Round Robin Scheduling

- Circular queue
- 10 – 100 ms is typical timeslice quantum (q)
- Worst case wait: $D(n,q) = (n-1) \cdot q$
    - $\lim_{q \to \infty} D(n,q) = \infty$
        - Becomes FCFS
    - $\lim_{q \to 0} D(n,q) = 0$
        - Looks like each process runs at the same time
- It is the fairest algorithm

What causes switching?

- Time slice expires (preemption interrupt)
- CPU burst ends with a wait on child or signal or I/O
- Process terminates

80% of CPU bursts should be less than q
The context switch time should be no more than 5% of q - $(0.05)q$

In CS-391, car in a box, we will probably want to go back to round robin scheduling.

Multilevel Queue Scheduling

Ready processes are classified algorithmically (dynamically) and placed into **multiple prioritized ready queues.**

Classic Example

Higher Priority: Foreground process (FG or FPG) ready queue
A process that has the user's attention. An active window or set of windows with which the user is interacting or a parent window and all its children.
Lower Priority: Background process ready queue
Running outside the user's attention
System level processes

The foreground processes always take priority. This, like prioritized scheduling, can result in starvation and thus some sort of aging must be implemented.

Modern Example

Highest: system processes
Critical to keep the system running. Belog to the OS
interactive user processes
Using in foreground of behavioral interaction with the computer
network processes
editing processes
batch processes
Things that can take a long time to run; large jobs; no instant feedback