CMSC216: Finale

Chris Kauffman

Last Updated:
Thu May 9 09:14:00 AM EDT 2024

Logistics

Goals

- Threads Wrap
- Final Exam Logistics
- Review

Assignments: P5

Due Fri 10-May-2024 11:59pm

Date	Event
Mon 06-May	Lab13/HW13 Due
Tue 07-May	Threads Wrap
Wed 08-May	Dis: Review
	Feedback Due
Thu 09-May	**Lec: Practice Exam**
Fri 10-May	P5 Due
Mon 13-May	Final Exam
	4-6pm
	Normal Lecture Location

Questions on anything?

Announcements

Final Exam Question Unlocked

https://piazza.com/class/lrqszzrlvo46gm/post/1051

- Response Rate >= 80% on Course Experiences
- As per our arrangement, Piazza post above has a Final Exam question as it will appear on the final exam.
- **Staff will not discuss this questions or its answer with students** but students may freely discuss answers together to prepare for the question.
- Feedback may still be submitted through Fri 10-May 11:59pm

Course Exit Survey on Canvas

- Open on Canvas, due by Sun 12-May 11:59pm
- 1 Engagement Point for Completing it

Final Exam Logistics

- Final Exam in person in normal lecture location
- ~1.5 pages F/B like 3rd Midterm Exam
  Processes, Memory System Hardware, Virtual Memory, Threads, P5 Material
- ~1.5 page F/B Comprehensive Review, tie together concepts that pervaded the semester
  (F/B = Front/Back)
- 2 hours long, Open Resource

What have we done?

C Programming

Lowest of the “high-level” languages, gives fairly direct control over capabilities of the machine at the expense of coding difficulty and ease of mistakes

Assembly Programming

Tied directly to what a processor can do, studied x86-64 specifically, exposes processor internals like registers, instructions, operand sizes, etc.

Computing Hardware

Basics components like CPU, Registers, Cache Memory, DRAM, Disks, how they interact

Operating System Basics

Programs exist in an environment usually managed by an OS, provides abstractions like Processes, Files, Threads, along with the ability to manipulate and coordinate these through System Calls

Did I miss anything?

Further Coursework / Activities

- **CMSC411 Computer Systems Architecture**: Develops hardware/software interface in more detail, study pipelines + superscalar features in more detail, examine multi-core systems
- **CMSC412 Operating Systems**: Study internal design issues associated with operating systems, handling hardware, tradeoffs on different approaches to management, theoretical algorithms around resource coordination.

Summer Practice

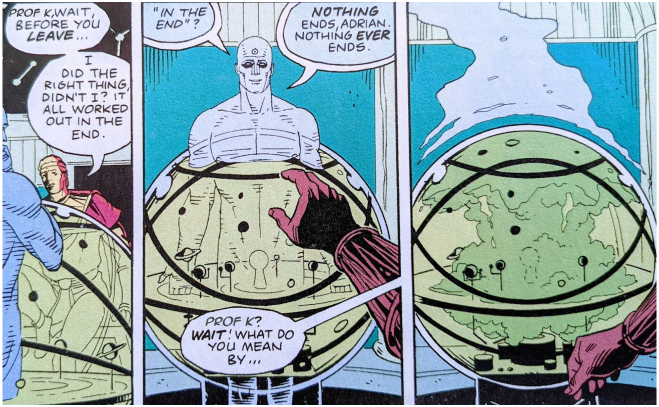
Students often ask what they could do during a break to keep up their computing skills. Here are a few ideas.

- READ: The Art of Unix Programming by Eric S. Raymond Fantastic philosophical and pragmatic discussion of how to build systems that work especially in the Unix environment. (free online)
- COMPLETE: If you didn't finish a project in this course or another, take some time to do so.
- EXTEND: If you use VS Code, Write an Extension for it that does something interesting. This will teach you MUCH about modern software development
- BUILD: Buy an Arduino Microcontroller ($10) and get a "Blinky" routine to run; it's C code! Adafruit has tons of fun toys with accompanying tutorials.
- REST: Take some time away from the screen for fun. Recharging is as important for people as for phones. Play outside. See some people in person. Breathe.

Practice Final

- Take a few minutes to look this over on your own then together
- Kauffman will answer a few questions on it and post solutions later today

Nothing Ever Ends


PROF K, WAIT, BEFORE YOU LEAVE...
I DID THE RIGHT THING, DIDN'T I? IT ALL WORKED OUT IN THE END.
"IN THE END"?
NOTHING ENDS, ADRIAN. NOTHING EVER ENDS.
PROF K? WAIT! WHAT DO YOU MEAN BY...

- What you learned will recur in your career at some point and demonstrate whether you learned it well the first time or need another pass.
- Some of it will change in the future and make you feel old.
- Expect this and stay determined.

Conclusion

It's been a hell of a semester.
I'm proud of all of you.
Keep up the good work.
Stay safe. Happy Hacking.

CMSC412 or bust

Why yes, I do know assembly.

valgrind: got my back on memory!

C code: I know this!

My code is full of shift, bit shift!

Submission accepted

```
movq $0, %eax
retq   #success!
```

```
> make
nothing to be done
```

10/10 Tests Passed

gdb: yeah you know me!

I struggled and built skills

Stack or Heap? I know which!

Debugging complete!

You're STILL filled with

DETERMINATION!

fork()'d and wait()'d

Whew, time for a snack