**Assignment 3: Deadlock Avoidance**James Miller  
Grand Canyon University  
CST 315: Operating Systems  
Dr. Citro  
March 7, 2023

**GitHub Link**

**Detailed description of the scenario explaining your approach to implementation in C:**

To prevent a deadlock, this program uses a mutex and a timer to make sure that only one thread can access the shared resource at once. When several processes are in a deadlock and waiting for each other to release resources that they require, the program freezes. This program releases the mutex and waits for a random period before trying to acquire the resource again if a process is unable to do so because it is already in use. This avoids a deadlock by allowing other processes to access the resource.

**Assessment of Efficiency:**

The program successfully prevents deadlocks by permitting processes to wait for a random amount of time before trying to obtain the resource again. This method prevents numerous programs from becoming blocked while waiting for one another, allowing the program to continue operating efficiently. This approach may have draw backs, if the waiting time is too lengthy, the program's overall performance may suffer; if the waiting time is too brief, it may not be enough to avoid deadlocks. Additionally, if there are too many processes waiting for the same resource, the waiting time may become excessive, causing the program to slow down or even crash. If problems were identified with this strategy, an alternative approach could be to use a resource allocation algorithm that prevents deadlocks from occurring. A better way is the Banker's Algorithm, which ensures that resources are allocated in such a way that no deadlock can occur. This algorithm requires the program to keep track of the current resource allocation and the maximum resources that each process can request, and it can be used to prevent deadlocks from occurring in a more efficient and reliable manner.

**FlowchartDiagram

Description automatically generated**

**References**

*Handling deadlocks*. (2020, December 27). GeeksforGeeks. https://www.geeksforgeeks.org/handling-deadlocks/