

1)MCQS-

i) Which algorithm is used in finding all pairs shortest distance?

Answer- a) Dynamic Programming is correct answer.
(Floyd-Warshall algorithm)

ii) 0/1 knapsack is based on _____ method?

Answer- b) Branch & Bound(https://www.youtube.com/watch?v=yV1d-b_NeK8&t=135s)

c)Dynamic Programming (<https://www.youtube.com/watch?v=nLmhmb6NzcM>)

2 answers possible as knapsack 0/1 can be solved by both but write c Dynamic Programming as it is more efficient.

iii)A ____ is a round trip path along n edges of G that visits every vertex once and returns to its starting position.

Answer-

d)Hamiltonian Cycle

(used in Travelling salesman problem)

iv) The upper bound on the time complexity of the nondeterministic sorting algorithm is?

Answer-

$O(n)$

Non deterministic algorithms always reduce time complexity.

(<https://www.youtube.com/watch?v=ZNe1ziMExGg>)

Non-Deterministic Algorithm

Outcome of NDA lgo will be restricted to specific set of possibilities.

To specify such algorithms, we introduce three new functions:

1. $\text{Choice}(S)$: arbitrarily chooses one element from set S
2. $\text{Failure}()$: signals an unsuccessful solution
3. $\text{Success}()$: signals an successful solution

Problem-1: Searching x on $A[1:n]$, $n \geq 1$. On success returns j if $A[j] = x$ or returns 0 otherwise.

1. $j = \text{Choice}(1, n)$;
2. if $(A[j] = x)$ then $\{\text{write}(j); \text{Success}();\}$
3. $\text{write}(0); \text{Failure}();$

Problem-2: Sorting array $A[1:n]$ of positive integers in ascending order.

Algorithm Nsort(A, n) // sort n positive integers.

```
{ for  $i = 1$  to  $n$  do  $B[i] = 0$ ; // Initialize  $B[]$ 
  for  $i = 1$  to  $n$  do
```

```
{
   $j = \text{Choice}(1, n)$ ;
  if  $(B[j] \neq 0)$  then  $\text{Failure}()$ ;
   $B[j] = A[i]$ ;
```

```
}
for  $i = 1$  to  $(n-1)$  do // Verify order
  if  $(B[i] > B[i+1])$  then  $\text{Failure}()$ ;
```

```
write( $B[1:n]$ );
```

```
Success();
}
```



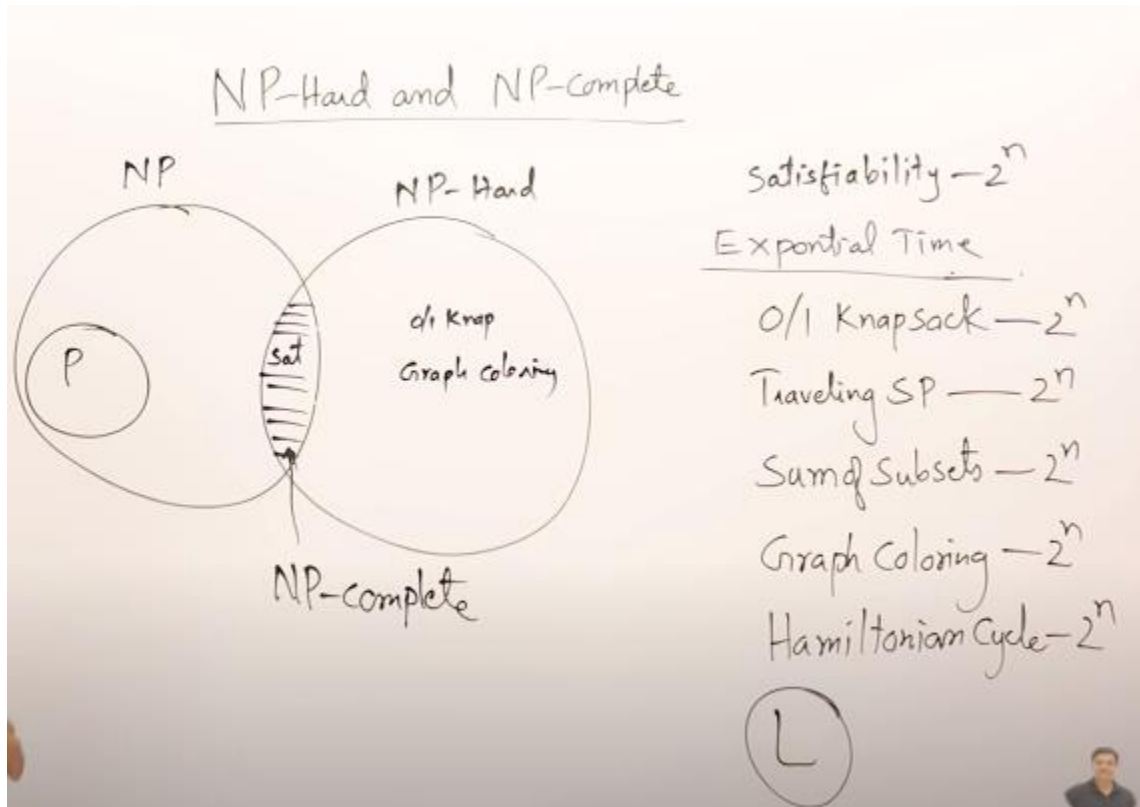
v) Choose the correct answer-

I. Theory of NP-Completeness provides a method for providing polynomial time for NP Problems.

II. All NP Problems are NP-Hard.

Answer- a) I is false and II is true

Explanation- <https://youtu.be/e2cF8a5aAhE>, <https://www.geeksforgeeks.org/np-completeness-set-1/>



Similar mcq(for practise)

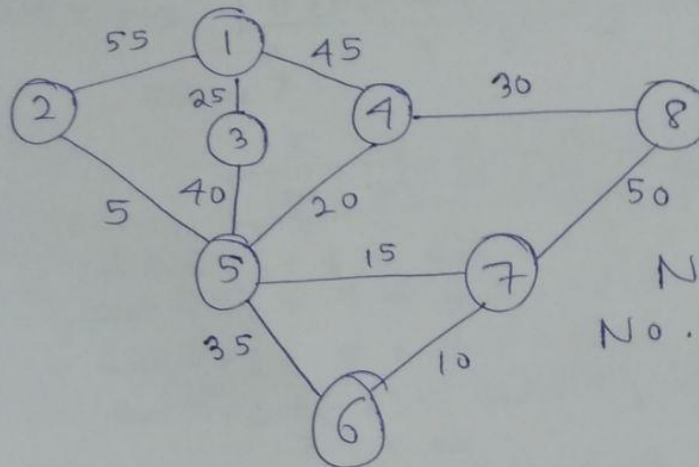
The following are the statements regarding the NP problems. Chose the right option from the following options:

I. All NP-complete problems are not NP-hard.

II. Some NP-hard problems are not known to be NP-complete.

Answer-Only II is true.

Identify the Minimum Spanning Tree
Using Kruskal's Algo



No. of nodes = 8
 No. of edges = 11

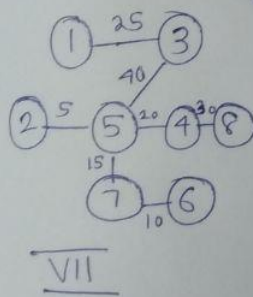
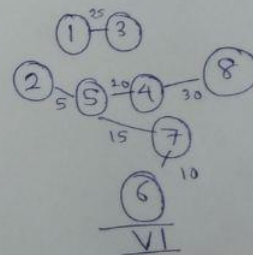
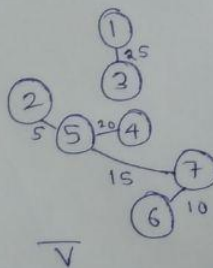
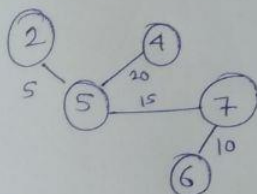
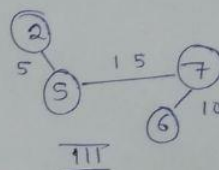
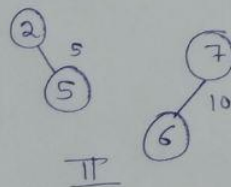
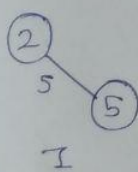
Configuration of edge

wt	u	v
55	1	2
25	1	3
45	1	4
5	2	5
40	3	5
20	4	5
30	4	8
50	8	7
15	7	5
10	7	6
35	5	6

Page 2 of creating MST using Kruskal

Sorting weights in ascending order
(greedy approach)

wt	u	v
5	2	5
10	7	6
15	7	5
20	4	5
25	1	3
30	4	8
35	5	6
40	3	5
45	1	4
50	8	7
55	1	2



Minimum Spanning Tree Configuration

No. of nodes = 8		
No. of edges = 7		
u	v	wt
1	3	25
3	5	40
2	5	5
5	4	20
4	8	30
5	7	15
7	6	10

Minimum Sum of all edges is 145.

