327: Object-oriented programming

Lecture 22

11/29/2021

Professor Barron

Concurrent programming

- Process
  - Everything needed to run
  - process id
  - virtual address space
  - code
  - handles to open files (and other resources)
  - security context
  - environment variables
  - at least one main thread

Concurrent programming

- Thread
  - Scheduled for execution on a core of the CPU (by the OS)
  - shared virtual address space
    - with other threads in the same process
  - execution context
    - program counter
    - machine registers
    - stack

Python GIL

- Global interpreter lock
- Only one thread can execute bytecode at a time
- Prevents race conditions and ensures thread safety
- Necessary for Cpython’s memory management
- Specific to Cpython
- more detail here
  - https://wiki.python.org/moin/GlobalInterpreterLock

Challenges in concurrent programs

- Synchronizing access to memory (and other resources)
- Where and how to split up the work?
- Waiting for resources or conditions
  - reducing advantage of parallelism
  - deadlock
- Non-determinism
- Overhead
  - context switching
  - communicating between processes
- Some programming patterns are helpful in managing these challenges

In Python…

- Use threads if the task is I/O heavy
- Use processes if the task is CPU heavy
- Use neither if the performance or maintenance overhead outweighs the benefits