Simon Myhre

Lab6

1.
a)
The expected output of the program is 10 threads being created in a for loop. Where it will run in the *PrintMessage(). If it goes wrong it the pthread_create it will return -1. Then it will return the error code shown in res.

b)
First the thread prints out in order from 0-9. That's because the for loop goes from 0-9. But then there are suddenly 10 threads that is performing in *printMessage. Depending on what thread is the fastest it will print in that order. If that made sense. If 9 is faster than 8, the 9 will print before 8 .The randomness is a result of multi-threading cpu, kernel and hardware. So it prints kind of randomly. The id is printed because of tid points to the threadID. The way to make them print in order is by using locks, mutexes and conditions.

2.
a)
The output is very similar as 1. But the problem is solved with the randomness. Sleep(2) (waiting for 2 seconds) has been added. That means the computer has time to do one thread at the time. Join was added. This was keep main thread waiting before the other threads are done

b)
And that explains the output. Because it waits, it has time to perform one thread at the time. Now it will be performed in order, because the computer manages to complete each thread before the next one starts.
Another reason the program works in the correct order is because of the join. This makes the main not do the next create thread until the other thread is done

3.
Exercise 2:
a)
The expected output of the program is that first the Thread1Print runs. Then Thread2Print runs. Then the main. The output should be: "Im thread 1" "Im thread 2" "Im thread 0". Or, maybe because the programmer hasn't included join. So maybe he wanted it in the order thread 0, 1 and then 3.

b)
The output is "Im a thread 0". This means thread 1, and 2 hasent been done it task. That was to print the printf in the thread1/2prints functions.

Simon Myhre

Exercise 3:
a)
The output now is:
Im thread 2
Im thread 1
Im thread 0

b)
Yes. Because main waits for both thread 1 and 2, It will proceed as expected. The meaning of the NULL is that is not expected any return from the functions. This could have been a (void*) &variable that could point to the value of the return