**Multithreading – Thread Creation assignments**

Mandatory

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

Ans:

**🡪Modify the thread function** to accept user input (username) and allocate memory dynamically for it. Return the username string back to the caller.

**🡪Parent and child thread IDs**: Print the thread IDs of both the parent (main thread) and the child (new thread).

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

Ans:

🡪**Main function**: Read a line of text and pass it to a new function threadproc.

🡪**threadproc()**: Create two child threads:

* One thread counts the words.
* Another thread counts the characters.

🡪**Helper threads**: Define two functions to count words (startroutine1()) and characters (startroutine2()).

🡪**Return values**: Return the counts from the threads to the caller (main thread).

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

Ans:

🡪Complete the program where parts are missing (TBD).

🡪Use valgrind or similar tools to check for memory leaks.

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

Ans:

🡪Parse the command-line arguments.

🡪Create an array of threads (maximum 5).

🡪For each word, create a separate thread to append "\_ed".

🡪After processing, the main thread waits for the threads to complete and displays the result.

A screen shot of a computer program

Description automatically generated