

Operating Systems exercise – week 6 – Selected answers.

Deadlock 2.

1. Consider a system consisting of four resources of the same type that are shared by three processes, each of which needs at most two resources. Show that the system is deadlock-free.

Answer.

Suppose the system is deadlocked. This implies that each process is holding one resource and is waiting for one more. Since there are three processes and four resources, one process must be able to obtain two resources. This process requires no more resources and therefore it will return its resources when done.

2. One method of deadlock prevention by denying the ‘wait for’ condition requires that processes must request all of the resources they will need before the system will allow them to compete for resources. The system grants resources on an ‘all or none’ basis. Discuss the advantages and disadvantages of this method.

No answer

3. What is meant by imposing a total ordering on the resources when attempting to prevent deadlock by denying the fourth necessary condition.

Answer.

Each resource is assigned a unique number such that resources needed typically by processes early are given low numbers, and resources needed later are given higher numbers.

4. What is a safe state?

Answer.

question and answer to be included in portfolio that is to be handed in.

5. Consider the following snapshot of a system:

	Allocation				Max				Available			
	A	B	C	D	A	B	C	D	A	B	C	D
P0	0	0	1	2	0	0	1	2	1	5	2	0
P1	1	0	0	0	1	7	5	0				
P2	1	3	5	4	2	3	5	6				
P3	0	6	3	2	0	6	5	2				
P4	0	0	1	4	0	6	5	6				

Answer the following questions using the banker’s algorithm:

- a) What is the content of the matrix Need?
- b) Is the system in a safe state?

c) If a request from process P1 arrives for (0,4,2,0), can the request be granted immediately?

Answer.

question and answer to be included in portfolio that is to be handed in.

6. 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).

a) Can deadlock occur? If so, give an example. If not, which necessary condition cannot occur?

b) Can indefinite blocking occur?

Answer.

a) Deadlock cannot occur because preemption exists.

b) Yes. A process may never acquire all the resources it needs if they are continuously preempted by a series of requests such as those of process C.

Simple Windows system programming.

14) Now we are in a position to modify the program above. We are going to change it so that now instead of copying a file from one file to another, the code will copy a file but output it to the console window. So it is a little like the DOS Type command.

To do this you need to know about one further function.

GetStdHandle – this returns a HANDLE and takes as an argument nStdHandle a DWORD that is the numerical value of the standard input/output stream. The permissible values are:

STD_INPUT_HANDLE – for keyboard input

STD_OUTPUT_HANDLE – for console output

STD_ERROR_HANDLE – error output (normally the console)

These identify which of the standard I/O values you need to use. You can now modify the code to send the file contents to the console screen.

a) Use WriteFile with the appropriate HANDLE you now get from GetStdHandle.

b) NOTE – YOU NEED TO DELETE (OR COMMENT OUT) THE CODE THAT CLOSSES THE hOut handle. This is because it will close the console for output and so you won't get ANY further output to the console output.

c) You will have to change the number of arguments in the if statement that guards the usage prompt as the program will now be invoked with only one parameter, as

FileCopy <file to be copied to screen>.

No answer – question and answer to be included in portfolio that is to be handed in.