Data Structure and Algorithm Practicals

12. Practical based on Greedy Algorithm-Prim's

```
function createAdjMatrix(V, G) {

  var adjMatrix = [];

  // create N x N matrix filled with 0 edge weights between all vertices
  for (var i = 0; i < V; i++) {
    adjMatrix.push([]);
    for (var j = 0; j < V; j++) { adjMatrix[i].push(0); }
  }

  // populate adjacency matrix with correct edge weights
  for (var i = 0; i < G.length; i++) {
    adjMatrix[G[i][0]][G[i][1]] = G[i][2];
    adjMatrix[G[i][1]][G[i][0]] = G[i][2];
  }

  return adjMatrix;

}

function prims(V, G) {

  // create adj matrix from graph
  var adjMatrix = createAdjMatrix(V, G);

  // arbitrarily choose initial vertex from graph
  var vertex = 0;

  // initialize empty edges array and empty MST
  var MST = [];
  var edges = [];
  var visited = [];
  var minEdge = [null,null,Infinity];

  // run prims algorithm until we create an MST
  // that contains every vertex from the graph
  while (MST.length !== V-1) {

    // mark this vertex as visited
    visited.push(vertex);

    // add each edge to list of potential edges
    for (var r = 0; r < V; r++) {
      if (adjMatrix[vertex][r] !== 0) {
        edges.push([vertex,r,adjMatrix[vertex][r]]);
      }
    }

    // find edge with the smallest weight to a vertex
    // that has not yet been visited
    for (var e = 0; e < edges.length; e++) {
      if (edges[e][2] < minEdge[2] && visited.indexOf(edges[e][1]) === -1) {
        minEdge = edges[e];
      }
    }

    // remove min weight edge from list of edges
    edges.splice(edges.indexOf(minEdge), 1);

    // push min edge to MST
    MST.push(minEdge);

    // start at new vertex and reset min edge
    vertex = minEdge[1];
    minEdge = [null,null,Infinity];

  }

  return MST;

}

// graph vertices are actually represented as numbers
// like so: 0, 1, 2, ... V-1
var a = 0, b = 1, c = 2, d = 3, e = 4, f = 5;

// graph edges with weights
// diagram of graph is shown above
var graph = [
  [a,b,2],
  [a,c,3],
  [b,d,3],
  [b,c,5],
  [b,e,4],
  [c,e,4],
  [d,e,2],
  [d,f,3],
  [e,f,5]
];

// pass the # of vertices and the graph to run prims algorithm
console.log(prims(6, graph));
```