```
function dijkstrasAlgorithm (start, edges) {
    const vertices = edges.length

    const minDistances = []
    for (let edge in edges) {
        minDistances.push(Infinity)
    }

    minDistances[start] = 0

    const visited = new Set()

    while (visited.size !== vertices) {
        const [currMinDistance, vertex] = findVertexWithMinDistance(minDistances, visited)

        if (currMinDistance == Infinity) break

        visited.add(vertex)

        for (const edge of edges[vertex]) {
            const [destination, distanceToDestination] = edge

            if (visited.has(destination)) continue

            const newPath = currMinDistance + distanceToDestination

            const currPath = minDistances[destination]

            if (newPath < currPath) minDistances[destination] = newPath
        }
    }

    return minDistances.map((k) => k === Infinity ? -1 : k )
}

function findVertexWithMinDistance (distances, visited) {
    let minDistance = Infinity
    let vertex = -1

    for (let [currVertex, currMinDistance] of distances.entries()) {
        if (visited.has(currVertex)) continue

        if (currMinDistance < minDistance) {
            minDistance = currMinDistance
            vertex = currVertex
        }
    }

    return [minDistance, vertex]
}
```

```
console.log(dijkstrasAlgorithm(0, [
  [
    [1, 7]
  ],
  [
    [2, 6],
    [3, 20],
    [4, 3]
  ],
  [
    [3, 14]
  ],
  [
    [4, 2]
  ],
  [],
  []
]))
```