1. Processes and threads
   1. Explain the difference between a process and a thread.
      1. A process is an execution of a sequence of instructions, also known av a program. A thread is created by the process to run the program. The thread is able to create multiple thread to run the same program code but runs in different places in the code at different times.
   2. **Describe a scenario where it is desirable to:**
   * Write a program that uses threads for parallel processing
     1. Mainly used when running a program where all the processing units share the same memory.
   * Write a program that uses processes for parallel processing
     1. Mainly used when the processing units store there own memory, and information is passed between them.
   1. . Explain why each thread requires a thread control block (TCB).
      1. Thread Control Block (TCB) is a data structure in the operating system kernel which contains thread-specific information needed to manage it. The TCB keeps track of the stack information, the registry and thread metadata
   2. What is the difference between cooperative (voluntary) threading and pre-emptive (involuntary) threading? Briefly describe the steps necessary for a context switch for each case.
      1. In cooperative threading the operating system never initiates a context switch from running a process to another. Instead the processes yields control when idle, or periodically. Hence the name voluntary threading. The opposite, pre-emptive threading means that the operating system is allowed to step in and do a context switch from one thread to another at any time.
2. C program with POSIX threads
3. The function "go" is called, but the threads are created beforehand in pthread\_create. In "go" it prints out the "Hello from thread" and creates an exit value.
4. Creating at thread and scheduling it are two different things, and we can only make the assumption that the different threads are running on their own virtual processors with unpredictable speed. Therefore, any interleaving is possible.
5. When the program starts, a main thread begins running main., That thread creates 10 threads. The minimum has to be 2 because we dont know if the other threads have completed before thread 8 prints "Hello", so thread 8 and the main thread are the minimum. The maximum is that thread 8 runs first and therefor there will be 11 threads as a maximum.

1. The use of pthread\_join makes the loop check for completion. The function waits for threads to finish, then returns the value passed to pthread\_exit by that thread.
2. All other threads than thread 5 would finish, and thread five would be the last to return.
3. When pthread\_join returns for thread X, thread X is in terminated state