1. What is race condition?

* A race condition occurs when two or more threads can access shared data and try to change it at the same time. Sicne the thread scheduling algorithm can swap between threads at any time, the order in which the threads will attempt to access the shared data is not known. Therefore, the result of the change in data is dependent on the thread scheduling algorithm. Hence the term "race", because both threads are "racing" to access/change the data.

1. Why race condition is difficult to reproduce and debug?

* Race condition is difficult to reproduce and debug because the result at the end of race conditions depends on the relative timing and nondeterministic between threads. When running the debugger, the problems that may occur in production may disappear. “Heisenbug” in race condition, which is a computer bug that disappears or alters its characteristics when an attempt is made. Therefore, in order to avoid race condition, caution must be taken during the software designing instead of fixing

1. Summaries the Parallel Programming Patterns section.

* Parallel Programming Patterns are little tricks to writing repetitive code. This trick are widely use in the work place because they work in practice. As such, these tricks have been heavily documented in an effort to help need programmers as well as veterans. The two main types of patterns are strategies and concurrent execution mechanisms. Strategist like algorithmic and implementation allow you to choose what tasks can be done by what processes and the structure of the program or the data it meant to be computing. The major CEMs are Process or Thread control, deciding how processes are controlled at run time, and Coordination which decide how task will work together to complete a computation. Coordination does this by message passing between processes on one multiprocessor or on multiple machines, or through mutual exclusion treads on one memory system.

1. Compare the following:

Collective synchronization (barrier) with Collective communication (reduction)

Master-worker with fork join

* With Collective synchronization with Collective communication several task will converge to a single point and will not proceed until all tasks have reached it. Master-worker with fork join will divide a master process into several worker processes that will fork off and compute data before combining again.