**Multithreading – Thread Creation Assignments**

1. Refer code in “simple\_thread.c”.

a. Modify the thread function to read and return username read from the user. Allocate heap memory for the user name and free in caller after displaying it.

b. Display the thread id’s of parent and child thread

A screen shot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screen shot of a computer

Description automatically generated

2. Refer the code “pthread\_creation.c”. Modify the existing functions as below.

a. Main()- read a line of text and pass to new threadproc function below

threadproc() – to create 2 child threads to count words and to count characters, display the received values, return the values to the caller

Other thread functions to be used by threadproc() are given below

startroutine1()—to count words and return word count to caller

startroutine2() – to return the character count to the caller

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

3. Refer the program in “thread\_prg.c”. Complete the TBD sections , check the final solution for memory leak if any

**Memory Allocation for fun\_thread\_args\_t**:

fun\_thread\_args\_t \*args = calloc(1, sizeof(fun\_thread\_args\_t));

**Error Handling**:

if (args == NULL) {

perror("Memory allocation failed");

return NULL;

}

if (rc != 0) {

fprintf(stderr, "Error creating thread: %d\n", rc);

free(args);

return NULL;

}

**Main Function**:

if (\_launch\_fun\_thread(&fun\_thread1, 0xABCDEF12345678ULL, "super FUN thread 1") == NULL) {

fprintf(stderr, "Failed to create thread 1\n");

return EXIT\_FAILURE;

}

A screenshot of a computer screen

Description automatically generated

A screen shot of a computer program

Description automatically generated

4. Write a program

a. to read a set of words as command line arguments and to create an array of threads (Consider a maximum of 5 words )

b. process each word using an separate thread. Let each thread append “\_ed” to the input word and return to main thread

c. main thread to wait for completion of each thread, retrieve returned string display with thread number, free memory

A screenshot of a computer program

Description automatically generated

A screen shot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated