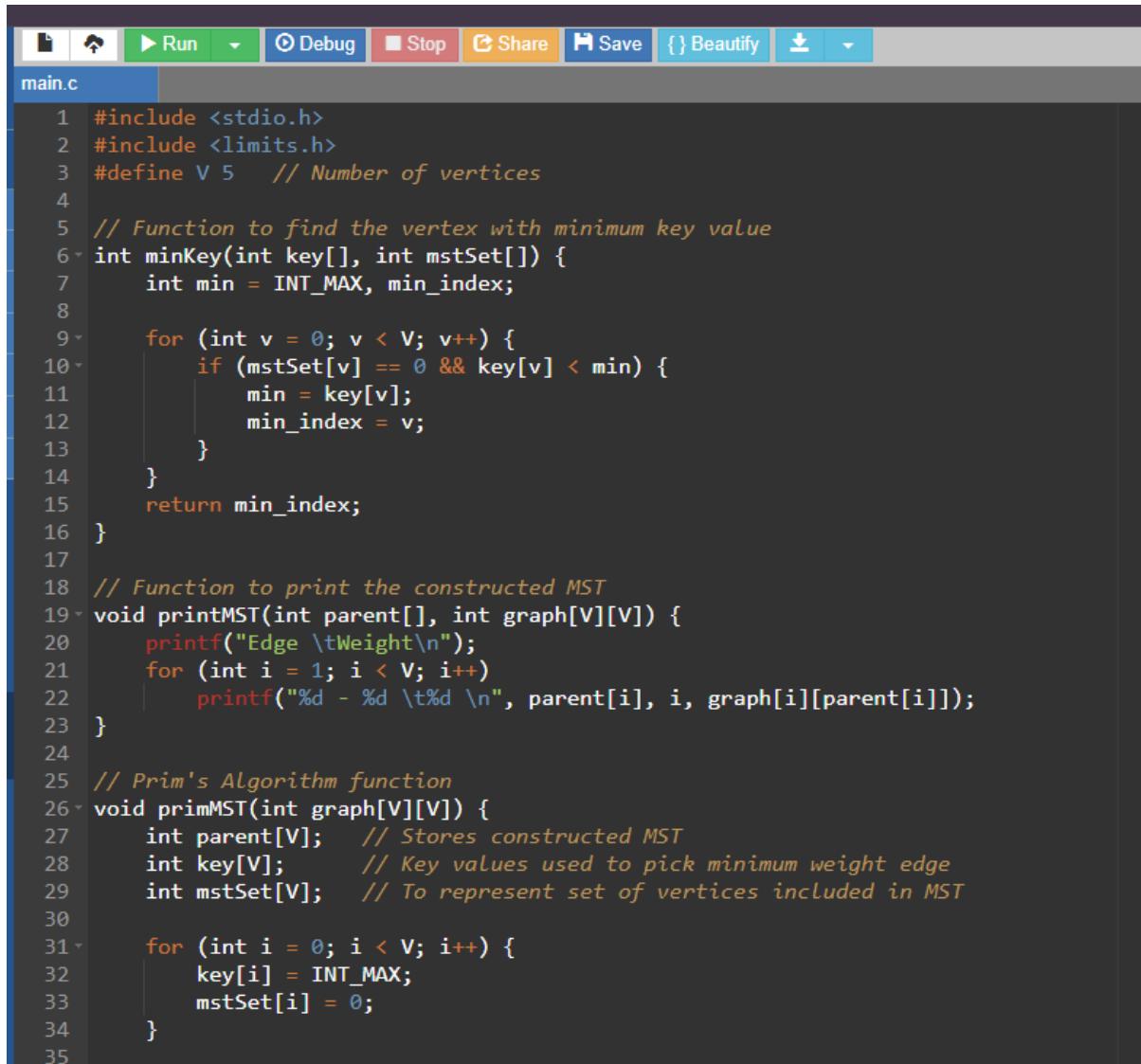


## Prims algorithm:

Code:



The screenshot shows a code editor window with a dark theme. The title bar has icons for file operations (New, Open, Save, etc.) and application-specific buttons (Run, Debug, Stop, Share, Save, Beautify). The current file is "main.c". The code itself is a C program implementing Prim's algorithm for Minimum Spanning Tree (MST) construction.

```
1 #include <stdio.h>
2 #include <limits.h>
3 #define V 5 // Number of vertices
4
5 // Function to find the vertex with minimum key value
6 int minKey(int key[], int mstSet[]) {
7     int min = INT_MAX, min_index;
8
9     for (int v = 0; v < V; v++) {
10         if (mstSet[v] == 0 && key[v] < min) {
11             min = key[v];
12             min_index = v;
13         }
14     }
15     return min_index;
16 }
17
18 // Function to print the constructed MST
19 void printMST(int parent[], int graph[V][V]) {
20     printf("Edge \tWeight\n");
21     for (int i = 1; i < V; i++)
22         printf("%d - %d \t%d \n", parent[i], i, graph[i][parent[i]]);
23 }
24
25 // Prim's Algorithm function
26 void primMST(int graph[V][V]) {
27     int parent[V]; // Stores constructed MST
28     int key[V]; // Key values used to pick minimum weight edge
29     int mstSet[V]; // To represent set of vertices included in MST
30
31     for (int i = 0; i < V; i++) {
32         key[i] = INT_MAX;
33         mstSet[i] = 0;
34     }
35 }
```

The screenshot shows a code editor window with the file 'main.c' open. The code implements Prim's algorithm to find a Minimum Spanning Tree (MST) for a given graph. The code includes a driver function 'main()' that initializes a 5x5 adjacency matrix 'graph' and calls the 'primMST()' function.

```
34     }
35
36     key[0] = 0;           // Start from first vertex
37     parent[0] = -1;      // First node is root of MST
38
39     for (int count = 0; count < V - 1; count++) {
40         int u = minKey(key, mstSet);
41         mstSet[u] = 1;
42
43         for (int v = 0; v < V; v++) {
44             if (graph[u][v] && mstSet[v] == 0 && graph[u][v] < key[v]) {
45                 parent[v] = u;
46                 key[v] = graph[u][v];
47             }
48         }
49     }
50
51     printMST(parent, graph);
52 }
53
54 // Driver code
55 int main() {
56     int graph[V][V] = {
57         {0, 2, 0, 6, 0},
58         {2, 0, 3, 8, 5},
59         {0, 3, 0, 0, 7},
60         {6, 8, 0, 0, 9},
61         {0, 5, 7, 9, 0}
62     };
63
64     primMST(graph);
65
66     return 0;
67 }
68
```

Output:

The screenshot shows a terminal window displaying the output of the 'main()' program. The output lists the edges and their weights that were included in the Minimum Spanning Tree, followed by a message indicating the program has finished execution.

```
Edge    Weight
0 - 1    2
1 - 2    3
0 - 3    6
1 - 4    5

....Program finished with exit code 0
Press ENTER to exit console.
```