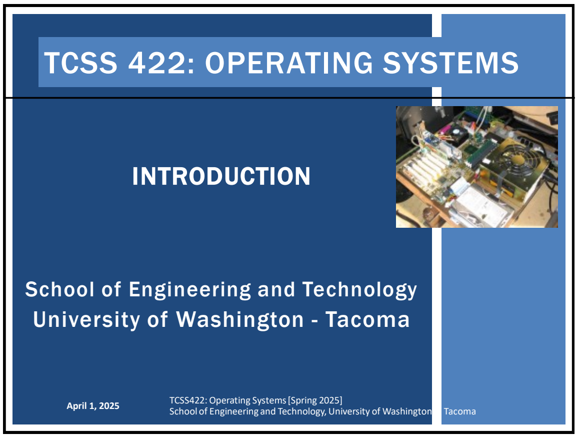


TCSS 422: OPERATING SYSTEMS

INTRODUCTION

School of Engineering and Technology
University of Washington - Tacoma

April 1, 2025 TCSS422: Operating Systems [Spring 2025]
School of Engineering and Technology, University of Washington - Tacoma

1

Join by Web PollEv.com/weslloyd

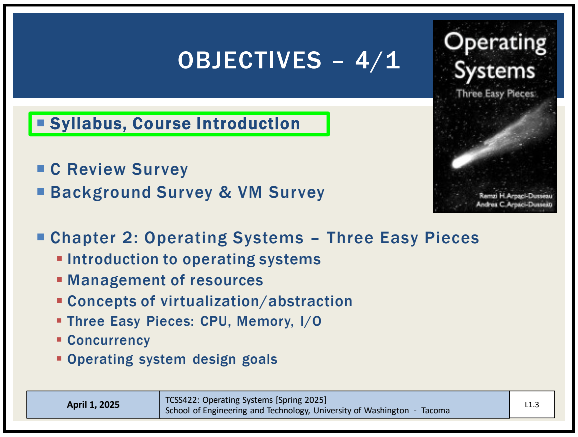
INTRODUCTIONS: What is your name? nickname / alias? and list one or more areas of interest in Computer Science:

0

Join by Web
PollEv.com/weslloyd

Join by QR code
Scan with your camera app

2



OBJECTIVES – 4/1

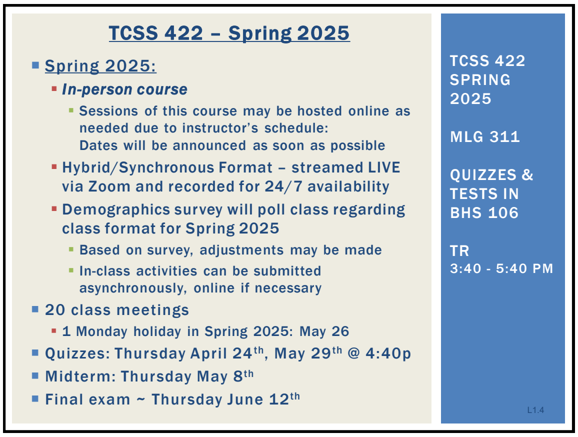
- **Syllabus, Course Introduction**

- C Review Survey
- Background Survey & VM Survey

- Chapter 2: Operating Systems – Three Easy Pieces
  - Introduction to operating systems
  - Management of resources
  - Concepts of virtualization/abstraction
  - Three Easy Pieces: CPU, Memory, I/O
  - Concurrency
  - Operating system design goals

April 1, 2025 TCSS422: Operating Systems [Spring 2025]
School of Engineering and Technology, University of Washington - Tacoma L1.3

3



TCSS 422 – Spring 2025

- Spring 2025:
  - *In-person course*
    - Sessions of this course may be hosted online as needed due to instructor's schedule:
      Dates will be announced as soon as possible
  - Hybrid/Synchronous Format – streamed LIVE via Zoom and recorded for 24/7 availability
  - Demographics survey will poll class regarding class format for Spring 2025
    - Based on survey, adjustments may be made
    - In-class activities can be submitted asynchronously, online if necessary
- 20 class meetings
  - 1 Monday holiday in Spring 2025: May 26
- Quizzes: Thursday April 24th, May 29th @ 4:40p
- Midterm: Thursday May 8th
- Final exam ~ Thursday June 12th

TCSS 422
SPRING
2025

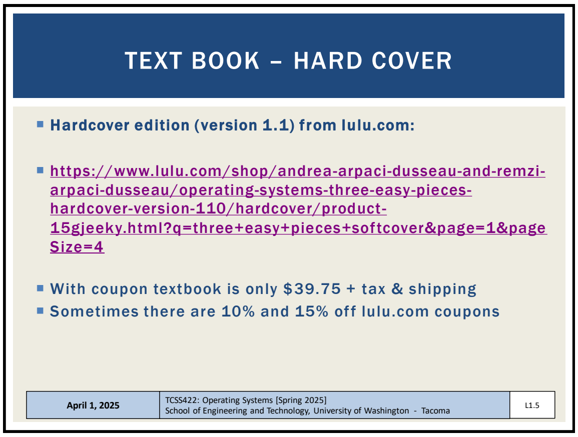
MLG 311

QUIZZES &
TESTS IN
BHS 106

TR
3:40 - 5:40 PM

L1.4

4



TEXT BOOK – HARD COVER

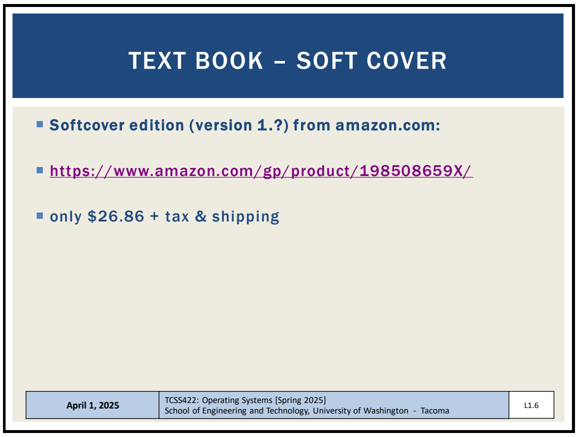
- Hardcover edition (version 1.1) from lulu.com:

- https://www.lulu.com/shop/andrea-arpaci-dusseau-and-remzi-arpaci-dusseau/operating-systems-three-easy-pieces-hardcover-version-110/hardcover/product-15gjeeky.html?q=three+easy+pieces+softcover&page=1&pageSize=4

- With coupon textbook is only $39.75 + tax & shipping
- Sometimes there are 10% and 15% off lulu.com coupons

April 1, 2025 TCSS422: Operating Systems [Spring 2025]
School of Engineering and Technology, University of Washington - Tacoma L1.5

5



TEXT BOOK – SOFT COVER

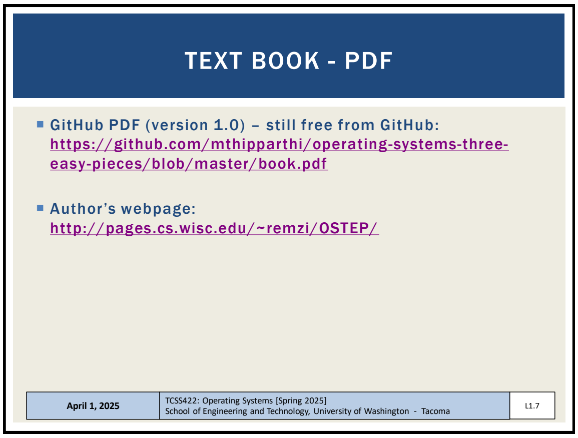
- Softcover edition (version 1.?) from amazon.com:

- https://www.amazon.com/gp/product/198508659X/

- only $26.86 + tax & shipping

April 1, 2025 TCSS422: Operating Systems [Spring 2025]
School of Engineering and Technology, University of Washington - Tacoma L1.6

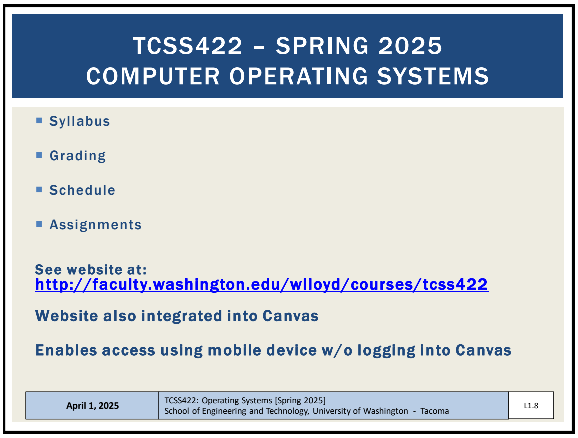
6



TEXT BOOK - PDF

- GitHub PDF (version 1.0) – still free from GitHub: https://github.com/mthipparthi/operating-systems-three-easy-pieces/blob/master/book.pdf
- Author's webpage: http://pages.cs.wisc.edu/~remzi/OSTEP/

7



TCSS422 – SPRING 2025 COMPUTER OPERATING SYSTEMS

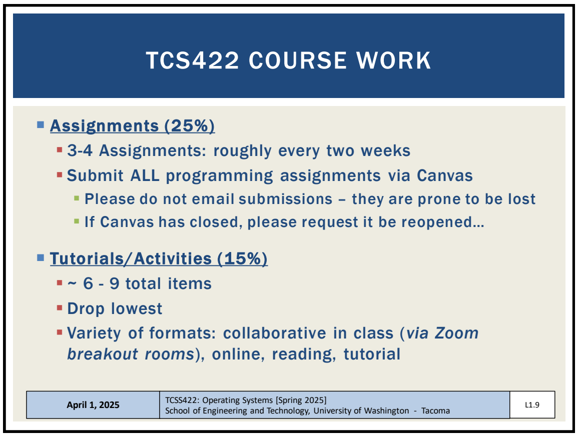
- Syllabus
- Grading
- Schedule
- Assignments

See website at:
http://faculty.washington.edu/wlloyd/courses/tcss422

Website also integrated into Canvas

Enables access using mobile device w/o logging into Canvas

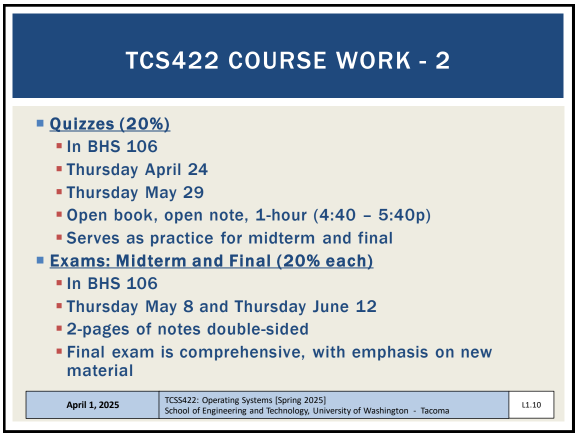
8



TCS422 COURSE WORK

- Assignments (25%)
  - 3-4 Assignments: roughly every two weeks
  - Submit ALL programming assignments via Canvas
    - Please do not email submissions – they are prone to be lost
    - If Canvas has closed, please request it be reopened...
- Tutorials/Activities (15%)
  - ~ 6 - 9 total items
  - Drop lowest
  - Variety of formats: collaborative in class (*via Zoom breakout rooms*), online, reading, tutorial

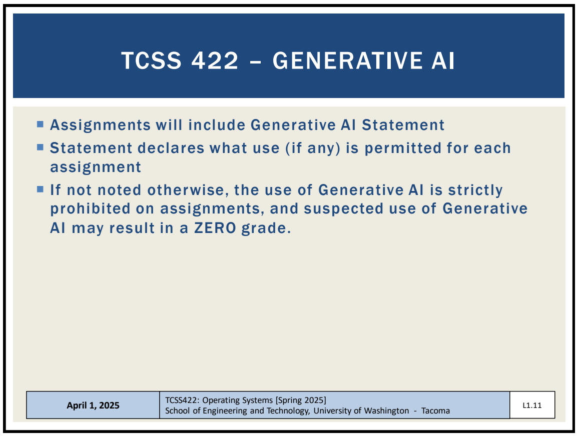
9



TCS422 COURSE WORK - 2

- Quizzes (20%)
  - In BHS 106
  - Thursday April 24
  - Thursday May 29
  - Open book, open note, 1-hour (4:40 – 5:40p)
  - Serves as practice for midterm and final
- Exams: Midterm and Final (20% each)
  - In BHS 106
  - Thursday May 8 and Thursday June 12
  - 2-pages of notes double-sided
  - Final exam is comprehensive, with emphasis on new material

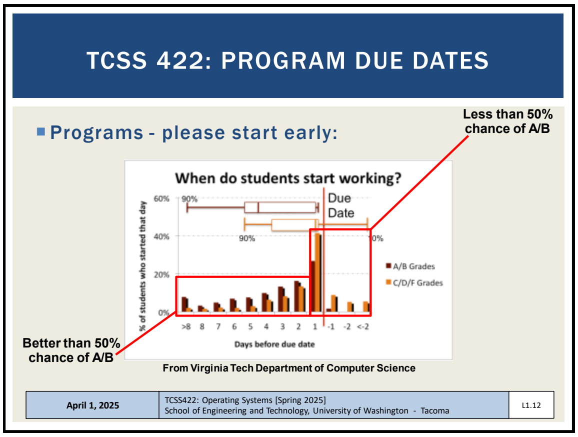
10



TCSS 422 – GENERATIVE AI

- Assignments will include Generative AI Statement
- Statement declares what use (if any) is permitted for each assignment
- If not noted otherwise, the use of Generative AI is strictly prohibited on assignments, and suspected use of Generative AI may result in a ZERO grade.

11



TCSS 422: PROGRAM DUE DATES

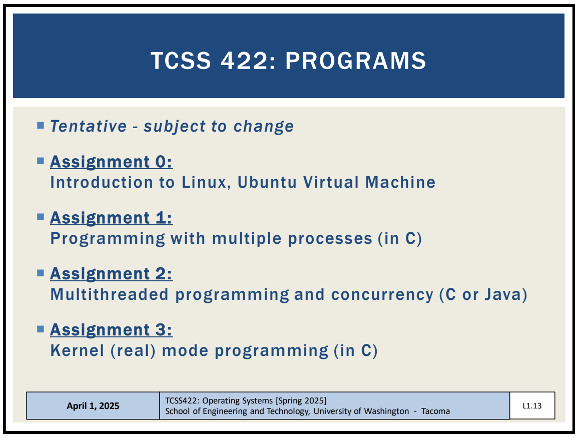
- Programs - please start early:

Less than 50% chance of A/B

Better than 50% chance of A/B

From Virginia Tech Department of Computer Science

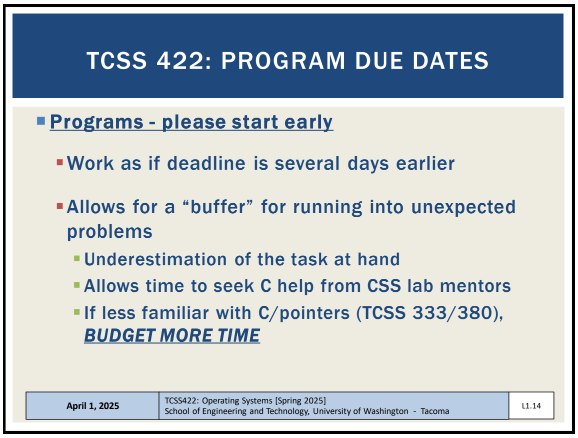
12



TCSS 422: PROGRAMS

- *Tentative - subject to change*
- **Assignment 0:**
  Introduction to Linux, Ubuntu Virtual Machine
- **Assignment 1:**
  Programming with multiple processes (in C)
- **Assignment 2:**
  Multithreaded programming and concurrency (C or Java)
- **Assignment 3:**
  Kernel (real) mode programming (in C)

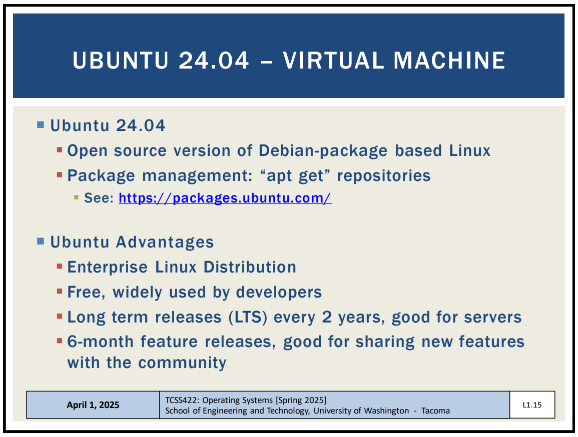
13



TCSS 422: PROGRAM DUE DATES

- **Programs - please start early**
  - Work as if deadline is several days earlier
  - Allows for a “buffer” for running into unexpected problems
    - Underestimation of the task at hand
    - Allows time to seek C help from CSS lab mentors
    - If less familiar with C/pointers (TCSS 333/380), ***BUDGET MORE TIME***

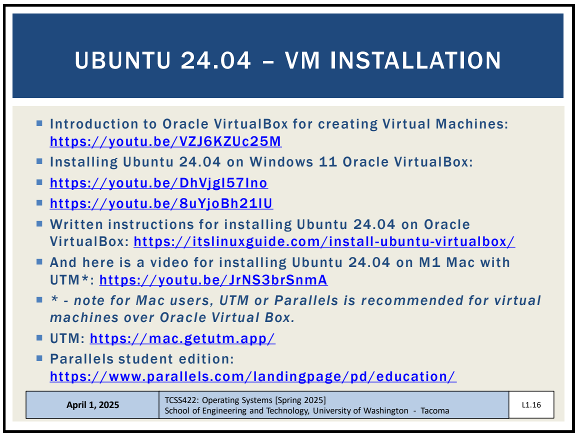
14



UBUNTU 24.04 – VIRTUAL MACHINE

- Ubuntu 24.04
  - Open source version of Debian-package based Linux
  - Package management: “apt get” repositories
    - See: https://packages.ubuntu.com/
- Ubuntu Advantages
  - Enterprise Linux Distribution
  - Free, widely used by developers
  - Long term releases (LTS) every 2 years, good for servers
  - 6-month feature releases, good for sharing new features with the community

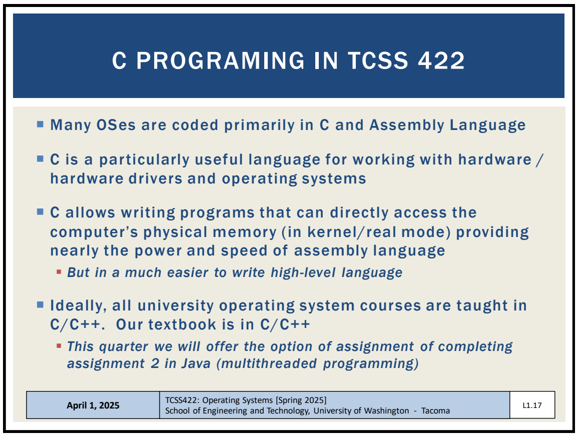
15



UBUNTU 24.04 – VM INSTALLATION

- Introduction to Oracle VirtualBox for creating Virtual Machines: https://youtu.be/VZJ6KZUc25M
- Installing Ubuntu 24.04 on Windows 11 Oracle VirtualBox:
- https://youtu.be/DhVjgI57Ino
- https://youtu.be/8uYjoBh21IU
- Written instructions for installing Ubuntu 24.04 on Oracle VirtualBox: https://itslinuxguide.com/install-ubuntu-virtualbox/
- And here is a video for installing Ubuntu 24.04 on M1 Mac with UTM*: https://youtu.be/JrNS3brSnmA
- ** - note for Mac users, UTM or Parallels is recommended for virtual machines over Oracle Virtual Box.*
- UTM: https://mac.getutm.app/
- Parallels student edition: https://www.parallels.com/landingpage/pd/education/

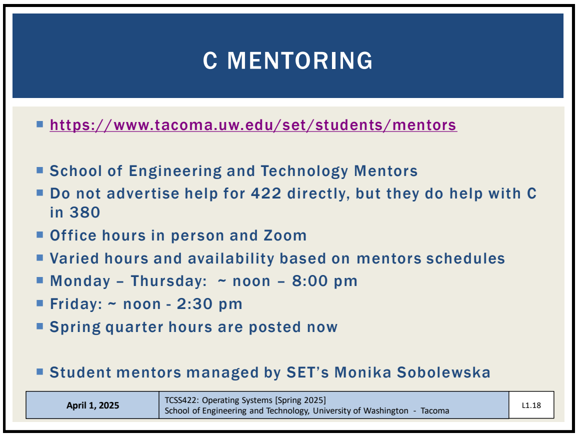
16



C PROGRAMING IN TCSS 422

- Many OSes are coded primarily in C and Assembly Language
- C is a particularly useful language for working with hardware / hardware drivers and operating systems
- C allows writing programs that can directly access the computer’s physical memory (in kernel/real mode) providing nearly the power and speed of assembly language
  - *But in a much easier to write high-level language*
- Ideally, all university operating system courses are taught in C/C++. Our textbook is in C/C++
  - *This quarter we will offer the option of assignment of completing assignment 2 in Java (multithreaded programming)*

17



C MENTORING

- https://www.tacoma.uw.edu/set/students/mentors
- School of Engineering and Technology Mentors
- Do not advertise help for 422 directly, but they do help with C in 380
- Office hours in person and Zoom
- Varied hours and availability based on mentors schedules
- Monday – Thursday: ~ noon – 8:00 pm
- Friday: ~ noon - 2:30 pm
- Spring quarter hours are posted now
- Student mentors managed by SET’s Monika Sobolewska

18

INSTRUCTOR HELP

- Office hours: TBD, after class
  - CP 229 and Zoom
  - Additional hours based on survey results
  - Also available by appointment
- Take ***ownership*** of your educational outcome
  - ~10 weeks in TCSS 422 is very small relative to entire IT career
  - Make the most of this ***limited*** opportunity
    - Maximize your educational investment
  - ****** Ask questions in class ******
  - Also questions after class, email, Canvas discussion boards
  - Seek help using UWT resources, the Internet, YouTube videos (video.google.com) and online tutorials

CLASS PARTICIPATION

- **Questions and discussion are strongly encouraged**
  - Leverage your educational investment
  - All questions are encouraged!
  - This instructor appreciates questions at all levels – there is no judgement for any question
- **Feedback surveys**
  - How much is new vs. review?
  - Checking the pace…
  - What is unclear? It's helpful to know when topics are not clear
  - Use the survey to write questions and feedback that come to you during the lecture
  - Up to 2% Extra Credit for participating in surveys
- **Poll-EV**

OBJECTIVES – 4/1

- Syllabus, Course Introduction

- **C Review Survey**
- Background Survey & VM Survey

- Chapter 2: Operating Systems – Three Easy Pieces
  - Introduction to operating systems
  - Management of resources
  - Concepts of virtualization/abstraction
  - Three Easy Pieces: CPU, Memory, I/O
  - Concurrency
  - Operating system design goals

C REVIEW SURVEY

C QUIZ – IN CANVAS

OBJECTIVES – 4/1

- Syllabus, Course Introduction

- C Review Survey
- **Background Survey & VM Survey**

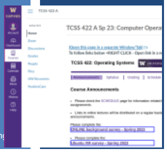
- Chapter 2: Operating Systems – Three Easy Pieces
  - Introduction to operating systems
  - Management of resources
  - Concepts of virtualization/abstraction
  - Three Easy Pieces: CPU, Memory, I/O
  - Concurrency
  - Operating system design goals

PLEASE COMPLETE BACKGROUND & VM SURVEYS

SEE LINKS AT:
http://faculty.washington.edu/wlloyd/courses/tcss422/announcements.html

or in *Canvas* under "**Announcements**"

we will resume at ~5:00 pm



TCSS 422 A

Home
Zoom
Discussions
Grades
People
Files
UW Resources
HuskiesCard

TCSS 422 A Sp 23: Computer Opera

Open this page in a separate Window/Tab
To follow links below <RIGHT CLICK – Open link in a n

TCSS 422: Operating Systems

Announcements | Syllabus | Grading | Schedule

Course Announcements

- Please check the SCHEDULE page for information related to assignments.
- Links to online lectures will be distributed on a regular basis announcements.

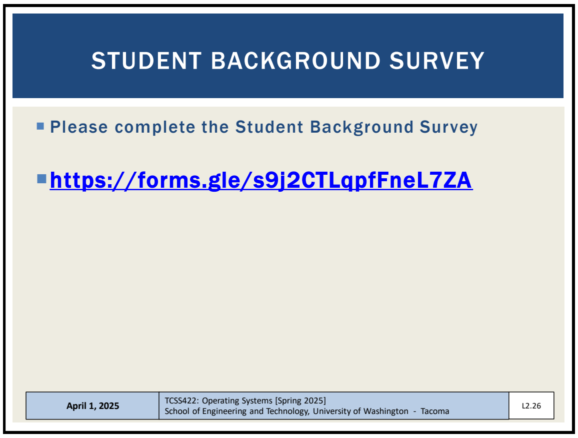
Please complete the:
ONLINE background survey – Spring 2023

- Please complete the:
Ubuntu VM survey – Spring 2023

WE WILL RETURN AT 5:00PM

April 1, 2025 TCSS422: Operating Systems [Spring 2025]
School of Engineering and Technology, University of Washington - coma L1.25

25

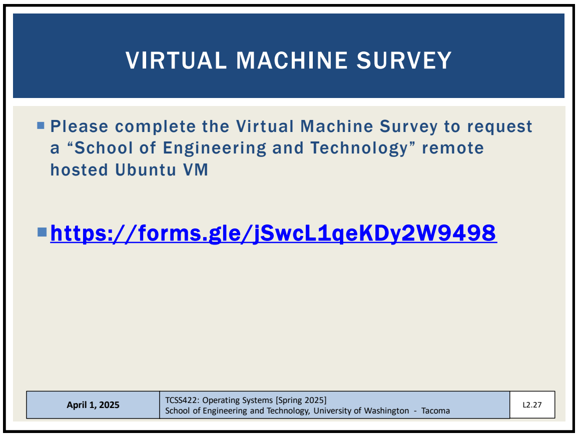


STUDENT BACKGROUND SURVEY

- Please complete the Student Background Survey
- https://forms.gle/s9j2CTLqpfFneL7ZA

April 1, 2025 TCSS422: Operating Systems [Spring 2025]
School of Engineering and Technology, University of Washington - Tacoma L2.26

26

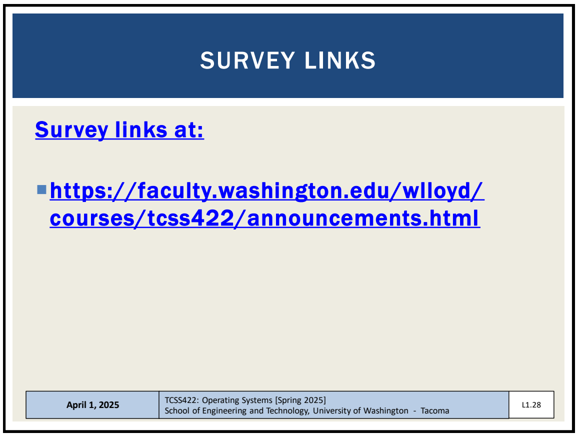


VIRTUAL MACHINE SURVEY

- Please complete the Virtual Machine Survey to request a “School of Engineering and Technology” remote hosted Ubuntu VM
- https://forms.gle/jSwcL1qeKDy2W9498

April 1, 2025 TCSS422: Operating Systems [Spring 2025]
School of Engineering and Technology, University of Washington - Tacoma L2.27

27



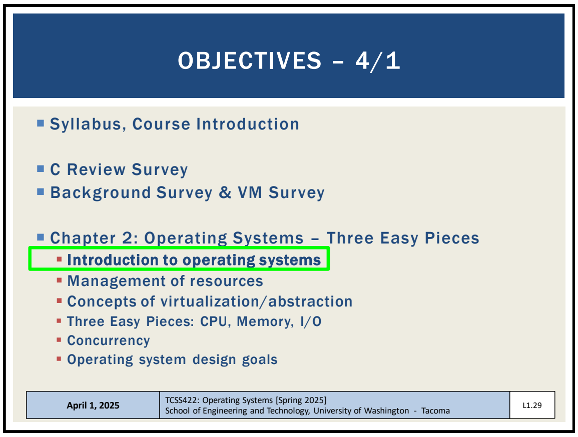
SURVEY LINKS

Survey links at:

- https://faculty.washington.edu/wlloyd/courses/tcss422/announcements.html

April 1, 2025 TCSS422: Operating Systems [Spring 2025]
School of Engineering and Technology, University of Washington - Tacoma L1.28

28

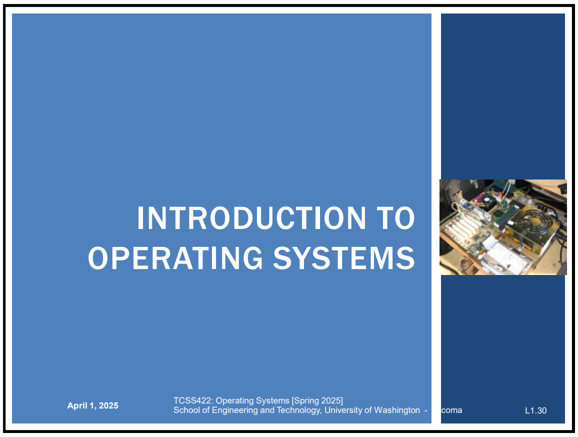


OBJECTIVES – 4/1

- Syllabus, Course Introduction
- C Review Survey
- Background Survey & VM Survey
- Chapter 2: Operating Systems – Three Easy Pieces
  - Introduction to operating systems
  - Management of resources
  - Concepts of virtualization/abstraction
  - Three Easy Pieces: CPU, Memory, I/O
  - Concurrency
  - Operating system design goals

April 1, 2025 TCSS422: Operating Systems [Spring 2025]
School of Engineering and Technology, University of Washington - Tacoma L1.29

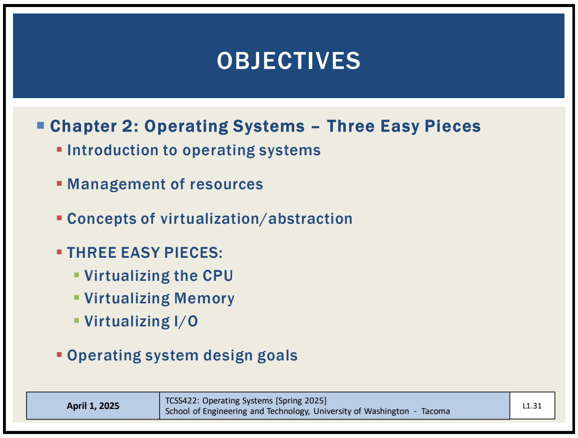
29



INTRODUCTION TO OPERATING SYSTEMS

April 1, 2025 TCSS422: Operating Systems [Spring 2025]
School of Engineering and Technology, University of Washington - coma L1.30

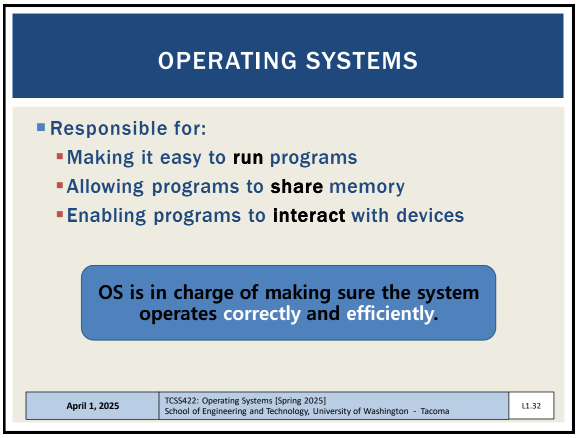
30



OBJECTIVES

- Chapter 2: Operating Systems – Three Easy Pieces
  - Introduction to operating systems
  - Management of resources
  - Concepts of virtualization/abstraction
  - THREE EASY PIECES:
    - Virtualizing the CPU
    - Virtualizing Memory
    - Virtualizing I/O
  - Operating system design goals

31

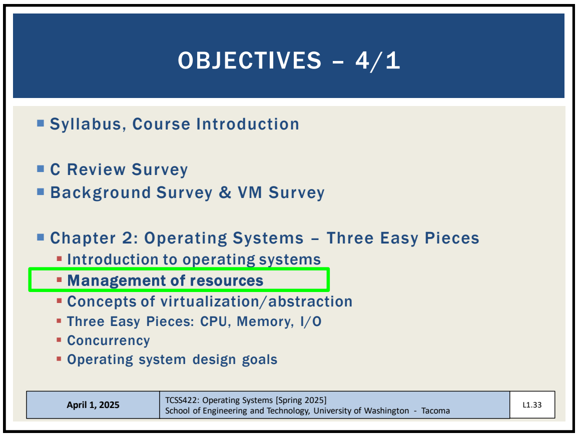


OPERATING SYSTEMS

- Responsible for:
  - Making it easy to **run** programs
  - Allowing programs to **share** memory
  - Enabling programs to **interact** with devices

OS is in charge of making sure the system operates correctly and efficiently.

32



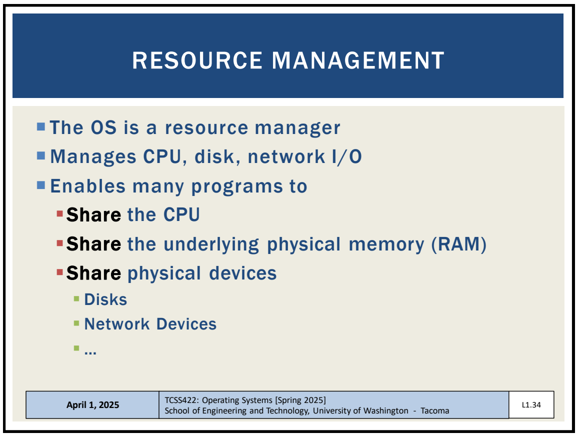
OBJECTIVES – 4/1

- Syllabus, Course Introduction

- C Review Survey
- Background Survey & VM Survey

- Chapter 2: Operating Systems – Three Easy Pieces
  - Introduction to operating systems
  - **Management of resources**
  - Concepts of virtualization/abstraction
  - Three Easy Pieces: CPU, Memory, I/O
  - Concurrency
  - Operating system design goals

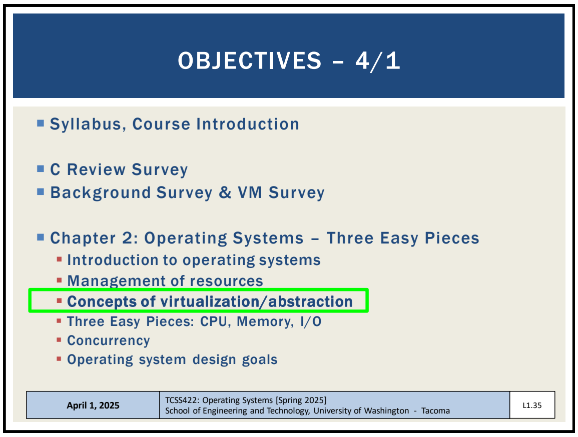
33



RESOURCE MANAGEMENT

- The OS is a resource manager
- Manages CPU, disk, network I/O
- Enables many programs to
  - **Share** the CPU
  - **Share** the underlying physical memory (RAM)
  - **Share** physical devices
    - Disks
    - Network Devices
    - …

34



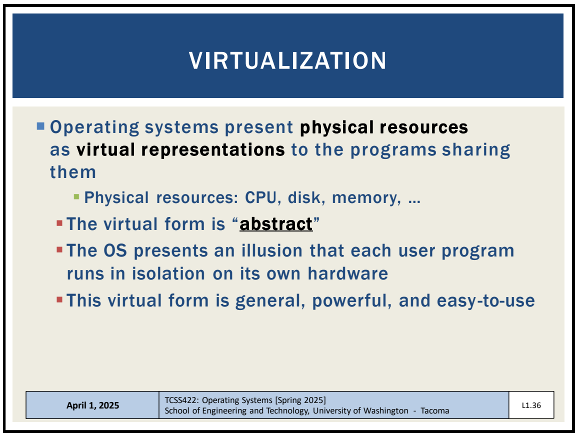
OBJECTIVES – 4/1

- Syllabus, Course Introduction

- C Review Survey
- Background Survey & VM Survey

- Chapter 2: Operating Systems – Three Easy Pieces
  - Introduction to operating systems
  - Management of resources
  - **Concepts of virtualization/abstraction**
  - Three Easy Pieces: CPU, Memory, I/O
  - Concurrency
  - Operating system design goals

35



VIRTUALIZATION

- Operating systems present **physical resources** as **virtual representations** to the programs sharing them
    - Physical resources: CPU, disk, memory, …
  - The virtual form is "**abstract**"
  - The OS presents an illusion that each user program runs in isolation on its own hardware
  - This virtual form is general, powerful, and easy-to-use

36

ABSTRACTIONS

- What form of abstraction does the OS provide?
  - CPU
    - Process and/or thread
  - Memory
    - Address space
    - → large array of bytes
    - All programs see the same "size" of RAM
  - Disk
    - Files

37

WHY ABSTRACTION?

- Allow applications to reuse common facilities
- Make different devices look the same
  - Easier to write common code to use devices
    - Linux/Unix Block Devices
- Provide higher level abstractions
- More useful functionality

38

ABSTRACTION CHALLENGES

- What level of abstraction?
  - How much of the underlying hardware should be exposed?
    - What if **too much**?
    - What if **too little**?
- What are the correct abstractions?
  - Security concerns

39

OBJECTIVES – 4/1

- Syllabus, Course Introduction

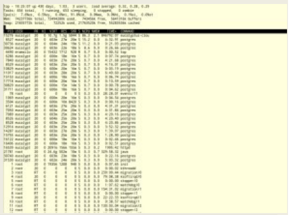
- C Review Survey
- Background Survey & VM Survey

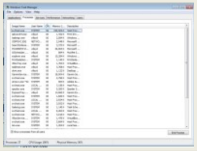
- Chapter 2: Operating Systems – Three Easy Pieces
  - Introduction to operating systems
  - Management of resources
  - Concepts of virtualization/abstraction
  - **Three Easy Pieces: CPU**, Memory, I/O
  - Concurrency
  - Operating system design goals

40

VIRTUALIZING THE CPU

- Each running program gets its own "virtual" representation of the CPU
- Many programs seem to run at once
- Linux: "top" command shows process list
- Windows: task manager





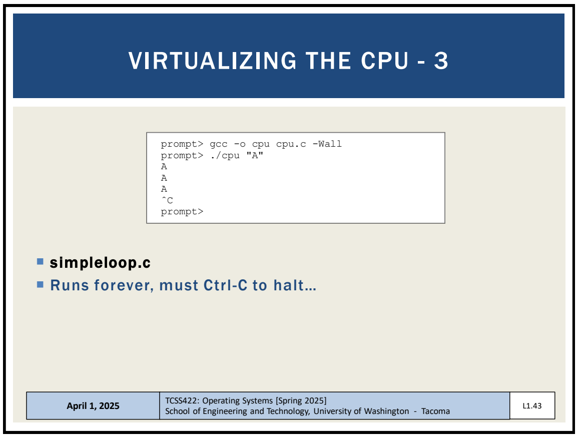
41

VIRTUALIZING THE CPU - 2

- Simple Looping C Program (**simpleloop.c**)

```c
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#include <assert.h>
#include "common.h"

int
main(int argc, char *argv[])
{
        if (argc != 2) {
                fprintf(stderr, "usage: cpu <string>\n");
                exit(1);
        }
        char *str = argv[1];
        while (1) {
                Spin(1); // Repeatedly checks the time and
                returns once it has run for a second
                printf("%s\n", str);
        }
        return 0;
}
```

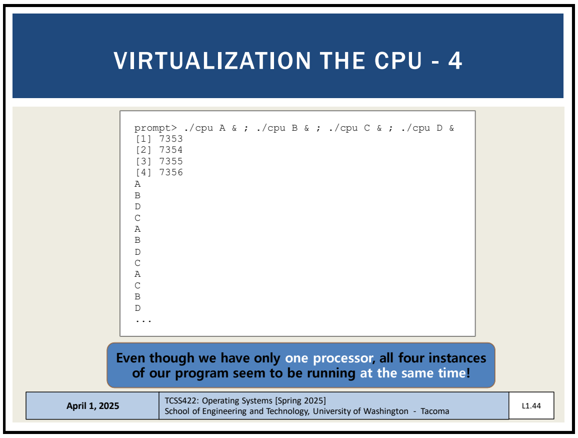
42



VIRTUALIZING THE CPU - 3

```
prompt> gcc –o cpu cpu.c -Wall
prompt> ./cpu "A"
A
A
A
ˆC
prompt>
```

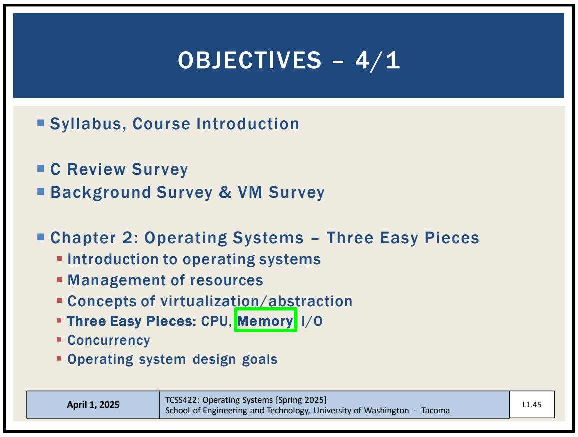
- **simpleloop.c**
- Runs forever, must Ctrl-C to halt…



VIRTUALIZATION THE CPU - 4

```
prompt> ./cpu A & ; ./cpu B & ; ./cpu C & ; ./cpu D &
[1] 7353
[2] 7354
[3] 7355
[4] 7356
A
B
D
C
A
B
D
C
A
C
B
D
...
```

Even though we have only one processor, all four instances of our program seem to be running at the same time!

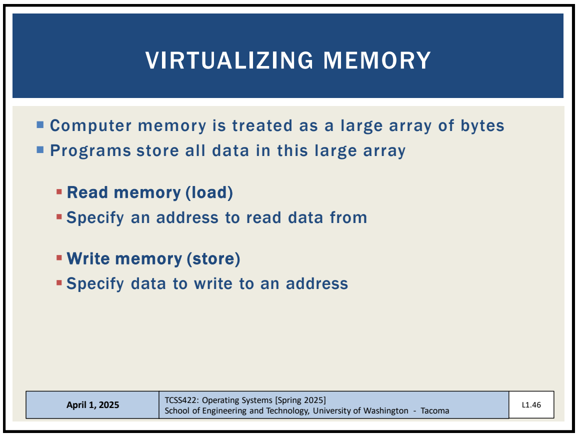


OBJECTIVES – 4/1

- Syllabus, Course Introduction

- C Review Survey
- Background Survey & VM Survey

- Chapter 2: Operating Systems – Three Easy Pieces
  - Introduction to operating systems
  - Management of resources
  - Concepts of virtualization/abstraction
  - **Three Easy Pieces:** CPU, **Memory**, I/O
  - Concurrency
  - Operating system design goals

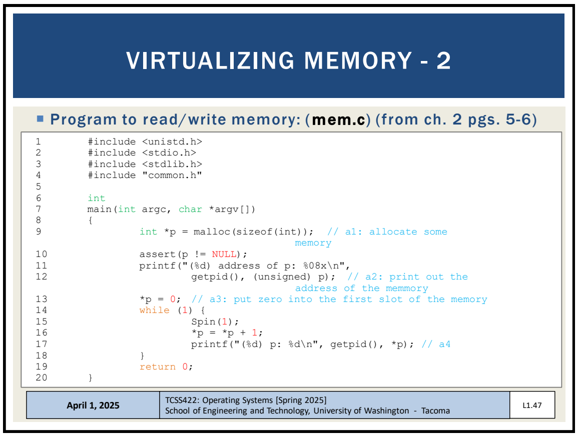


VIRTUALIZING MEMORY

- Computer memory is treated as a large array of bytes
- Programs store all data in this large array

  - **Read memory (load)**
  - Specify an address to read data from

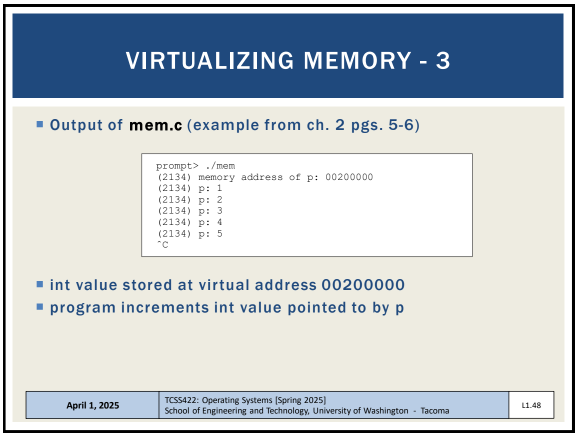
  - **Write memory (store)**
  - Specify data to write to an address



VIRTUALIZING MEMORY - 2

- Program to read/write memory: (**mem.c**) (from ch. 2 pgs. 5-6)

```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include "common.h"

int
main(int argc, char *argv[])
{
        int *p = malloc(sizeof(int));  // a1: allocate some
                                  memory
        assert(p != NULL);
        printf("(%d) address of p: %08x\n",
                getpid(), (unsigned) p);  // a2: print out the
                                  address of the memmory
        *p = 0;  // a3: put zero into the first slot of the memory
        while (1) {
                Spin(1);
                *p = *p + 1;
                printf("(%d) p: %d\n", getpid(), *p); // a4
        }
        return 0;
}
```

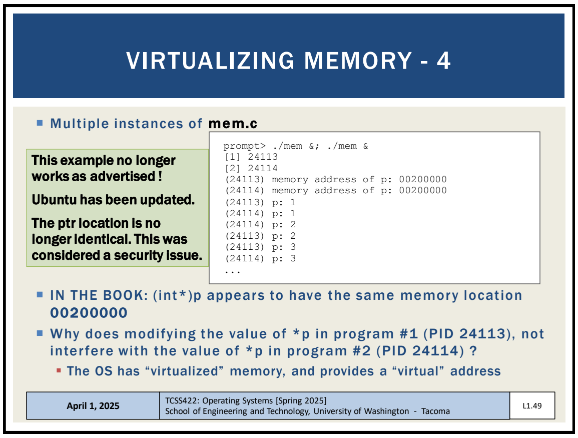


VIRTUALIZING MEMORY - 3

- Output of **mem.c** (example from ch. 2 pgs. 5-6)

```
prompt> ./mem
(2134) memory address of p: 00200000
(2134) p: 1
(2134) p: 2
(2134) p: 3
(2134) p: 4
(2134) p: 5
ˆC
```

- int value stored at virtual address 00200000
- program increments int value pointed to by p



VIRTUALIZING MEMORY - 4

- Multiple instances of **mem.c**

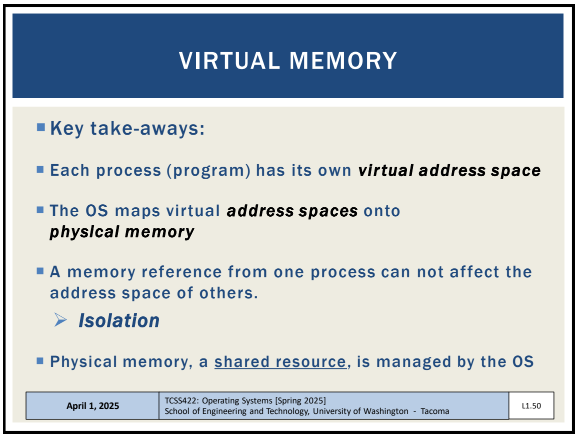
This example no longer works as advertised !

Ubuntu has been updated.

The ptr location is no longer identical. This was considered a security issue.

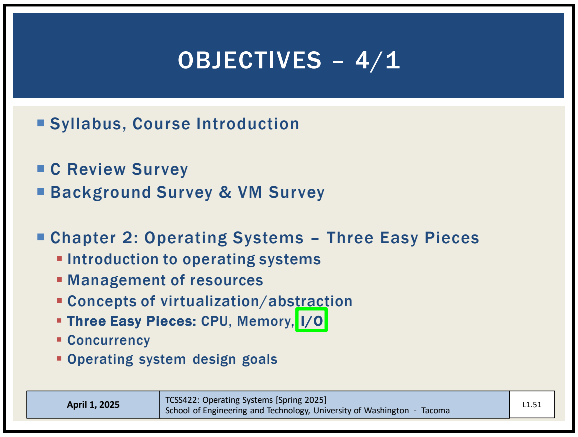
```
prompt> ./mem &; ./mem &
[1] 24113
[2] 24114
(24113) memory address of p: 00200000
(24114) memory address of p: 00200000
(24113) p: 1
(24114) p: 1
(24114) p: 2
(24113) p: 2
(24113) p: 3
(24114) p: 3
...
```

- IN THE BOOK: (int*)p appears to have the same memory location **00200000**
- Why does modifying the value of *p in program #1 (PID 24113), not interfere with the value of *p in program #2 (PID 24114) ?
  - The OS has "virtualized" memory, and provides a "virtual" address



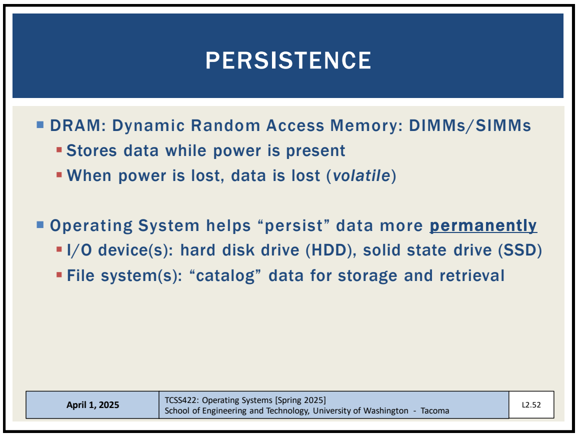
VIRTUAL MEMORY

- Key take-aways:
- Each process (program) has its own ***virtual address space***
- The OS maps virtual ***address spaces*** onto ***physical memory***
- A memory reference from one process can not affect the address space of others.
  - ***Isolation***
- Physical memory, a <u>shared resource</u>, is managed by the OS



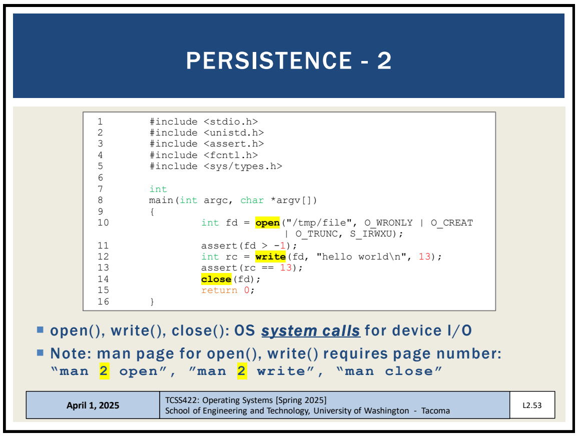
OBJECTIVES – 4/1

- Syllabus, Course Introduction
- C Review Survey
- Background Survey & VM Survey
- Chapter 2: Operating Systems – Three Easy Pieces
  - Introduction to operating systems
  - Management of resources
  - Concepts of virtualization/abstraction
  - **Three Easy Pieces:** CPU, Memory, **I/O**
  - Concurrency
  - Operating system design goals



PERSISTENCE

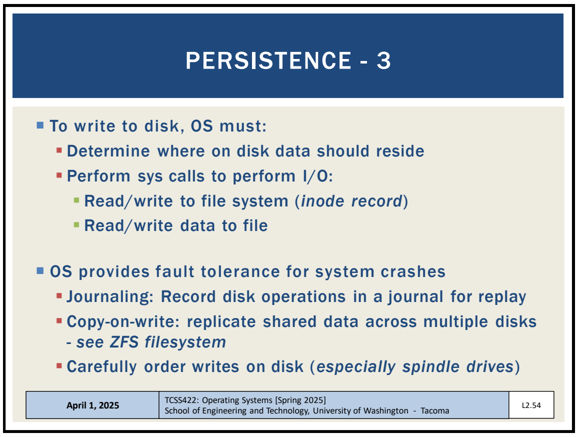
- DRAM: Dynamic Random Access Memory: DIMMs/SIMMs
  - Stores data while power is present
  - When power is lost, data is lost (*volatile*)
- Operating System helps "persist" data more **<u>permanently</u>**
  - I/O device(s): hard disk drive (HDD), solid state drive (SSD)
  - File system(s): "catalog" data for storage and retrieval



PERSISTENCE - 2

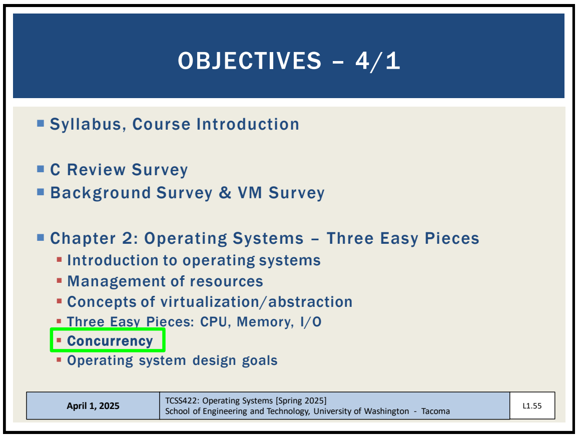
```
#include <stdio.h>
#include <unistd.h>
#include <assert.h>
#include <fcntl.h>
#include <sys/types.h>

int
main(int argc, char *argv[])
{
        int fd = open("/tmp/file", O_WRONLY | O_CREAT
                         | O_TRUNC, S_IRWXU);
        assert(fd > -1);
        int rc = write(fd, "hello world\n", 13);
        assert(rc == 13);
        close(fd);
        return 0;
}
```

- open(), write(), close(): OS ***<u>system calls</u>*** for device I/O
- Note: man page for open(), write() requires page number: "man 2 open", "man 2 write", "man close"



PERSISTENCE - 3

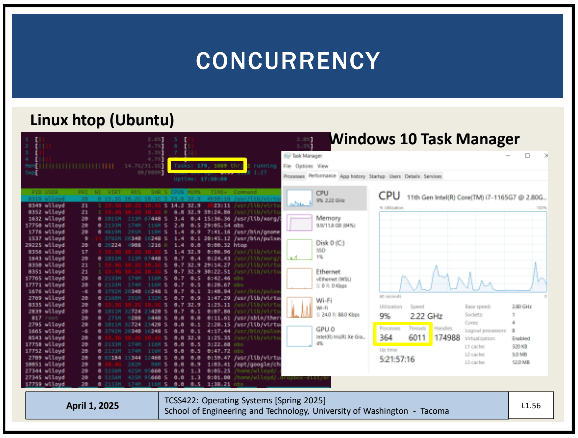
- To write to disk, OS must:
  - Determine where on disk data should reside
  - Perform sys calls to perform I/O:
    - Read/write to file system (*inode record*)
    - Read/write data to file
- OS provides fault tolerance for system crashes
  - Journaling: Record disk operations in a journal for replay
  - Copy-on-write: replicate shared data across multiple disks - *see ZFS filesystem*
  - Carefully order writes on disk (*especially spindle drives*)



OBJECTIVES – 4/1

- Syllabus, Course Introduction
- C Review Survey
- Background Survey & VM Survey
- Chapter 2: Operating Systems – Three Easy Pieces
  - Introduction to operating systems
  - Management of resources
  - Concepts of virtualization/abstraction
  - Three Easy Pieces: CPU, Memory, I/O
  - **Concurrency**
  - Operating system design goals

55



CONCURRENCY

Linux htop (Ubuntu)

Tasks: 179, 1089 thr

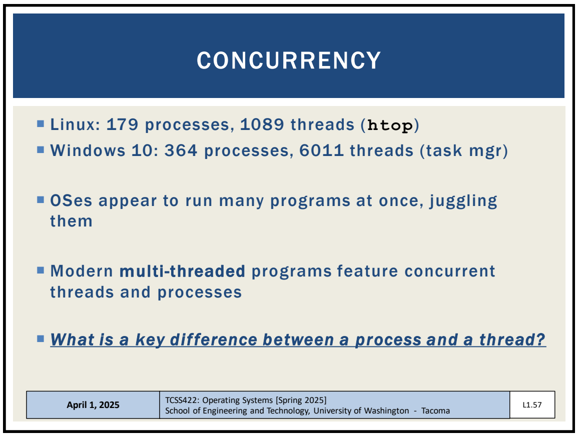
Windows 10 Task Manager

Task Manager
File Options View
Processes Performance App history Startup Users Details Services
CPU 9% 2.22 GHz
Memory 9.9/11.8 GB (84%)
Disk 0 (C:) SSD 1%
Ethernet vEthernet (WSL) S: 0 R: 0 Kbps
Wi-Fi Wi-Fi S: 24.0 R: 88.0 Kbps
GPU 0 Intel(R) Iris(R) Xe Gra... 4%
% Utilization
100%
60 seconds
0

CPU 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80G...

Utilization 9%
Speed 2.22 GHz
Processes 364
Threads 6011
Handles 174988
Up time 5:21:57:16
Base speed: 2.80 GHz
Sockets: 1
Cores: 4
Logical processors: 8
Virtualization: Enabled
L1 cache: 320 KB
L2 cache: 5.0 MB
L3 cache: 12.0 MB

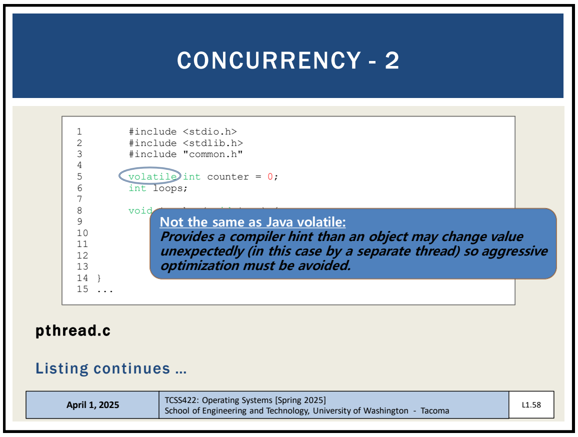
56



CONCURRENCY

- Linux: 179 processes, 1089 threads (`htop`)
- Windows 10: 364 processes, 6011 threads (task mgr)
- OSes appear to run many programs at once, juggling them
- Modern **multi-threaded** programs feature concurrent threads and processes
- ***What is a key difference between a process and a thread?***

57



CONCURRENCY - 2

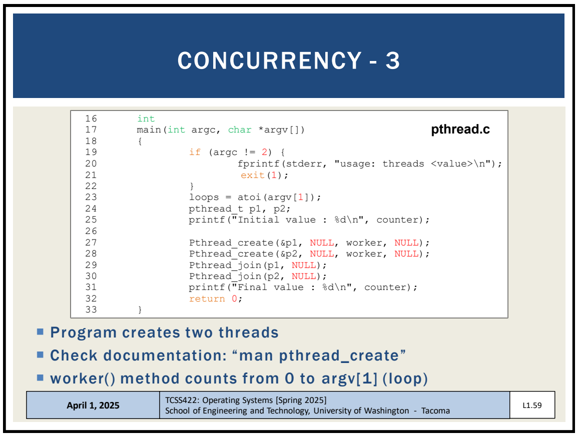
```
#include <stdio.h>
#include <stdlib.h>
#include "common.h"

volatile int counter = 0;
int loops;

void






}
...
```

Not the same as Java volatile:
Provides a compiler hint than an object may change value unexpectedly (in this case by a separate thread) so aggressive optimization must be avoided.

pthread.c

Listing continues …

58



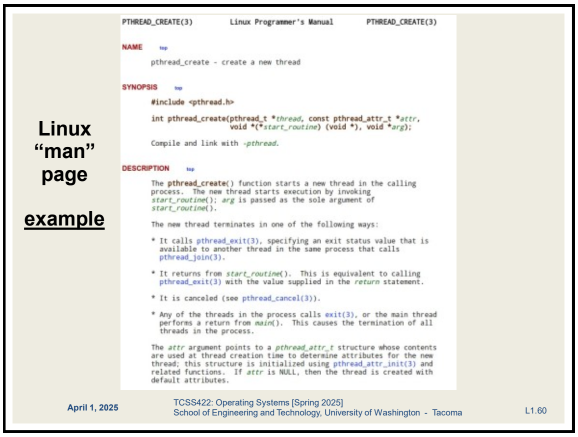
CONCURRENCY - 3

pthread.c

```
int
main(int argc, char *argv[])
{
        if (argc != 2) {
                fprintf(stderr, "usage: threads <value>\n");
                exit(1);
        }
        loops = atoi(argv[1]);
        pthread_t p1, p2;
        printf("Initial value : %d\n", counter);

        Pthread_create(&p1, NULL, worker, NULL);
        Pthread_create(&p2, NULL, worker, NULL);
        Pthread_join(p1, NULL);
        Pthread_join(p2, NULL);
        printf("Final value : %d\n", counter);
        return 0;
}
```

- Program creates two threads
- Check documentation: “man pthread_create”
- worker() method counts from 0 to argv[1] (loop)

59



Linux “man” page

example

```
PTHREAD_CREATE(3)          Linux Programmer's Manual          PTHREAD_CREATE(3)

NAME    top

       pthread_create - create a new thread

SYNOPSIS    top

       #include <pthread.h>

       int pthread_create(pthread_t *thread, const pthread_attr_t *attr,
                          void *(*start_routine) (void *), void *arg);

       Compile and link with -pthread.

DESCRIPTION    top

       The pthread_create() function starts a new thread in the calling
       process.  The new thread starts execution by invoking
       start_routine(); arg is passed as the sole argument of
       start_routine().

       The new thread terminates in one of the following ways:

       * It calls pthread_exit(3), specifying an exit status value that is
         available to another thread in the same process that calls
         pthread_join(3).

       * It returns from start_routine().  This is equivalent to calling
         pthread_exit(3) with the value supplied in the return statement.

       * It is canceled (see pthread_cancel(3)).

       * Any of the threads in the process calls exit(3), or the main thread
         performs a return from main().  This causes the termination of all
         threads in the process.

       The attr argument points to a pthread_attr_t structure whose contents
       are used at thread creation time to determine attributes for the new
       thread; this structure is initialized using pthread_attr_init(3) and
       related functions.  If attr is NULL, then the thread is created with
       default attributes.
```

60

CONCURRENCY - 4

- Command line parameter argv[1] provides loop length
- Defines number of times the shared counter is incremented

- Loops: 1000

```
prompt> gcc -o pthread pthread.c -Wall -pthread
prompt> ./pthread 1000
Initial value : 0
Final value : 2000
```

- Loops 100000

```
prompt> ./pthread 100000
Initial value : 0
Final value : 143012 // huh??
prompt> ./pthread 100000
Initial value : 0
Final value : 137298 // what ???
```

April 1, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L1.61

61

CONCURRENCY - 5

- When loop value is large why do we not achieve 200,000 ?

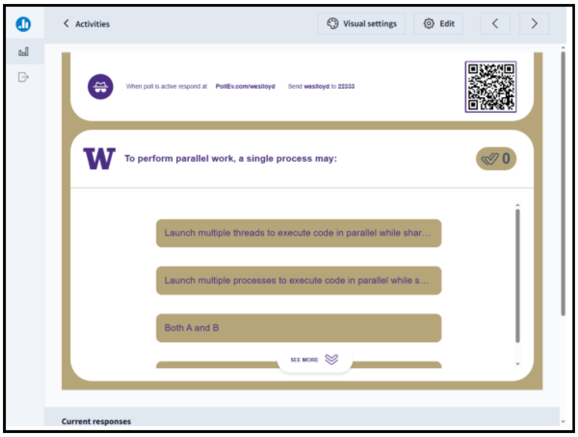
- C code is translated to (3) assembly code operations

1. Load counter variable into register
2. Increment it
3. Store the register value back in memory

- These instructions happen concurrently and VERY FAST
- (P1 || P2) write incremented register values back to memory, While (P1 || P2) read same memory
- Memory access here is **unsynchronized** (**non-atomic**)
- *Some of the increments are lost*

April 1, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L1.62

62



When poll is active respond at PollEv.com/weslloyd Send weslloyd to 22333

To perform parallel work, a single process may:

0

Launch multiple threads to execute code in parallel while shar...

Launch multiple processes to execute code in parallel while s...

Both A and B

63

PARALLEL PROGRAMMING

- To perform parallel work, a single process may:

- A. Launch multiple threads to execute code in parallel while sharing global data in memory

- B. Launch multiple processes to execute code in parallel without sharing global data in memory

- C. Both A and B

- D. None of the above

April 1, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L1.64

64

OBJECTIVES – 4/1

- Syllabus, Course Introduction

- C Review Survey
- Background Survey & VM Survey

- Chapter 2: Operating Systems – Three Easy Pieces
  - Introduction to operating systems
  - Management of resources
  - Concepts of virtualization/abstraction
  - Three Easy Pieces: CPU, Memory, I/O
  - Concurrency
  - **Operating system design goals**

April 1, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L1.65

65

SUMMARY: OPERATING SYSTEM DESIGN GOALS

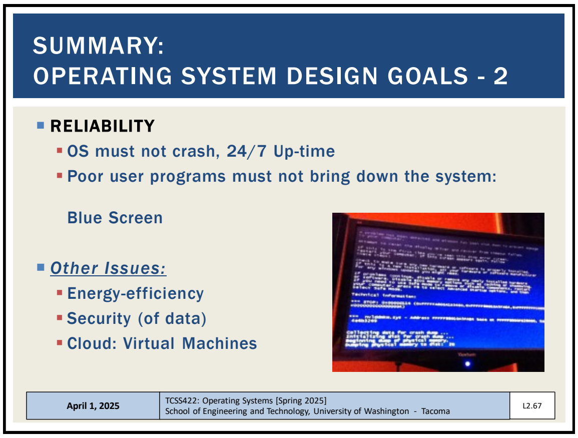
- **ABSTRACTING THE HARDWARE**
  - Makes programming code easier to write
  - Automate sharing resources – save programmer burden

- **PROVIDE HIGH PERFORMANCE**
  - Minimize overhead from OS abstraction (Virtualization of CPU, RAM, I/O)
  - Share resources fairly
  - Attempt to tradeoff performance vs. fairness → consider priority

- **PROVIDE ISOLATION**
  - User programs can't interfere with each other's virtual machines, the underlying OS, or the sharing of resources

April 1, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L2.66

66



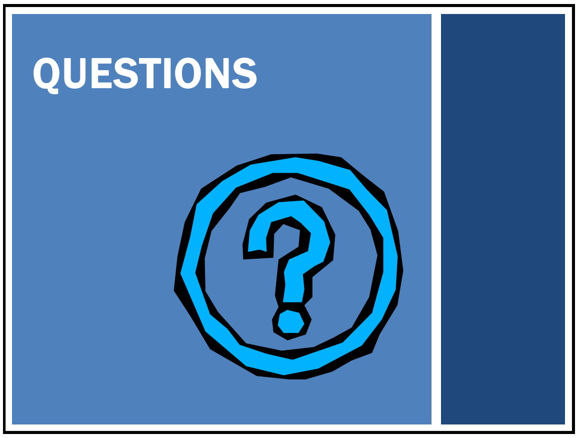
SUMMARY: OPERATING SYSTEM DESIGN GOALS - 2

- **RELIABILITY**
  - OS must not crash, 24/7 Up-time
  - Poor user programs must not bring down the system:

    Blue Screen
- ***Other Issues:***
  - Energy-efficiency
  - Security (of data)
  - Cloud: Virtual Machines

April 1, 2025 | TCSS422: Operating Systems [Spring 2025]
School of Engineering and Technology, University of Washington - Tacoma | L2.67

67



QUESTIONS

68