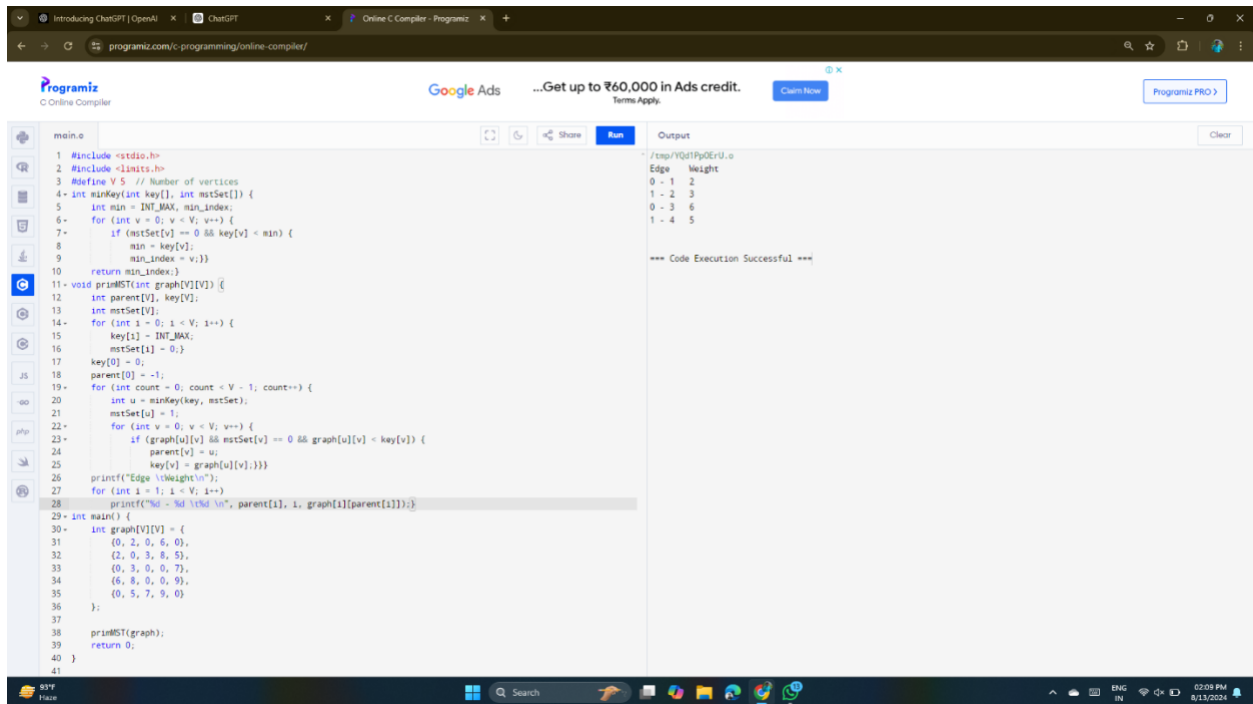


## 1. C Code for MINIMUM SPANNING TREE



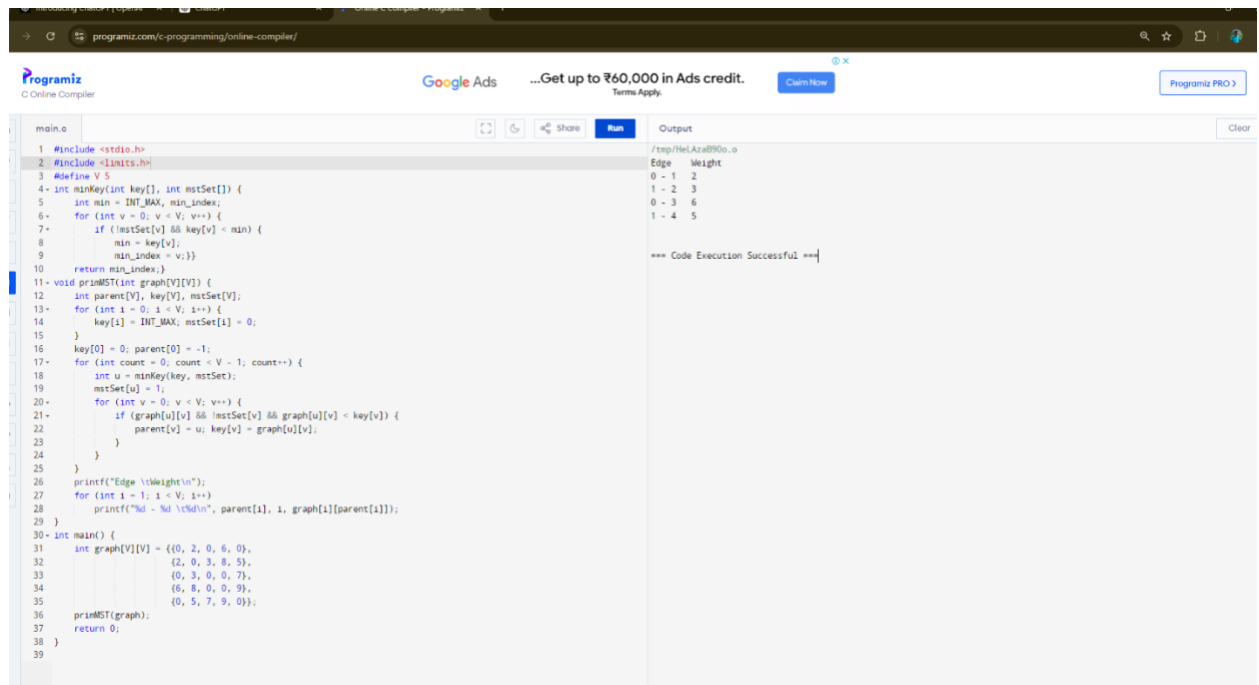
The screenshot shows the Programiz online C compiler interface. The code is for a Minimum Spanning Tree (MST) using Prim's algorithm. It defines a graph with 5 vertices and 9 edges. The edges are: (0,1) weight 2, (0,3) weight 6, (0,4) weight 5, (1,2) weight 3, (2,3) weight 8, (3,4) weight 9, (4,5) weight 7, (5,6) weight 8, and (6,7) weight 9. The output shows the MST edges and their weights: (0,1) weight 2, (0,3) weight 6, (0,4) weight 5, (1,2) weight 3, (2,3) weight 8, (3,4) weight 9, (4,5) weight 7, (5,6) weight 8, and (6,7) weight 9.

```
1 #include <stdio.h>
2 #include <limits.h>
3 #define V 5 // Number of vertices
4 int minKey(int key[], int mstSet[]) {
5     int min = INT_MAX, min_index;
6     for (int v = 0; v < V; v++) {
7         if (mstSet[v] == 0 && key[v] < min) {
8             min = key[v];
9             min_index = v;
10        }
11    }
12    return min_index;
13}
14 void primMST(int graph[V][V]) {
15    int parent[V], key[V], mstSet[V];
16    for (int i = 0; i < V; i++) {
17        key[i] = INT_MAX;
18        mstSet[i] = 0;
19    }
20    key[0] = 0;
21    parent[0] = -1;
22    for (int count = 0; count < V - 1; count++) {
23        int u = minKey(key, mstSet);
24        mstSet[u] = 1;
25        for (int v = 0; v < V; v++) {
26            if (graph[u][v] && mstSet[v] == 0 && graph[u][v] < key[v]) {
27                parent[v] = u;
28                key[v] = graph[u][v];
29            }
30        }
31        printf("Edge \tWeight\n");
32        for (int i = 1; i < V; i++) {
33            printf("%d - %d \t%d\n", parent[i], i, graph[i][parent[i]]);
34        }
35    }
36}
37 int main() {
38    int graph[V][V] = {
39        {0, 2, 0, 6, 0},
40        {2, 0, 3, 8, 5},
41        {0, 3, 0, 0, 7},
42        {6, 8, 0, 0, 9},
43        {0, 5, 7, 9, 0}
44    };
45    primMST(graph);
46    return 0;
47}
```

Output:

```
Edge \tWeight
0 - 1 2
1 - 2 3
0 - 3 6
1 - 4 5
*** Code Execution Successful ***
```

## 2. C Code for PRIMS ALGORITHM



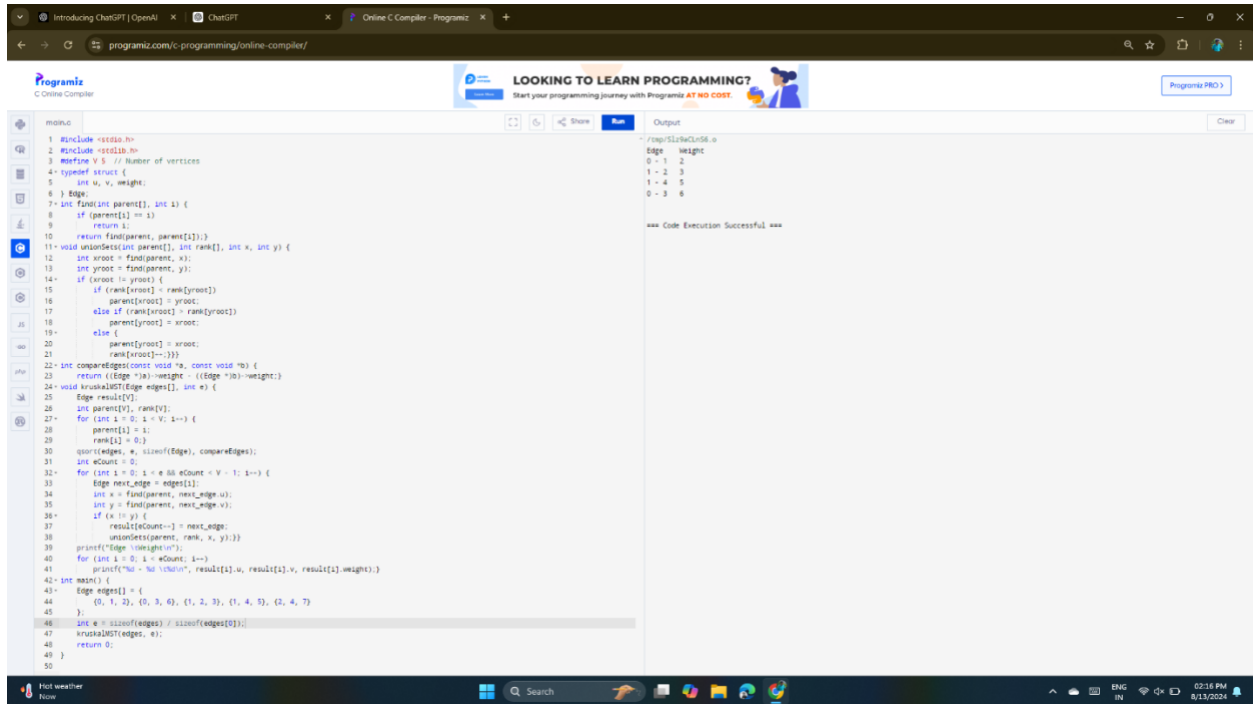
The screenshot shows the Programiz online C compiler interface. The code is for a Minimum Spanning Tree (MST) using Prim's algorithm. It defines a graph with 5 vertices and 9 edges. The edges are: (0,1) weight 2, (0,3) weight 6, (0,4) weight 5, (1,2) weight 3, (2,3) weight 8, (3,4) weight 9, (4,5) weight 7, (5,6) weight 8, and (6,7) weight 9. The output shows the MST edges and their weights: (0,1) weight 2, (0,3) weight 6, (0,4) weight 5, (1,2) weight 3, (2,3) weight 8, (3,4) weight 9, (4,5) weight 7, (5,6) weight 8, and (6,7) weight 9.

```
1 #include <stdio.h>
2 #include <limits.h>
3 #define V 5
4 int minKey(int key[], int mstSet[]) {
5     int min = INT_MAX, min_index;
6     for (int v = 0; v < V; v++) {
7         if (!mstSet[v] && key[v] < min) {
8             min = key[v];
9             min_index = v;
10        }
11    }
12    return min_index;
13}
14 void primMST(int graph[V][V]) {
15    int parent[V], key[V], mstSet[V];
16    for (int i = 0; i < V; i++) {
17        key[i] = INT_MAX;
18        mstSet[i] = 0;
19    }
20    key[0] = 0;
21    parent[0] = -1;
22    for (int count = 0; count < V - 1; count++) {
23        int u = minKey(key, mstSet);
24        mstSet[u] = 1;
25        for (int v = 0; v < V; v++) {
26            if (graph[u][v] && !mstSet[v] && graph[u][v] < key[v]) {
27                parent[v] = u;
28                key[v] = graph[u][v];
29            }
30        }
31        printf("Edge \tWeight\n");
32        for (int i = 1; i < V; i++) {
33            printf("%d - %d \t%d\n", parent[i], i, graph[i][parent[i]]);
34        }
35    }
36}
37 int main() {
38    int graph[V][V] = {
39        {0, 2, 0, 6, 0},
40        {2, 0, 3, 8, 5},
41        {0, 3, 0, 0, 7},
42        {6, 8, 0, 0, 9},
43        {0, 5, 7, 9, 0}
44    };
45    primMST(graph);
46    return 0;
47}
```

Output:

```
Edge \tWeight
0 - 1 2
1 - 2 3
0 - 3 6
1 - 4 5
*** Code Execution Successful ***
```

### 3. C Code for KRUSHKAL ALGORITHM



The screenshot displays a web browser window with the URL `programiz.com/c-programming/online-compiler/`. The page features a header for "Programiz C Online Compiler" and a promotional banner for "LOOKING TO LEARN PROGRAMMING?". The main area is divided into a code editor on the left and an output window on the right.

The code in the editor implements Kruskal's Algorithm in C. It includes headers `<stdio.h>` and `<stdlib.h>`, defines the number of vertices `V` as 5, and uses a union-find data structure to manage the graph. The algorithm sorts the edges by weight and iteratively adds them to the solution if they do not create a cycle. The final output is a list of edges in the Minimum Spanning Tree (MST).

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #define V 5 // Number of vertices
4 typedef struct {
5     int u, v, weight;
6 } Edge;
7 int find(int parent[], int i) {
8     if (parent[i] == i)
9         return i;
10    return find(parent, parent[i]);
11 void unionSets(int parent[], int rank[], int x, int y) {
12     int xroot = find(parent, x);
13     int yroot = find(parent, y);
14     if (xroot == yroot)
15         return;
16     if (rank[xroot] < rank[yroot])
17         parent[xroot] = yroot;
18     else if (rank[xroot] > rank[yroot])
19         parent[yroot] = xroot;
20     else {
21         parent[yroot] = xroot;
22         rank[xroot]++;
23     }
24 int compareEdges(const void *a, const void *b) {
25     return ((Edge *)a)->weight - ((Edge *)b)->weight;
26 void kruskalMST(Edge edges[], int e) {
27     Edge result[V];
28     int parent[V], rank[V];
29     for (int i = 0; i < V; i++) {
30         parent[i] = i;
31         rank[i] = 0;
32     }
33     qsort(edges, e, sizeof(Edge), compareEdges);
34     int eCount = 0;
35     for (int i = 0; i < e && eCount < V - 1; i++) {
36         Edge nextEdge = edges[i];
37         int u = find(parent, nextEdge.u);
38         int v = find(parent, nextEdge.v);
39         if (u != v) {
40             result[eCount++] = nextEdge;
41             unionSets(parent, rank, u, v);
42             printf("Edge %d\n", i);
43         }
44     }
45     for (int i = 0; i < eCount; i++)
46         printf("%d - %d\n", result[i].u, result[i].v, result[i].weight);
47 int main() {
48     Edge edges[] = {
49         {0, 1, 2}, {0, 3, 4}, {1, 2, 3}, {1, 4, 5}, {2, 4, 7}
50     };
51     int e = sizeof(edges) / sizeof(edges[0]);
52     kruskalMST(edges, e);
53     return 0;
54 }
```

The output window shows the following results:

```
Edge weight
0 - 1 2
1 - 2 3
1 - 4 5
0 - 3 4
*** Code Execution Successful ***
```

The Windows taskbar at the bottom indicates the system time as 02:16 PM on 8/13/2024.