



**Problem Set:** Assignment 04  
**Points:** 10  
**Date Set:** See Slate  
**Course:** CS2006 Operating Systems

**Semester:** Spring 2022  
**Due Date:** See Slate  
**Instructor:** Dr. Nauman

Note: Code in the following is intentionally left low-quality. Please write the code yourself.

## 1 Threads and Race Conditions

1. The following code deals with basic thread creation.

(a) Write the following program and observe and *explain* the output.

```
1  #include <stdio.h>
2  #include <pthread.h>
3  #include <stdlib.h>
4
5  void* thread1() {
6      for (int c = 0; c < 10; c++)
7          printf("Hello\n");
8  }
9
10 void* thread2() {
11     for (int c = 0; c < 10; c++)
12         printf("World\n");
13 }
14
15 int main() {
16     int status;
17     pthread_t tid1, tid2;
18
19     pthread_create( &tid1, NULL, thread1, NULL );
20     pthread_create( &tid2, NULL, thread2, NULL );
21
22     pthread_join( tid1, NULL );
23     pthread_join( tid2, NULL );
24
25     return 0;
26 }
```

(b) Modify the program to create four threads using the same two functions (thread1 and thread2)

(c) Run both versions and include screenshots of the output.

Note To compile the program, you need to include the following switch in the compilation command: `-lpthread`. This tells gcc to use the registered pthread library.

2. Take a look at the following code:

```
1  /* Includes */
2  #include <unistd.h>    /* Symbolic Constants */
3  #include <sys/types.h> /* Primitive System Data Types */
4  #include <errno.h>     /* Errors */
5  #include <stdio.h>     /* Input/Output */
6  #include <stdlib.h>    /* General Utilities */
7  #include <pthread.h>   /* POSIX Threads */
8  #include <string.h>    /* String handling */
9
10 #define NUM_RUNS 1000000
11
12 /* prototype for thread routine */
13 void handler ( void *ptr );
14
15 int counter; /* shared variable */
16
17 int main() {
18     int i[2];
19     pthread_t thread_a;
20     pthread_t thread_b;
21
22     i[0] = 0; /* argument to threads */
23     i[1] = 1;
24
25     pthread_create (&thread_a, NULL, (void *) &handler, (void *) &i[0]);
26     pthread_create (&thread_b, NULL, (void *) &handler, (void *) &i[1]);
27
28     pthread_join(thread_a, NULL);
29     pthread_join(thread_b, NULL);
30
31     printf("-----\n");
32     printf("Final counter value: %d\n", counter);
33     printf("Error:                %d\n", (NUM_RUNS*2-counter));
34     exit(0);
35 }
36
37 void handler ( void *ptr ) {
38     int iter = 0;
39     int thread_num;
40     thread_num = *((int *) ptr);
41     printf("Starting thread: %d \n", thread_num);
42
43     while(iter < NUM_RUNS) {
44         counter++;
45         iter += 1;
46     }
47
48     printf("Thread %d, counter = %d \n", thread_num, counter);
49     pthread_exit(0); /* exit thread */
50 }
```

(a) Compile and execute the program.

(b) Answer the following questions:

- i. What should be the value of the counter variable at the end?
- ii. What is the value you get?
- iii. How large is the error and how much does it vary on different runs?
- iv. How much *user time* (roughly) does the program take to run on your system?

## 2 Submission

Provide screenshots of everything you have done in one PDF file. Name of the file must be your NU-ID. Also make sure you write your name, course section and ID on the first page of the PDF. If you do not follow these instructions, you may not get any credit for your work.

Submit the assignment of Slate. No other medium of submission is acceptable.