IC221 System Programming
Spring 2015 NAME:_____
HW10

COLLABORATOR(S):_____

1. What is a process group and how does it relate to a job in the shell?

5/3/1/0

2. How long will the following shell command run for? And why?

sleep 10 | sleep 20 | sleep 100 | sleep 30 | sleep 1

5/3/1/0

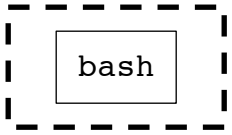
3. Explain the difference between sequential and parallel execution of a command line?

5/3/1/0

4. For the following set of shell commands draw the process groupings at the last command execution.

```
#> cat | cat | cat > output &
#> sleep 20 | sleep 30 &
#> ps
```

10/8/5/3/0



__/25

NAME: _____

5. For each of the system calls associated with process groupings, match them to their description. 5/3/1/0

setpgrp() _____	(a)	Returns the process group id of the calling process
setpgid() _____	(b)	Sets the process group id of the calling process to its pid
getpgrp() _____	(c)	Returns the process group of a process identified by a pid
getpgid() _____	(d)	Sets the process group of the process identified by pid to a specified pgid

6. For each system call, briefly describe the resulting action: 5/3/1/0

getpgid(0)	
setpgid(0,0)	
setgpid(0,pgid)	
setpgid(pid, 0)	

7. Consider the following code snippet, what is the output and why? 10/8/5/1/0

```
int main(){
    pid_t cpid;
    cpid = fork();
    setpgrp();
    if(cpid == 0){
        if( getpid() == getpgid()){
            printf("C: SAME PGID\n");
        }
        _exit(0);
    }else if(cpid > 0){
        if(getpgid(cpid) == cpid){
            printf("P: SAME PGID\n");
        }
        wait();
        _exit(0);
    }
    _exit(1); //fork failed_
}
```

__/20

NAME: _____

0/8/5/3/0 8. Consider the following code snippet. If we were to run this program in a terminal, will it be properly terminated by Ctrl-c? If so, why? If not, why not?

```
int main(){
    pid_t cpid;
    cpid = fork();
    if( cpid == 0 ){
        setpgrp();
        while(1);
    }else if( cpid > 0 ){
        wait();
        _exit(0);
    }
    _exit(1); //fork failed
}
```

/8/5/3/0 9. Consider the following code snippet with the open file **fight.txt** containing the text **_Go_Navy!_Beat_Army!** where **_** indicates a space. What is the output of this program, and why?

```
int main(){
    pid_t cpid;
    int fd = open( /* fight.txt */);
    char buf[1024];

    cpid = fork();
    if( cpid == 0 ){
        read(fd, buf, 10);
        _exit(0);

    }else if( cpid > 0 ){
        wait(); /* wait for child*/

        read(fd,buf, 10);
        write(1, buf, 10);
        _exit(0);
    }
    _exit(1); //fork failed
}
```

5/3/1/0 10. The **pipe()** system call sets the value of two file descriptors in an array: what is index 0 used for and what is index 1 used for?

5/3/1/0 11. What does it mean to "widow" a pipe? Why must the write end typically be widowed?

__/30

NAME: _____

12. What is the default action when a process writes to a pipe more data than the kernel buffer can hold? Can this default action be changed?

5/3/1/0

13. If the open file **fight.txt** containing the text **_Go_Navy!_Beat_Army!** where **_** indicates a space. What is the output to **stdout** and what is the output to **output.txt**, and why?

10/8/5/3/0

```
int main(){
    int fd_in = open( /* fight.txt */);
    int fd_out = open(/* output.txt */)
    char buf[1024]

    close(0);
    dup2(fd_in,0);

    close(1);
    dup2(fd_out,1);

    while(scanf("%s",buf) != EOF){
        printf("%s\n",buf);
    }
    return 0;
}
```

14. Add the necessary code using **dup2()** and **close()** such that the child's write to **stdout** will be read by the parent through **stdin**.

10/8/5/3/1/0

```
int main(){
    pid_t cpid;
      int pfd[2], n
      char gonavy[] = "Go Navy!";
    char buffer[1024];

    pipe(pfd);

    cpid = fork();
    if( cpid == 0 ){
```

```
      write(1, gonavy,strlen(gonavy));
    }else if( cpid > 0 ){
```

```
      n = read(0, buffer, 1024);
      write(1,buffer,n);
    }
    _exit(1); //fork failed
}
```

__/25