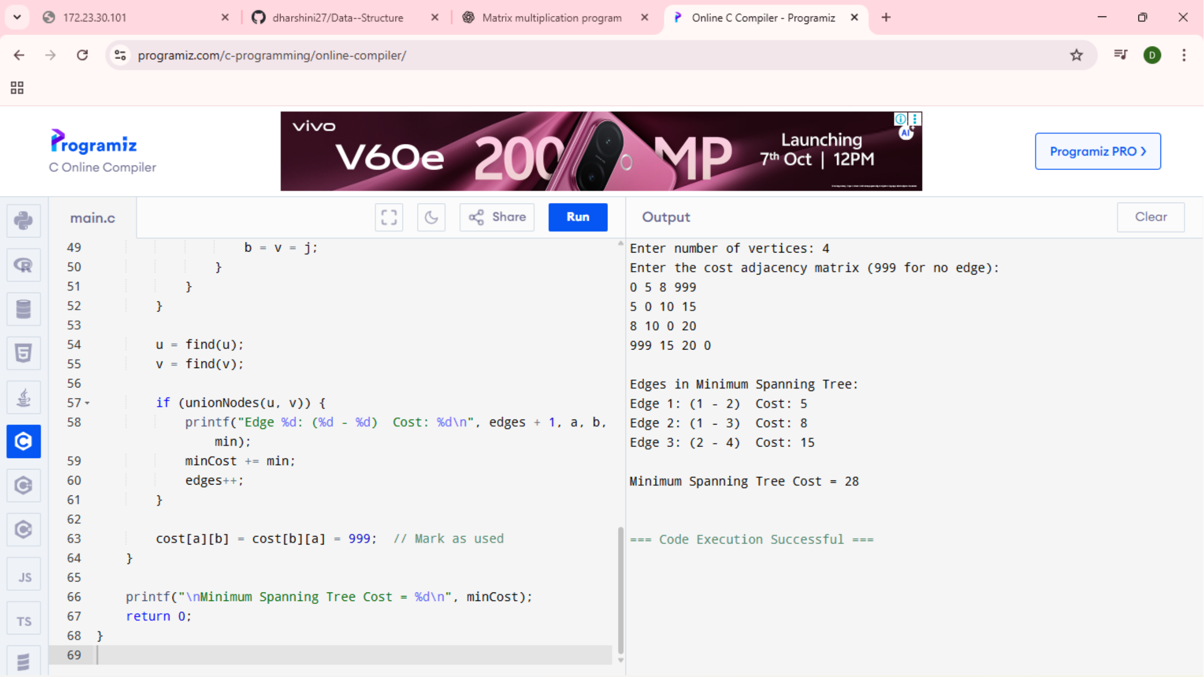
```
#include <stdio.h>

#define MAX 30

int parent[MAX];

// Function to find parent (with path compression)
int find(int i) {
    while (parent[i])
        i = parent[i];
    return i;
}

// Function to perform union
int unionNodes(int i, int j) {
    if (i != j) {
        parent[j] = i;
        return 1;
    }
    return 0;
}

int main() {
    int n, edges, u, v, a, b, i, j;
    int min, minCost = 0;
    int cost[MAX][MAX];

    printf("Enter number of vertices: ");
```

```
scanf("%d", &n);

printf("Enter the cost adjacency matrix (999 for no edge):\n");
for (i = 1; i <= n; i++)
    for (j = 1; j <= n; j++)
        scanf("%d", &cost[i][j]);

for (i = 1; i <= n; i++)
    parent[i] = 0;

edges = 0;

printf("\nEdges in Minimum Spanning Tree:\n");
while (edges < n - 1) {
    min = 999;
    for (i = 1; i <= n; i++) {
        for (j = 1; j <= n; j++) {
            if (cost[i][j] < min) {
                min = cost[i][j];
                a = u = i;
                b = v = j;
            }
        }
    }

    u = find(u);
    v = find(v);
```

```
        if (unionNodes(u, v)) {
            printf("Edge %d: (%d - %d)  Cost: %d\n", edges + 1, a, b, min);
            minCost += min;
            edges++;
        }

        cost[a][b] = cost[b][a] = 999;  // Mark as used
    }

    printf("\nMinimum Spanning Tree Cost = %d\n", minCost);
    return 0;
}
```


172.23.30.101
dharshini27/Data--Structure
Matrix multiplication program
Online C Compiler - Programiz
programiz.com/c-programming/online-compiler/
Programiz
C Online Compiler
Programiz PRO ›
main.c
Share
Run
Output
Clear

```
                    b = v = j;
                }
            }
        }

        u = find(u);
        v = find(v);

        if (unionNodes(u, v)) {
            printf("Edge %d: (%d - %d)  Cost: %d\n", edges + 1, a, b,
                min);
            minCost += min;
            edges++;
        }

        cost[a][b] = cost[b][a] = 999;  // Mark as used
    }

    printf("\nMinimum Spanning Tree Cost = %d\n", minCost);
    return 0;
}
```

```
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: (1 - 2)  Cost: 5
Edge 2: (1 - 3)  Cost: 8
Edge 3: (2 - 4)  Cost: 15

Minimum Spanning Tree Cost = 28


=== Code Execution Successful ===
```