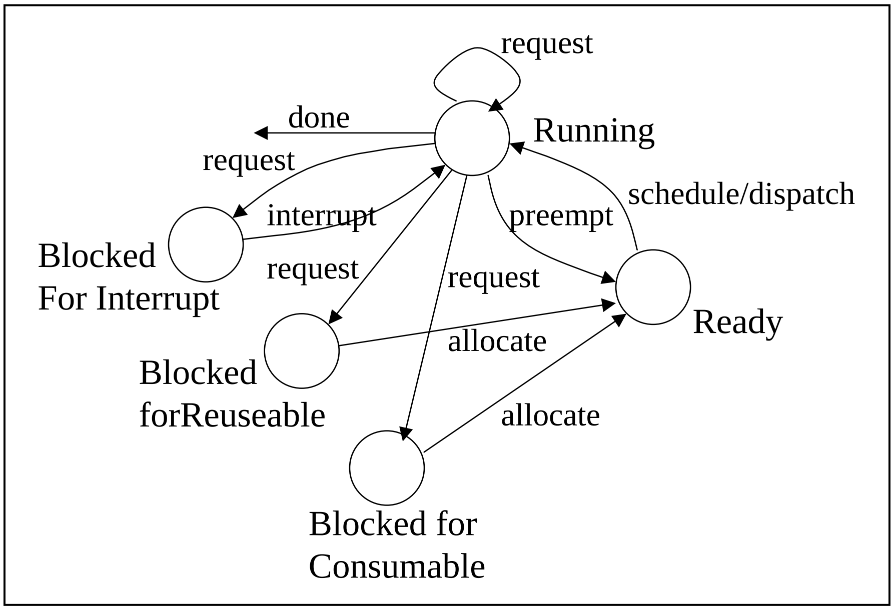
Page 230: problem 7

Here is a solution. You might want the return from interrupt to make this process be Ready instead of Running.



Page 281, Problem 6

RR scheduling with a time quantum of 15

a. Gantt chart

0 15 30 45 60 75 85 100 115 130 140 145 160 175 190 205

p_0	p_1	p_2	p_0	p_1	p_2	p_3	p_4	p_0	p_1	p_3	p_4	p_0	p_4	p_0

b. Turnaround time for p_3 = 65

c. Average wait time

W(p_0) = 0
W(p_1) = 5
W(p_2) = 20
W(p_3) = 5
W(p_4) = 15

Average = (0+5+20+5+15)/5 = 9

Page 322: problem 4

a. This solution forces the two cooperating processes to alternate visits to the critical section. Thus, there is an implicit timing dependency between the two processes, e.g., if one is much slower than the other, then the fast process will have to synchronize with the slow process on each cycle through the loop. Also, if either stops then the other can also only run through the critical section at most one more time
b. This algorithm does not solve the problem since it might allow both processes to be in the critical section at the same time. Suppose that the two processes "simultaneously" execute the while-test; they will both pass the test, set their respective flag and enter the critical section. The algorithm is not safe.
c. This algorithm attempts to resolve the problem described in part (b) by setting the flag before attempting to test the opposite flag. Now, if both processes set their respective flag variables "simultaneously," then the two processes are deadlocked.