

class topology:

def init (self, arr):

self.nodes = arr

self.edges = []

def add_direct_connection (self, p1, p2, cost):

self.edges.append ((p1, p2, cost))

self.edges.append ((p2, p1, cost))

def dist_use_route (self)

import collections

for node in self.nodes:

dist = collections.defaultdict (int)

next_hop = (node, node)

for otn in self.nodes:


```
if obj == node:
    dist[obj] = 1000000
```

```
for i in range(len(self.nodes)-1):
    for edge in self.edges:
        src, dest, cost = edge
        if dist[src] + cost < dist[dest]:
            dist[dest] = dist[src] + cost
```

```
src.print(node, dist, next_hop):
```

```
for dest, cost in dist.items():
    print(f"next_hop {dest}")
nodes = ['A', 'B', 'C', 'D', 'E']
t = topology(nodes)
```

```
t.add_direct_connection(A, B, 1)
t.add_direct_connection(A, C, 5)
t.add_direct_connection(B, C, 3)
t.add_direct_connection(B, E, 9)
t.add_direct_connection(C, D, 4)
t.add_direct_connection(D, F, 2)
```

```
t.dist_use_route()
```

```
nodes = input("Enter nodes").split()
```

```
t = topology(nodes)
```

```
edges = int(input("Edges:"))
```

```
t.dist_use_route()
```