CPSC/ECE 3220 – Summer 2017 – Final Exam – Modified Version Name: ______________________

Word Bank for Chapters 9-13

Write one of the words or terms from the following list into the blank appearing to the left of the appropriate definition. Note that there are more words and terms than definitions. (1 pt. each)

absolute path
access control list
cache miss
capability list
copy-on-write
defragment
demand paging
extent
hard link
logical record
mount
not recently used
page coloring
persistent storage
phase change behavior
physical record
prefetch
process migration
relative path
self paging
soft link
thrashing
volume
zero-copy I/O
zero-on-reference

1. ________________________ The unit of data transfer for an application.

2. ________________________ The unit of data transfer for a physical device.

3. ________________________ To bring data into a cache before it is needed.

4. ________________________ An abstraction that corresponds to a logical disk.

5. ________________________ A directory entry that maps a file name to a file number.

6. ________________________ A directory entry that maps a file name to another file name.

7. ________________________ A path name that is interpreted relative to the root directory.

8. ________________________ A list of (object, access rights) tuples held by a user or application.

9. ________________________ A path name that is interpreted relative to the current working directory.

10. _______________________ Abrupt changes in a program's working set, causing bursty cache miss rates.

11. _______________________ A variable-size region of a file that is stored in a contiguous region on a disk.

12. _______________________ A method for clearing memory only if the memory is used, rather than in advance.

13. _______________________ An approach that eliminates the copy across the kernel-user boundary of large blocks of data.

14. _______________________ A list of (user, access rights) tuples held by an object, which may be stored explicitly or in a compressed format.

15. _______________________ The assignment of physical page frames to virtual addresses by partitioning frames based on which portions of cache they will use.

16. _______________________ Using address translation hardware to run a process without all of its memory physically present. When the process references a missing page, the hardware traps to the kernel, which brings the missing page into memory from disk.

NOTE: In order to receive any credit for the short-answer questions, your explanations must go beyond generic, superficial answers such as “better”, “efficient”, “faster”, etc. You must explain your reasoning.

Part 1: Objectives of an Operating System

17. The textbook describes the OS acting in three different roles. Identify at least two distinct actions, features, or services provided by the OS in the role of glue. (2 pts.)

Part 2: The Kernel

Kernel mode / User mode. Circle ***one or both*** of K and U, as applies. (1 pt. each)

18. K / U Interrupts can be permitted to occur in this mode.
19. K / U A store instruction is allowed to execute in this mode.
20. K / U In this mode the processor limits the set of instructions that can be executed.
21. K / U In this mode the processor limits the set of physical memory locations that can be accessed.

22. There are two reasons we discussed as to why a processor would switch to a kernel-owned stack and save the user SP before saving the additional user information. One reason was to protect the information stored from unauthorized access or change. Identify and briefly explain the second reason. (1 pt.)

Part 3: Programs, Process, and Threads

Program / Multithreaded Process / Thread. Circle ***only one*** of P, MTP, or T, as applies. (1 pt. each)

23. P / MTP / T The UNIX fork() system call creates an object of this type.
24. P / MTP / T This object is stored as an executable file after compilation.
25. P / MTP / T This object has a scheduling state (e.g., running, ready, waiting).
26. P / MTP / T This object has a one-to-one association with a SP (stack pointer).
27. P / MTP / T This object has a one-to-one association with a PC (program counter).
28. P / MTP / T In a virtual memory paging system, this object has a one-to-one association with a page table.

Part 4: Concurrency and Synchronization

29. Consider user-mode updates to a shared counter. The shared counter is initialized to zero.

```
        int local;                              int local;
T1S1:   local = shared_counter;         T2S1:   local = shared_counter;
T1S2:   local = local + 1;              T2S2:   local = local + 1;
T1S3:   shared_counter = local;         T2S3:   shared_counter = local;
```

The sequence { T1S1; T1S2; T1S3; T2S1; T2S2 T2S3 } changes the value of the shared counter to 2. Starting over from an initial value of zero, consider the sequence { T1S1; T2S1; T1S2; T2S2; T1S3; T2S3 }. Show the value of the shared counter after this sequence. Is there a problem with this sequence? If so, explain. (1 pt.)

30. Would you suggest using a spinlock to synchronize between the two threads in question 29 above? Why, why not, or under what conditions? (2 pts.)

31. The multiprocessor queueing lock described in the textbook has a class/structure definition as follows:

```
Class Lock{
  private:
    int value = FREE;
    SpinLock splock;
    Queue waiting;
  public:
    void acquire();
    void release();
}
```

The uniprocessor version of the queueing lock disables and later re-enables interrupts to protect updates to the state variables without using a spinlock. What is the purpose of a spinlock being included in the multiprocessor version? (2 pts.)

32. Show how optimistic concurrency control would be used in a thread to increment a shared counter by one using the compare and swap instruction. (2 pts.)

Part 5: Deadlock

33. Give the four necessary conditions for deadlock. (2 pts.)

Deadlock Prevention / Avoidance / Detection and Recovery. Circle ***only one*** of P, A, or D&R. (1 pt. each)

34. P / A / D&R Banker's algorithm.
35. P / A / D&R Rollback and retry using transactions.
36. P / A / D&R Hierarchical ordering of resource requests.

Part 6: Scheduling

FIFO / RR / MFQ / SJF-preemptive. Circle ***one or more*** of F, R, M, S, as applies. (2 pts. each)

37. F / R / M / S Is preemptive.
38. F / R / M / S Allows starvation.

39. Devise a workload when FIFO is optimal for average response time. (1 pt.)

40. From the multiprocessor queueing lock definition shown in question 31 above, each lock has a separate waiting queue. From the discussion in the textbook, each condition variable also has a separate waiting queue. How many ready queues does a typical modern OS have? Explain your answer. (2 pts.)

41. How does priority aging differ from priority donation (also known as priority inheritance)? (2 pts.)

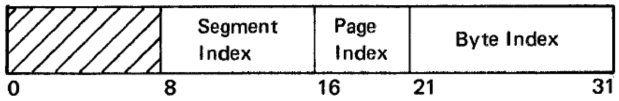
42. When would you use priority donation? (1 pt.)

Part 7: Address Translation and Virtual Memory

Paging / Segmentation. Circle ***one or both*** of P or S, as applies. (1 pt. each)
43. P / S Can have internal fragmentation.
44. P / S Process memory image is divided into variable-length regions that match the program structure.

45. Explain what the term “superpage” means and how the concept helps improve TLB performance. (2 pts.)

An early address translation mechanism for the IBM S/370 used 24-bit virtual addresses with four choices of formats for paged segmentation. Shown below is one of the formats. Except for the rightmost 31, the numbers shown are the leftmost bit positions of the address component fields in a 32-bit word; the segment offset is composed of bits 16-31.



Answer questions 46-47 using powers of 2 for the address format above. Use bytes as the addressable units.

46. Consider a data structure of 256 KB. How many segments do you need to store this structure? (1.5 pts.)

47. How many pages do you need to store the data structure in question 46? (1.5 pts.)

48. You want to implement zero-on-reference pages in a paged virtual memory. However, there is no zero-on-reference bit in the hardware-defined page table entry and no hardware support for this function. Explain the approach you need to take to implement this feature. (5 pts.)

49. Suppose an application is assigned 4 pages of physical memory and the memory is initially empty. It then references pages in the following sequences (using letters for VPN references in the manner of the textbook). Using the diagram format of the textbook, show how the system would fault pages into the four frames of physical memory using the LRU replacement policy. Indicate page hits with a plus sign ("+"). (10 pts.)

REF	**A**	**B**	**A**	**C**	**B**	**D**	**A**	**D**	**E**	**F**	**A**	**G**	**E**	**B**	**A**
1															
2															
3															
4															

Part 8: File, Directories, and Storage Devices

50. Label the following steps that occur in opening an existing file in their proper sequential order, 1-4. (1 pt. each)

____ find the directory entry for the named file

____ initialize the file position pointer in the OFTE to the first byte (or record)

____ check the access permissions and return an error code if the requested access is not allowed

____ create a process-local OFTE and record the access permission under which the file was opened

51. Why are hard links to directory files typically not allowed in a file system? (3 pts.)

52. The time required for a sequence of five reads to random sectors on a disk can take almost five times longer than the time required to read five contiguous sectors. What are the factors involved in disk access that lead to such a disparity? (4 pts.)

53. The textbook states that he Microsoft File Allocation Table can be used as both a data structure that identifies blocks that belong to files as well as free blocks. Explain. (5 pts.)

54. A sparse file has one or more ranges of empty space that are surrounded by file data. Explain why the UNIX FFS inode can easily support sparse files but the Microsoft File Allocation Table cannot. (5 pts.)

55. What is the reason that file systems organize the contents of a directory file as a tree structure rather than a linked list? (3 pts.)

56. In the file system API there is a seek(fileDescriptor, offset) system call. What does this system call do? (3 pts.)

XC. What does it mean to “open” a memory-mapped file? Explain. (up to 3 pts.)