

```

#include <stdio.h>

#define MAX 10

#define INF 999

int main() {

    int cost[MAX][MAX];

    int visited[MAX] = {0};

    int n, i, j, edges = 1;

    int min, u, v, totalCost = 0;


    printf("Enter number of vertices: ");

    scanf("%d", &n);


    printf("Enter the cost adjacency matrix (999 for no edge):\n");

    for (i = 0; i < n; i++)

        for (j = 0; j < n; j++)

            scanf("%d", &cost[i][j]);


    visited[0] = 1; // Start from vertex 0


    printf("\nEdges in Minimum Spanning Tree:\n");


    while (edges < n) {

        min = INF;

        for (i = 0; i < n; i++) {

            if (visited[i]) {

                for (j = 0; j < n; j++) {

                    if (!visited[j] && cost[i][j] < min) {

```

```

        min = cost[i][j];

        u = i;

        v = j;

    }

}

}

}

printf("Edge %d: (%d - %d) Cost: %d\n", edges, u, v, min);

totalCost += min;

visited[v] = 1;

edges++;

}

printf("\nMinimum Spanning Tree Cost = %d\n", totalCost);

return 0;

}

```

The screenshot displays the Programiz Online C Compiler interface. The top navigation bar includes the Programiz logo, a search bar, and links to 'Work faster in complex files.', 'Learn more', 'Adobe Illustrator', and 'Programiz PRO >'. The main workspace is divided into two panels: a code editor on the left and an output console on the right.

The code editor shows a C program for finding a Minimum Spanning Tree (MST) using Prim's algorithm. The code is as follows:

```

main.c
25- for (i = 0; i < n; i++) {
26-     if (visited[i]) {
27-         for (j = 0; j < n; j++) {
28-             if (!visited[j] && cost[i][j] < min) {
29-                 min = cost[i][j];
30-                 u = i;
31-                 v = j;
32-             }
33-         }
34-     }
35- }
36-
37- printf("Edge %d: (%d - %d) Cost: %d\n", edges, u, v, min);
38- totalCost += min;
39- visited[v] = 1;
40- edges++;
41- }
42-
43- printf("\nMinimum Spanning Tree Cost = %d\n", totalCost);
44- return 0;
45- }
46-

```

The output console shows the following results:

```

Enter number of vertices: 4
Enter the cost adjacency matrix (999 for no edge):
0 5 8 999
5 0 10 15
8 10 0 20
999 15
20 0

Edges in Minimum Spanning Tree:
Edge 1: (0 - 1) Cost: 5
Edge 2: (0 - 2) Cost: 8
Edge 3: (1 - 3) Cost: 15

Minimum Spanning Tree Cost = 28

=== Code Execution Successful ===

```