CompSci 421 Operating Systems **Spring'15**

Instructor: George Georgiev

Office: HS, Room 217

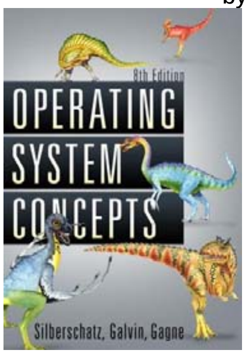
Office Hours: MWF 11:30 – 12:30 , or by appointment

Phone: (920) 424 - 11 80 **E-mail:** georgiev@uwosh.edu

Lectures: MW 8:00 - 9:00 , HS 212
Labs: F 8:00 – 9:00 , HS 101C

Required Text: Operating System Concepts, 8th Edition

by Abraham Silberschatz, Peter Baer Galvin, Greg Gagne John Wiley & Sons.



Web site for the course: http://www.uwosh.edu/faculty_staff/georgiev/subjects/CSC421/

<u>Current Catalog Description</u>

An introduction to operating systems concepts. Topics covered include: interrupts, memory allocation, virtual memory techniques, process scheduling and synchronization, deadlocks, resource allocation, and file systems. A major programming project will be assigned to provide experience with operating system design.
Prerequisite: Computer Science 271 and Mathematics 212 each with a grade of C or better.
(Spring) Required.

<u>Course Outcomes</u>

Outcomes #1

1a. Demonstrate by explaining or describing objectives and functions of modern operating systems
1b. Demonstrate sound knowledge of the user and system views of the operating system and able to differentiate between these two viewpoints.
1c. Adequately explain the structure and mode of operations of today's operating system as they relate to such concepts as multiprogramming, time-sharing, swapping, interrupt, and dual mode operation.
1d. Demonstrate intimate knowledge of the issues of design and implementation as well as the desired user and system properties.
1e. Show understanding of the influences of security, distributed and special-purpose systems, and different computing environments on modern operating system design.
1 f. Demonstrate the ability to discuss the tradeoffs inherent in operating system design and able to identify potential threats and their safeguards in designed systems.

Outcomes #2

2a. Demonstrate ability to compare and contrast the many possible methods (e.g., simple, layered, modular, and microkernel) of structuring the operating system.
2b. Show sound knowledge of the difference between asynchronous and synchronous interrupts and of the relative advantages of interrupts over polling.
2c. Able to discuss the system call concept and able to differentiate between the needs for the system call interface and the application program interface.
2d. Show how system programs manage the resources used by applications.
2e. Demonstrate the ability to illustratively explain why the services provided to users constitute a different set of functions than that provided to the system.

Outcomes #3

3a. Able to explain process state and process control block using appropriate examples to highlight their components.

3b. Able to illustratively describe the creation and termination, scheduling, and interprocess communication features of a process.

3c. Show the ability to compare and contrast the various types of multithreading models.

3d. Demonstrate a deep understanding of some of the issues encountered with multithreaded programs (e.g., fork () and exec() system calls, cancellation, signal handling, thread pools, thread specific data, and scheduler activations).

3e. Able to describe possible run-time problems arising from the concurrent operation of many separate tasks.

3f. Demonstrate the ability to explain how software and hardware are individually used to solve mutual exclusion problem.

3g. Able to discuss the evaluation criteria for selecting a particular system's CPU scheduling algorithm and the various available criteria comparing CPU scheduling algorithms.

Outcomes #4

4a. Able to compare and contrast paging and segmentation techniques.

4b. Know how to evaluate the tradeoffs of the components parts in the memory hierarchy in terms of size, cost, and access time.

4c. Able to compare and contrast demand paging and copy on write techniques.

4d. Show the ability to analyze such techniques or policies as page replacement, swapping, and thrashing.

Outcomes #5

5a. Able to discuss file system design tradeoffs including access methods, file sharing, file locking, directory structures, and protection.

5b. Demonstrate sound knowledge and deep understanding of a file's attributes and operations.

5c. Demonstrate a deep understanding of the details of a local file system and directory structure implementation.

Program Learning Outcomes Addressed:

Major Topics Covered in the Course

- Introduction to operating systems - definitions, simple batch systems, multiprogramming batched systems, mainframe systems, desktop systems, multiprocessor systems, distributed systems, clustered systems, real-time systems, handheld systems. Computing environments (2 weeks).
- Computer system structure, interrupts, system calls, I/O - computer system operation, interrupts and system calls, I/O structure (busy waiting, interrupt driven, direct memory access), storage structure, storage hierarchy, hardware protection, general system architecture (2 weeks).
- Operating system structure - system components, OS services, system calls, system programs, system structure, virtual machines, system design and implementation, system generation (1 week).
- Processes - process concept, process scheduling, operation on processes, cooperating processes, interprocess communication, communication in client-server system (2 weeks).
- Threads - multithreading models, pthreads, Solaris 2 threads, Windows threads, Linux threads, Java threads (1 week).
- CPU scheduling - scheduling criteria, scheduling algorithms, multiple processor scheduling, real-time scheduling, algorithm evaluation (1 week).
- Process synchronization - the critical section problem, synchronization hardware, semaphores, classical problems of synchronization, deadlocks, starvation, synchronization in contemporary OS (2 weeks).
- Memory management - swapping, logical versus physical addressing space, contiguous allocation, paging, segmentation, segmentation with paging (2 weeks).
- Virtual memory - demand paging, process creation, page replacement, allocation of frames, thrashing, OS examples (1 week).

Topics for interprocess communication (IPC) in UNIX programming

- UNIX operating system, basic modules, basic services, help manual, interruption handling, kernel (1 week).
- Implementing C program in UNIX environment (1 week).
- Dual mode of operation, library functions, system calls - main groups (1 week).
- Process creation, hierarchy, states, address space, synchronization, switching to another process (1 week).
- Threads - basic features, creation, synchronization (1 week).
- Fundamental system calls for files in UNIX, open(), read(), write(), dup(), synchronization, buffering (1 week).
- PC via network – TCP/IP and UDP sockets, IP and port addressing, system calls for sockets, client/server model (1 week).
- Mechanisms for IPC in UNIX - unnamed pipes, creation, using, and communicating related processes via pipe, basic features of unnamed pipes, synchronization features of pipes (1 week).
- Named pipes - client/server model for communicating two unrelated process via named pipe (1 week).
- A group of IPC in UNIX - message queue, system calls for message queue, client/server model using message queue (1 week).
- Shared memory as a mechanism for IPC - operations, system calls, example (1 week).
- Semaphores - critical section problem, semaphores as an IPC mechanism in UNIX, operations, system calls (1 week).
- Distributed computing – RPC (Remote Procedure Call). (1 week).
- Java threads – basic features, creation, synchronization (monitors, condition variables) (1 week).

Course Requirements:

There will be three exams, unannounced quizzes, laboratory assignments and programming assignments. The material for all exams will come from either a material covered in class, homework problems, lab work, and/or assignment reading.

Complete all required work on time. In the event that an exam must be missed, or required work can not be completed on time, due to illness or other serious and unavoidable circumstance, notify the professor as far in advance as possible by phone or e-mail.

You are encouraged to work in groups on your assignments. You can discuss assigned problems with other people or groups, but you must individually design and write your own solutions/code for all exams, and assignments. Submitting modified versions of other people's or group's work as your own is considered cheating.

The programming assignments are due by the beginning of the class on the due date (electronic copy e-mail, and a paper (hard) copy of the assignment is due at the beginning of class). Programs will be accepted up to three days late subject to the following penalties:

Turned in	Penalty
After class on the due date	10%
1 day late	25%
2 days late	50%
3 days late	75%

Saturdays, Sundays, and holidays count when computing penalties.

If you work in a team, you will submit one electronic copy and one paper copy of the assignment with names on it and percent of participation (e.g. equal). Partners will earn equal scores on the assignment. You may work alone on some assignments and in a team on others. You may change the team during the semester.

Laboratory assignments will be in the teaching lab. The materials will be placed on D2L. You are encouraged to discuss the lab assignment with others before and during the lab hours, but each student must demonstrate her or his own solution.

There will be no make up for unannounced quizzes.

There will be one make up for the exams, which will cover all topics. It will be at the end of the semester.

Make up will be given if you call before the exam, make arrangements, have a medical certificate signed by the physician, and have a note from the Dean of Students Office.

The three exams will be announced at least a week before taking place.

Evaluation:

Three Exams (20% each):	~60%
Programming Assignments (equal weight):	~20%
Unannounced quizzes and class work (equal weight):	~10%
Laboratory Assignments (equal weight):	~10%

Grading:

Score	Grade
>= 92	A
90-92	A-
88-90	B+
82-88	B
80-82	B-
78-80	C+
72-78	C
70-72	C-
68-70	D+
62-68	D
60-62	D-
< 60	F

Feedback:

Your comments and questions about all aspects of the course (content, grading, teaching methods, pace, textbook, etc) are welcome. You can use e-mail or talk to me during office hours.