3.6.1 Algorithm

The algorithm for single source shortest path is given as

```
Algorithm Single_Short_Path( p,cost,Dist,n)
{
// cost is an adjacency matrix storing cost of each edge i.e. cost[1 : n,1 : n]. Given
// graph can be represented by cost.
```

```
// Dist is a set that stores the shortest path from the source vertex 'p' to any other
// vertex in the graph.
// S stores all the visited vertices of graph. It is of Boolean type array.
// initially
 for i ←1 to n do
{
 S[i] ← 0;
  Dist ← cost[p,i];
}
S[p] ←1 // set p^th vertex to true in array S and i.e. put p in S
Dist[p] ← 0.0;
for val ← 2 to n–2 do
{
   // obtain n–1 paths from p
 Choose q from the vertices that are not visited (not in S) and with minimum distance.
Dist[q] = min{Dist[i]};
  S[q] ← 1// put q in S
  /*update the Distance values of the other nodes*/
   for (all node r adjacent to q with S[r] = 0) do
     if( Dist[r]>(Dist[q]+cost[p,q])) then
        Dist[r] ← Dist[q]+Dist[p,q];
  }
 }
```