CS439

Fall 2013

Final Exam Solutions and Criteria

December 11, 2013

Name:
EID:

- The exam is closed book and closed notes.
- Be a smart test taker—if you get stuck on one problem, go on to the next. Don't waste your time giving details that the question does not request.
- Please give **clear** answers. If you give more than one answer, we will grade only the first answer, so please clearly mark out any answer you do not want us to grade.
- Show your work. Partial credit is possible, but only if you show intermediate steps.
- Make sure you clearly write your name and EID on this page.
- No electronics of any kind, including phones, laptops, and calculators, may be used during this exam.
- Scratch paper is included at the end of the exam and available upon request.
- 106 points are available on this exam, but the exam will be graded out of 100. Questions have been assigned half points which total 212.

Good Luck!

Question	*Half Points Lost*	*Half Points Earned*	*Maximum Half Points*
1			12
2			12
3			8
4			10
5			16
6			10
7			22
8			11
9			14
10			12
11			21
12			30
13			34
Total			212 (Half Points)

1. **Operating Systems and Dual Mode Execution**

   (a) [4pts] One of the main goals of an Operating System is to provide protection. Name two ways in which it provides protection, and for each way indicate why protection is necessary.

   ```
   Virtual Memory: other processes could access each other's memory
   Process/address space: isolates process data (same reasoning as VM)
   File Systems/file permissions: prevents users/processes from accessing
                                  file data
   Dual-Mode Execution: prevents processes from executing all instructions
      unless they are particularly trusted

   [there may be others]

   -2 each way wrong
   -2 no understanding of how chosen example fits into protection scheme
   ```

   (b) [2pts] Does the switch from user mode to kernel mode require a context switch? Defend your answer.

   ```
   No - thread/process that was rnning now executing OS code

   -2 wrong answer
   -1 wrong defense
   ```

   (c) The ability to overlap I/O and computation was a huge step forward in OS development.

      i. [2pts] Why?

      ```
      Expensive CPU can be kept busy while slow I/O is taking place

      -2 wrong
      ```

      ii. [4pts] What future step did it enable? Defend your answer.

      ```
      Multi-programming: one process can be blocked in I/O and another process
         can execute---without the ability to overlap, not possible

      -2 no multi-programming (or description of multi-programming) or
         other reasonable step
      -2 no reasonable defense
      ```

2. **Processes and Threads Short Answer**

(a) [2pts] In a given system, there is a process with 5 user-level threads, a process with 3 kernel-level threads, and 2 single-threaded processes. All are ready and waiting for the CPU. How many entries are in the ready queue?

6
-2 wrong

(b) Given the following poorly-written code (check your return values!):

```
int gv = 3;
int main(){
   gv += 2;
   fork();
   foo();
}

void foo(){
   gv+=1;
   printf("%d\n", gv);
}
```

i. [2pts] What is the output? Describe in detail why it is that way.

6
6

fork() creates another process, both of which have a copy of gv, both of which add 1, and then print

-2 wrong output
-2 wrong explanation
[max of 2]

ii. [3pts] Are there other possible outputs? Why or why not? If so, give another example of what the printed value of `gv` might be and explain how to force the program to execute in the same way each time.

No.
-3 wrong

(c) In the code above, assume that the lines `fork();` and `foo();` have been replaced by `thread_create(foo);`, where `foo()` is the thread function.

i. [2pts] What is the output? Describe in detail why it is that way.

6
only one thread was created, so only one thread executes foo();
-2 wrong

ii. [3pts] Are there other possible outputs? Why or why not? If so, give another example of what the printed value of `gv` might be and explain how to force the program to execute in the same way each time.

```
No, since only one thread modifies gv at any given time.
OR
yes, there could be no output since the main thread exits immediately
after calling thread_create()

-3 answer is not either of those
```

3. **Synchronization Short Answer**

(a) [4pts] For Winter Break, UT Austin plans to provide shuttle buses for students traveling home to Dallas or Houston. Students may sign up in person on the sign up sheet in the Texas Union. Every few hours, the organizer of the event will see how many people have registered on that sheet, call the bus company and reserve buses with the exact number of necessary seats, and then replace the sign up sheet with a new one. The organizer is fortunate that the students, once signed up, do not change their minds about riding the bus.

What problems exist in this situation and what can be done to solve them? Be sure to think in terms of concurrent processing and include concepts from this course in your answer.

```
Race Condition!  What is students sign up between the checking of the sheet
and the sheet being replaced?

Solution: Remove/replace sheet at same time as seeing how many people have
registered

-4 no realization of race condition (-1 missing term)
-2 missing solution
```

(b) [4pts] In class, we decided that spin locks used to lock large segments of code are Bad, but we did find a use for them—what was it?

```
In the lock_acquire/lock_release implementation

Also allowed:
fast response time

-4 wrong
```

4. **Synchronization** Some children are building a snowman, but for it to be complete they need a hat. They have found some hats at their respective houses, but none of them are quite right. So the children decide to wait until the perfect hat arrives. Fortunately for the children, there is a hat fairy that periodically sends perfect snowmen hats out into the universe. Write the following functions:

```
void build_snowman();
void hat_fairy();
```

which may be executed by multiple groups of children (threads) or fairies (also threads) at once, using

(a) [4pts] semaphores, and

(b) [6pts] monitors.

```
Semaphore hat_sema;

void build_snowman(){
  <gather snow>
  <build snowman>

  hatSema->down();

  <add hat>
  <watch snowman dance>
}

void hat_fairy(){
   hatSema_up();
}
```

```
a) -2 use of resource variables
   -2 use of while
   -2 have a lock lock
   -2 sleep
   -2 missing down
   -2 missing up
   -2 more than one semaphore
   -2 each random extra synch
```

```
Lock lock;
CondVar hatCV;
int hat = 0;

void build_snowman(){
   lock->lock_acquire();
   <gather snow>
   <build_snowman>

   while(hat < 1)
      hatCV->wait()
   hat--;

   <add hat>
   <watch snowman dance>
   lock->release();
}
void hat_fairy(){
   lock->lock_acquire()
   hat++;
   hatCV->signal();
   lock->lock_release();
}
```

```
b) -4 no condition variable
      -2 missing wait
      -2 missing signal
   -4 missing variables to manage resources
      -4 while vs. if or no loop
        -2 improper loop
        -1 each failure to adjust
   -4 no locks
      -2 no use or not at beginning and end
      -2 multiple locks
   -2 busy wait
   -2 sleep
   -1 each random extra synch
```

5. **Virtual Memory Short Answer**

(a) Using no more than forty words each, describe:

i. [4pts] the goal of virtual memory,

```
Virtual memory 1) allows processes to "use" more memory than is physically
available and 2) allows multiple processes to exist in memory at once

Also: eliminate external fragmentation

-2 no mention of larger amounts per process
-2 no mention of multiple processes
```

ii. [4pts] how virtual memory works, and

```
Virtual memory divides process address spaces into pages and then maps
those pages into physical memory frames as they are needed.  Unneeded
pages are left on disk.

-2 no mention of pages
-2 no mention of mapping pages
-2 no mention of pages on disk
[max of -4]
```

iii. [2pts] how virtual memory performs in a system currently demanding more memory than is physically available.

```
Poorly.  Lots of page faulting.  A load control policy is necessary.
-2 no mention of poorly/bad/not well/something similar
-1 no mention of page fault
-1 no description of swapping processes/load control policy
```

(b) A virtual memory system's efficiency in a heavily loaded environment is often dependent on its page replacement algorithm. One such algorithm is the enhanced clock (or second chance) algorithm.

i. [2pts] The enhanced clock algorithm tries to minimize the cost of page replacement by not replacing pages with what characteristics?

```
recently used or written
-1 each missing
```

ii. [4pts] Describe how the enhanced clock algorithm works.

```
Enhanced clock gives a second chance to pages that have been written.
It evicts pages that are not recently used or written.  Pages
that are recently used have their clock bit set to 0, pages that are
not recently used but are written either have their writes scheduled or
the OS "remembers" they are dirty.  Either way, the dirty bit is cleared.
```

```
-1 no mention of evicting 00 pages
-1 no mention of reset of clock bit
-1 no mention of handling write
-1 no mention of clearing modified bit
```

6. **Memory Management**

(a) Virtual memory systems are very complicated, so a new programmer at your company has proposed that the current memory system can be simplified for ease of implementation. The programmer has proposed that, instead, processes be allocated contiguously in physical memory. Since you took CS439, you know this idea is bad. In fact, you know this system has been used before!

i. [2pts] What was the historical version of this system called?

```
Relocation
-2 wrong
```

ii. [4pts] Give two disadvantages of the proposed system.

```
External fragmentation
Entire process must fit into memory
  could also say: limited memory for the process
                  limited degree of multiprogramming

[there may be others]

-2 each missing
```

(b) In class, we discussed paging, segmentation, and segmented paging.

i. [2pts] Name one advantage segmentation has over paging, and describe why that advantage is unique to segmentation. (That is, why the advantage does not apply to paging.)

```
Protection for specific code segments: paging can't distinguish the
segments Max VAS for each segment: paging only has max VAS for all areas

[may be others]

-2 wrong advantage
-1 no explanation of why it doesn't apply to paging
```

ii. [2pts] Name one advantage segmented paging has over segmentation, and explain why that advantage is unique to segmented paging. (That is, why the advantage does not apply to segmentation.)

```
No external fragmentation: segmentations suffers from this because it fits
   variable-sized pieces into memory
Share pages w/in segments: finer granularity of sharing, not possible with
   segments since segments have same permissions throughout

[may be others]

-2 wrong disadvantage
-1 no explanation
```

7. **File Systems and their Design**

(a) NTFS uses a different type of file layout than those used by Linux/Unix systems.

i. [2pts] Name two advantages to this system.

```
Extents provide locality
No need to follow pointers/do further access for small files
Upper file size bounded by disk and metadata

[may be others]

-1 each missing
[max of 2]
```

ii. [1pts] Name one disadvantage.

```
External fragmentation
Difficult implementation

-1 each missing
[max of 1]
```

(b) [3pts] Assuming your machine is using a variant of the Fast File System, that you must begin at the root directory, and none of the necessary information is cached, how many disk accesses are necessary to access the file `/u/ans/cs439/final_exam_answers.txt`? You may assume the relevant data is in the first data block of each access.

```
10

-3 wrong
-1 if shows correct work
```

(c) [16pts] For each type of file layout, mark whether it provides (or suffers from) each feature using ✓to indicate support, `X` to indicate no support, and `D` to indicate `It Depends`. You may further explain any `D` answers if you feel it necessary, but we don't promise to read your explanations.

	Random Access	**Large File Support**	**External Fragmentation**	**Internal Fragmentation**
Contiguous	✓	D	✓	X
Linked	X	✓	X	✓
Multi-Level Indexed	✓	✓	X	✓
Flexible Tree with Extents	✓	✓	✓	X

```
-1 each wrong
```

8. **Fancy File Systems**

(a) [4pts] What is the difference between coherence and consistency?

```
coherence governs accesses to a single memory location
consistency governs accesses across memory locations

-2 each wrong
```

(b) [3pts] Sun's NFS protocol provides reliability via what type of semantics?

```
at least once

If they answered client-initiated consistency, let them have it.

-3 wrong
```

(c) [4pts] Google designed its file system, GFS, to be scalable. Name two steps it takes to ensure scalability.

```
Server only has metadata
Chunk servers actually serve files, adn there are many of them
Each chunk is replicated three times
...

-2 each missing/wrong
```

9. **Networking**

(a) The TCP protocol is very careful to provide built-in congestion controls.

i. [4pts] Why is congestion control a particular concern for TCP?

```
Its attempt to provide reliability over a heavily congested network
can lead to more congestion and bring down the network (congestive
collapse)

-2 no mention of leading to more congestion
-2 no mention of congestive collapse
```

ii. [4pts] Name 2 of the 5 mechanisms TCP uses for congestion control, and provide a very short explanation for each.

```
Slow start: begins with congestion window set to 1 segment
Additive increase/multiplicative decrease: after slow start, will increase
   linearly but decrease by half on loss event
Reaction to timeout: reduces size of congestion window on loss event
Round trip time estimation: Allows a lot of variance
Exponential retransmit timer backoff: timer for retransmit will backoff
   exponentially in the event of a loss

-2 each missing method
  -1 wrong description

[max of 4]
```

(b) [6pts] What is the relationship of TCP, IP, and Ethernet? Explain how they fit together. (Diagrams with explanations are encouraged.)

```
 TCP [app to app]
  |
  v
  IP [end machine to end machine]
  |
  v
Ethernet [machine to router, switch, etc]
```

10. **Parallel and Distributed Systems**

(a) [4pts] How does the difference between a parallel system and a distributed system affect the ways in which we are able to program applications for those systems?

`Parallel can use shared memory, but distrubuted must use message passing`

`-2 no mention of parallel and shared memory`
`-2 no mention of distributed must use message passing`

(b) [2pts] We studied logical clocks and the happens before relationship. Why are these concepts useful in distributed systems but not necessary in parallel systems?

`no global clock in distributed systems, but processes/threads in a parallel system share a clock`

`-1 missing no global clock in distributed systems`
`-1 missing yes clock in parallel systems`

(c) In class, we also studied the Two-Phase Commit (2PC) protocol.

i. [4pts] What is the purpose of 2PC?

`distributed atomicity`

`-4 wrong`

ii. [2pts] Under what circumstances can a 2PC operation commit if the server goes down before sending the final message?

`None`

`-2 wrong`

11. **Miscellaneous**

(a) RAID can provide two advantages over standard disks.

i. [2pts] What are they?

```
throughput and fault-tolerance
```

ii. [1pts] Which reason is more important to you?

```
Either or both
```

iii. [4pts] What level of RAID would you choose to best accomplish the goal you chose in (ii)? Why? (Either the RAID level number or a brief description is okay here.)

```
Most will choose between 0 and 5
0: stripes, increases throughput
1: mirrors, increases fault-tolerance
2: we didn't go over this.  bit-striped with parity, provides both
3: byte-striped with parity, provides both
4: block-striped with parity, provides both
5: block striping with distributed parity, providse both

-4 RAID choice doesn't promote goal
  -2 if goal is one and they choose an option that does both, unless
     they explain that
```

(b) [6pts] In class, we studied several ways to prevent deadlock, both in our applications and in the OS.

Give one example of each of the deadlock avoidance, prevention, and detection techniques. Tell whether each is in use today, and either explain when it is used or why it is not used. If the technique is used, explain in what situations it is an appropriate technique.

That is: type of technique: example, if used, when? or if not, why not?

```
prevention: resource ordering, in applications
avoidance: banker's algorithm, no, too limiting---processes can't enter the
   system, resources can't leave, must declare all resource needs up front
detection: build resource allocation graph to detect cycles, no---too time
   consuming

-1 each incorrect example
-1 each incorrect when or why not
-1 for swapping psrevention and avoidance
```

(c) [2pts] What is the current most popular strategy for dealing with deadlock in the OS?

```
Ostrich Algorithm
```

-2 wrong

(d) When designing a system, a computer scientist should always consider security.

i. [2pts] Name two security design principles that should guide the system design.

Economical
Complete Mediation
Open design/no "security through obscurity"
Separation of privileges
Least privilege
Least common Mechanism
Acceptability
Fail-safe

Also allowed:
check inputs

-1 each missing
[max 2]

ii. [4pts] Name one common mistake that leads to security holes.

no checking the size of input before copying to variable
or not limiting the size of input to variable size
or buffer overflow
or security through obscurity

[may be others]

-4 wrong

12. **Overview and Pintos**

(a) [18pts] For each of the following pieces of a computer system we studied, identify system component of which it is a member (e.g., virtual memory, processes, ...), and whether it is hardware or software.

i. Translation Look-aside Buffer (TLB) `VM, HW`
ii. Protocol Control Block (PCB) `Networks, SW`
iii. Superblock `FS, SW`
iv. Page Table `VM, SW`
v. Process Control Block (PCB) `Processes, SW`
vi. Network Interface Card (NIC) `Network, HW`
vii. Base and Bounds Registers `Memory Management, VM, HW`
viii. Thread Control Block (TCB) `Processes or Threads, SW`
ix. Process Identifier (PID) `Processes, SW`

```
Each item is worth 2 pts
-1 wrong (or unreasonable) component
-1 wrong HW/SW
```

(b) Which component(s) of the computer system gives us the ability to:

i. [2pts] grow processes?

```
VM
others?
-2 wrong
```

ii. [2pts] run multiple processes on a single CPU so that they appear to be concurrent?

```
timer interrupt, CPU scheduler, VM, Processes
others?
-2 each missing (only one needed)
[max 2]
```

(c) [4pts] NFS uses RPC. Suppose that NFS is running on a server that uses RAID. How does that affect the design of NFS?

```
Doesn't
-4 wrong
```

(d) [4pts] In Project 3, you implemented virtual memory on top of a page table Pintos provided for you. How did you interface to that page table to let it know a particular page was being put in a particular frame?

```
install_page() or pagedir_set_page()
explanations may or may not be accepted.  we can discuss.
-4 wrong
```

13. **True or False** [34pts, 2 each]
The following are true or false questions. If you believe true or false is not an adequate answer, provide a short explanation with your answer.

(a) `T` Early CPU schedulers were humans.

(b) `F` When a process is in the waiting state, it is only waiting for the CPU.

(c) `T` Threads in the same process share a PID.

(d) `F` Threads in the same process share a stack.

(e) `T` This code cannot deadlock when executed by multiple threads:

```
thread_func()
{
   semaA->down();
   semaB->down();
   semaC->down();
   semaB->up();
   semaC->up();
   semaA->up();
}
```

(f) `T` A virtual memory system with a page size of 32 bytes and 4 pages has 7 bits in its virtual address.

(g) `F` If the disk is full but a process needs another page, then a page replacement algorithm is run to evict a page and create space.

(h) `F` In today's OSes, interrupts are the preferred method of communicating with disks.

(i) `F` It is fastest to access the inside track of a hard drive because the head has less distance to travel.

(j) `F` A solid state disk's access time could be improved by lowering rotational latency.

(k) `T` The FFS ensures locality through its use of the first-free algorithm.

(l) `F` When a client and server communicate using sockets, they communicate over the same socket that the client originally used to connect.

(m) `F` Message passing is simpler for the application programmer than RPC.

(n) `F` In distributed computing, the Bully Algorithm chooses a leader by electing the node that has participated in the majority of the recent communications.

(o) `T` The Banker's Algorithm guarantees there will be no deadlock in the system.

(p) `F` Security has been a focus of OS development since the beginning.

(q) `Ha.` CS439 is tough, but you finished! Good job!

Congratulations on being finished with CS439!
Enjoy the break!