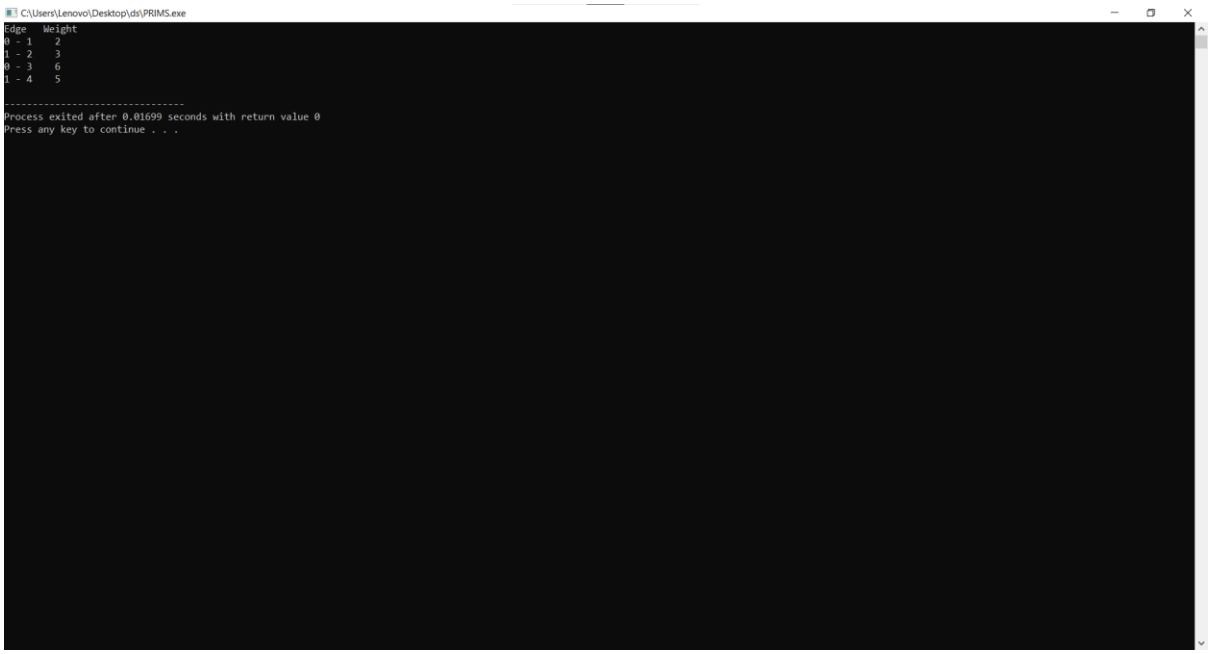
```
#include <stdio.h>
#include <limits.h>

#define V 5

int minKey(int key[], int mstSet[]) {
   int min = INT_MAX, min_index;
   int v;
   for (v = 0; v < V; v++)
      if (mstSet[v] == 0 && key[v] < min)
         min = key[v], min_index = v;

   return min_index;
}

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

void primMST(int graph[V][V]) {
   int parent[V]; // Array to store constructed MST
   int key[V], i, v, count;
   int mstSet[V];

   for (i = 0; i < V; i++)
      key[i] = INT_MAX, mstSet[i] = 0;

   // Always include first 1st vertex in MST.
```

```
    key[0] = 0;
    parent[0] = -1;

    // The MST will have V vertices
    for (count = 0; count < V - 1; count++) {
        int u = minKey(key, mstSet);
        mstSet[u] = 1;

        for (v = 0; v < V; v++)

            if (graph[u][v] && mstSet[v] == 0 && graph[u][v] < key[v])
                parent[v] = u, key[v] = graph[u][v];
    }


    printMST(parent, V, graph);
}

int main() {

    int graph[V][V] = { { 0, 2, 0, 6, 0 }, { 2, 0, 3, 8, 5 },
            { 0, 3, 0, 0, 7 }, { 6, 8, 0, 0, 9 }, { 0, 5, 7, 9, 0 }, };

    primMST(graph);

    return 0;
}
```


C:\Users\Lenovo\Desktop\ds\PRIMS.exe
Edge Weight
0 - 1 2
1 - 2 3
0 - 3 6
1 - 4 5
Process exited after 0.01699 seconds with return value 0
Press any key to continue . . .