Wayne State University

CSC 4421 - Winter 2020
Computer Operating Systems Labs
Lab 06 - Threads

Instructor: Rui Chen - section 001

Points Possible: 100

Tasks

Read the man page of the following functions `pthread_create`, `pthread_join`.

Read the man page of the following functions `pthread_mutex_init`, `pthread_mutex_lock`, `pthread_mutex_trylock`, `pthread_mutex_unlock`, `pthread_mutex_destroy`.

For each task, fullfill the requirements provided in the comments, or fill the blank.

Compile the code and make sure it is executable. What is the output of the code?

```
#include<pthread.h>
#include<stdio.h>

#define NUM_THREADS 5

void *PrintHello(void *threadid)
{
    long tid;
    tid = (long)threadid;
    printf("Hello World! It's me, thread #%ld!\n", tid);
    pthread_exit(NULL);
}
int main (int argc, char *argv[])
{
  // create an array of thread struct instances with appropriate length

    long t;
    for(t=0; t<NUM_THREADS; t++){
        printf("In main: creating thread %ld\n", t);
        // start a new thread and call the appropriate routine with. You need to handle
              errors.
        // args of the routine should be cast as (void *)t
    }
    /* Last thing that main() should do */
    pthread_exit(NULL);
}
```

Task1.c

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>

void *print_message_function( void *ptr );

main() {
    pthread_t thread1, thread2;
    char *message1 = "Thread 1";
    char *message2 = "Thread 2";
    int iret1, iret2;

    /* Create independent threads each of which will execute function */
    iret1 = pthread_create(&thread1, NULL, print_message_function, (void*) message1);
    iret2 = pthread_create(&thread2, NULL, print_message_function, (void*) message2);

  // use thread join function to wait for the thread thread1 to terminate

    // do the same for thread2

  // print the return value of each thread

    return(0);
}

void *print_message_function( void *ptr ) {
    char *message;
    message = (char *) ptr;
    printf("%s \n", message);
}
```

Task2.c

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>

void *functionC();

pthread_mutex_t mutex1 = PTHREAD_MUTEX_INITIALIZER;
int counter = 0;

main() {
    int rc1, rc2;
    pthread_t thread1, thread2;
    /* Create independent threads each of which will execute functionC */
    if((rc1=pthread_create( &thread1, NULL, __________ , NULL))) {
        printf("Thread creation failed: %d\n", rc1);
    }
    if((rc2=pthread_create( &thread2, NULL, __________ , NULL))) {
        printf("Thread creation failed: %d\n", rc2);
    }
    pthread_join( __________ , NULL);
    pthread_join( __________ , NULL);
    return(0);
}
void *functionC() {
    pthread_mutex_lock( __________ );
    counter++;
    printf("Counter value: %d\n",counter);
    pthread_mutex_unlock( __________   );
}
```

Task3.c