**CG2271 Lab 6 Report**

1. Most tasks are written in endless loops. The tasks state determines when they get to run.
2. app\_main() gets executed the moment osThreadNew(app\_main, NULL, NULL) is called.
3. The thread/task state of the app\_main() is in the running state.
4. The state of app\_main() goes to the blocked state
5. Yes, since currently there is only 1 thread. However, the use of osDelay() is to free up the CPU to do other important tasks. Normal delay is still making use of the CPU to execute code in order to generate the required delay.
6. The changes that need to be made are
   1. osThreadNew(app\_main, NULL, NULL) -> osThreadNew(led\_red\_thread, NULL, NULL);
   2. void app\_main (void \*argument) -> void led\_red\_thread (void \*argument)
7. we observed that the colour of the LED is yellow. This is due to led\_red\_thread and led\_green\_thread use of os\_delay(). Hence the program toggles between the red and green LEDs at high frequency which results in only a yellow colour being observed. (might be red led or green led only due to deadlock. Deadlock occurs in this scenario as there is no pre-emption since both the threads are of same priority. Hence, they are not able to be de-allocated or forcibly removed.)
8. Yes.
9. Only 1 colour of LED either red or green is on. This is due to OS\_ROBIN\_ENABLE being set to 0 which means that RTX is unable to switch to tasks that are in READY state