week5 section

Ezra Zigmond
ezigmond@college.harvard.edu

October 4, 2015

pset 2 feedback

- "Magic numbers" (Use #define instead)
- DRY (Don't repeat yourself)
- Maximize code reuse
- Comments (much better!)

week4 Topics

- input redirection
- file I/O
- memory layout
- dynamically allocated memory
- pointers
- pset4

Redirection

- $>$ for output e.g. `./hello > output.txt`
- $>>$ to append instead of overwriting
- $<$ to use a file as input e.g. `./hello < input.txt`
- $|$ takes the output e.g. `./hello | ./hello2`

File I/O

- So far you've read and written to stdin and stdout (your screen)
- But you can also read and write files!
- Close every file you open (avoid memory leaks)
- Common file functions: fopen(), fread(), fwrite(), fgets(), fputs(), fgetc(), fputc(), fclose()

The Stack

- Two regions of memory: stack and heap
- Each function gets its own stack frame
- Variables within a stack frame have local scope and are protected from other functions
- Compiler must be able to determine the size of the stack frames

The heap and malloc()

- What if you don't know how much memory you need?
- Any memory allocated during runtime is allocated dynamically on the heap using malloc
- `int* ptr = malloc(sizeof(int));`
- You have to `free()` any memory you allocate

Pointers...

- A pointer is an address
- Addresses have values