Concurrent Programming: *Introduction*

Concurrent Programming is Hard!

- The human mind tends to be sequential
- The notion of time is often misleading
- Thinking about all possible sequences of events in a computer system is at least error prone and frequently impossible
- *n* concurrent flows, $\binom{n}{2} + \binom{n}{3} + \cdots + \binom{n}{n-1} = O(2^n)$

Concurrent Programming is Hard!

- **Classical problem classes of concurrent programs:**
  - ***Races:*** outcome depends on arbitrary scheduling decisions elsewhere in the system
    - Example: who gets the last seat on the airplane?
    - Example from previous lecture: shell child exits before added to list
  - ***Deadlock:*** improper resource allocation prevents forward progress
    - Example: traffic gridlock in 4-way stop
    - Example in async-*SIGNAL*-safety: printf (locks buffer), signal, printf
  - ***Livelock / Starvation / Fairness***: external events and/or system scheduling decisions can prevent sub-task progress
    - Example: people always jump in front of you in line
    - Example in kernel: BAD scheduler schedules process with lowest PID
- **Many aspects of concurrent programming are beyond the scope of our course… but not all!**