

```

#include <stdio.h>

#include <limits.h>

#define vertices 6

int minimum_key(int k[], int mst[])
{
    int minimum = INT_MAX, min,i;
    for (i = 0; i < vertices; i++)
        if (mst[i] == 0 && k[i] < minimum )
            minimum = k[i], min = i;
    return min;
}

void prim(int g[vertices][vertices])
{
    /* create array of size equal to total number of vertices for storing the MST*/
    int parent[vertices];

    /* create k[vertices] array for selecting an edge having minimum weight*/
    int k[vertices];
    int mst[vertices];
    int i, count,edge,v; /*Here 'v' is the vertex*/
    for (i = 0; i < vertices; i++)
    {
        k[i] = INT_MAX;
        mst[i] = 0;
    }
    k[0] = 0; /*It select as first vertex*/
    parent[0] = -1; /* set first value of parent[] array to -1 to make it root of MST*/
    for (count = 0; count < vertices-1; count++)
    {

```

```

    edge = minimum_key(k, mst);
    mst[edge] = 1;
    for (v = 0; v < vertices; v++)
    {
        if (g[edge][v] && mst[v] == 0 && g[edge][v] < k[v])
        {
            parent[v] = edge, k[v] = g[edge][v];
        }
    }
}

/*Print the constructed Minimum spanning tree*/
printf("\n Edge \t Weight\n");
for (i = 1; i < vertices; i++)
    printf(" %d <-> %d   %d \n", parent[i], i, g[i][parent[i]]);

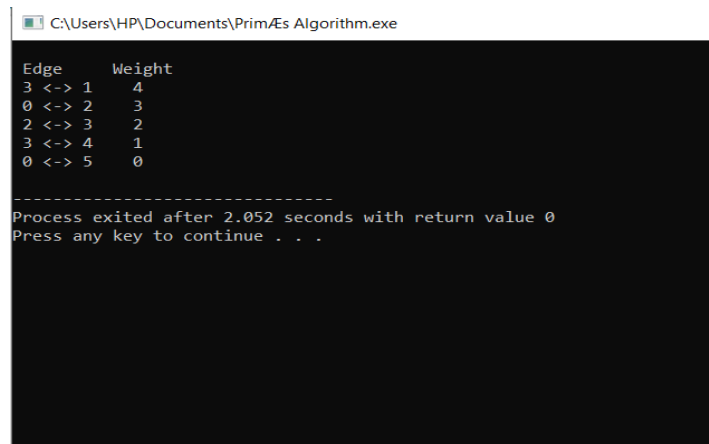
}

int main()
{
    int g[vertices][vertices] = {{0, 0, 3, 7, 0},
                                   {0, 0, 3, 4, 0},
                                   {3, 10, 0, 2, 6},
                                   {0, 9, 2, 0, 1},
                                   {0, 0, 6, 1, 0},
                                   };

    prim(g);

    return 0;
}

```



```

C:\Users\HP\Documents\Prim\Es Algorithm.exe

Edge      Weight
3 <-> 1    4
0 <-> 2    3
2 <-> 3    2
3 <-> 4    1
0 <-> 5    0

-----
Process exited after 2.052 seconds with return value 0
Press any key to continue . . .

```