OS CPU Managment

"I only need it for just a second, really…"

OS Manages Resources

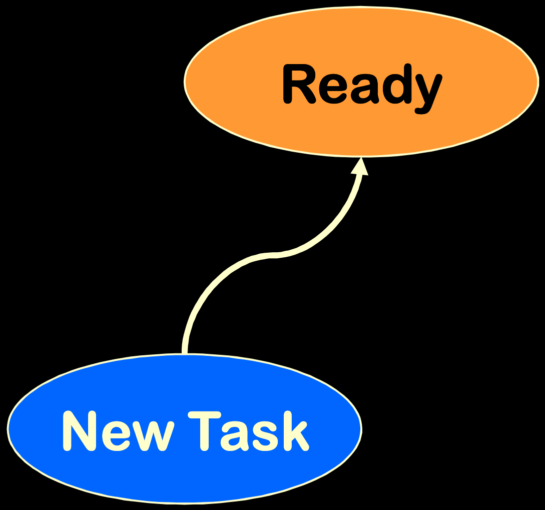
- Memory
- **CPU**
  - **Processes**
- I/O Devices
- Information
  - Files

Sharing Nicely

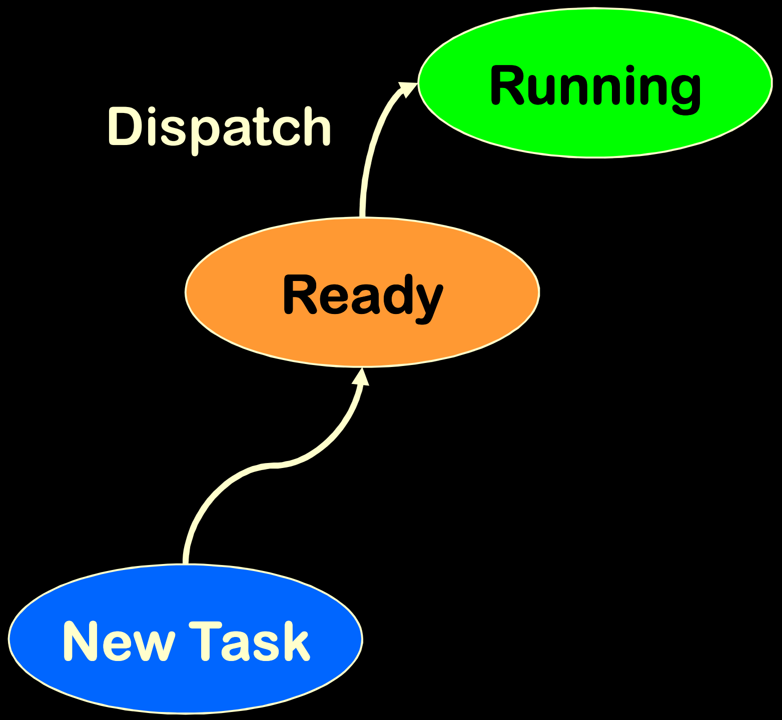
Process Management

New Task

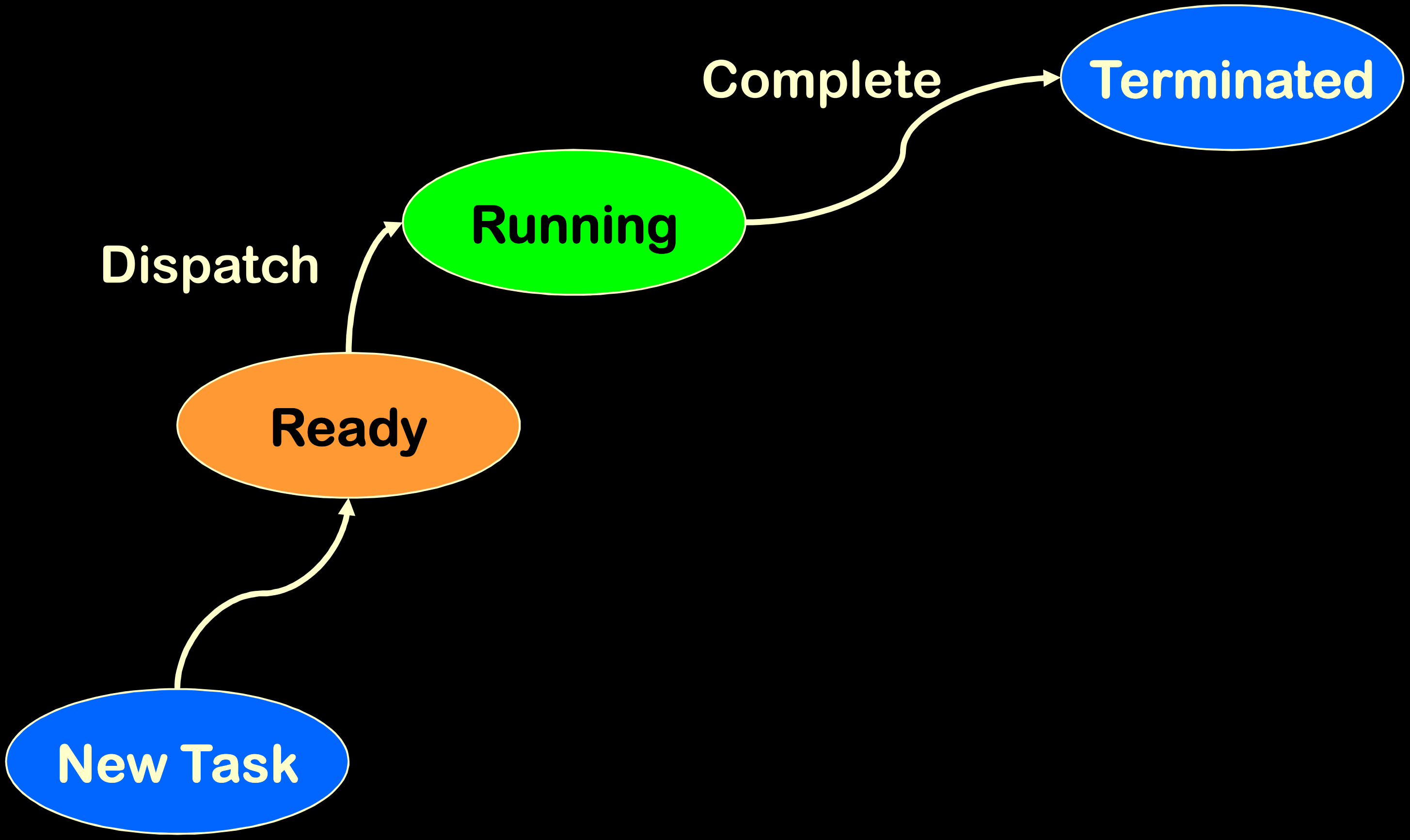
Process Management



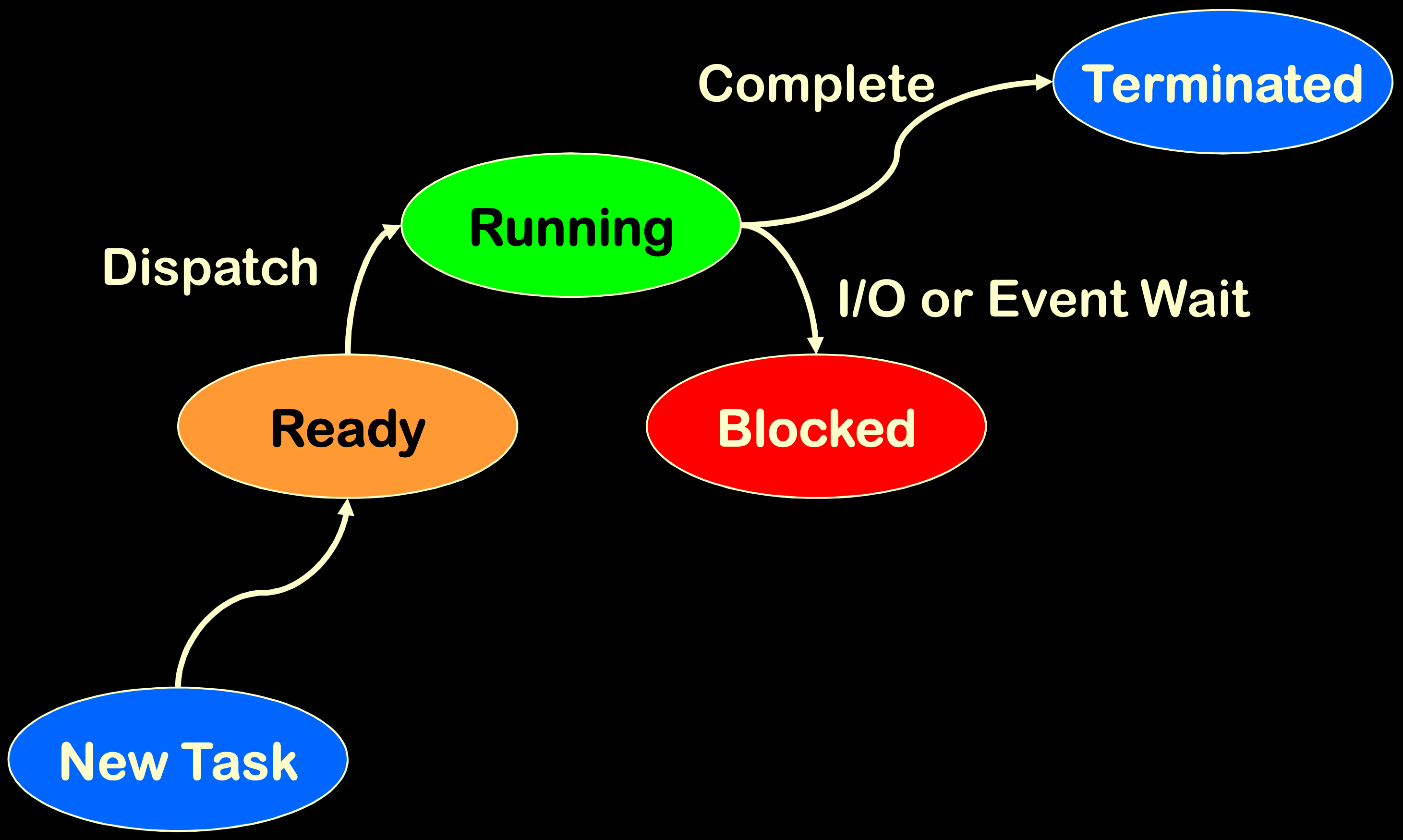
Process Management



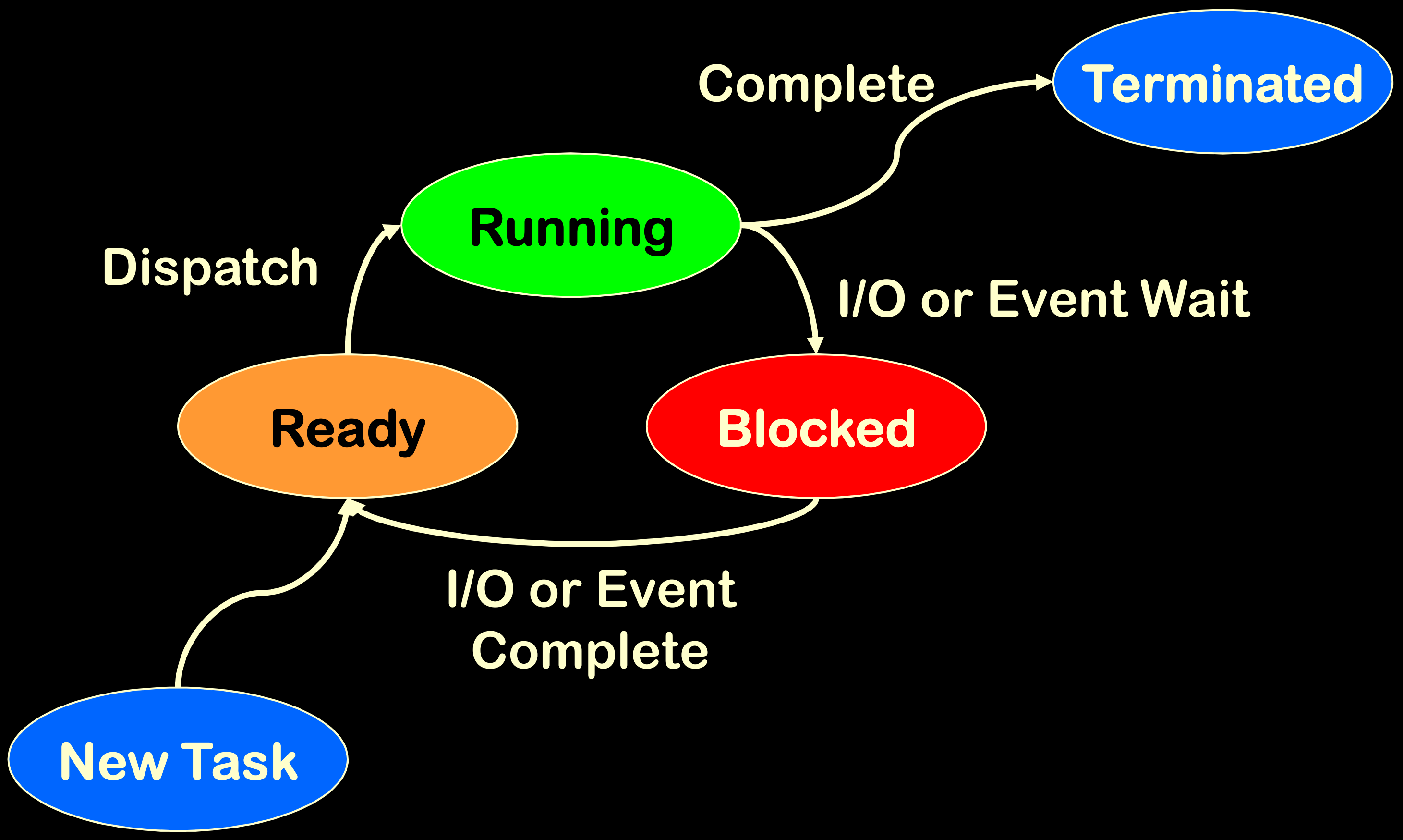
Process Management



Process Management



Process Management



Context Switch

```
       br   main
x:     .word 0
main:  lda  x,d
       deci x,d
       breq done
       adda x,d
       sta  x,d
       br   main
       ...
```

NEW TASK

```
       br   main
x:     .word 0
main:  lda  x,d
       deci x,d
       breq done
       adda x,d
       sta  x,d
       br   main
       ...
```

NEW TASK

Context Switch

```
      br   main
x:    .word 0
main: lda  x,d
      deci x,d
      breq done
      adda x,d
      sta  x,d
      br   main
      ...
```

RUNNING

```
      br   main
x:    .word 0
main: lda  x,d
      deci x,d
      breq done
      adda x,d
      sta  x,d
      br   main
      ...
```

NEW TASK

Context Switch

```
      br   main
x:    .word 0
main: lda  x,d
      deci x,d
      breq done
      adda x,d
      sta  x,d
      br   main
      ...
```

BLOCKED

```
      br   main
x:    .word 0
main: lda  x,d
      deci x,d
      breq done
      adda x,d
      sta  x,d
      br   main
      ...
```

NEW TASK

1. Save whatever is in Accumulator
2. Save whatever is in Program Counter

Context Switch

```
      br   main
x:    .word 0
main: lda  x,d
      deci x,d
      breq done
      adda x,d
      sta  x,d
      br   main
      ...
```

BLOCKED

```
      br   main
x:    .word 0
main: lda  x,d
      deci x,d
      breq done
      adda x,d
      sta  x,d
      br   main
      ...
```

NEW TASK

1. Load Base register for P2
2. Clear Accumulator
3. Set Program Counter to 0
4. Go!

Context Switch

```
      br   main
x:    .word 0
main: lda  x,d
      deci x,d
      breq done
      adda x,d
      sta  x,d
      br   main
      ...
```

BLOCKED

```
      br   main
x:    .word 0
main: lda  x,d
      deci x,d
      breq done
      adda x,d
      sta  x,d
      br   main
      ...
```

RUNNING

Context Switch

```
      br   main
x:    .word 0
main: lda  x,d
      deci x,d
      breq done
      adda x,d
      sta  x,d
      br   main
      ...
```

BLOCKED

```
      br   main
x:    .word 0
main: lda  x,d
      deci x,d
      breq done
      adda x,d
      sta  x,d
      br   main
      ...
```

BLOCKED

1. Save whatever is in Accumulator
2. Save whatever is in Program Counter

Context Switch

```
       br   main
x:     .word 0
main:  lda  x,d
       deci x,d
       breq done
       adda x,d
       sta  x,d
       br   main
       ...
```

READY

```
       br   main
x:     .word 0
main:  lda  x,d
       deci x,d
       breq done
       adda x,d
       sta  x,d
       br   main
       ...
```

BLOCKED

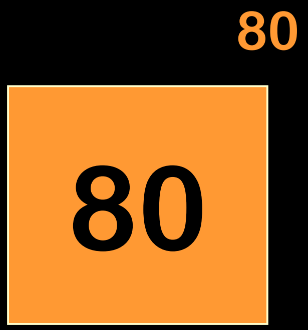
1. Load Base register for P1
2. Restore P1 Accumulator
3. Restore P1 Program Counter
4. Go!

If more than one process is READY, who goes next?

First Come, First Served

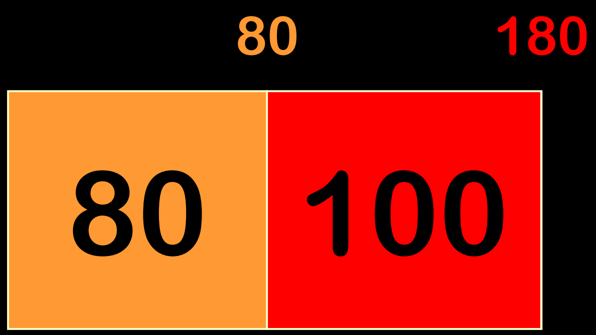
Job	Run time
1	80
2	100
3	150
4	50

First Come, First Served



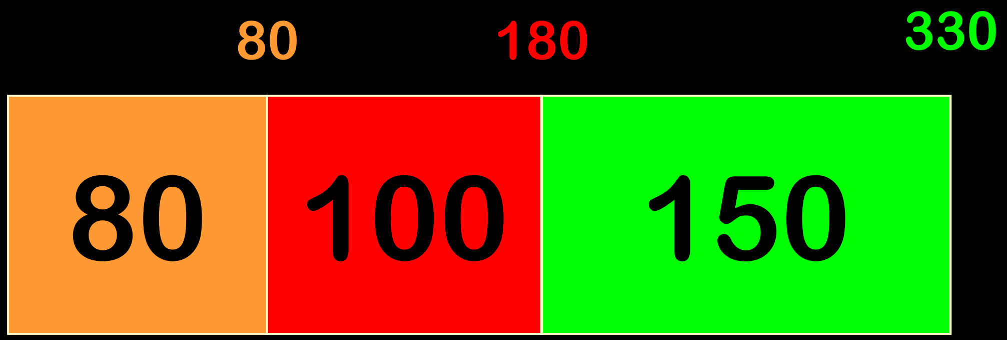
Job	Run time	Delay	Turnaround
1	80	0	80
2	100		
3	150		
4	50		

First Come, First Served



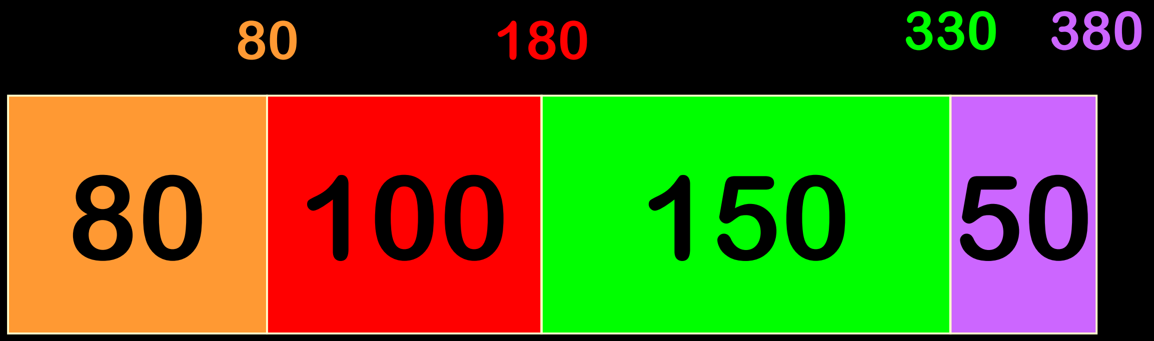
Job	Run time	Delay	Turnaround
1	80	0	80
2	100	80	180
3	150		
4	50		

First Come, First Served



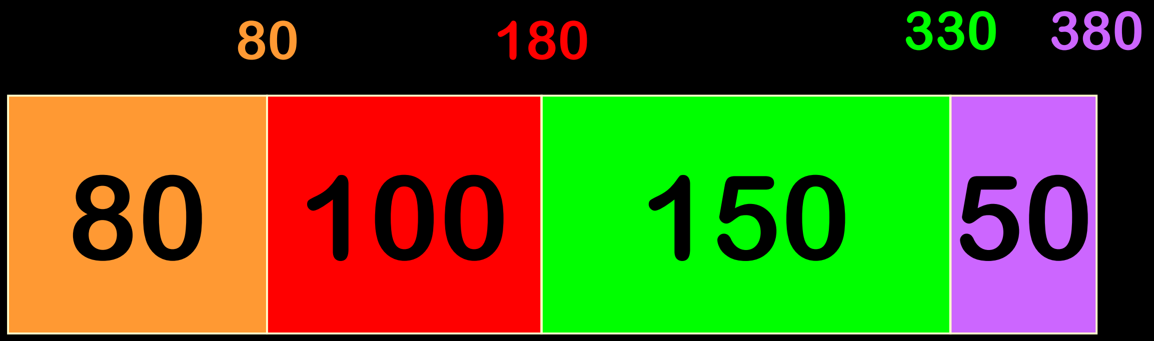
Job	Run time	Delay	Turnaround
1	80	0	80
2	100	80	180
3	150	180	330
4	50		

First Come, First Served



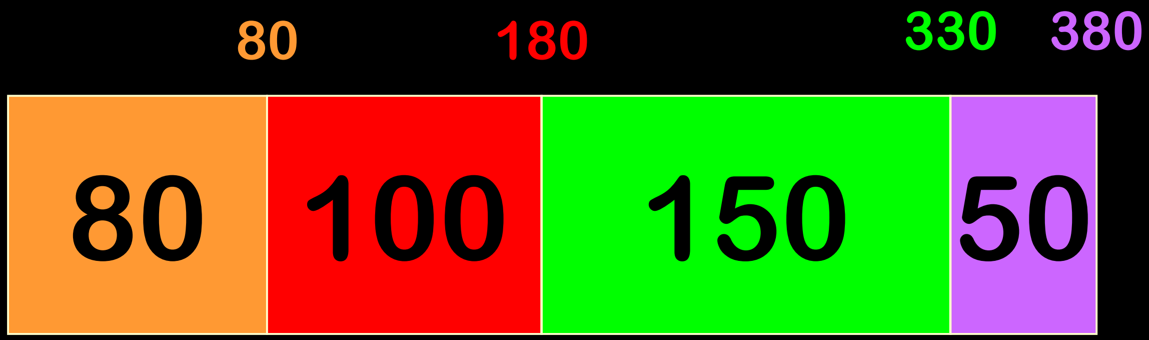
Job	Run time	Delay	Turnaround
1	80	0	80
2	100	80	180
3	150	180	330
4	50	330	380

First Come, First Served



Job	Run time	Delay	Turnaround
1	80	0	80
2	100	80	180
3	150	180	330
4	50	330	380
Avg	95	148	243

First Come, First Served

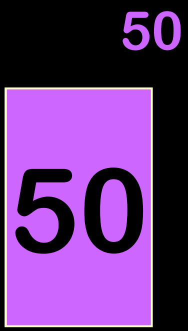


Job	Run time	Delay	Turnaround	Delay %
1	80	0	80	0%
2	100	80	180	80%
3	150	180	330	120%
4	50	330	380	660%
Avg	95	148	243	215%

Shortest Job First

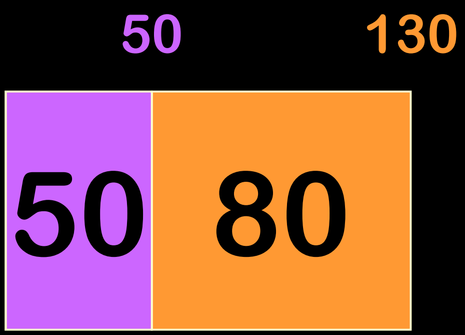
Job	Run time	Delay	Turnaround	Delay %
1	80			
2	100			
3	150			
4	50			
Avg	95			

Shortest Job First



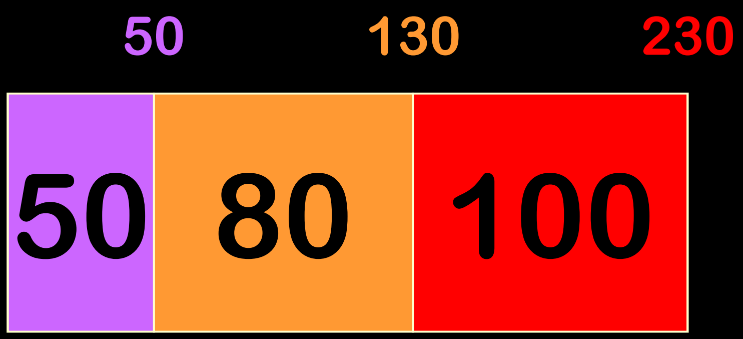
Job	Run time	Delay	Turnaround	Delay %
1	80			
2	100			
3	150			
4	50	0	50	0%
Avg	95			

Shortest Job First



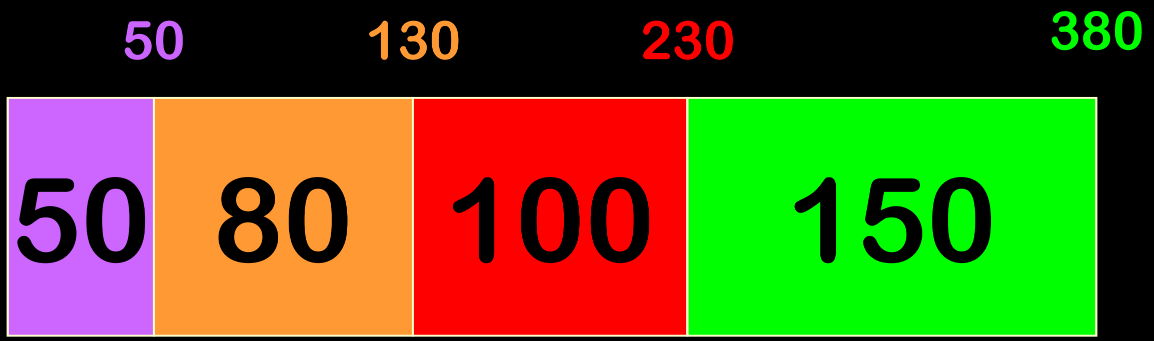
Job	Run time	Delay	Turnaround	Delay %
1	80	50	130	63%
2	100			
3	150			
4	50	0	50	0%
Avg	**95**			

Shortest Job First



Job	Run time	Delay	Turnaround	Delay %
1	80	50	130	63%
2	100	130	230	130%
3	150			
4	50	0	50	0%
Avg	95			

Shortest Job First



Job	Run time	Delay	Turnaround	Delay %
1	80	50	130	63%
2	100	130	230	130%
3	150	230	380	153%
4	50	0	50	0%
Avg	**95**	**103**	**198**	**87%**

Pre-emptive Round Robin

Job	Run time	Delay	Turnaround	Delay %
1	80			
2	100			
3	150			
4	50			
Avg	95			

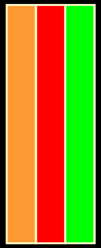
Pre-emptive Round Robin

Job	Run time	Delay	Turnaround	Delay %
1	80	0		0
2	100			
3	150			
4	50			
Avg	95			

Pre-emptive Round Robin

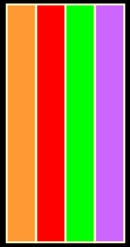
Job	Run time	Delay	Turnaround	Delay %
1	80	0		0
2	100	10		10%
3	150			
4	50			
Avg	95			

Pre-emptive Round Robin



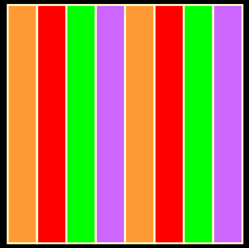
Job	Run time	Delay	Turnaround	Delay %
1	80	0		0
2	100	10		10%
3	150	20		13%
4	50			
Avg	95			

Pre-emptive Round Robin



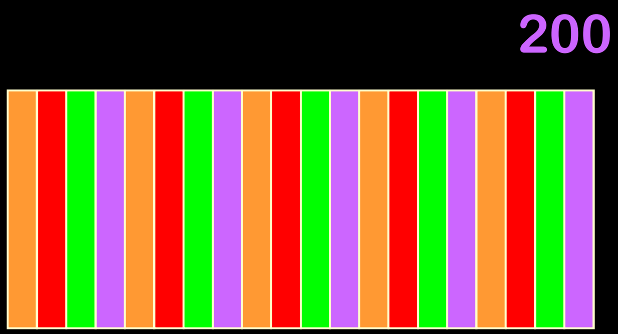
Job	Run time	Delay	Turnaround	Delay %
1	80	0		0
2	100	10		10%
3	150	20		13%
4	50	30		60%
Avg	95	15		21%

Pre-emptive Round Robin



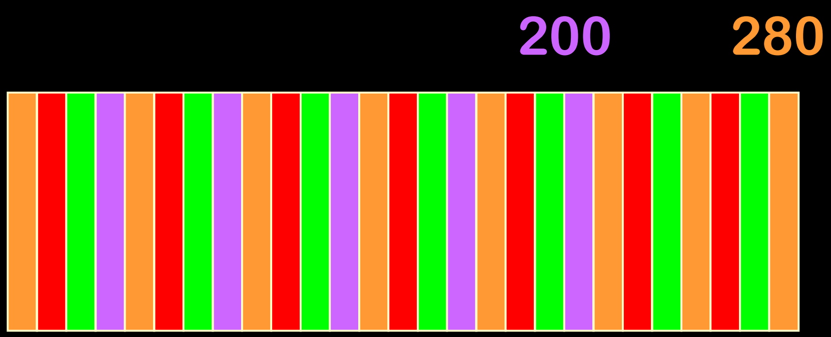
Job	Run time	Delay	Turnaround	Delay %
1	80	0		0
2	100	10		10%
3	150	20		13%
4	50	30		60%
Avg	95	15		21%

Pre-emptive Round Robin



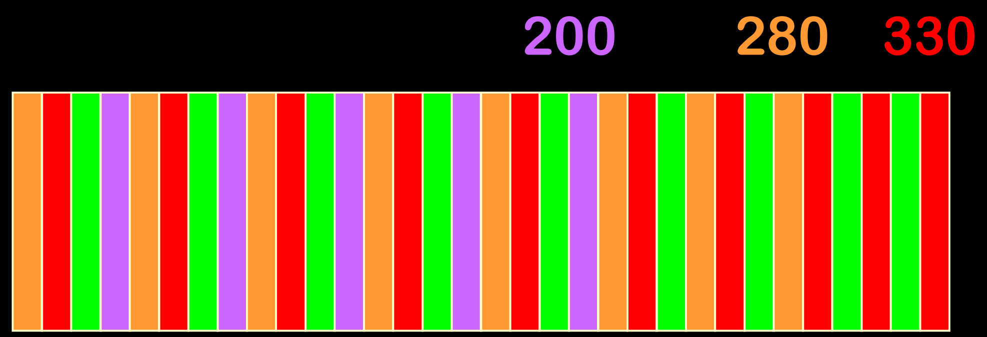
Job	Run time	Delay	Turnaround	Delay %
1	80	0		0
2	100	10		10%
3	150	20		13%
4	50	30	200	60%
Avg	95	15		21%

Pre-emptive Round Robin



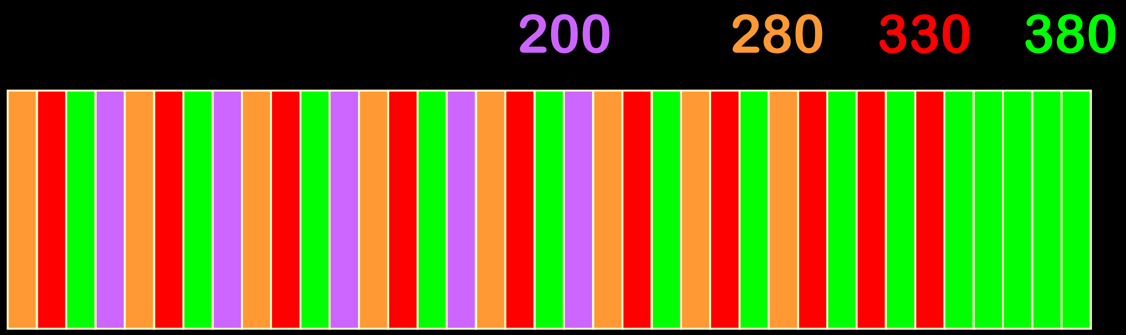
Job	Run time	Delay	Turnaround	Delay %
1	80	0	280	0
2	100	10		10%
3	150	20		13%
4	50	30	200	60%
Avg	95	15		21%

Pre-emptive Round Robin



Job	Run time	Delay	Turnaround	Delay %
1	80	0	280	0
2	100	10	330	10%
3	150	20		13%
4	50	30	200	60%
Avg	95	15		21%

Pre-emptive Round Robin



Job	Run time	Delay	Turnaround	Delay %
1	80	0	280	0
2	100	10	330	10%
3	150	20	380	13%
4	50	30	200	60%
Avg	95	15	298	21%

Average	Delay	Turnaround	Delay %
First come, first served	148	243	215%
Shortest time first	103	198	87%
Pre-emptive round robin	15	298	21%

CPU Scheduling

What are we optimizing? What's "fair"?

- First come, first served
  - Sounds fair
  - Easy to implement!

CPU Scheduling

What are we optimizing? What's "fair"?

- First come, first served
- Shortest job first
  - Sounds good, unless you're a long job
  - How do you know how long it will take?

CPU Scheduling

What are we optimizing? What's "fair"?

- First come, first served
- Shortest job first
- Round robin
  - Pre-emptive (harsh)
  - Complicated, expensive
  - Everyone makes some progress quickly

It's better than that....
But more complicated:

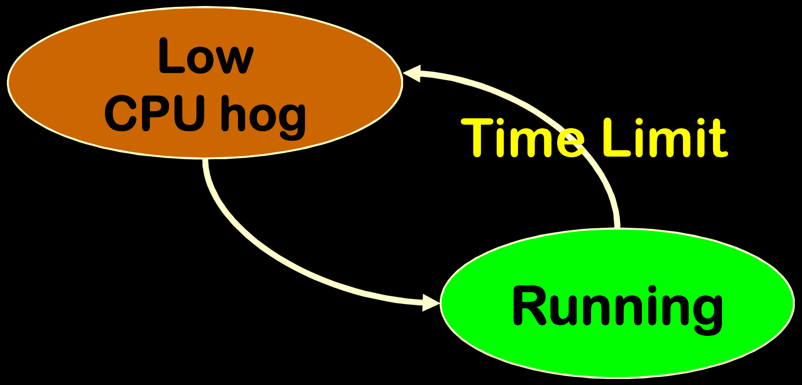
- Processes block frequently
  - Waiting for input (keyboard, disk, ...)
  - Waiting for output to complete
  - Waiting for page swap
  - Waiting for some other resource

It's better than that…. But more complicated:

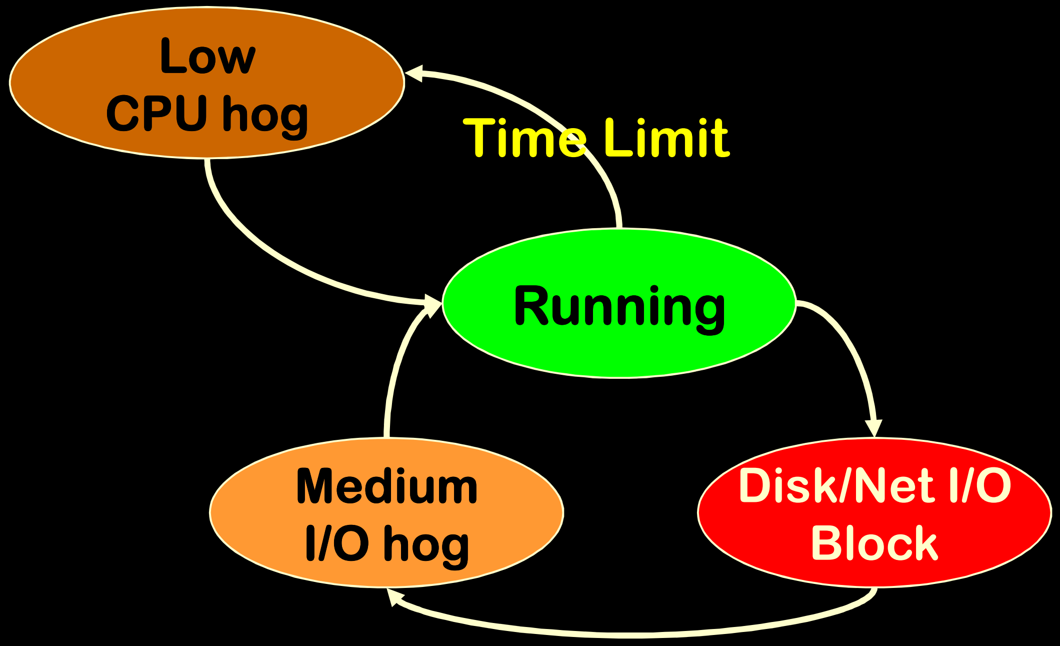
- Processes block frequently
  - Waiting for input (keyboard, disk, …)
  - Waiting for output to complete
  - Waiting for page swap
  - Waiting for some other resource
- Interaction with memory management strategy

Process States – with Priority

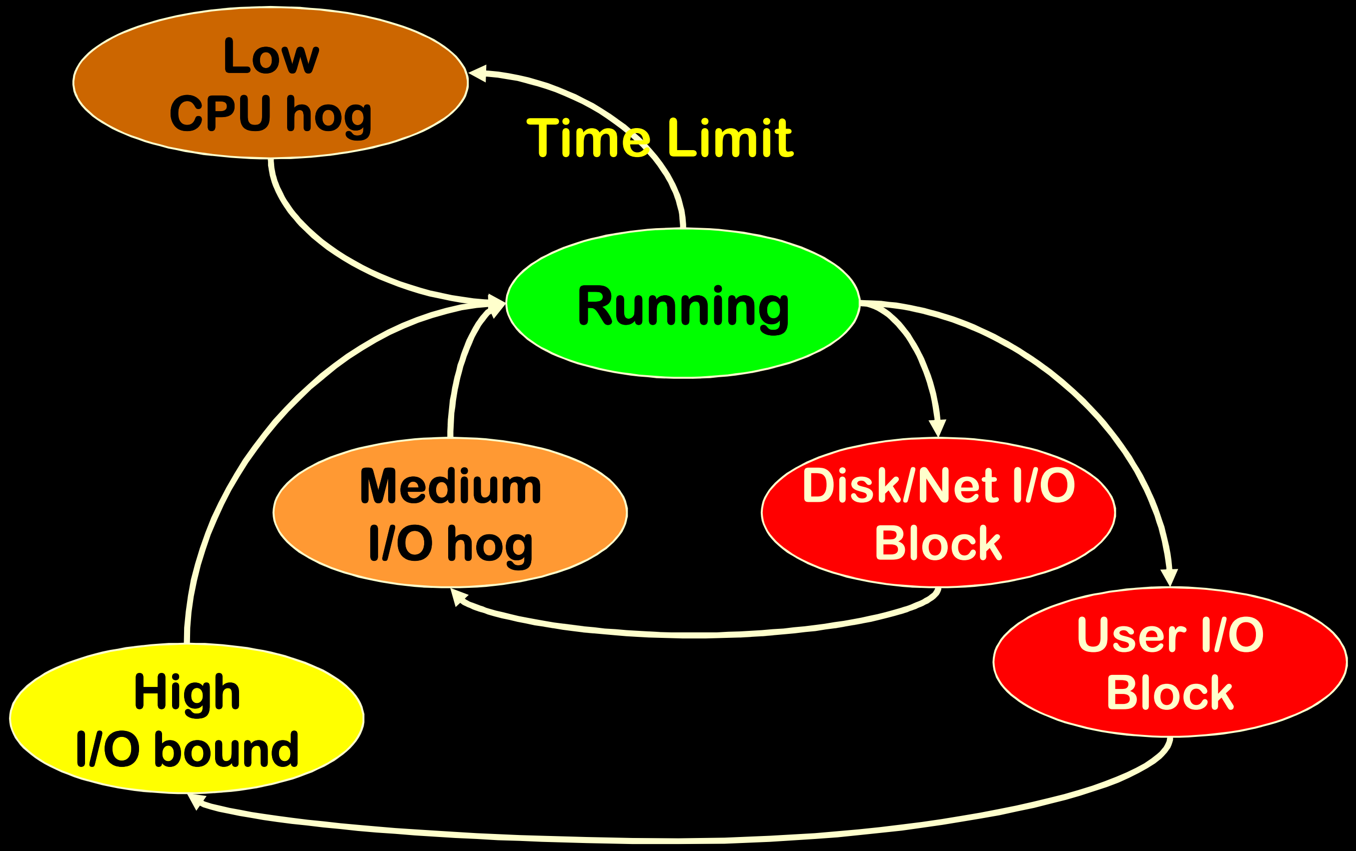
Process States – with Priority



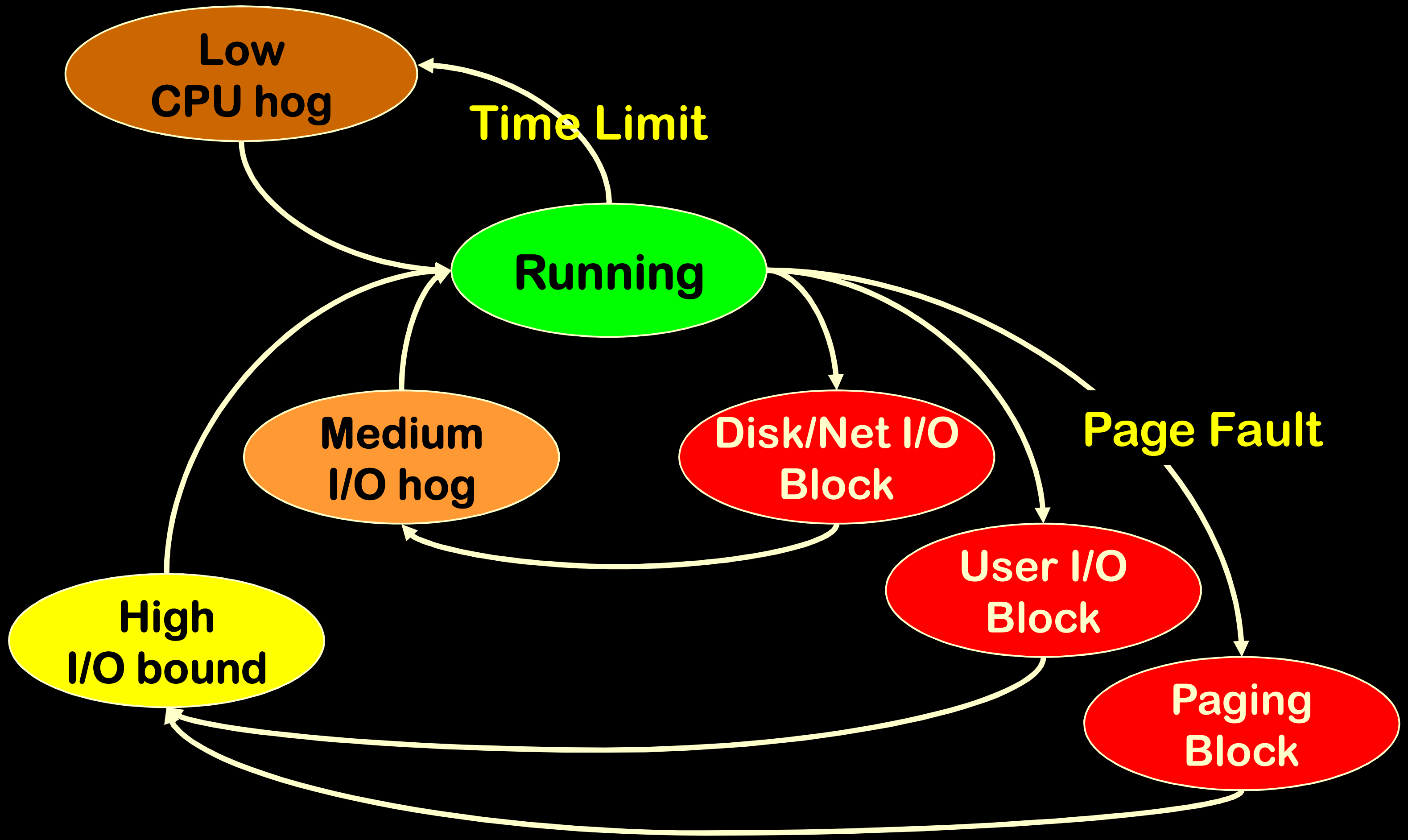
Process States – with Priority



Process States – with Priority



Process States – with Priority



Process Control Block

- Location of each page in memory (page map table)
- Saved accumulator, program counter, base & bound register values, status bits...
- State (running, waiting, ready)
- Priority

13
14
15
16
17
18

about 15 items
express lane
5
about 15 items
express lane
6
BE A SMART MONEY MANAGER