +x
- File redirection
  - >, >>, 2>, 1>

Recursion

- Proof by Induction
  - Related concept
  - Show for n = 1 (base case)
  - Prove for n + 1 (recursive case)

Motivation

- For learning C: widespread in embedded systems.
  - Most medical devices
  - Hardware is controlled by software
- File I / O
  - Finally, something useful
- Modularity and Abstraction:
  - Modern programs are big

Really Big

- View Linux Kernel: http://lxr.linux.no/
- Source: http://en.wikipedia.org/wiki/Source_lines_of_code

Operating System	Lines of Code in <u>Millions</u>
Windows NT 3.1	4-5
Windows XP	40
Mac OS X 10.4	86
Linux Kernel 2.6.32	12.6
Debian 5.0	324

Modularity

- Desirable traits in a unit of code:
  - Single purpose
  - Side-effect free
  - Independent
  - Portable
- These are general guidelines

Rules of Thumb

- Should be able to rewrite a function, without having to rewrite your entire program.
- Avoid code duplication

Abstraction

- Most important aspect of computer science.
- Easy to understand!
- We are surrounded by them, use them every day.

Abstraction Examples

- Computer vs. Dell Optiplex GX280
- File vs. 2048 bytes starting at 0xAE0018B0
- List vs. Sorted Linked List

Your Code Should Provide:

- Abstractions:
  - Could replace the Binary Search Tree with another structure, perhaps a heap.
- Modularity:
  - Can call insert() function with input from a file or the keyboard.

File I/O

- Use fopen and fclose

```
FILE *fp;
fp = fopen(“input.txt”, “r”);
```

- Then use fprintf and fscanf

```
int lenght;
fscanf(fp, “%d”, &length);
```