Name	Atul sharma
UID no.	2021700060
Experiment No.	06

AIM:	Prims Algorithm
ALGORITHM/ THEORY:	The working of Prim's algorithm can be described by using the following steps: ***Step 1:*** *Determine an arbitrary vertex as the starting vertex of the MST.* ***Step 2:*** *Follow steps 3 to 5 till there are vertices that are not included in the MST (known as fringe vertex).* ***Step 3:*** *Find edges connecting any tree vertex with the fringe vertices.* ***Step 4:*** *Find the minimum among these edges.* ***Step 5:*** *Add the chosen edge to the MST if it does not form any cycle.* ***Step 6:*** *Return the MST and exit*
PROGRAM:	

```c
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <limits.h>

#define MAX_VERTICES 100 // Maximum number of vertices
#define INF INT_MAX // Infinity

typedef struct {
    int u, v, weight; // Edge with vertices u and v and weight
w
} Edge;

int parent[MAX_VERTICES]; // Parent of each vertex
Edge edges[MAX_VERTICES]; // Edges in the MST
```

```c
int num_edges = 0; // Number of edges in the MST

// Find the parent of a vertex
int find(int v) {
    if (parent[v] != v) {
        parent[v] = find(parent[v]);
    }
    return parent[v];
}

// Union two sets of vertices
void union_sets(int u, int v) {
    parent[find(u)] = find(v);
}

// Comparator function for sorting edges by weight
int compare_edges(const void* a, const void* b) {
    Edge* e1 = (Edge*)a;
    Edge* e2 = (Edge*)b;
    return e1->weight - e2->weight;
}

// Find the MST of a graph with n vertices and m edges
void mst(int n, int m, Edge* edges) {
    // Initialize the parent of each vertex to itself
    for (int i = 0; i < n; i++) {
        parent[i] = i;
    }
    // Sort the edges by weight
    qsort(edges, m, sizeof(Edge), compare_edges);
    // Add edges to the MST until all vertices are connected
    for (int i = 0; i < m && num_edges < n - 1; i++) {
        int u = edges[i].u;
        int v = edges[i].v;
        if (find(u) != find(v)) {
            union_sets(u, v);
            edges[num_edges++] = edges[i];
        }
    }
}

int main() {
    int n, m;
    printf("Enter the number of vertices: ");
    scanf("%d", &n);
```

```
    printf("Enter the number of edges: ");
    scanf("%d", &m);
    printf("Enter the edges:\n");
    for (int i = 0; i < m; i++) {
        scanf("%d%d%d", &edges[i].u, &edges[i].v,
&edges[i].weight);
    }
    mst(n, m, edges);
    printf("The MST is:\n");
    for (int i = 0; i < num_edges; i++) {
        printf("%d - %d: %d\n", edges[i].u, edges[i].v,
edges[i].weight);
    }
    return 0;
}
```

RESULT:

```
PS C:\Users\maazs\OneDrive\Desktop\Studies\DAA\DAA Coding> cd "c:\Users\maazs\OneDrive\Desktop\Studies\DAA\DAA Coding\" ; if ($?) { gcc pri
ms.c -o prims } ; if ($?) { .\prims }
Enter the number of vertices: 6
Enter the number of edges: 6
Enter the edges:
2
4
2
4
6
7
6
8
9
2
10
1
10
12
3
12
8
4
The MST is:
2 - 10: 1
2 - 4: 2
```

CONCLUSION:

Understood how Prims Algorithm work to find out Minimum Spanning Tree.