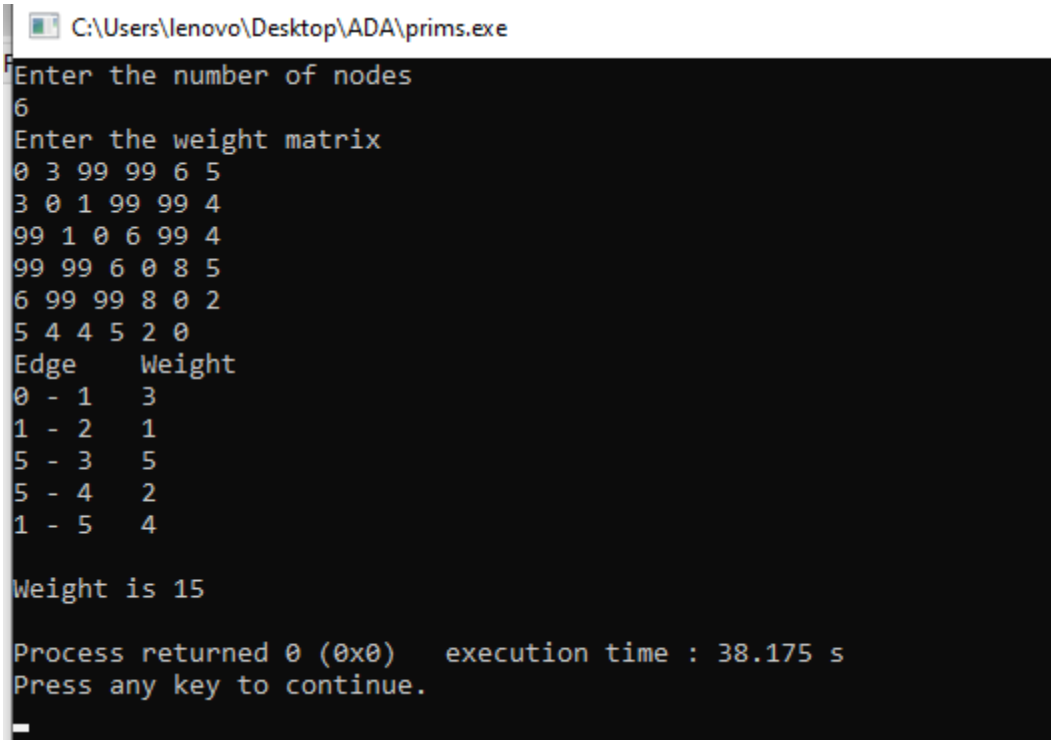
Find minimum cost spanning tree of a given undirected graph using prims algorithm

```
#include <limits.h>
#include <stdbool.h>
#include <stdio.h>
#define V 10

int minKey(int key[], bool mstSet[], int n)
{
        int min = INT_MAX, min_index;

        for (int v = 0; v < n; v++)
                if (mstSet[v] == false && key[v] < min)
                        min = key[v], min_index = v;

        return min_index;
}

int printMST(int parent[], int graph[V][V], int n)
{
   int weight = 0;
        printf("Edge \tWeight\n");
        for (int i = 1; i < n; i++)
                printf("%d - %d \t%d \n", parent[i], i,
                        graph[i][parent[i]]);

   for(int i=1;i<n;i++){
      weight += graph[i][parent[i]];
   }
```

```
   printf("\nWeight is %d \n",weight);
}

void prims(int graph[V][V],int n)
{
        int parent[V];
        int key[V];
        bool mstSet[V];

        for (int i = 0; i < n; i++)
                key[i] = INT_MAX, mstSet[i] = false;

        key[0] = 0;

        parent[0] = -1;

        for (int count = 0; count < n - 1; count++) {

                int u = minKey(key, mstSet, n);

                mstSet[u] = true;

                for (int v = 0; v < n; v++)

                        if (graph[u][v] && mstSet[v] == false
                                && graph[u][v] < key[v])
                                parent[v] = u, key[v] = graph[u][v];
        }
```

```
        printMST(parent, graph, n);
}

int main()
{
   int graph[V][V],n;
   printf("Enter the number of nodes\n");
   scanf("%d",&n);
   printf("Enter the weight matrix\n");
   for(int i=0;i<n;i++){
      for(int j=0;j<n;j++)
         scanf("%d",&graph[i][j]);
   }

        prims(graph,n);
        return 0;
}
```

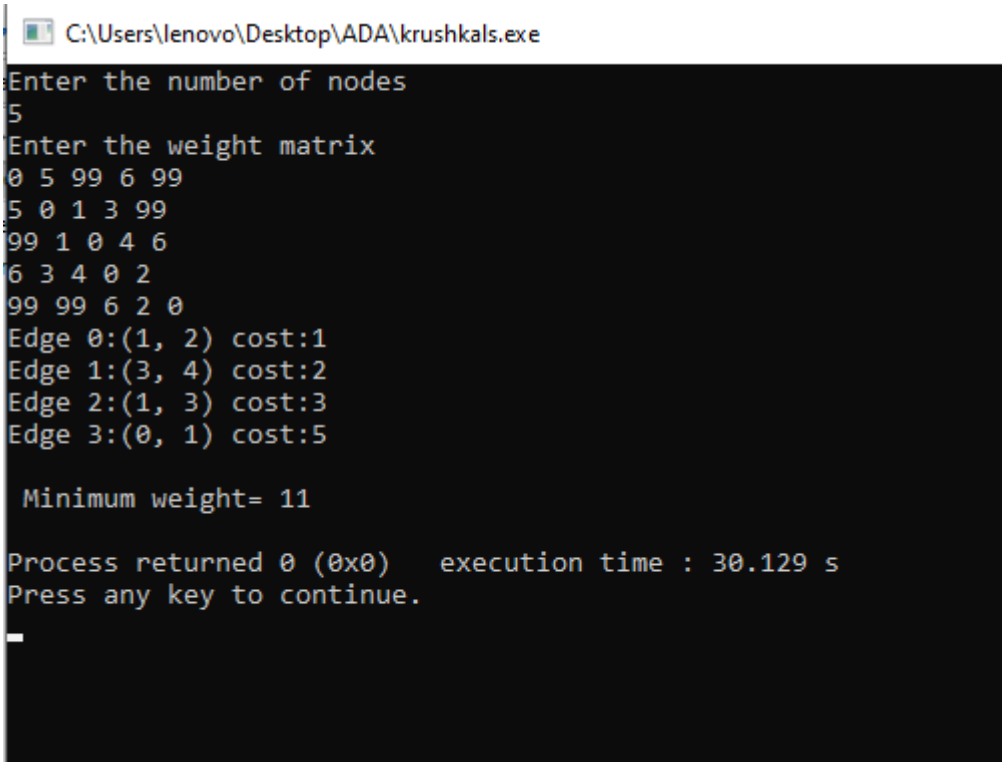


Find MST of a given undirected graph using krushkals algorithm.

```
#include<stdio.h>
#include <stdbool.h>
#define INT_MAX 99
#define V 5
int n;
int parent[V];
int find(int i)
{
   while (parent[i] != i)
      i = parent[i];
   return i;
}

void union1(int i, int j)
```

```
{
    int a = find(i);
    int b = find(j);
    parent[a] = b;
}

void kruskalMST(int cost[][V])
{
    int mincost = 0;

    for (int i = 0; i < V; i++)
        parent[i] = i;

    int edge_count = 0;
    while (edge_count < V - 1) {
        int min = INT_MAX, a = -1, b = -1;
        for (int i = 0; i < V; i++) {
            for (int j = 0; j < V; j++) {
                if (find(i) != find(j) && cost[i][j] < min) {
                    min = cost[i][j];
                    a = i;
                    b = j;
                }
            }
        }

        union1(a, b);
        printf("Edge %d:(%d, %d) cost:%d \n",
               edge_count++, a, b, min);
```

```
        mincost += min;
    }
    printf("\n Minimum weight= %d \n", mincost);

}

int main()
{

    int cost[V][V];
    printf("Enter the number of nodes\n");
    scanf("%d",&n);
    printf("Enter the weight matrix\n");
    for(int i=0;i<n;i++){
        for(int j=0;j<n;j++)
            scanf("%d",&cost[i][j]);
    }

    kruskalMST(cost);

    return 0;
}
```

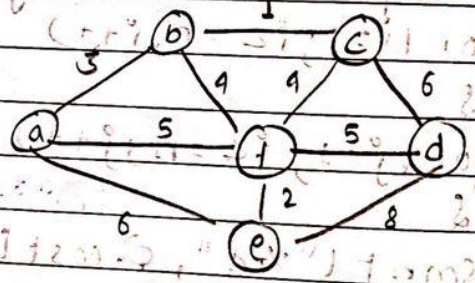

C:\Users\lenovo\Desktop\ADA\krushkals.exe
Enter the number of nodes
5
Enter the weight matrix
0 5 99 6 99
5 0 1 3 99
99 1 0 4 6
6 3 4 0 2
99 99 6 2 0
Edge 0:(1, 2) cost:1
Edge 1:(3, 4) cost:2
Edge 2:(1, 3) cost:3
Edge 3:(0, 1) cost:5
Minimum weight= 11
Process returned 0 (0x0) execution time : 30.129 s
Press any key to continue.

classmate
Date 3rd Aug
Page 27

Find minimum cost spanning tree of a given undirected graph using prim's algorithm.

```
#include<stdio.h>
int cost[10][10], vt[10], et[10][10], vis[10], j, n;
int sum=0;
int m=1;
int e=0;
void prims();
int main()
{
int i;
Printf("Enter no. of vertices \n");
Scanf("%d", &n);
Printf("Enter cost adjacency matrix \n");
for(i=1; i<=n; i++)
{
for(j=1; j<=n; j++)
{
Scanf("%d", &cost[i][j]);
}
vis[1]=0;
}
Prism();
Printf("edges of spanning tree\n");
for(i=1; i<=e; i++)
{
Printf("%d %d \t", et[i][0], et[i][1]);
Printf("weight = %d\n", sum);
return 0;
}
void prims()
{
int s, min, m, k, u, v;
vt[m]=1;
```

```
vis[m] = 1;
for(s=1; s<n; s++)
{
j=m;
min=999;
while (j>0)
{
k=vt[j];
for(m=2; m<=n; m++)
{
if(vis[m]==0)
{
if(cost[k][m]<min)
{
min = cost[k][m];
u = k;
v=m;
}
}
}
j--;
}
vt[++m] = v;
et[s][0] = u;
et[s][1] = v;
e++;
vis[v]=1;
sum = sum + min;
}
}
```



Output

Enter the no. of vertices 6
Enter cost ad. matrix

0	3	99	99	6	5
3	0	1	99	99	4
99	1	0	6	99	4
99	99	6	0	8	5
6	99	99	8	0	2
5	4	4	5	2	0

edges of spanning tree
1,2 2,3 3,6 6,5 6,4
weight = 15

Find MCST of a given undirected graph using kruskals.

```
#include <stdio.h>
int find (int v, int parent[10])
{
while (parent [v]! = v)
 {
 v = parent [v];
 }
return v;
}
void unionL (int i, int j , int parent[10])
{
if (i<j)
 parent [j]= i;
else
parent [i]= j;
}
void kruskal (int n, int a[10][10])
{
int count , k, min , sum , i, j, t[10][10], u, v, parent[10];
count=0;
k=0;
sum=0;
for (i=0; i<n; i++)
parent [i]=i;
while (count!=n-1)
{
min =999;
for (i=0; i<n; i++)
{
for (j=0; j<n; j++)
{
if (a[i][j]<min && a[i][j]!=0)
{
```

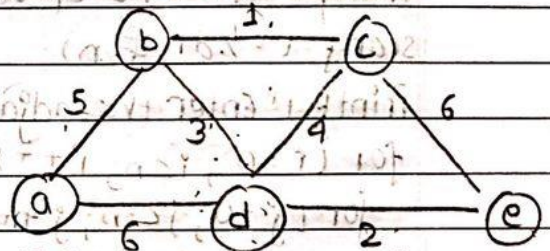
```
min=a[i][j];
u=i;
v=j;
}
}
}
i=find(u,parent);
j=find(v,parent);
if(i!=j)
{
union(i,j,parent);
t[k][0]=u;
t[k][1]=v;
k++;
count++;
sum=sum+a[u][v];
}
a[u][v]=a[v][u]=999;
}
if(count==n-1)
{
printf("spanning tree\n");
for(i=0;i<n-1;i++)
{
printf("%d %d\n",t[i][0],t[i][1]);
}
printf("cost of spanning tree = %d\n",sum);
}
else
printf("spanning tree doesn't exist\n");
}
int main()
{
int n,i,j,a[10][10];
```

```
printf("Enter the number of nodes\n");
scanf("%d", &n);
Printf("enter the adjacency matrix\n");
for(i=0; i<n; i++)
{
for(j=0; j<n; j++)
 scanf("%d", &a[i][j]);
}
Kruskal(n,a);
return 0;
}
```

Output

Enter the no. of node
5
Enter the ad. matrix

0	5	99	6	99
5	0	1	3	99
99	1	0	4	6
6	3	4	0	2
99	99	6	2	0



Spannin tree

1 - 2
3 - 4
1 - 3
0 - 1

cost of spanning tree = 11

Done
28/8/23