**Assignment 6**

**COMP3300**

1. Consider the following resource-allocation policy. Requests and releases for resources are allowed at any time. If a request for resources cannot be satisfied because the resources are not available, then we check any processes that are blocked, waiting for resources. If they have the desired resources, then these resources are taken away from them and are given to the requesting process. The vector of resources for which the waiting process is waiting is increased to include the resources that were taken away. For example, consider a system with three resource types and the vector Available initialized to (4, 2, 2). If process P0 asks for (2, 2, 1), it gets them. If P1 asks for (1, 0, 1), it gets them. Then, if P0 asks for (0, 0, 1), it is blocked (resource not available). If P2 now asks for (2, 0, 0), it gets the available one (1, 0, 0) and one that was allocated to P0 (since P0 is blocked). P0’s Allocation vector goes down to (1, 2, 1), and its Need vector goes up to (1, 0, 1).
   1. Can deadlock occur? If so, give an example. If not, which necessary condition cannot occur?

Deadlock cannot occur because there is pre-emption.

* 1. Can indefinite blocking occur?

Yes, indefinite blocking can occur. A process may never acquire all the resources it needs if they are continuously pre-empted by a series of requests such as those of process C.

1. A computer has six tape drives, with n processes competing for them. Each process may need two drives. For which values of n is the system deadlock free?

With three processes, each one can have two drives. With four processes, the distribution of drives will be (2,2,1,1), allowing the first two processes to finish. With five processes, the distribution of drives will be (2,1,1,1,1). which still allows the first one to finish. With six processes, each holding one tape drive, and wanting another one, we have a deadlock. Thus, for n < 6, the system is deadlock free.

1. Jane and John are getting divorce. To divide their property, they have agreed on the following algorithm: Every morning, each one may send a letter to the other's lawyer requesting one item of property. Since it takes a day for letters to be delivered, they have agreed that if both discover that they have requested the same item on the same day, the next day they will send a letter canceling the request. Among their property are their dog, Woofer, Woofer's doghouse, their canary, Tweeter, and Tweeter's cage. The animals love their houses, so it has been agreed that any division of property separating an animal from its house is invalid, requiring the whole division to start over from scratch. Both Jane and John desperately want Woofer. In order for them to go on (separate) vacations, each spouse has programmed a personal computer to handle the negotiating. When they come back from vacation, the computers are still negotiating. Why? Is deadlock possible and why? Is starvation possible and why? Discuss.

If both programs ask for Woofer first, the computers will starve with endless sequence: request Woofer, cancel request, request Woofer, cancel request, etc. If one of them asks for the doghouse and the other asks for the dog, we have a deadlock, which is detected by both parties and then broken, but it is just repeated on the next cycle. Either way, if both computers have been programmed to go after the dog or the doghouse first, either starvation or deadlock ensues. There is not really much difference between the two here. In most deadlock problems, starvation does not seem serious because introduction random delays will usually make it very unlikely. That approach does not work here.

1. Consider a system that runs 5000 jobs per month. The IT management would like to install a deadlock-avoidance algorithm (like the banker's algorithm) on the system but it is proven that as a result, the average execution time per job will increase by about 10% and the average turnaround time will increase by about 20%. However, management argues that since the machine currently has 30% idle time, all of the 5000 jobs per month could still be executed without any problem.
   1. What are the arguments for installing the deadlock-avoidance algorithm?

An argument for installing deadlock avoidance in the system is that we could ensure deadlock would never occur. In addition, despite the increase in turnaround time, all 5,000 jobs could still run.

* 1. What are the arguments against installing the deadlock-avoidance algorithm?

An argument against installing deadlock avoidance software is that deadlocks occur infrequently, and they cost little when they do occur.

1. Consider a swapping system in which memory consists of the following hole sizes in memory order: 10 KB, 4 KB, 20 KB, 18 KB, 7 KB, 9 KB, 12 KB, and 15 KB. Which hole is taken for successive segment requests of
2. 12 KB
3. 10 KB
4. 9 KB

for (a) first fit, (b) best fit, (c) worst fit, and (d) next fit.

First fit: (a) 20KB (b) 10KB (c) 18KB

Best fit: (a) 12KB (b) 10KB (c) 9KB

Worst fit: (a) 20KB (b) 18KB (c) 15KB

Next fit: (a) 20 KB (b) 18KB (c) 9KB