University of Illinois at Urbana-Champaign
Department of Computer Science

Midterm Examination

CS 323 Operating System Design
Fall, 1998

9–9:50 am Wednesday October 14
1320 Digital Computer Laboratory

Print your name and ID number neatly in the space provided below; print your name at the upper right corner of every page.

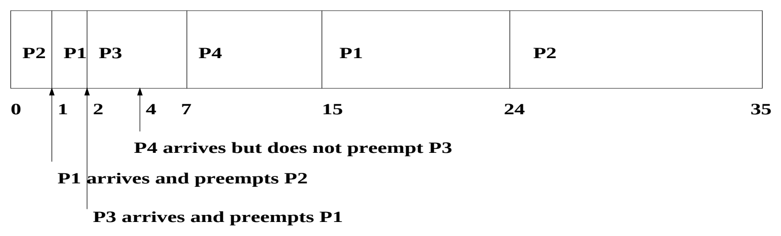
Name:
ID Number:

This is a **closed book, closed notes** exam. You may not use calculators or similar electronic devices.

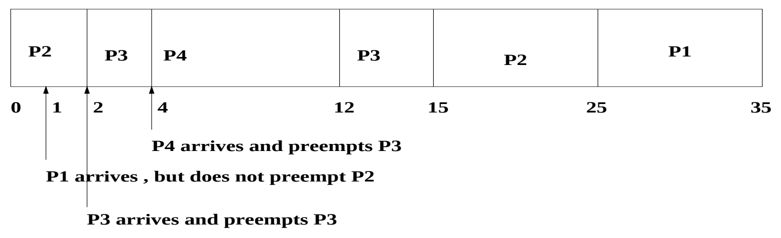
Do all parts of all five problems in this booklet. This booklet should include this title page, plus 5 additional pages (one page per problem). Do your work inside this booklet, using the backs of pages if needed. Problems are of various difficulty degree, hence if you don't know the answer immediately, progress to the following problem and come back to the unsolved problem later.

Problem	Score	Grader
1		
2		
3		
4		
5		
Total		

1 Preemptive Shortest Job First


P2
P1
P3
P4
P1
P2
0
1
2
4
7
15
24
35
P4 arrives but does not preempt P3
P1 arrives and preempts P2
P3 arrives and preempts P1

2. Preemptive Priority


P2
P3
P4
P3
P2
P1
0
1
2
4
12
15
25
35
P4 arrives and preempts P3
P1 arrives , but does not preempt P2
P3 arrives and preempts P3

1 Problem (25 Points)

Consider the workload in Table 1:

Process	**Burst Time**	**Priority**	Arrival Time
P_1	10ms	4	1ms
P_2	12ms	3	0ms
P_3	5ms	2	2ms
P_4	8ms	1	4ms

Table 1: *Process Workload*

1. (10 Points) Provide the schedule (draw Gantt charts) using *preemptive Shortest Job First*, and *preemptive priority*(a smaller priority number implies higher priority).

   Answer:

2. (5 Points) What is the waiting time of each process for the scheduling algorithms above?

   Answer:

```
1. Preemptive Shortest Job First

P1:= 13, P2:= 23, P3:= 0, P4:= 3

2. Preemptive Priority

P1:= 24, P2:= 13, P3:= 8, P4:= 0
```

3. (10 Points) What is the average waiting time of the above specified scheduling disciplines?

   Answer:

```
1. Preemptive Shortest Job First

AWT:= (13 + 23 + 0 + 3)/4 = 39/4 = 9.75ms

2. Preemptive Priority

AWT:+ (24 + 13 + 8 + 0)/4 = 45/4 = 11.25ms
```

2 Problem (15 Points)

Consider two processes P_k and P_l using the following code for their synchronization:

```
repeat
  flag[k]:= true;
  while flag[l] do no-op;
     critical section
  flag[k]:= false;
     remainder section
until false;
```

Does this code satisfy the three requirements for a solution to the critical section problem? If yes, explain why and justify each requirement. If no, explain why not and justify your claim.
Answer:
No, this solution does not satisfy all three requirements. Mutual exclusion is satisfied, but the progress and bounded waiting are not satisfied.
The mutual exclusion is satisfied due to the 'flag' variable followed by the 'while' loop which does not allow both processes enter the critical section at the same time the critical region.
The progress is not satisfied because both processes can wait on the while instruction and do 'no-op' if both set the flag to true at the same time and there is nobody to set the flag to false.
The bounded waiting is not satisfied because even if one process gets to the critical region, sets the flag to false, this process can loop forever in the remainder section, hence violating the bounded waiting condition of the other process.

3 Problem (20 Points)

Consider two resource types A and B, where A has 12 instances and B has 12 instances. Furthermore, consider the following snapshot of a system in Table 2:

	Allocation		**Request**		**Available**	
	A	B	A	B	A	B
P_0	2	0	2	4	4	5
P_1	3	2	8	2		
P_2	1	4	5	3		
P_3	2	1	0	1		
P_4	0	0	4	2		

Table 2: *Resource Snapshot of a System*

1. (10 Points) Determine if this system is in deadlocked state and give a safe sequence if no deadlock exists.

   Answer:

   This system is not in deadlock and the safe sequence is P_0, P_2, P_3, P_4, P_1. The reason is that there is enough available to satisfy all the requests in the specified order. Note that there exist also other safe sequences such as P_4, P_3, P_2, P_0, P_1 or P_0, P_3, P_1, P_2, P_4.

2. (10 Points) Consider modified *request Matrix* in Table 3 with the same Allocation Matrix as above.

	Request	
	A	B
P_0	7	4
P_1	8	2
P_2	5	7
P_3	0	1
P_4	4	2

Table 3: *Modified Request Matrix*

   Determine if the system with the modified Request Matrix is in deadlocked state and give a safe sequence if no deadlock exists.

   Answer:

   This system is in deadlocked state because: P_0, P_1, P_2 can't get requested resources because there is not enought available resources. P_3 can get enough resources, and after releasing its own resources, the available is (6,5). P_4 can satisfy the request because there is enough available, however this process does not release any resources, hence, there is not enough available to run $P_0, P_1 or P_2$.

4 Problem - 20 Points

1. (10 Points) Consider two level 16-bit paging architecture. Assume 32 bytes page size. Specify the format of the logical address, i.e. how many bits are allocated to which part within the logical address.

   Answer: logical page in two level paging system has the format (p_1, p_2, d). p_1 represents the displacement within the page of the page tables, p_2 represents the displacement within the actual page table, and d represents the offset within the page. p_1 takes 7 bits, p_2 takes 4 bits, d takes 5 bits ($2^5 = 32$). d takes 5 bits because the page is 32 bytes, p_2 takes 4 bits because if we page the page table each piece of the page table will have 2^4 entries of 2 byte (16-bit) physical addresses. p_1 will have 7 bits : 16-9=7 (16-bit logical address).

   Note that the assumption is that logical and physical addresses are 16 bit (16-bit paging architecture).

2. Explain how is the translation done from a logical address to a physical address in segmentation hardware (no paging should be considered in this translation).

The (s,d) address will be translated as follows: First $STBR$ address will be taken which includes the start of the segment table. The we take s which represents the offset in the segment table. In the segment table entry we will find the limit of the segment and the beginning of the physical base address f. The hardware checks the d parameter if the offset is within the segment limit. If this is not true, then trap of OS is triggered, If the d parameter is within the segment limit, then we procceed and create the physical address which consist of the physical base address f in the segment table and the offset within the segment d. The physical address will be $f + d$.

5 Problem - 20 Points

1. Consider two processes P and Q which are communicating directly. What sequence of instructions should P and Q execute to achieve reliable communication from P to Q?

   Answer:

```
P                              Q
---------------------------------------------
send(Q, msg);                    receive(P, msg);
set timeout
receive(Q,msg);                  reply(P,msg);
if timeout then re-send(Q,msg);
```

2. Consider two processes P and Q which communicate via mailboxes; from P to Q via mailbox A and from Q to P via mailbox B. Assume mailbox B is initially empty. Assume reliable communication link between processes P and Q. What is the sequence of instruction P should execute in order for P to determine that Q terminated?

   Answer:

```
P
----------------------------
send(A,msg);
set timeout
receive(B,msg);
if timeout then Q-terminated;
```