Karim Lounis

Currently Ph.D student at school of computing (Lab 631).

Research in IT security, IoT security, network security.

Research assistant in security (Germany then Luxembourg).

Master’s in Security from Paris-XII University (France, 2014).

Master’s in Networks & Distributed Systems from USTHB (Algeria, 2013).

Bachelor’s in Networks & Telecom’ from USTHB (Algeria, 2011).

Can visit my website for more details: https://lounis.weebly.com/

Question about Master’s degree, Ph.D, or my research field?

Office hours: Mondays 10:30am-12:00pm room 241 at Goodwin Hall.
What about the course?

Operating Systems a.k.a., CISC324
http://research.cs.queensu.ca/~cisc324/
What is an Operating System?
What is an Operating System?
What is an Operating System?

**Definition**

**Operating system:** is a set of programs (modules) that cooperate for the good management and use of computer resources.

The operating system provides a set of services to both computer users and user programs:

- It provides computer users with a set of services to:
  - Execute their programs.
  - Exploit computer resources in an efficient way.
  - Make programming tasks easier for developers.
  - Present a virtual machine.

- It provides user programs with an environment for their execution.
What is an Operating System?

Services provided by the operating system to the users (\{programs\}):

**User interface**: Could be a Batch Interface [1945-1968], a CLI (Command Line Interface)[1969-] or GUI (Graphical User Interface) [1970-].

**Program execution**: Programs need to be loaded into the main memory and get executed till completion. The operating system ensures that.

**I/O operations**: Running programs may require I/O operations, the operating system provides special functions to talk to I/O devices.

**File system manipulation**: Programs may need to manipulate files. The Operating system provides special functions for such manipulations.

**Communication**: Programs may need to communicate with each other, locally or remotely. The Operating system provides special functions to a safe, secure, and reliable communication.
What is an Operating System?

**Error detection**: The Operating system needs to detect and correct errors when they occur during programs execution. For each type of error, the operating system takes the appropriate action.

Services provided by the operating system to the users and system itself:

**Resource allocation**: The operating system ensures that resources are shared in an optimal manner among multiple users and jobs.

**Accounting**: The Operating system keeps track of which user use how much and what kind of resources for accounting and statistics.

**Protection and Security**: The operating system provides certain security services to its users (or processes) to preserve their security and privacy.
What do you need for the course?

What do you need for the course?

**Course reader:** By Prof. Blostein, a highly recommended document that you can use.
What do you need for the course?

**Lecture slides:** Will be posted online.
What do you need for the course?

Background:

- Have used Java and C programming languages before.
- Have worked on Linux environment.
- Some notions about x86 assembly language (e.g., Mov Ax, Bx).

Miscellaneous:

- Online materials: Course videos (e.g., Youtube), online books, ...., course materials from other universities.
- Offline materials: Computer, Books (hardcopy), ...,
What’s the course skeleton?

The course is made of the following components:

- There are 12 weeks of lectures (3 lectures/week, 1h/lecture).
  
  **Mondays 8:30am, Tuesdays 10:30am, and Thursdays 9:30am.**

- There are 6 assignments.

- There are 6 labs.

- There are 2 midterm exams (45m).

- There is one final Exam (3h).

- There are 4 Teaching Assistants:
  
  - Taher Ahmed Mohammed Ghaleb, 17tamg@queensu.ca.
  - Thomas Parker, 12tpjp@queensu.ca.
  - Omar El Zarif, 18oez@queensu.ca.
  - Junlu Zhou (a.k.a., Sylvia), jz52@queensu.ca.
What do you need to succeed in the course?

Not that many:

1. Attend the course lectures, focus while I am talking, ask questions if not clear.
2. Complete your assignments and hand them on time.
3. Complete your labs and submit them on time.
4. If you did not understand something in the course, try to not miss office hours.
5. In the worst case use email.
What’s the course organization?

- **Week 1-2**: Intro, overview, and review.
  - Review computer architecture (Von Neumann model)
    e.g., CPU, memories, and I/O devices.
  - Review of instruction execution cycle
    e.g., How does a CPU executes: `while(1){i++;;}`?
  - Process management: from creation to execution then termination.
    e.g., How OS keeps track of thousand of processes?
  - Interrupt system and context switching
    e.g., What happens when CPU executes `0 0` or user types `Ctrl+Alt+Del`?
  - Concurrent processes: fake and real concurrency
    e.g., How can multiple processes execute simultaneously on one CPU?
  - Notions about threads: lightweight processes
    e.g., How to increase parallelism and lightening context switching?
What’s the course organization?

- **Week 3-4-5: Process synchronization**
  - Shared data corruption in concurrent programming
    e.g., Assuming i=5 and P1: i++ and P2: i--, what is the value of i?
  - Critical section problem
    e.g., understand and identify programs critical sections.
  - Synchronization mechanism
    e.g., How to keep critical sections safe from corruption?
  - Classical synchronization problems
    e.g., Dinning philosophers, Readers/Writers, Consumer/Producer?
  - Semaphores as a general synchronization mechanism
    e.g., How to use semaphores to synchronize concurrent processes execution?
  - Deadlocks and deadlocks avoidance, prevention, and recovery
    e.g., How to keep concurrent processes safe from infinitely waiting for each other?
What’s the course organization?

- **Week 6-7-8-9**: Memory management

  - Memory pyramid
    - e.g., What are the different types of memories?
  - Address space and address translation
    - e.g., How programs see and address the physical memory?
  - Old fashion memory management systems
    - e.g., How multiple programs can be simultaneously loaded onto memory?
  - Virtual memory
    - e.g., How can we have a memory that seems to have an infinite capacity?
  - Paging and segmentation
    - e.g., How to better use the memory (better sharing)?
What’s the course organization?

- **Week 10-12**: Subject to modification but will cover:
  - Monitors
    - e.g., How can we easily use synchronization mechanisms in programming language?
  - Message passing
    - e.g., How processes communicate with each other locally and remotely?
  - CPU Scheduling
    - e.g., How the OS selects which process is allocated the CPU and for how much time?
  - OS security
    - e.g., How OSs provide security services for their users?
  - OS architecture
    - e.g., What is a monolithic kernel and microkernel?
When do we start?
When do we start?

Assignment 1, Lab 1, and Lab 2 are due Jan 22.
End