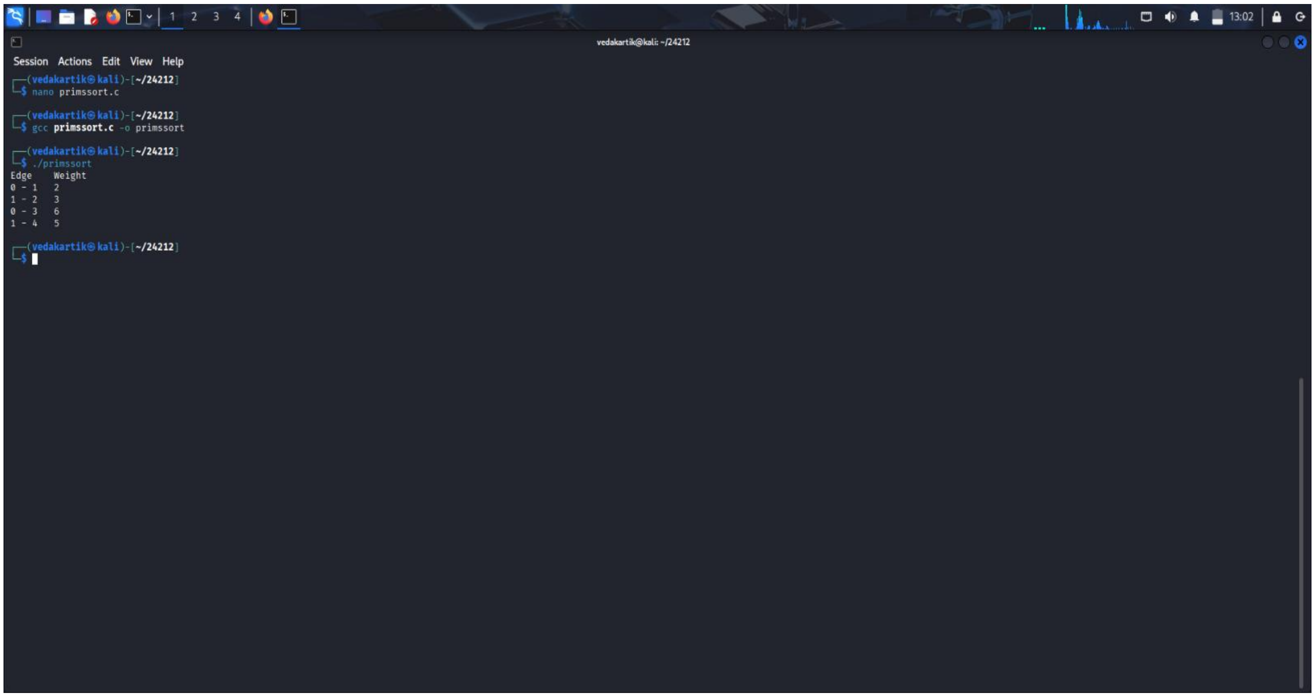
DAA LAB ASSIGNMENT-4

1. Prims Algorithm:

CODE:

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

#define V 5

int min_key(int key[], int mst_set[]) {
    int min = INT_MAX, min_index;
    for (int v = 0; v < V; v++) {
        if (mst_set[v] == 0 && key[v] < min) {
            min = key[v];
            min_index = v;
        }
    }
    return min_index;
}

void prim(int graph[V][V]) {
    int parent[V];
    int key[V];
    int mst_set[V];
    for (int i = 0; i < V; i++) {
        key[i] = INT_MAX;
        mst_set[i] = 0;
    }
    key[0] = 0;
    parent[0] = -1;
    for (int count = 0; count < V - 1; count++) {
        int u = min_key(key, mst_set);
        mst_set[u] = 1;
        for (int v = 0; v < V; v++) {
            if (graph[u][v] && mst_set[v] == 0 && graph[u][v] <
key[v]) {
                key[v] = graph[u][v];
                parent[v] = u;
            }
        }
    }
    printf("Edge \tWeight\n");
    for (int i = 1; i < V; i++) {
        printf("%d - %d \t%d\n", parent[i], i, graph[i][parent[i]]);
    }
}

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}
    };
    prim(graph);
    return 0;
}
```

<u>OUTPUT:</u>



```
┌──(vedakartik㉿kali)-[~/24212]
└─$ nano primssort.c

┌──(vedakartik㉿kali)-[~/24212]
└─$ gcc primssort.c -o primssort

┌──(vedakartik㉿kali)-[~/24212]
└─$ ./primssort
Edge    Weight
0 - 1   2
1 - 2   3
0 - 3   6
1 - 4   5

┌──(vedakartik㉿kali)-[~/24212]
└─$ 
```

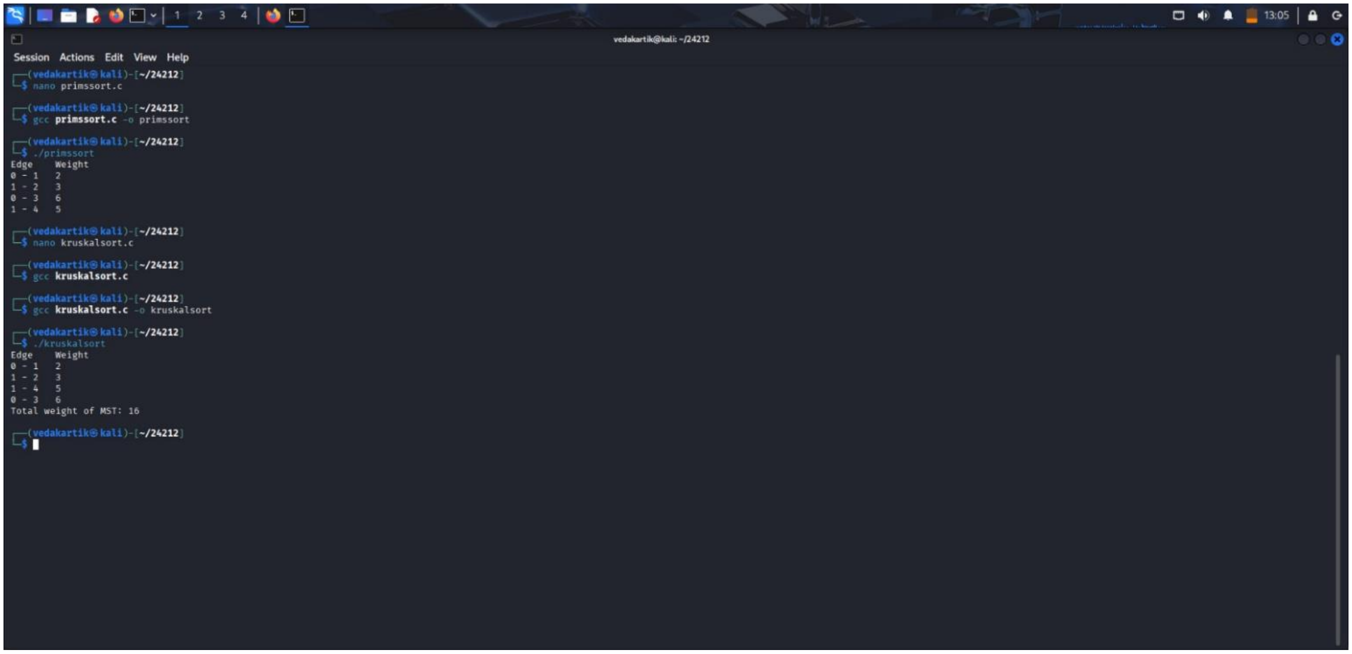
2. KRUSKAL'S ALGORITHM:

PROGRAM:

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


#define V 5
#define E 7
typedef struct {
   int u, v, weight;
} Edge;
int find(int parent[], int i) {
   if (parent[i] == -1)
       return i;
   return find(parent, parent[i]);
}
void union_set(int parent[], int x, int y) {
   int xroot = find(parent, x);
   int yroot = find(parent, y);
   parent[xroot] = yroot;
}

int compare(const void *a, const void *b) {
   return ((Edge*)a)->weight - ((Edge*)b)->weight;
}
```

```
void kruskal(Edge edges[], int parent[]) {
    qsort(edges, E, sizeof(Edge), compare);
    int mst_weight = 0;
    printf("Edge \tWeight\n");
    for (int i = 0; i < E; i++) {
        int u = edges[i].u;
        int v = edges[i].v;
        int w = edges[i].weight;
        int x = find(parent, u);
        int y = find(parent, v);
        if (x != y) {
            printf("%d - %d \t%d\n", u, v, w);
            mst_weight += w;
            union_set(parent, x, y);
        }
    }
    printf("Total weight of MST: %d\n", mst_weight);
}
int main() {
    Edge edges[] = {
        {0, 1, 2},
        {0, 3, 6},
        {1, 2, 3},
        {1, 4, 5},
        {2, 4, 7},
        {3, 4, 9},
        {2, 3, 8}
    };
    int parent[V];
    for (int i = 0; i < V; i++) { parent[i] = -
        1;
    }
    kruskal(edges, parent); return 0;
}
```

OUTPUT:



```
┌──(vedakartik㉿kali)-[~/24212]
└─$ nano primssort.c

┌──(vedakartik㉿kali)-[~/24212]
└─$ gcc primssort.c -o primssort

┌──(vedakartik㉿kali)-[~/24212]
└─$ ./primssort
Edge    Weight
0 - 1   2
1 - 2   3
0 - 3   6
1 - 4   5

┌──(vedakartik㉿kali)-[~/24212]
└─$ nano kruskalsort.c

┌──(vedakartik㉿kali)-[~/24212]
└─$ gcc kruskalsort.c

┌──(vedakartik㉿kali)-[~/24212]
└─$ gcc kruskalsort.c -o kruskalsort

┌──(vedakartik㉿kali)-[~/24212]
└─$ ./kruskalsort
Edge    Weight
0 - 1   2
1 - 2   3
1 - 4   5
0 - 3   6
Total weight of MST: 16

┌──(vedakartik㉿kali)-[~/24212]
└─$ █
```

G VEDA KARTIK

CH.SC.U4CSE24212