Operating Systems Comprehensive Exam

Fall 2005

Student ID # _____

10/27/2005

You must complete all of part I (60%)

You must complete two of the three sections in part II (20% each)

Part I: You `must` complete `all` of this section.

1. _____ linked libraries can support shared library code, allowing one copy of a library routine to be used by several different processes.

   absolute **relative** **static** **dynamic** **none of these is correct**

2. When it is not known at compile time where a process will reside in memory, _____ code must be generated.

   logical **physical** **absolute** **relocatable**

3. A UNIX process calls *`fork()`* to create a child process as shown: *`pid = fork();`*

   a) What value will be assigned to *`pid`* in the parent process by the call to *`fork()`*?

   the parent's process id **the child's process id** **zero** **none of these**

   b) What value will be assigned to *`pid`* in the child process by the call to *`fork()`*?

   the parent's process id **the child's process id** **zero** **none of these**

4. Memory compaction can be used to minimize the effects of _____ fragmentation.

   internal **external** **neither answer is correct**

5. The Banker's algorithm is used for deadlock _____.

   denial **prevention** **avoidance** **recovery**

6. Belady's anomaly can affect the performance of the _____ page replacement algorithm.

   FIFO **LRU** **optimal** **SJF**

7. _____ access files are made of fixed length records that allow programs to read and write records in no particular order.

   sequential **direct** **logical** **none of these is correct**

8. When an I/O request is being handled for a user's process, which term refers to the policy of returning
   control to the user process before the I/O is completed?

   synchronous I/O **asynchronous I/O** **delayed I/O** **none of these**

9. Which multithreading model requires that a new kernel thread be created for each new user thread?

   many-to-one **one-to-one** **many-to-many** **none of these is correct**

10. A process that does not affect, and is not affected by, another process is referred to as:

static **independent** **cooperating** **dynamic** **unbounded**

Matching: choose the best answer for each question from the list below:

11. _____ File allocation method that stores all file block pointers in one block.

12. _____ Situation that occurs when a process spends more time paging than executing.

13. _____ File allocation method that has each disk block point to the next block in the file.

14. _____ A synchronization tool used to control access to critical sections.

15. _____ The main advantage of using a layered OS design.

A. binding	B. linked allocation
C. increased modularity	D. external
E. deadlock	F. indexed allocation
G. contiguous allocation	H. semaphore
I. thrashing	K. improved performance

Matching: choose the best answer for each question from the list below:

16. _____ is the separation of a user's logical memory from physical memory and is commonly

17. implemented by _____.

18. _____ To reduce the number of entries in a file allocation table, file blocks are grouped into these.

19. _____ Allows I/O devices to transfer data to memory without passing it through the CPU.

20. _____ Occurs when attempting to access a page that is not in memory.

A. clusters	B. device driver
C. page fault	D. limit register
E. demand paging	F. virtual memory
G. threads	H. deadlock
I. fragmentation	K. direct memory addressing
L. buffers	M. compaction

21. Briefly explain what happens during the context switch between the execution of two processes?

22. For the following, indicate which type of fragmentation is possible (circle one):

(a) Memory paging:	**Internal**	**External**	**Both are possible**
(b) Memory segmentation:	**Internal**	**External**	**Both are possible**
(c) Linked file allocation:	**Internal**	**External**	**Both are possible**
(d) Contiguous file allocation:	**Internal**	**External**	**Both are possible**

23. Name the four necessary conditions for a deadlock:

24. List two of the criteria used for comparing CPU scheduling algorithms and explain what they measure:

25. A solution to the critical section problem must satisfy three requirements, name **two** of them:

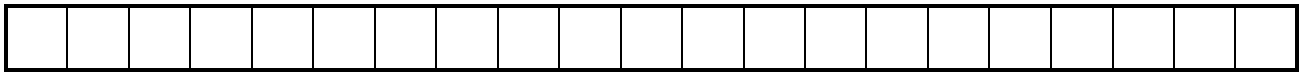
II. You must complete `two` of the following three sections. If you complete more than two sections, clearly indicate which two sections that you want graded. Otherwise, the first two sections will be graded and the third ignored.

A. Given the following set of processes, answer the questions below. Assume that each new process arrives after the interrupted process has been returned to the ready queue. If two processes arrive at the same time, or have the same remaining burst time, schedule them in process number order.

Process Id	Burst Time	Arrival Time
1	5	0
2	3	1
3	6	3
4	4	4

Fill in the following Gantt charts as specified and answer the questions associated with each part. Write the process number of the executing process in the cell for each time unit.

1) **First-Come-First-Served** (non-preemptive):

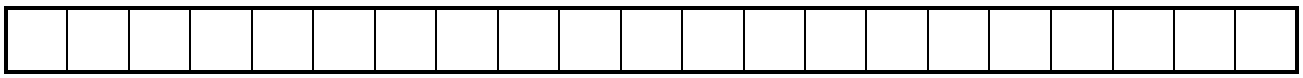


Average waiting time = **Which process had the longest response time _____**

2) **Shortest-job-first** (preemptive):

Average waiting time = **What was the turnaround time for process 1 _____**

3) **Round Robin** (time slice (quantum) is 2 time units):



Average waiting time = **What was the turnaround time for process 4 _____**

B. Given the following list of page references, in execution order:

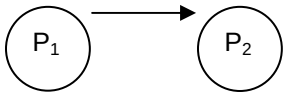
```
5, 1, 3, 4, 5, 2, 3, 2, 5, 1, 4, 2
```

Given the number of available frames shown in the table below, how many page faults will occur for each of the following page replacement algorithms? (All pages are initially empty.)

You must **show your work** to receive partial credit, otherwise only your answers will count.

Algorithm	**# of page faults**
FIFO with 4 frames	
LRU with 3 frames	
Optimal with 3 frames	

C. Assume that $X_{i,j}$ is a semaphore synchronizing processes `i` and `j`, and that all $X_{i,j}$ are initialized to 0. Let S_i represent the statements defining process P_i. On the right of the page, construct a precedence graph showing the order in which the processes will execute. Represent the process P_i with a circle and show a precedence relationship between 2 processes with an arrow.

Example: 

P_1: { wait($X_{5,1}$);
wait($X_{6,1}$);
S_1; }

P_2: { wait($X_{3,2}$);
signal($X_{2,5}$);
S_2;
wait($X_{6,2}$); }

P_3: { S_3;
signal($X_{3,2}$);
signal($X_{3,6}$); }

P_4: { wait($X_{6,4}$);
signal($X_{4,5}$);
S_4; }

P_5: { wait($X_{2,5}$);
S_5;
wait($X_{4,5}$);
signal($X_{5,1}$); }

P_6: { wait($X_{3,6}$);
S_6;
signal($X_{6,1}$);
signal($X_{6,2}$);
signal($X_{6,4}$); }