Operating System (CS-33101)
MCA – 3rd Semester
Assignment 2

Q.1. Consider the process state transition diagram with two suspend states. Suppose that it is time for the OS to dispatch a process and that there are processes in both the Ready state and the Ready/Suspend state, and that at least one process in the Ready/Suspend state has higher scheduling priority than any of the processes in the Ready state. Two extreme policies are as follows: (a) Always dispatch from a process in the Ready state, to minimize swapping, and (b) always give preference to the highest-priority process, even though that may mean swapping when swapping is not necessary. Suggest an intermediate policy that tries to balance the concerns of priority and performance.

Q.2. Consider the following code and explain how many processes will be created. Also, explain what will be the output of following code:

```
#include <stdio.h>
#include <unistd.h>
int main()
{
    if (fork() && (!fork())) {
        if (fork() || fork()) {
            fork();
        }
    }
    printf("2 ");
    return 0;
}
```

Q.3. Provide two programming examples in which multithreading does not provide better performance than a single-threaded solution.

Q.4. Consider a multiprocessor system and a multithreaded program written using the many-to-many threading model. Let the number of user-level threads in the program be more than the number of processors in the system. Discuss the performance implications of the following scenarios.

(a) The number of kernel threads allocated to the program is less than the number of processors.

(b) The number of kernel threads allocated to the program is equal to the number of processors.

(c) The number of kernel threads allocated to the program is greater than the number of processors but less than the number of user-level threads.