Organization

CPSC 457: Principles of Operating Systems
Winter 2024

Contains slides from Pavol Federl, Mea Wang, Andrew Tanenbaum and Herbert Bos, Silberschatz, Galvin and Gagne

Jonathan Hudson, Ph.D.
Instructor
Department of Computer Science
University of Calgary

Tuesday, 28 November 2024

UNIVERSITY OF CALGARY

Welcome!

Jonathan Hudson

Assistant Professor (Teaching)

L01 TueThu 9:30 - 10:55 AM ST143

(attempts may be made to broadcast using zoom, not record)

Office: ICT 712

Office hours: 11:00-11:50 AM TueThur or by email-scheduled appointments.

jwhudson@ucalgary.ca

https://pages.cpsc.ucalgary.ca/~jwhudson/CPSC457W24/

In this course…

- from the calendar:

  "An introduction to operating systems principles. Performance measurement; concurrent programs;
  the management of information, memory and processor resources."

- main goals:
  - understand fundamental operating system concepts, such as processes, threads, scheduling, memory, file systems
  - learn about and apply interesting data structures and algorithms
  - **become a better (system) programmer**

Operating systems

- we use different operating systems every day
- nearly all modern electronic devices use them
  - desktop computers, laptops, servers, network routers
  - even phones, watches, thermostats, TVs, cars, fridges, …
- operating system is software that manages hardware and software

Why OS?

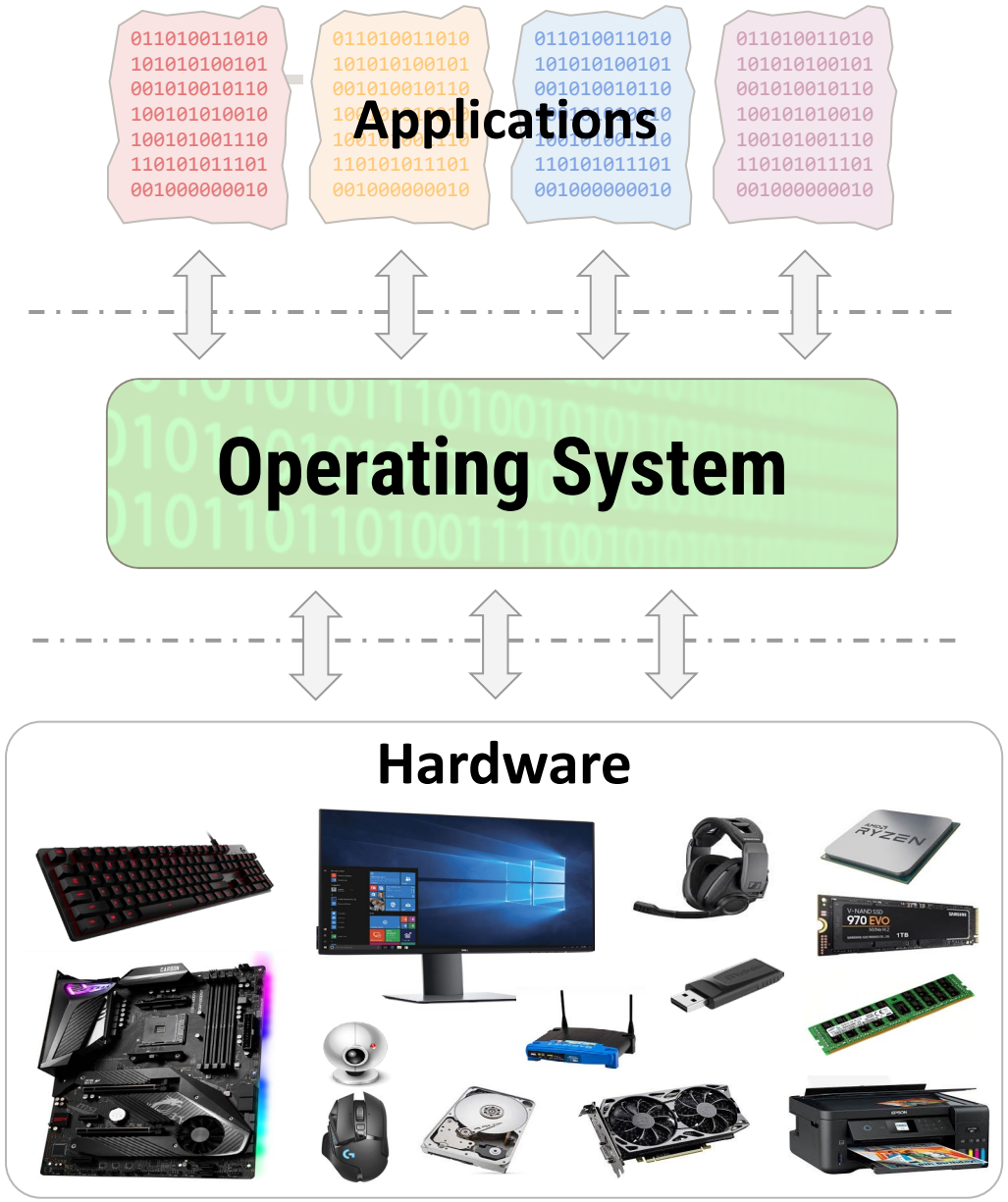
- OS makes a computer easier to use by users:
  - installing/running applications
  - running multiple applications simultaneously
  - installing drivers for new hardware
  - managing files and directories
  - providing nice user interfaces
  - ...

Why OS?

- OS makes programming easier for developers:
  - applications use hardware (read/write/printf…)
  - but raw hardware is difficult to use (remember CPSC 355 ?)
  - OS provides higher-level and easier-to-use abstractions to hardware, e.g. printf()
  - provides controlled allocation for **efficient** and **fair** resource use

- OS hides the complexity of the underlying hardware and gives the developer a nicer view of the computer (through APIs, called **system calls**)

What is an Operating System ?



- OS is software that sits between applications and hardware
- applications talk to the hardware via OS
- understanding OS is key to **system programming**
- useful techniques: data structures, conflict resolution, concurrency, resource management, communication …

UNIVERSITY OF CALGARY

Programming with/without OS

```
#include <stdio.h>

int main() {
  printf("Hello World\n");
  return 0;
}
```

```
[jwhudson@csx2 01]$ cat hello.cpp
#include <stdio.h>
int  main() {
        printf("Hello, World\n");
        return 0;
}[jwhudson@csx2 01]$ g++ hello.cpp
[jwhudson@csx2 01]$ ./a.out
Hello, World
[jwhudson@csx2 01]$
```

Programming without OS

application

printf_nvidia()

printf_ati()

write_sata()

write_usb()

NVIDIA video card

ATI video card

SATA disk

SCSI disk

hardware

Programming with OS

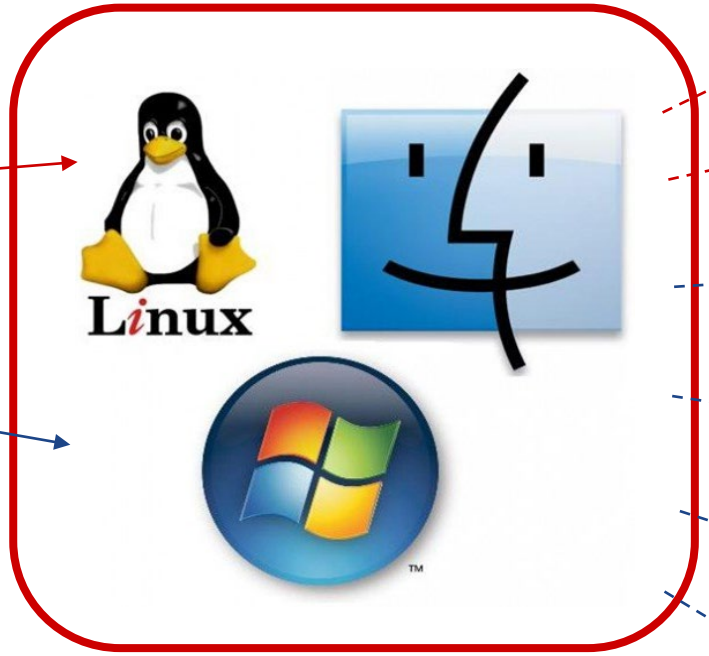
application

operating system

hardware

printf()

write()



NVIDIA video card

ATI video card

SATA disk

SCSI disk

SSD disk

NFS

Logistics

- lectures and tutorials will be **in-person**
- slides by themselves are not enough, they are just outlines…
  - **take notes**, ask questions, take more notes!
  - if you miss a lecture, copy someone else's notes (good reason to make friends)
- lectures will **not** be **recorded**
- tutorials will probably be not recorded (check with your TA)

Website

- Course Website (outside D2L but also linked within)
  - Assignment specs
  - Slides
- D2L
  - Assignment submissions
  - Grades
  - Occasionally announcements
  - Tutorial materials could be posted there as well (check with your TA)

Communication

- in class
  - I usually start and end with a Q&A
  - feel free to ask questions during lectures
- tutorials
  - talk to your TAs
  - email your TAs
- office hours
  - Tue/Thur 11-12pm after class
  - Please inform instructor if you are planning to show up

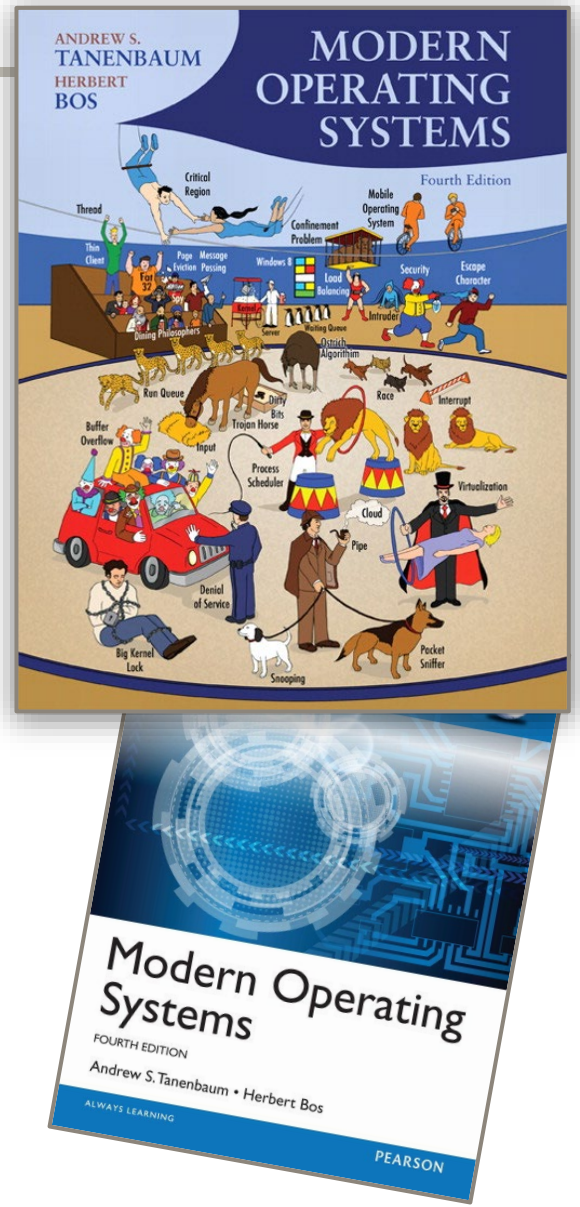
Email

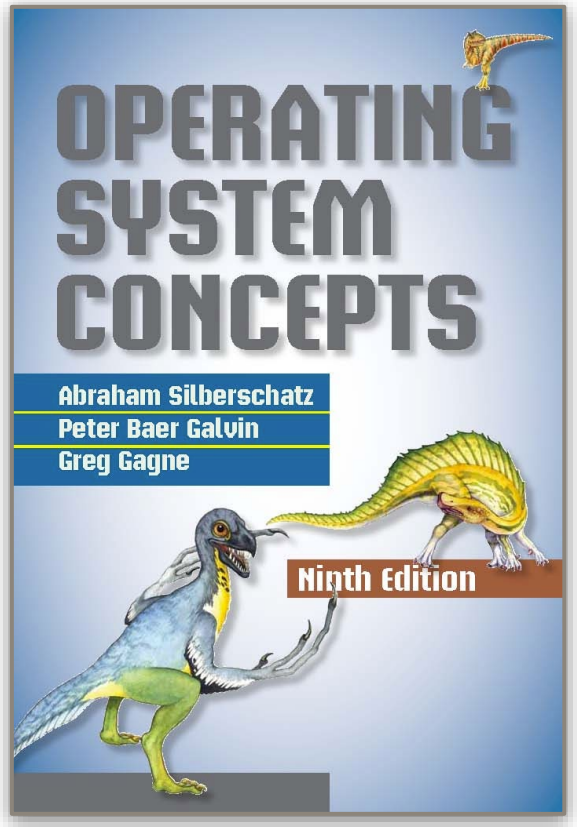
- questions about assignments should go to your TAs
  - get your TA's email in first tutorial or course website
  - email subject should start with "**CPSC 457 W24**"
- email instructor:
  - ask only questions that your TA was unable to answer, or cannot answer
    e.g. assignment extensions
  - email subject should start with "**CPSC 457 W24**"
  - include the **name of your TA** and your **student ID** !!!

Tutorials

- are not optional (*)
- reviews of lecture material
- occasionally new material
- **assignment discussions & hints**
- start next week

Recommended textbooks





- should be available in bookstore https://www.calgarybookstore.ca
- check options (print/ebook/…)
- we will cover mostly first 6 chapters of Tanenbaum

UNIVERSITY OF CALGARY

Assignments

- 6 assignments, worth 15%, 19%, 21%, 22%, 8% and 15%, respectively
- assignments constitute 100% of the grade!!!
- all assignments count toward grade
- **no way to make up for a bad/missed assignment**
- assignments released throughout the term, back-to-back
- 6 late day bank, pay attention to deadlines posted on D2L
- final grade computed as a weighted sum of all assignments and converted to term letter grade using cutoffs*:

0	60	63	67	71	75	79	83	87	91	95	100
F	D	D+	C-	C	C+	B-	B	B+	A-	A	A+

* could be curved

Assignments

- read assignment specs as soon as they are released
- if you have questions, ask your TAs as soon as possible
  - your TAs are humans
  - do not expect answers during evenings/weekends
- ask questions in class & during office hours
- assignments require **substantial amount of coding in C++**
- some assignments may include written questions
- submit to D2L (as specified in instructions)
  - submit individual files, eg. `solutions.pdf, file1.cpp, file2.cpp`
  - check your submission by re-downloading it
- deadlines are strict – after 6 day late bank

Assignments

- **C++** unless explicitly stated otherwise
- your programs must work, i.e. wrong output = 0 marks
- your programs must be efficient
  - by default your code must finish under **10 seconds** on any valid input
  - this limit may be defined otherwise in the assignment specs
- your programs should compile without warnings,
  warnings are usually bugs, leading to unpredictable results ⟶ wrong outputs
  ⟶ 0 marks

⇒ your assignments **must** run on the departmental **linuxlab** machines

https://ucalgary.service-now.com/it?id=kb_article&sys_id=2ba72f11dbb8ebc07cab5068dc9619fd

Assignments

- read the specifications carefully
- **create your own test inputs** based on specifications
  - few sample inputs will be included, but these are not enough
- assignments will be graded based on **correctness & efficiency**
  - marking based on several (hidden) tests
  - each test marked independently
  - total mark = sum of all test marks
  - code compiled using `-O2` and timed on `linuxlab.cpsc.ucalgary.ca` machines
  - wrong output or program takes too long will be rewarded with 0 marks for test
- start working on assignments early, submit early, even if incomplete, double check submission
- ask questions

Where to work on assignments

- in-person on Linux lab machines on main floor of MS (supported)
- remotely using SSH by connecting to linuxlab (some support from TAs)
  - my favorite: VS Code with remote editing via SSH
  - 2nd favorite: SSH + text editor (emacs/vim/nano)
  - least favorite: local editor + WinSCP/FileZilla
- other options (very little to no support from TAs)
  - install Linux on your own computer, e.g. single OS, dual boot
  - install Linux in a virtual machine (vmware, virtualbox, parallels, bootcamp)
  - use WSL, cygwin, docker
  - use OS X
- no matter what you choose: your assignments MUST run on the linuxab machines!!!

your assignments MUST run on the departmental linuxlab machines!!!

your assignments MUST run on the departmental linuxlab machines!!!

your assignments MUST run on the departmental linuxlab machines!!!

your assignments MUST run on the departmental linuxlab machines!!!

your assignments MUST run on the departmental linuxlab machines!!!

your assignments MUST run on the departmental linuxlab machines!!!

your assignments MUST run on the departmental linuxlab machines!!!

your assignments MUST run on the departmental linuxlab machines!!!

your assignments MUST run on the departmental linuxlab machines!!!

your assignments MUST run on the departmental linuxlab machines!!!

your assignments MUST run on the departmental linuxlab machines!!!

UNIVERSITY OF CALGARY

Academic misconduct

- all assignments must reflect **your own individual** work
- plagiarism is a serious academic misconduct

  “A single offence of cheating, plagiarism, or other academic misconduct, on term work, tests, or final examinations, etc., may lead to disciplinary probation or a student’s suspension or expulsion from the faculty by the dean, if it is determined that the offence warrants such action.”

- refer to the University Calendar for more details
- please read the Academic Integrity Student Handbook

  https://www.ucalgary.ca/live-uc-ucalgary-site/sites/default/files/teams/9/AI-Student-handbook-1.pdf

UNIVERSITY OF CALGARY

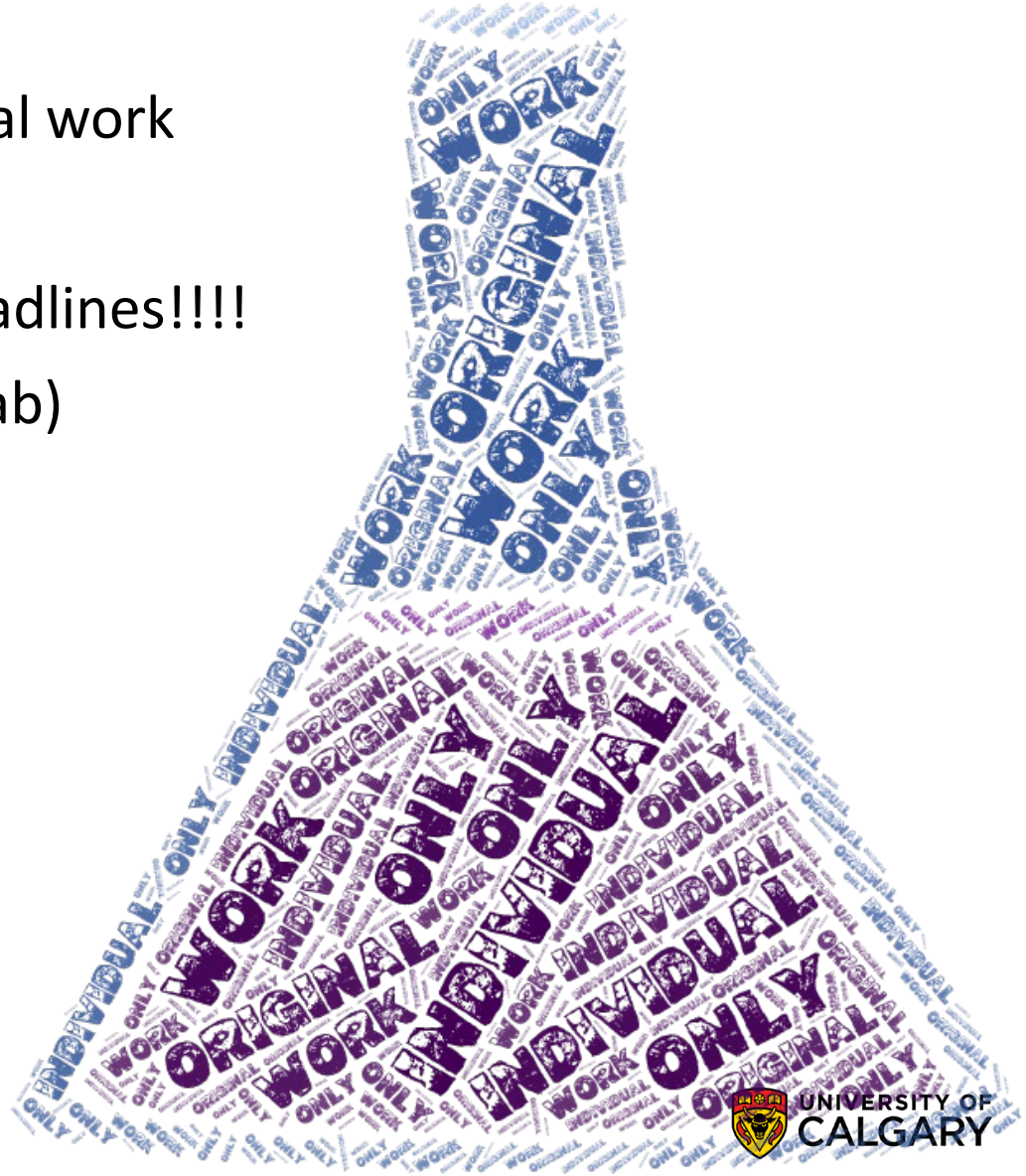
Common academic misconduct offences

- collaborating on individual components
  - e.g. sitting next to each other while programming, screen sharing, dividing work
  - renamed variables != unique work (or any other cosmetic changes)
- copying friend's work with/without permission, sharing work with friends
  - e.g. posting code or screenshots on Discord
  - some students try to be too 'helpful'
- sharing solutions after deadline
  - not realizing some students get extensions
- copying from internet
  - e.g. copying previous year solutions from GitHub, GitLab, StackOverflow, GeeksForGeeks
- posting assignments publicly !!!
  - e.g. public GitHub repos
- buying solutions / hiring help
  - chegg.com, coursehero.com

Academic misconduct

- make sure the work you submit is your own original work

- **do not** share your code with others, even after deadlines!!!!
- **do not** post your code publicly (eg. GitHub or GitLab)
  - use private repos instead
- **do not** post screenshots
- **do not** share screens
- **do not** share pseudocode
- **do not** look up previous assignments online
- **do not** look at anyone else's code
- **do not** discuss assignments with other students



UNIVERSITY OF CALGARY

Academic misconduct

- Can I discuss or help other students with assignments?
  - safe answer: **NO**
  - longer answer: at your own risk
  - if you help someone or ask someone for help, and they submit similarly looking code, you both will be investigated for academic misconduct

- Can I copy someone's solution and then include a citation?
  - **NO**. This is still misconduct – "Failure to comply with instructor's expectations"
  - all assignments in this course are expected to be based on individual work!!!

- when in doubt, check with your instructor

Academic misconduct

- we may use electronic plagiarism checkers (MOSS), and and compare all student submissions, including some online (e.g. GitHub) solutions, and solutions from previous semesters
- usual penalty for 1st offence is an F on the assignment + 2 compulsory workshops,
  and one or more unpleasant interviews
- 2nd offence is usually an F in the course
- etc.

Assignments - one more time

- submit on **D2L** but I may experiment with **Gradescope** this semester
- 0 marks for late assignments after 6 day late day bank, non-working assignments, wrong outputs, too inefficient
- **C++** [1]
- **10 seconds** limit [1]
- compile without warnings
- must run on the **linuxlab** machines
- **individual work**

1. unless specified otherwise

Advice from students from a previous semester

1. Based on your experience this semester, what should a student do to succeed in this course?

- A student should make sure to start the assingments early, and should make sure to attend every single class. Also, make sure they fully read and understand the assignemnts.
- As long as you do your assignments early you should be able to finish this course with a good mark.
- Attend classes and practice C++
- Attend lectures and tutorials, be engaged (ask questions, take notes) and start assignments as early as possible.
- Attend tutorials, start assignments well in advanced. Going to lectures helps understand all the theory for the assignments (as expected).
- Follow the course material.
- Go to all classes, start on assignments early
- Start assignments early.
- Students should have good prior knowledge of C and C++. Students should be excited about learning to code better by finding efficiencies, working with the operating system, and using parallel processing. Students have to be willing to use bash commands and command line text editors such as emacs. Students should start on assignments early and should plan out the assignments on paper before beginning to code.
- practice programming a lot. Start assignment earlier, and don't take this course in spring term

CPSC passwords

- if you forgot your CPSC password, go to password.ucalgary.ca
- use your UCIT credentials to login
- reset your password (under "Other Accounts")
- Note: your CPSC account is different from your UCIT account!

How to succeed

- come to lectures/tutorials and take notes
- if lectures are not enough, read the textbook
- **read assignment specs immediately, start assignments early, work by yourself**
- ask questions (me and TAs)
- come to office hours

*"What I **hear**, I forget.*
*What I hear and **see**, I remember a little.*
*What I hear, see and **ask** questions about*
*or **discuss** with someone else, I begin to understand.*
*What I hear, see, discuss, and **do**, I acquire knowledge and skill.*
*What I **teach** to another, I master."*

from "Active learning: 101 strategies to teach any subject," Silberman, M. (1996)

Questions?

Onward to …
Basic Concepts

Jonathan Hudson
jwhudson@ucalgary.ca
https://pages.cpsc.ucalgary.ca/~jwhudson/

UNIVERSITY OF CALGARY