COMP1521 Week 10

YAY Last Week of Term :DDD

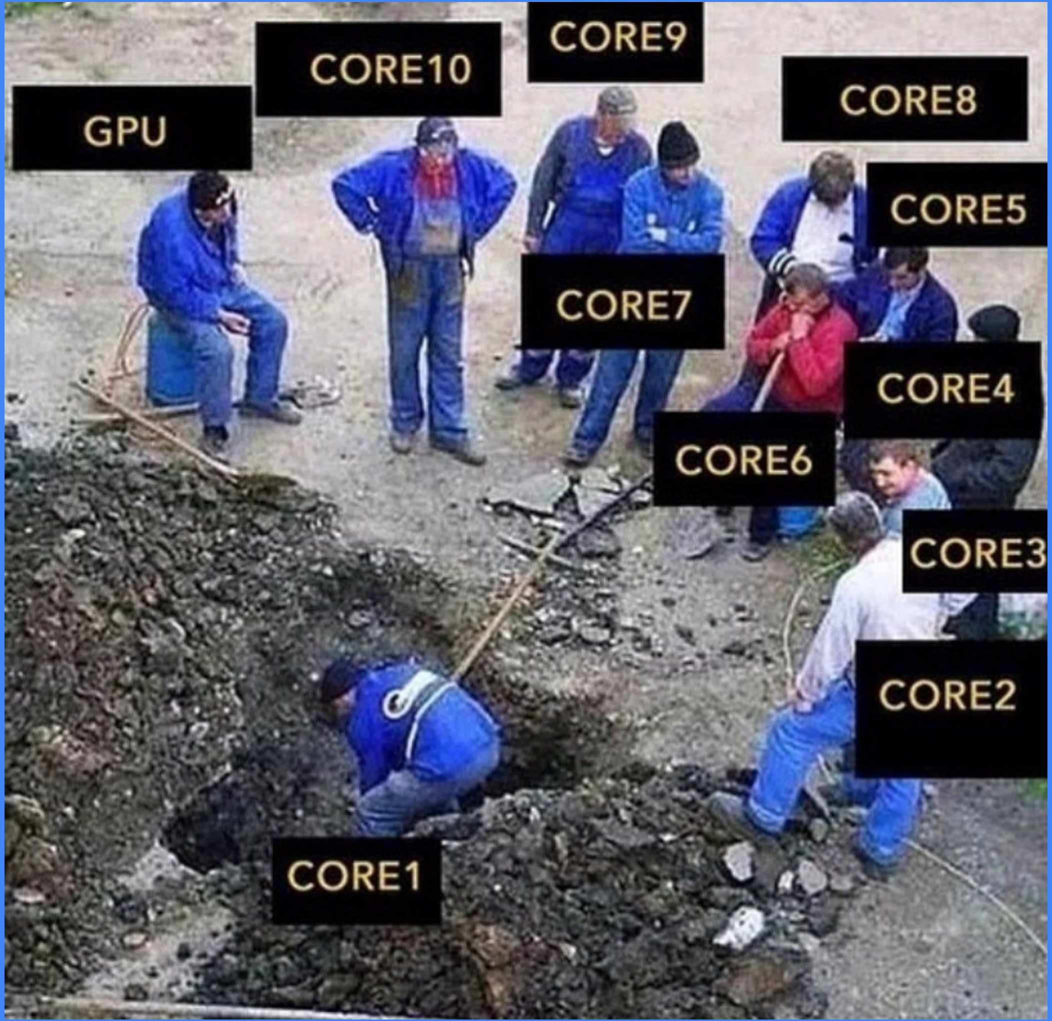
Admin stuff

- Help sessions if you need help with the assignment
- My experience Due Soon
- Prac exam during lab session
- weekly test this week as well

Overview

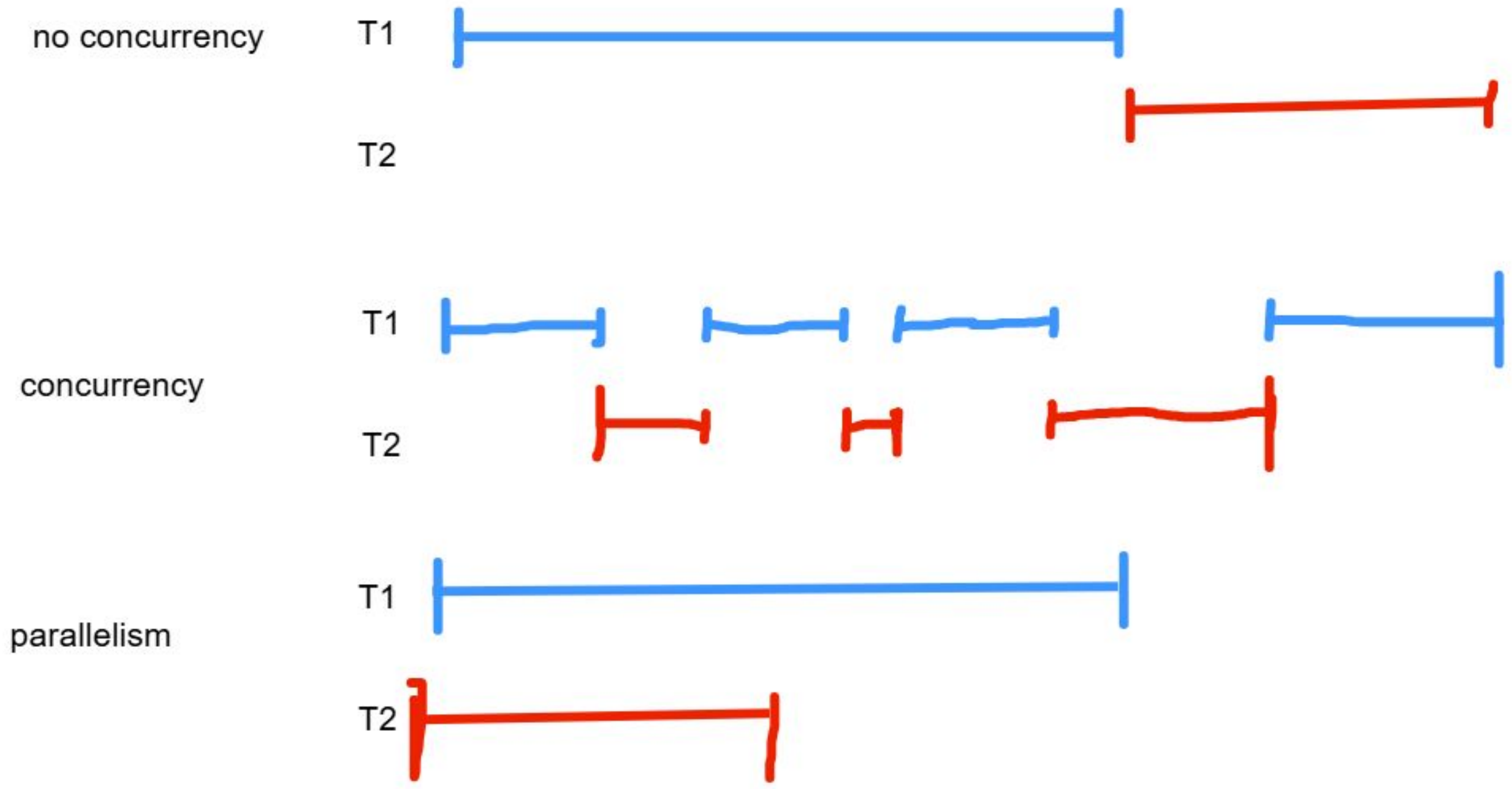
- Q2
- Concurrency and Parallelism
  - Q3
- POSIX Threads (pthreads)
  - Q4-7
- Combined Questions for above topics
  - Rest of questions

Concurrency vs Parallelism


CORE9
CORE10
GPU
CORE8
CORE5
CORE7
CORE4
CORE6
CORE3
CORE2
CORE1

Tutorial Question 3

What is the difference between concurrency and parallelism?


no concurrency
T1
T2
concurrency
T1
T2
parallelism
T1
T2

Concurrency

- A very general concept that refers to code running with multiple threads of execution
- It may or may not be executing simultaneously
- In the context of utilising one core, concurrency is the ability to pause and switch between tasks (threads)
  - Slower than completing than no concurrency as it takes time to pause

Parallelism

- A specific utilisation of concurrency such that multiple threads of execution are running truly in parallel
- Usually utilised to increase program performance.
- This means parallelism is not possible on a computer without multi-core or multi-data capabilities.

Food for thought

Why do we use concurrency in single core systems even if it’s slower?

POSIX Threads

Tutorial Question 3

How can we make our programs concurrent in C?

Tutorial Question 4

Write a C program that creates a thread that infinitely prints some message provided by main (eg. "Hello\n"), while the main (default) thread infinitely prints a different message (eg. "there!\n").

Tutorial Question 5

The following C program attempts to say hello from another thread:

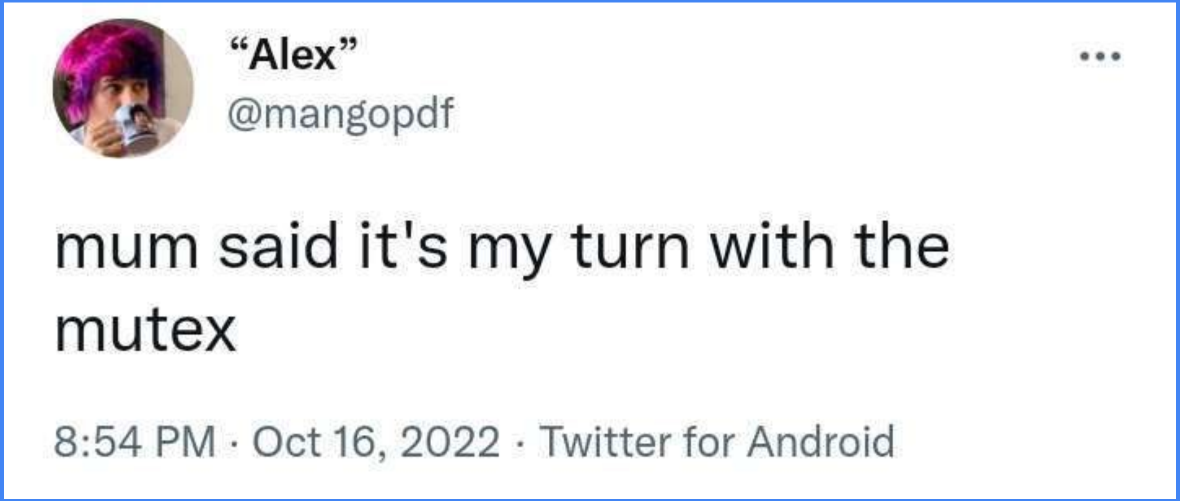
```c
#include <stdio.h>
#include <pthread.h>

void *thread_run(void *data) {
    printf("Hello from thread!\n");

    return NULL;
}

int main(void) {
    pthread_t thread;
    pthread_create(
        &thread,    // the pthread_t handle that will represent this thread
        NULL,       // thread-attributes -- we usually just leave this NULL
        thread_run, // the function that the thread should start executing
        NULL        // data we want to pass to the thread -- this will be
                    // given in the `void *data` argument above
    );

    return 0;
}
```

Tutorial Question 6

Write a C program that creates a thread which infinitely prints the message "feed me input!\n" once per second (sleep), while the main (default) thread continuously reads in lines of input, and prints those lines back out to stdout with the prefix: "you entered: ".

Tutorial Question 7

The following C program attempts to increment a global variable in two different threads, 5000 times each.


“Alex”
@mangopdf

mum said it's my turn with the mutex

8:54 PM · Oct 16, 2022 · Twitter for Android

THANKS FOR BEING SUCH AN AMAZING CLASS