Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Operating Systems (CS550)**

**Exam 2**

**Part I (20 points).**

Circle either true or false to indicate that the following statements are true or false. Evaluate “truth” in terms of the material covered in this course. The statements below are intended to have an unambiguous answer.

1. **T** F The context switcher may move processes from the run state to the

ready state.

2. **T** F When performing atomic operations such as file I/O, one may use

transaction logging and/or checkpoint restart to reduce the risk of

data loss caused by system failures.

3. **T** F A monitor may provide mutual exclusion at the level of an

function or method.

4. **T** F Shortest Remaining Time Next scheduling is a preemptive policy,

meaning that if a process with a shorter run time than the one currently executing enters the ready queue, the currently running process will be placed in the ready queue and the new process will enter the run state.

5. **T** F MPI\_Irecv is a non-blocking function.

6. **T** F In dynamic scheduling, process and thread priorities may change.

7. **T** F To prevent deadlock in the dining philosophers problem, you may

disallow circular wait.

8. **T** F Mutual exclusion allows for one process to access a shared

resource while all others needing that resource wait.

9. **T** F In CPU scheduling, the scheduler’s main purpose is to shift processes

between the run and ready states.

10. **T** F There are exactly four conditions necessary for deadlock to occur.

**Part II (80 points).**

Read instructions and questions carefully.

1. Describe the difference between binary and counting semaphores, and explain which are needed in the Reader-Writer problem (10 points).

Binary semaphores have a counter that ranges from 0 to 1 (there is exactly one permit) whereas counting semaphores have a counter that ranges from 0 to n (i.e. there are n permits). Therefore, only one process or thread may acquire a binary semaphore whereas n processes or threads may acquire a counting semaphore.

In the Readers-Writers problem, for a Reader priority implementation, two binary semaphores are required. One is to provide mutually exclusive access to the data by Readers and Writers, the other is to protect a counter that tracks the number of Readers accessing the data.

1. Given 5 processes P1, P2, P3, P4, and P5 that start at the same time, having CPU burst times of 72, 11, 30, 68, and 52 nanoseconds, respectively, draw a Gantt chart for those processes using the First Come, First Served scheduling policy (5 points).  
     
   0 72 83 113 181 233

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| P1 | P2 | P3 | P4 | P5 |

1. Assuming the same processes as in problem 4 and sixth process arriving at time 40ns, with a burst time of 40ns, draw a Gantt chart for those processes assuming a Shortest Remaining Time Next scheduling policy (10 points).

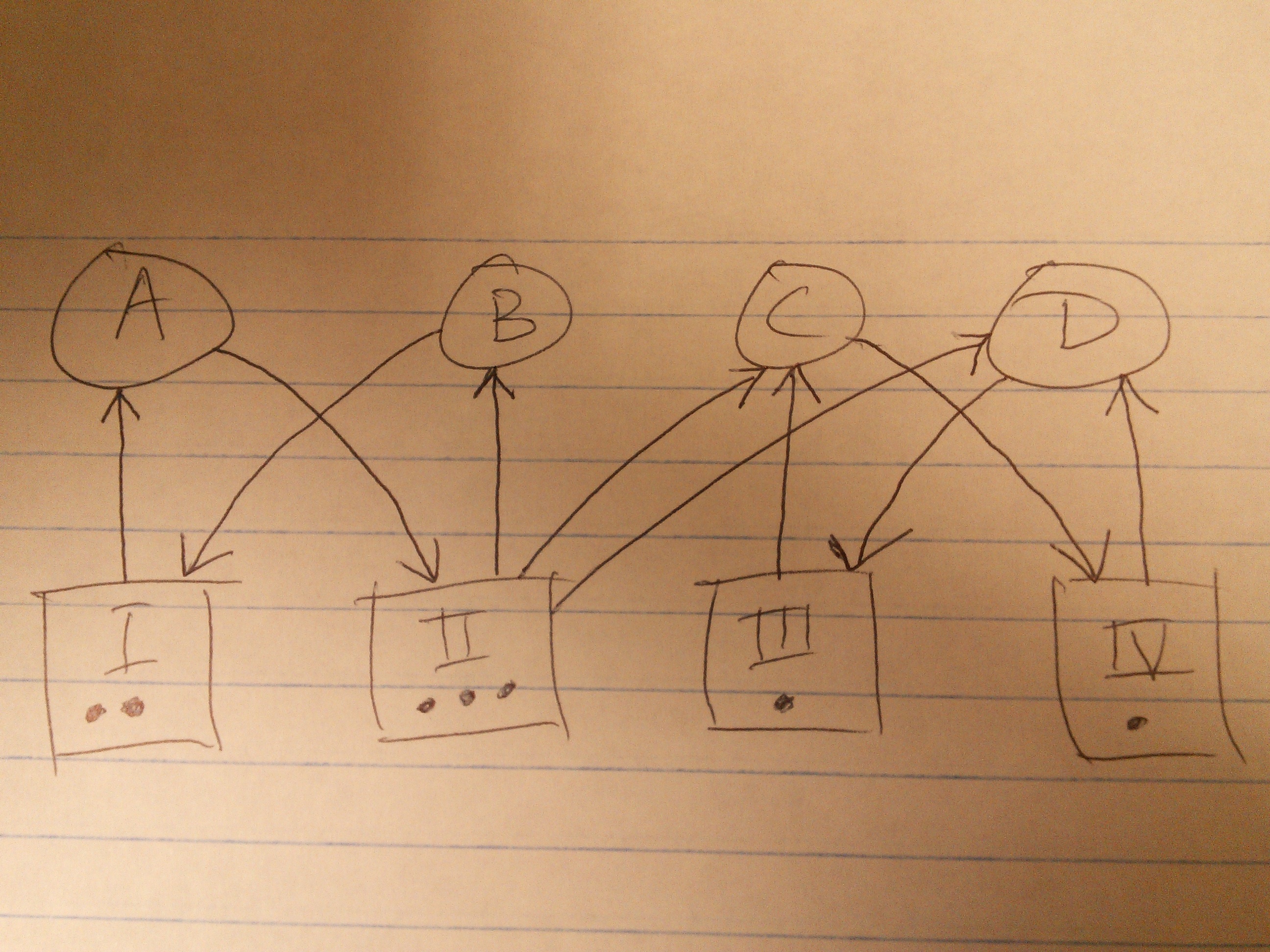
0 11 41 81 133 201 273  
 +---+---+---+---+---+---+  
 | P2| P3| P6| P5| P4| P1|  
 +---+---+---+---+---+---+

1. Briefly explain internal and external memory fragmentation, with respect to paged memory. (5 points).

With respect to paged memory, internal fragmentation may occur on the last page of a process’s address space because a process may not consist of enough instructions or data to fill that page. The reason for this is that a page is the smallest amount of memory that may be reserved for a process by the memory manager. External fragmentation cannot occur in paged memory because the entirety of memory is represented using pages, and any of these may be grouped together in a list to represent a running program.

1. Draw a resource allocation graph for the following situation:   
     
   There are 4 processes, A, B, C, and D.  
   There are 2 resources of type I.  
   There are 3 resources of type II.  
   There is one resource of type III.  
   There is one resource of type IV.   
     
   Process A owns one resource of type I and requests one resource of type II.  
   Process B owns one resource of type II and requests one resource of type I.  
   Process C owns one resource each of type II and III and requests one of type IV.  
   Process D owns one resource each of type II and IV and requests one of type III.

Explain if deadlock exists, assuming mutual exclusion, hold and wait, and no preemption. Justify your answer (10 points).



Processes A and B may run to completion in this example, but deadlock does indeed exist in this example as there is a circular wait between processes C and D as there is exactly 1 unit of resource III and exactly 1 unit of resource IV. Note that process C owns 1 unit of process III and process D owns 1 unit of process IV. Process D needs 1 unit of process III and process C needs 1 unit of process IV. This is circular wait by definition.

1. Explain the differences between journaling and non-journaling file systems (5 points).

Journaling file systems keep a transaction log of modifications to files so that if a rollback to a previous state is required, such an action may be accomplished. Non-journaling file systems, on the other hand, do not provide such a transaction log.

1. Provide pseudocode for the dining philosophers problem with four philosophers. Assume philosophers are represented by threads, and all are created within a main function. Your solution must indicate an appropriate scheduling policy, when threads should be started/joined, and must include appropriate semaphores where necessary. Philosophers should indicate when they are eating (10 points).

sem\_t chopsticks[4];

function philosopher(int tid)

while true

1. Check if the index value of the left chopstick is less than the index value of the right chopstick. If true, do 2 then 3. Otherwise, do 3 then 2.
2. Attempt to acquire the semaphore for the left chopstick, once available, acquire the left chopstick.
3. Attempt to acquire the semaphore for the right chopstick once available, acquire the right chopstick.
4. Eat for specified time period
5. Release semaphores for both chopsticks.
6. Think for a specified time period

end while

end function philosopher

function main

1. Initialize each semaphore for the chopsticks as a binary semaphore.
2. Start threads 0 through 3 using a Round Robin scheduling policy where each thread represents 1 philosopher
3. Join threads 0 through 3

end function main

1. What is the difference between a page and a frame? (5 points)

A page is a logical representation of a contiguous, fixed length, block of memory, which may represent a portion of data or a program whereas a frame is the physical representation of that memory. Both frames and pages generally represent the same amount of data for example 1 frame might be 1KB in size and the page would be the operating system’s abstract (virtual) representation of that memory while the frame is the physical representation in RAM. Pages are often the smallest-sized amount of memory that the OS may reserve on behalf of a running program.

1. Apply the Banker’s algorithm to the following situation. There are four resource types with the total number of system resources expressed as the tuple (1, 2, 3, 2). There are three processes with current allocations of (1, 0, 0, 1), (0, 0, 1, 1), and (0, 2, 0, 0). The maximum claim of each process is (1, 0, 0, 2), (0, 0, 3, 1), and (1, 2, 3, 1). Assume each process can complete its work once it receives its maximum claim. Determine if a safe state can be achieved, being sure to show all work (10 points).

Total resources (1,2,3,2), currently allocated – (1, 2, 1, 2)

P1 allocation (1,0,0,1), max claim (1, 0, 0, 2)

P2 allocation (0,0,1,1), max claim (0, 0, 3, 1)

P3 allocation (0,2,0,0), max claim (1, 2, 3, 1)

(0, 0, 2, 0) is available

Allocate (0, 0, 2, 0) to P2, now P2 can run to completion.

P2 releases its previous allocation of (0, 0, 1, 1) and currently allocated is

(1, 2, 0, 1)

Now, (0, 0, 3, 1) is available.

P1 allocation (1,0,0,1), max claim (1, 0, 0, 2)

P3 allocation (0,2,0,0), max claim (1, 2, 3, 1)

Allocate (0, 0, 0, 1) to P1, and it may run to completion.

P1 releases its previously acquired resources and now, (1, 0, 3, 2) is available.

P3 may now meet its max claim by acquiring (1, 0, 3, 1) resources, and P3 may run to completion, thus the system is in a safe state.

1. Explain how deadlock prevention differs from deadlock avoidance (10 points).

Deadlock prevention is disallowing one of the four conditions (hold and wait, circular wait, no preemption, or mutual exclusion) necessary for deadlock, whereas deadlock avoidance is analyzing the system state to determine if the next set of steps taken by a group of processes will result in a deadlock, and allowing processes to continue with their current actions only if those steps do not result in a deadlock.