

1. The algorithm has the following steps-

(a) Determine the minimum number in A . Designate this as Min . This takes $O(n)$ time.

(b) **for** $i := 1$ **to** n **do**

$$B[i] = A[i] - Min$$

(c) **for** $i := 1$ **to** n **do**

$$C[i] = B[i] * n^5$$

(d) C contains **Integers** in the range $[1 : n^{10}]$ or $[0 : n^{10} - 1]$. These can be sorted in $O(n)$ time using **Radix-Sort**.

Note:- The elements in C can be considered to be **Integers**, after truncating their real parts, preserving their relative orders.

2.

$$\frac{p_6}{w_6} > \frac{p_1}{w_1} > \frac{p_7}{w_7} > \frac{p_2}{w_2} > \frac{p_4}{w_4} > \frac{p_3}{w_3} > \frac{p_5}{w_5} > \frac{p_9}{w_9} > \frac{p_8}{w_8}$$

$$S = (1, 1, 0, 0.8, 0, 1, 1, 0, 0), F^*(I) = 85$$

3. One possible minimum spanning tree has the following edges: $(3, 4), (4, 5), (2, 4), (4, 7), (1, 6)$ and $(1, 3)$. The total weight is 21.

4. **Note:-** Here the weight on every edge **of the graph** is increased. In this case, T would still continue to be a MCST.

Proof :- Consider any optimal algorithm to construct a MCST. Let $(e_1, e_2, \dots, e_{|E|})$ be the order in which the edges are considered for constructing the MCST T . Even after the weight on every edge is increased, this order will remain the same and hence the tree output will be the same.

5. At the beginning of the algorithm $dist[s] = 0; dist[1] = 5; dist[2] = 21; dist[3] = 11; dist[4] = dist[5] = \infty$.

In stage 1, node 1 has the minimum $dist$ value and hence is inserted into the set S . Nodes 2 and 4 are neighbors of 1 and hence we have to check if the $dist$ values of these nodes have to be modified. Since $dist[2] > dist[1] + W(1, 2)$, we change $dist[2]$ to 9. Likewise we set $dist[4] = 13$.

In stage 2, node 2 has the minimum $dist$ value and hence is inserted into S . Nodes 3, 4 and 5 are neighbors of 2. The new $dist$ values of these nodes become: $dist[3] = 11; dist[4] = 12; dist[5] = 21$.

In stage 3, node 3 has the minimum $dist$ value and it becomes a part of S . Node 4 is the only neighbor of 3. The $dist$ value of 4 does not change.

In stage 4, node 4 has the minimum $dist$ value. Node 5 is the only neighbor of 4 and the new $dist$ value of 5 becomes 17.

In stage 5, the node 5 also enters S . Algorithm terminates then.

Thus the shortest paths from s to the nodes 1, 2, 3, 4, and 5 are 5, 9, 11, 12, and 17, respectively.

6. As discussed in class, if $f_i(y)$ is the optimal profit for $\text{KNAP}(1, j, y)$, the recurrence relation for $f_i(y)$ is given by: $f_i(y) = \max\{f_{i-1}(y), f_{i-1}(y - w_i) + p_i\}$. Also, $f_0(y) = 0$ for all non-negative values of y and $f_i(y) = -\infty$ when y is negative. From these relations we compute $f_0(y), f_1(y), f_2(y), f_3(y), f_4(y)$ for all $0 \leq y \leq 5$. These values are shown in the following table.

Function	$y = 0$	$y = 1$	$y = 2$	$y = 3$	$y = 4$	$y = 5$
f_0	0	0	0	0	0	0
f_1	0	20	20	20	20	20
f_2	0	20	20	30	30	30
f_3	0	20	20	35	35	45
f_4	0	20	20	35	45	45

For example, $f_3(5) = \max\{f_2(5), f_2(5 - 2) + 15\} = \max\{30, 30 + 15\} = 45$. Also, $f_4(4) = \max\{f_3(4), f_3(4 - 3) + 25\} = \max\{35, 20 + 25\} = 45$; and so on. Thus the optimal profit is 45.