Dijkashtra

```
graph = {
   'A':[('B',1),('C',4)], # ('B',1) is tuple of list.
   'B':[('A',1),('C',2),('D',5)],
   'C':[('A',4),('B',2),('D',1)],
   'D':[('B',5),('C',1)]
}
source = 'A'
distances = {node: float('inf') for node in graph} # node is dictionary
distances[source] = 0
visited = set()

while len(visited) < len(graph):
   min_node = None
   min_distance = float('inf')

   for node in graph:
      if node not in visited and distances[node] < min_distance:
         min_distance = distances[node]
         min_node = node

   if min_node is None :
      break

   visited.add(min_node)
   #print()
   # print(visited)
   # print(min_node)

   for neighbor, weight in graph[min_node]:
      new_distance = distances[min_node]+ weight
```

```
        if new_distance < distances[neighbor]:
            distances[neighbor] = new_distance
            #print(distance)


print()
for node, dist in distances.items():
    print(f"Shortest distance from {source} to {node}: {dist}")
```