```
// Prim's Algorithm in C

#include<stdio.h>
#include<stdbool.h>

#define INF 9999999

// number of vertices in graph
#define V 5

// create a 2d array of size 5x5
//for adjacency matrix to represent graph
int G[V][V] = {
  {0, 9, 75, 0, 0},
  {9, 0, 95, 19, 42},
  {75, 95, 0, 51, 66},
  {0, 19, 51, 0, 31},
  {0, 42, 66, 31, 0}};

int main() {
  int no_edge;  // number of edge

  // create a array to track selected vertex
  // selected will become true otherwise false
  int selected[V];

  // set selected false initially
  memset(selected, false, sizeof(selected));

  // set number of edge to 0
  no_edge = 0;

  // the number of egde in minimum spanning tree will be
  // always less than (V -1), where V is number of vertices in
  //graph

  // choose 0th vertex and make it true
  selected[0] = true;

  int x;  //  row number
  int y;  //  col number

  // print for edge and weight
  printf("Edge : Weight\n");

  while (no_edge < V - 1) {
    //For every vertex in the set S, find the all adjacent vertices
    // , calculate the distance from the vertex selected at step 1.
    // if the vertex is already in the set S, discard it otherwise
    //choose another vertex nearest to selected vertex  at step 1.

    int min = INF;
    x = 0;
    y = 0;

    for (int i = 0; i < V; i++) {
      if (selected[i]) {
        for (int j = 0; j < V; j++) {
```

```
          if (!selected[j] && G[i][j]) {  // not in selected and there is
an edge
            if (min > G[i][j]) {
              min = G[i][j];
              x = i;
              y = j;
            }
          }
        }
      }
    }
    printf("%d - %d : %d\n", x, y, G[x][y]);
    selected[y] = true;
    no_edge++;
  }

  return 0;
}
```

```
1 - 3 : 19
3 - 4 : 31
3 - 2 : 51


...Program finished with exit code 0
Press ENTER to exit console.
```