CSE 450 Operating Systems
Homework 4

Name: __partial solutions______

Question 2. Consider the following snapshot of a system (P=Process, R=Resource) :

Available			
R_a	R_b	R_c	R_d
1	5	2	0

Maximum Demand				
	R_a	R_b	R_c	R_d
P_0	0	3	1	2
P_1	1	7	5	0
P_2	2	3	5	6
P_3	0	6	5	2
P_4	0	6	5	6

Current Allocation				
	R_a	R_b	R_c	R_d
P_0	0	0	1	2
P_1	1	0	0	0
P_2	1	3	5	4
P_3	0	6	3	2
P_4	0	0	1	4

Answer the following questions using banker's algorithm:

a) [5 points] Calculate the *Needs* matrix: (= maximum demand – current allocation)

Needs				
	R_a	R_b	R_c	R_d
P_0	**0**	**3**	**0**	**0**
P_1	**0**	**7**	**5**	**0**
P_2	**1**	**0**	**0**	**2**
P_3	**0**	**0**	**2**	**0**
P_4	**0**	**6**	**4**	**2**

b) [10 points] Is the system in a safe state? If so, show how you derive a safe order with Safety Algorithm in which the processes can run. Show the different values of the *work* vector after each iteration. What is the sequence of processes that the algorithm implicitly created?

Work = available = [1, 5, 2, 0]
Flag = [0, 0, 0, 0, 0, 0]
For i =0; i<5; i++
Find i that satisfies:
1) Need[i] <= work
2) Flag[i] == 0

Find i = 3, that [0, 0, 2, 0] <= work [1,5,2,0], so work = work + allocation[3] = [1, 11,3,2]

Next, find i = 4 satisfies the conditions, so work = work + allocation[4] = [1, 11, 4, 6],
Next, i can go with either 0 or 2, assume we go with i=0, then work = work + allocation[0] = [1,11,5,8];
Next, i = 1, work = work + allocation[1] = [2,11,5,8];
Next i =2, work=work + allocation [2] = [3,14,10,12].

So the sequence we found from safety algorithm is P_3, P_4, P_0, P_1, P_2; [note: there could be other sequences available as well].

c) If a request from process P_0 arrives for (0, 3, 0, 0), can the request be granted immediately? Justify your answer, using only the knowledge of the sequence you found at sub-question (b).

~~A request [0,3, 0, 0] from P_0 won't be granted immediately, since [0,3,0,0] !<= available [1,5,2,0]~~

1. Request [0, 3, 0, 0] < Available = [1, 5, 2 ,0]
2. Request [0, 3, 0, 0] <= need[0] = [0, 3, 0, 0]

We assume the request will be granted, then
Work = Available = available – request = [1, 2, 2, 0], and allocation [0] = allocation[0] + request = [0, 3, 1, 2], needs[0] = [0, 0, 0, 0]

Run the safety check as follows:

For i =0; i<5; i++
    Find i that satisfies:
        1) Need[i] <= work
        2) Flag[i] == 0

First, we found i = 3, needs[3] = [0, 0, 2, 0] < work = [1, 2, 2, 0], then work = work + allocation [3] = [1, 8, 5, 2], flag[3]=true;
We next try i=4, needs[4] = [0, 6, 4, 2] < work = [1, 8, 5, 2], so now work = work + allocation[4] = [1, 8, 6, 6], flag[4]=true … similarly, we could find i = 0, 1, 2 satisfying the conditions, so we could find a sequence of processes, P_3, P_4, P_0, P_1, P_2, which supports the immediate allocation of [0, 3, 0, 0] to process P_0.