```
Algorithm ShortestPaths($v, cost, dist, n$)
// $dist[j]$, $1 \leq j \leq n$, is set to the length of the shortest
// path from vertex $v$ to vertex $j$ in a digraph $G$ with $n$
// vertices. $dist[v]$ is set to zero. $G$ is represented by its
// cost adjacency matrix $cost[1:n, 1:n]$.
{
    for $i := 1$ to $n$ do
    { // Initialize $S$.
        $S[i]$ := false; $dist[i] := cost[v, i]$;
    }
    $S[v]$ := true; $dist[v] := 0.0$; // Put $v$ in $S$.
    for $num := 2$ to $n$ do
    {
        // Determine $n - 1$ paths from $v$.
        Choose $u$ from among those vertices not
        in $S$ such that $dist[u]$ is minimum;
        $S[u]$ := true; // Put $u$ in $S$.
        for (each $w$ adjacent to $u$ with $S[w]$ = false) do
            // Update distances.
            if ($(dist[w] > dist[u] + cost[u, w])$) then
                    $dist[w] := dist[u] + cost[u, w]$;
    }
}
```

Algorithm 4.15 Greedy algorithm to generate shortest paths