5/3/2/1

1. Match each scheduling strategy to its description:

Shortest Job Next ___

Preemptive Round Robin ___

Priority Scheduling ___

First Come First Serve ___

Multilevel Queue Scheduling ___

(a) Every job runs for a set amount of time before moving on to the next

(b) Job's of the highest priority always runs next.

(c) Jobs are run in the order they are created

(d) Jobs in the same group run with the same priority, but other priorities might run too.

(e) Job that finishes first runs with higher priority

5/3/2/1

2. Which of the above scheduling algorithms is used by Unix-like Operating Systems?

5/3/2/1

3. Ther term "nice" is usd to describe the priority of a process in Unix systems. A processs with a higher priority has a nice value that is low or high? Explain?

5/3/2/1

4. Why is it the case that an unprivlidged process can only decrease its priority but cannot increase its priority?

5/3/2/1

5. Run the **ps** command on any machine in the 302 lab. There exists a process running with level 16 and level 15. What is the name of that command:

Machine: mich302csd [   ] u.academy.usna.edu

Nice 15 Command: [   ] pid: [   ]

Nice 16 Command: [   ] pid: [   ]

___/25

5/3/2/1 6. What are three main process states? Explain each.

```
┌─────────────────────────────────────────────┐
│                                             │
│                                             │
│                                             │
│                                             │
└─────────────────────────────────────────────┘
```

5/3/2/1 7. Provide two reasons why a process may be waiting but not running?

```
┌─────────────────────────────────────────────┐
│                                             │
│                                             │
│                                             │
│                                             │
└─────────────────────────────────────────────┘
```

8. For the small progream snippet, describe the likely state that the program *could* be in at the marks? Explain.

```c
            int d,i,j,sum;
            for(i=0;i<100;i++){
                    printf("Give me a number:\n");
(a) ───────►        scanf("%d",&d);

                    sum = 0;
                    for(j=0;j<d;j++){
                            sum += j;       ◄───────  (b)
                    }

                    printf("Sum is: %d\n", sum);
            }
```

2/1   2/1

(a)
```
┌──────────────────────────┐
│                          │
│                          │
│                          │
│                          │
└──────────────────────────┘
```

(b)
```
┌──────────────────────────┐
│                          │
│                          │
│                          │
│                          │
└──────────────────────────┘
```

5/3/2/1 9.What is a zombie and how are they created? (process zombies, not human zombies)

```
┌─────────────────────────────────────────────┐
│                                             │
│                                             │
│                                             │
└─────────────────────────────────────────────┘
```

5/3/2/1 10. Why are zombies a bad thing? (process zombies not human zombies)

```
┌─────────────────────────────────────────────┐
│                                             │
│                                             │
│                                             │
└─────────────────────────────────────────────┘
```
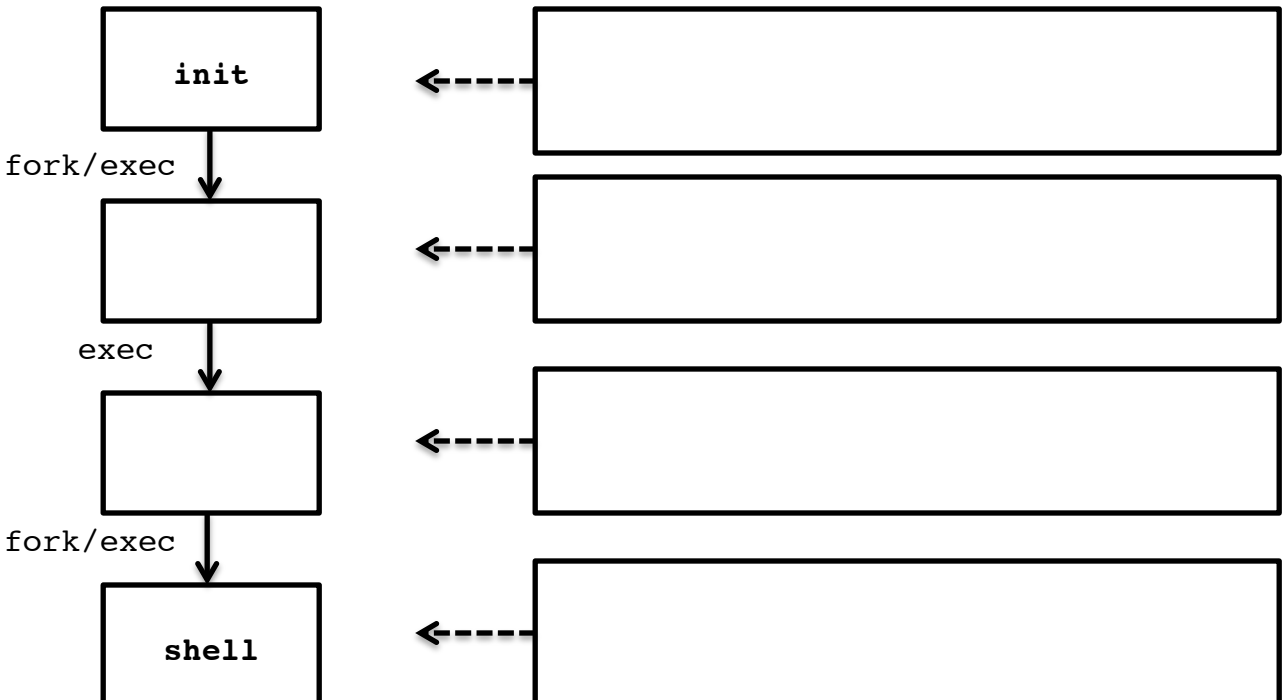
____/24

4/2/1   11. What is an orphan process? What process "adopts" all
orphans?

3/2/1   12. What is a tty? The modern and the anachronistic tty?

3/2/1   13. How many core tty's are launched at boot? One of the tty's
is reserved for what purpose?

4/2/1   14. The terminal device driver controls and manages access to
what service?

5/3/2/1   15. From **init** complete the chain exec's and fork's all the way
down the shell. Describe the task of each.

```
┌──────────┐                ┌────────────────────────────┐
│   init   │ ◄------------  │                            │
└──────────┘                └────────────────────────────┘
  fork/exec
      │
      ▼
┌──────────┐                ┌────────────────────────────┐
│          │ ◄------------  │                            │
└──────────┘                └────────────────────────────┘
     exec
      │
      ▼
┌──────────┐                ┌────────────────────────────┐
│          │ ◄------------  │                            │
└──────────┘                └────────────────────────────┘
  fork/exec
      │
      ▼
┌──────────┐                ┌────────────────────────────┐
│  shell   │ ◄------------  │                            │
└──────────┘                └────────────────────────────┘
```

___/19

;/3/2/1 16. The terminal device driver handles a number of signals to jobs.
Which keys do you press to indicate to the terminal device driver to
terminate a job and which do you press to have the driver stop a
job?

5/3/2/1 17. What is the difference between a **job** and a **process** with
resepect to tasks executing from the shell?

18. For each of the sequence of commands, what would be the output
of **jobs**, generally. Be sure to indicate which jobs are running and
which are stopped.

4/2/1   a) #> cat &
            #> vi &
            #> jobs

4/2/1   b) #> sleep 100 &
            #> sleep 200 &
            #> jobs

4/2/1   c) #> cat
              ^Z
            #> bg
            #> jobs

5/3/2/1 19. What happens when a background process tries to read from **stdin**?

5/3/2/1 20. What happens when a background process tries to write to
stderr or **stdout**? Why does this differ from **stdin**?

___/32