Greedy Method

• Requires either Mimimum or Maximum

• Fesible solution : Which satisfying the condition.

• Optimal solution : Fesible and Best. only one.

1. Greedy Method.
2. Dynamic Programing.
3. Branch and Bound.

Optimization Method

ALGORITHM. Greedy(a,n)

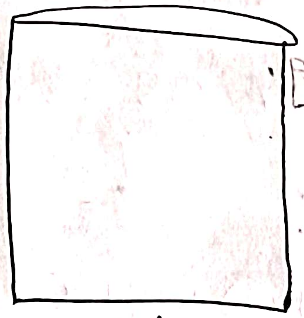
```
{
    for i=1 to n do
    {
        x= select(a);
        if Feasible (x) Then
            Solution= solution+x;
    }
}
```

Knapsack problem.

$n = 7$ Objects : 0 1 2 3 4 5 6 7

$m = 15$ Profits : P 10 5 15 7 6 18 3

Weights : W 2 3 5 7 1 4 1



15Kg

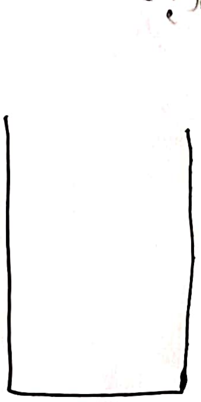
- Fill the bag as the profit be Maximum.
- Objects are dividiable.

Soln:

Step 1: Profit per kg.

So, $\frac{P}{W}$: 5 1.66 3 1 6 4.5 3

$$x : \begin{pmatrix} 2 & 2/3 & 5 & 0 & 1 & 4 & \\ x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \end{pmatrix}$$



15 − 1 = 14
14 − 2 = 12
12 − 4 = 8
8 − 5 = 3
3 − 1 = 2
2 − 2 = 0

Total profit:

$1 \times 2 + \frac{2}{3} \times 3 + 1 \times 5 + 0 \times 7$
$+ 1 \times 1 + 1 \times 4 + 1 \times 1$
$= 15$

Date:

41

Job sequencing with Deadlines

Q.1

n = 5

Jobs	J_1	J_2	J_3	J_4	J_5
profits	20	15	10	5	1
deadlines	2	2	1	3	3

Job Consider	Slot asign	Solution	Profit
J_1	[1,2]	J_1	20
J_2	[0,1] [1,2]	$J_1 J_2$	20+15
J_3	[0,1] [1,2]	J_1, J_2	20+15
J_4	[0,1][1,2][2,3]	J_1, J_2, J_3	20+15+5
J_5	"	"	40

• Profit Decreasing order.

Q.2

Jobs	J_1	J_2	J_3	J_4	J_5	J_6	J_7
Profits	35	30	25	20	15	12	5
Deadlines	3	4	4	2	3	1	2

0 —J_4— 1 —J_3— 2 —J_1— 3 —J_2— 4

Date:

42

Optical merge Pattern

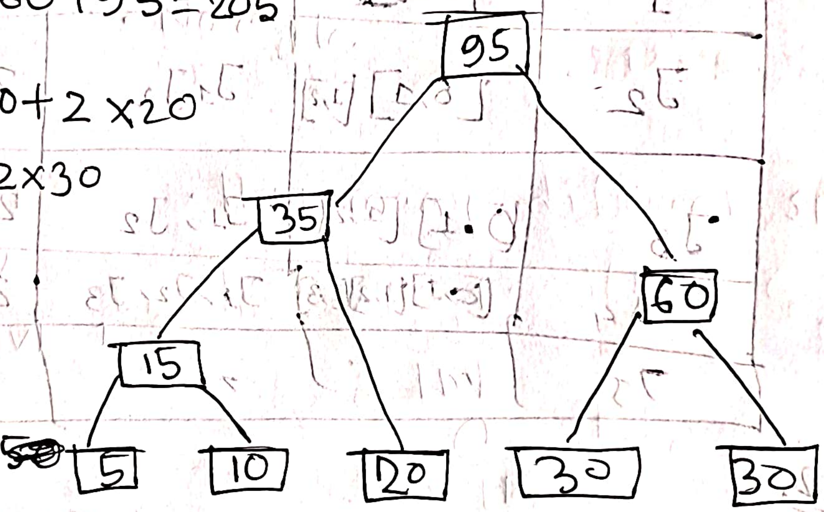
- Always select two small size elements for optimal solution.

Ex. lists: x_1 x_2 x_3 x_4 x_5

Sizes: 20 30 10 5 30

→ Sizes: $15+35+60+95=205$

or $3\times5+3\times10+2\times20$
$+2\times30+2\times30$
$=205$



Date:

43)

Huffman coding

Message → BCCABBDDAECCBBAEDDCC.

lenghth = 20

Ascii = 8 bit

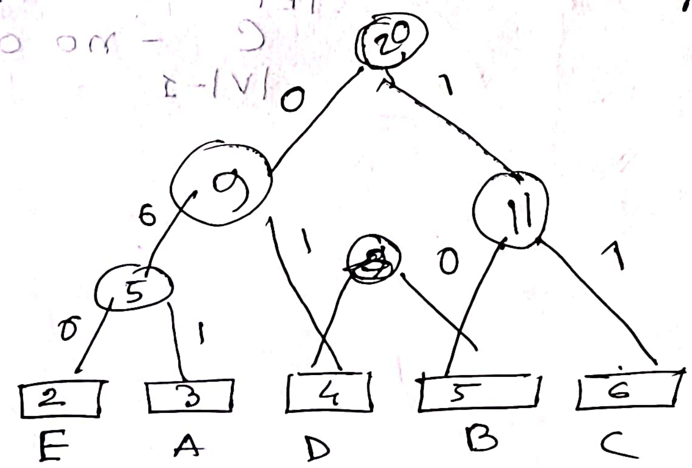
A
B
C
D
E

Total: 8X20 = bits.

Here,

Character	Count/frequency	Code
A	3	3X3=9
B	5	5X2=10
C	6	6X2=12
D	4	4X2=8
E	2	2X3=6

5x8 + 12 + Message = 45 bit



A = 001

B = 10

C = 11

D = 01

E = 000

Total Bits =

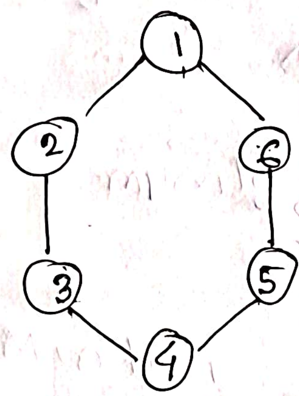
40+12+45
= 97

Date :

44.

Minimum cost spanning Tree

Here,



$G = (V, E)$

$V = \{1, 2, 3, 4, 5, 6\}$

$E = \{(1,2), (2,3), (3,4) \ldots\}$.

- Spanning Tree has n vertices and $n-1$ edges.
- No cycle.

So, $S \subseteq G$

$S = (V', E')$

$V' = V$

$E' = |V| - 1$

- How many different spanning Tree are possible $= {}^{6}C_{5}$

$= {}^{|E|}C_{|V|-1}$ – no. of cycles.

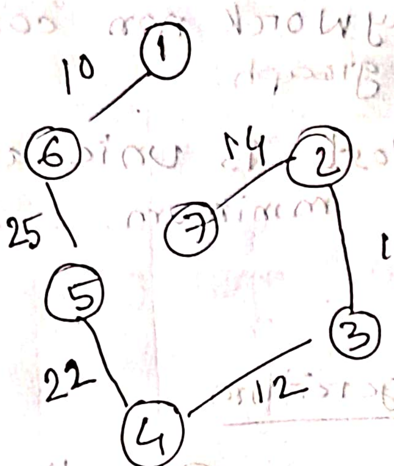
$= {}^{|E|}C_{|V|-1}$ – no of cycles.

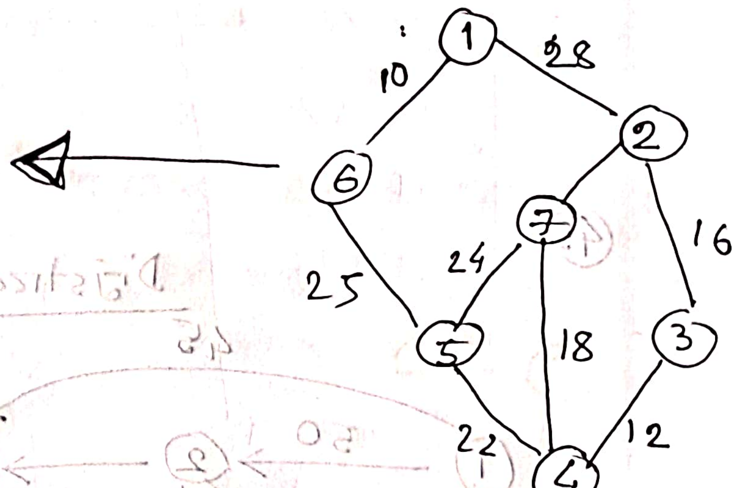
Prim's Algorithm

<u>Steps:</u>

1. Find the minimum edges.
2. Always select a minimum cost edge from the graph but make sure that it should be connected to already selected vertices.

<u>Solve:</u>

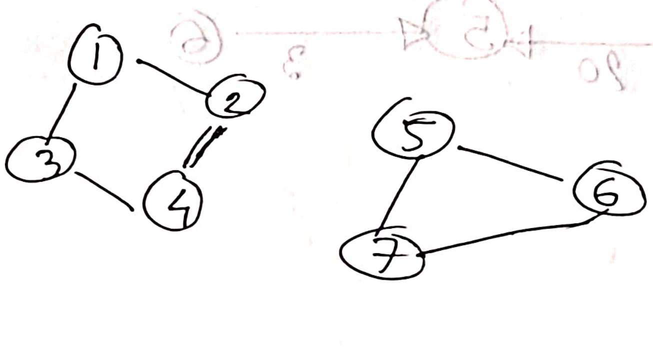




cost = 99.

- for non connected graph we cannot find spanning Tree.

<u>Ex.</u>



Kruskal's ALGORITHM

1. Always select the minimum cost edge.
2. Be careful about cycle.

Time complexity: $\Theta(|V||E|)$

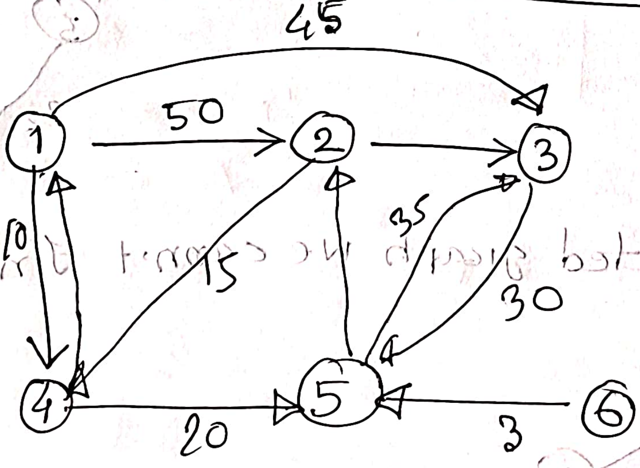
$\Theta = (n \cdot e) = \Theta(n^2)$.

If Min heap:

$\Theta(n \log n)$

- May work non connected graph.
- Cost is unic and minimum.

Dijstra Algorithm



Relaxation:

if $(d[u] + c(u,v) < d[v])$

$d[v] = d[u] + c(u,v)$

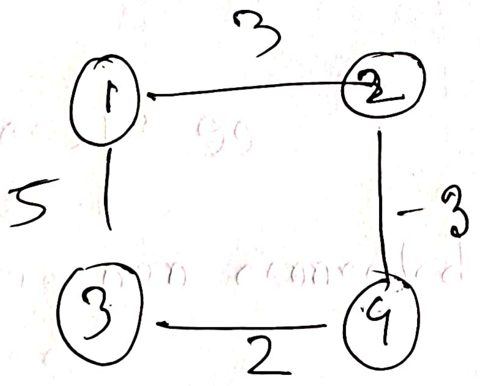
$\Theta(n^2)$ Worstcase

Starting vertex 1:

Selected V.	2	3	4	5	6
4	50	45	10	∞	∞
5	50	45	10	25	∞
2	45	45	10	25	∞
3	45	45	10	25	∞
6	45	45	10	25	∞

Drawback:

Dijstra Algorithm may work or may not work of negetive edges.



Work.