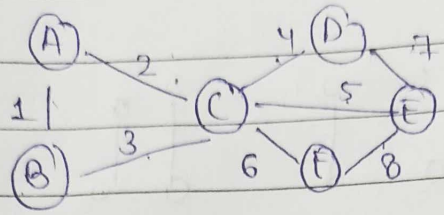
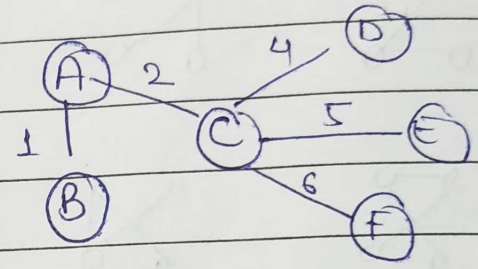
Tutorial 13

Yash Agarwal
2110460039



Kruskal

~~A-B 1~~
~~A-C 2~~
X ~~B-C 3~~
~~C-D 4~~
~~C-E 5~~
~~C-F 6~~
~~D-E 7~~
~~F-E 8~~

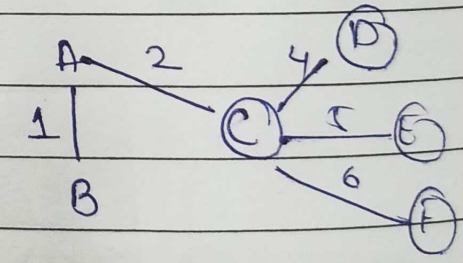


Prism

Source node A

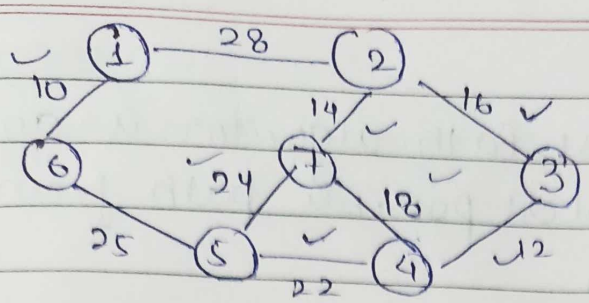
A - B	1	selected
A - C	2	selected
B - C	3	cycle
C - D	4	⇒ selected
C - E	5	selected
C - F	6	selected

finished ⇒ size = no of vertices



Date

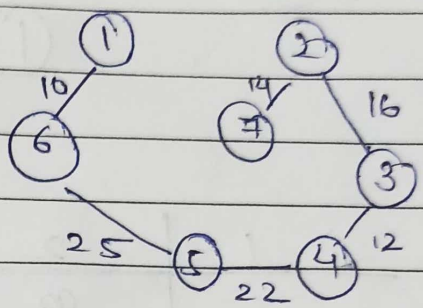
Page



2110039

Kruskal

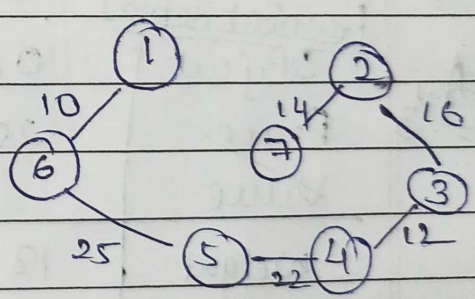
1-6	10
3-4	12
2-7	14
2-3	16
7-4	18
5-4	22
5-7	24
6-5	25
1-2	28



Prim's

Source (1)

	1-2	28
select	1-6	10
select	6-5	25
	5-7	24
select	5-4	22
select	4-3	12
	4-7	18
select	3-2	16
select	2-7	14

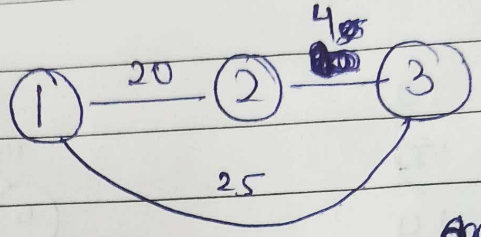


Stop because visited node = no. of vertices

Ans 2

Dijkstra's Shortest Path algoritm is an algoritm used to find a shortest possible path from source to all other vertices

Ex



1	2	3
	∞	∞
1	20	25
1,2	20	24

d(neb) > d[nodu] + weight[nod

then

d(neb) = d[nodu] + w[nod

1	2	3
0	20	24

Any

1-2 20

1-2-3 24

Ans 3

Fractional knapsack

Object	OB1	OB2	OB3
Price / value	25	24	15
weight	18	15	10
Price/wt	1.3	1.6	1.5

Capacity = 20

Sort wrt Price/wt.

OB2 > OB3 > OB1

24 + 1.5 x 5 = 24 + 7.5 = 31.5

is max

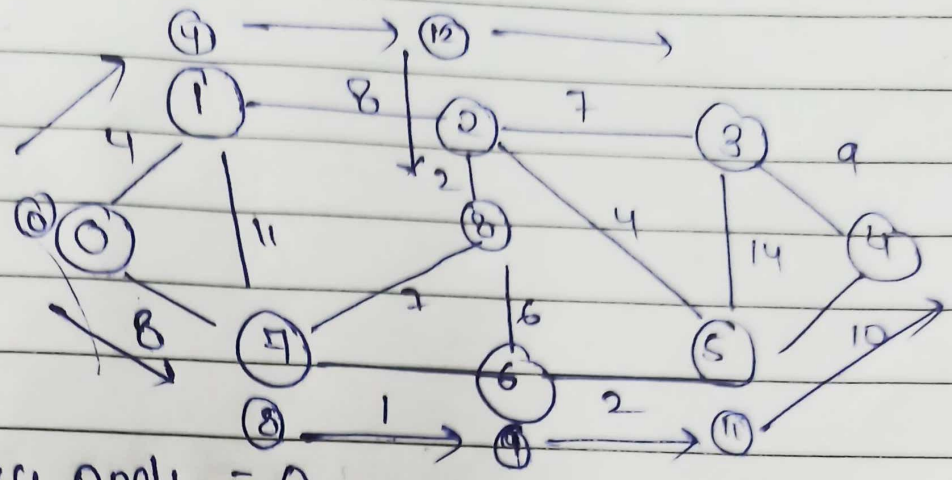
Coinage Problem

```
int main (){
int arr[ ]={100, 50, 20, 10, 5, 1};
int n=6;
int coin=0;
int Amount= 647;
while (Amount) {

        int i=0;
        for(i=0; i<n; i++) {
            if (Amount >= arr[i])
            {

                Amount -= arr[i];
                coin ++;
                break;
            }
        }
        if (i==n) {
            cout << "Not Possible";
        }
    }
    cout << coin ;
}
```

Ans 4



Source node = 0

0	1	2	3	4	5	6	7	8
0	∞	∞	∞	∞	∞	∞	∞	∞
0,1	4	12	∞	∞	∞	∞	8	∞
0,1,2	4	12	19	∞	17	∞	8	14
0,1,2,7	4	12	19	∞	17	9	8	14
0,1,2,7,6	4	12	19	∞	11	9	8	14
0,1,2,7,6,5	4	12	19	21	11	9	8	14
0,1,2,7,6,5,4,3	4	12	19	21	11	9	8	14

Ans

0	1	2	3	4	5	6	7	8
0	4	12	19	21	11	9	8	14

0-1 = 4
0-7 = 8
0-1-2 = 12
0-7-6 = 11
0-1-2-3 = 19
0-7-6-5-4 = 21
0-7-6-5 = 11
0-1-2-8 = 14