

CPSC 457
OPERATING SYSTEMS
MIDTERM EXAM

Department of Computer Science
University of Calgary
Professor: Carey Williamson

March 9, 2010

This is a CLOSED BOOK exam. Textbooks, notes, laptops, calculators, personal digital assistants, cell phones, and Internet access are NOT allowed.

It is a 75 minute exam, with a total of 60 marks. There are 11 questions, and 9 pages (including this cover page). Please read each question carefully, and write your answers legibly in the space provided. You may do the questions in any order you wish, but please USE YOUR TIME WISELY.

When you are finished, please hand in your exam paper and sign out. Good luck!

Student Name: _____

Student ID: _____

Score: _____ / 60 = _____ %

Multiple Choice

Choose the best answer for each of the following 6 questions, for a total of 6 marks.

- 1 1. Three important design principles in operating systems are:
 - (a) caching, virtualization, and support for concurrency
 - (b) abstraction, fairness, and context-switching
 - (c) throughput, response time, and support for priority
 - (d) parallelism, concurrency, and multi-everything
 - (e) policy, mechanism, and superuser control
 - (f) all of the above

- 1 2. Changing the permissions on your files is easy in Linux because:
 - (a) there is a built-in command-line utility for doing so
 - (b) there is a C programming library routine for doing so
 - (c) there is a shell program for doing so
 - (d) there is a Java program for doing so
 - (e) there is a superuser on call for doing so
 - (f) all of the above

- 1 3. In *message-passing* IPC on a Linux system, the maximum message size permitted is:
 - (a) 1 byte
 - (b) 4 bytes
 - (c) 32 bytes
 - (d) 64 bytes
 - (e) 140 bytes
 - (f) none of the above

- 1 4. When a process is created using the classical `fork()` system call, which of the following is **not** inherited by the child process?
- (a) process address space
 - (b) process ID
 - (c) user ID
 - (d) open files
 - (e) signal handlers
 - (f) none of the above
- 1 5. The *text segment* of a process address space contains:
- (a) the statically allocated data associated with the process
 - (b) the dynamically allocated data associated with the process
 - (c) the executable code associated with the process
 - (d) the inter-process communication (IPC) messages for the process
 - (e) the text-messaging chat messages for the process
 - (f) all of the above
- 1 6. The threading model supported by the Linux operating system is:
- (a) many-to-one
 - (b) one-to-one
 - (c) one-to-many
 - (d) many-to-many
 - (e) all of the above
 - (f) none of the above

Operating System Principles

10 7. In class, we have illustrated operating system principles using examples from former and current operating systems, including Multics, Unix, Linux, Windows, and Mac OS.

(a) (2 marks) What is an *Operating System (OS)*?

(b) (5 marks) List and briefly describe any 5 of the typical services provided by an OS.

(c) (3 marks) Most operating system functionality can be provided using a variety of mechanisms, including system calls, built-in commands, and user-level programming support. Give 3 examples, from either Linux, MacOS, or Windows, of useful operating system functionality, indicating clearly the mechanism by which the service is provided.

Operating System Concepts and Definitions

12 8. For each of the following pairs of terms, define each term, making sure to clarify the key difference(s) between the two terms.

(a) (2 marks) “process” and “processor”

(b) (2 marks) “fork()” and “exec()”

(c) (2 marks) “signal” and “pipe”

(d) (2 marks) “C program” and “shell program”

(e) (2 marks) “pre-emptive” and “non-preemptive”

(f) (2 marks) “CPU-bound” and “I/O-bound”

Processes and Threads

- 12 9. Most modern operating systems provide support for both *processes* and *threads*.
- (a) (2 marks) What is a *process*?

 - (b) (2 marks) What is a *thread*?

 - (c) (3 marks) List any 6 of the attributes that an operating system maintains to keep track of information about a process.

 - (d) (3 marks) List the 3 most important attributes that an operating system maintains to keep track of information about a thread.

 - (e) (2 marks) List 2 reasons why the scheduling of processes and threads on a multi-processor system is more complicated than scheduling them on a uni-processor system.

CPU Scheduling

- 10 10. Consider the following set of jobs to be scheduled for execution on a single CPU system.

Job	Arrival Time	Size (msec)	Priority
J_1	0	10	2 (Silver)
J_2	2	8	1 (Gold)
J_3	3	3	3 (Bronze)
J_4	10	4	2 (Silver)
J_5	12	1	3 (Bronze)
J_6	15	4	1 (Gold)

- (a) (2 marks) Draw a Gantt chart showing FCFS scheduling for these jobs.
- (b) (2 marks) Draw a Gantt chart showing (non-preemptive) SJF scheduling.
- (c) (2 marks) Draw a Gantt chart showing non-preemptive PRIORITY scheduling.
- (d) (2 marks) Draw a Gantt chart showing preemptive PRIORITY scheduling.
- (e) (2 marks) Which of the foregoing scheduling policies provides the lowest waiting time for this set of jobs? What is the waiting time with this policy? (Show your work)

Operating System Utilities

10 11. The output on the next page is from a lightly-used Linux system on the recent weekend. Use the output and your knowledge of Linux systems to answer the following questions:

- (a) (1 mark) On what date (approximately) was this system most recently rebooted?
- (b) (1 mark) How many distinct users are logged in on this system?
- (c) (1 mark) Approximately how many processes are currently present on this system?
- (d) (1 mark) How much physical memory (RAM) does this particular Linux system have?
- (e) (1 mark) What percentage (approximately) of the memory is currently in use?
- (f) (1 mark) Which process (PID or name) has consumed the most CPU time so far?
- (g) (1 mark) Which process (PID or name) is currently consuming the most memory?
- (h) (1 mark) Which process (PID or name) is the oldest process on this system?
- (i) (1 mark) Which process (PID or name) has created the most child processes?
- (j) (1 mark) What is your professor's favourite editor?

```
[carey@csl]$ w
09:11:09 up 184 days, 17:43, 3 users, load average: 0.52, 0.37, 0.35
USER      TTY      FROM          LOGIN@   IDLE   JCPU   PCPU WHAT
carey     pts/5    csg           06:32    0.00s  0.47s  0.00s w
carey     pts/6    ict736a      Thu09   19:41m  0.04s  0.04s -csh
jess      pts/8    ict624       08:16   31:25m  0.02s  0.06s -bash
```

```
[carey@csl]$ ps
  PID TTY          TIME CMD
 1179 pts/5    00:00:00 csh
 1407 pts/5    00:00:00 emacs-x
 3721 pts/5    00:00:00 sleep
 3740 pts/5    00:01:00 ./fractal
 3748 pts/5    00:00:00 ps
```

```
[carey@csl]$ ps -l
F S  UID  PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  214 1179 1178  0  75   0 - 1434 rt_sig pts/5    00:00:00 csh
0 T  214 1407 1179  0  75   0 - 3133 finish pts/5    00:00:00 emacs-x
0 S  214 3721 1179  0  77   0 -  985 -      pts/5    00:00:00 sleep
0 R  214 3753 1179  0  75   0 - 1102 -      pts/5    00:00:00 ps
0 R  214 3740 1179  0  75   0 - 4270 -      pts/5    00:01:00 ./fractal
```

```
[carey@csl]$ top
top - 09:11:25 up 184 days, 17:58, 3 users, load average: 0.26, 0.34, 0.34
Tasks: 179 total, 2 running, 141 sleeping, 36 stopped, 0 zombie
Cpu(s): 2.5%us, 5.6%sy, 0.0%ni, 91.9%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%
Mem: 3369524k total, 2915008k used, 454516k free, 356296k buffers
Swap: 2040244k total, 54544k used, 1985700k free, 2361776k cached
```

```
  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
29105 cristo   16   0 1290 1260 1260 R   23  3.8   54:13.56  linux
3567  carey    15   0   400   400  400 T    8  0.6    0:00.24  top
1407  carey    15   0 3133 2820  311 S    5  7.2    0:00.18  emacs-x
3740  carey    15   0 4270 4100  170 S    2  8.1    0:01.00  ./fractal
3721  carey    15   0   980   980  980 S    2  1.2    0:00.04  sleep
   1  root     15   0 2060  476  448 S    0  0.0    0:41.56  init
   2  root     RT  -5    0    0    0 S    0  0.0    0:07.75  migration/0
   3  root     34  19    0    0    0 S    0  0.0    0:07.17  ksoftirqd/0
   4  root     RT  -5    0    0    0 S    0  0.0    0:00.00  watchdog/0
```

*** THE END ***