CS4650 – FALL 1999 – FINAL EXAM

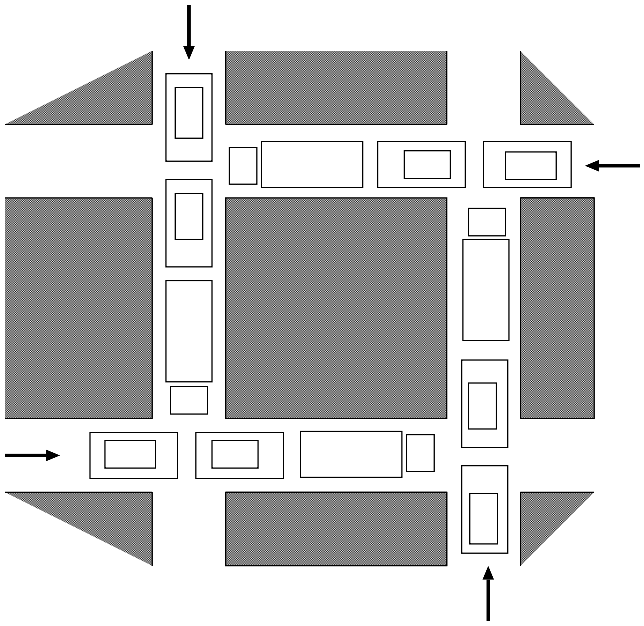
Malek Mouhoub
(12/15/99)

Exercise 1 : short questions

1. An I/O device interrupts when it needs service. When an interrupt occurs, what are the different methods used by the operating system to determine which I/O device caused the interrupt ? Explain briefly each method.
2. What is a thread ? what are the differences between multi-treading and multi-processing ? how are threads handled in UNIX, LUNIX and WINDOWS NT ?
3. What advantage is there in having different time-quantum sizes on different levels of a multilevel queuing system ?
4. What makes LINUX different from UNIX ?

Exercise 2

Consider the traffic deadlock depicted by the figure below.



1. Show that the four necessary conditions for deadlock indeed hold in this example.
2. State a simple rule that will avoid deadlocks in this system.

Exercise 3

Consider the following snapshot of a system :

	Allocation	*Max*	*Available*
	ABCD	ABCD	ABCD
P_0	0012	0012	1520
P_1	1000	1750	
P_2	1354	2356	
P_3	0632	0652	
P_4	0014	0656	

1. Does this situation correspond to a deadlock avoidance or deadlock detection ?
2. Is it possible to use a graph representation to solve this problem ?
3. What is the content of the matrix *Need* ?
4. Is the system in a safe state ?
5. If a request from process P_1 arrives for (0,4,2,0) can the request be granted immediately ?
6. Suppose that you have coded the deadlock-avoidance safety algorithm and now have been asked to implement the deadlock-detection algorithm. Can you do so by simply using the safety algorithm code and redefining $Max = Waiting + Allocation_i$, is a vector specifying the resources process i is waiting for, and $Allocation_i$ is as defined in the course.

Exercise 4

1. What are the requirements a solution to the critical section problem must satisfy ?
2. For each solution (program in C) presented in the following pages, check if the requirements are satisfied.

Solution 1

```
turn =1;
p1()
{
    for(;;) {
        while (turn ==2);
        Crit1();
        turn =2;
        Remainder1();
    }
}
```

```
p2()
{
    for(;;) {
        while (turn ==1);
        Crit2();
        turn =1;
        Remainder2();
    }
}
```

Solution 2

```
c1 = c2 =1;
p1()
{
    for(;;) {
        while (c2 ==0);
        c1 = 0;
        Crit1();
        c1 =1;
        Remainder1();
    }
}
```

```
p2()
{
    for(;;) {
        while (c1 ==0);
        c2 = 0;
        Crit2();
        c2 =1;
        Remainder2();
    }
}
```

Solution 3

```
c1 = c2 =1;
p1()
{
    for(;;) {
        c1 = 0;
        while (c2 ==0);
        Crit1();
        c1 =1;
        Remainder1();
    }
}
```

```
p2()
{
    for(;;) {
        c2 = 0;
        while (c1 ==0);
        Crit2();
        c2 =1;
        Remainder2();
    }
}
```

Solution 4

```
c1 = c2 =1;
turn = 1;
p1()
{
    for(;;) {
        c1 = 0;
        while (c2 ==0)
            if (turn == 2) {
                c1 = 1;
                while (turn == 2);
                c1 = 0;
    }
        Crit1();
        turn = 2;
        c1 =1;
        Remainder1();
    }
}
```

```
p2()
{
    for(;;) {
        c2 = 0;
        while (c1 ==0)
            if (turn == 1) {
                c1 = 1;
                while (turn == 1);
                c2 = 0;
    }
        Crit2();
        turn = 1;
        c2 =1;
        Remainder2();
    }
}
```

Exercise 5

Consider the two-dimensional array A :

```
var A : array[1..100] of array[1..100] of integer;
```

where A[1][1] is at location 200, in a paged memory system with pages of size 200. A small process is in page 0 (locations 0 to 199) for manipulating the matrix; thus, every instruction fetch will be from page 0.
For three pages frames, how many pages faults are generated by the following array-initialization loops, using LRU replacement, and assuming page frame 1 has the process in it, and the other two are initially empty :

```
a:  for j:= 1 to 100 do
      for i:=1 to 100 do
        A[i][j] := 0;

b:  for i:= 1 to 100 do
      for j:=1 to 100 do
        A[i][j] := 0;
```

Exercice 6

Consider the following page reference string :

1,2,3,4,2,1,5,6,2,1,2,3,7,6,3,2,1,2,3,6

How many pages faults would occur for the following replacement algorithms, assuming one, four and seven frames ? Remember that all frames are initially empty, so your first unique page will all cost one page each.

- LRU replacement
- FIFO replacement
- Optimal replacement