Department of Mathematical Sciences
Carnegie Mellon University

21-393 Operations Research II
Test 1

Name:_____

Problem	Points	Score
1	35	
2	35	
3	30	
Total	100	

Q1: (35pts)
A system can be in 3 states 1,2,3 and the cost of moving from state i to state j in one period is $c(i,j)$, where the $c(i,j)$ are given in the matrix below. The one period discount factor α is $1/2$.
The aim is to find a policy which simultaneously minimises the discounted cost of operating from any starting state. Start with the policy

$$\pi(1) = 2, \pi(2) = 1, \pi(3) = 2.$$

Evaluate this policy. Is it optimal? If not find an improved policy.
YOU DO NOT NEED TO EVALUATE THIS NEW POLICY OR FIND AN OPTIMAL STRATEGY.
The matrix of costs is

$$\begin{bmatrix} 5 & 10 & 1 \\ 8 & 2 & 2 \\ 1 & 10 & 2 \end{bmatrix}$$

Q2: (35pts)
Let $I = \{a, a+1, \ldots, b\}$ and $J = \{c, c+1, \ldots, d\}$ be two integer intervals. Every pair of sub-intervals $I' \subseteq I, J' \subseteq J$ is given a value $v(I', J') > 0$. Here we are restricting our attention to intervals that have integer endpoints. Give a Dynamic Programming algorithm for partitioning I into consecutive intervals $I_1, I_2, \ldots, I_m$ and J into consecutive intervals $J_1, J_2, \ldots, J_m$ in order to maximise the total value $v(I_1, J_1)+v(I_2, J_2)+\cdots+v(I_m, J_m)$. The intervals chosen must be such that $I_t \cap J_t \neq \emptyset$ for $t = 1, 2, \ldots, m$. Here, as in "breaking up a stick" m is not fixed, but is determined by the algorithm.
(If this is not possible, for example if b, c then the maximum total value will be zero.)

Q3: (30pts)
We are given a digraph $D = ([n], E)$ and edge lengths $\ell(e) \geq 0$ for $e \in E$. Dikkstra's algorithm below will find a shortest path from vertex 1 to every other vertex.

Dijstra's Algorithm:

1. **begin**
2. **for** $i = 2, \ldots, n$, $d(i) \leftarrow \ell(1, i)$, $P_i \leftarrow (1, i)$; $S_1 \leftarrow \{1\}$;
3. **for** $k = 2, \ldots, n$ **do**;
4. **begin**
5. $d(i) = \min\{d(j) :\notin S_k\}$;
6. $S_{k+1} \leftarrow S_k \cup \{i\}$;
7. **for** $j \notin S_{k+1}$ **do**
8. **if** $d(j) > \ell(P_i) + \ell(i, j)$ then $d(j) \leftarrow \ell(P_i) + \ell(i, j)$, $P_j \leftarrow (P_i, j)$;
9. **end**
10. **end**

Suppose now that $E = R \cup B$ and our paths are allowed to contain at most k edges from R. Give the specific changes to the above that allow the algorithm to find shortest paths with the extra restrictions.