# Name: Aman

# 2019014

## Task 1:

**fork():** The fork() system call creates a clone of our program(process), which becomes the child of the caller program. So these parent and child processes will have identical copies of the computer's address space, code, and stack.

**pthread_create():** The pthread_create() system call creates a new thread in the program and both of these threads have same process. Both threads will have shared memory and they can communicate with each other. The second thread will share open files, data, current working directory, user and group ID's, signal handlers and signal dispositions. But the new thread won't share it's stack, thread ID, and registers.

### Code explanation:

The fork() system call code: In this case both parent and child process have different memory space so both will get global value as 10. In case of parent process value of global variable is increasing linearly by 1. It goes up to 100, ie 10,11,12,…..100. And in case of child process as it has different memory so it will have global value as 10 and it will be decreasing linearly up to -90, ie 10,9,8,7…..-90.

The pthread_create() system call code : In case of pthread_create(), parent thread is increasing the value of global from 10 to 100 linearly i.e. 10, 11, 12……100 and then child thread is running and decreasing the value of global linearly from 100 to -90 i.e. 100, 99, 98, 97,…. ,-90. As in case of pthread both the threads share the same memory therefore global started decreasing from the point where the other thread left it.

Some time we are noticing here context switching. In that case if the parent thread started first and let say it went up to 'x' i.e. 10, 11, 12, 13,….x and we notice a context switching and child process goes to 'y' i.e. x, x-1, x-2, x-3, ….. y.

This process continues until one of the thread completes its while loop and after that the second thread completes its process and we have joined these threads to let both the threads complete their task.