This exam is closed book, closed notes. All cell phones must be turned off. No calculators may be used.

You have two hours to complete this exam.

There are two parts to this exam: the first is true/false, the second is multiple choice. Some of the T/F questions are very simple, and some will take you awhile to determine. We expect you’ll end up spending approximately equal amounts of time on each part.

Write all of your answers on the accu-scan form with a #2 pencil.

These exam questions must be returned at the end of the exam, but we will not grade anything in this booklet.

Unless stated (or implied) otherwise, you should make the following assumptions:
  o The OS manages a single uniprocessor
  o All memory is byte addressable
  o The terminology lg means log_2
  o 2^{10} bytes = 1KB
  o 2^{20} bytes = 1MB
  o Page table entries require 4 bytes
  o Data is allocated with optimal alignment, starting at the beginning of a page
  o Assume leading zeros can be removed from numbers (e.g., 0x06 == 0x6).

Good luck!
Part 1: Straight-forward True/False [1 point each]
Designate if the statement is True (a) or False (b).
1) An operating system is defined as hardware that converts software into a useful form for applications.
2) Examples of resources that the OS must manage include CPU, memory, and disk.
3) The abstraction that the OS provides for the CPU is a virtual address space.
4) A process is defined as an execution stream (or thread of control) in the context of a process state.
5) The address space of a process is part of its process state.
6) A process is identical to a program.
7) A process is identical to a thread.
8) Two processes reading from the same virtual address will access the same contents.
9) A modern OS virtualizes a single CPU with time-sharing.
10) Entering a system call involves changing from user mode to kernel mode.
11) When a user-level process wishes to call a function inside the kernel, it directly jumps to the desired function.
12) An example of a mechanism inside the OS is the process dispatcher.
13) Cooperative multi-tasking requires hardware support for a timer interrupt.
14) A timer tick is identical to a time slice.
15) PCB stands for process control base.
16) On a uniprocessor system, there may only be one ready process at any point in time.
17) A FIFO scheduler schedules ready processes according to their arrival time.
18) The convoy effect occurs when high priority jobs must wait for lower priority jobs.
19) A SJF scheduler uses the past run time (i.e., cpu burst) of a job to predict future run time (i.e., cpu burst).
20) A STCF scheduler guarantees that it will schedule the ready job with the smallest remaining cpu burst.
21) A RR scheduler may preempt a previously running job.
22) An RR scheduler tends to decrease average response time as the time-slice is decreased.
23) The shorter the time slice, the more a RR scheduler gives similar results to a FIFO scheduler.
24) If all jobs arrive at the same point in time, a SJF and an STCF scheduler will behave the same.
25) If all jobs have identical run lengths, a FIFO and a SJF scheduler will behave the same.
26) If all jobs have identical run lengths, a RR scheduler provides better average turnaround time than FIFO.
27) A RR scheduler is guaranteed to provide the optimal average turnaround time for a workload.
28) A SJF scheduler requires an oracle to predict how long each job will perform I/O in the future.
29) With a MLFQ scheduler, compute-bound jobs are given higher priority.
30) With a MLFQ scheduler, high priority jobs have longer time-slices than low priority jobs.
31) With a MLFQ scheduler, jobs run to completion as long as there is not a higher priority job.
32) The OS provides the illusion to each process that it has its own address space.
33) The static portion of an address space cannot contain any data.
34) Stacks are used for procedure call frames, which include local variables and parameters.
35) Pointers should not reference the heap.
36) An instruction pointer register is identical to a program counter.
37) A modern OS virtualizes memory with **time-sharing**.
38) A **virtual address** is identical to a **logical address**.
39) With **dynamic relocation**, hardware dynamically translates an address on every memory access.
40) An **MMU** is identical to a **Memory Management Unit**.
41) The OS may not manipulate the contents of an MMU.
42) A disadvantage of **segmentation** is that different portions of an address space cannot grow independently.
43) A disadvantage of segmentation is that **segment tables** require a significant amount of space in memory.
44) With pure segmentation (and no other support), fetching and executing an instruction that performs a store from a register to memory will involve exactly **one memory reference**.
45) Paging approaches suffer from **internal fragmentation**, which grows as the size of a page grows.
46) A **physical page** is identical to a **frame**.
47) The **size** of a virtual page is always identical to the size of a physical page.
48) The **number** of virtual pages is always identical to the number of physical pages.
49) If **8 bits** are used in a virtual address to designate an offset within a page, each page must be exactly **256 bytes**.
50) If a physical address is 24 bits and each page is 4KB, the top 10 bits exactly designate the physical page number.
51) If a virtual address is 16 bits and each page is 128B, then each address space can contain up to 512 pages.
52) A linear **page table** efficiently maps physical page numbers to virtual page numbers.
53) Given a fixed page size, the size of a linear page table increases with a larger address space.
54) Given a constant number of bits in a virtual address, the size of a linear page table increases with larger pages.
55) Given a 28-bit virtual address and 1KB pages, each linear page table will consume $2^{18}$ bytes.
56) Page table entries are stored in the PCB of a process when a context switch occurs.
57) Compared to pure segmentation, a linear page table doubles the required number of memory references.
58) A disadvantage of paging is that it is difficult to track **free** memory.
59) A disadvantage of paging is that all pages within an address space must be allocated.
60) A **TLB** is identical to a **Translation Lookahead Buffer**.
61) A TLB **caches** translations from full virtual addresses to full physical addresses.
62) If a workload sequentially accesses 4096 4-byte integers stored on 256 byte pages, the TLB is likely to have a **miss rate** around $2^8$ (ignore any other memory references).
63) If a workload **sequentially** accesses data, the TLB miss rate will decrease as the page size increases.
64) A workload that sequentially accesses data is likely to have good **temporal locality**, but not necessarily good **spatial locality**.
65) **TLB reach** is defined as the number of TLB entries multiplied by the size of each TLB entry.
66) On a **context switch**, the TLB must be flushed to ensure that one process cannot access the memory of another process.
67) A longer scheduling time slice is likely to decrease the overall TLB miss rate in the system.
68) On a **TLB miss**, the desired page must be fetched from disk.
69) With a TLB, only the outermost page table of each process needs to be accessed.
If the valid bit is clear (equals 0) in a PTE needed for a memory access, the running process is likely to be killed by the OS.

There is a separate page table for every active process in the system.

An inverted page table is efficiently implemented in hardware.

One advantage of adding segmentation to paging is that it potentially reduces the size of the page table.

One advantage of adding paging to segmentation is that it reduces the amount of internal fragmentation.

A single page can be shared across two address spaces by having each process use the same page table.

An advantage of a multi-level page table (compared to a linear page table) is that it potentially reduces the number of required memory accesses to translate an address.

A page directory is identical to the outermost level of the page table.

With a multi-level page table, the complete VPN is used as an index into the page directory.

TLBs are more beneficial with multi-level page tables than with linear (single-level) page tables.

Given a 2-level page table (and no TLB), exactly 2 memory accesses are needed to fetch an instruction.

With a multi-level page table, hardware must understand the format of PTEs.

In the memory hierarchy, a backing store is faster than the memory layer above that uses that backing store.

If the present bit is clear in a needed PTE, then the running process is likely to be killed by the OS.

A page fault is identical to a page miss.

When the dirty bit is set in a PTE, the contents of the TLB entry do not match the contents in the page table.

The OS can run a single process whose allocated address space exceeds the amount of physical memory available in the system.

The OS can run multiple processes whose total allocated address space exceeds the amount of physical memory available in the system.

When a page fault occurs, it is less expensive to replace a clean page than a dirty page.

When a page fault occurs, the present bit of the victim page (i.e., the page chosen for replacement) will be cleared by the OS.

A TLB miss is usually faster to handle than a page miss.

Demand paging is identical to anticipatory paging.

Prefetching of pages helps sequential workloads to avoid page misses.

LRU is an example of a mechanism for determining which page should be replaced from memory.

The OPT replacement policy replaces the page that is used the least often in the future.

LRU always performs as well or better than FIFO.

OPT always performs as well or better than FIFO.

LRU with N+1 pages of memory always performs as well or better than LRU with N pages of memory.

FIFO with N+1 pages of memory always performs as well or better than FIFO with N pages of memory.

LRU-K and 2Q use both how recently and how frequently a page has been accessed to determine which page should be replaced.

The clock policy replaces the least-recently-used page belonging to any process in the system.
Part 2: Multiple-Choice Questions [4 points each]
Assume three jobs arrive at approximately the same time, but Job A arrives slightly before Job B, and Job B arrives slightly before job C. Job A requires 2 sec of CPU, Job B is 8 secs, and Job C is 7 secs. Assume a time-slice of 1 sec.

101) Given a FIFO scheduler, what is the **turnaround time** of job B?
   a. 0 seconds  
   b. 2 seconds  
   c. 8 seconds  
   d. 10 seconds  
   e. None of the above

102) Given a FIFO scheduler, what is the **average response time** of the three jobs?
   a. 1 second  
   b. 2 seconds  
   c. 4 seconds  
   d. 9.67 seconds  
   e. None of the above

103) Given a RR scheduler, what is the **turnaround time** of job B?
   a. 1 second  
   b. 4 seconds  
   c. 16 seconds  
   d. 17 seconds  
   e. None of the above

104) Given a RR scheduler, what is the **average response time** of the three jobs?
   a. 1 second  
   b. 2 seconds  
   c. 3 seconds  
   d. 12.33 seconds  
   e. None of the above

105) Given a SJF scheduler, what is the **turnaround time** of job B?
   a. 2 seconds  
   b. 9 seconds  
   c. 16 seconds  
   d. 17 seconds  
   e. None of the above

106) Given a SJF scheduler, what is the **average response time** of the three jobs?
   a. 2 seconds  
   b. 3.67 seconds  
   c. 9 seconds  
   d. 9.33 seconds  
   e. None of the above
Assume the OS schedules a workload containing three jobs with the following characteristics:

<table>
<thead>
<tr>
<th>Job</th>
<th>Arrival Time</th>
<th>CPU burst</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>B</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>C</td>
<td>12</td>
<td>2</td>
</tr>
</tbody>
</table>

107) Which scheduler minimizes the completion time of the entire workload?
   a. FIFO
   b. RR
   c. SJF
   d. STCF
   e. None of the above

108) Given a STCF scheduler, what is the response time for job B?
   a. 5 seconds
   b. 7 seconds
   c. 10 seconds
   d. 20 seconds
   e. None of the above

109) Given a STCF scheduler, what is the response time for job C?
   a. 0 seconds
   b. 2 seconds
   c. 12 seconds
   d. 14 seconds
   e. None of the above

Assume you have an architecture with 1KB address spaces and 16KB of physical memory. Assume you are performing dynamic relocation with a base-and-bounds register. The base register contains 0x0000037d (decimal 893) and the bounds register contains 506 (decimal). Translate each of the following virtual addresses into physical addresses.

110) Virtual address 0x02e7 (decimal: 743) is physical address:
   a. 0x0000025d0 (decimal: 9680)
   b. 0x00002581 (decimal: 9601)
   c. 0x00003bc7 (decimal: 15303)
   d. Segmentation Violation
   e. None of the above

111) Virtual address 0x01ef (decimal: 495) is physical address:
   a. 0x0000056c (decimal: 1388)
   b. 0x00000e93 (decimal: 3731)
   c. 0x00000d54 (decimal: 3412)
   d. Segmentation Violation
   e. None of the above

112) Virtual address: 0x01a0 (decimal: 416) is physical address:
   a. 0x000000c05 (decimal: 3077)
   b. 0x00000051d (decimal: 1309)
   c. 0x00000051e (decimal: 1310)
   d. Segmentation Violation
   e. None of the above
Assume dynamic relocation is performed with a **linear page table**. Assume the address space size is 16KB, phys mem size is 64KB, and page size is 256 bytes. In a PTE, the high-order bit is the VALID bit. If the bit is 1, the rest of the entry is the PFN. If the bit is 0, the page is not valid. The following are the contents of the page table (from entry 0 down to the max size):

<table>
<thead>
<tr>
<th>Virtual Address 0x175e is physical address:</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. 0x775e</td>
</tr>
<tr>
<td>b. 0xc75e</td>
</tr>
<tr>
<td>c. 0xd85e</td>
</tr>
<tr>
<td>d. Invalid</td>
</tr>
<tr>
<td>e. None of the above</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Virtual address 0x1940 is physical address:</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. 0x7340</td>
</tr>
<tr>
<td>b. 0xa540</td>
</tr>
<tr>
<td>c. 0xf1940</td>
</tr>
<tr>
<td>d. Invalid</td>
</tr>
<tr>
<td>e. None of the above</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Virtual address 0x3b1e is physical address:</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. 0x7b1e</td>
</tr>
<tr>
<td>b. 0xb01e</td>
</tr>
<tr>
<td>c. 0xb00e</td>
</tr>
<tr>
<td>d. Invalid</td>
</tr>
<tr>
<td>e. None of the above</td>
</tr>
</tbody>
</table>
116) When accessing virtual address 0x3457, what will be the first page accessed (decimal)?
   a. 3
   b. 51
   c. 54
   d. 64
   e. Error or None of the above

117) What contents (i.e., value) will be read on the first access (hexademical)?
   a. 0x0d
   b. 0x7f
   c. 0x92
   d. 0xaf
   e. Error or None of the above

118) What will be the second page accessed (hex)?
   a. 0x0a
   b. 0x0c
   c. 0x1e
   d. 0x92
   e. Error or None of the above

119) What is the corresponding final physical address?
   a. 0x197
   b. 0x457
   c. 0x477
   d. 0x497
   e. Error or None of the above

120) What contents (i.e., value) will be read from the final physical address?
   a. 0x05
   b. 0x07
   c. 0x1c
   d. 0x1d
   e. Error or None of the above

121) When accessing Virtual Address 0x5830 what will be the first page accessed (decimal)?
   a. 51
   b. 54
   c. 58
   d. 59
   e. Error or None of the above
Assume the OS is performing page replacement on only 4 pages of physical memory. Assume the following access stream of virtual pages:

Access: 8
Access: 7
Access: 4
Access: 2
Access: 5
Access: 4
Access: 7
Access: 3
Access: 4
Access: 5
Access: 9
Access: 5
Access: 2
Access: 7
Access: 6
Access: 2
Access: 9
Access: 9

122) If the OS uses the OPT replacement policy, how many misses will it incur?
   a. 8
   b. 9
   c. 10
   d. 11
   e. None of the above

123) If the OS uses the FIFO replacement policy, how many misses will it incur?
   a. 8
   b. 9
   c. 10
   d. 11
   e. None of the above

124) If the OS uses the LRU replacement policy, how many misses will it incur?
   a. 8
   b. 9
   c. 10
   d. 11
   e. None of the above

Congratulations on finishing a very long and detailed exam...