CMPE 343 Systems Programming

Department: Computer Engineering

Instructor Information
Name: Assoc. Prof. Dr. Gurcu Oz
E-mail: gurcu.oz@emu.edu.tr
Office: CMPE 220
Office Tel: 1054

Assistant Information

Meeting times and places
Tuesday 8:30-10:20, Room CMPE 126
Thursday 8:30-10:20, Room CMPE 126
Monday 16:30-18:20, Lab 134

Program Name: Computer Engineering	**Program Code:** 25

Course Code	**Credits**	**Year/Semester**
CMPE343	4 Cr	2018-2019 Fall

☒ Required Course ☐ Elective Course (click on and check the appropriate box)

Prerequisite(s)
CMPE242 Operating Systems

Catalog Description
Systems programming in an OS environment. UNIX and the objectives of systems programming in UNIX. A program in the UNIX environment. Command line parameters. System calls and their classification. System calls for interprocess communication and for networking programming. Processes as fundamental objects in UNIX. Creating a process. Process identifier. Parent process identifier. Child process identifier. Basic concepts of threads and multithreaded programming. Interprocess communication (IPC), its purpose and using in systems programs. Mechanisms of interprocess communication in UNIX. Importance of interprocess communication for computer networks and distributed systems. Unnamed and named pipes for interprocess communication. Message queues, shared memory, signals and semaphores. Sockets and their using for interprocess communication in computer networks. A client-server paradigm of interprocess communication. The client-server model and its implementation with sockets. Using IP addresses and port numbers with sockets. TCP and UDP sockets for communication in networks. Organization of a Web client-server network system. Remote procedure call (RPC) for networks, its operation and parameter passing. A survey of systems programming aspects in Windows operating systems.

Course Web Page
http://cmpe.emu.edu.tr/courses/cmpe343

Textbook(s)
(**) Haviland, K. et al., UNIX System Programming, 2nd ed., Addison-Wesley, 1999. ISBN-10: 0201877589
(***) Molay, B., Understanding Unix/Linux Programming: A guide to Theory and Practice, Prentice-Hall, 2003. ISBN: 9780130083968 (paperback)
(*) Curry, D.A., UNIX Systems Programming for SVR4, O'Reilly & Associates, 1996.

Indicative Basic Reading List
Gray, J.S., Interprocess Communication in UNIX: The Nooks & Crannies, Prentice-Hall, 1997.
Vahalia, U., UNIX Internals: The New Frontiers, Prentice-Hall, 1995.
Bloomer, J., Power Programming with RPC, O'Reilly & Associates, 1992.
Hart, J.M., Win32 System Programming, Addison-Wesley, 1997.

Topics Covered and Class Schedule
(4 hours of lectures per week)

Weeks 1-2	Scopes of systems programming. Development a program in UNIX. Command line parameters and their use. System calls in UNIX, their classification and implementation in UNIX. A general system call interface.
Weeks 3-5	Processes as fundamental dynamic objects in UNIX. Creation of processes. Parallel running of processes. State diagram of a process. System calls for processes: fork(), system(), exec(), wait().

Week 6	Files and directories in UNIX. System calls for files and their use for creation and accessing files. Relationship between a parent process and its child processes.
Week 7	Basic concepts of threads and multithreaded processes. System calls for threads in UNIX.
Weeks 8-9	Interprocess communication mechanisms, their purpose, classification and related system calls. Unnamed and named pipes and related programming. (Midterm exam)
Weeks 11-12	Message queues. A client-server system with message queues. Semaphores and shared memory for interprocess communication. Signals and their use and programming with them.
Weeks 13-14	Sockets for remote interprocess communication. IP and port addressing of processes for communication through sockets. UDP sockets. A UDP-based client-server system and the related system calls. TCP sockets for reliable remote interprocess communication and the related system calls. Conclusion. (Final exam)

Laboratory Schedule
(2 hours of laboratory per week)

Week 3	Introductory laboratory work for UNIX
Week 4	Study of processes in UNIX
Week 5	Advanced work on processes
Week 7	Threads in UNIX
Week 8	Unnamed pipes
Week 12	Understanding message queues
Week 13	UDP sockets for interprocess communication
Week 14	TCP sockets for interprocess communication

Course Learning Outcomes

On successful completion of the course, the student is expected to be able to:

(1) design programs with command line parameters in UNIX (c2,c3)
(2) tell the difference between conventional function calls versus system calls in UNIX (k1,k2,k3)
(3) classify system calls in UNIX (k1,k2)
(4) describe the concept of process and use processes in programs (e1,e2,e3,c3)
(5) understand file system and file system calls in UNIX (k1,k2,k3,c3)
(6) describe the threads and the relation to the process (k1,k2,k3,c3)
(7) differentiate communication between processes and between threads in the same process (e1,e2,e3,c3)
(8) define mechanisms for local and remote interprocess communication (IPC) mechanisms in UNIX (k1,k2,k3)
(9) implement the client-server models with local IPC mechanisms (c1,c2,c3)
(10) implement the client-server models with remote IPC mechanism (sockets) (c1,c2,c3)

	Method	**No**	**Percentage**
Assessment	Midterm Exam(s)	1	45%
	Lab Work(s)	8	10%
	Final Examination	1	45%
	Attendance	Mandatory	-

Attendance and Participation: Attendance to every lecture is mandatory.
There will be **no points** for the attendance.

Policy on makeups: Only one comprehensive make-up examination will be given to those who miss any of the exams (midterm or final) and will cover all the topics listed above. **The make-up exam will be given to only those who provide a valid excuse in writing within the next three working days following the missed exam. Students who miss an exam due to a serious medical condition are required to provide official documentation (doctor's report approved by the Student Health Center).**

The re-sit exam will cover both midterm and final topics, and it will replace both midterm and final.

Policy on labs

- There are **no makeups** for missed lab works.
- The student who repeats this course **must perform** all lab works.
- Each lab work will be explained **one week before** by the instructor during the lecture.
- Each exam will include topics from laboratory works.

Policy on cheating and plagiarism: Plagiarism (which also includes any kind of cheating in exams, assignments, and lab works) is a disciplinary offence and will be dealt with accordingly. Furthermore, the penalty of plagiarism is to get grade zero for the corresponding exam, assignment, or lab work.

Contribution of Course to Criterion 5
Credit Hours for:

Mathematics & Basic Science : 0
Engineering Sciences and Design : 4
General Education : 0

Prepared by: Assoc. Prof. Dr. Gürcü Öz	**Date Prepared: 24** September 2018

DETAILED COURSE CONTENTS (Fall 2018-2019)

1. Scope and tasks of systems programming. Traditional areas of systems programming. Systems programming in an OS environment (using OS program services). UNIX, its history, features and services. The objectives of systems programming in UNIX. (Lecture notes; ***Ch. 1).

2. A program in the UNIX environment. Steps in the development of a program in UNIX. Command line parameters. Environment variables. Libraries. Printing error messages. (Lecture notes, and *Ch.2, pp. 47 - 56).

3. System calls, their importance for systems programming, and classification. A system call and a conventional function call. System calls for interprocess communication and for network programming. General system call interface in UNIX. (Lecture notes, **Ch.1(Section 1.3), ***Section 2.7.2).

4. Processes as fundamental objects in UNIX. Creating a process. Process ID. Parent process ID. Child process ID. (Lecture notes, and *Ch. 11, **Ch. 1(section 1.2), **Ch. 5, ***Sections 8.1 and 8.2).

5. Using processes. More about the **fork()** system call. A family of **exec** system calls. The **system()** system call. **exit()** and **wait()** system calls and their using. (Lecture notes, and *Ch. 11, **Ch.5, ***Sections 8.4.3 and 8.4.4).

6. Basic concepts of threads and multithreaded programming (Lecture notes, **Ch.12, Section 12.6.2, ***Ch. 14).

7. System calls for files, their purpose and using in programs. Programming operations for files and directories. (Lecture notes, and *Ch.3, 4, 5, and **Ch.2, 3 and ***Sections 2.5.1 and 2.5.2)

8. Interprocess communication, its purpose and using in systems programs. Mechanisms of interprocess communication in UNIX. Interprocess communication for computer networks. A client-server paradigm of interprocess communication in networks (Lecture notes).

9. Unnamed and named pipes for interprocess communication. Impossibility of using unnamed pipes in UNIX for network communication (Lecture notes, and *Ch. 13, pp. 353 - 366, **Ch.7, ***Section 10.6).

10. Message queues. (Lecture notes, and *Ch. 13, pp. 377 - 382, **Ch. 8).

11. Signals and semaphores. (Lecture notes, and *Ch. 13, pp. 385 - 389, **Ch. 6, ***Sections 6.4 and 15.4.2)

12. Shared memory. (Lecture notes, and *Ch. 13, pp. 382 - 385, **Ch. 8).

13. Sockets and their using for interprocess communication in computer networks. Client/Server model and its implementation with sockets in computer networks. IP addressing with sockets in networks. Port numbers. TCP and UDP sockets for connection-oriented and connection-less communication in networks. Organization of a Web client-server network system (Lecture notes, and *Ch. 13, pp. 367 - 374 and *Ch.14, **Ch.10, ***Section 11.5).

14. **Optional topic**: Remote procedure call (RPC) for networks, its operation and parameter passing. Client/Server network programming with RPC. (Lecture notes).

15. **Optional topic**: Introductory concepts of systems and network programming in Windows operating systems. TCP and UDP sockets for network communication in Windows environment.

GUIDELINES ON LAB WORKS

[1] Introduction

Lab works is a very important component in mastering computer engineering courses. The objective of lab works is to give students a practical experience in the corresponding courses. For this reason, the attendance of lab works by students is obligatory. To get the most from lab works, the student must make the necessary preparations before each lab work, perform the lab work as a small research project, fix the results of the lab work in a notebook, and answer questions asked by lab work assistants during the lab work. Below are some guidelines that should be taken into account by students and lab work assistants.

[2] Supporting materials

All supporting materials, including lab work descriptions, necessary programs, program tools, and sources of additional information, are available to students on a web page for the corresponding course. In particular, materials on lab works in CMPE343 can be accessed via http://cmpe.emu.edu.tr/courses/cmpe343 .

[3] Order of performing of lab works

Lab works are performed in the order in which they are given by the instructor. Before the week of performing a lab work, the instructor gives the name of the lab work and explains this lab work at a lecture. To benefit from this explanation, the students should have a printed description of the lab work.

[4] Preparation to lab works

The student should come to each lab work in time and fully prepared. The preparation means that the student has the printed description of the lab work, the original and updated programs (if required for this lab work), and is capable to answer the questions

listed in the lab work description and asked by lab work assistants. In the preparation to a lab work, the student should read the related lecture material, including the corresponding section of the textbook. Trying to answer the questions listed in the description of the lab work, the student should do it independently as part of home work.

[5] Performing lab works

During each lab work, the student should carry out all steps listed in the description of the lab work. It is assumed that the student knows how to utilize the underlying OS (Windows or UNIX) to run the necessary tools and to develop programs. The results of performing of each step of the lab work must be fixed in a notebook and shown to a lab work assistant on his/her request. If the student uses some tool for the first time, a brief description of its use must be done in a notebook for the subsequent use. For example, the student should have a list of commands of UNIX and of the text editor "pico" that is used to create source program texts in UNIX. Without the fixed results, the lab work can be considered as not finalized by a lab work assistant. In the evaluation of the lab work for this student, the assistant will take into account the preparation to the lab work, the activity of the student during the lab work, the obtained and fixed results, and the answers of the student to questions. The student can be given a bonus if he/she performs additional experiments related to the lab work.

[6] Keeping lab work notes

After performing a lab work and getting a mark for it, the student is highly recommended to keep all materials of the performed lab work for a possible use in future. This is necessary, first of all, for the preparation to a quiz or exam, in which questions related to this lab work can be asked. These materials can be helpful also in a graduation project later if the student is given a graduation project topic related to the lab work.

[7] If the student missed a lab work

There are no make-ups for missed lab works.

[8] If the student repeats the course

The student who repeats the course must perform all lab works for this course.

[9] Help from a lab work assistant

During a lab work, the student can expect a help from a lab work assistant. This help generally includes the following: providing information on the used operating system, the underlying network (IP addresses, port numbers, etc.), servers to download the necessary data, on how to install and launch a tool required for this lab work. The student should not expect that he/she will get the explanation of the lab work from a lab work assistant. Instead, the student must attend the lecture at which the explanation of the lab work is given by the instructor and get prepared to this lab work. The assistants

should not be expected to participate in the debugging of the lab work programs; this must be done by the student independently.