SINGAPORE POLYTECHNIC
SCHOOL OF ELECTRICAL & ELECTRONIC ENGINEERING
ET0023 OPERATING SYSTEMS

TUTORIAL 4 – Threads

1. Using a table, distinguish between a Program, Process and a Thread.
   Include a section on what is required to create them.

2. Threads are considered as Light-weight processes.
   What is the information that is required to be stored by a thread? For each of the information type stored, explain the reason why it needs to be stored.

3. In this problem, you are to compare reading a file using a single-threaded file server and a multi-threaded server.
   It takes 15 msec to get a request for work, dispatch it and do the rest of the necessary processing, assuming that the data needed are in the block cache (called a Cache Hit).
   If a disk operation is needed, as is the case one-third of the time, an additional 75 msec is required, during which time the thread sleeps (called a Cache Miss).

   How many requests per second can the server handle if it is single threaded?
   How many requests per second can the server handle If it is multi-threaded?

   Hints:
   i. Calculate the average time for an access (1/3 of the time is a cache hit, 2/3s of the time is a cache miss)
   ii. Divide 1sec by the average time and you will get the number of requests/sec.
   iii. Consider what happens on a single threaded server (similar to a single process)
   iv. Consider what happens on a multi-threaded server (the threads work in parallel)

4. What are the benefits of using threads in a multi-programming environment?
   Illustrate your answers using a HTTP server that answers requests using threads.

5. Explain the use of the functions: thread_create, thread_wait, thread_yield and thread_exit .
   Who would use these functions? The programmer or OS ?

6. Use Google to search for the Linux man pages of exec(), fork() and clone().
   Briefly explain their functions and differences.

   Note:
   Q5 and Q6 will give you an idea of how processes and threads are implemented in the Linux OS, through function calls.
   You will not be required to write any C/C++ code to demonstrate this, however, looking through examples will give you good ideas on how it can be achieved.