Threads – Exercise Lab 2

1. Write a program that creates 10 threads. Have each thread execute the same function and pass each thread a unique number. Each thread should print “Hello, World (thread n)” five times where ‘n’ is replaced by the thread’s number. Use an array of pthread t objects to hold the various thread IDs. Be sure the program doesn’t terminate until all the threads are complete.

2. Write a program that computes the square roots of the integers from 0 to 99 in a separate thread and returns an array of doubles containing the results. In the meantime the main thread should display a short message to the user and then display the results of the computation when they are ready.

3. WAP that takes an array and its size as arguments. Later, try to use Merge sort to sort the array elements. As, in Merge sort, input array is divided into two halves, use two different threads to sort the two sub-arrays, recursively. Every thread would create two new threads, and would wait for them to complete. At the end, every thread would combine the sorted result of its child threads.

4. Suppose you are building a C++ string class that you intend to use in a multithreaded program. You are worried about your string objects possibly getting corrupted if they are updated by more than one thread at a time. You consider adding a mutex as a member of each string and locking that mutex whenever any string method is called. Discuss the implications of this design. Write a main program to test your C++ String class.