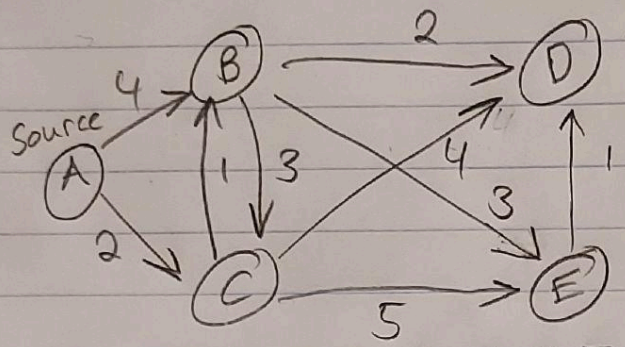
Dijkstra's Algorithm

- Finds shortest path from source node to all other nodes.
- Works for weighted and directed graphs.



Unvisited nodes: B C D E Dist: A B C D E

0 ? ? ? ?

0 4 2 ? ?

Go to node with smallest edge (node must be unvisited)

Unvisited nodes: B D E Go to B

0 3 2 6 7

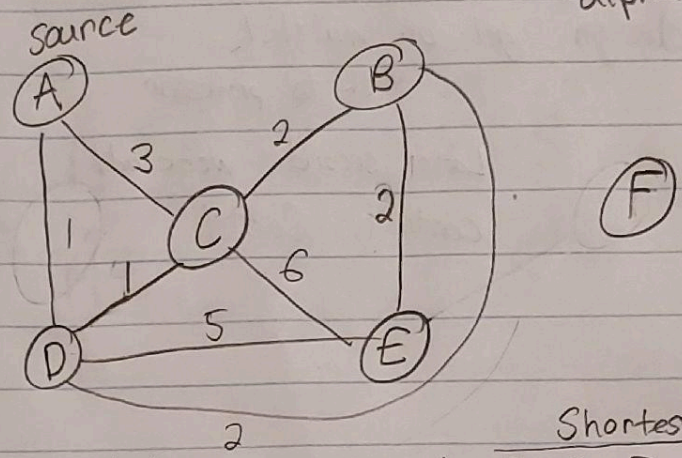
Unvisited nodes: D E

0 3 2 5 6

Repeat Process

I am going to implement the following graph in my code:

alphabet_graph



	Unvisited	Shortest path A B C D E F
1	B C D E F	0 ∞ 3 1 ∞ ∞
2	B C E F	0 1+2 1+1 1 1+5 ∞
3	B E F	0 3 2 1 6 ∞
4	E F	0 3 2 1 3+2 ∞
5	f	0 3 2 1 5 ∞
6		0 3 2 1 5 ∞

A to B
A->D->B

A to C
A->D->C

A to D
A->D

A to E
A->D->B->E

Pseudocode

```
all nodes distance = ∞
source node distance = 0
Queue = vector of unvisited nodes
current = source node
while Queue is not empty:
    for all of current's edges:
        if destination -> distance > current -> distance + edge weight
            dest->dist = cur->dist + edge weight
            dest->prev = current
    update current and remove old current from queue
Print out dists and routes.
```