2 Lecture: History and Definition of an O.S.

Outline:

Announcements
About the lab
About the assignment
Everything you wanted to know about C development
Aside: Review of 357
  Review of Unix IO
  Processes, etc.
  Compilation

2.1 Announcements

- Be reading along in the book. Tanenbaun and Woodhull are good writers.
- Coming attractions:

Event	**Subject**	**Due Date**			**Notes**
asgn2	LWP	Mon	Jan 26	23:59	
asgn3	dine	Wed	Feb 4	23:59	
lab03	problem set	Mon	Feb 9	23:59	
midterm	stuff	Wed	Feb 11		
lab04	scavenger hunt II	Wed	Feb 18	23:59	
asgn4	/dev/secret	Wed	Feb 25	23:59	
lab05	problem set	Mon	Mar 9	23:59	
asgn5	minget and minls	Wed	Mar 11	23:59	
asgn6	Yes, really	Fri	Mar 13	23:59	
final (sec01)		Fri	Mar 20	10:10	
final (sec03)		Fri	Mar 20	13:10	

  Use your own discretion with respect to timing/due dates.

- Late Days (how they work)
- Beware of malloc(): there are a kajillion implementations out there, but you need to do your own.
- If you've been through this before unsuccessfully, *tell me.* I can ask you awkward questions that'll improve your odds of never doing 453 again.
- tryLab01
  - `~pn-cs453/bin/longlines.pl`
- tryAsgn1
  - (*don't copy it*)
  - Run it on a 64-bit machine (e.g. unix1–4)
    (Make sure it's one with 32-bit libraries; unix5 doesn't have them for whatever reason.)
  - Consider the effects of architecture (how big is an int?) and uninitialized data

- Office hours
  - Come
  - Or talk to me during lab. I'm guaranteed to be available.
- "`https://www.cs.vu.nl/~ast/intel/`"
- `gdb` and `valgrind` are your friends.

2.2 About the lab

Remember man name(section)

- pipeit (`ls | sort -r > outfile`)

  What does this mean?

- There are three processes here
- You are the plumber/reaper
- This demonstrates the process abstraction that an OS provides. All communication and synchronization takes place through the OS.
- These must be concurrent (why?)

2.3 About the assignment

- `malloc(3)`: How does it work?
- libraries: Two forms
  - static (`libmalloc.a`)
  - shared object (`libmalloc.so`)
    - `LD_LIBRARY_PATH`
    - `LD_PRELOAD`
- don't call `sbrk(2)` for every call to `malloc(3)` (quilting analogy)
- remember how pointer arithmetic works (in the size of the pointee)
- `uintptr_t` from `<stdint.h>`
- About that Makefile...
- Note: the order of link commands matters to `gcc`
- Also Note: setting environment variables:

[ba]sh	`VAR=value`
	`export VAR`
[t]csh	`setenv VAR value`

2.4 Everything you wanted to know about C development

- The Environment
- Linking
- Loading
- Make
- gdb

2.5 Aside: Review of 357

2.5.1 Review of Unix IO

- file descriptors
- `open(2)` vs. `fopen(3)` (and permissions)
- `dup(2)` and `dup2(int old, int new)`
- pipes, how they work

2.5.2 Processes, etc.

- Lifecycle

Birth	fork()
Death	termination (exit(),_exit(),return,abort(),signal)
Afterlife	reaping with wait() or waitpid()

  - `wait(2)`
  - `waitpid(2)`
  - `WIFEXITED()/WIFSIGNALLED()/`
  - `WEXITSTATUS()`

2.5.3 Compilation

- The compiler (gcc,ACK,clang)
- The linker (ld, gcc)
  - `-L`*path*
  - `-l`*name*
  - `LD_LIBRARY_PATH`
  - `LD_PRELOAD`
- The loader (ld.so)
- Libraries
  - Static (*.a). Made with `ar(1)`
  - Dynamic (*.so,*.dll,*.dylib) Made with the compiler
- Some thoughts on Make