1. Output after replacing exit with return
   * Parent sees i = 0
   * Parent sees i = 1
   * Child sees i = 1
   * Parent sees i = 2
   * Child sees i = 2
   * Child sees i = 0
2. Output after running the program. (replacing exit with return, including the header file pthread.h and commenting out the while(1) loop which made program go into infinite wait state)

* Parent says a: 1
* id: 1 a: 2 b: 1
* id: 2 a: 3 b: 1

Thread 1 and 2 complete

1. It can prove to be a good idea as the context switching becomes fast owing to the fact that system calls are now avoided. Also, it gets best of user and kernel level implementations as many short lived user level threads are mapped to a big pool.

m>>n: - If the ULT is mapped on to the KLT than it’s a highly favorable condition. System call as well as overhead decreases.

m>n: - The use of multi-threading for applications is favorable and reasonable.

m=n: - This type of condition cannot be considered as multi-threading because for one user process there is only one kernel process. So it should not be preferred.

m<n: - In this type of condition, reasonable and overhead increases.

m<<n: - This condition is highly not preferable as system call and overhead increases drastically.