Assignment 2

Output:

1. "matrix task" using most of the CPU utilization because of its complex computation.
2. The priority of communication tasks needs to be increased to meet the timing constraint
3. When the priority of "communication task" is increased, the "matrix task" takes longer to complete.
4. "matrix task" is greater than 1 second.

Code:

```
//-----------------Task Handler-----------------------------
static void priority_task()
{
    while (1)
    {
        printf("Matrix Execution Time: %d \n", matrix_execution_time);
        fflush(stdout);
        if (matrix_execution_time > 1000)
        {
            vTaskPrioritySet(communication_handle, 4);
            printf("Communication task set to priority 4 \n");
            fflush(stdout);
        }
        else if (matrix_execution_time < 200)
        {
            vTaskPrioritySet(communication_handle, 2);
            printf("Communication task set to priority 2 \n");
            fflush(stdout);
        }
        vTaskDelay(10);
    }
}

/*-----------------------------------------------------------*/
```

```
//------------------------- Assignment 2---------------------


   // Created Task

   xTaskCreate((pdTASK_CODE)matrix_task, (signed char*)"Matrix", 1000, NULL, 3, &matrix_handle);
   xTaskCreate((pdTASK_CODE)communication_task, (signed char*)"Communication", configMINIMAL_STACK_SIZE, NULL, 1, &communication_handle);
   xTaskCreate((pdTASK_CODE)priority_task, (signed char*)"Priority", configMINIMAL_STACK_SIZE, NULL, 1, &priority_task_handle);



//------------------------- Assignment 2---------------------
   vTaskStartScheduler();                                  // Initilized the Scheduler
```

```
void vApplicationTickHook(void)
{
    /* This function will be called by each tick interrupt if
    configUSE_TICK_HOOK is set to 1 in FreeRTOSConfig.h.  User code can be
    added here, but the tick hook is called from an interrupt context, so
    code must not attempt to block, and only the interrupt safe FreeRTOS API
    functions can be used (those that end in FromISR()). */

#if ( mainCREATE_SIMPLE_BLINKY_DEMO_ONLY != 1 )
    {
        vFullDemoTickHookFunction();
    }
#endif /* mainCREATE_SIMPLE_BLINKY_DEMO_ONLY */



if (matrix_handle == xTaskGetCurrentTaskHandle())
{
    matrix_execution_time++;
}
else
{
    matrix_execution_time = 0;
}

}
/*-----------------------------------------------------------*/
```