* multithreading is when you ask a computer to do things in parallel
  + It is not completely multi thread on a one CPU computer but it should still work
* And python isn’t a completely true multi threader because of its Global Interpreter Lock
* The problems that may come with multi threading is that you are asking threads to get the data
  + You have to make sure that it is doing this correctly or else things will get jumbled up
* To tell the computers what to do you need to have synchronization primitives
  + These are simple programming tricks to make all the threads working in synchrony
* Things that are used in multithreading are Locks, RLocks, Semaphores, Events, Conditions, and Barriers
* Locks are the simplest form of synchronization in python
  + It has two states locked and unlocked
  + It is created in an unlocked state and has two principal methods acquire and release
  + Acquire methods locks the lock and blocks the execution until the release method in some other coroutine sets it to unlocked
* The lock again and returns a tree
  + The release should only be called in the locked state
* If the release is called in the unlocked state you will receive an RunTimeError