CSCI 4220 Lab 4

Lab 4: Threading and Recursive Add

Due Wednesday October 7th, 2020

In this lab you will practice using POSIX threads (`pthreads`) to do some simple recursive addition.

First consider the recursive `add()` function. If we were not using threads, this function would look like `int add(int a, int b) { b ? return 1 + add(a,b-1) : return a; }` and would return the sum of any number `a` plus any non-negative number `b`. This is a highly condensed version of the function, I recommend you write it out on multiple lines to be more readable.

Your task is to use threads in a manner similar to the fib_thread.c example from lecture to add every combination of numbers [1…(MAX_ADDAND-1)] to every combination of numbers [1 … MAX_ADDAND] using `add()` and output the results **after** creating all computation threads. Remember that `pthread_create` only takes one argument as a function, so you'll have to do something with a `struct`. You should not wait for a thread to finish before creating another thread. You should collect and display all results in the order the threads were made.

You may also want to make use of `pthread_self()` to get the current thread's ID. There is no networking component to this lab.

Submission

Submit a single C file called `lab4.c` with your team's solution. The book code will not be included for this lab.

Sample output

Example output with `MAX_ADDAND` (argument to program) set to `5`
(note that this is not the same as just counting 1-25):

```
???@???:~/Teaching/NetProgF19/longlabs$ gcc lab4.c -lpthread
???@???:~/Teaching/NetProgF19/longlabs$ ./a.out 5
Main starting thread add() for [1 + 1]
Main starting thread add() for [1 + 2]
Thread 140170751706880 running add() with [1 + 1]
Main starting thread add() for [1 + 3]
Thread 140170743252736 running add() with [1 + 2]
Main starting thread add() for [1 + 4]
Thread 140170734798592 running add() with [1 + 3]
Main starting thread add() for [1 + 5]
Thread 140170726344448 running add() with [1 + 4]
Main starting thread add() for [2 + 1]
Thread 140170717890304 running add() with [1 + 5]
Main starting thread add() for [2 + 2]
Thread 140170709436160 running add() with [2 + 1]
Main starting thread add() for [2 + 3]
Thread 140170700982016 running add() with [2 + 2]
Main starting thread add() for [2 + 4]
Thread 140170692527872 running add() with [2 + 3]
```

```
Main starting thread add() for [2 + 5]
Thread 140170684073728 running add() with [2 + 4]
Main starting thread add() for [3 + 1]
Thread 140170675619584 running add() with [2 + 5]
Main starting thread add() for [3 + 2]
Thread 140170667165440 running add() with [3 + 1]
Main starting thread add() for [3 + 3]
Thread 140170658711296 running add() with [3 + 2]
Main starting thread add() for [3 + 4]
Thread 140170650257152 running add() with [3 + 3]
Main starting thread add() for [3 + 5]
Thread 140170641803008 running add() with [3 + 4]
Main starting thread add() for [4 + 1]
Thread 140170633348864 running add() with [3 + 5]
Main starting thread add() for [4 + 2]
Thread 140170624894720 running add() with [4 + 1]
Main starting thread add() for [4 + 3]
Thread 140170616440576 running add() with [4 + 2]
Main starting thread add() for [4 + 4]
Thread 140170607986432 running add() with [4 + 3]
Main starting thread add() for [4 + 5]
Thread 140170599532288 running add() with [4 + 4]
Thread 140170591078144 running add() with [4 + 5]
In main, collecting thread 140170751706880 computed [1 + 1] = 2
In main, collecting thread 140170743252736 computed [1 + 2] = 3
In main, collecting thread 140170734798592 computed [1 + 3] = 4
In main, collecting thread 140170726344448 computed [1 + 4] = 5
In main, collecting thread 140170717890304 computed [1 + 5] = 6
In main, collecting thread 140170709436160 computed [2 + 1] = 3
In main, collecting thread 140170700982016 computed [2 + 2] = 4
In main, collecting thread 140170692527872 computed [2 + 3] = 5
In main, collecting thread 140170684073728 computed [2 + 4] = 6
In main, collecting thread 140170675619584 computed [2 + 5] = 7
In main, collecting thread 140170667165440 computed [3 + 1] = 4
In main, collecting thread 140170658711296 computed [3 + 2] = 5
In main, collecting thread 140170650257152 computed [3 + 3] = 6
In main, collecting thread 140170641803008 computed [3 + 4] = 7
In main, collecting thread 140170633348864 computed [3 + 5] = 8
In main, collecting thread 140170624894720 computed [4 + 1] = 5
In main, collecting thread 140170616440576 computed [4 + 2] = 6
In main, collecting thread 140170607986432 computed [4 + 3] = 7
In main, collecting thread 140170599532288 computed [4 + 4] = 8
In main, collecting thread 140170591078144 computed [4 + 5] = 9
```