CECS 526 Assignment 4 (2 point)
Due : 2/25/2020 by class time, on BeachBoard

Given the following code of a monitor that is designed to solve the readers/writers problem in FCFS order while maintaining exclusive write and concurrent read:

```
RW : monitor;
begin
    readercount : integer;
    busy : boolean;
    FCFS_Q, write_Q : condition;

    procedure startread;
    begin
        if busy OR FCFS_Q.queue OR write_Q.queue then FCFS_Q.wait;
        readercount := readercount + 1;
        FCFS_Q.signal;
    end startread;

    procedure endread;
    begin
        readercount := readercount -1;
        if readercount = 0 then
            if write_Q.queue then write_Q.signal
            else FCFS_Q.signal;
    end endread;

    procedure startwrite;
    begin
        if busy OR readercount ≠ 0 OR FCFS_Q.queue then FCFS_Q.wait;
        if readercount ≠ 0 then write_Q.wait;
        busy := true;
    end startwrite;

    procedure endwrite;
    begin
        busy := false;
        if FCFS_Q.queue then FCFS_Q.signal;
    end endwrite;

    begin /* Main program of the monitor, used for initialization */
        readercount := 0;
        busy := false;
    end;
end RW;
```

Suppose the above monitor is used to control the access to a shared file among a number of readers and writers. Assume that the sequence of arrivals of processes that require access to the shared resource be: R1, W1, R2, R3, W2. Here R1, R2, R3, W1, and W2 are processes that are in existence concurrently. For simplicity, assume further the following:

i) These arrivals are 1 second apart.

ii) Each of the monitor procedure takes 1 second to execute (not counting the time in queue). For simplicity, you may assume 1 second even when only part of the procedure gets executed, and when the procedure resumes execution another second will be used.

iii) Each operation, read or write, takes 5 seconds.

iv) All processes that require access to the shared file will follow the 3-step procedure:

- Reader: call RW.startread; read; call RW.endread.
- Write: call RW.startwrite; write; call RW.endwrite.

Note: The time durations provided above are to enable you to determine the approximate timing of events. Leave an entry in the table below blank when there is no change.

Name:Varun Lingabathini_____

Time in Second	Process	Procedure executed	Variables		Condition variable queues		Process removed from any queue
			readercount	busy	FCFS_Q	write_Q	
Initially	None	None	0	false	empty	empty	None
0	R1	startread	1				
[1-5]	R1	read	1				
1	W1	startwrite	1		W1		
2	R2	startread	1		W1,R2		
3	R3	startread	1		W1,R2,R3		
4	W2	startwrite	1		W1,R2,R3,W2		
6	R1	endread	0		R2,R3,W2		W1
7	W1	Startwrite(contd.)	0	true	R2,R3,W2		none
[8-12]	W1	write	0	true	R2,R3,W2		none
13	W1	endwrite	0	false	R3,W2		R2
14	R2	Startread(contd.)	1		W2		R3
[15-19]	R2	read	1		W2		none
15	R3	Startread(contd.)	2		empty		W2
[16-20]	R3	read	2				None
16	W2	Startwrite(contd.)	2			W2	none
20	R2	endread	1			W2	none
21	R3	endread	0			empty	W2
22	W2	Startwrite(contd.)	0	true			none
[23-27]	W2	write	0	true			
28	W2	endwrite	0	false			
finally	none	none	0	false	empty	empty	none