4/7/24 Week - 8

I) Knapsack problem Knapsack using n=4 :-

II) Prims Algorithm to find MST :-

```
#include<stdio.h>
#include<limits.h>

#define MAX 100

void prims(int n, int cost[MAX][MAX],
           int INF)
{
    int s[MAX], d[MAX], P[MAX), T[MAX]
       [2];
    int i, j, min, source, sum=0, k=0,
    u;

min = INF;
source = 0;
for(i=0; i<n; i++) {
 for(j=0; j<n; j++) {
    if(cost[i][j] != 0 && cost[i][j] <
     min) {
          min = cost[i][j];
          source = i;
       }
    }
 }
 for(i=0; i<n; i++) {
   s[i] = 0;
   d[i] = cost[source][i];
   P[i] = source;
 }
```

```
s[source] =1;

for(i=1;i<n; i++)
{
  min =INF;
  v =-1;
  for(j=0;j<n; j++)
  {
     if[s[j] ==0 && d[j] <=min)
     {
        min =d[j];
         u= j;
     }
  }
    T[K] [0] =u;
    T[K] [1] =p[u];
    K++;

   sum+=cost[u] [p] [u] ];
    s[u] =1;

   for(j=0;j<n;j++)
   {
      P[j] =u;

int main()
{
 int n, cost[MAX] [MAX], i,j ;
 int INF = INT_MAX;

printf("Enter the no of Vertices :");
  Scanf("%d", &n);
```

```
    printf ("Enter no of cost adjacency matrix
    for(i=0;i<n; i++)
      {
         for(j=0; j<n; j++)
         {  if (cost [i][j] ==9999)
          {  cost[i][j]=INF;
          }
      }
  }
  prims (n, cost, INF);
  return 0;
}
```

Output :- Enter no of Vertices : 5

Enter cost adjacency matrix :

```
 0    5    15  20    9999
 5    0    25  9999  9999
15   25     0  30    37
20  9999   30   0    35
9999 9999  37  35     0
```

Spanning tree exists MST is :

```
0->1
0->2
0->3
3->4
```

The cost of spanning tree is 75//

+ Prime c program:

```
#include <stdio.h>

# define N 4
# define CAPACITY 7

int max(int a, int b) {
    if(a>b)
    {
        return a;
    }
    return b;
}

Void Knapsack(int weight[], int profits[]){
    int i, w;
    int dp[N+1][capacity +1];

    for(i=0; i<=N; i++) {
      for(w=0 ; W<= capacity ; w++)
        {
          if(i==0 || W==0)
            dp[i][w]=0;
        else if (weight[i-1] <=w)
          dp[i][w] = max(profits[i-1]
          +dp[i-1][w-weights[i-1]],
           dp[i-1][w]);
        else
          dp[i][w] = dp[i-1][w];
    }
}
```

```
        printf("\nObjects selected in the
         Knapsack:\n");
        for(i=0; i<N; i++){
            if(selectedObjects[i]==1)
                printf("Objects %d, weights[i], p[i]
        }
}


int main(){
        int weight[N];
        int profits[N];

        printf("Enter the weights:\n");
        for(int i=0; i<n; i++){
                scanf("%d", &weights[i]);
        }

        printf("Enter the profits:\n");
        for(int i=0; i<n; i++){
                scanf("%d", &profits[i]);
        }

        printf("Knapsack Capacity:%d\n",
                capacity);
        printf("objects:\n");
        for(int i=0; i<n; i++){
                printf("Object %d-weight:%d, profit
                 :%d\n", i+1, weights[i], profits[i]
        }
        Knapsack(weight, profits);

        return 0;
```

3

```
Output :- Enter the weight:
  1 3 4 5

Enter the profits:
1 4 5 7

Knapsack capacity:7
  Objects
   Obj 1-Weight :1, Profit:1
   Obj 2 - Weight : 3, Profit : 4
   Obj 3 - Weight : 4; Profit : 5
   Obj 4 - Weight : 5, Profit : 7

  Table Value:
0    0    0    0    0    0    0    0
0    1    1    1    1    1    1    1
0    1    1    4    5    5    5    5
0    1    1    4    5    6    6    6
0    1    1    4    5    7    8    9

Object selected in Knapsack:
  Object 2 (W: 3 : P:0)
   Object 3 (W:4, P:5)
```

4/7/24