**Study Guide for Midterm Exam**

**Comp 4956 2015**

Lecture notes: Parts 1, 2, 3 and 4

Labs: Lab assignments and activities, and issues discussed in lab

Readings: as specified in assignments and power-point presentations

Project general architecture and your team’s solution and tasks

**Learning Outcomes / Questions**

1. What are the main two roles that the Windows registry has?

Administrative Management

* Keeps track of users
* Keeps track of performance data

OS Management

* Keeps track of available disks
* keeps track of applications and application data

1. What type of data is stored in the registry? Is this data static or dynamic?

* Keys
  + similar to a disk’s directories
  + container that can consist of other keys (subkeys) or values
  + top-level keys are *root keys*
* Values
  + similar to files on a disk
  + store data

1. Where the registry is physically stored?

* The hard disk

1. Is the registry part of the memory pool? Why or why not?

* Yes. Registries are interfaces to several of in-memory structures maintained by the executive and kernel

1. Give 3 examples of data types defined for the registry.

* REG\_DWORD
* REG\_BINARY
* REG\_SZ

1. Describe the six root key of the registry. Which key is not visible in the registry’s editor?

* HKEY\_CURRENT\_USER
  + Stores data associated with currently logged-in user
* HKEY\_USER
  + Stores data for all users registered on this machine
* HKEY\_CLASSES\_ROOT
* HKEY\_LOCAL\_MACHINE
  + Stores system-related information
* HKEY\_PERFORMANCE\_DATA
  + Stores performance information
* HKEY\_CURRENT\_CONFIG
  + Stores info about the current hardware profile

1. Per-user class registration was introduced first in Windows 2000. Why per-user class registration has been added to the registry?

* to close a security hole where a non-privileged user could change or delete keys in HKEY\_CLASSES\_ROOT, and
* so roaming profiles (log into windows server domain) could contain customizations

1. Regarding the registry, where is the performance data stored? Describe the mechanism for accessing the performance data.

* HKEY\_PERFORMANCE\_DATA
* Data is not actually stored in a table
* It’s just a key that points to performance data providers

1. What is the Windows Management Interface?

* WMI actually stands for Windows Management Instrumentation. It is an API that allows developers to manage and control Windows devices and systems in a network.

1. In what language(s) is (are) the WMI written?
2. What are the WMI features?

* Works locally and remotely
* Is bi-directional
* Extensible
* Windows objects can be observed with WMI
* Extensible to other objects
  + The application designer provides an interface such that wmi uses the interface for instrumentation and preformance

1. Why is needed a common WMI?
2. What are the WMI advantages?
3. How is WMI implemented in the .NET framework? What are the advantages and limitations of a .NET development platform for WMI?
4. In your assignment, what did you use to access the disk/process data? What about the registry?
5. Describe in your own words the process creation and termination in UNIX, Linux and Windows.
6. What is happening when the system call fork() is called in UNIX or Linux?
7. What is the role of the instruction wait() in: child\_pid = wait(&status);
8. What is a thread? Describe a thread data structure and characteristics in details. How is a thread different from a process and a fiber in Windows OS? What is the equivalent of a fiber in UNIX?
9. What is the difference between the microkernel and the Executive in Windows OS (what are their roles)?
10. How do processes communicate? (Consider all IPC mechanisms and shared memory method). What about threads?
11. How many stacks are implemented for threads in Windows? Why?
12. What is the role and details of implementation for HAL.dll? What is its content?
13. What is NTOSKRLN. EXE?
14. What are the process’ components in Windows? What about thread’s components?
15. Describe issues of vulnerability related to execution in kernel mode.
16. Discuss kernel-mode vs. user mode.
17. Draw the diagram of Windows OS indicating the user and kernel mode.
18. Name 5 .dlls that represent important parts of the Windows OS.
19. Explain the multithreaded execution in Windows (switching from one thread to another).
20. What is the benefit of using \_beginthreadex()?
21. Explain why the thread executions in assignment Multithread.exe (written in C) do not end at the same value of the counter.
22. Explain why the thread executions in assignment Multithread.exe (written in C#) end at the same value of the counter.
23. How many threads were created in your .NET Multithread assignment? Why?
24. What was the role of a delegate in the .NET Multithread assignment?
25. How is a semaphore implemented? Write the pseudo-code of the functions signal(), wait() and init().
26. Describe the functionality of a semaphore by giving an example of a situation that is solved with semaphores.
27. Describe the functionality of a mutex by giving an example of a situation that is solved with mutexes.
28. What is a monitor?
29. What is the difference between a monitor and a semaphore?
30. What situations are better solved with monitors? What about semaphores?
31. Write the pseudo code for the producer – consumer with a circular buffer for the most general situation which will satisfy multiple producers and multiple consumers:
    1. With semaphores
    2. With monitores
32. How are semaphores / monitors implemented in Windows OS and .NET?
33. Consider the multithread assignment:
    1. What solution did you use in the multithreaded assignment to communicate between the counting threads and the timer thread? Why?
    2. What safety mechanism should be implemented for their communication?
34. Explain the differences between beginthreadex() and CreateThread().
35. In your assignments you used Executive objects exposed via the Win32 interface or the .NET framework.
    1. Name four of them.
    2. Explain the Executive access to an object. How does an Executive component specify the use of an object?
36. In the .NET framework you have available several primitives for synchronization. Compare and contrast the Monitor, Semaphore and Mutex classes. For each explain what kind of synchronization is used: at the level of Kernel Windows objects or not. In what circumstances will you decide to use one of them?
37. What is the base and dynamic priority of threads in Windows? Why do threads have dynamic priorities?
38. In the Multithread assignment: Is the number of context switches performed by the dispatcher for each thread equal? Why?
39. In the Multithread assignment: What is a thread doing in kernel mode?