PROGRAM 25:

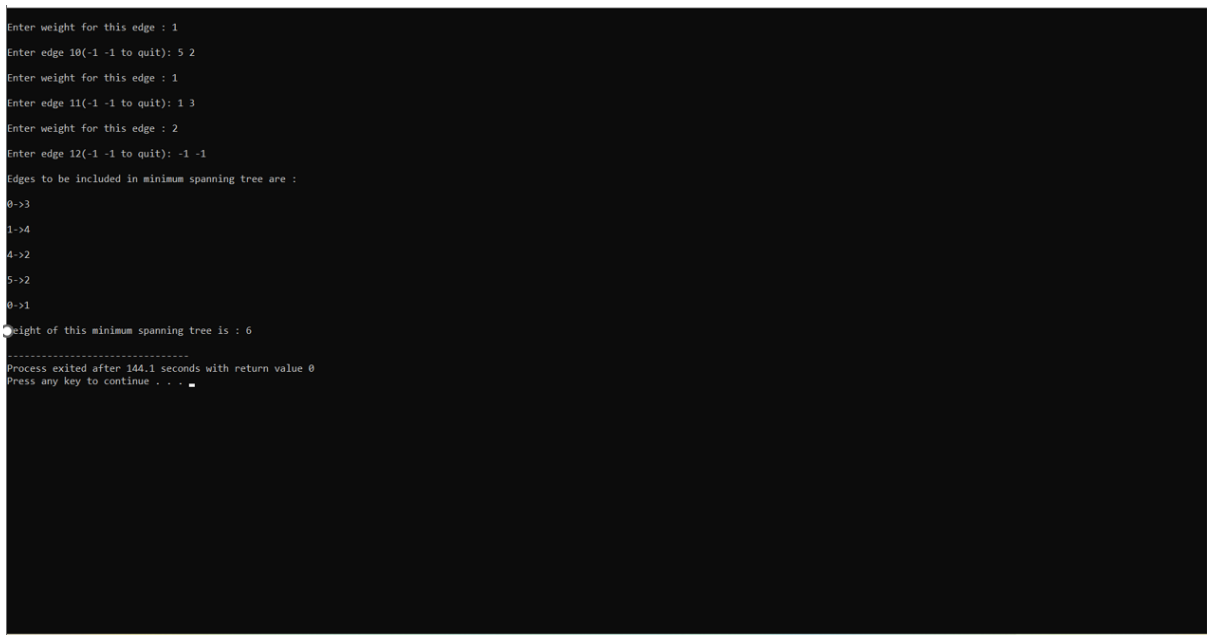
```
#include<stdio.h>
#include<stdlib.h>

#define MAX 100
#define NIL -1

struct edge
{
	int u;
	int v;
	int weight;
	struct edge *link;
}*front = NULL;

void make_tree(struct edge tree[]);
void insert_pque(int i,int j,int wt);
struct edge *del_pque();
int isEmpty_pque( );
void create_graph();

int n;

int main()
{
	int i;
	struct edge tree[MAX];
	int wt_tree = 0;
```

```
        create_graph();

        make_tree(tree);

        printf("\nEdges to be included in minimum spanning tree are :\n");
        for(i=1; i<=n-1; i++)
        {
                printf("\n%d->",tree[i].u);
                printf("%d\n",tree[i].v);
                wt_tree += tree[i].weight;
        }
        printf("\nWeight of this minimum spanning tree is : %d\n", wt_tree);

        return 0;

}

void make_tree(struct edge tree[])
{
        struct edge *tmp;
        int v1,v2,root_v1,root_v2;
        int father[MAX];
        int i,count = 0;

        for(i=0; i<n; i++)
                father[i] = NIL;



        while( !isEmpty_pque( ) && count < n-1 )
```

```
{
        tmp = del_pque();
        v1 = tmp->u;
        v2 = tmp->v;

        while( v1 !=NIL )
        {
                root_v1 = v1;
                v1 = father[v1];
        }
        while( v2 != NIL  )
        {
                root_v2 = v2;
                v2 = father[v2];
        }

        if( root_v1 != root_v2 )
        {
            count++;
                tree[count].u = tmp->u;
                tree[count].v = tmp->v;
                tree[count].weight = tmp->weight;
                father[root_v2]=root_v1;
        }
}

if(count < n-1)
{
        printf("\nGraph is not connected, no spanning tree possible\n");
        exit(1);
    }

}



void insert_pque(int i,int j,int wt)
{
    struct edge *tmp,*q;

    tmp = (struct edge *)malloc(sizeof(struct edge));
    tmp->u = i;
    tmp->v = j;
    tmp->weight = wt;


    if( front == NULL || tmp->weight < front->weight )
    {
        tmp->link = front;
        front = tmp;
    }
    else
    {
        q = front;
        while( q->link != NULL && q->link->weight <= tmp->weight )
            q = q->link;
        tmp->link = q->link;
        q->link = tmp;
        if(q->link == NULL)
```

```
                tmp->link = NULL;
        }
}
struct edge *del_pque()
{
        struct edge *tmp;
        tmp = front;
        front = front->link;
        return tmp;
}

int isEmpty_pque( )
{
        if ( front == NULL )
                return 1;
        else
                return 0;
}

void create_graph()
{
        int i,wt,max_edges,origin,destin;

        printf("\nEnter number of vertices : ");
        scanf("%d",&n);
        max_edges = n*(n-1)/2;

        for(i=1; i<=max_edges; i++)
        {
                printf("\nEnter edge %d(-1 -1 to quit): ",i);
                scanf("%d %d",&origin,&destin);
                if( (origin == -1) && (destin == -1) )
                break;
                printf("\nEnter weight for this edge : ");
                scanf("%d",&wt);
                if( origin >= n || destin >= n || origin<0 || destin<0)
                {
                printf("\nInvalid edge!\n");
                i--;
                }
                else
                insert_pque(origin,destin,wt);
        }
}
```

OUTPUT:



```
Enter weight for this edge : 1

Enter edge 10(-1 -1 to quit): 5 2

Enter weight for this edge : 1

Enter edge 11(-1 -1 to quit): 1 3

Enter weight for this edge : 2

Enter edge 12(-1 -1 to quit): -1 -1

Edges to be included in minimum spanning tree are :

0->3

1->4

4->2

5->2

0->1

eight of this minimum spanning tree is : 6

--------------------------------
Process exited after 144.1 seconds with return value 0
Press any key to continue . . .
```