Finding the shortest path in a maze: Dijkstra's algorithm

Łukasz Kwiecień

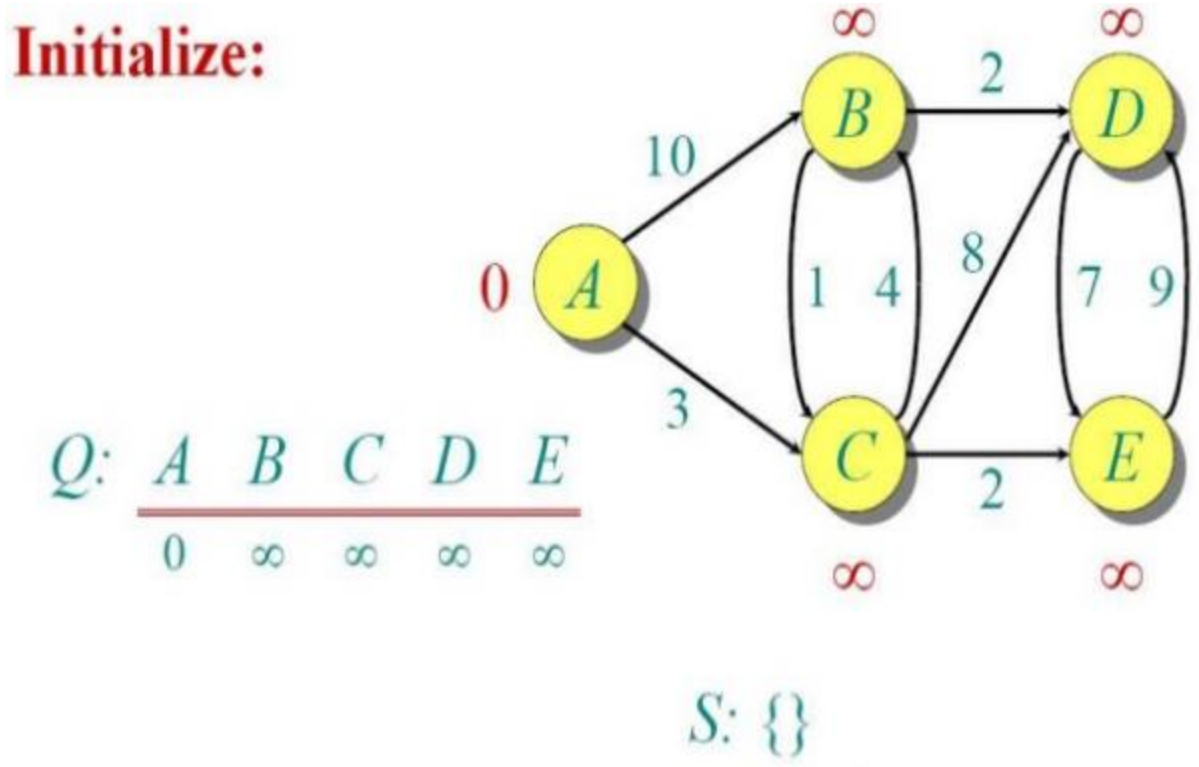
Dijkstra's Algorithm

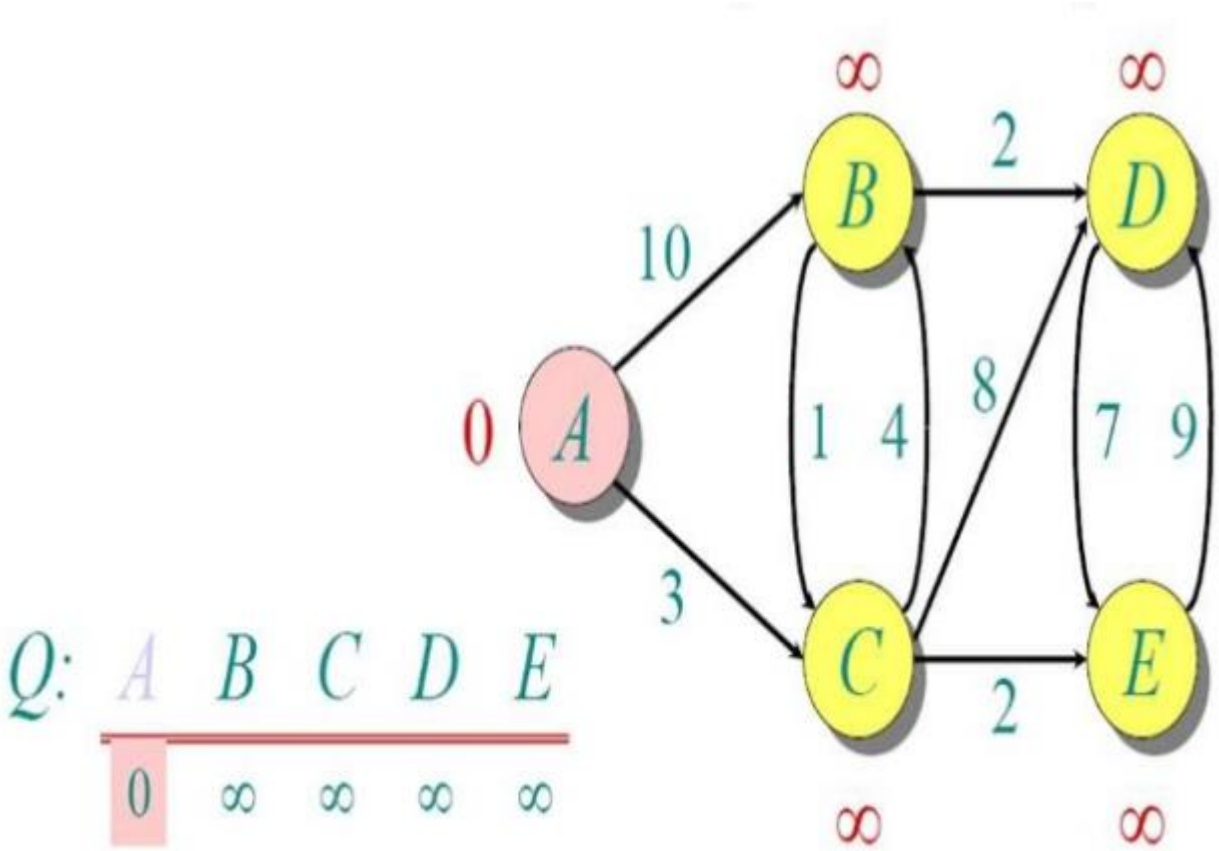
- Is a solution to the single-source shortest path problem in graph theory
- Created by Edsger W. Dijkstra in 1956
- Input – weighted graph and source vertex
- Output – lengths of shortest paths from a given source vertex to all other vertices

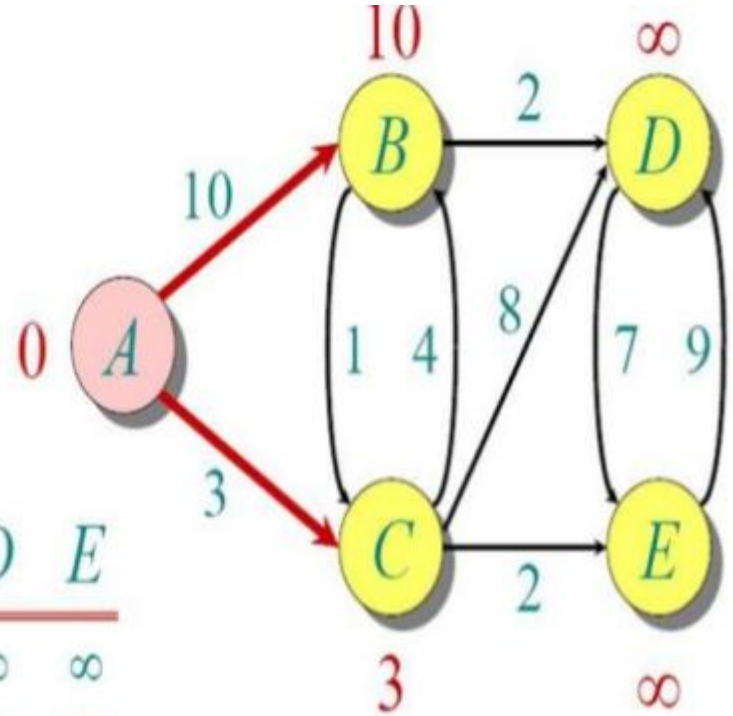
Pseudocode

```
1:  function Dijkstra(Graph, source):
2:      for each vertex v in Graph:                 // Initialization
3:          dist[v] := infinity                     // initial distance from source to vertex v is set to infinite
4:          previous[v] := undefined                // Previous node in optimal path from source
5:      dist[source] := 0                           // Distance from source to source
6:      Q := the set of all nodes in Graph          // all nodes in the graph are unoptimized - thus are in Q
7:      while Q is not empty:                       // main loop
8:          u := node in Q with smallest dist[ ]
9:          remove u from Q
10:         for each neighbor v of u:               // where v has not yet been removed from Q.
11:             alt := dist[u] + dist_between(u, v)
12:             if alt < dist[v]                    // Relax (u,v)
13:                 dist[v] := alt
14:                 previous[v] := u
15:     return previous[ ]
```

Example

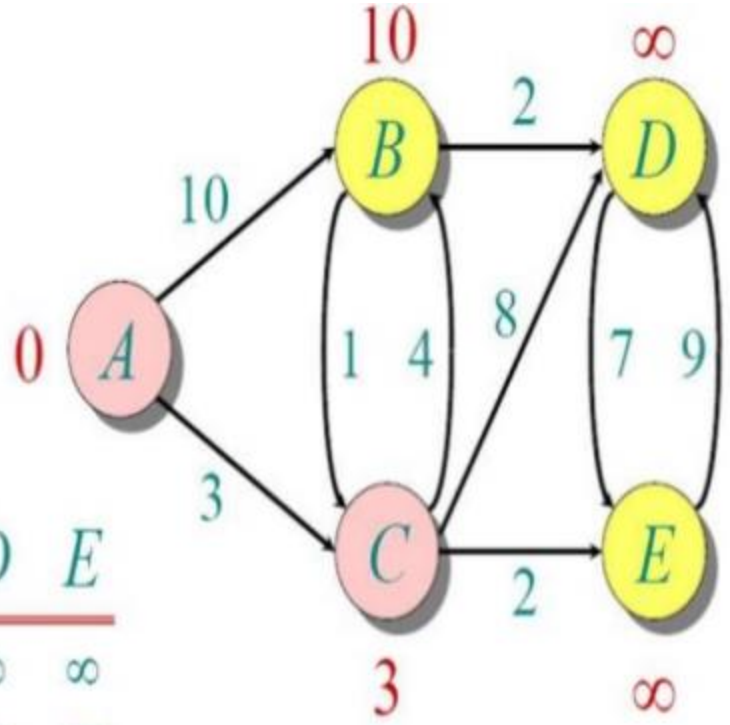



∞
∞
B
2
D
10
0
A
1
4
8
7
9
3
C
2
E
∞
∞
Q: A B C D E
0 ∞ ∞ ∞ ∞



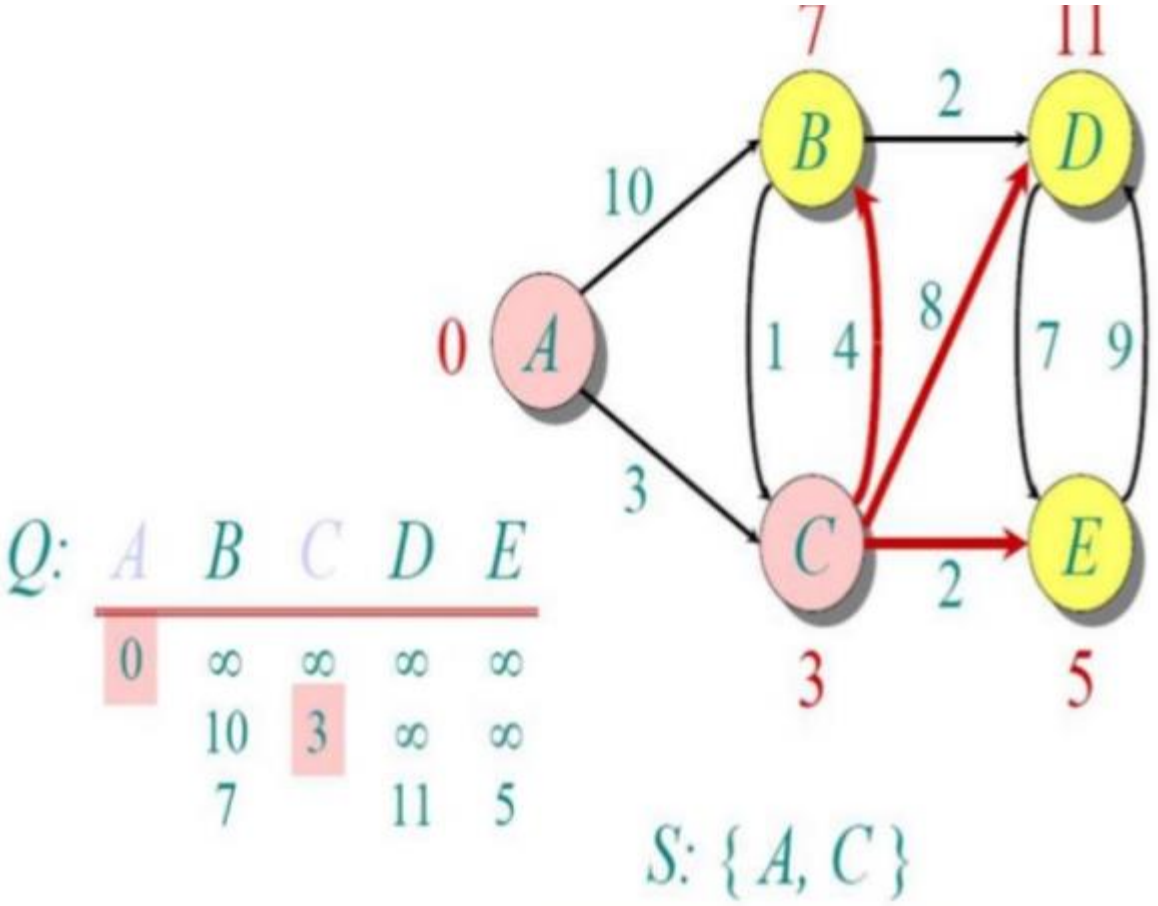
Q:	A	B	C	D	E
	0	∞	∞	∞	∞
		10	3	∞	∞

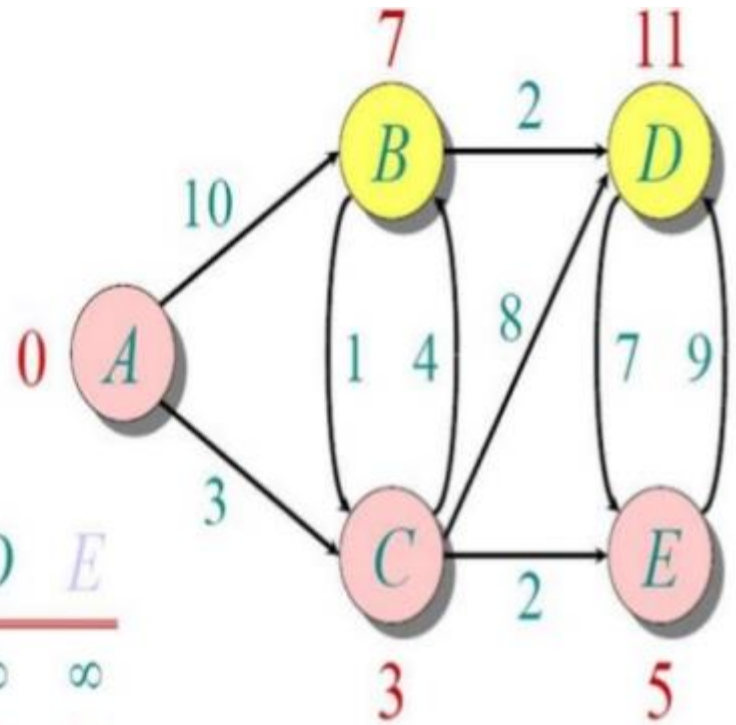
S: { A }



Q:	A	B	C	D	E
	0	∞	∞	∞	∞
		10	3	∞	∞

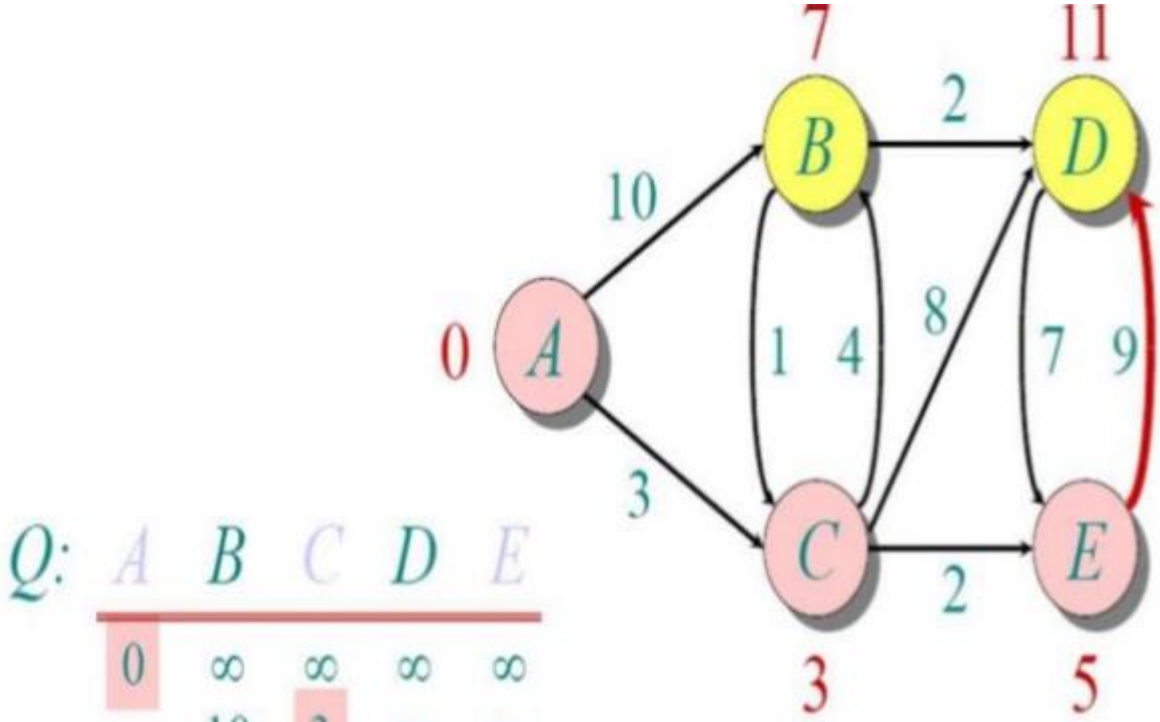
S: { A, C }

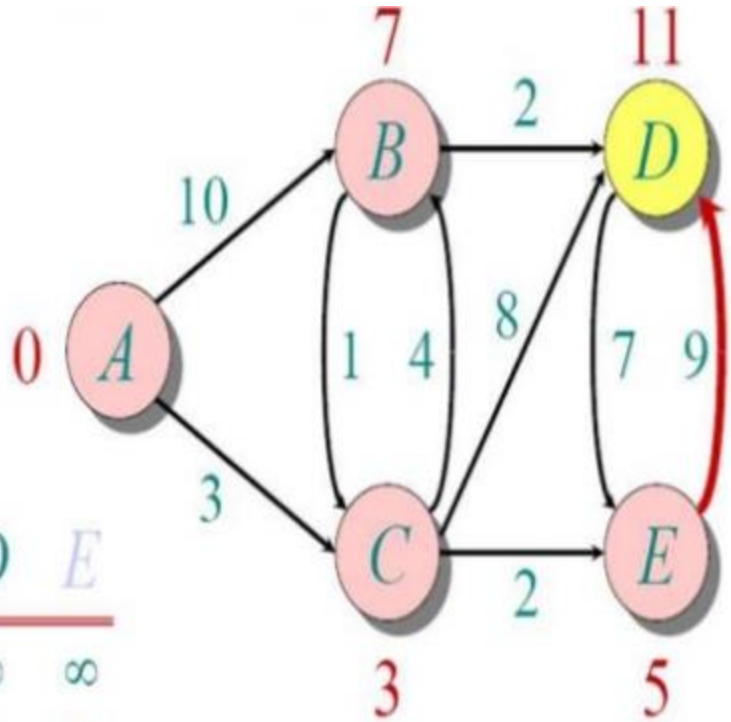

7
11
2
B
D
10
0
A
1
4
8
7
9
3
C
E
2
3
5
Q: A B C D E
0 ∞ ∞ ∞ ∞
10 3 ∞ ∞
7 11 5
S: {A, C}



Q:	A	B	C	D	E
	0	∞	∞	∞	∞
		10	3	∞	∞
		7		11	5

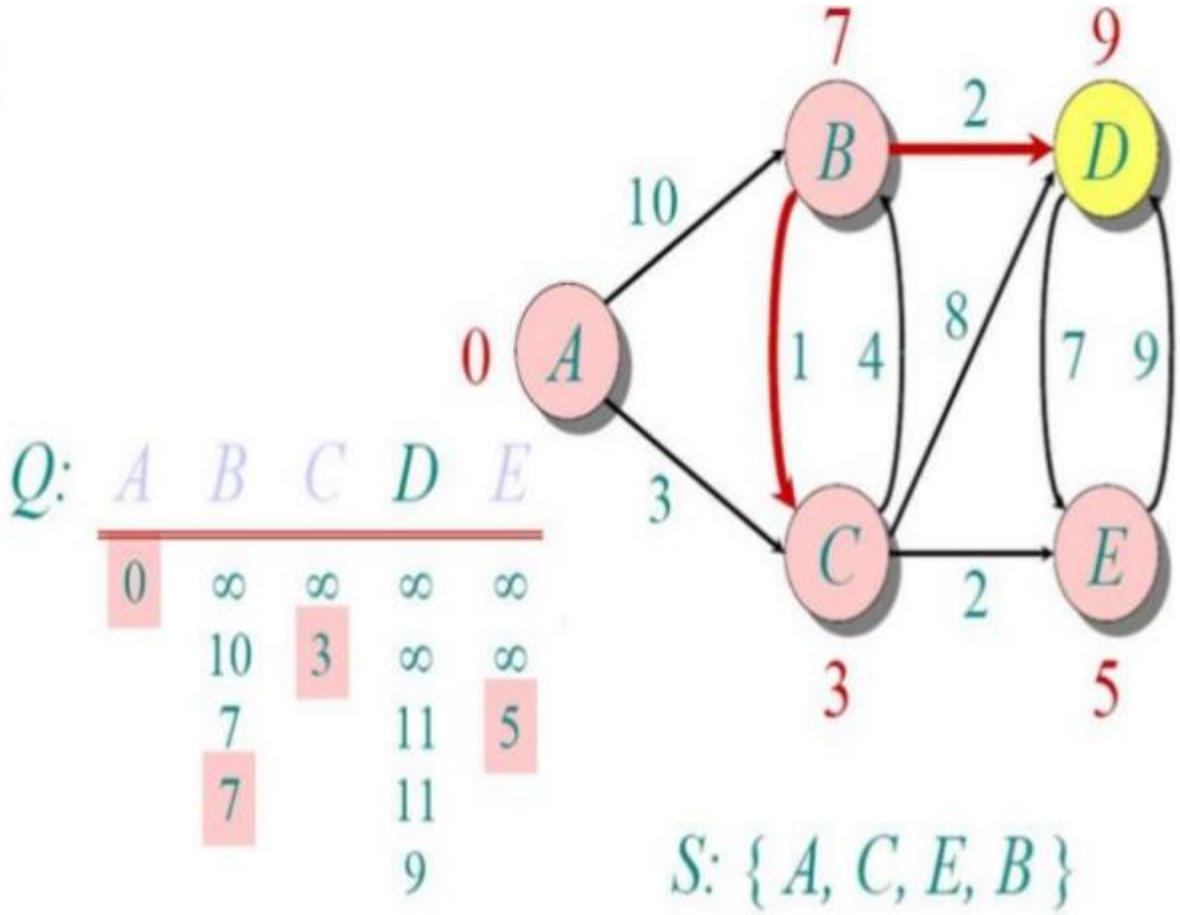
$S: \{A, C, E\}$

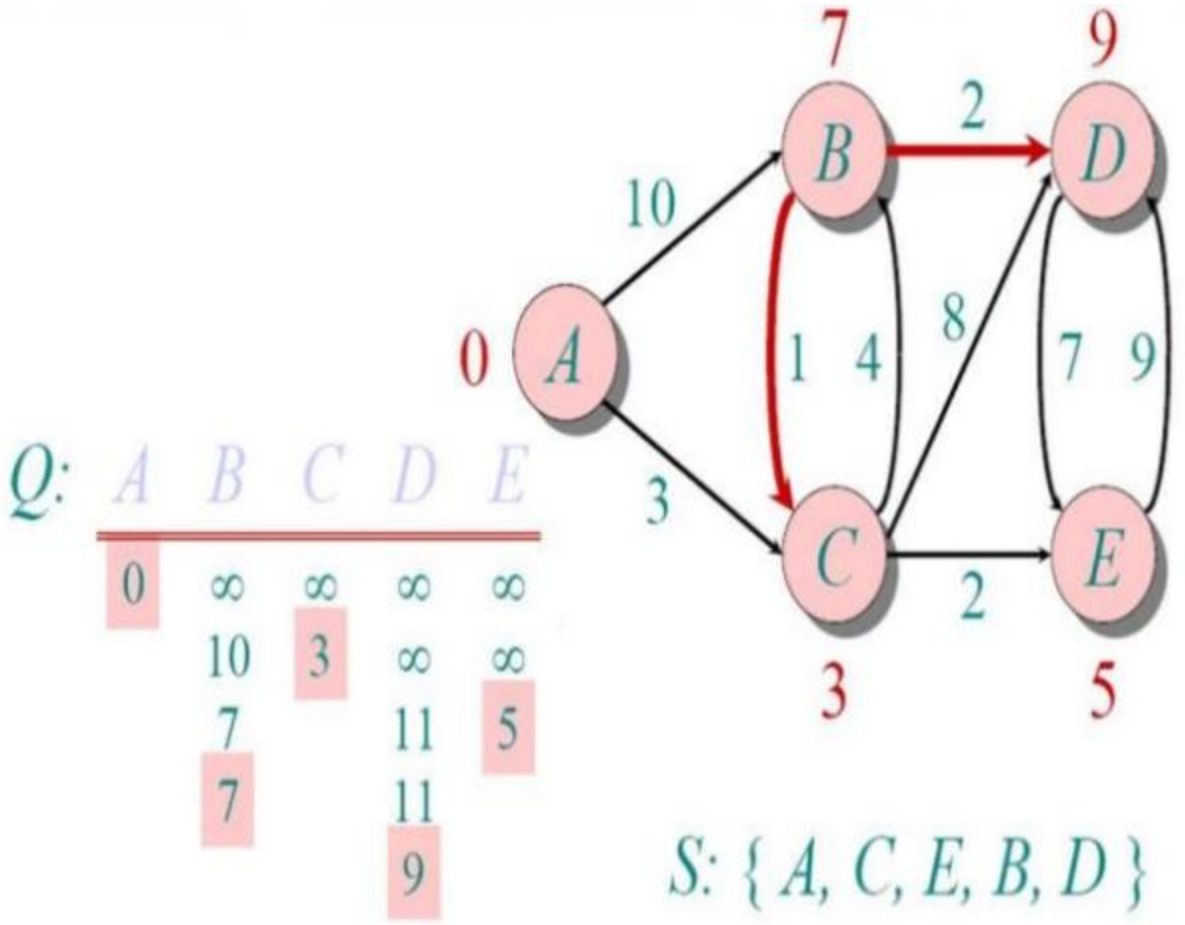

7
11
2
B
D
10
0
A
1
4
8
7
9
3
C
E
2
3
5
Q: A B C D E
0 ∞ ∞ ∞ ∞
10 3 ∞ ∞
7 11 5
7 11
S: { A, C, E }



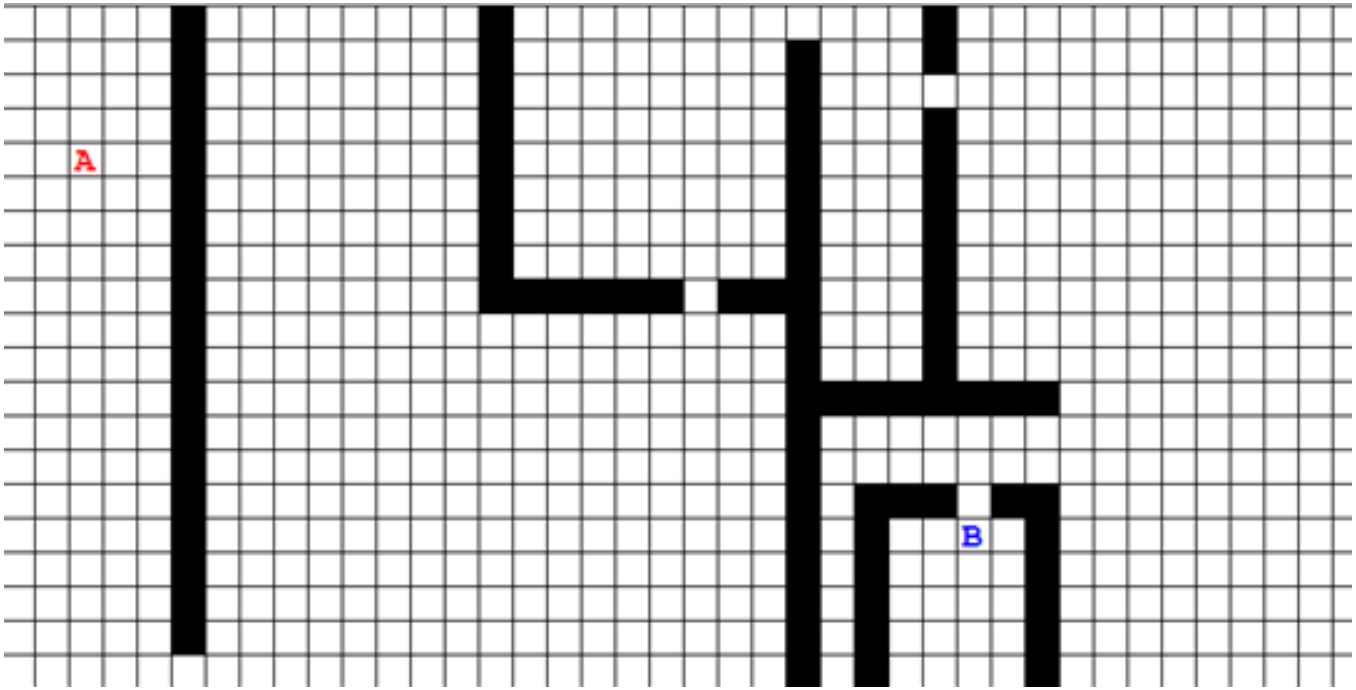
Q:	A	B	C	D	E
	0	∞	∞	∞	∞
		10	3	∞	∞
		7		11	5
		7		11	

$S: \{A, C, E, B\}$

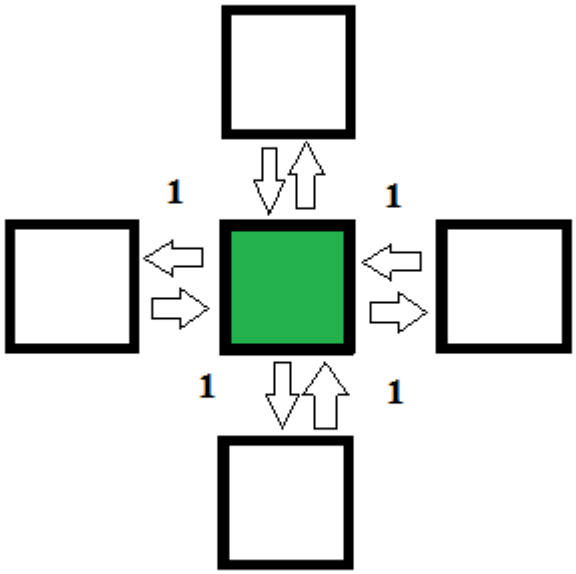

7
9
2
B
D
10
0
A
1
4
8
7
9
3
C
E
2
3
5
Q: A B C D E
0 ∞ ∞ ∞ ∞
10 3 ∞ ∞
7 11 5
7 11
9
S: { A, C, E, B }


7
9
2
B
D
10
0
A
1
4
8
7
9
3
C
E
2
3
5
Q: A B C D E
0 ∞ ∞ ∞ ∞
10 3 ∞ ∞
7 11 5
7 11
9
S: { A, C, E, B, D }

Finding the shortest path in a maze



Grid as a graph



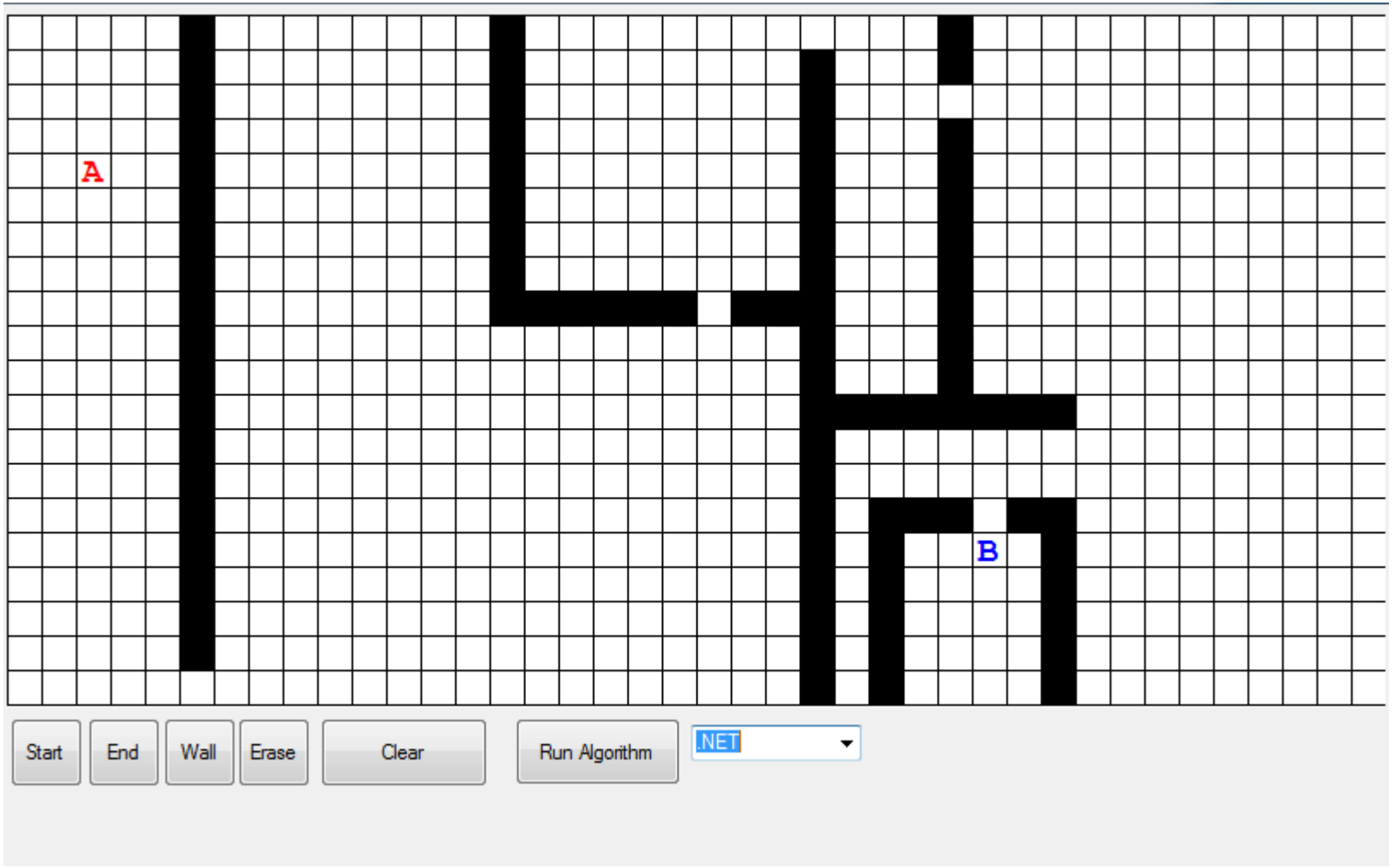
Pseudocode

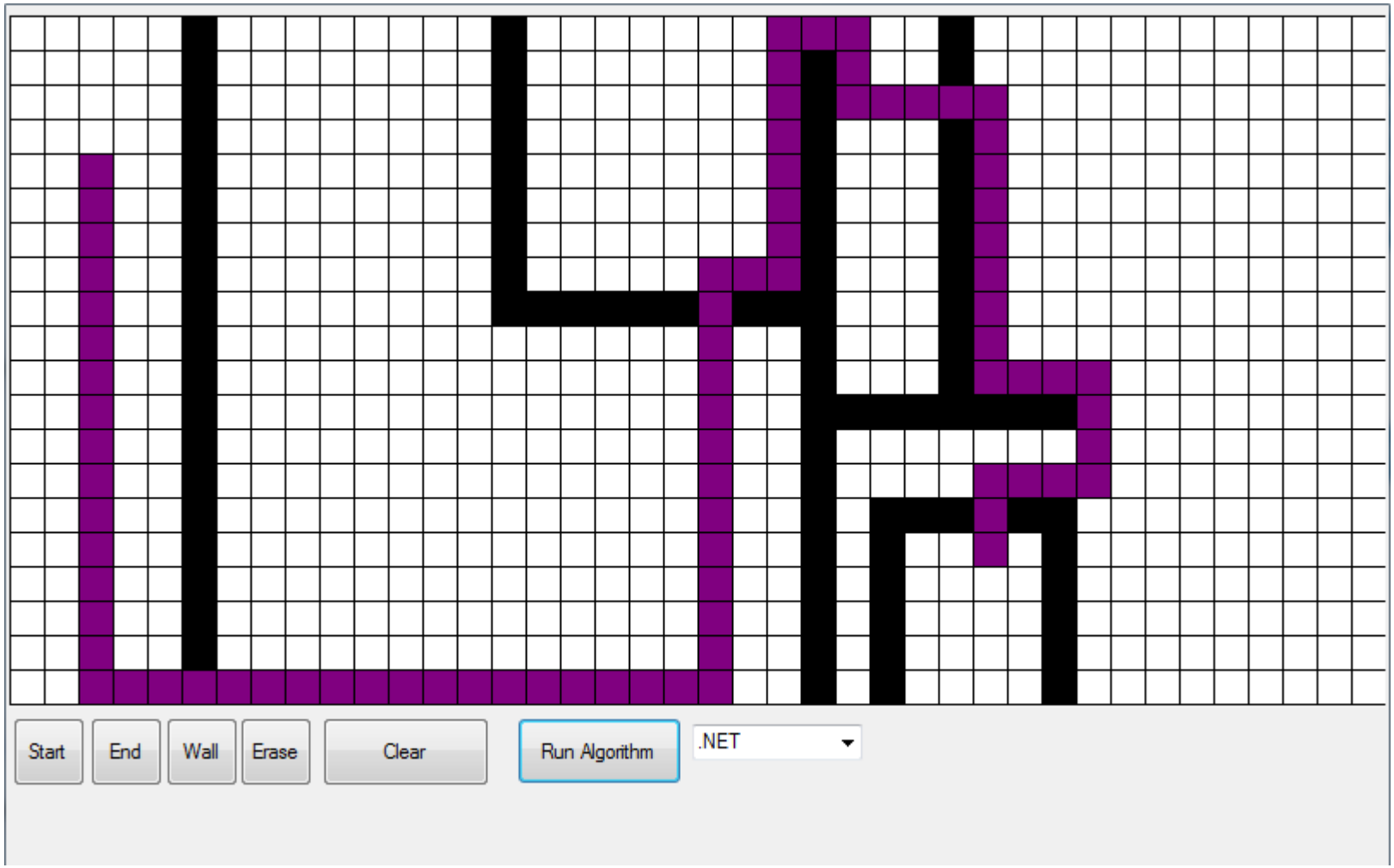
```
1:  function Dijkstra(Graph, source):
2:      for each vertex v in Graph:                 // Initialization
3:          dist[v] := infinity                     // initial distance from source to vertex v is set to infinite
4:          previous[v] := undefined                // Previous node in optimal path from source
5:      dist[source] := 0                           // Distance from source to source
6:      Q := the set of all nodes in Graph          // all nodes in the graph are unoptimized - thus are in Q
7:      while Q is not empty:                       // main loop
8:          u := node in Q with smallest dist[ ]
9:          remove u from Q
10:         for each neighbor v of u:               // where v has not yet been removed from Q.
11:             alt := dist[u] + dist_between(u, v)
12:             if alt < dist[v]                    // Relax (u,v)
13:                 dist[v] := alt
14:                 previous[v] := u
15:     return previous[ ]
```

input

0	0	1	0	0
2	0	1	0	0
0	0	1	3	0
0	0	1	1	0
0	0	0	0	0

GUI




Start
End
Wall
Erase
Clear
Run Algorithm
.NET