24/07/24 Week-9

→ Knapsack algorithm.

```
#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 weights[], 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(weights[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];
        }
    }
    int maxProfit = dp[N][CAPACITY];
    printf("Maximum profit: %d\n", maxProfit);
    int selectedobjects[N];
    int k=N, c=CAPACITY;
    while(k>0 && c>0) {
        if(dp[k][c] != dp[k-1][c]) {
            selectedobjects[k-1] = 1;
            c= c-weights[k-1]; }
```

```
    else {
        selectedObjects[k-1]=0;
    }
    k--;
}
printf("\nTable values (DP Table):\n");
for(i=0; i<=N; i++){
    for(w=0; w<=CAPACITY; w++) {
        printf("%d\t", dp[i][w]);
    }
    printf("\n");
}
printf("\nObjects selected in the knapsack:\n");
for(i=0; i<N; i++) {
    if(selectedObjects[i]==1)
        printf("Object %d (weight: %d, Profit: %d)\n",
i+1, weights[i], profits[i]); }
}
int main() {
    int weights[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(weights, profits);
    return 0;
}
```

output

Enter the weights: 1 3 4 5
Enter the profits: 1 4 5 7
knapsack capacity: 7
objects:
object 1 - weight: 1, Profit: 1
object 2 - weight: 3, Profit: 4
object 3 - weight: 4, Profit: 5
object 4 - weight: 5, Profit: 7
Maximum profit: 9
Table values (DP Table):

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	9
0	1	1	4	5	7	8	9

objects selected in knapsack:
object 2 (weight: 3, Profit: 4)
object 3 (weight: 4, Profit: 5)

→ Prims

```
#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], sum=0, i, j,
        k, min, u;
    min=INF;
    int 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;
    k=0;
    for(i=1; i<n; i++){
        min=INF;
        u=-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][l]=p[u];
    k++;
    sum+=cost[u][p[u]];
    s[u]=1;
    for(j=0; j<n; j++){
        if(s[j]==0 && cost[u][j]<d[j]){
            d[j]=cost[u][j];
            p[j]=u;}
        }
    }
    if(sum>=INF){
        printf("spanning tree does not exist\n");
    }else{
        printf("spanning tree exists and MST is:\n");
        for(i=0; i<n-1; i++){
            printf("(%d,%d)\n", T[i][0], T[i][1]);
        }
        printf("The cost of spanning tree is:%d\n", sum);
    }
}
int main(){
    int n, cost[MAX][MAX], i, j;
    int INF=INT_MAX;
    printf("Enter the number of vertices:");
    scanf("%d", &n);
    printf("Enter the cost adjacency matrix:\n");
    for(i=0; i<n; i++){
        for(j=0; j<n; j++){
            scanf("%d", &cost[i][j]);
            if(cost[i][j]==9999)
                cost[i][j]=INF;
        }
```

```
}
prims(n, cost, INF);
return 0;
}
```

Output

Enter the number of vertices: 5

Enter the cost adjacency matrix:

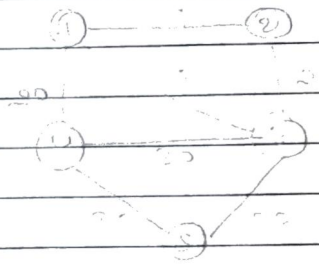
0 5 15 20 9999

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