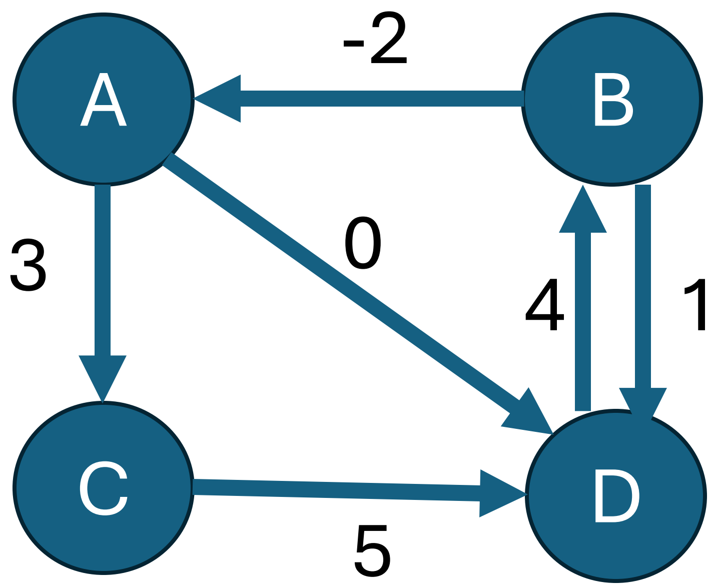
All Pairs Shortest Path

Floyd Warshall Algorithm

Floyd Warshall Algorithm

```
let dist be a |V| × |V| array of minimum distances initialized to ∞
for each edge (u, v)
    do dist[u][v] = w(u, v) // The weight of the edge (u, v)
for each vertex v do
    dist[v][v] = 0
for k from 1 to |V|
    for i from 1 to |V|
        for j from 1 to |V|
            if dist[i][j] > dist[i][k] + dist[k][j]
                dist[i][j] = dist[i][k] + dist[k][j]
            end if
```



to

from

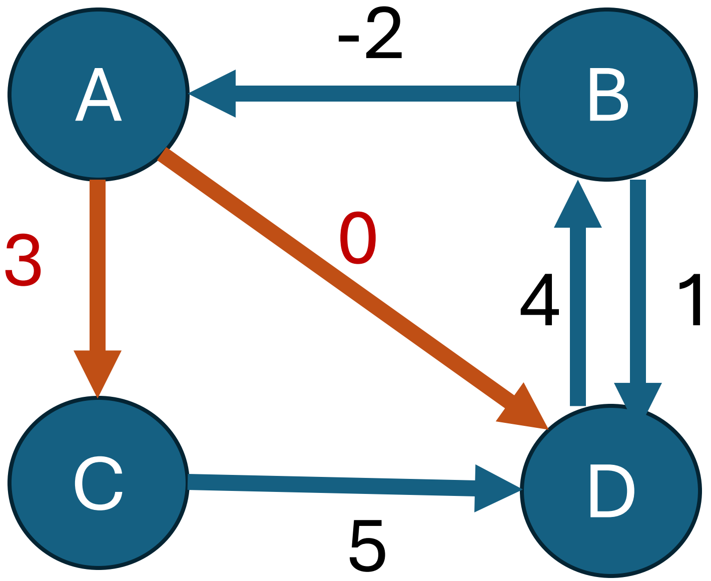
	A	B	C	D
A	∞	∞	∞	∞
B	∞	∞	∞	∞
C	∞	∞	∞	∞
D	∞	∞	∞	∞

to

from

	A	B	C	D
A				
B				
C				
D				

```
let dist be a |V| × |V| array of minimum distances initialized to ∞
for each edge ($u$, $v$)
    do dist[$u$][$v$] = w($u$, $v$) // The weight of the edge (u, v)
for each vertex $v$ do
    dist[$v$][$v$] = 0
for $k$ from 1 to |V|
    for $i$ from 1 to |V|
        for $j$ from 1 to |V|
            if dist[$i$][$j$] > dist[$i$][$k$] + dist[$k$][$j$]
                dist[$i$][$j$] = dist[$i$][$k$] + dist[$k$][$j$]
            end if
```



to

from

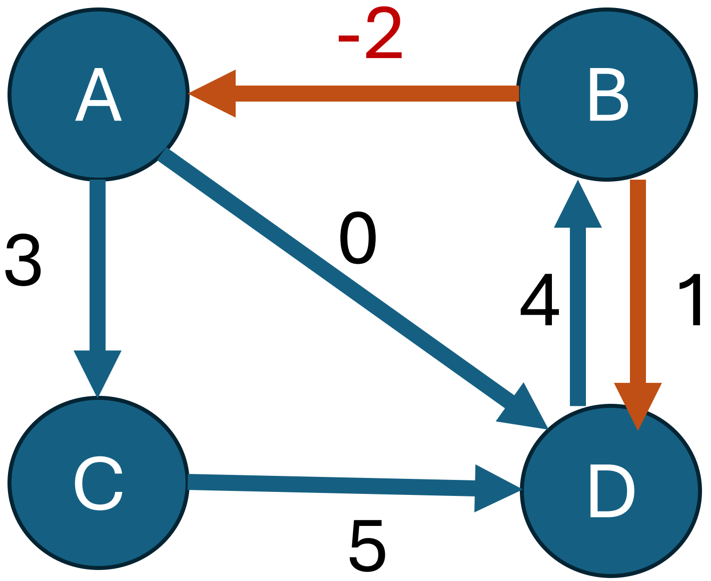
	A	B	C	D
A	∞	∞	3	0
B	∞	∞	∞	∞
C	∞	∞	∞	∞
D	∞	∞	∞	∞

to

from

	A	B	C	D
A			A	A
B				
C				
D				

```
let dist be a |V| × |V| array of minimum distances initialized to ∞
for each edge (u, v)
    do dist[u][v] = w(u, v) // The weight of the edge (u, v)
for each vertex v do
    dist[v][v] = 0
for k from 1 to |V|
    for i from 1 to |V|
        for j from 1 to |V|
            if dist[i][j] > dist[i][k] + dist[k][j]
                dist[i][j] = dist[i][k] + dist[k][j]
            end if
```



to

from

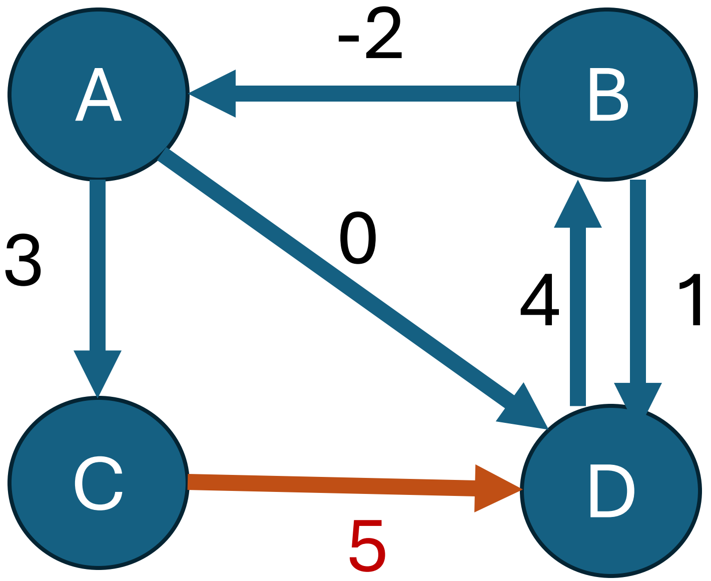
	A	B	C	D
A	∞	∞	3	0
B	-2	∞	∞	1
C	∞	∞	∞	∞
D	∞	∞	∞	∞

to

from

	A	B	C	D
A			A	A
B	B			B
C				
D				

```
let dist be a |V| × |V| array of minimum distances initialized to ∞
for each edge (u, v)
    do dist[u][v] = w(u, v) // The weight of the edge (u, v)
for each vertex v do
    dist[v][v] = 0
for k from 1 to |V|
    for i from 1 to |V|
        for j from 1 to |V|
            if dist[i][j] > dist[i][k] + dist[k][j]
                dist[i][j] = dist[i][k] + dist[k][j]
            end if
```



to

from

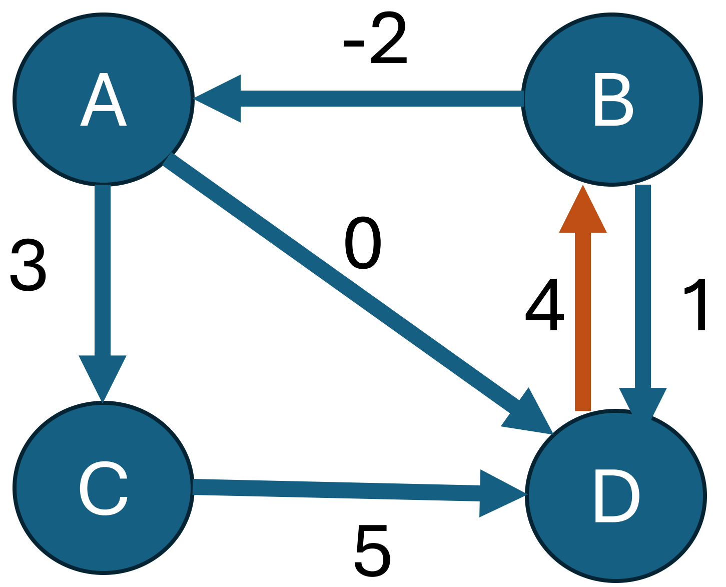
	A	B	C	D
A	∞	∞	3	0
B	-2	∞	∞	1
C	∞	∞	∞	5
D	∞	∞	∞	∞

to

from

	A	B	C	D
A			A	A
B	B			B
C				C
D				

```
let dist be a |V| × |V| array of minimum distances initialized to ∞
for each edge (u, v)
    do dist[u][v] = w(u, v) // The weight of the edge (u, v)
for each vertex v do
    dist[v][v] = 0
for k from 1 to |V|
    for i from 1 to |V|
        for j from 1 to |V|
            if dist[i][j] > dist[i][k] + dist[k][j]
                dist[i][j] = dist[i][k] + dist[k][j]
            end if
```



to

from

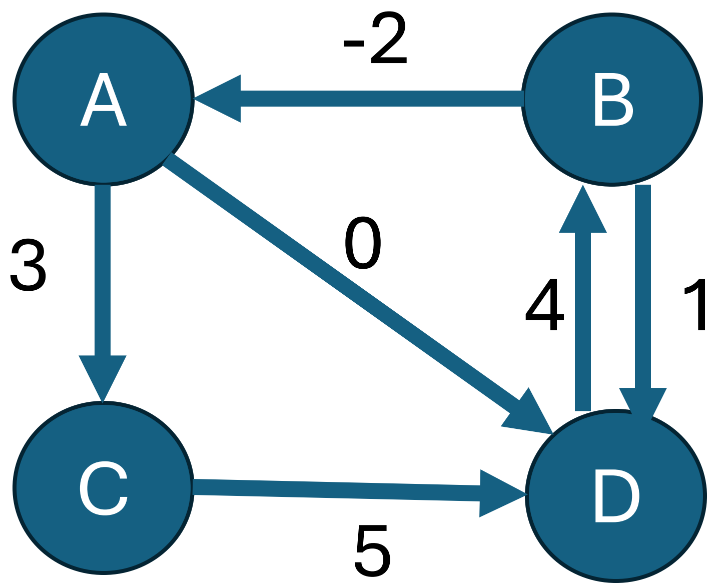
	A	B	C	D
A	∞	∞	3	0
B	-2	∞	∞	1
C	∞	∞	∞	5
D	∞	4	∞	∞

to

from

	A	B	C	D
A			A	A
B	B			B
C				C
D		D		

```
let dist be a |V| × |V| array of minimum distances initialized to ∞
for each edge (u, v)
    do dist[u][v] = w(u, v) // The weight of the edge (u, v)
for each vertex v do
    dist[v][v] = 0
for k from 1 to |V|
    for i from 1 to |V|
        for j from 1 to |V|
            if dist[i][j] > dist[i][k] + dist[k][j]
                dist[i][j] = dist[i][k] + dist[k][j]
            end if
```



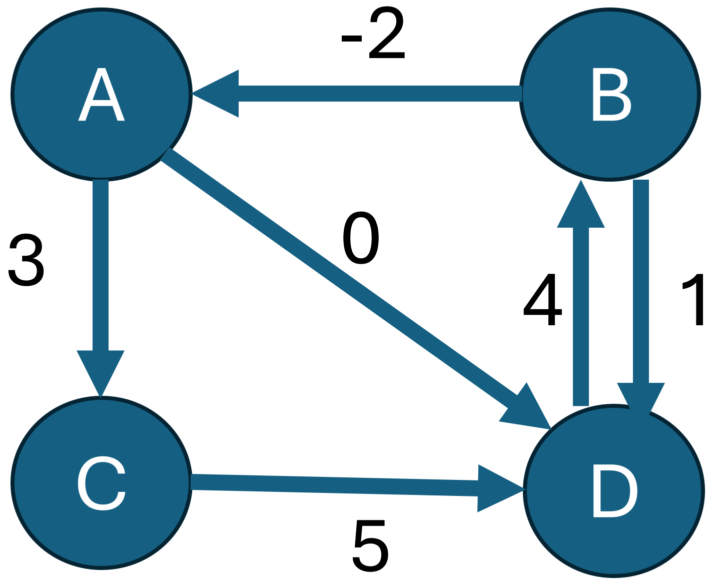
to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	∞	1
C	∞	∞	0	5
D	∞	4	∞	0

to

from	A	B	C	D
A			A	A
B	B			B
C				C
D		D		

```
let dist be a |V| × |V| array of minimum distances initialized to ∞
for each edge (u, v)
    do dist[u][v] = w(u, v) // The weight of the edge (u, v)
for each vertex v do
    dist[v][v] = 0
for k from 1 to |V|
    for i from 1 to |V|
        for j from 1 to |V|
            if dist[i][j] > dist[i][k] + dist[k][j]
                dist[i][j] = dist[i][k] + dist[k][j]
            end if
```



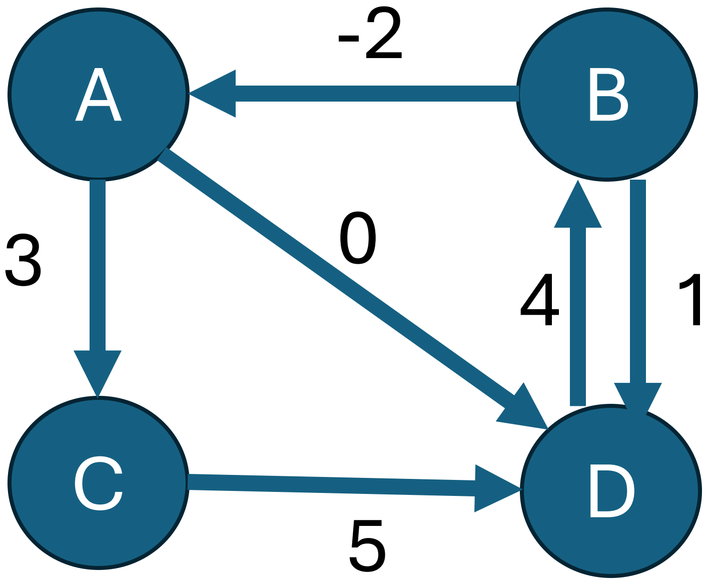
to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	∞	1
C	∞	∞	0	5
D	∞	4	∞	0

to

from	A	B	C	D
A			A	A
B	B			B
C				C
D		D		

```
let dist be a |V| × |V| array of minimum distances initialized to ∞
for each edge (u, v)
    do dist[u][v] = w(u, v) // The weight of the edge (u, v)
for each vertex v do
    dist[v][v] = 0
for k from 1 to |V|
    for i from 1 to |V|
        for j from 1 to |V|
            if dist[i][j] > dist[i][k] + dist[k][j]
                dist[i][j] = dist[i][k] + dist[k][j]
            end if
```



to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	∞	1
C	∞	∞	0	5
D	∞	4	∞	0

to

from

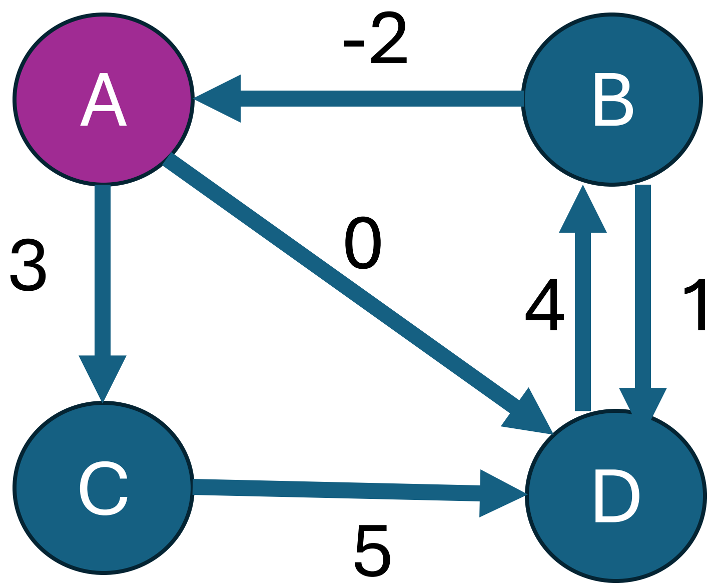
	A	B	C	D
A			A	A
B	B			B
C				C
D		D		

```
for k from 1 to |V|
    for i from 1 to |V|
        for j from 1 to |V|
            if dist[i][j] > dist[i][k] + dist[k][j]
                dist[i][j] = dist[i][k] + dist[k][j]
            end if
```

k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4



to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	∞	1
C	∞	∞	0	5
D	∞	4	∞	0

to

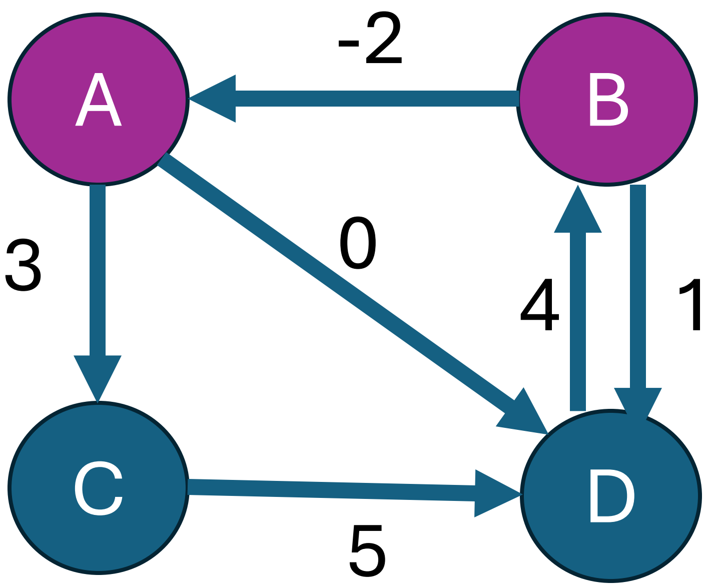
from

	A	B	C	D
A			A	A
B	B			B
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1   2   3   4
i = 1   2   3   4
j = 1   2   3   4

```
       AA             AA              AA
if dist[1][1] > dist[1][1] + dist[1][1]
     dist[1][1] = dist[1][1] + dist[1][1]
         AA            AA             AA
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	∞	1
C	∞	∞	0	5
D	∞	4	∞	0

to

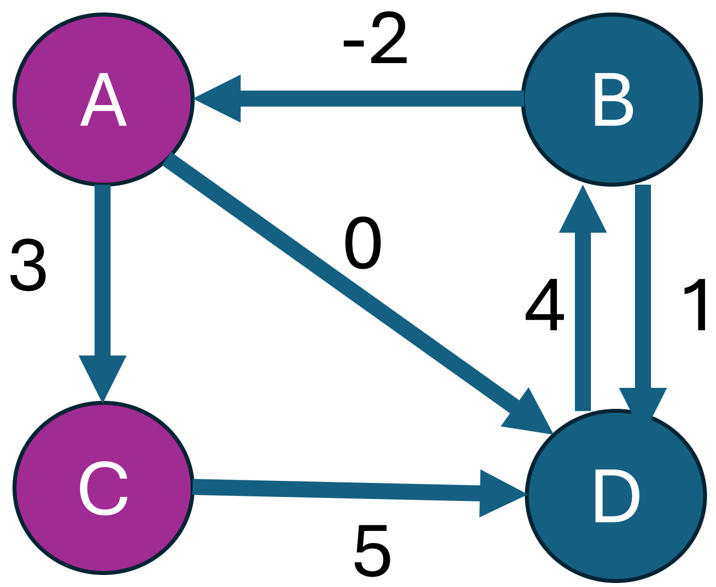
from	A	B	C	D
A			A	A
B	B			B
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

AB = ∞ > AA = 0 + AB = ∞

```
if dist[1][2] > dist[1][1] + dist[1][2]
    dist[1][2] = dist[1][1] + dist[1][2]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	∞	1
C	∞	∞	0	5
D	∞	4	∞	0

to

from	A	B	C	D
A			A	A
B	B			B
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

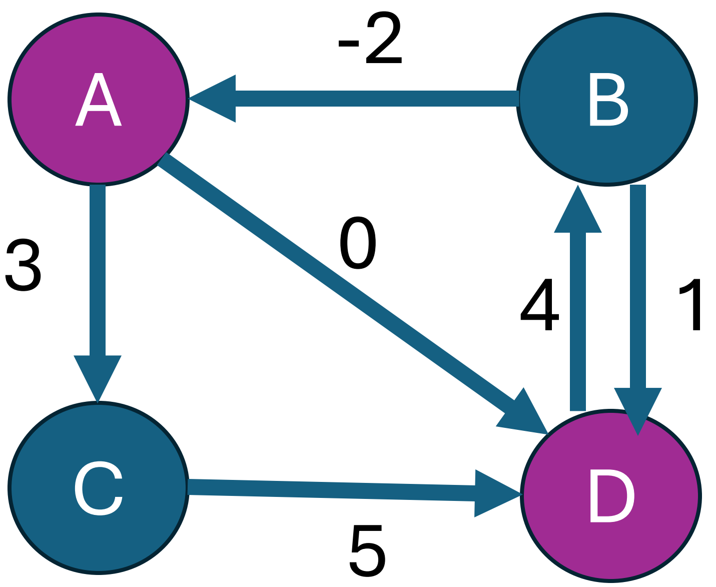
k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4

AC = 3 > AA = 0 + AC = 3

```
if dist[1][3] > dist[1][1] + dist[1][3]
    dist[1][3] = dist[1][1] + dist[1][3]
```



to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	∞	1
C	∞	∞	0	5
D	∞	4	∞	0

to

from

	A	B	C	D
A			A	A
B	B			B
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

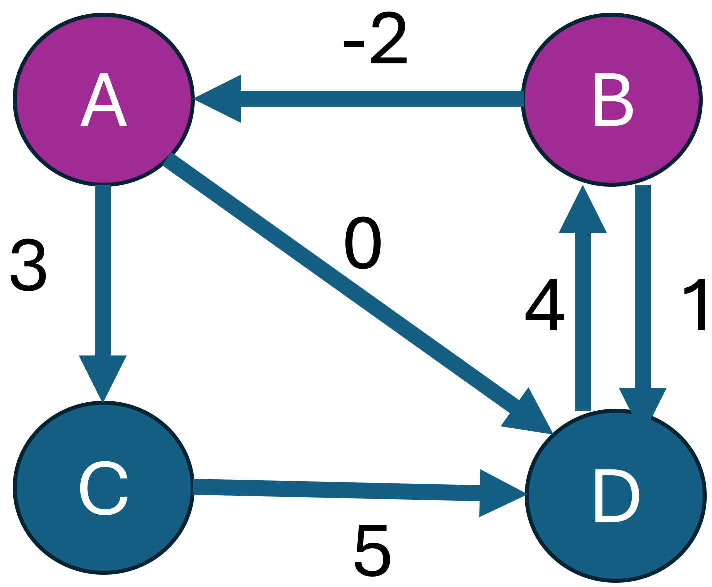
k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4

AD = 0 > AA = 0 + AD = 0

```
if dist[1][4] > dist[1][1] + dist[1][4]
    dist[1][4] = dist[1][1] + dist[1][4]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	∞	1
C	∞	∞	0	5
D	∞	4	∞	0

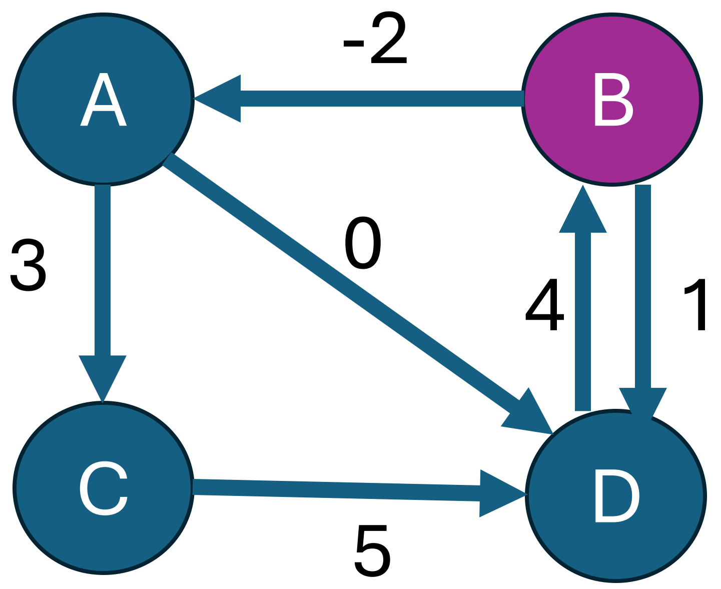
to

from	A	B	C	D
A			A	A
B	B			B
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
     dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1   2   3   4
i = 1   2   3   4
j = 1   2   3   4

```
        BA = -2      >      BA = -2    +      AA = 0
if dist[2][1] > dist[2][1] + dist[1][1]
     dist[2][1] = dist[2][1] + dist[1][1]
```



to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	∞	1
C	∞	∞	0	5
D	∞	4	∞	0

to

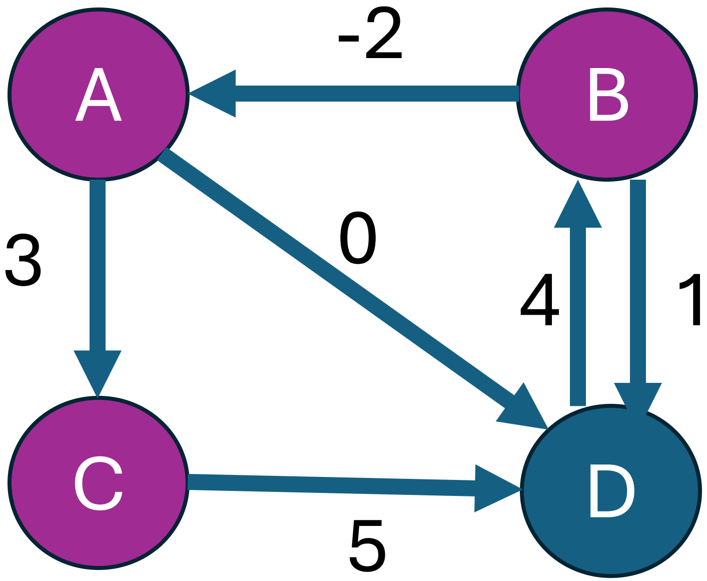
from

	A	B	C	D
A			A	A
B	B			B
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1   2   3   4
i = 1   2   3   4
j = 1   2   3   4

```
      BB = 0        >      BA = -2    +      AB = ∞
if dist[2][2] > dist[2][1] + dist[1][2]
    dist[2][2] = dist[2][1] + dist[1][2]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	1
C	∞	∞	0	5
D	∞	4	∞	0

to

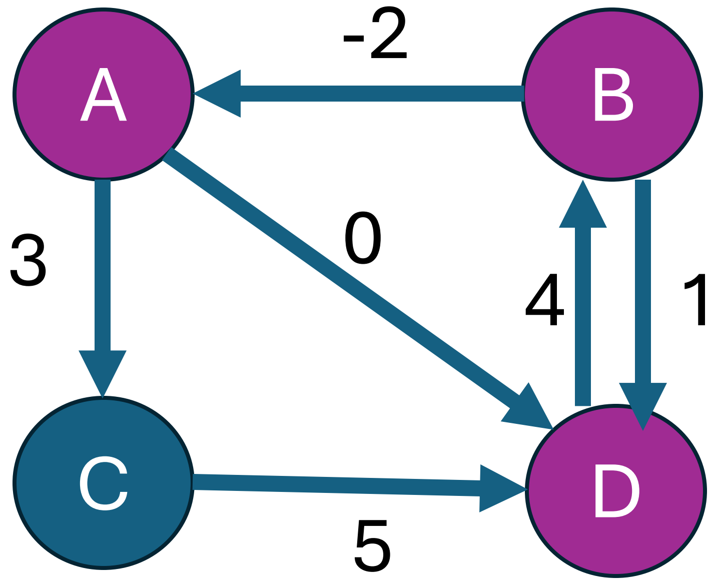
from	A	B	C	D
A			A	A
B	B		A	B
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k =	1	2	3	4
i =	1	2	3	4
j =	1	2	3	4

BC = ∞ > BA = -2 + AC = 3

```
if dist[2][3] > dist[2][1] + dist[1][3]
    dist[2][3] = dist[2][1] + dist[1][3]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

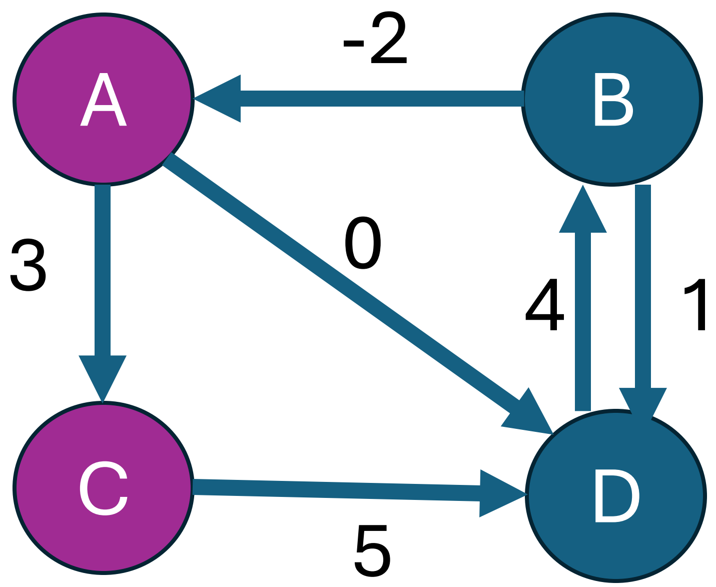
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

```
k = 1  2  3  4
i = 1  2  3  4
j = 1  2  3  4
```

YES!!!!!!

```
     BD = 1       >    BA = -2    +    AD = 0
if dist[2][4] > dist[2][1] + dist[1][4]
   dist[2][4] = dist[2][1] + dist[1][4]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

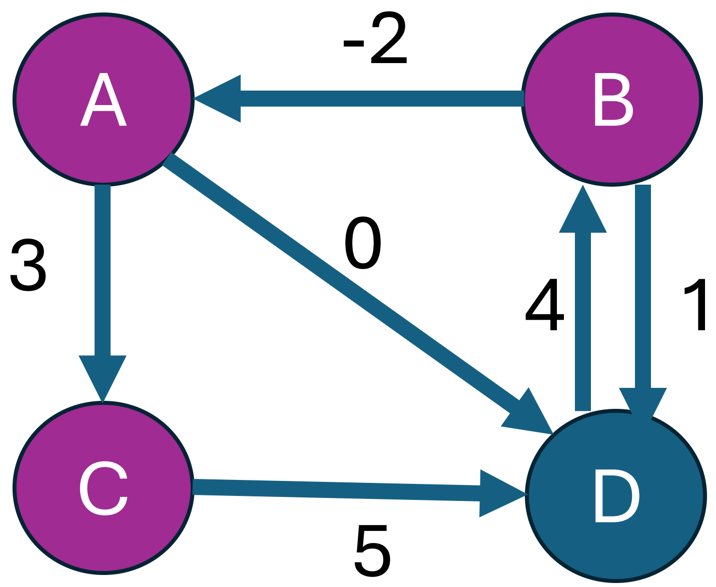
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1   2   3   4
i = 1   2   3   4
j = 1   2   3   4

CA = ∞   >   CA = ∞   +   AA = 0

```
if dist[3][1] > dist[3][1] + dist[1][1]
    dist[3][1] = dist[3][1] + dist[1][1]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

from	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

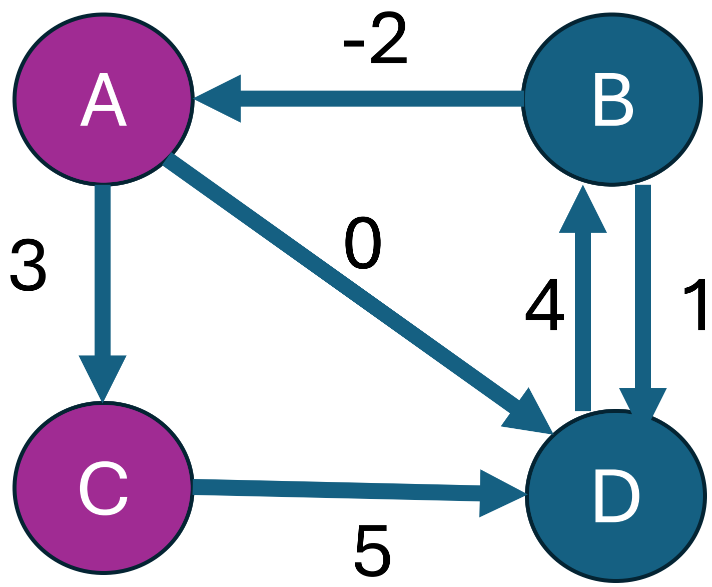
k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4

CB = ∞ > CA = ∞ + AB = 0

```
if dist[3][2] > dist[3][1] + dist[1][2]
    dist[3][2] = dist[3][1] + dist[1][3]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

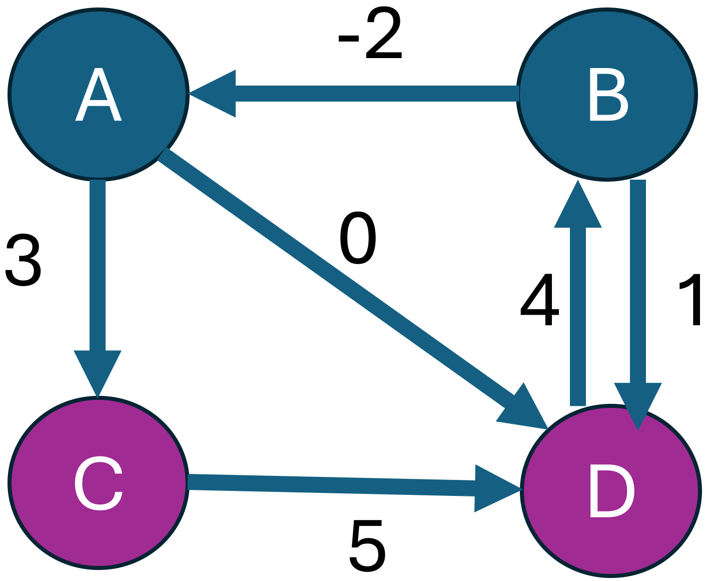
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

CC = 0 > CA = ∞ + AC = 3

```
if dist[3][3] > dist[3][1] + dist[1][3]
    dist[3][3] = dist[3][1] + dist[1][3]
```



```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

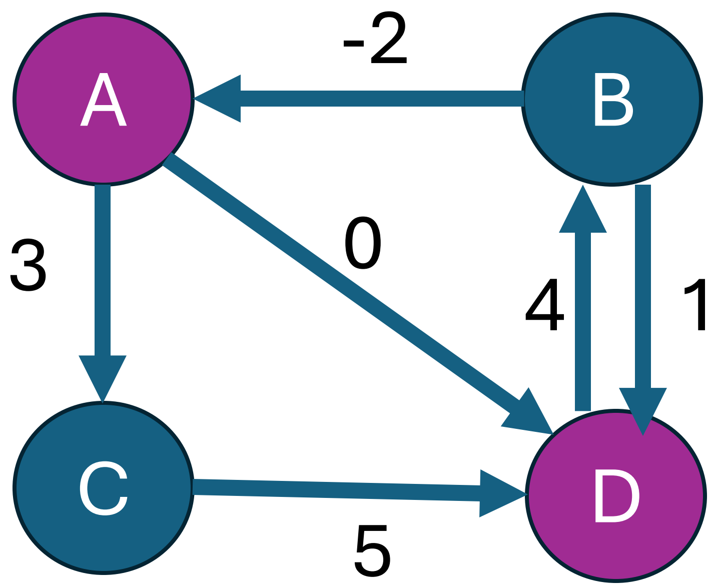
		to			
		A	B	C	D
from	A	0	∞	3	0
	B	-2	0	1	-2
	C	∞	∞	0	5
	D	∞	4	∞	0

		to			
		A	B	C	D
from	A			A	A
	B	B		A	A
	C				C
	D		D		

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

CD = 5 > CA = ∞ + AD = 0

```
if dist[3][4] > dist[3][1] + dist[1][4]
    dist[3][4] = dist[3][1] + dist[1][4]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

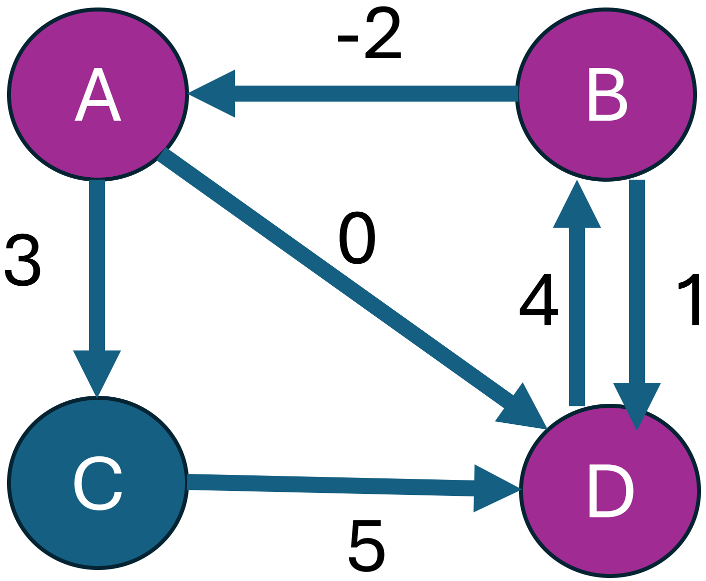
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1   2   3   4
i = 1   2   3   4
j = 1   2   3   4

DA = ∞   >   DA = ∞   +   AA = 0

```
if dist[4][1] > dist[4][1] + dist[1][1]
    dist[4][1] = dist[3][1] + dist[1][1]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

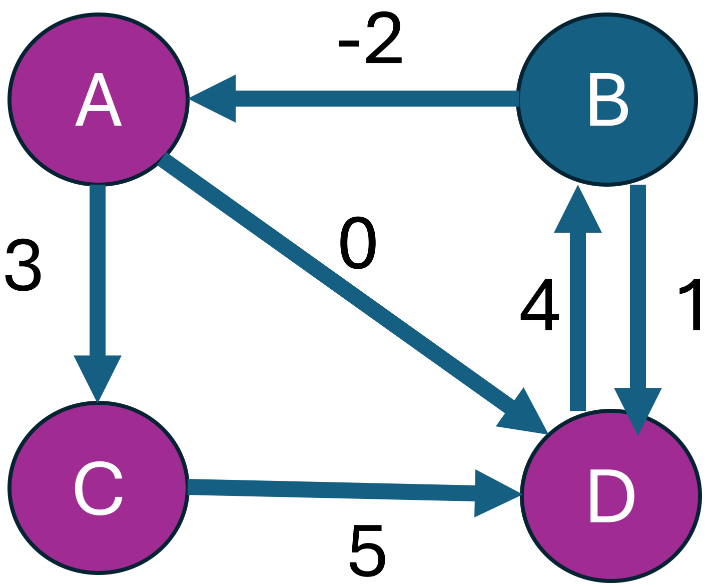
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

DB = 4 > DA = ∞ + AB = ∞

```
if dist[4][2] > dist[4][1] + dist[1][2]
    dist[4][2] = dist[3][1] + dist[1][2]
```



to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

from

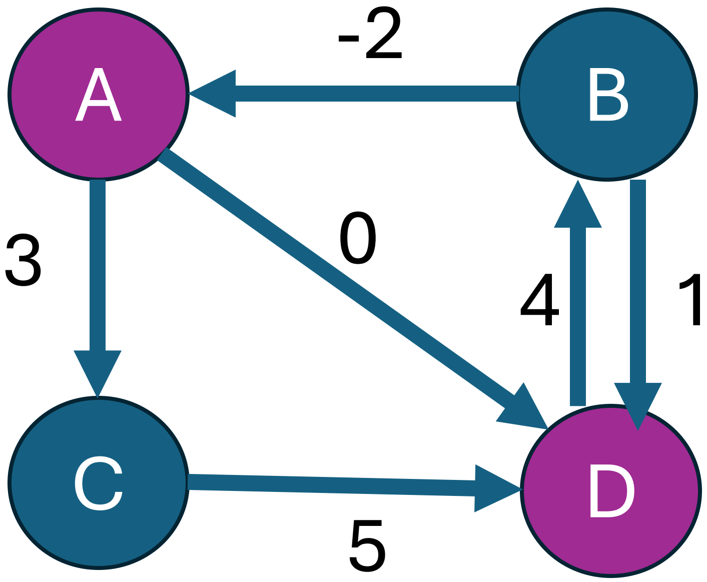
	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

DC = ∞ > DA = ∞ + AC = 3

```
if dist[4][3] > dist[4][1] + dist[1][3]
    dist[4][3] = dist[3][1] + dist[1][3]
```



to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

from

	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

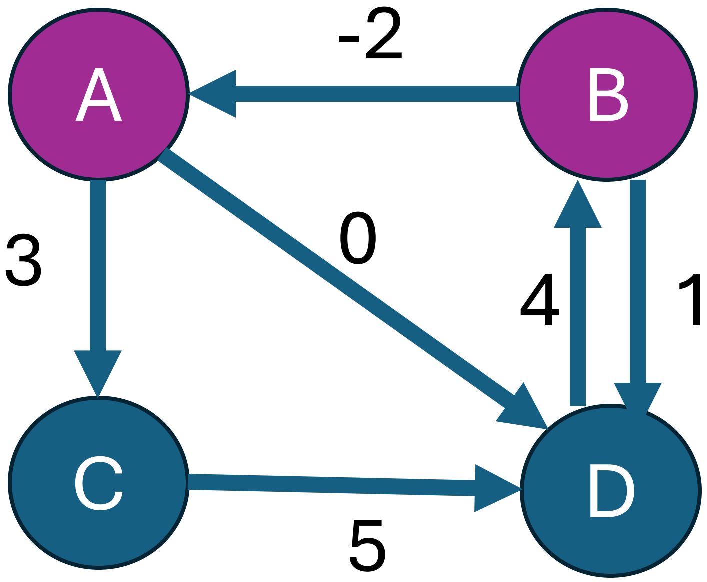
k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4

DD = 0 > DA = ∞ + AD = 0

```
if dist[4][4] > dist[4][1] + dist[1][4]
    dist[4][4] = dist[4][1] + dist[1][4]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

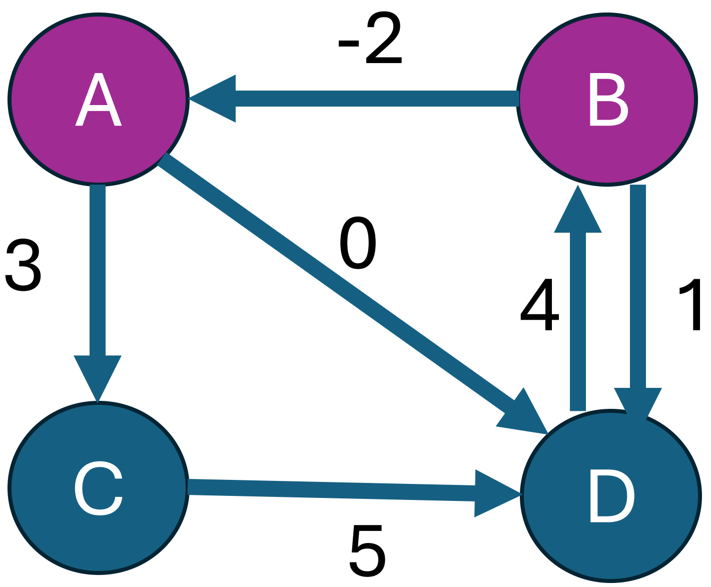
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

AA = 0 > AB = ∞ + BA = -2

```
if dist[1][1] > dist[1][2] + dist[2][1]
    dist[1][1] = dist[1][2] + dist[2][1]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

from	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

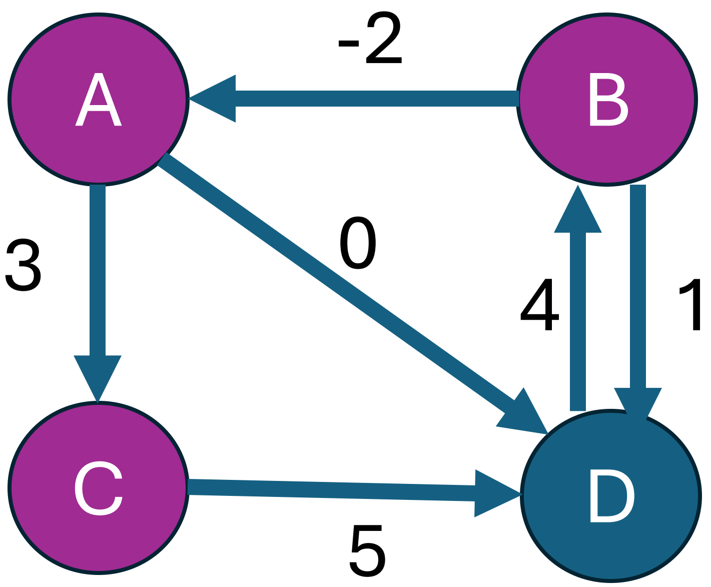
k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4

AB = ∞ > AB = ∞ + BB = 0

```
if dist[1][2] > dist[1][2] + dist[2][2]
    dist[1][2] = dist[1][2] + dist[2][2]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

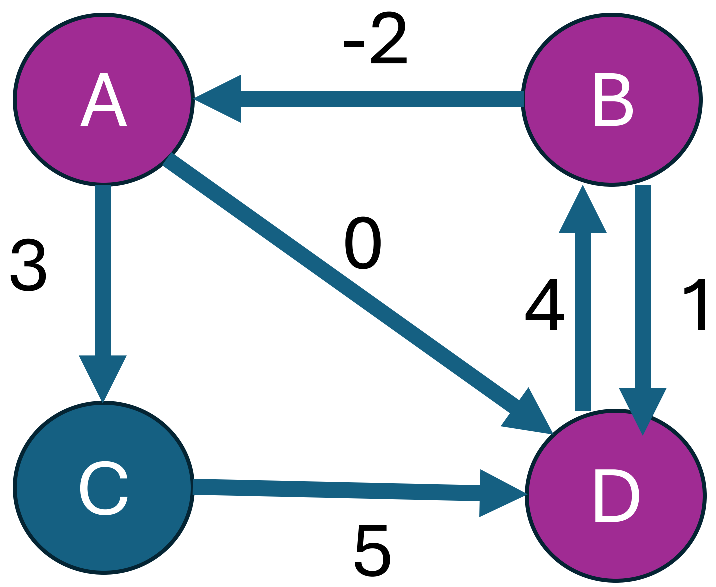
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

AC = 3 > AB = ∞ + BC = 1

```
if dist[1][3] > dist[1][2] + dist[2][3]
    dist[1][3] = dist[1][2] + dist[2][3]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

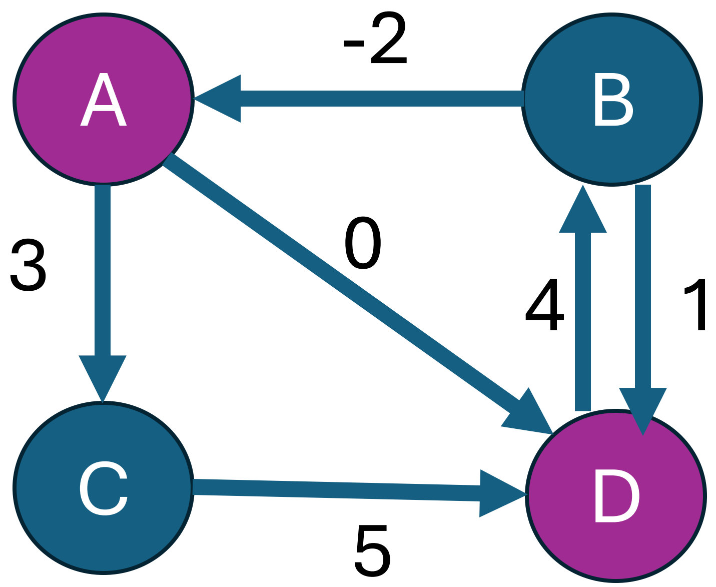
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

AD = 0 > AB = ∞ + BD = -2

```
if dist[1][4] > dist[1][2] + dist[2][4]
    dist[1][4] = dist[1][2] + dist[2][4]
```



to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

from

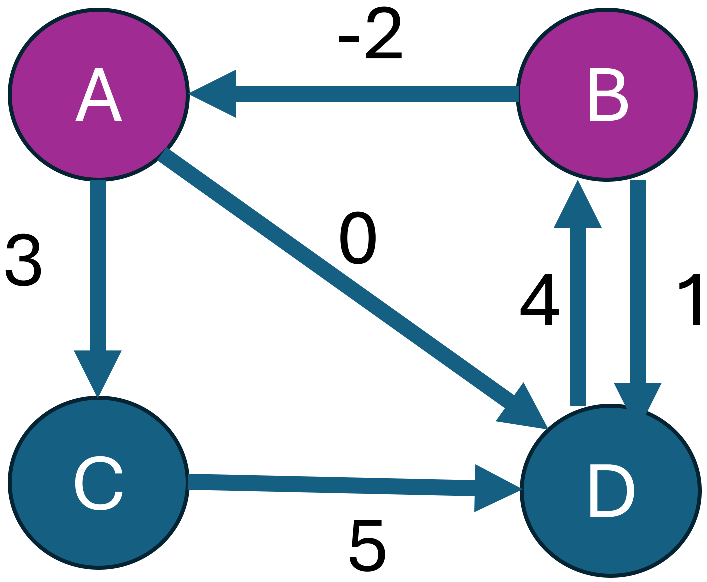
	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

AD = 0 > AB = ∞ + BD = -2

```
if dist[1][4] > dist[1][2] + dist[2][4]
    dist[1][4] = dist[1][2] + dist[2][4]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

from	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

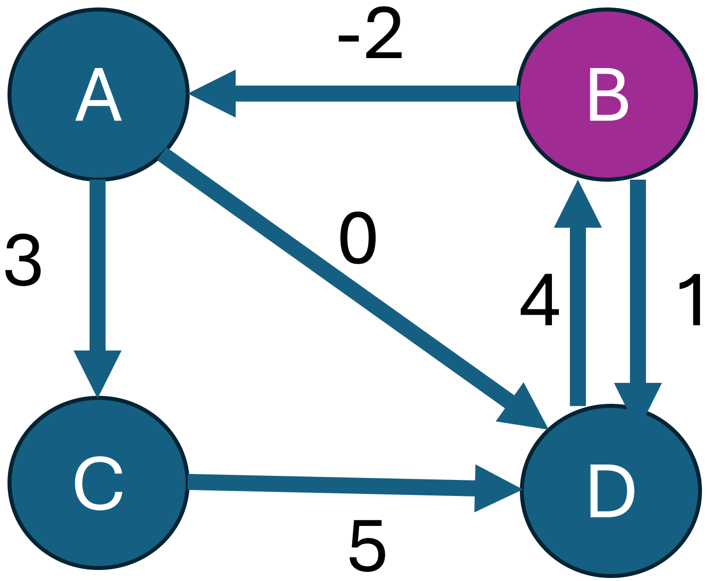
```
if dist[i][j] > dist[i][k] + dist[k][j]
   dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4

```
       BA = -2       >      BB = 0      +      BA = -2
if dist[2][1] > dist[2][2] + dist[2][1]
   dist[2][1] = dist[2][2] + dist[2][2]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

from	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

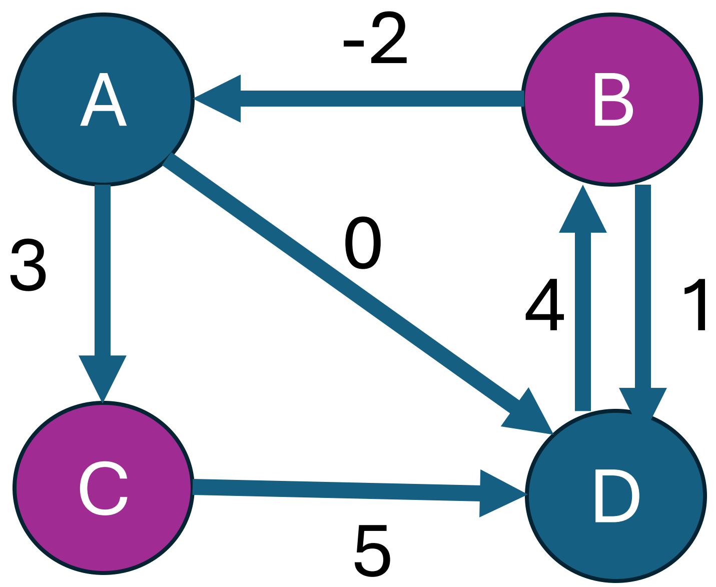
```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4

```
     BB = 0          >      BB = 0      +     BB=0
if dist[2][2] > dist[2][2] + dist[2][2]
    dist[2][2] = dist[2][2] + dist[2][2]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

from	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

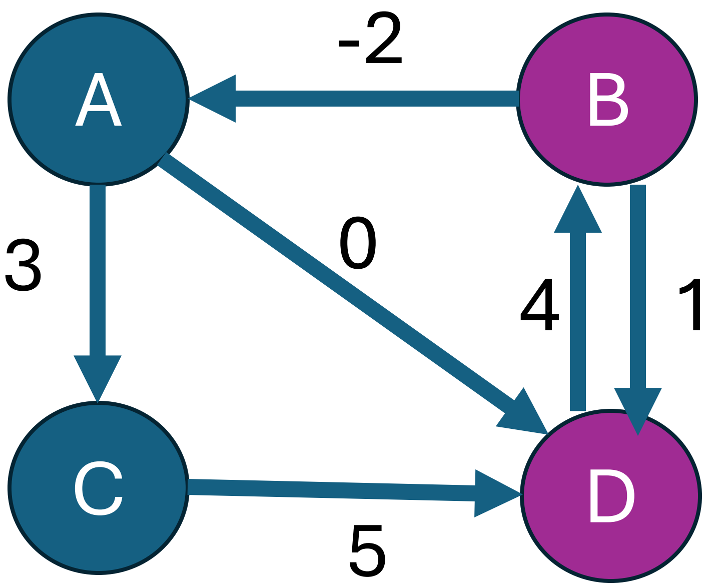
k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4

BC = 1 > BB = 0 + BC=1

```
if dist[2][3] > dist[2][2] + dist[2][3]
    dist[2][3] = dist[2][2] + dist[2][3]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

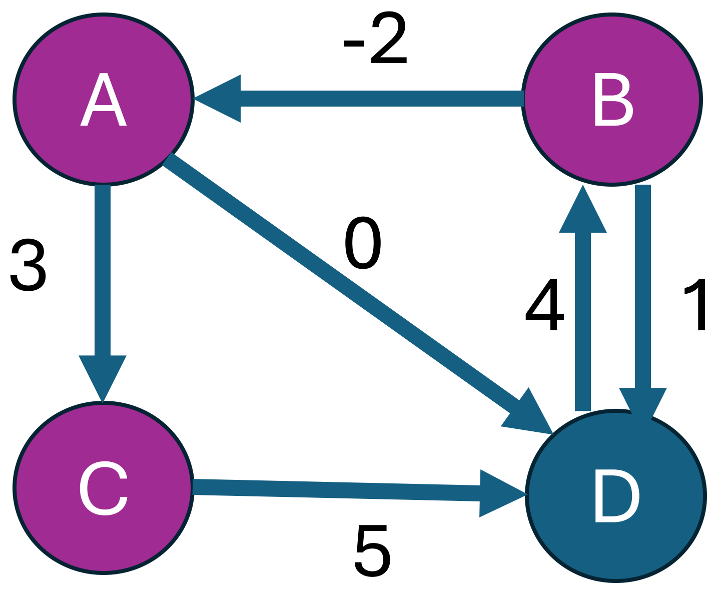
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

BD = -2 > BB = 0 + BD=-2

```
if dist[2][4] > dist[2][2] + dist[2][4]
    dist[2][4] = dist[2][2] + dist[2][4]
```



```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

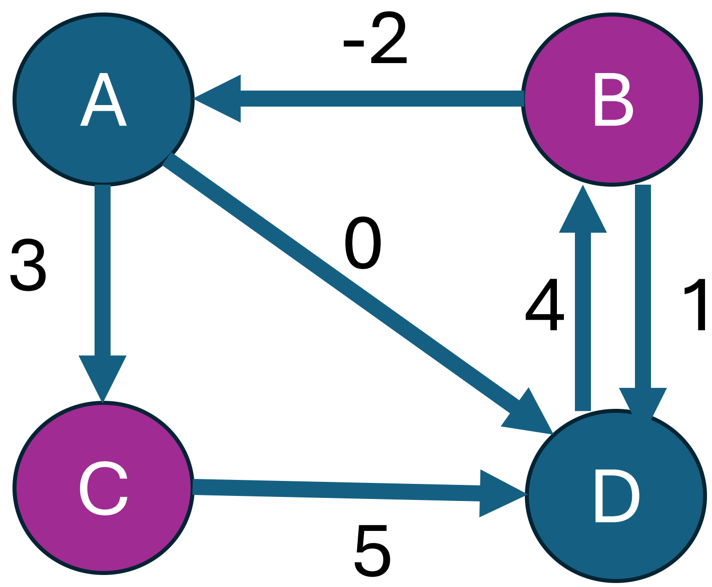
to

from	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

CA = ∞ > CB = ∞ + BA=-2

```
if dist[3][1] > dist[3][2] + dist[2][1]
    dist[3][1] = dist[3][2] + dist[2][1]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

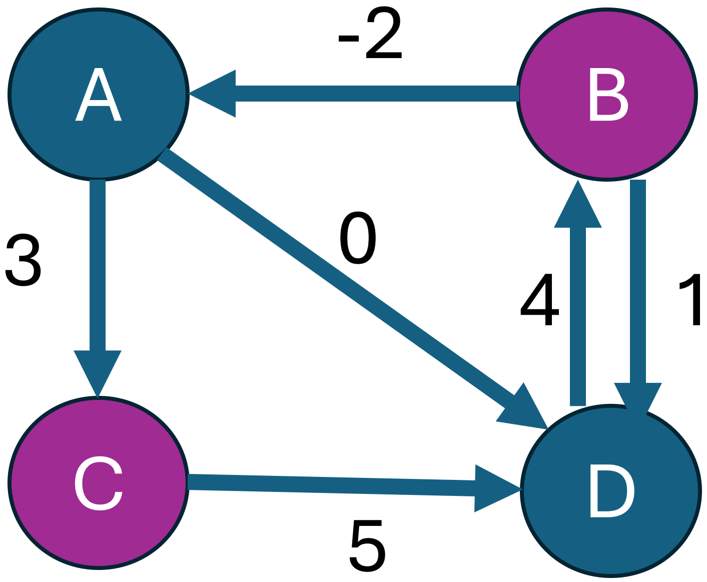
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

CB = ∞ > CB = ∞ + BB=0

```
if dist[3][2] > dist[3][2] + dist[2][2]
    dist[3][2] = dist[3][2] + dist[2][2]
```



```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

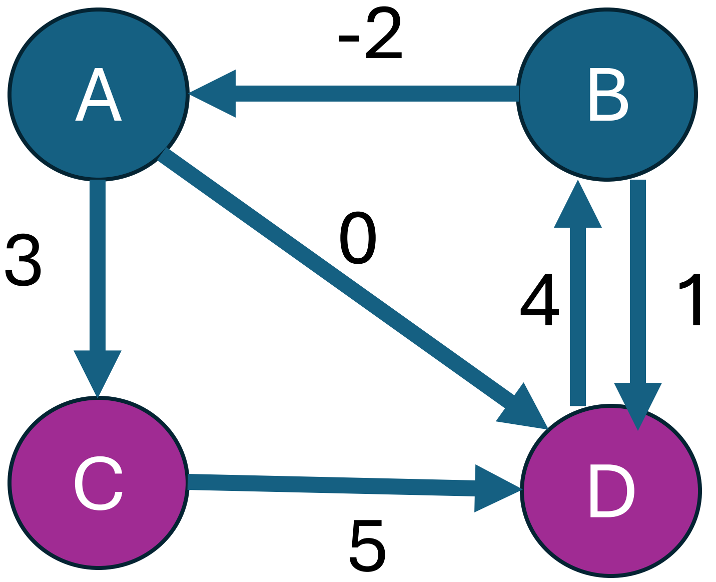
from

	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

CC = 0 > CB = ∞ + BC=1

```
if dist[3][3] > dist[3][2] + dist[2][3]
    dist[3][3] = dist[3][2] + dist[2][3]
```



to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	∞	4	∞	0

to

from

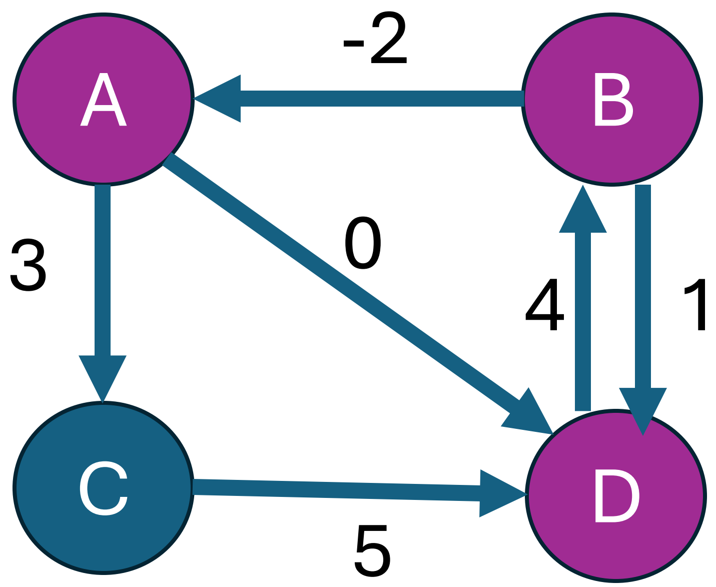
	A	B	C	D
A			A	A
B	B		A	A
C				C
D		D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

CD = 5 > CB = ∞ + BD=-2

```
if dist[3][4] > dist[3][2] + dist[2][4]
    dist[3][4] = dist[3][2] + dist[2][4]
```



to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	∞	0

to

from

	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D		

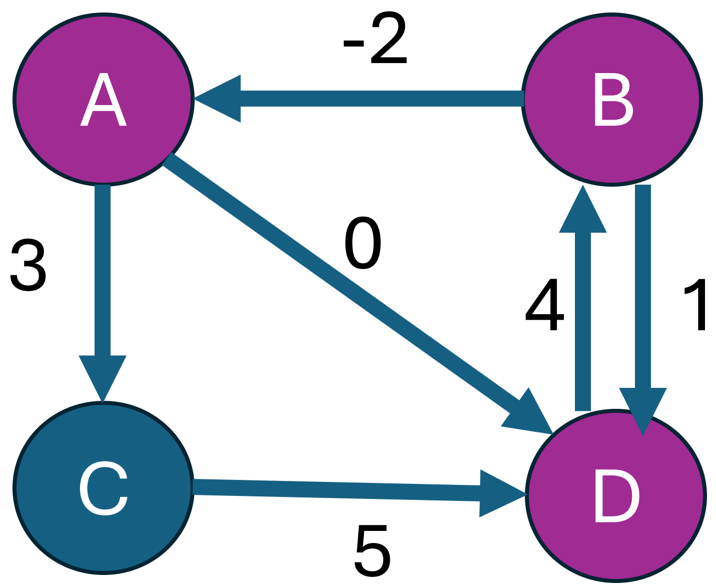
```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

YES!!!!!!

DA = ∞ > DB = 4 + BA=-2

```
if dist[4][1] > dist[4][2] + dist[2][1]
    dist[4][1] = dist[4][2] + dist[2][1]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	∞	0

to

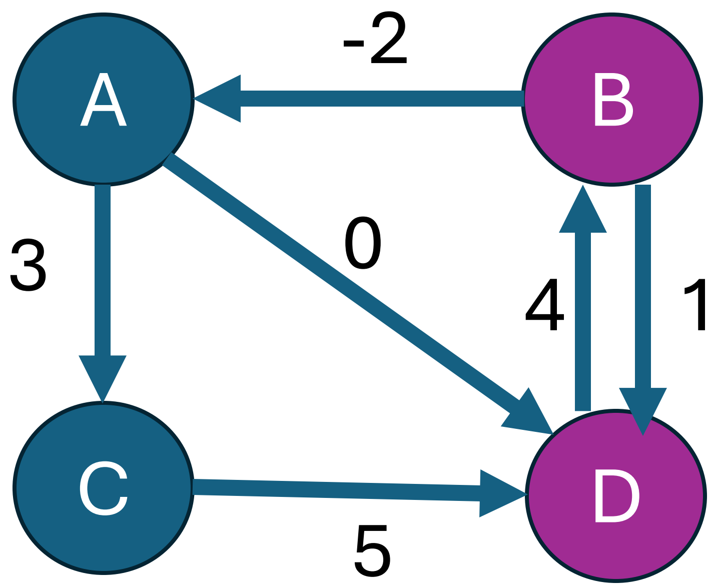
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

DA = ∞ > DB = 1 + BA=-2

```
if dist[4][1] > dist[4][2] + dist[2][1]
    dist[4][1] = dist[4][2] + dist[2][1]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	∞	0

to

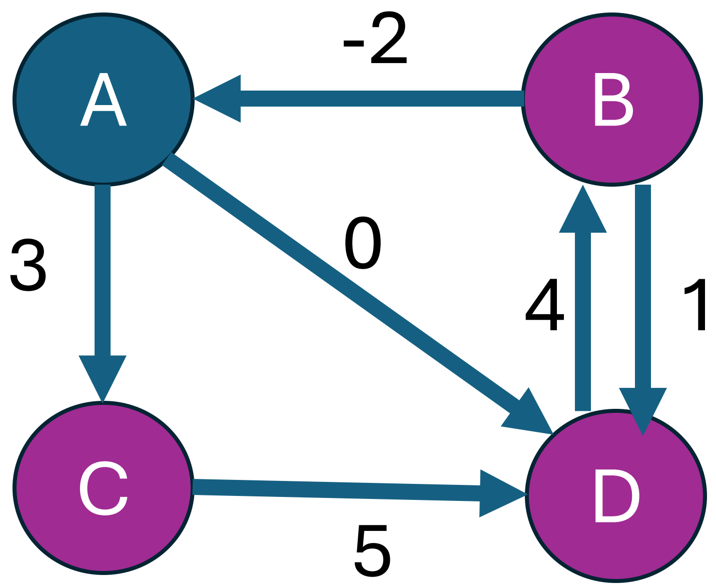
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D		

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

DB = 4 > DB = 4 + BB=0

```
if dist[4][2] > dist[4][2] + dist[2][2]
    dist[4][2] = dist[4][2] + dist[2][2]
```



```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

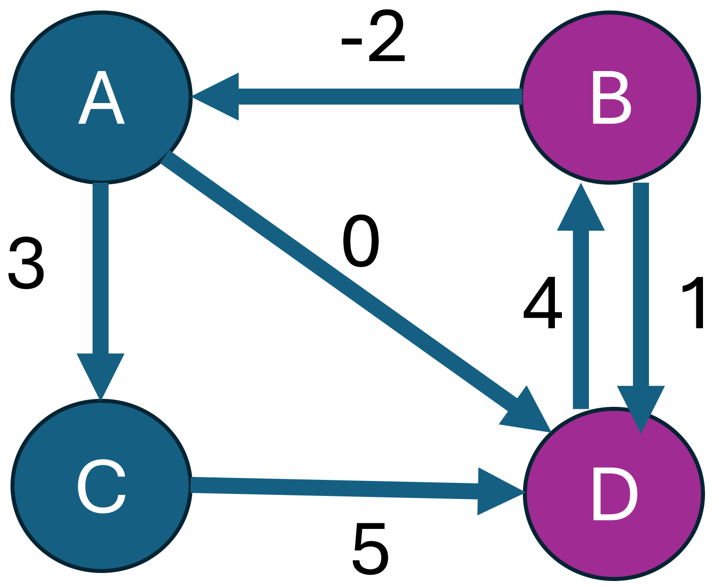
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

YES!!!!!!

DC= ∞ > DB = 4 + BC=1

```
if dist[4][3] > dist[4][2] + dist[2][3]
    dist[4][3] = dist[4][2] + dist[2][3]
```



to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

from

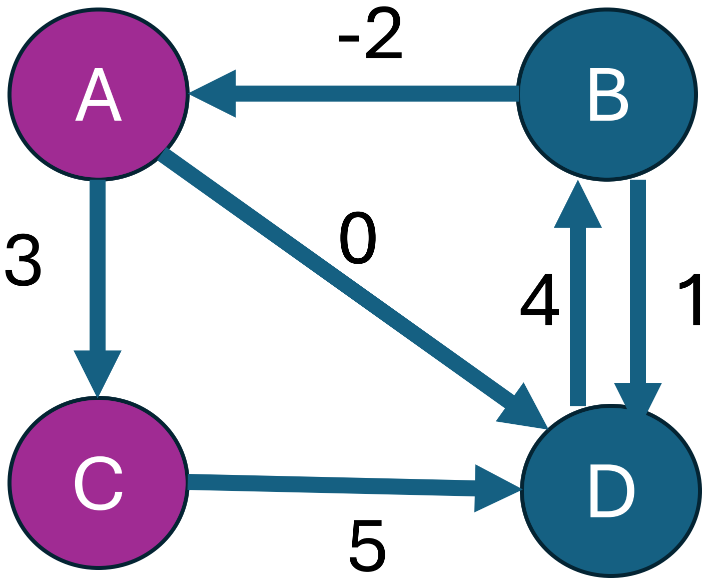
	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

DD= 0 > DB = 4 + BD=-2

```
if dist[4][4] > dist[4][2] + dist[2][4]
    dist[4][4] = dist[4][2] + dist[2][4]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

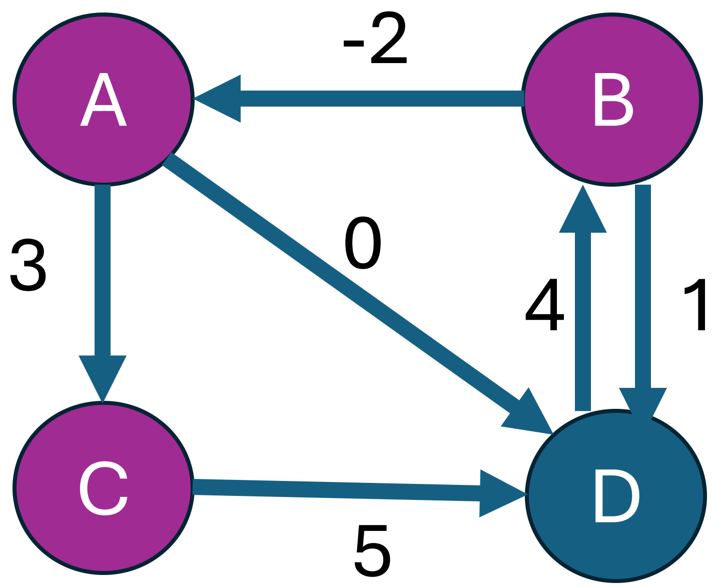
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

AA= 0 > AC = 3 + CA=∞

```
if dist[1][1] > dist[1][3] + dist[3][1]
    dist[1][1] = dist[1][3] + dist[3][1]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

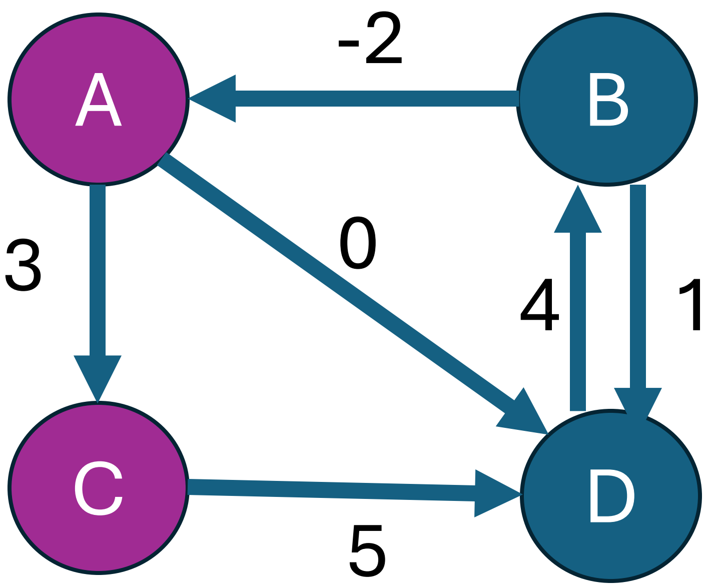
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

AB= ∞ > AC = 3 + CB=∞

```
if dist[1][2] > dist[1][3] + dist[3][2]
    dist[1][2] = dist[1][3] + dist[3][2]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

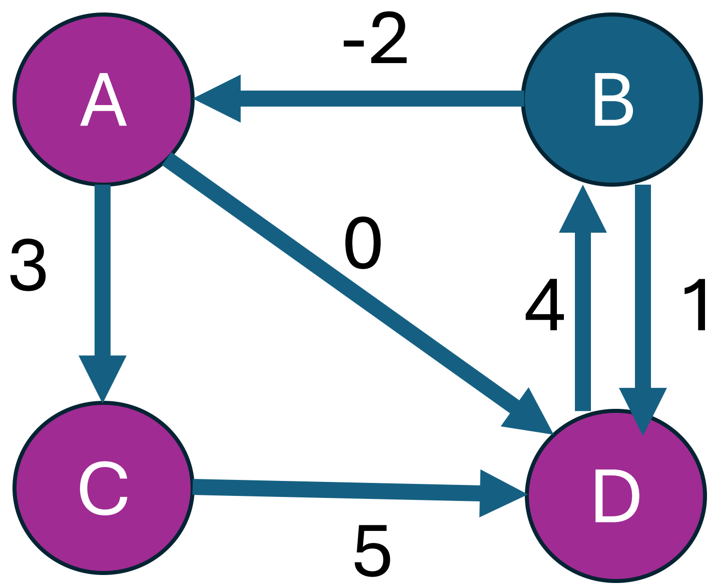
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

AC= 3 > AC = 3 + CC=0

```
if dist[1][3] > dist[1][3] + dist[3][3]
    dist[1][3] = dist[1][3] + dist[3][3]
```



to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

from

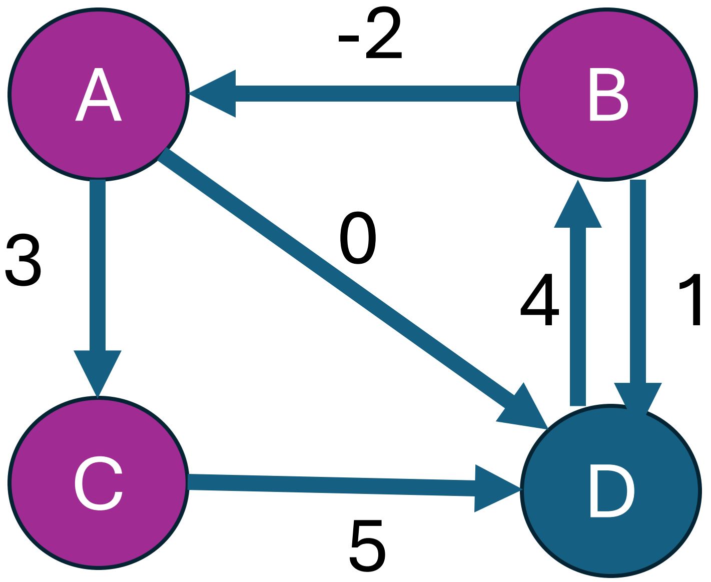
	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

AD= 0 > AC = 3 + CD=5

```
if dist[1][4] > dist[1][3] + dist[3][4]
    dist[1][4] = dist[1][3] + dist[3][4]
```



to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

from

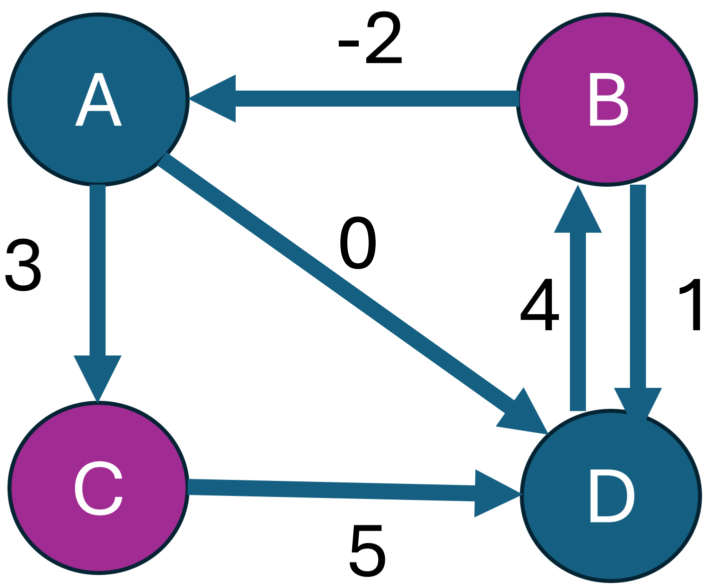
	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

BA= -2 > BC = 1 + CA= ∞

```
if dist[2][1] > dist[2][3] + dist[3][1]
    dist[2][1] = dist[2][3] + dist[3][1]
```



to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

from

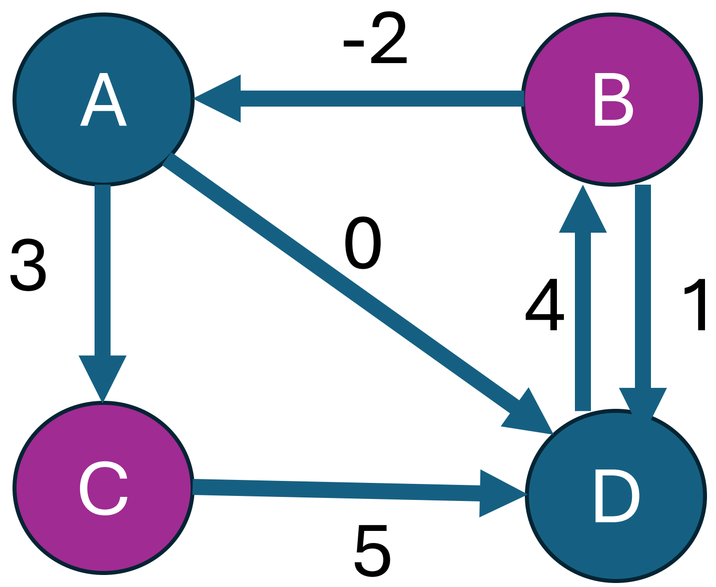
	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

BB= 0 > BC = 1 + CB= ∞

```
if dist[2][2] > dist[2][3] + dist[3][2]
    dist[2][2] = dist[2][3] + dist[3][2]
```



to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

from

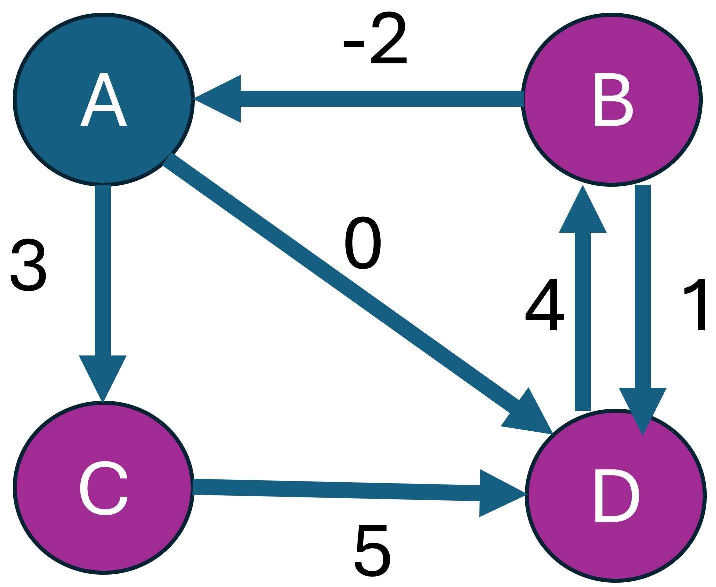
	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1   2   3   4
i = 1   2   3   4
j = 1   2   3   4

BC= 1   >   BC = 1   +   CC= 0

```
if dist[2][3] > dist[2][3] + dist[3][3]
    dist[2][3] = dist[2][3] + dist[3][3]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

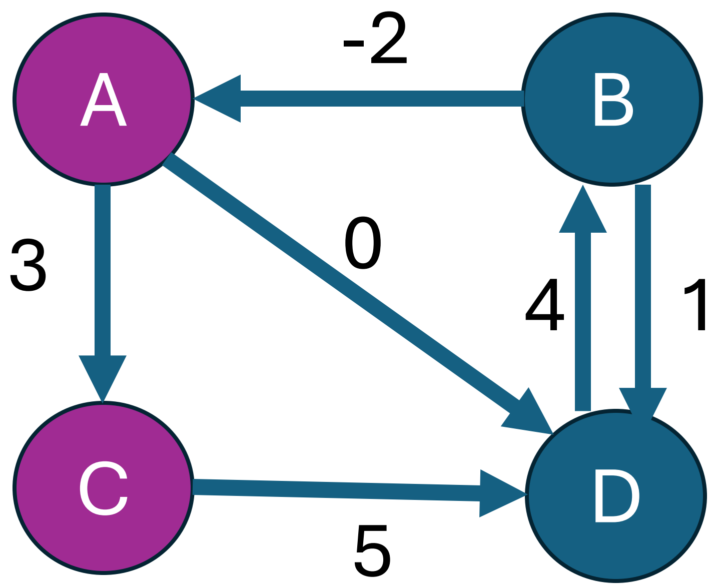
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

BD= -2 > BC = 1 + CD= 5

```
if dist[2][4] > dist[2][3] + dist[3][4]
    dist[2][4] = dist[2][3] + dist[3][4]
```



```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

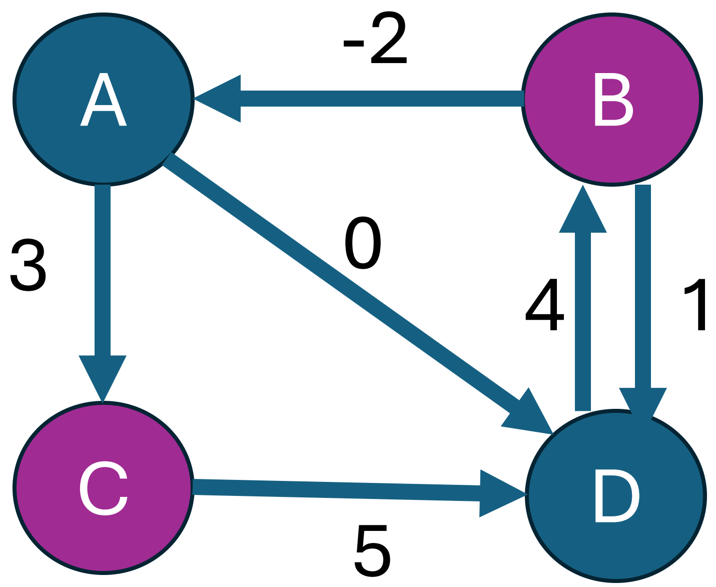
from

	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
k = 1  2  3  4
i = 1  2  3  4
j = 1  2  3  4
```

CA= ∞ > CC = 0 + CA= ∞

```
if dist[3][1] > dist[3][3] + dist[3][1]
    dist[3][1] = dist[3][3] + dist[3][1]
```



to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

from

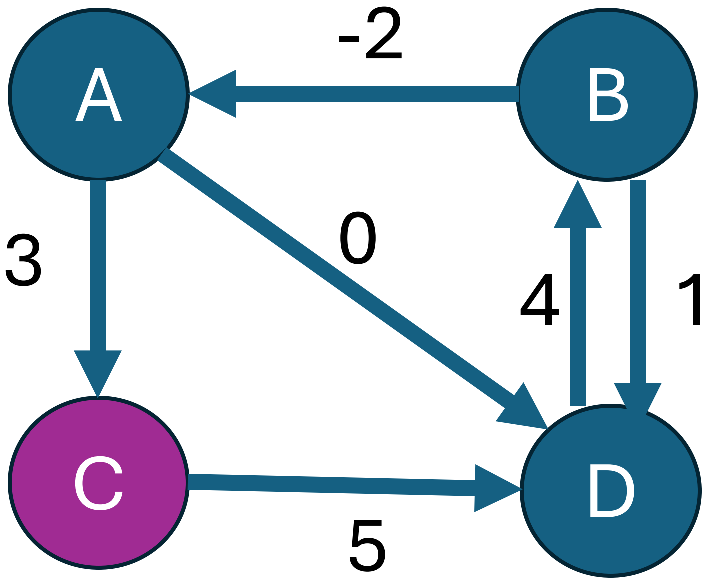
	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

CB= ∞ > CC = 0 + CB= ∞

```
if dist[3][2] > dist[3][3] + dist[3][2]
    dist[3][2] = dist[3][3] + dist[3][2]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

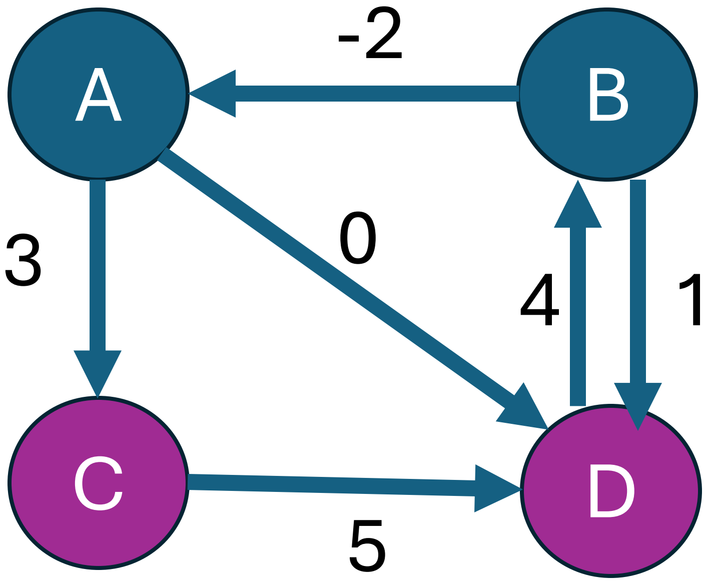
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

CC= 0 > CC = 0 + CC= 0

```
if dist[3][3] > dist[3][3] + dist[3][3]
    dist[3][3] = dist[3][3] + dist[3][3]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

from	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

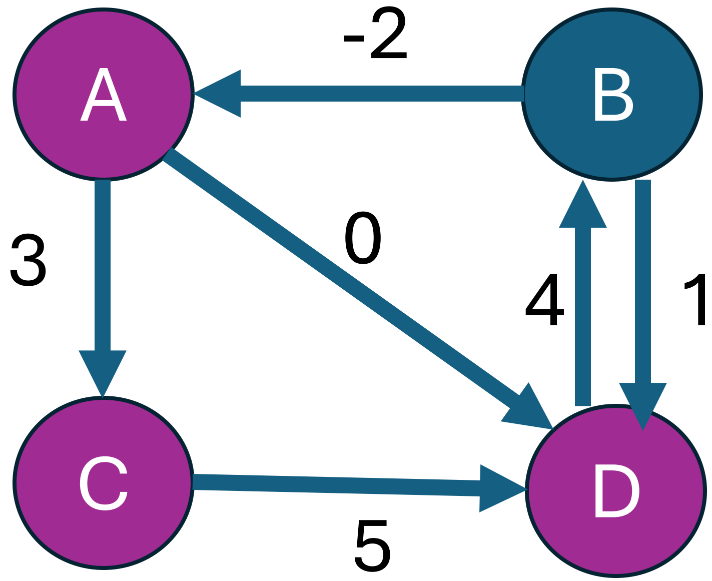
k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4

CD= 5 > CC = 0 + CD= 5

```
if dist[3][4] > dist[3][3] + dist[3][4]
    dist[3][4] = dist[3][3] + dist[3][4]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

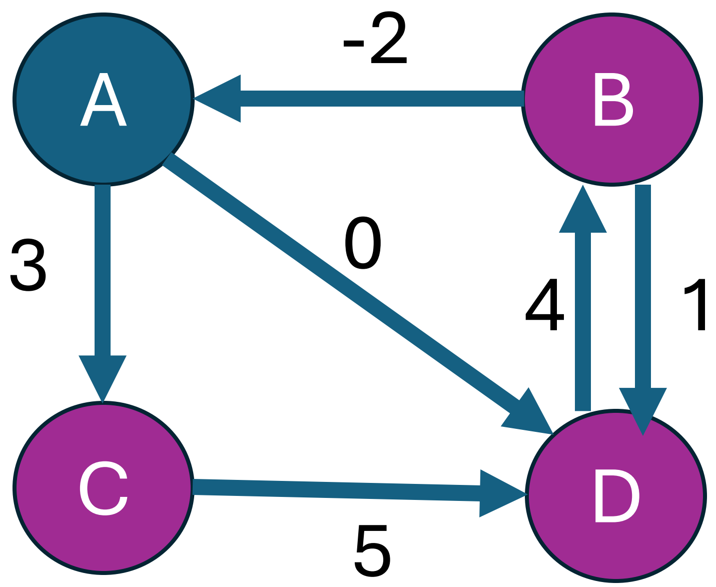
to

from	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1   2   3   4
i = 1   2   3   4
j = 1   2   3   4

```
     DA= 2          >      DC = 5     +     CA= ∞
if dist[4][1] > dist[4][3] + dist[3][1]
    dist[4][1] = dist[4][3] + dist[3][1]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

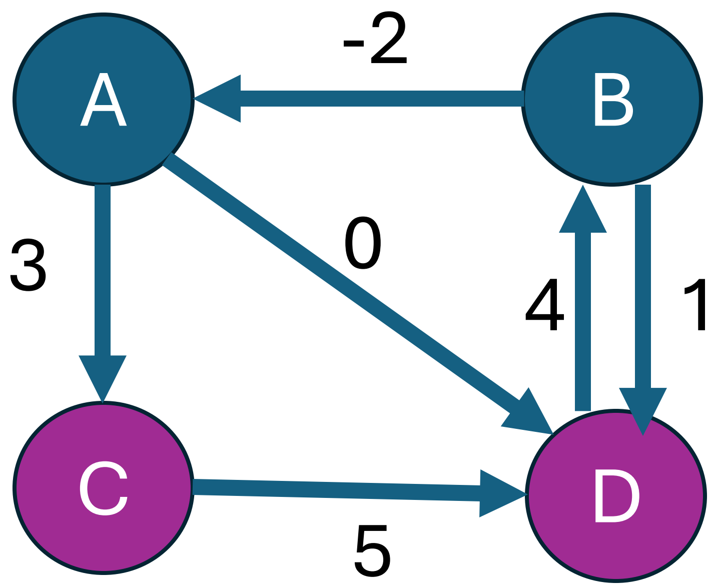
from	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

DB= 4 > DC = 5 + CB= ∞

```
if dist[4][2] > dist[4][3] + dist[3][2]
    dist[4][2] = dist[4][3] + dist[3][2]
```



```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

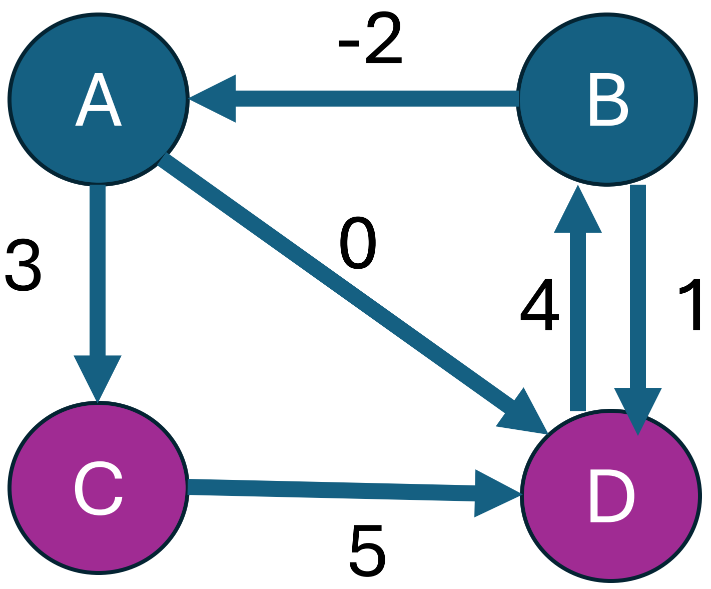
to

from	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

DC= 5 > DC = 5 + CC= 0

```
if dist[4][3] > dist[4][3] + dist[3][3]
    dist[4][3] = dist[4][3] + dist[3][3]
```



to

from	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

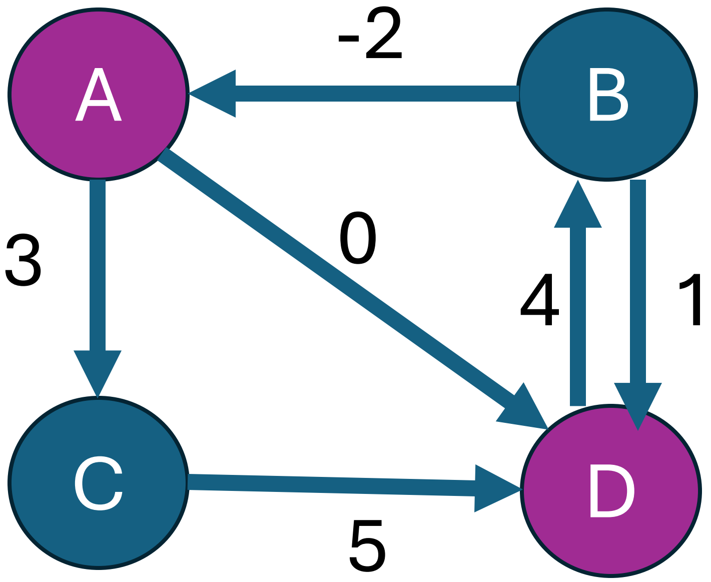
to

from	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

```
     DC= 5           >     DC = 5      +    CC= 0
if dist[4][3] > dist[4][3] + dist[3][3]
    dist[4][3] = dist[4][3] + dist[3][3]
```



to

from

	A	B	C	D
A	0	∞	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

from

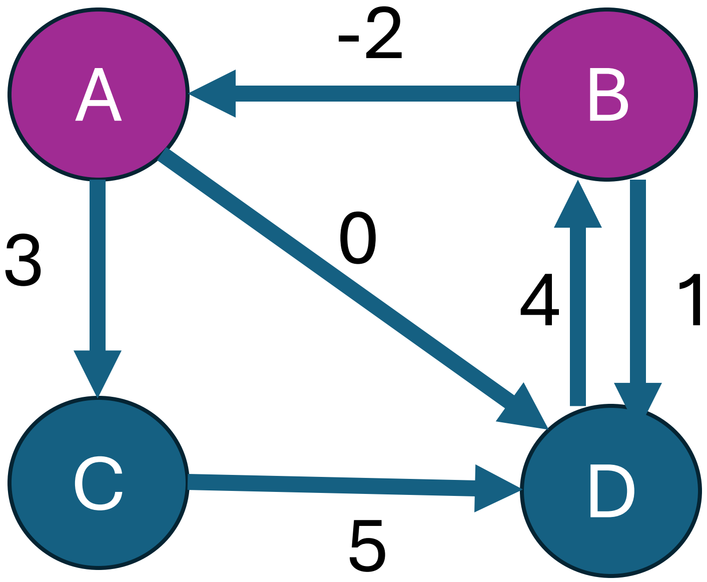
	A	B	C	D
A			A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

AA= 0 > AD = 0 + DA= 0

```
if dist[1][1] > dist[1][4] + dist[4][1]
    dist[1][1] = dist[1][4] + dist[4][1]
```



to

from	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

from	A	B	C	D
A		D	A	A
B	B		A	A
C				C
D	B	D	B	

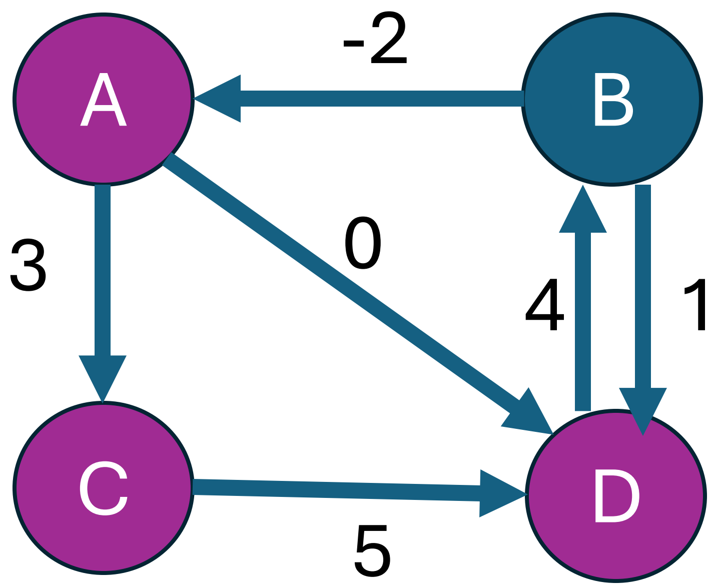
```
if dist[i][j] > dist[i][k] + dist[k][j]
     dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

YES!!!!!!

AB= ∞ > AD = 0 + DB= 4

```
if dist[1][2] > dist[1][4] + dist[4][2]
     dist[1][2] = dist[1][4] + dist[4][2]
```



to

from

	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

from

	A	B	C	D
A		D	A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

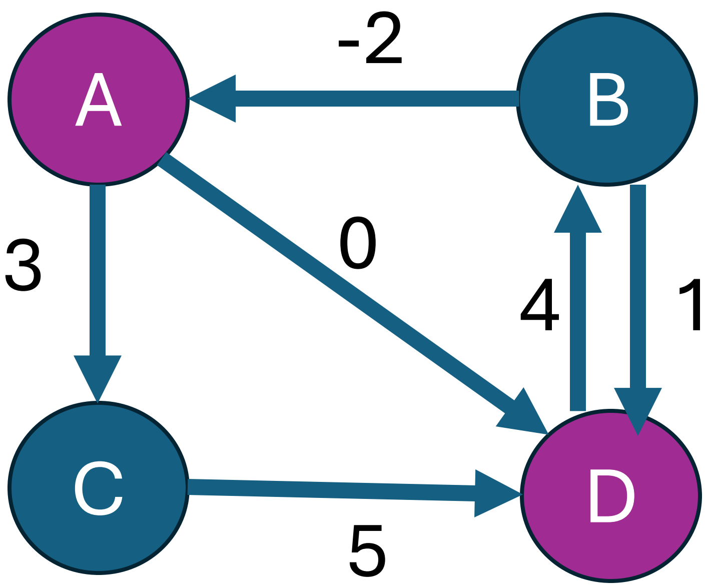
k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4

AC= 3 > AD = 0 + DC= 5

```
if dist[1][3] > dist[1][4] + dist[4][3]
    dist[1][3] = dist[1][4] + dist[4][3]
```



to

from	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

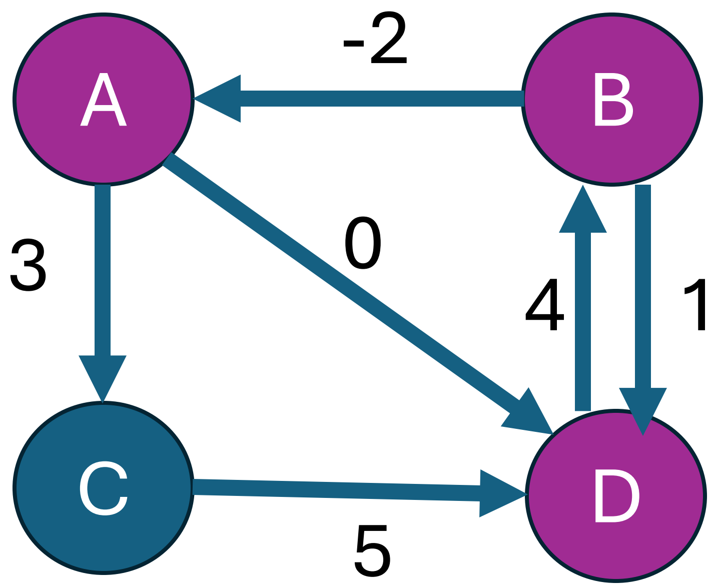
from	A	B	C	D
A		D	A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

AD= 0 > AD = 0 + DD= 0

```
if dist[1][4] > dist[1][4] + dist[4][4]
    dist[1][4] = dist[1][4] + dist[4][4]
```



```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

to

from	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

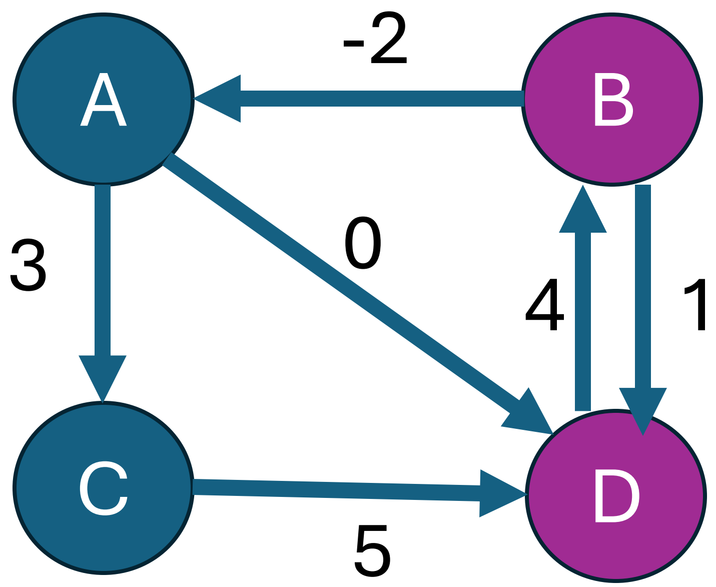
to

from	A	B	C	D
A		D	A	A
B	B		A	A
C				C
D	B	D	B	

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

BA= 0 > BD = -2 + DA= 2

```
if dist[2][1] > dist[2][4] + dist[4][1]
    dist[2][1] = dist[2][4] + dist[4][1]
```



		to			
		A	B	C	D
from	A	0	4	3	0
	B	-2	0	1	-2
	C	∞	∞	0	5
	D	2	4	5	0

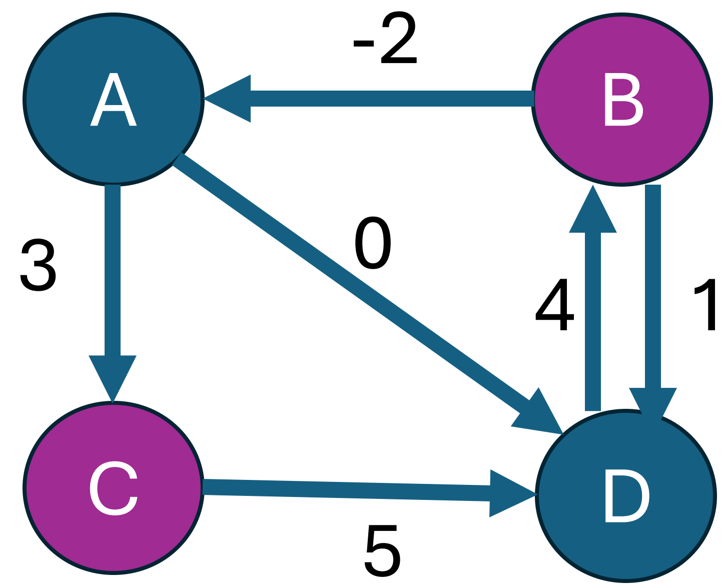
		to			
		A	B	C	D
from	A		D	A	A
	B	B		A	A
	C				C
	D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1   2   3   4
i = 1   2   3   4
j = 1   2   3   4

BB= 0   >   BD = -2   +   DB= 4

```
if dist[2][2] > dist[2][4] + dist[4][2]
    dist[2][2] = dist[2][4] + dist[4][2]
```



to

from	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

from	A	B	C	D
A		D	A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

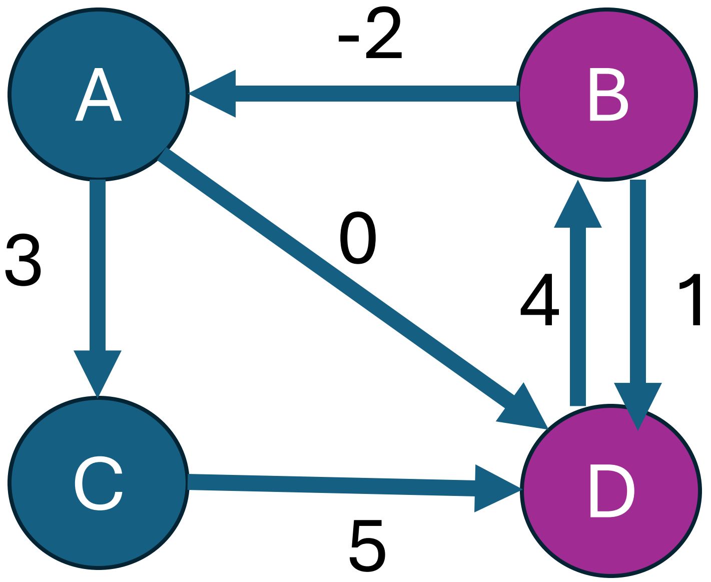
k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4

BC= 1 > BD = -2 + DC= 5

```
if dist[2][3] > dist[2][4] + dist[4][3]
    dist[2][3] = dist[2][4] + dist[4][3]
```



to

from

	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	∞	∞	0	5
D	2	4	5	0

to

from

	A	B	C	D
A		D	A	A
B	B		A	A
C				C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

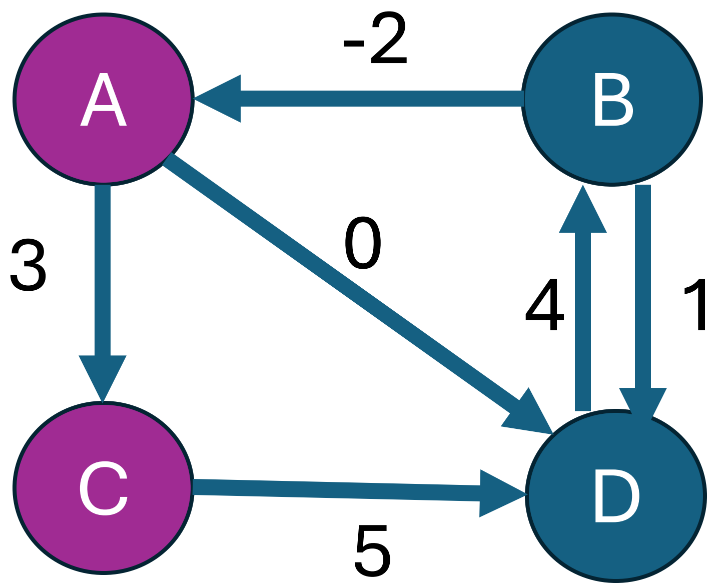
k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4

BD= -2 > BD = -2 + DD= 0

```
if dist[2][4] > dist[2][4] + dist[4][4]
    dist[2][4] = dist[2][4] + dist[4][4]
```



to

from

	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	7	∞	0	5
D	2	4	5	0

to

from

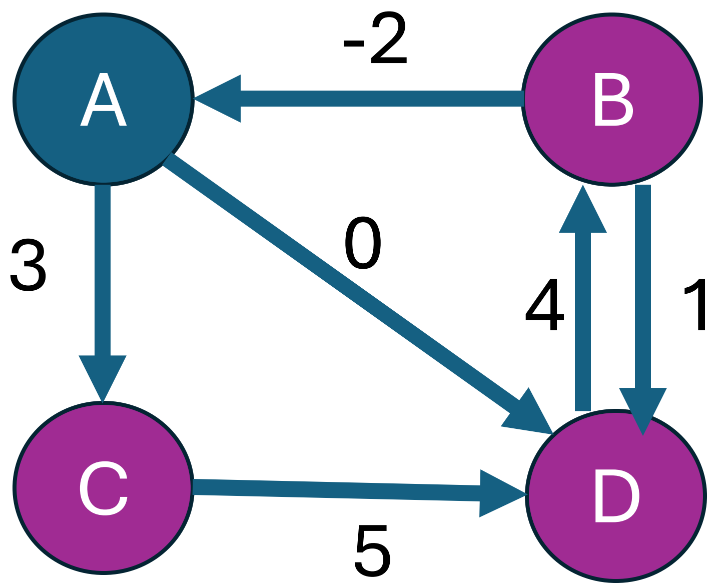
	A	B	C	D
A		D	A	A
B	B		A	A
C	D			C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

YES!!!!!!

```
      CA= ∞        >     CD = 5     +     DA= 2
if dist[3][1] > dist[3][4] + dist[4][1]
    dist[3][1] = dist[3][4] + dist[4][1]
```



to

from	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	7	9	0	5
D	2	4	5	0

to

from	A	B	C	D
A		D	A	A
B	B		A	A
C	D	D		C
D	B	D	B	

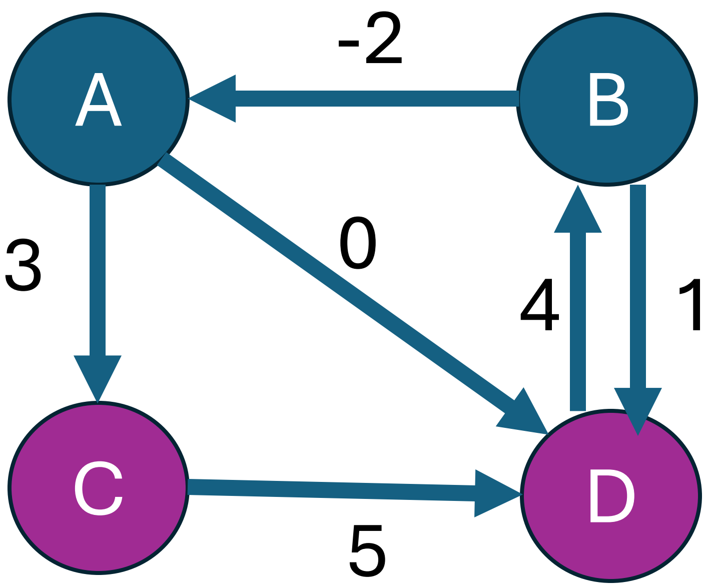
```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

YES!!!!!!

CB= ∞ > CD = 5 + DB= 4

```
if dist[3][2] > dist[3][4] + dist[4][2]
    dist[3][2] = dist[3][4] + dist[4][2]
```



to

from	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	7	9	0	5
D	2	4	5	0

to

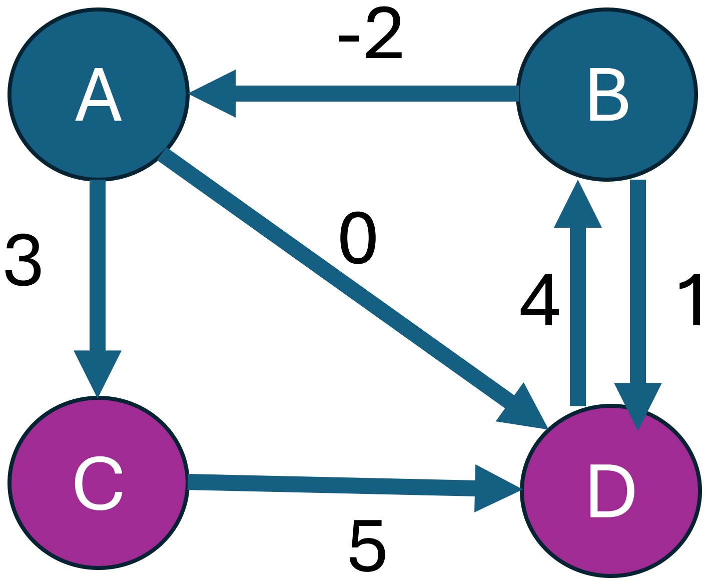
from	A	B	C	D
A		D	A	A
B	B		A	A
C	D	D		C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

```
k = 1  2  3  4
i = 1  2  3  4
j = 1  2  3  4
```

CC= 0 > CD = 5 + DC= 5

```
if dist[3][3] > dist[3][4] + dist[4][3]
    dist[3][3] = dist[3][4] + dist[4][3]
```



to

from

	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	7	9	0	5
D	2	4	5	0

to

from

	A	B	C	D
A		D	A	A
B	B		A	A
C	D	D		C
D	B	D	B	

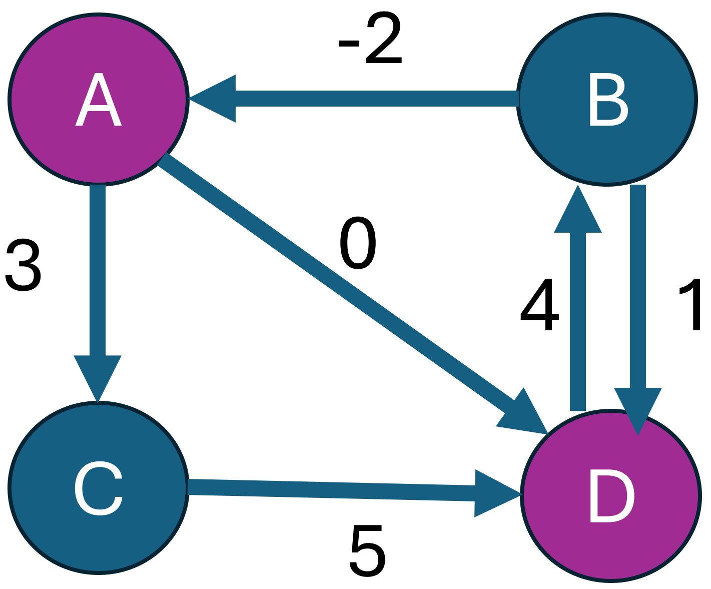
```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4

```
         CD= 5          >      CD = 5      +     DD= 0
if dist[3][4] > dist[3][4] + dist[4][4]
   dist[3][4] = dist[3][4] + dist[4][4]
```



to

from	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	7	9	0	5
D	2	4	5	0

to

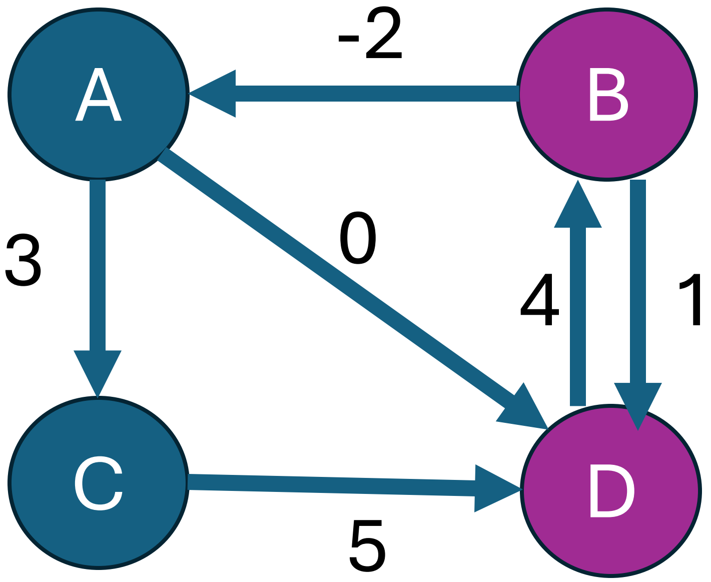
from	A	B	C	D
A		D	A	A
B	B		A	A
C	D	D		C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

DA= 2 > DD = 0 + DA= 2

```
if dist[4][1] > dist[4][4] + dist[4][1]
    dist[4][1] = dist[4][4] + dist[4][1]
```



to

from	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	7	9	0	5
D	2	4	5	0

to

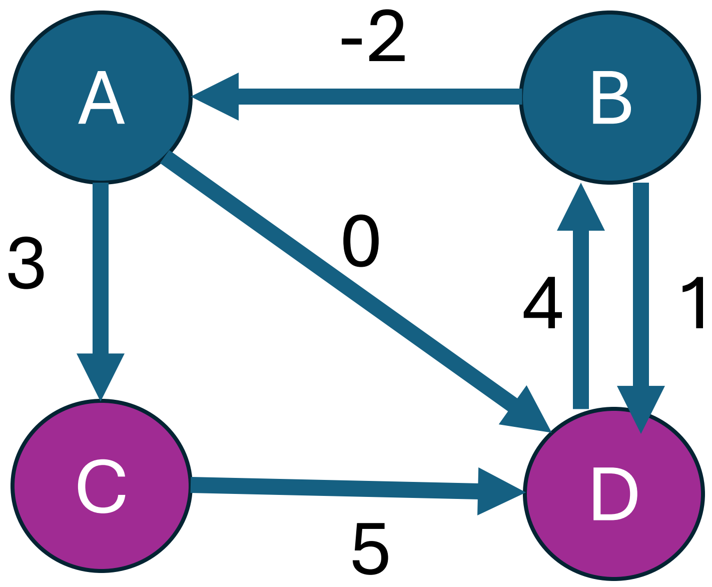
from	A	B	C	D
A		D	A	A
B	B		A	A
C	D	D		C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4
i = 1 2 3 4
j = 1 2 3 4

DB= 4 > DD = 0 + DB= 4

```
if dist[4][2] > dist[4][4] + dist[4][2]
    dist[4][2] = dist[4][4] + dist[4][2]
```



to

from

	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	7	9	0	5
D	2	4	5	0

to

from

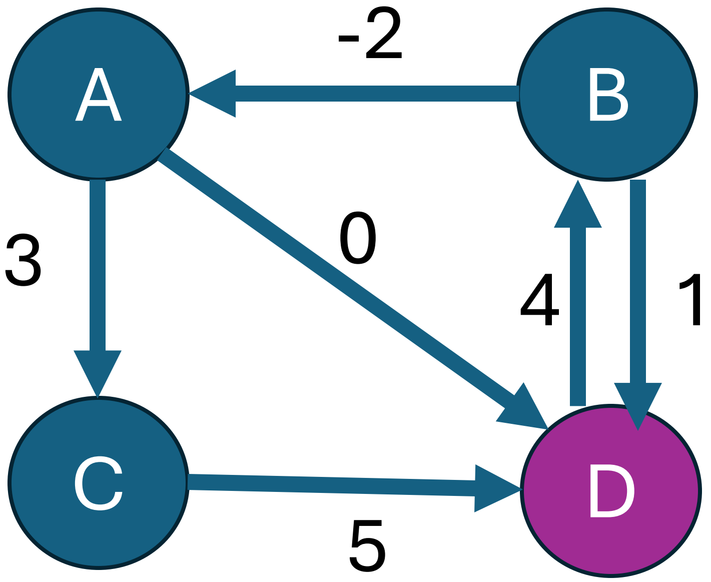
	A	B	C	D
A		D	A	A
B	B		A	A
C	D	D		C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1   2   3   4
i = 1   2   3   4
j = 1   2   3   4

DC= 5   >   DD = 0   +   DC= 5

```
if dist[4][3] > dist[4][4] + dist[4][3]
    dist[4][3] = dist[4][4] + dist[4][3]
```



to

from	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	7	9	0	5
D	2	4	5	0

to

from	A	B	C	D
A		D	A	A
B	B		A	A
C	D	D		C
D	B	D	B	

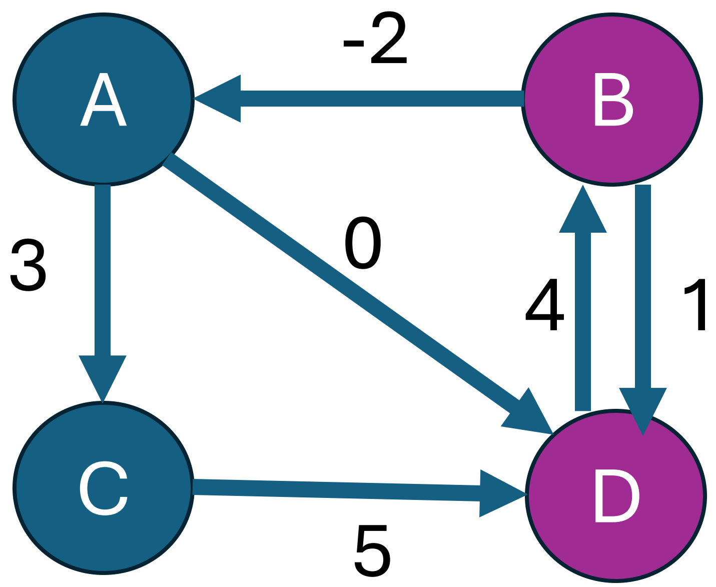
```
if dist[i][j] > dist[i][k] + dist[k][j]
   dist[i][j] = dist[i][k] + dist[k][j]
```

k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4

```
     DD= 0          >       DD = 0      +      DD= 0
if dist[4][4] > dist[4][4] + dist[4][4]
   dist[4][4] = dist[4][4] + dist[4][4]
```



to

from

	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	7	9	0	5
D	2	4	5	0

to

from

	A	B	C	D
A		D	A	A
B	B		A	A
C	D	D		C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

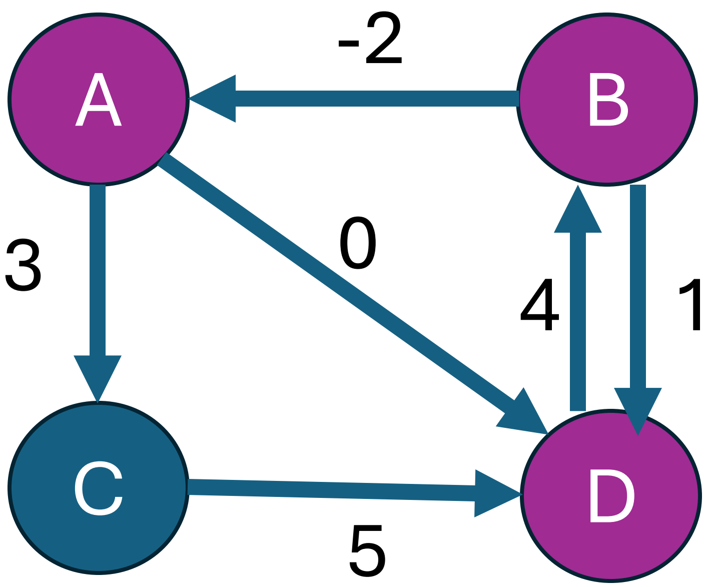
k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4

DD= 0 > DD = 0 + DD= 0

```
if dist[4][4] > dist[4][4] + dist[4][4]
    dist[4][4] = dist[4][4] + dist[4][4]
```



to

from

	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	7	9	0	5
D	2	4	5	0

to

from

	A	B	C	D
A		D	A	A
B	B		A	A
C	D	D		C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

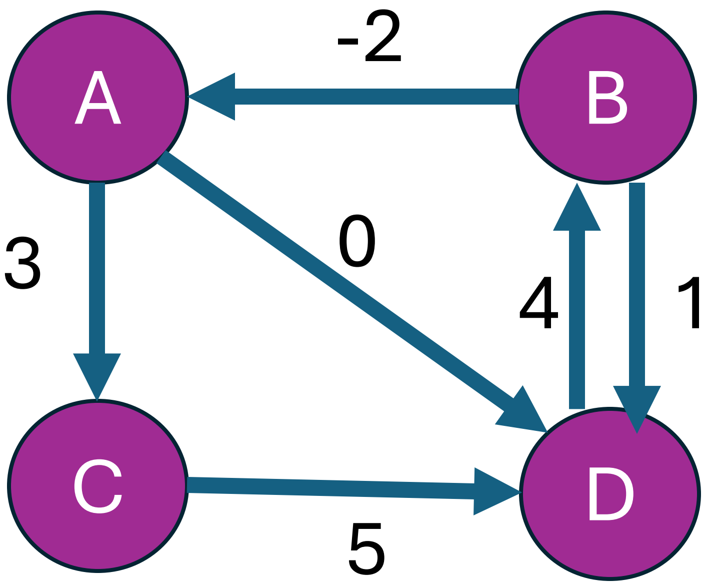
k = 1 2 3 4

i = 1 2 3 4

j = 1 2 3 4

DD= 0 > DD = 0 + DD= 0

```
if dist[4][4] > dist[4][4] + dist[4][4]
    dist[4][4] = dist[4][4] + dist[4][4]
```



to

from

	A	B	C	D
A	0	4	3	0
B	-2	0	1	-2
C	7	9	0	5
D	2	4	5	0

to

from

	A	B	C	D
A		D	A	A
B	B		A	A
C	D	D		C
D	B	D	B	

```
if dist[i][j] > dist[i][k] + dist[k][j]
    dist[i][j] = dist[i][k] + dist[k][j]
```

```
k = 1  2  3  4
i = 1  2  3  4
j = 1  2  3  4
```

```
       DD= 0         >      DD = 0     +    DD= 0
if dist[4][4] > dist[4][4] + dist[4][4]
    dist[4][4] = dist[4][4] + dist[4][4]
```

Johnson’s Algorithm

Johnson’s Algorithm

```
Johnson(G)
create G` where G`.V = G.V + {s}
G`.E = G.E + ((s, u) for u in G.V),
and weight(s, u) = 0 for u in G.V

if Bellman-Ford(s) == False
    return "The input graph has a negative weight cycle"
else:
    for vertex v in G`.V:
        h(v) = distance(s, v) computed by Bellman-Ford
    for edge (u, v) in G`.E:
        weight`(u, v) = weight(u, v) + h(u) - h(v)

D = new matrix of distances initialized to infinity
for vertex u in G.V:
    run Dijkstra(G, weight`, u) to compute distance`(u, v) for all v in G.V
    for each vertex v in G.V:
            D_(u, v) = distance`(u, v) + h(v) - h(u) return D
```

Johnson’s Algorithm

Johnson(G)

```
create G` where G`.V = G.V + {s}
G`.E = G.E + ((s, u) for u in G.V),
and weight(s, u) = 0 for u in G.V

if Bellman-Ford(s) == False
    return "The input graph has a negative weight cycle"
else:
    for vertex v in G`.V:
        h(v) = distance(s, v) computed by Bellman-Ford
    for edge (u, v) in G`.E:
        weight`(u, v) = weight(u, v) + h(u) - h(v)

D = new matrix of distances initialized to infinity
for vertex u in G.V:
    run Dijkstra(G, weight`, u) to compute distance`(u, v) for all v in G.V
    for each vertex v in G.V:
            D_(u, v) = distance`(u, v) + h(v) - h(u) return D
```

. Adding a new vertex, *ss*, to the graph and connecting it to all other vertices with a zero weight edge

Johnson’s Algorithm

```
Johnson(G)
create G` where G`.V = G.V + {s}
G`.E = G.E + ((s, u) for u in G.V),
and weight(s, u) = 0 for u in G.V

if Bellman-Ford(s) == False
    return "The input graph has a negative weight cycle"
else:
    for vertex v in G`.V:
        h(v) = distance(s, v) computed by Bellman-Ford
    for edge (u, v) in G`.E:
        weight`(u, v) = weight(u, v) + h(u) - h(v)

D = new matrix of distances initialized to infinity
for vertex u in G.V:
    run Dijkstra(G, weight`, u) to compute distance`(u, v) for all v in G.V
    for each vertex v in G.V:
            D_(u, v) = distance`(u, v) + h(v) - h(u) return D
```

Reweighting is a process by which edge weight is changed to satisfy two properties.

1. For all pairs of vertices u, v in the graph, if a certain path is the shortest path between those vertices before reweighting, it must also be the shortest path between those vertices after reweighting.
2. For all edges, (u, v), in the graph, weight(u, v) must be non-negative.

Johnson’s Algorithm

```
Johnson(G)
create G` where G`.V = G.V + {s}
G`.E = G.E + ((s, u) for u in G.V),
and weight(s, u) = 0 for u in G.V

if Bellman-Ford(s) == False
    return "The input graph has a negative weight cycle"
else:
    for vertex v in G`.V:
        h(v) = distance(s, v) computed by Bellman-Ford
    for edge (u, v) in G`.E:
        weight`(u, v) = weight(u, v) + h(u) - h(v)

D = new matrix of distances initialized to infinity
for vertex u in G.V:
    run Dijkstra(G, weight`, u) to compute distance`(u, v) for all v in G.V
    for each vertex v in G.V:
            D_(u, v) = distance`(u, v) + h(v) - h(u) return D
```

Finally, Dijkstra's algorithm is run on all vertices to find the shortest path. This is possible because the weights have been transformed into non-negative weights.

Johnson’s Algorithm

```
Johnson(G)
create G` where G`.V = G.V + {s}
G`.E = G.E + ((s, u) for u in G.V),
and weight(s, u) = 0 for u in G.V

if Bellman-Ford(s) == False
    return "The input graph has a negative weight cycle"
else:
    for vertex v in G`.V:
        h(v) = distance(s, v) computed by Bellman-Ford
    for edge (u, v) in G`.E:
        weight`(u, v) = weight(u, v) + h(u) - h(v)

D = new matrix of distances initialized to infinity
for vertex u in G.V:
    run Dijkstra(G, weight`, u) to compute distance`(u, v) for all v in G.V
    for each vertex v in G.V:
            D_(u, v) = distance`(u, v) + h(v) - h(u) return D
```

O(V)

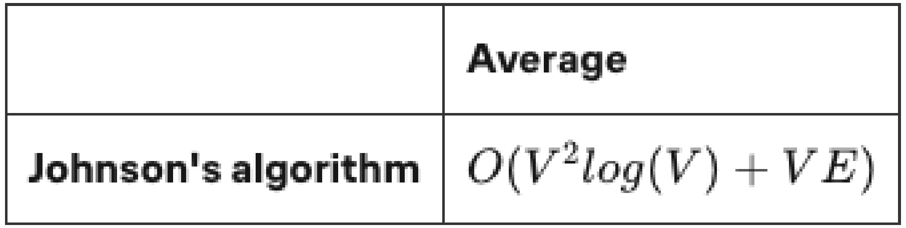
Johnson’s Algorithm

```
Johnson(G)
create G` where G`.V = G.V + {s}                                O(V)
G`.E = G.E + ((s, u) for u in G.V),
and weight(s, u) = 0 for u in G.V

if Bellman-Ford(s) == False
    return "The input graph has a negative weight cycle"       O(VE)
else:
    for vertex v in G`.V:
        h(v) = distance(s, v) computed by Bellman-Ford
    for edge (u, v) in G`.E:
        weight`(u, v) = weight(u, v) + h(u) - h(v)

D = new matrix of distances initialized to infinity
for vertex u in G.V:
    run Dijkstra(G, weight`, u) to compute distance`(u, v) for all v in G.V
    for each vertex v in G.V:
            D_(u, v) = distance`(u, v) + h(v) - h(u) return D
```

Johnson’s Algorithm



	Average
Johnson's algorithm	$O(V^2log(V)+VE)$

```
Johnson(G)
create G` where G`.V = G.V + {s}
G`.E = G.E + ((s, u) for u in G.V),
and weight(s, u) = 0 for u in G.V
```

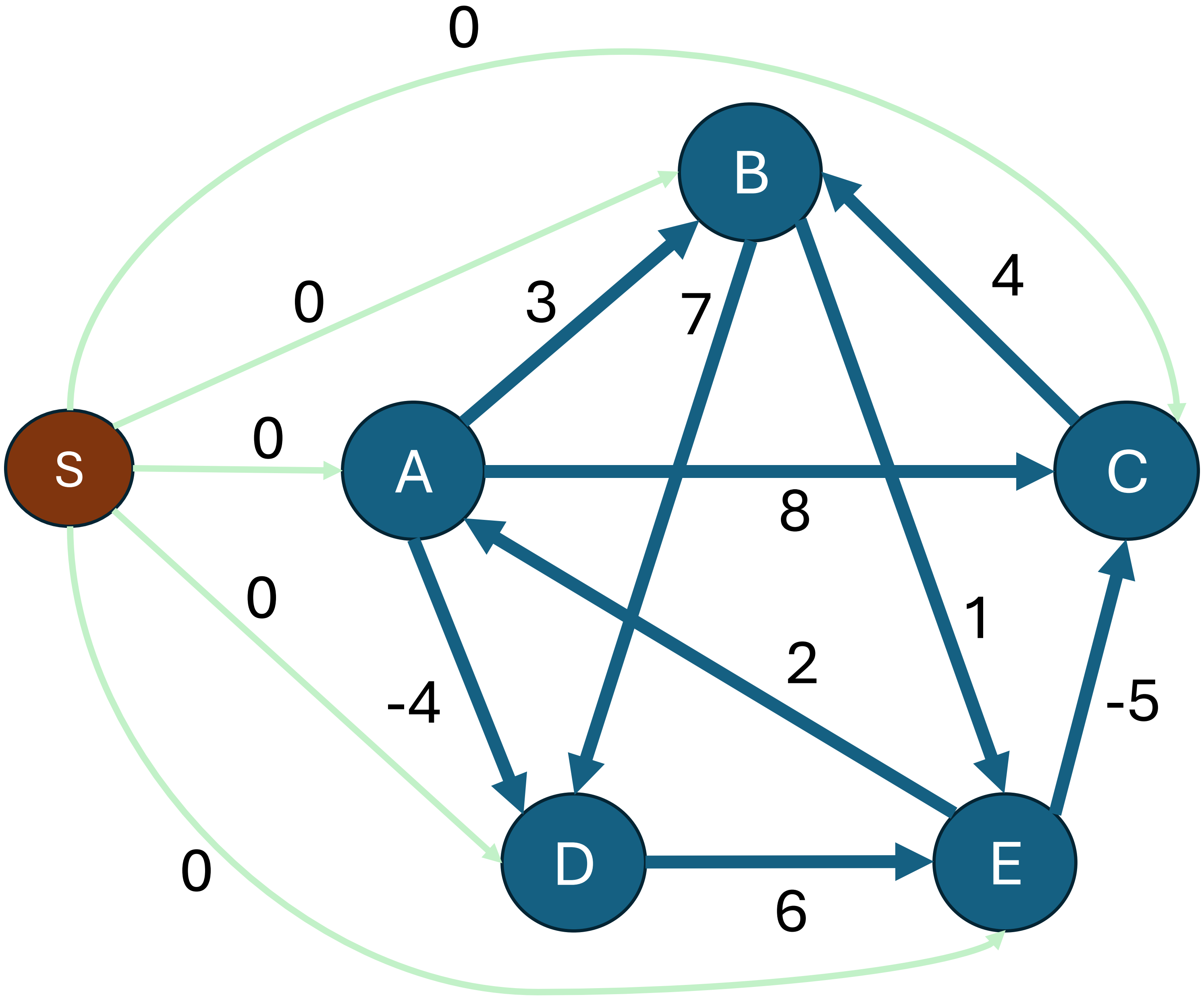
O(V)

```
if Bellman-Ford(s) == False
    return "The input graph has a negative weight cycle"
else:
    for vertex v in G`.V:
        h(v) = distance(s, v) computed by Bellman-Ford
    for edge (u, v) in G`.E:
        weight`(u, v) = weight(u, v) + h(u) - h(v)
```

O(VE)

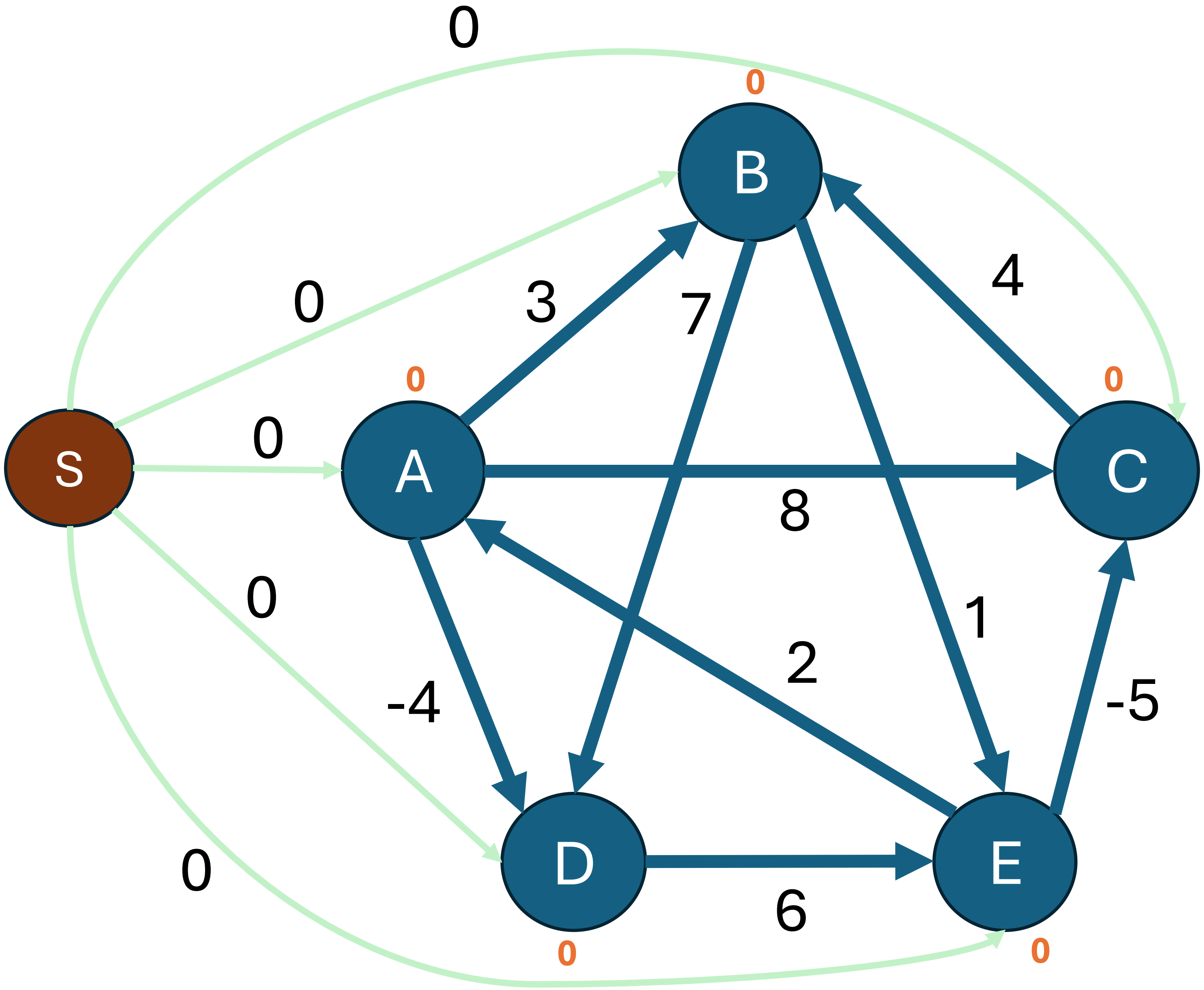
```
D = new matrix of distances initialized to infinity
for vertex u in G.V:
    run Dijkstra(G, weight`, u) to compute distance`(u, v) for all v in G.V
    for each vertex v in G.V:
            D_(u, v) = distance`(u, v) + h(v) - h(u) return D
```

O(V)
O((V+E)LogV)



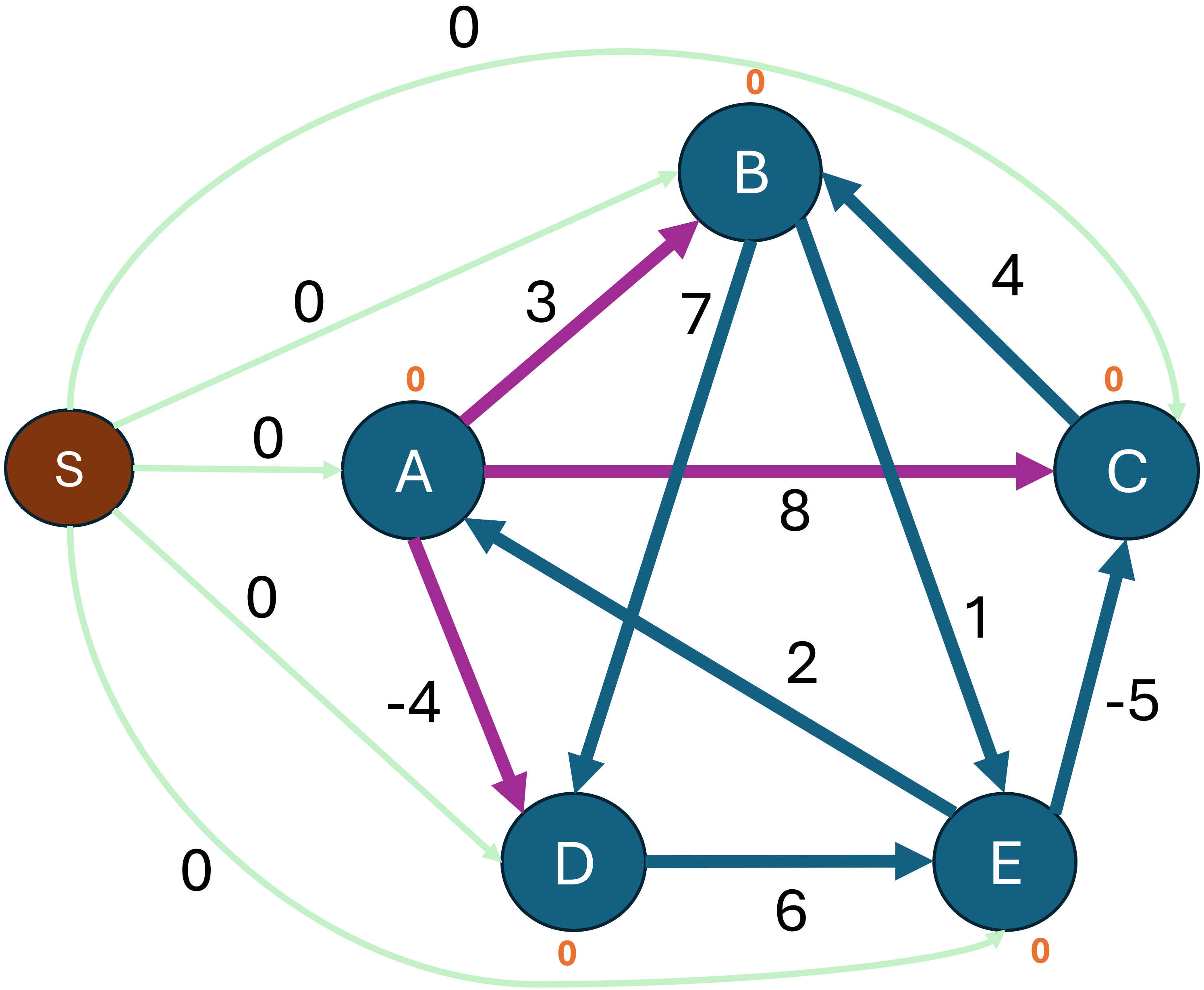
	Cost	Prev
S	0	-
A	0	S
B	0	S
C	0	S
D	0	S
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



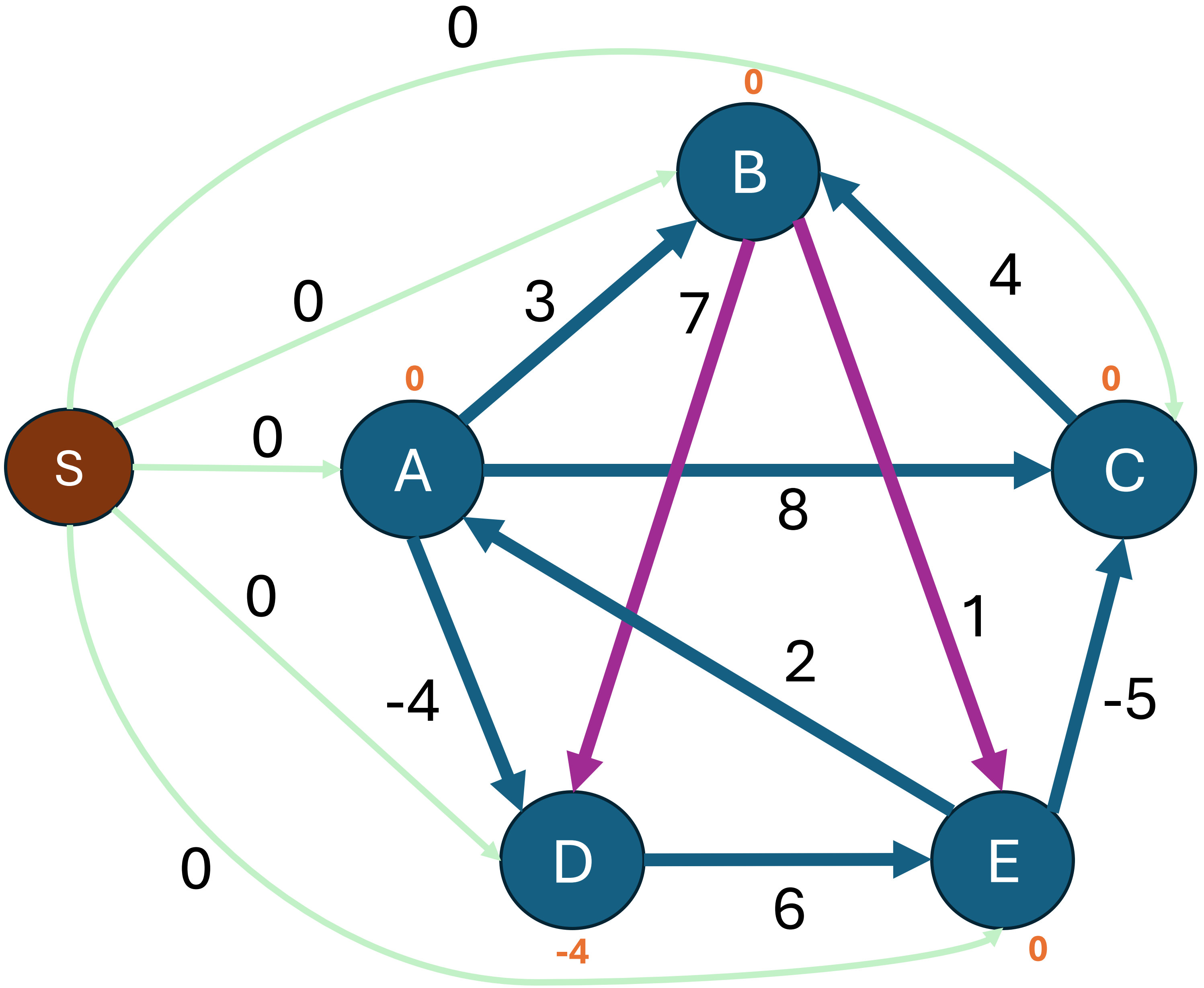
	Cost	Prev
S	0	-
A	0	S
B	0	S
C	0	S
D	0	S
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



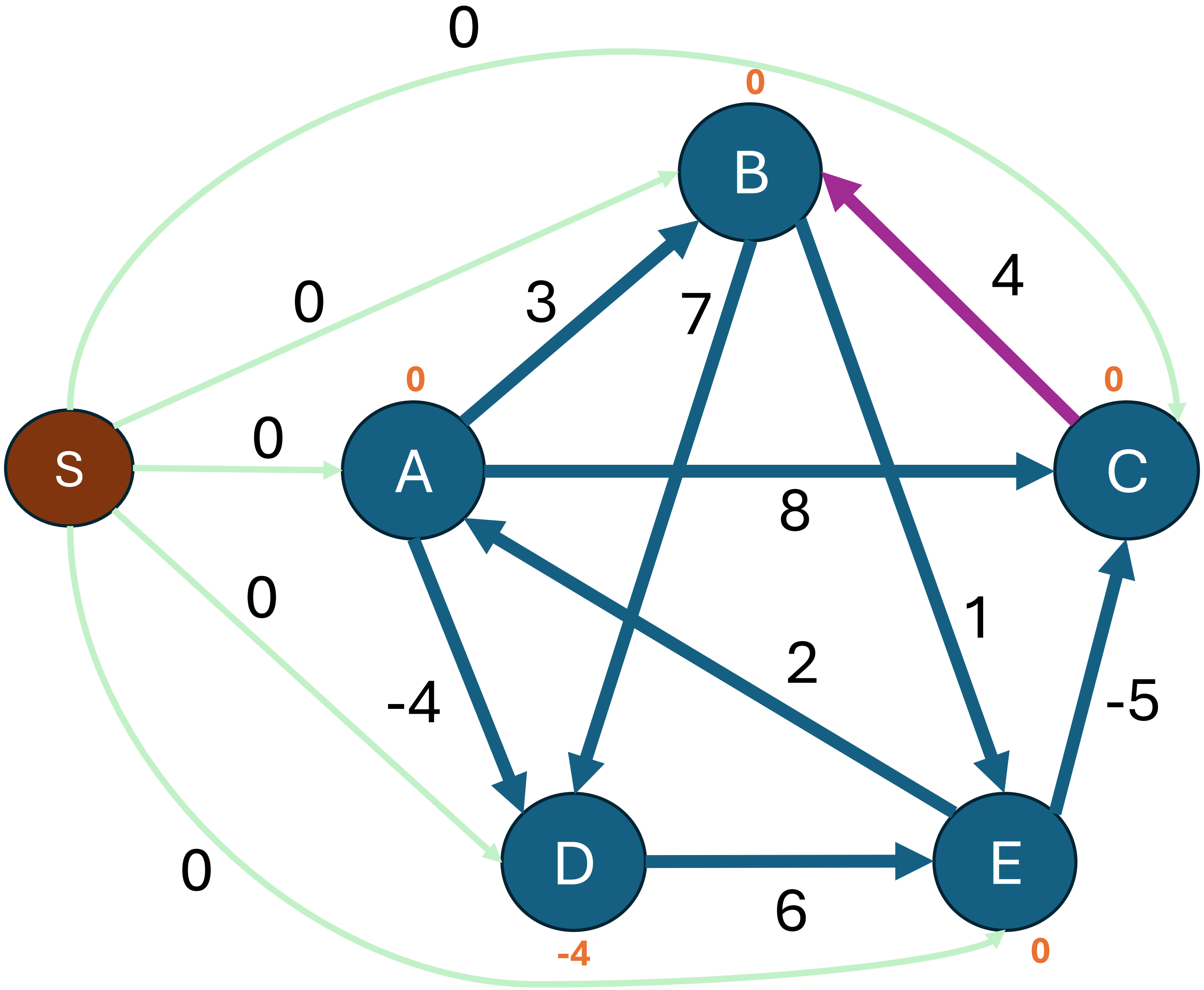
	Cost	Prev
S	0	-
A	0	S
B	0	S
C	0	S
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



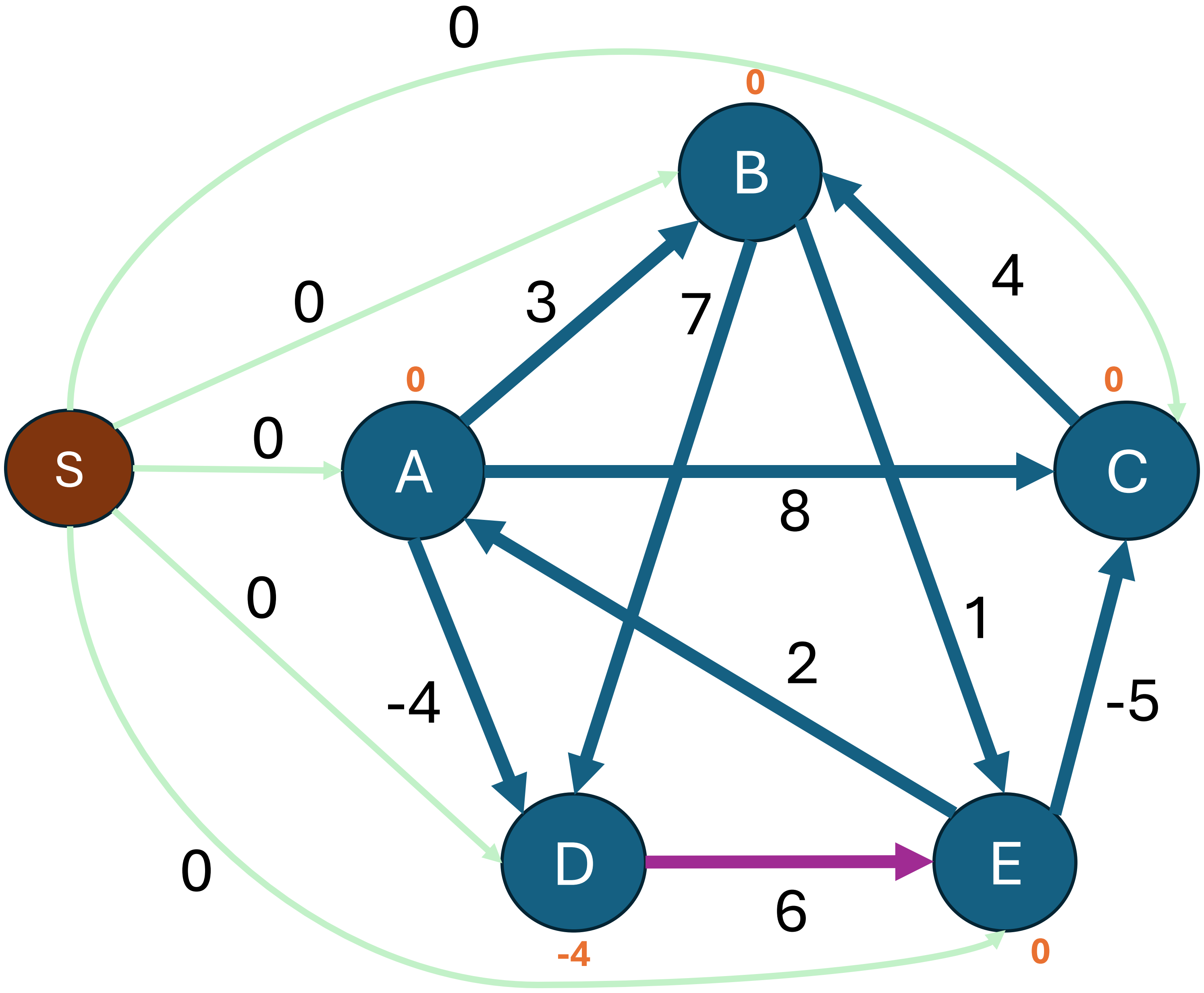
	Cost	Prev
S	0	-
A	0	S
B	0	S
C	0	S
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



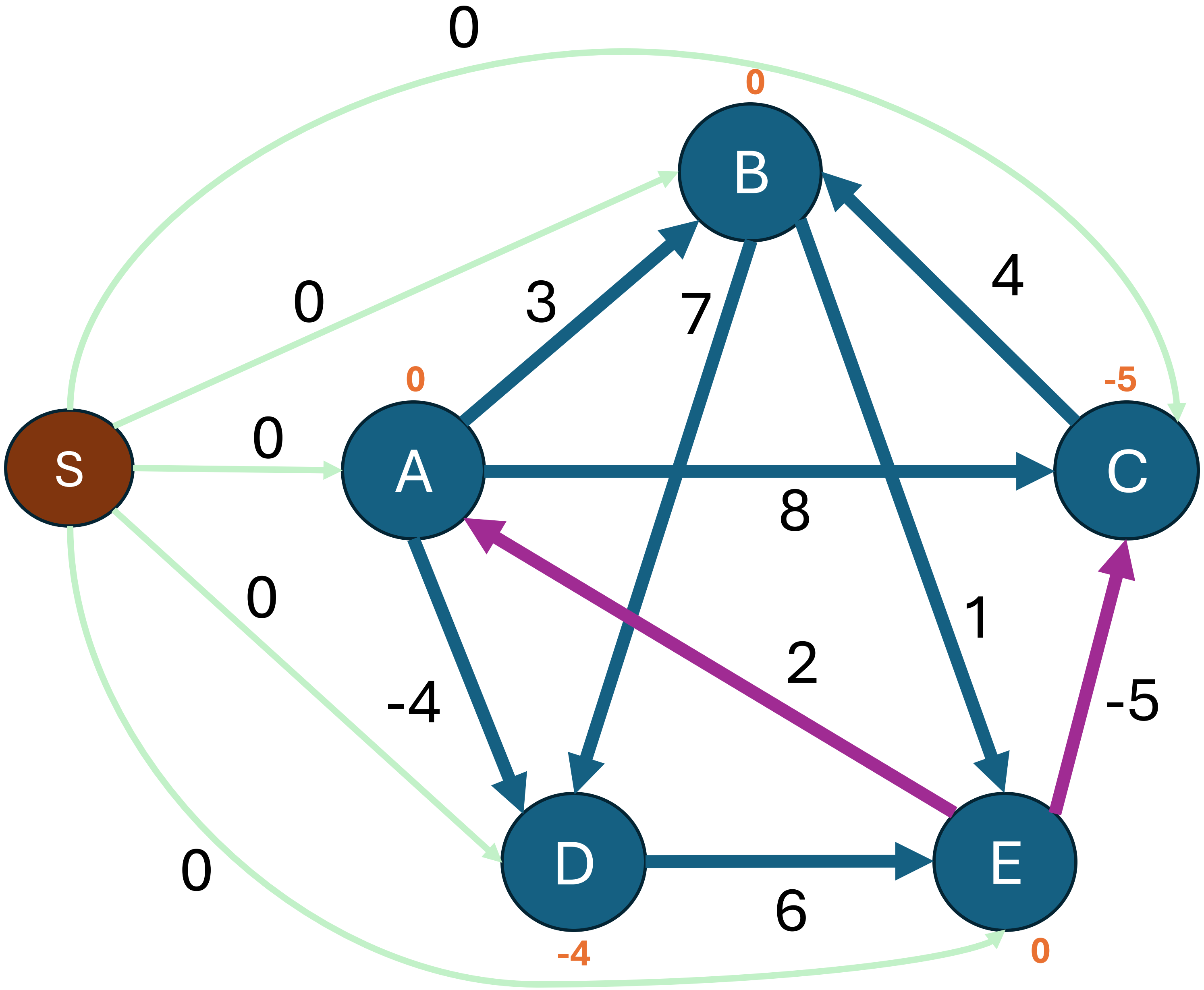
	Cost	Prev
S	0	-
A	0	S
B	0	S
C	0	S
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



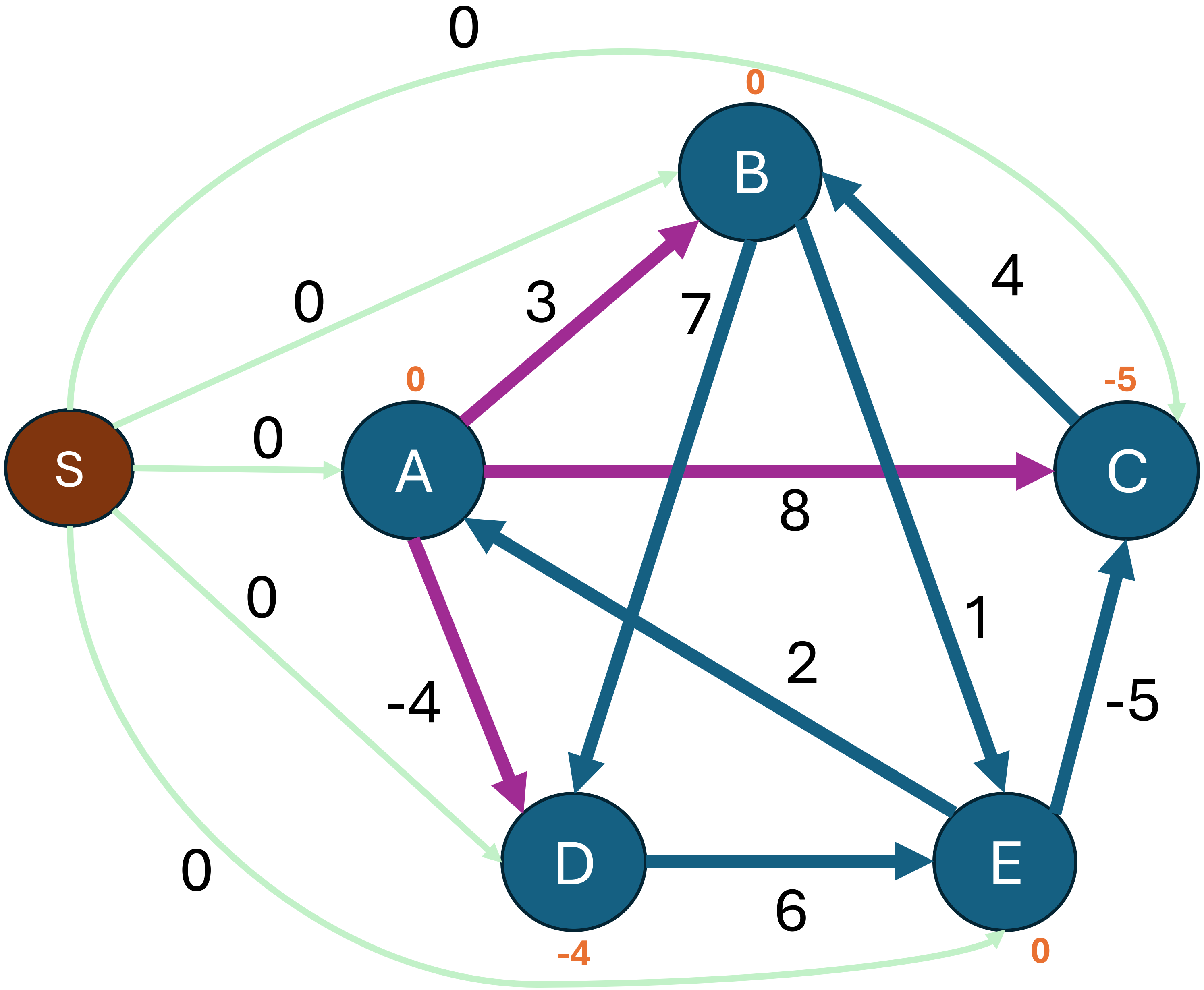
	Cost	Prev
S	0	-
A	0	S
B	0	S
C	0	S
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



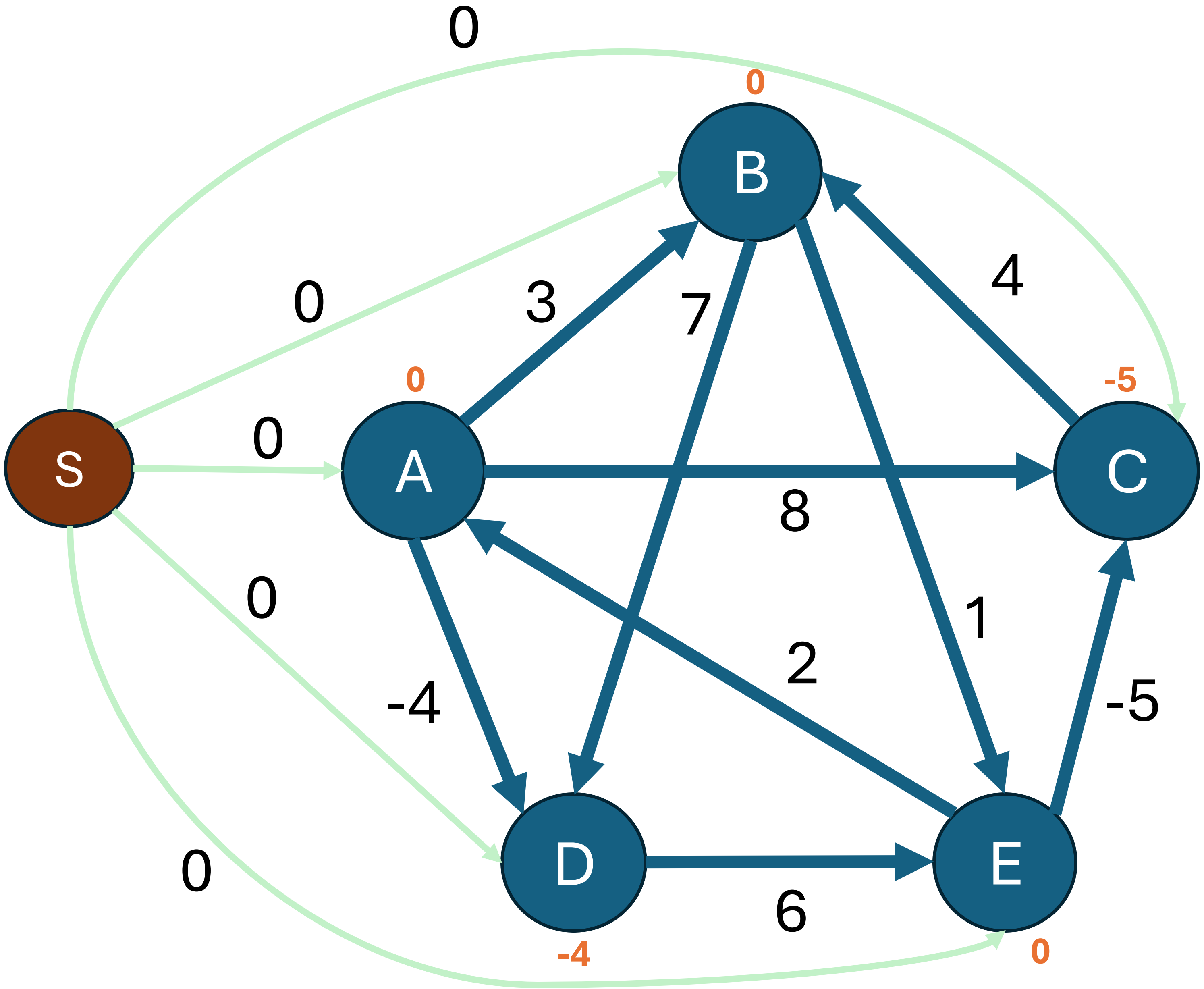
	Cost	Prev
S	0	-
A	0	S
B	0	S
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



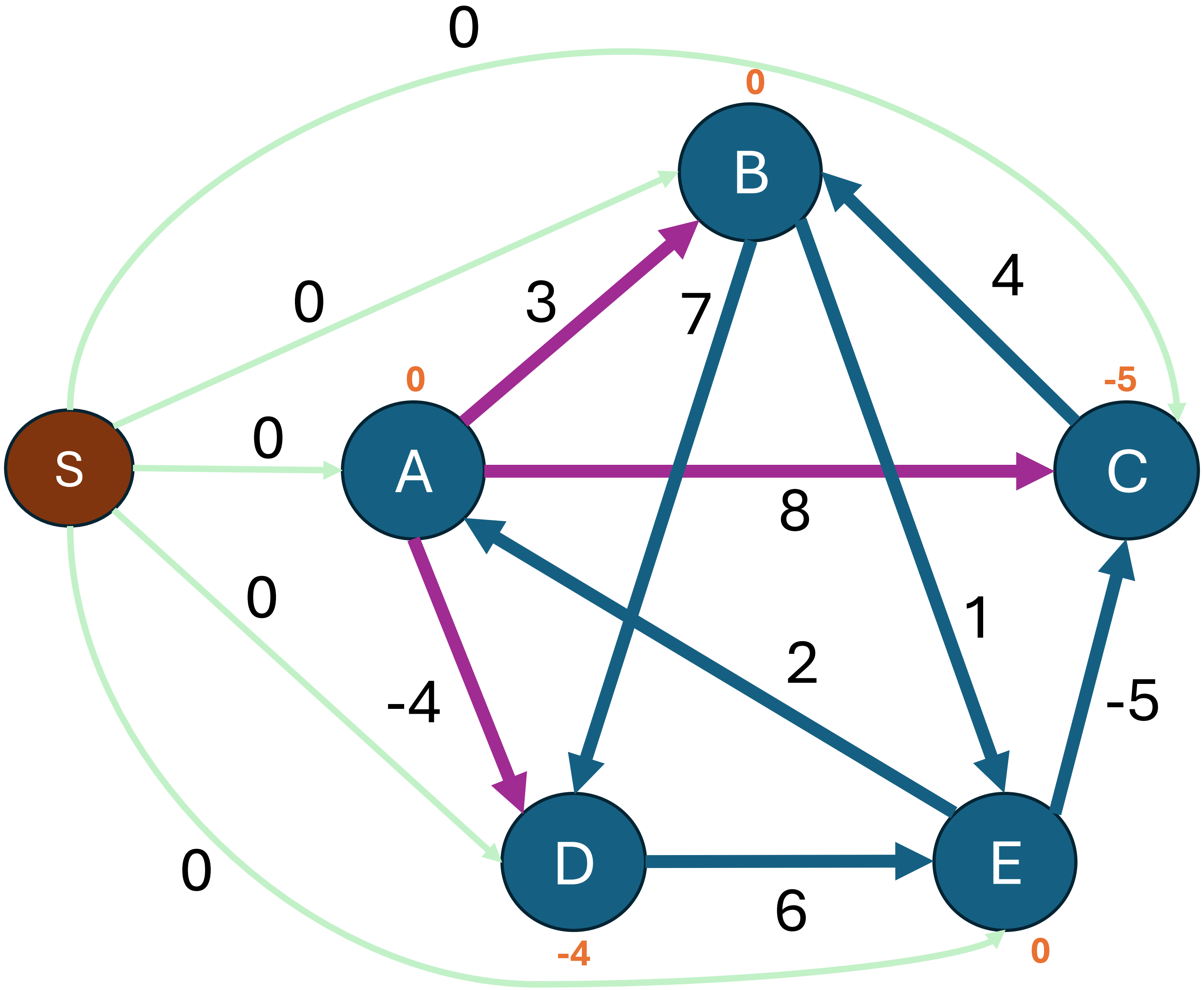
	Cost	Prev
S	0	-
A	0	S
B	0	S
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



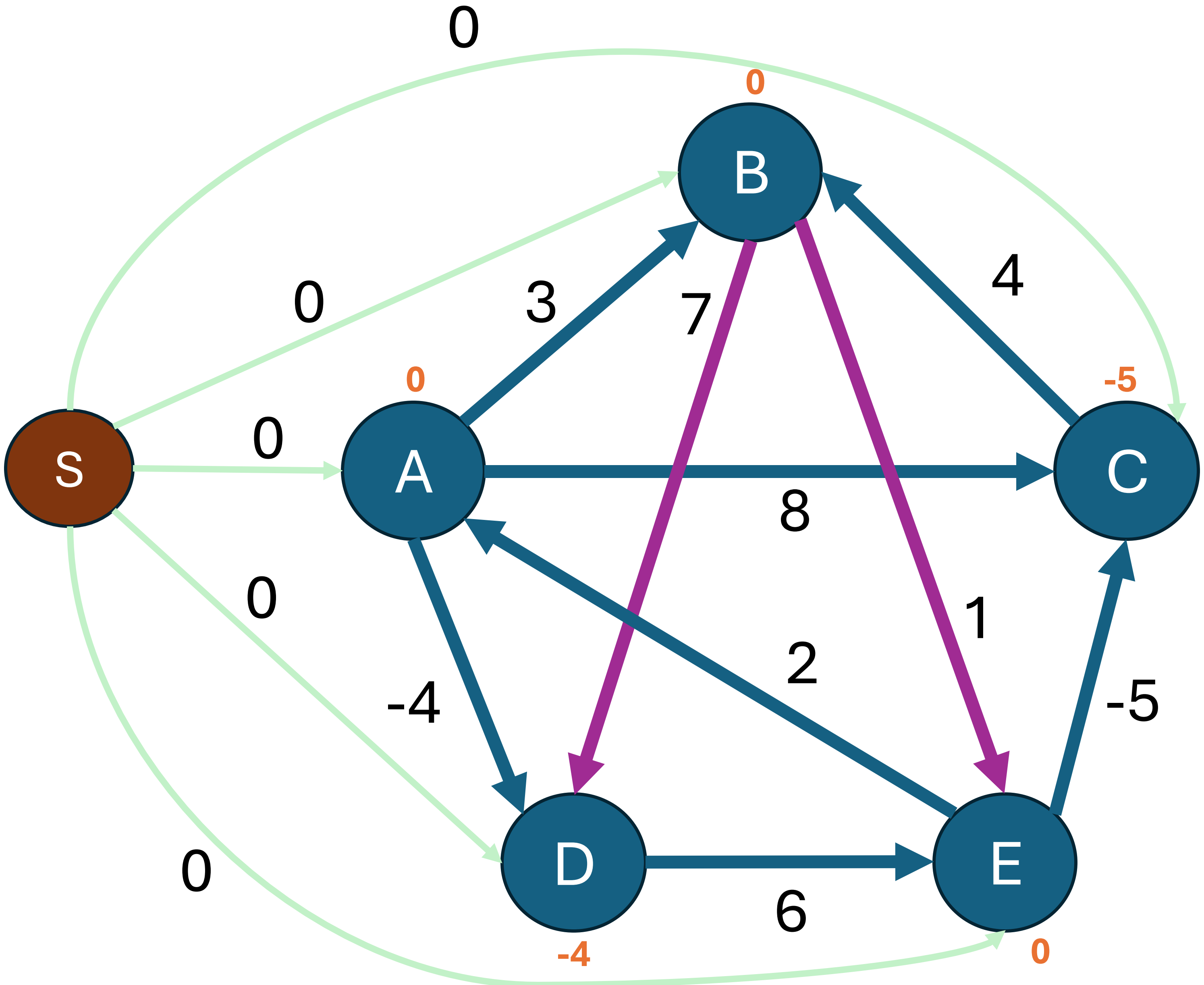
	Cost	Prev
S	0	-
A	0	S
B	0	S
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



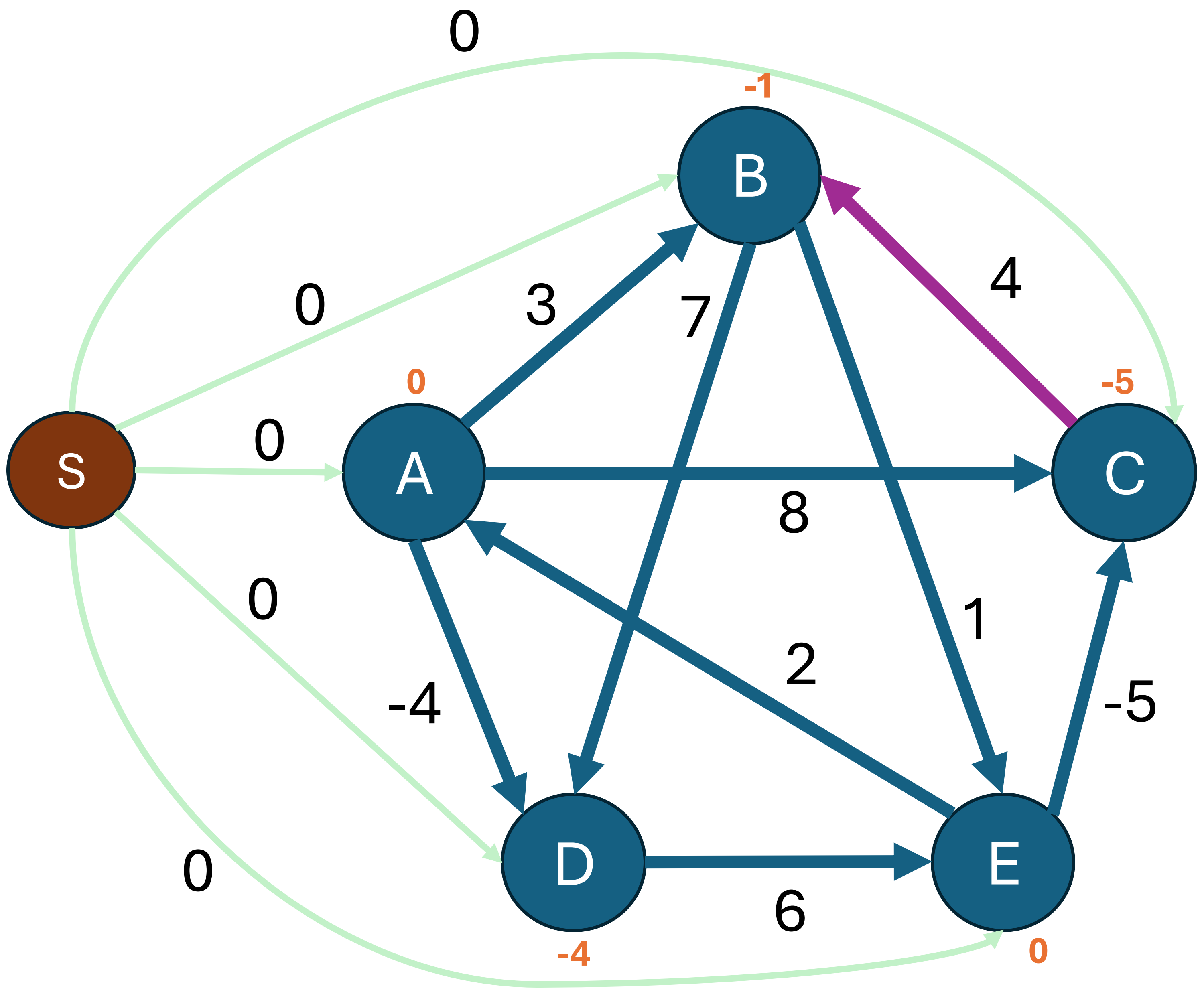
	Cost	Prev
S	0	-
A	0	S
B	0	S
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



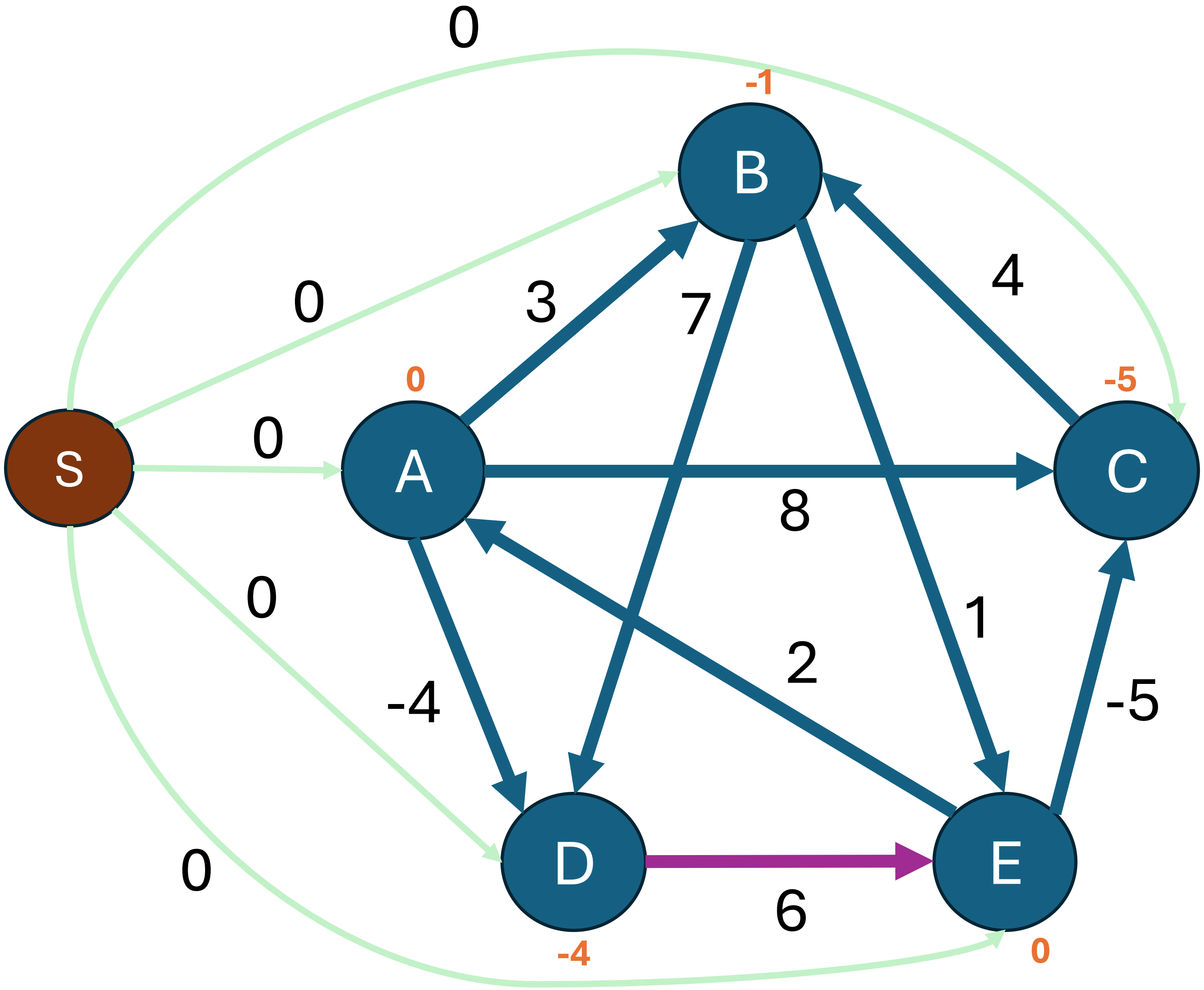
	Cost	Prev
S	0	-
A	0	S
B	0	S
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



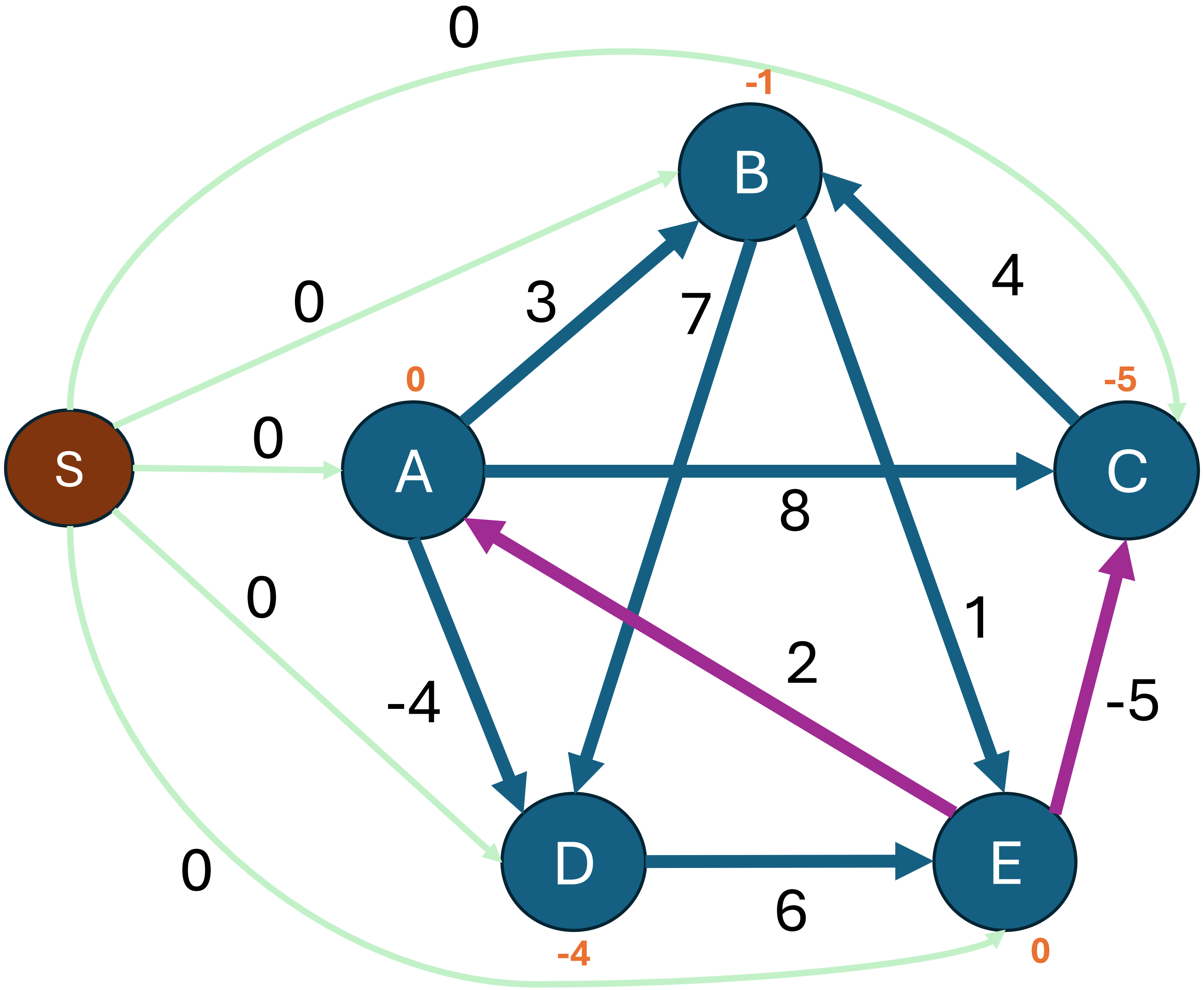
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



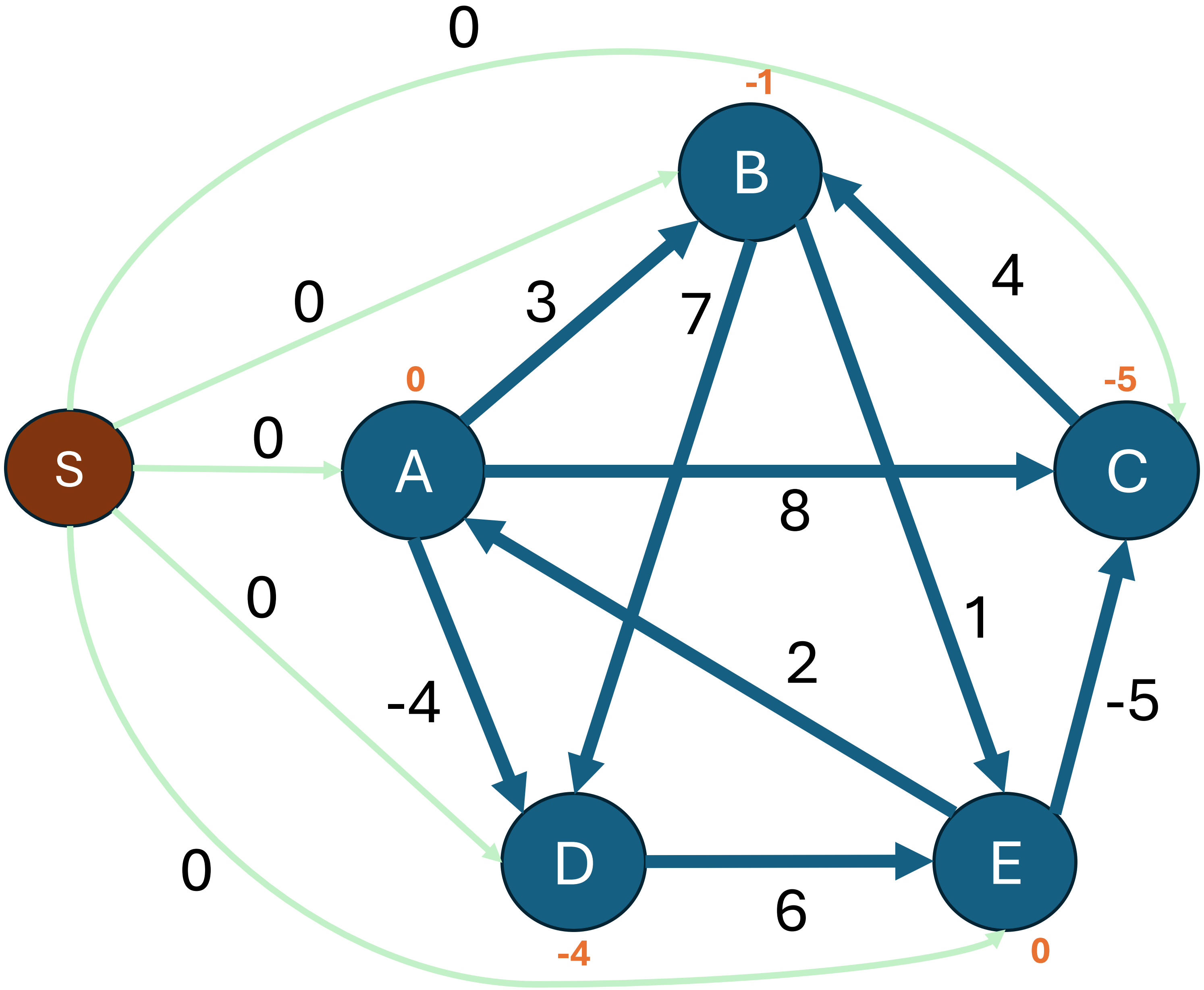
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



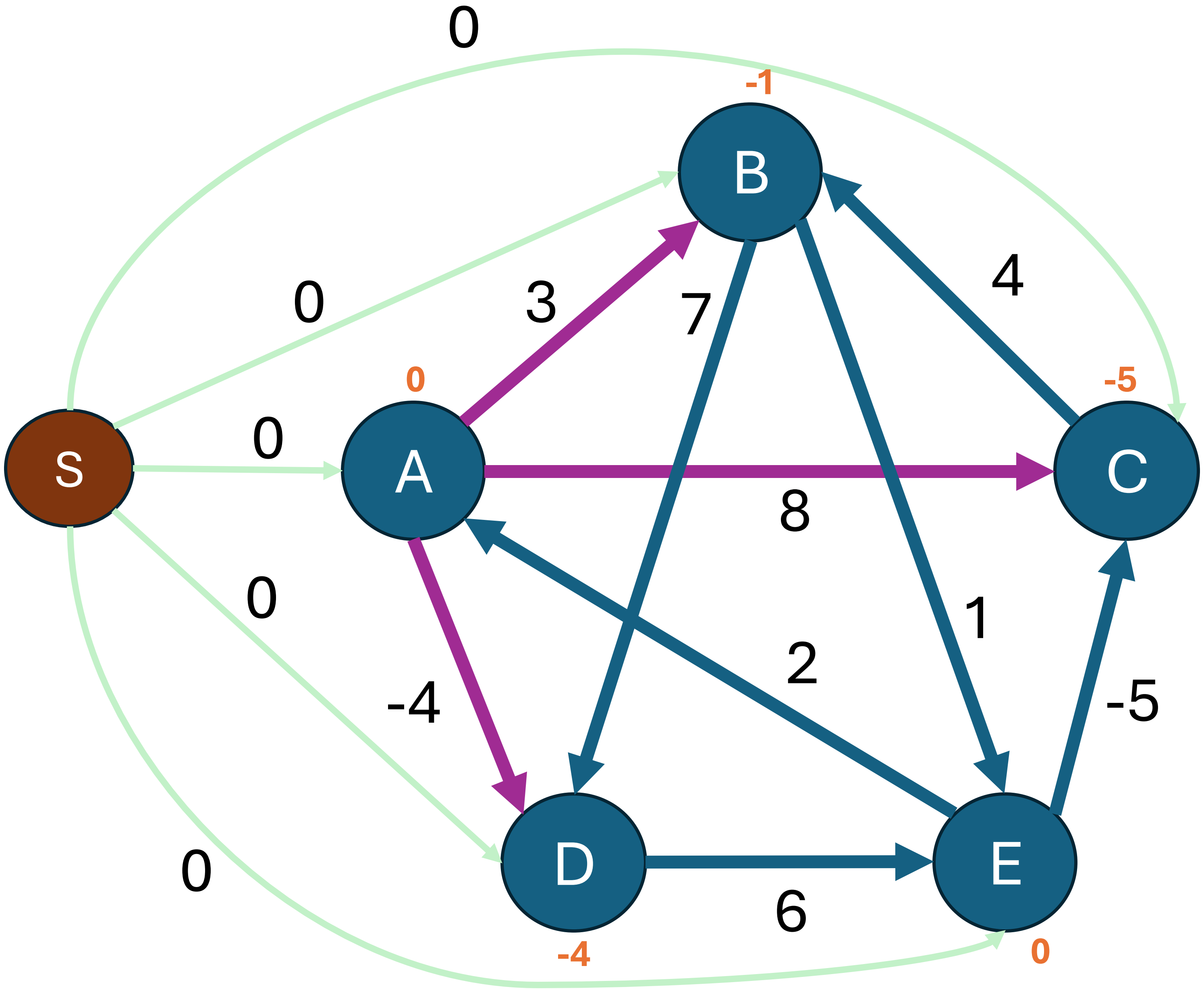
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



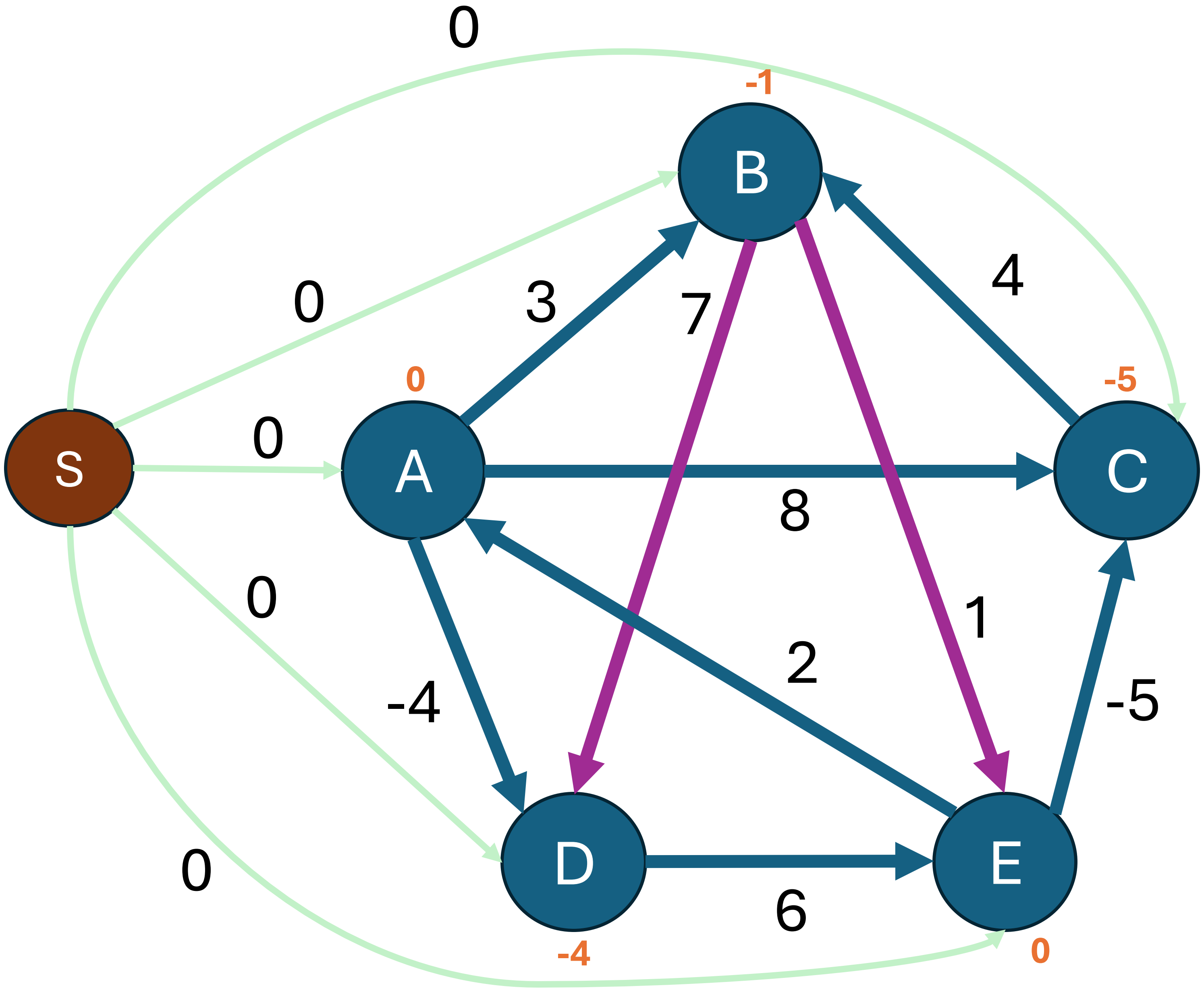
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



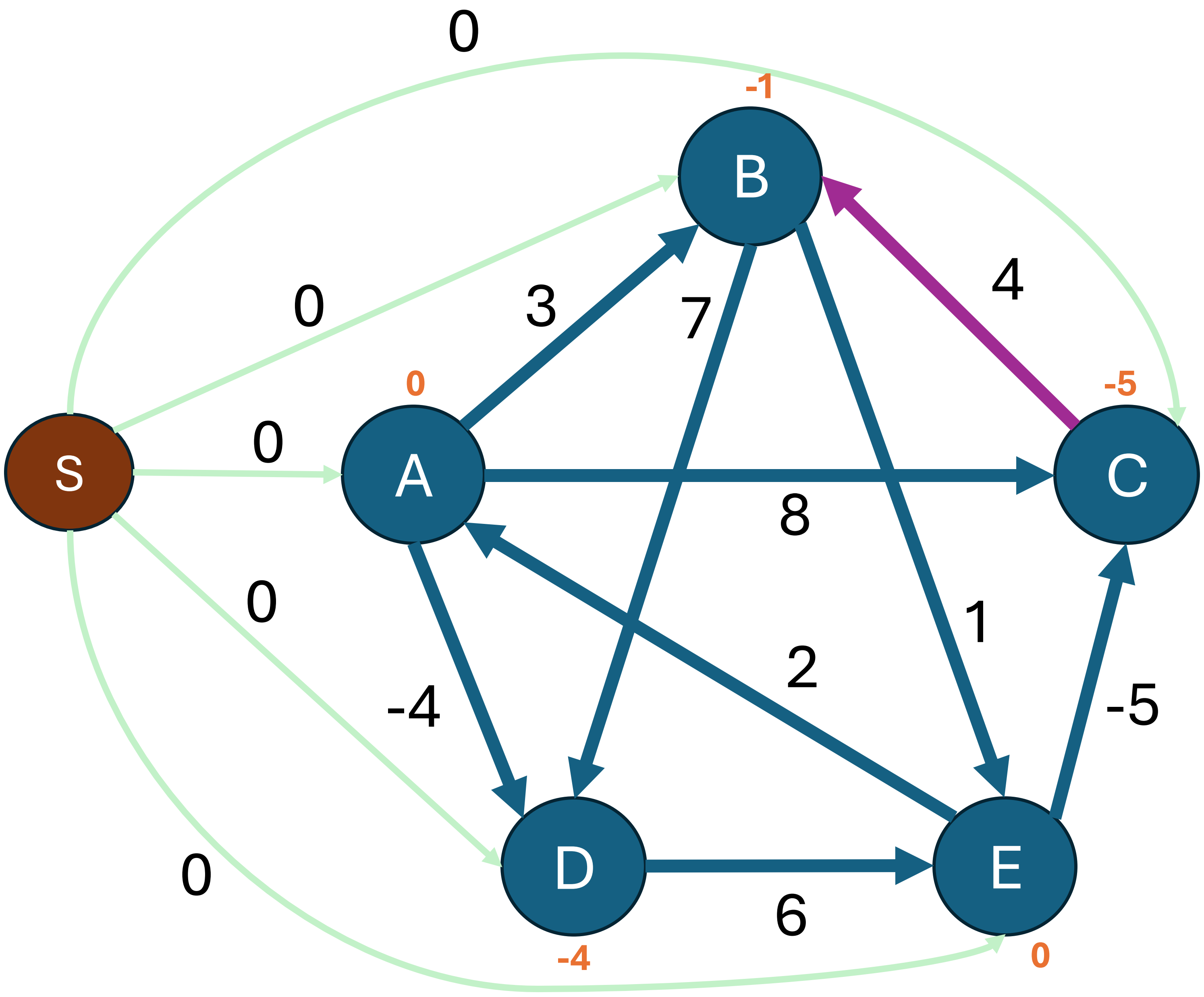
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



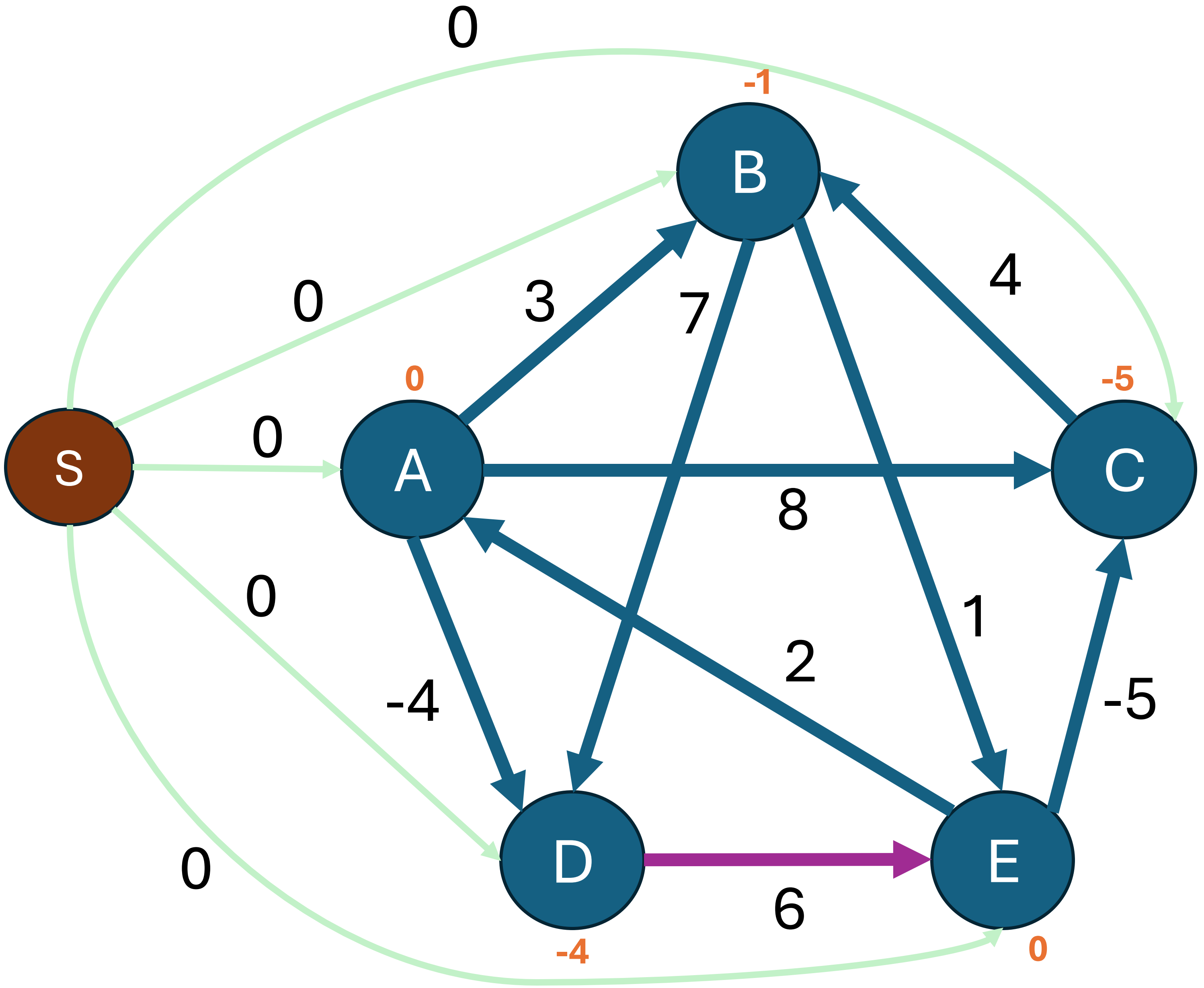
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



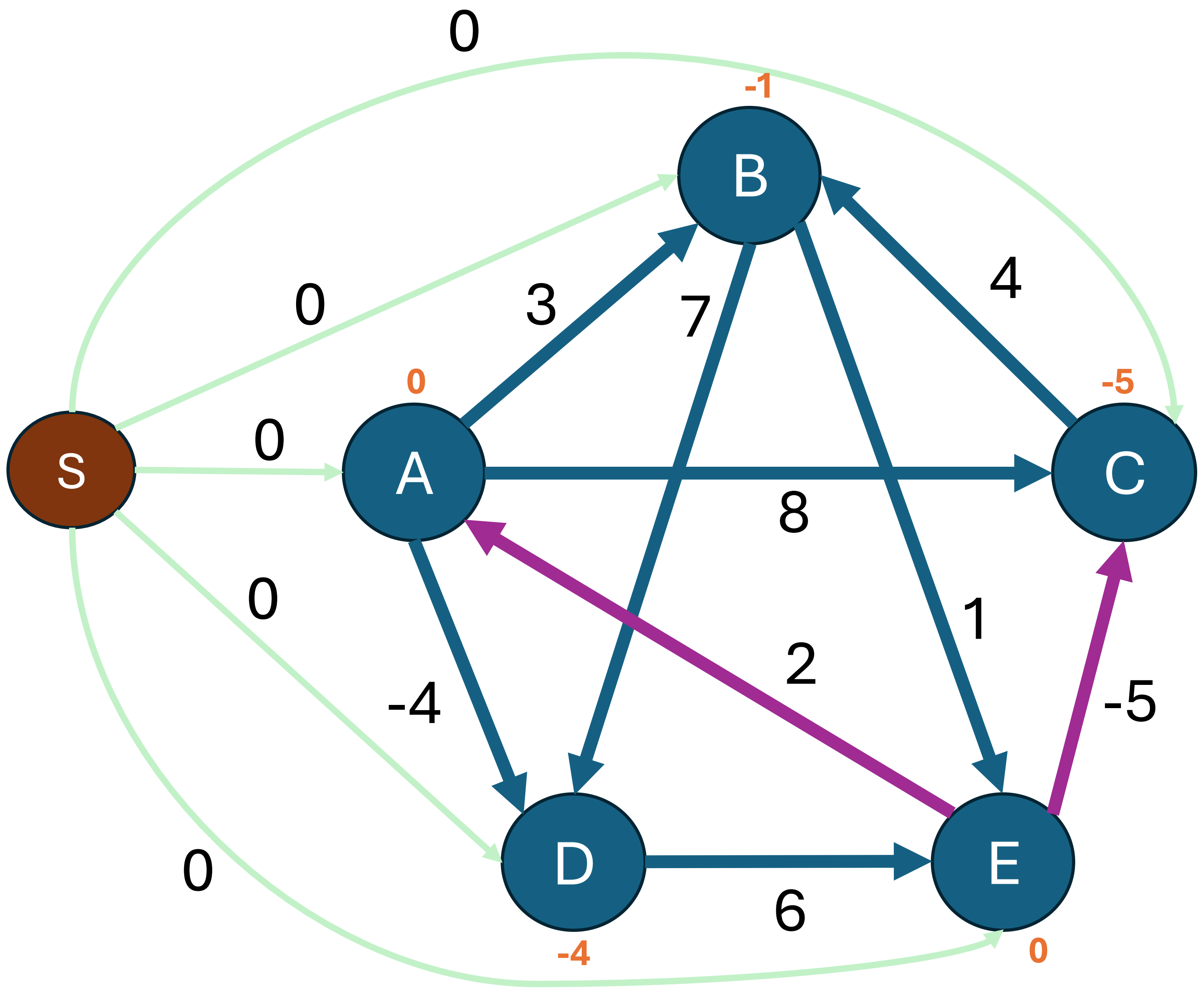
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

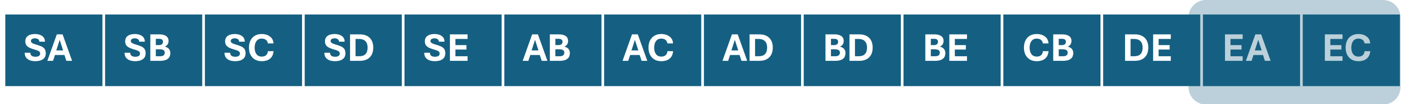
SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



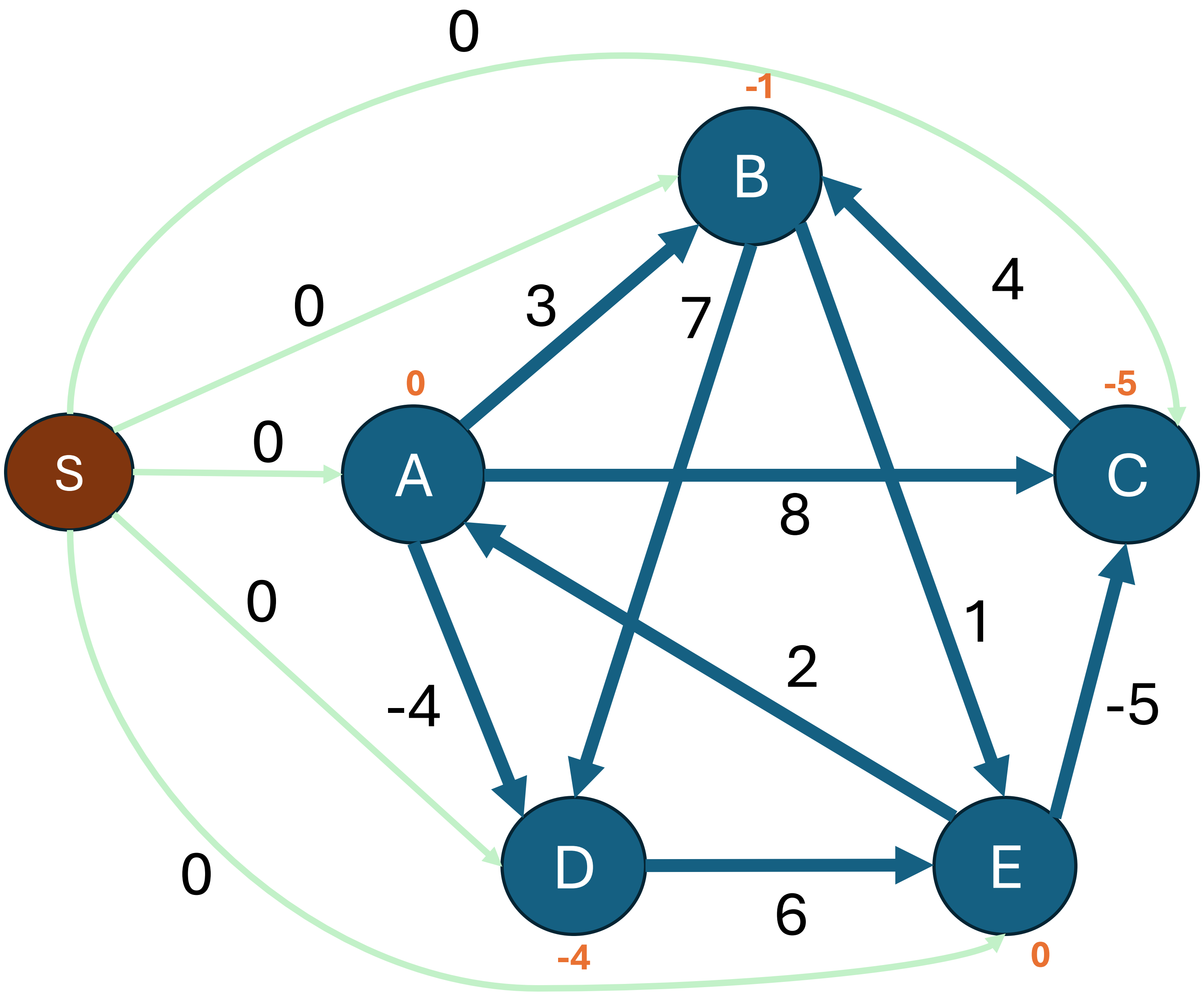
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



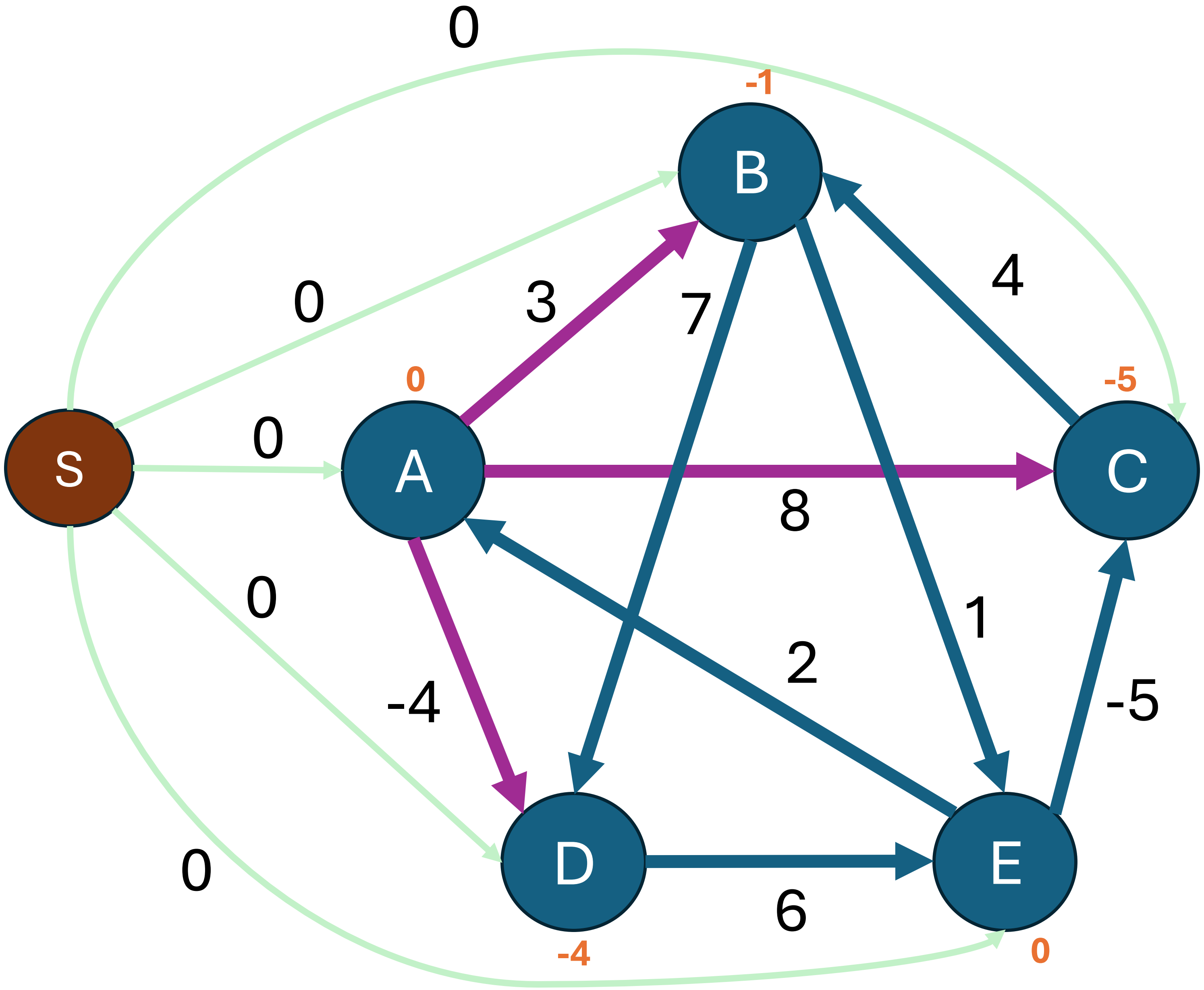


	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S



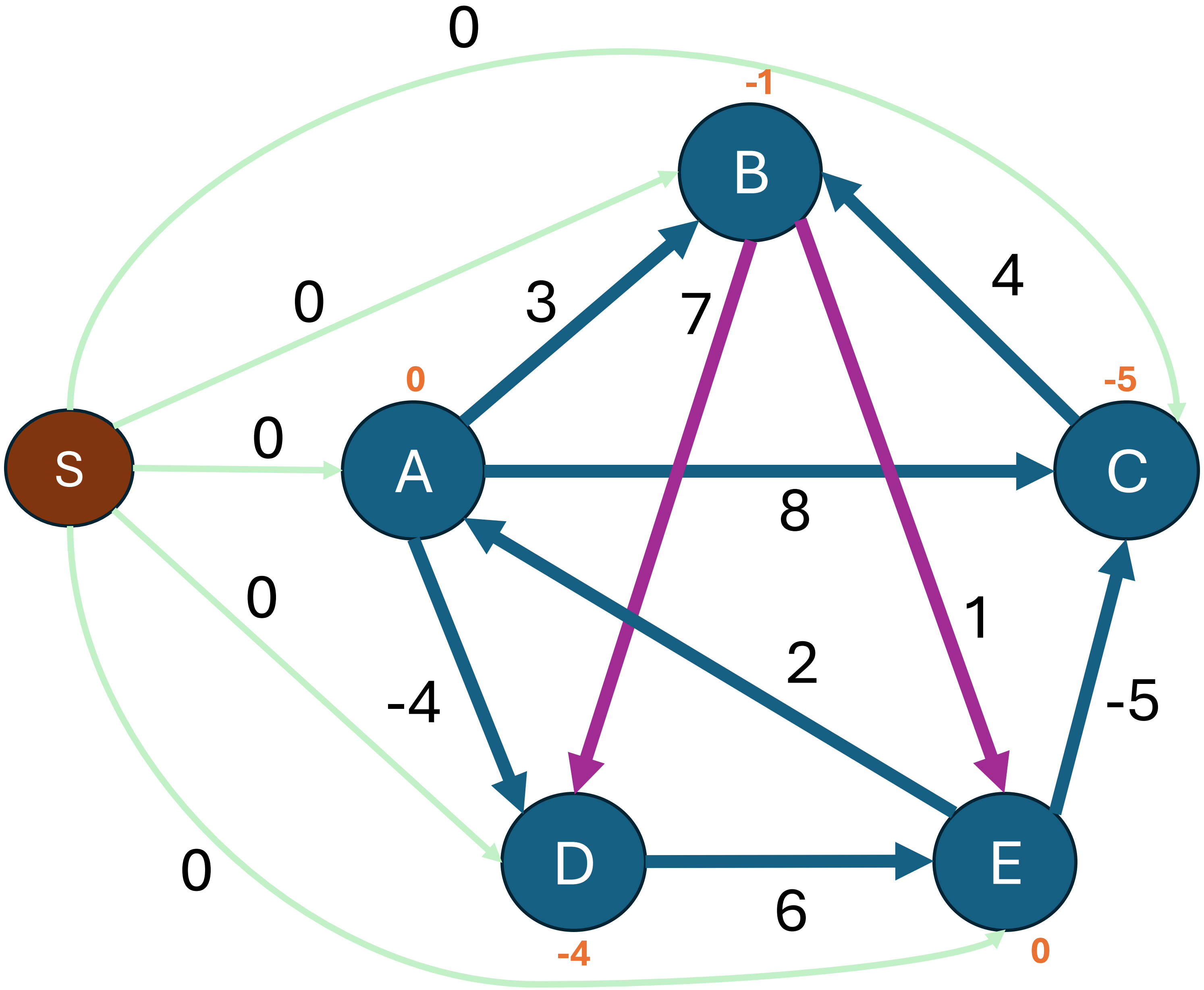
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



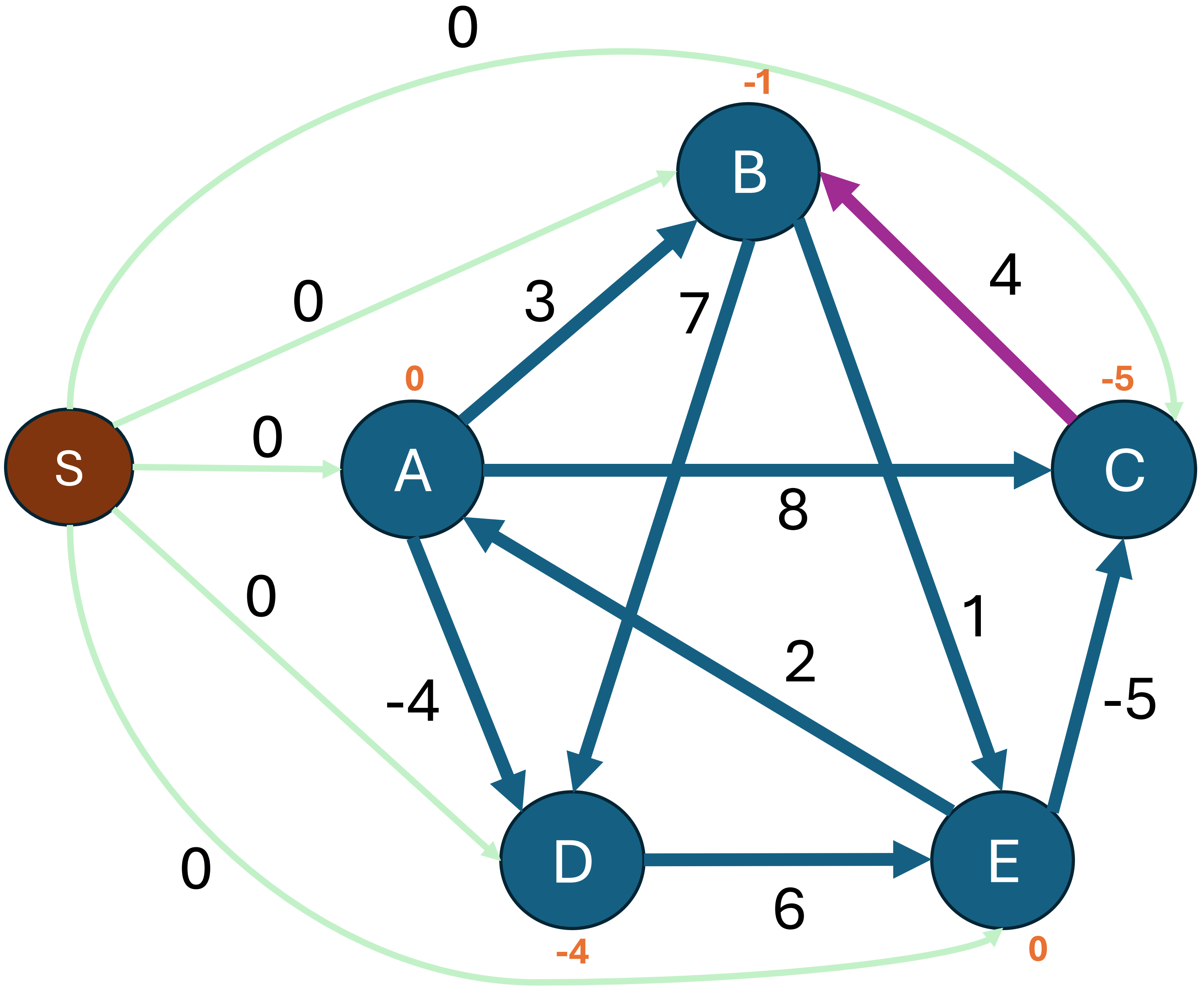
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



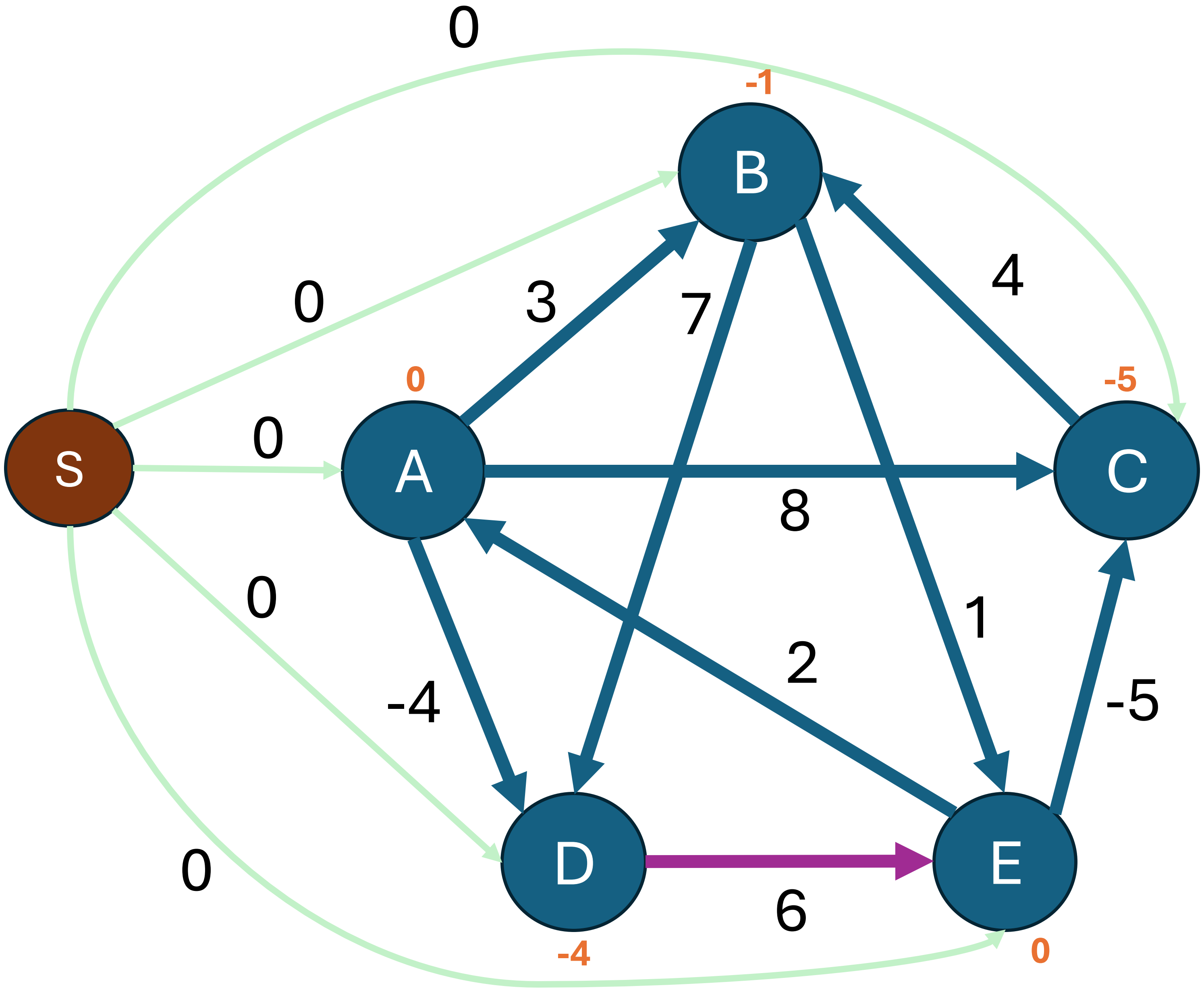
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



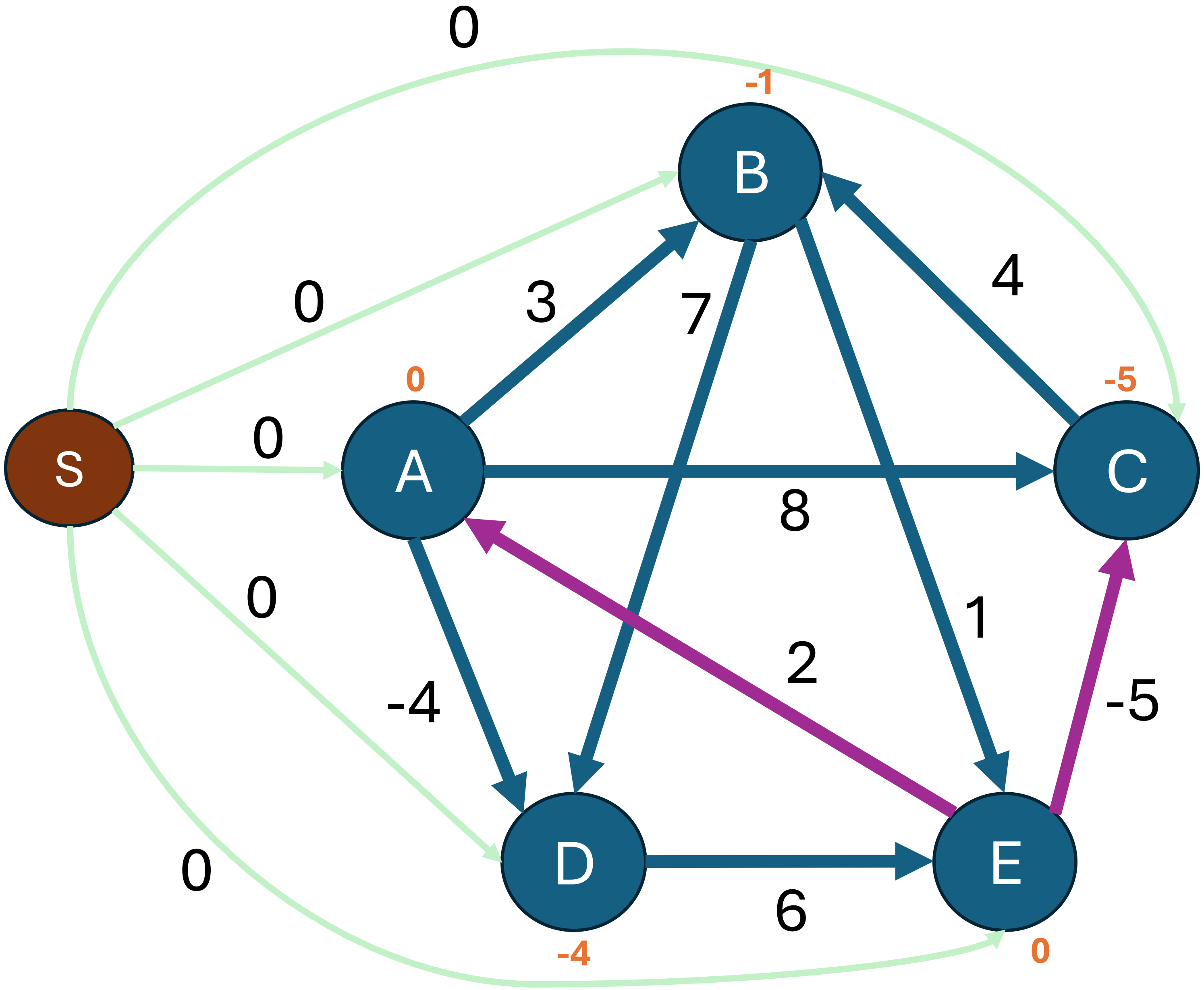
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



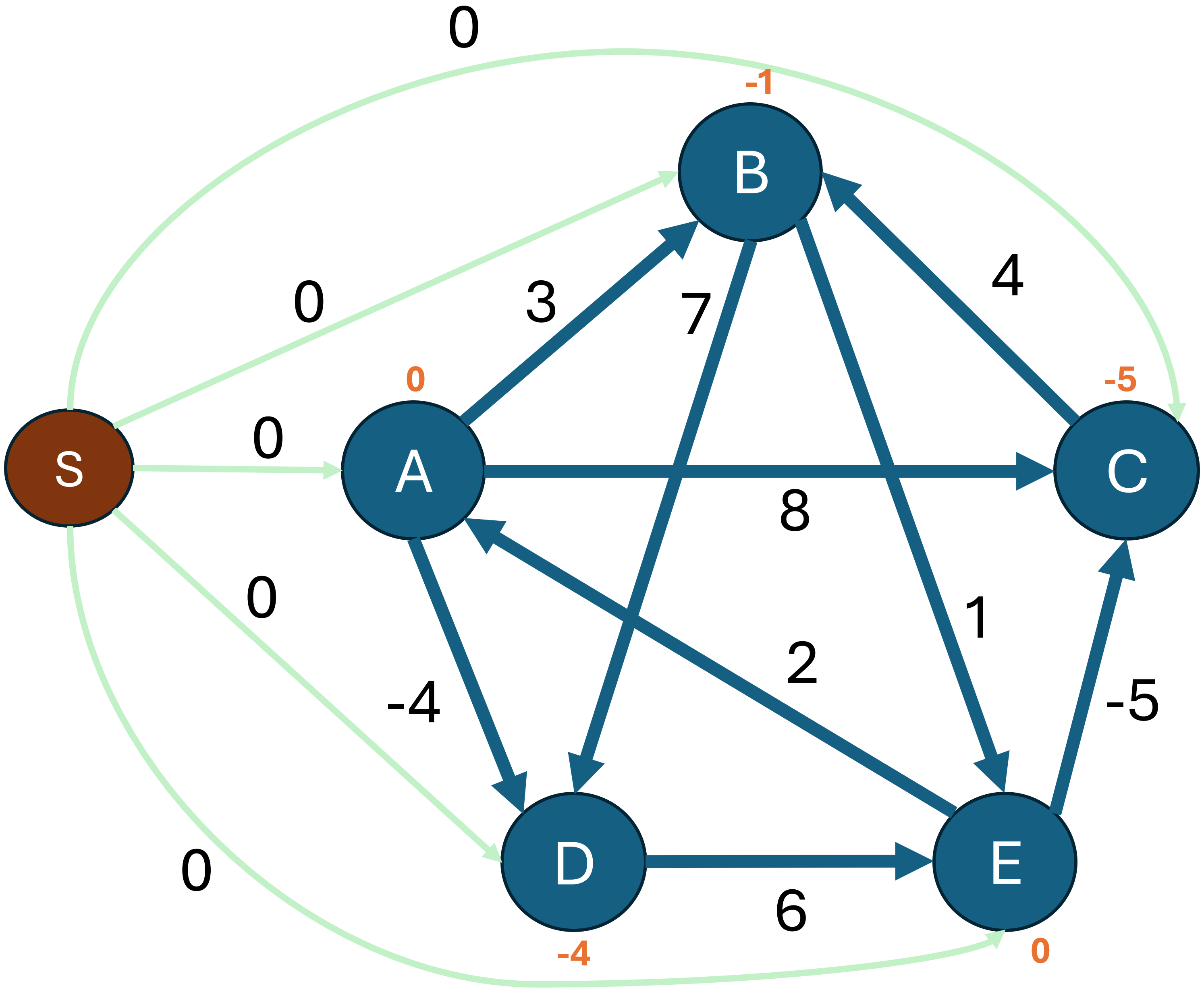
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



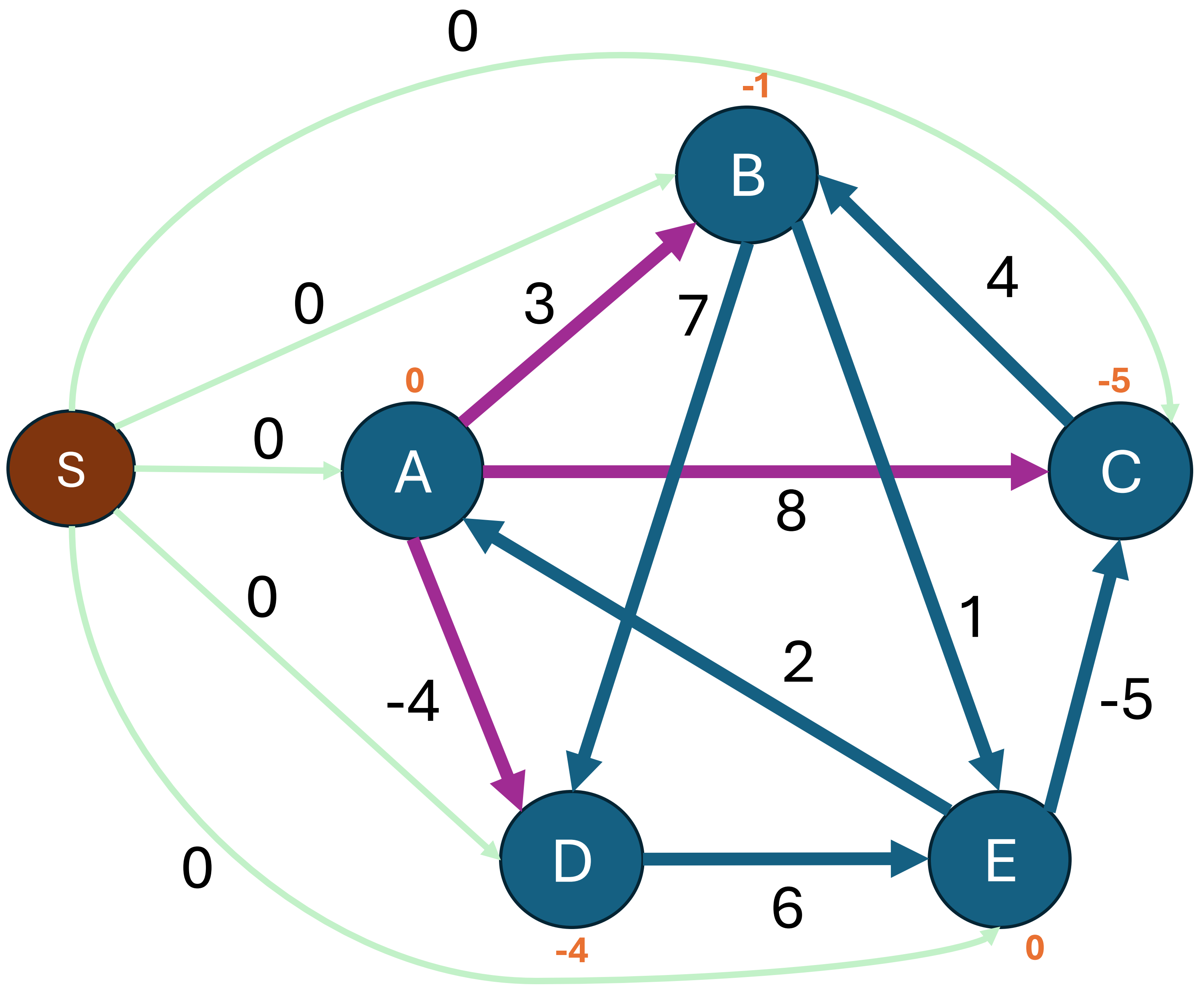
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



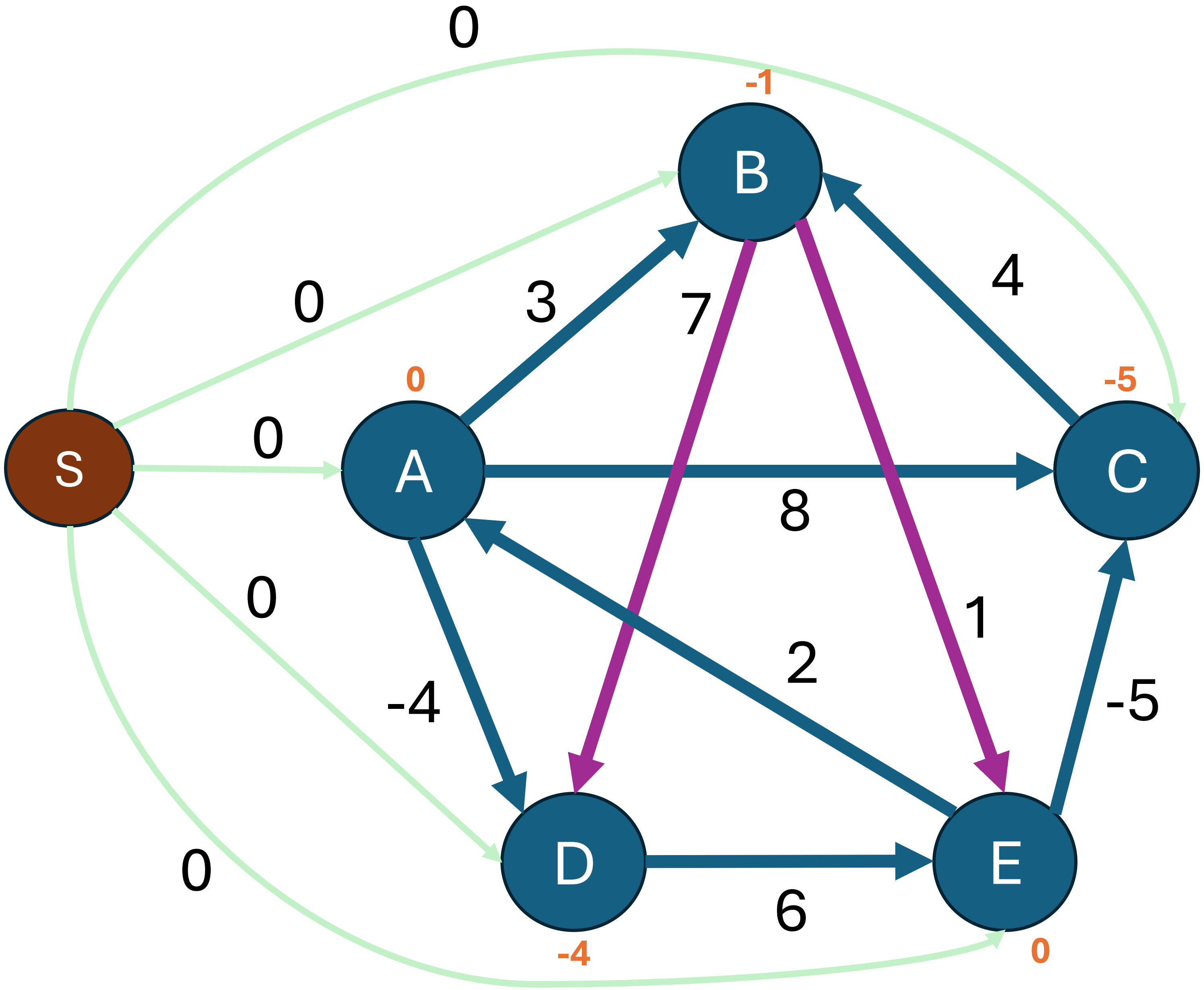
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



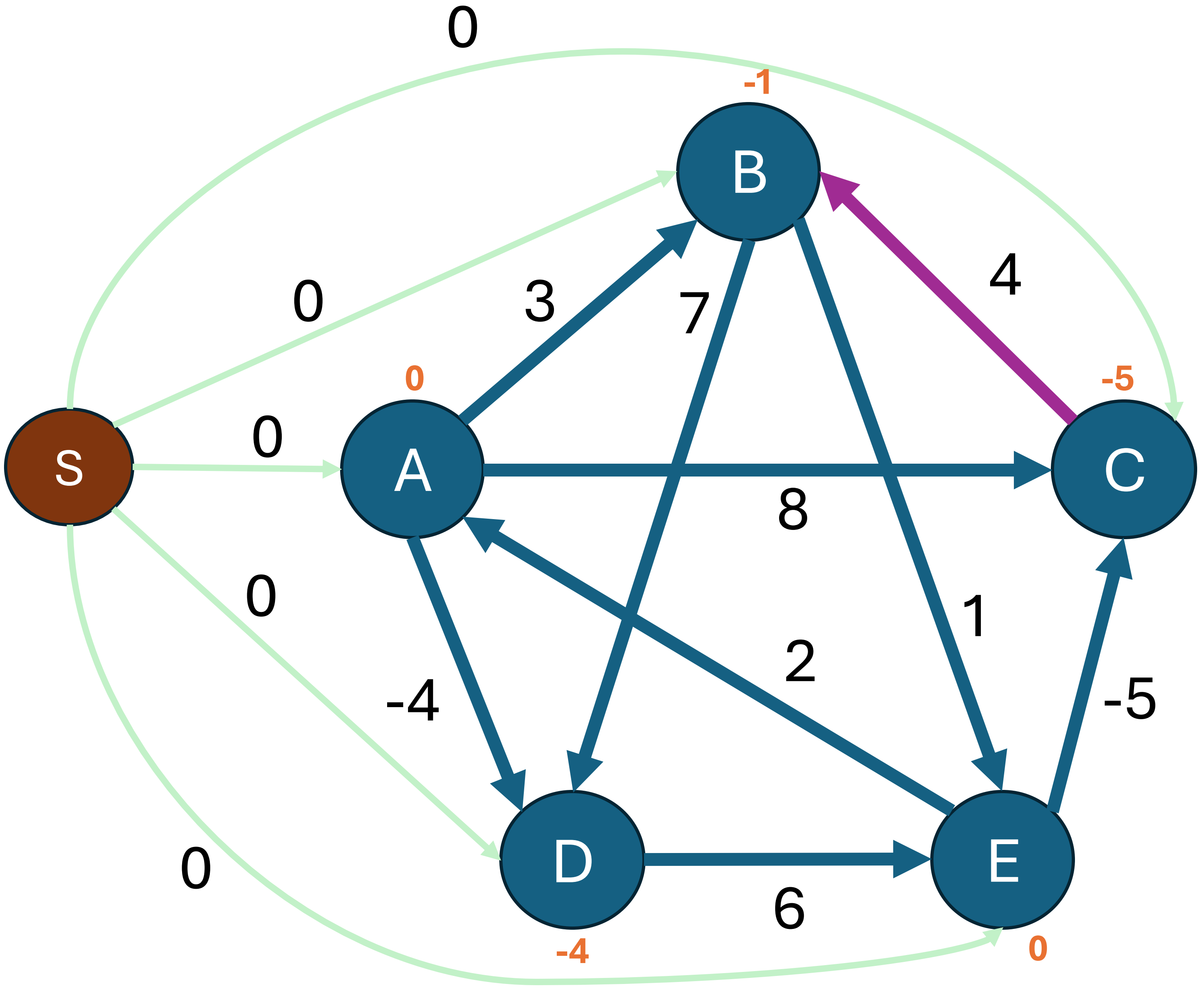
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



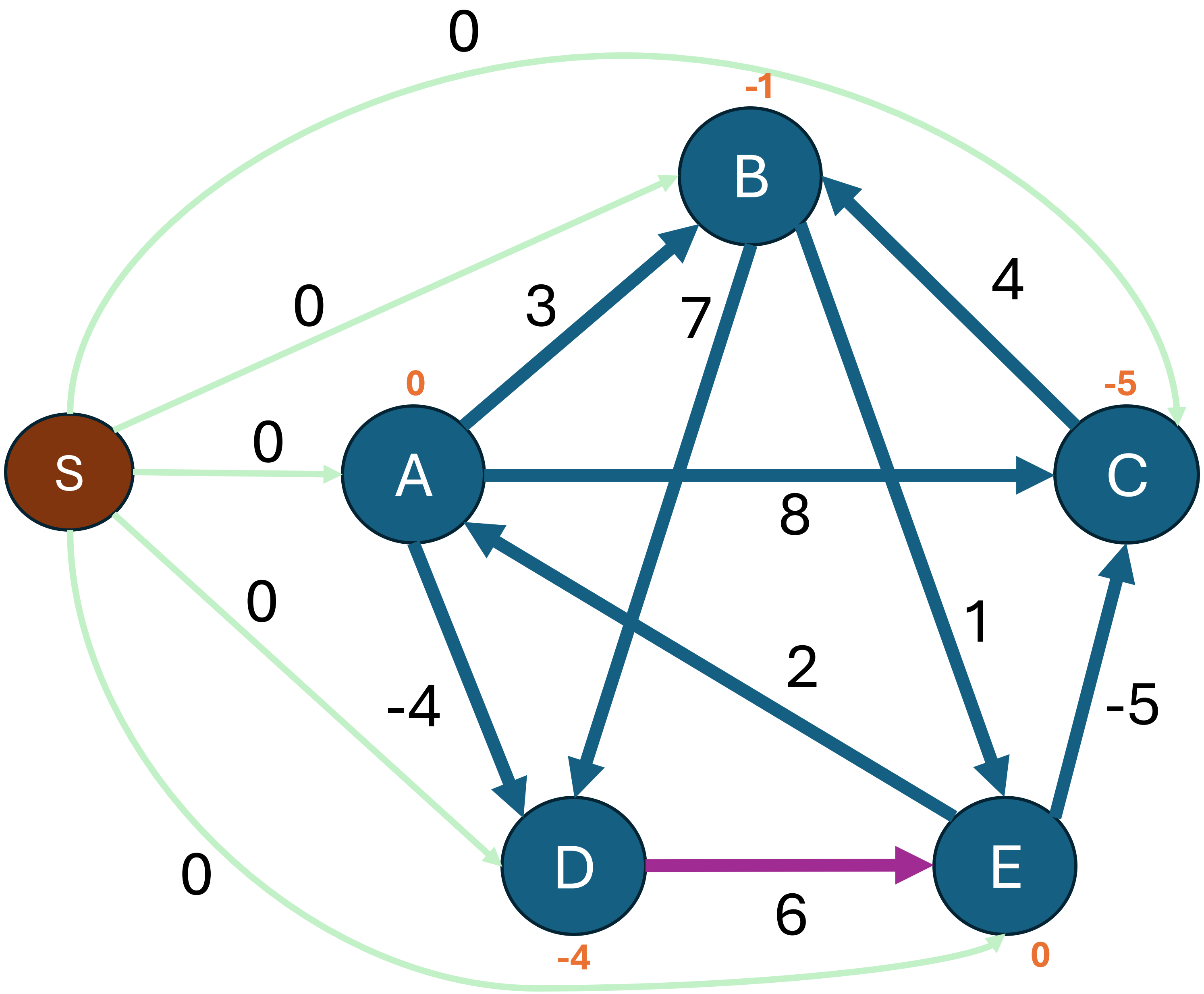
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



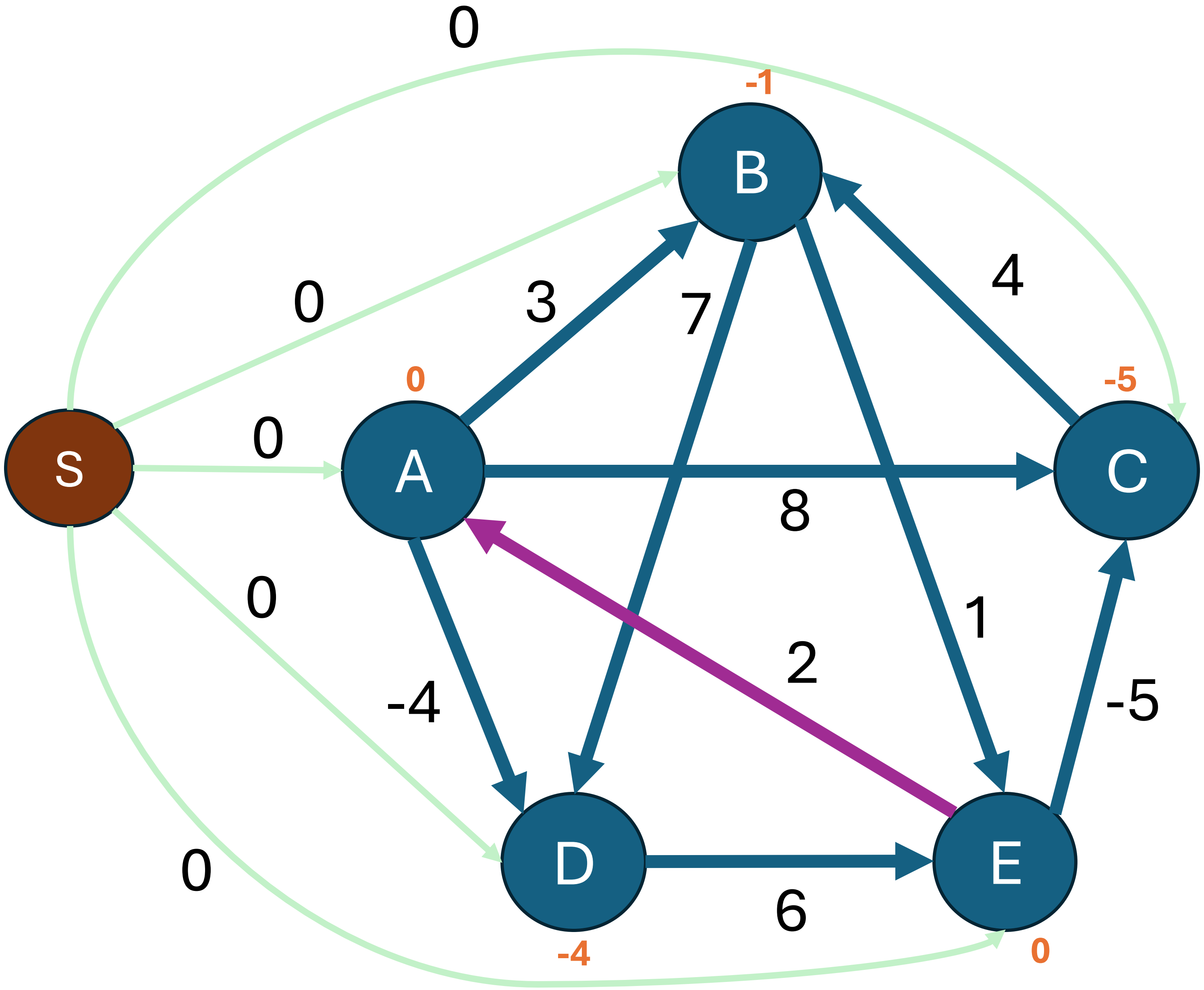
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



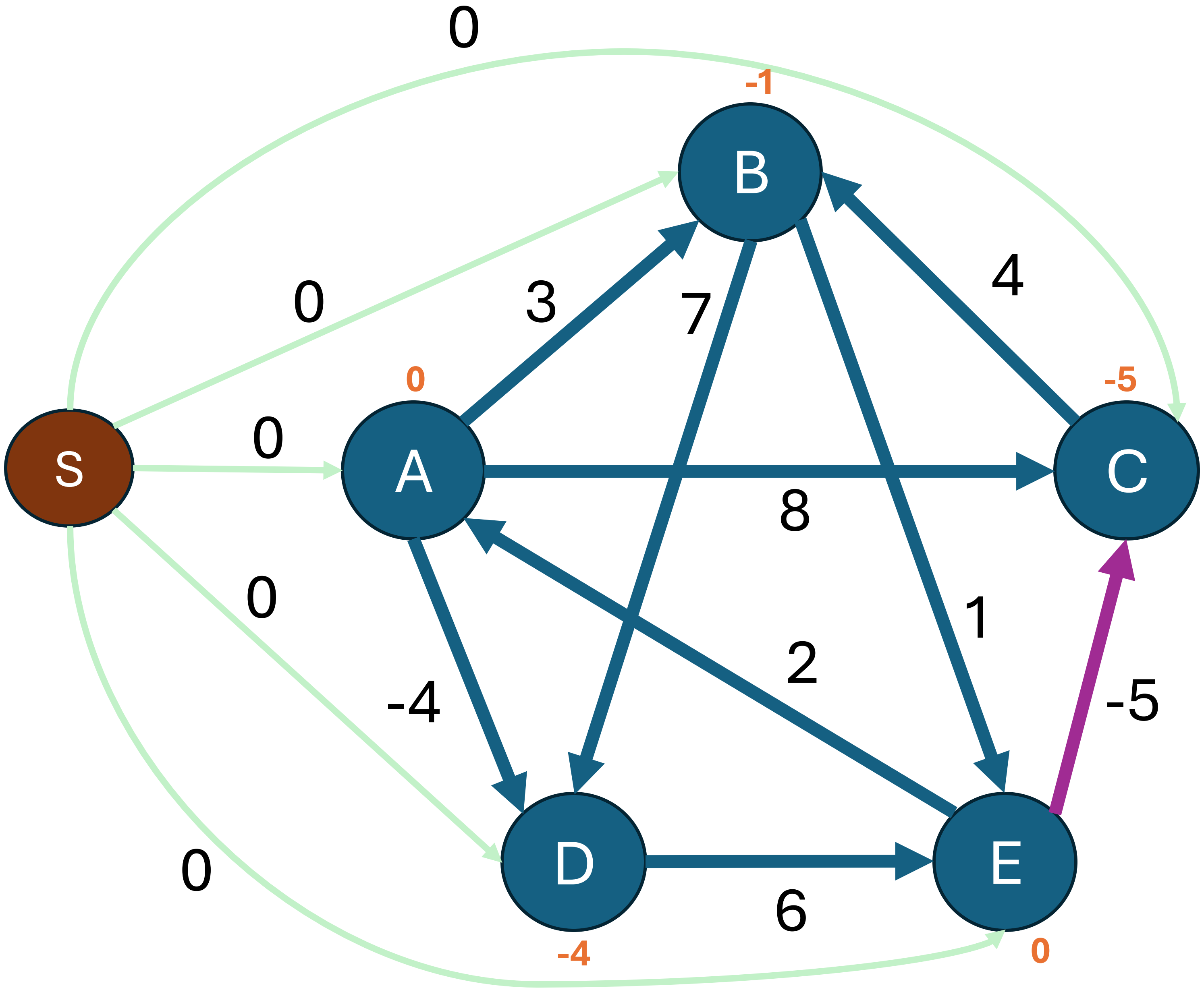
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



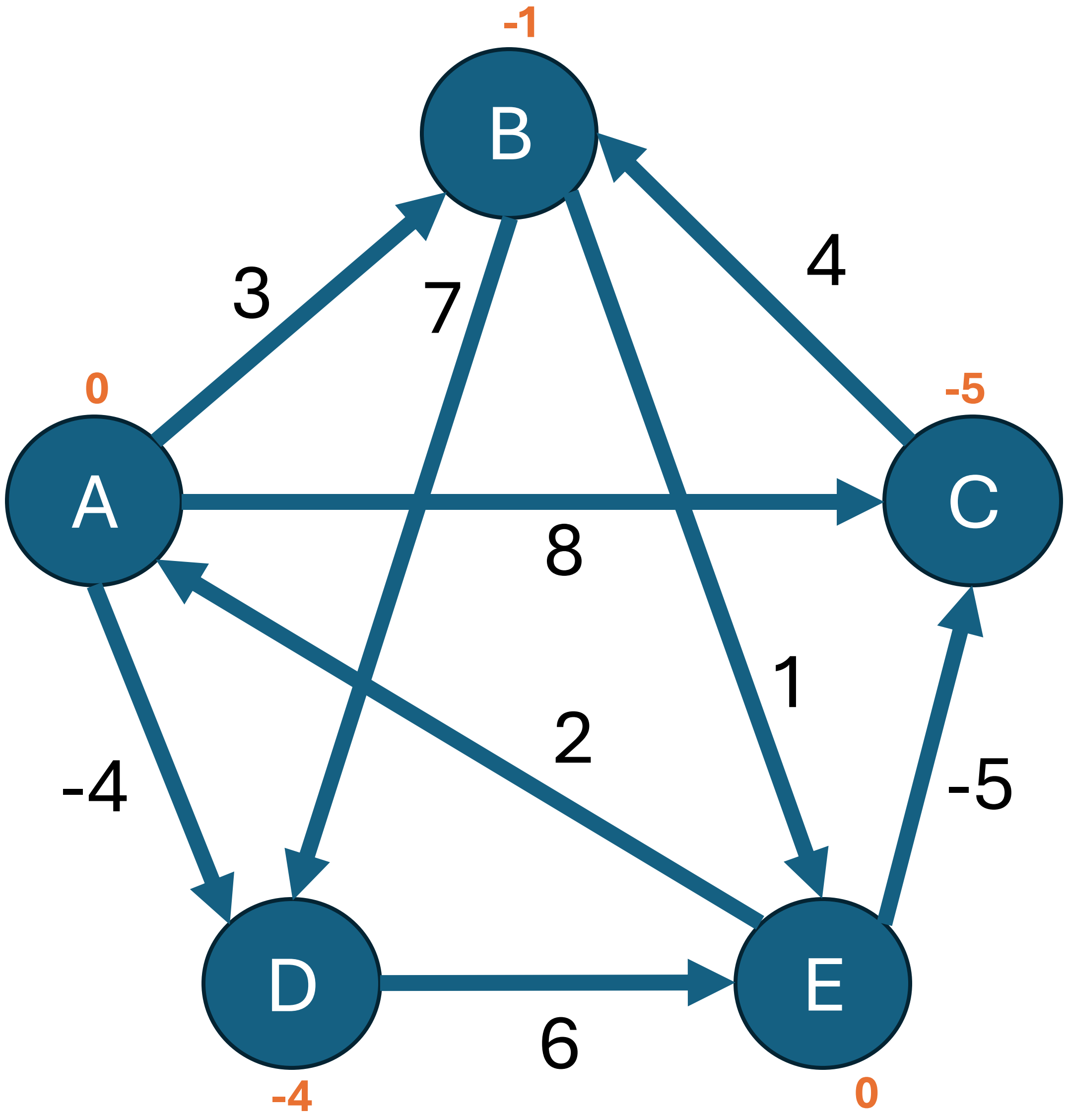
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



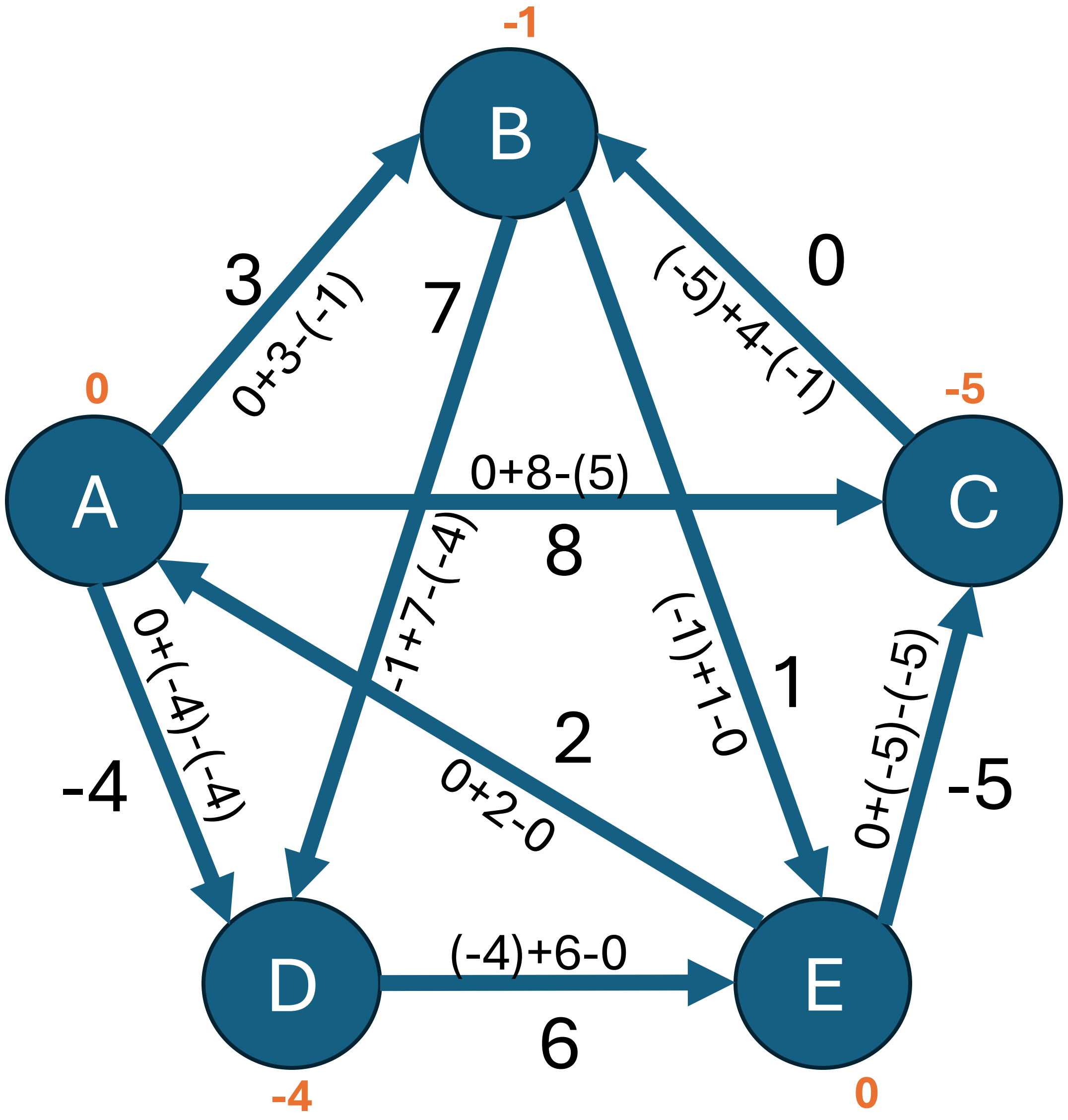
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC



	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

SA	SB	SC	SD	SE	AB	AC	AD	BD	BE	CB	DE	EA	EC

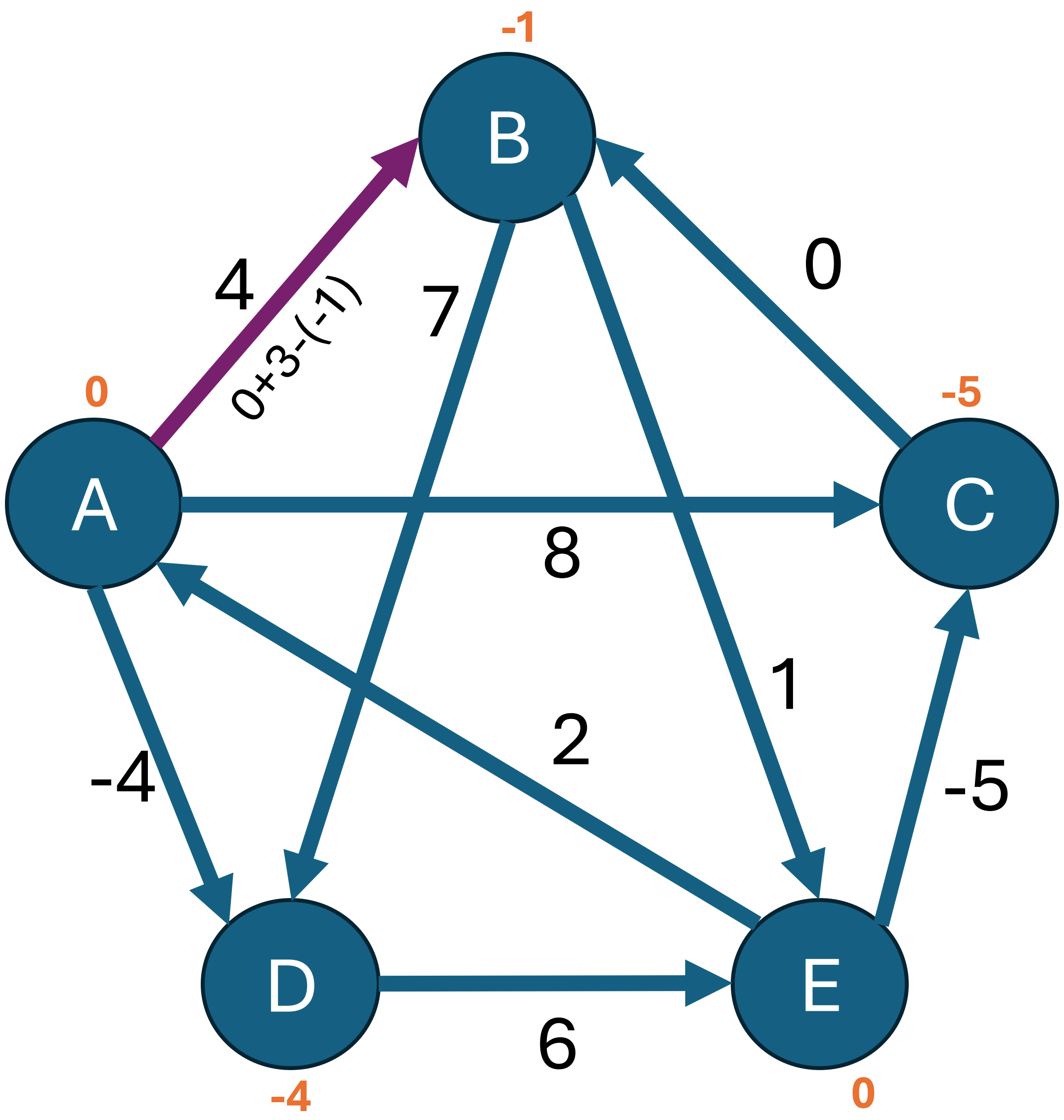


	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

$$w_{new}(u,v) = w(u,v) + h(u) - h(v) >= 0$$

AB	AC	AD	BD	BE	CB	DE	EA	EC

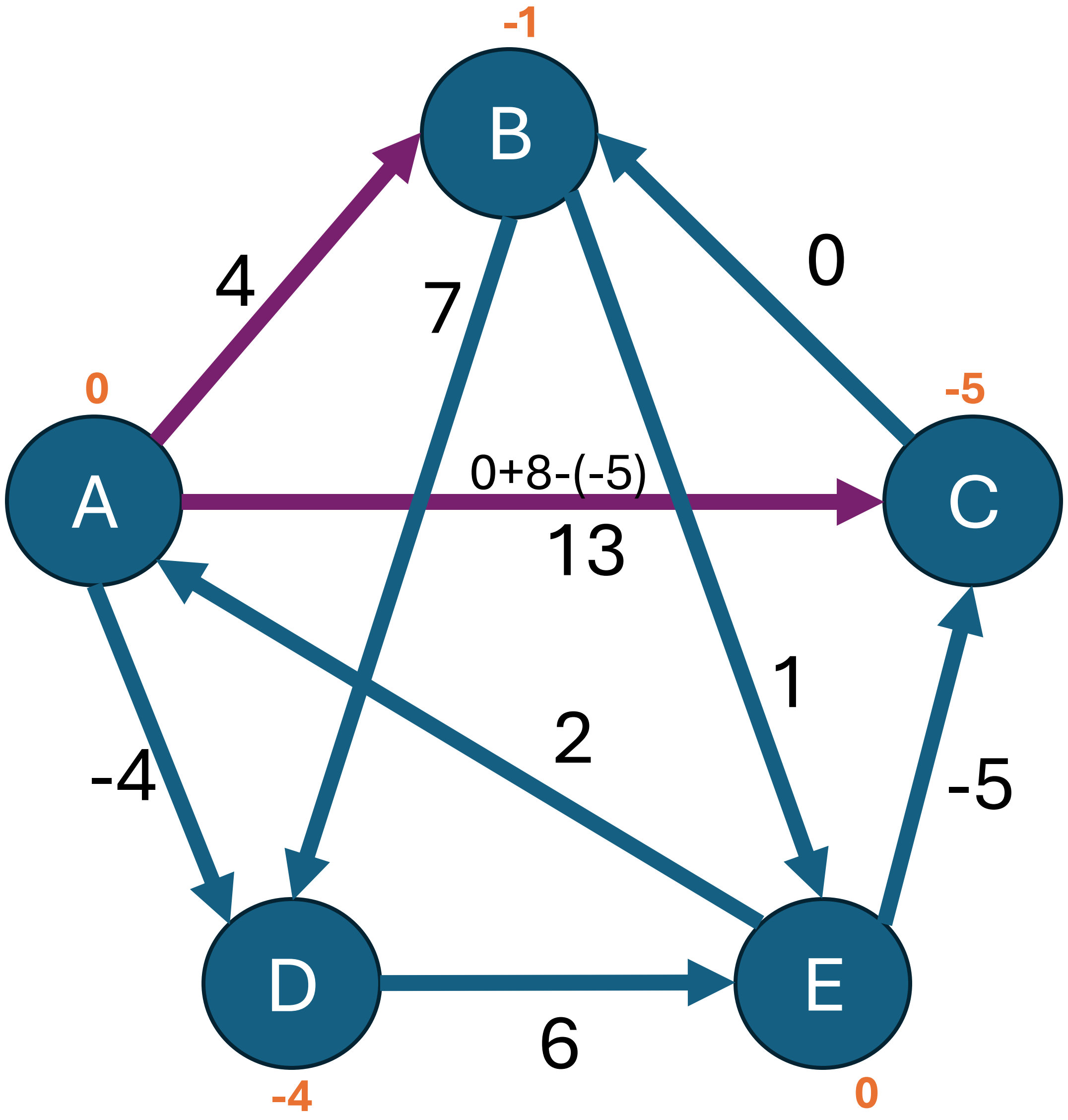
3



	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

AB	AC	AD	BD	BE	CB	DE	EA	EC

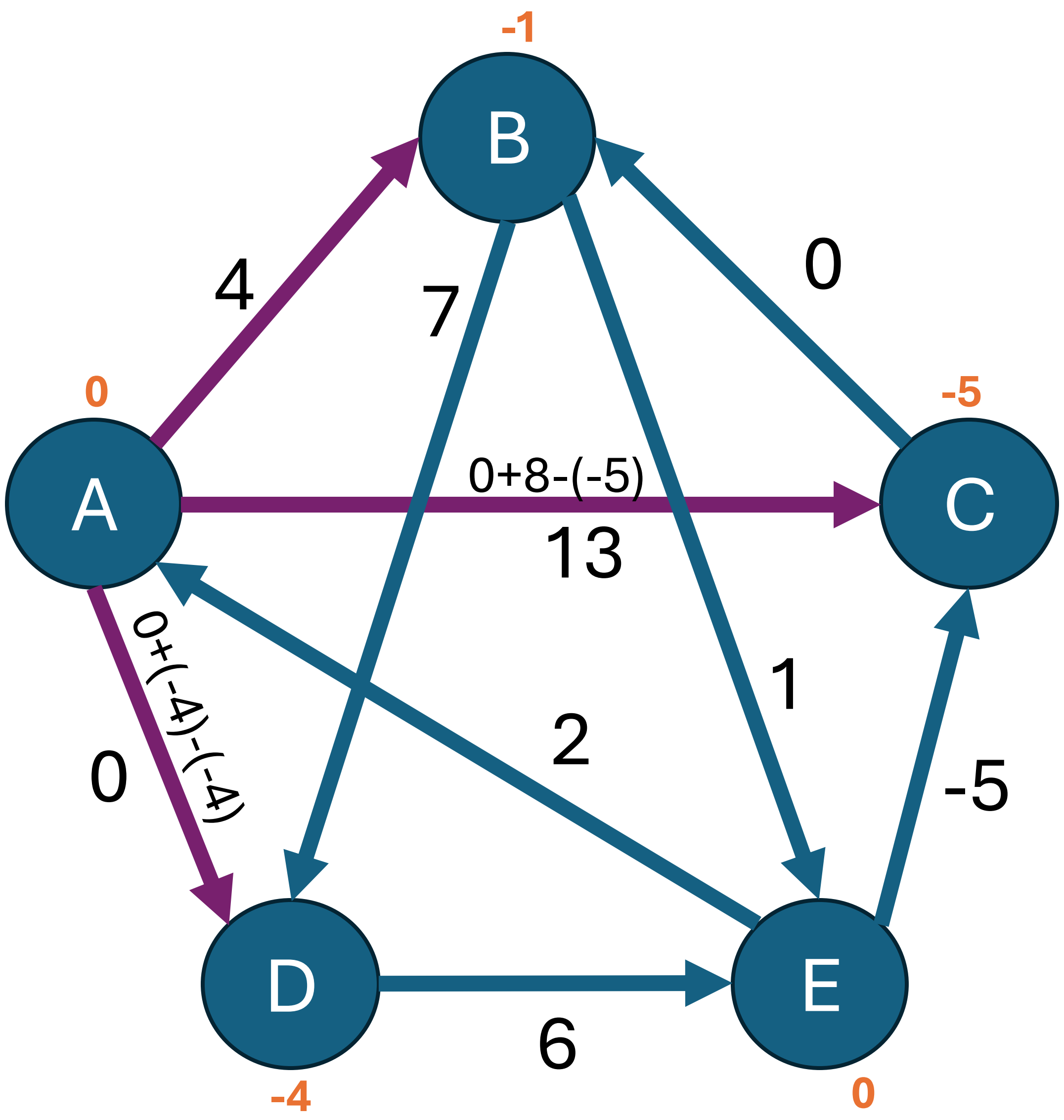
3



AB	AC	AD	BD	BE	CB	DE	EA	EC

	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

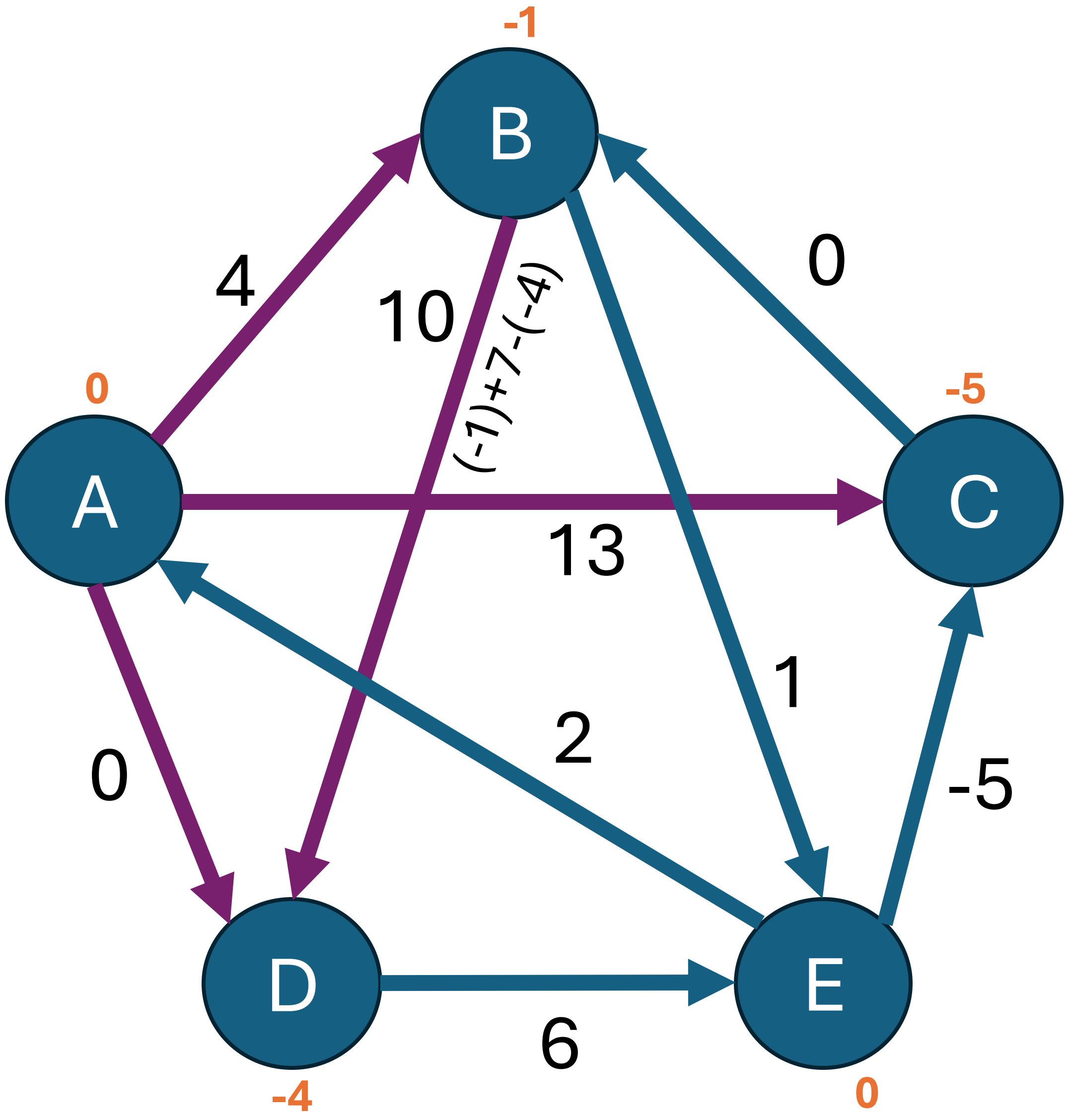
3



	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

AB	AC	AD	BD	BE	CB	DE	EA	EC

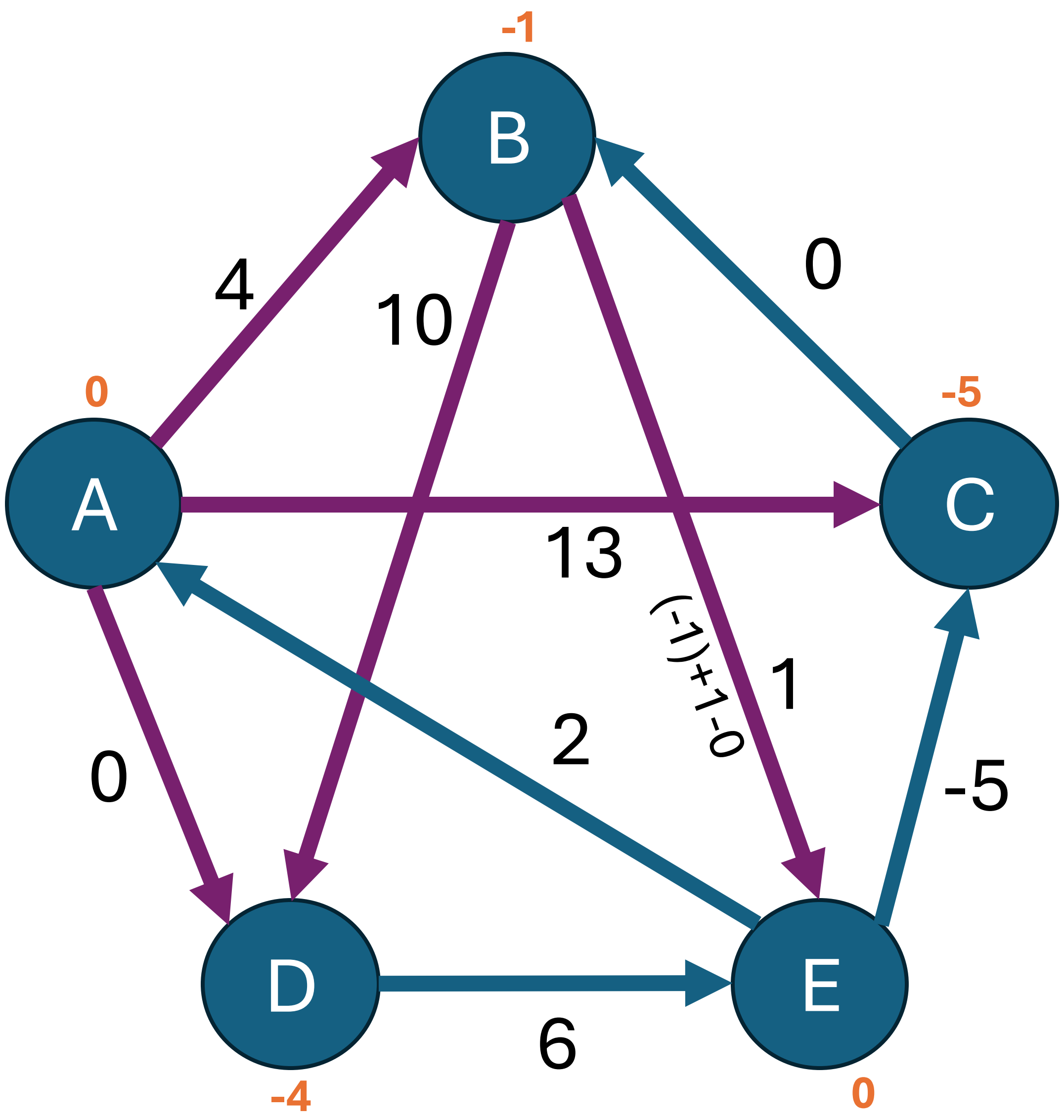
3



	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

AB	AC	AD	BD	BE	CB	DE	EA	EC

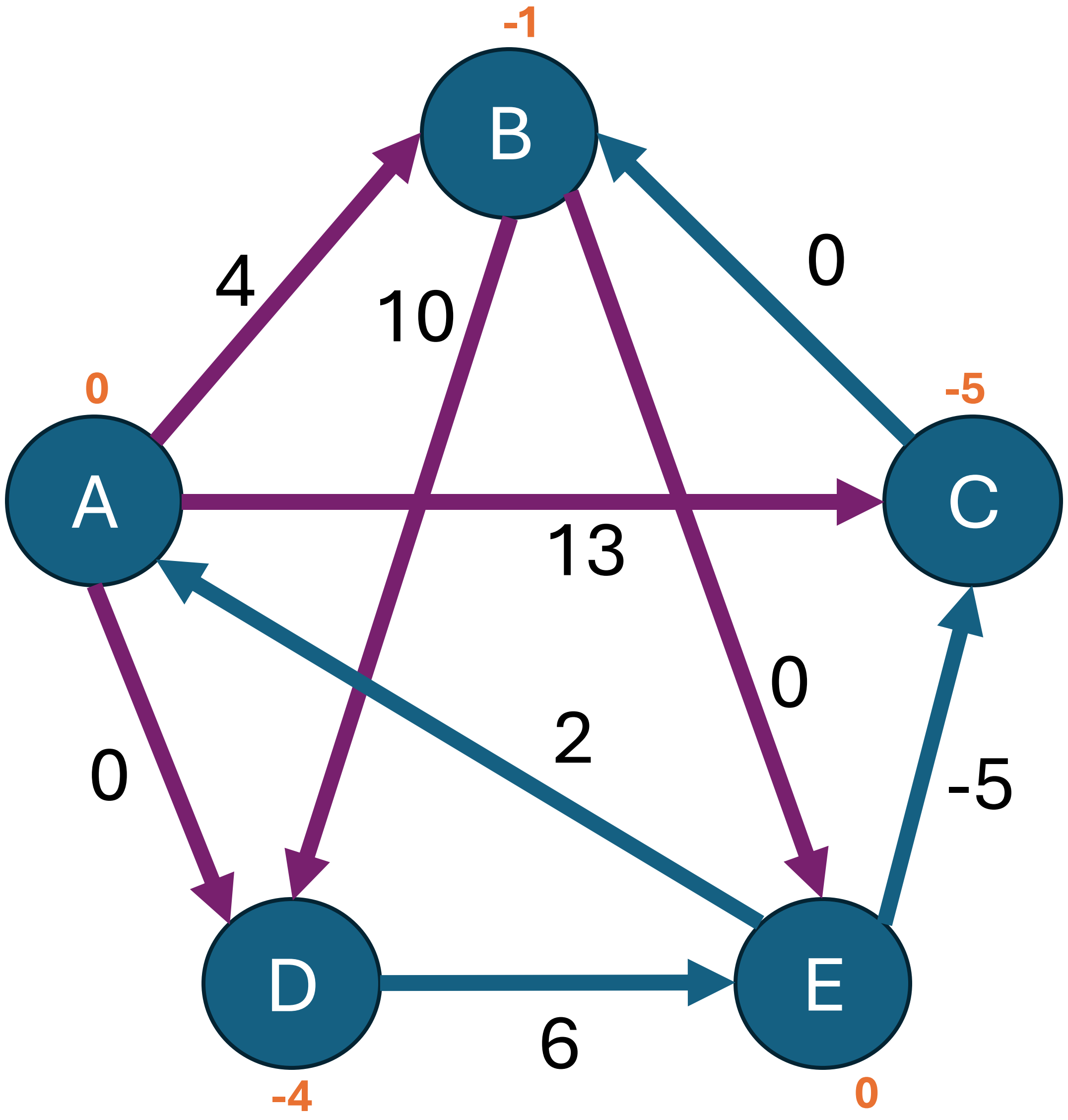
3



	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

AB	AC	AD	BD	BE	CB	DE	EA	EC

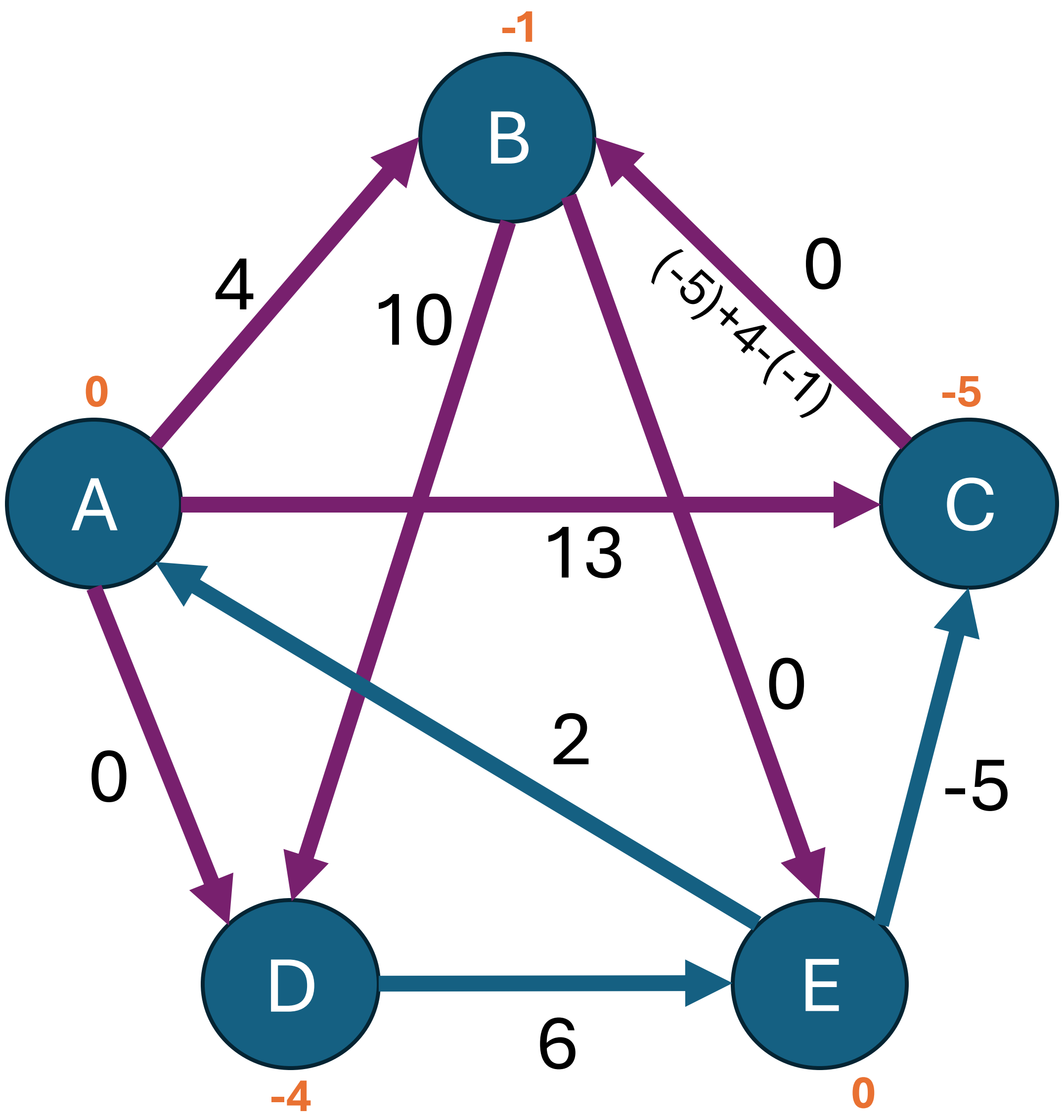
3



AB	AC	AD	BD	BE	CB	DE	EA	EC

	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

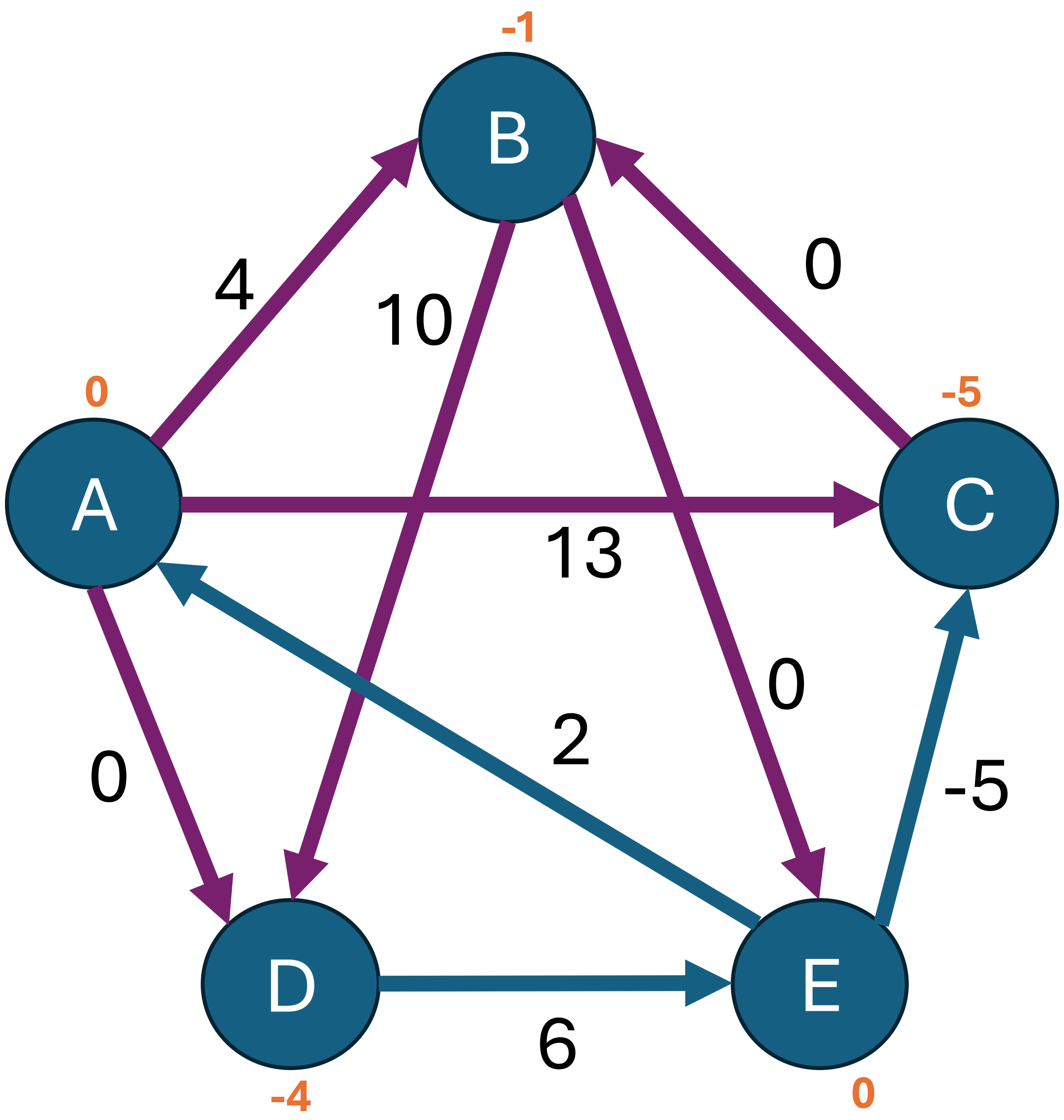
3



	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

AB	AC	AD	BD	BE	CB	DE	EA	EC

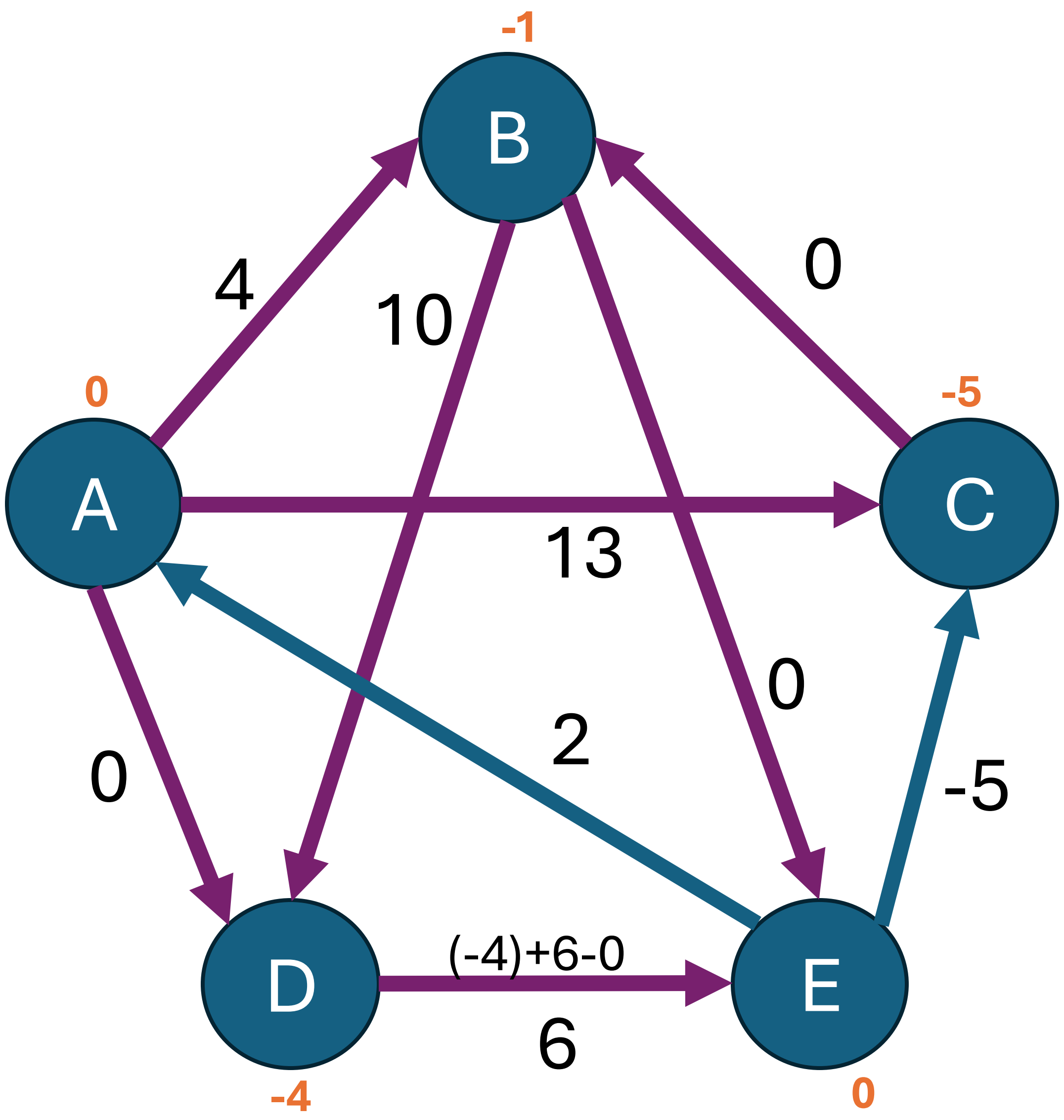
3



	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

AB	AC	AD	BD	BE	CB	DE	EA	EC

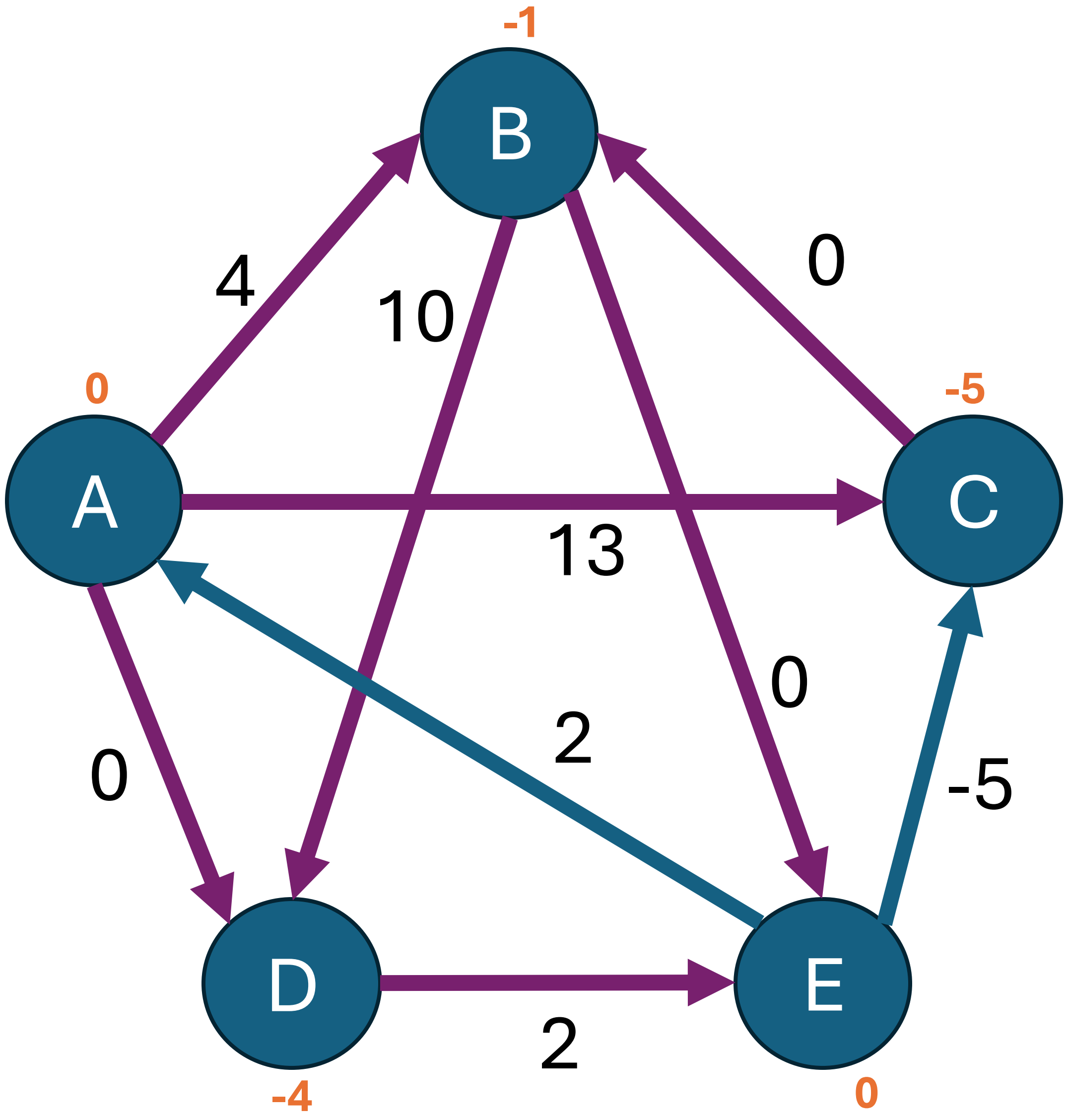
3



	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

AB	AC	AD	BD	BE	CB	DE	EA	EC

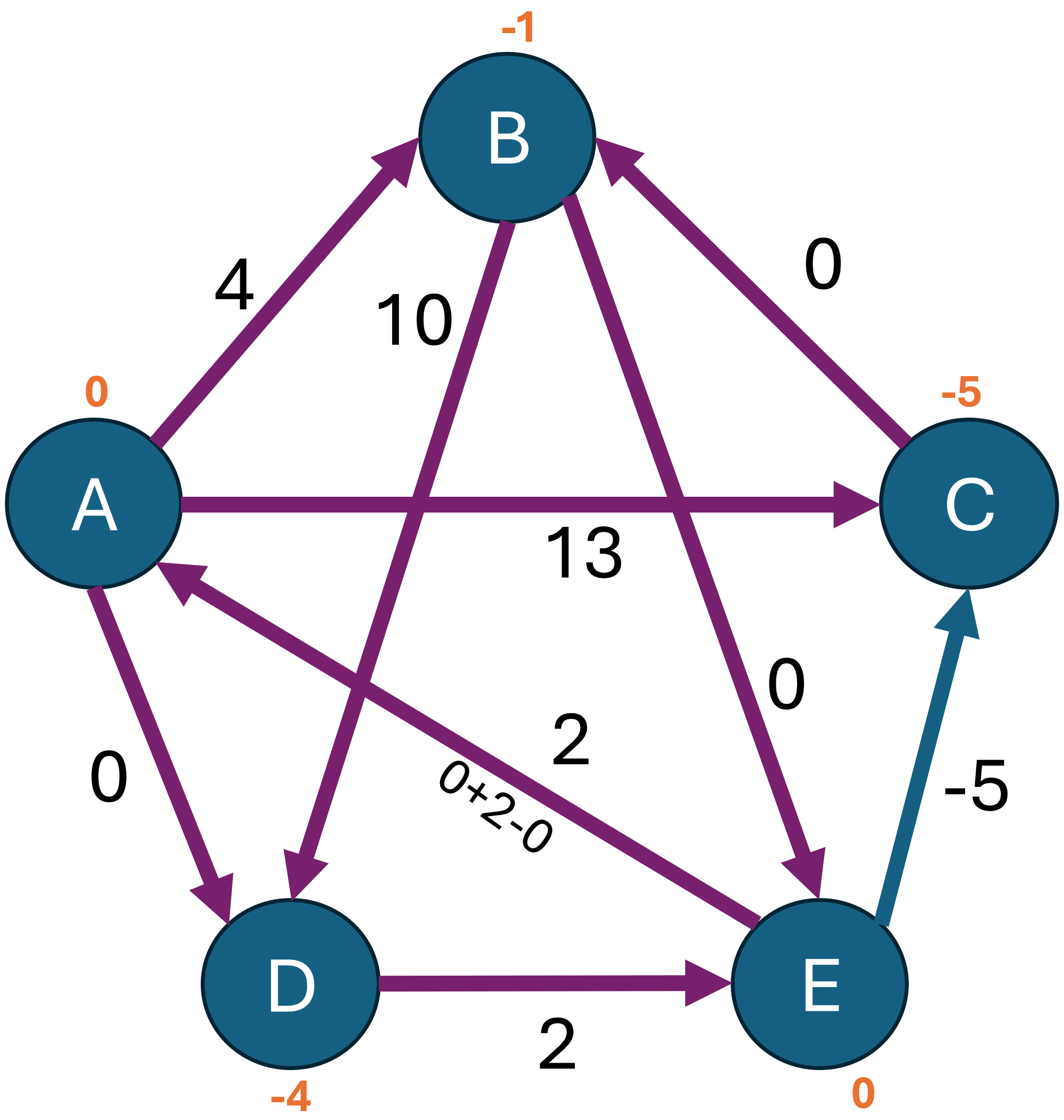
3



	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

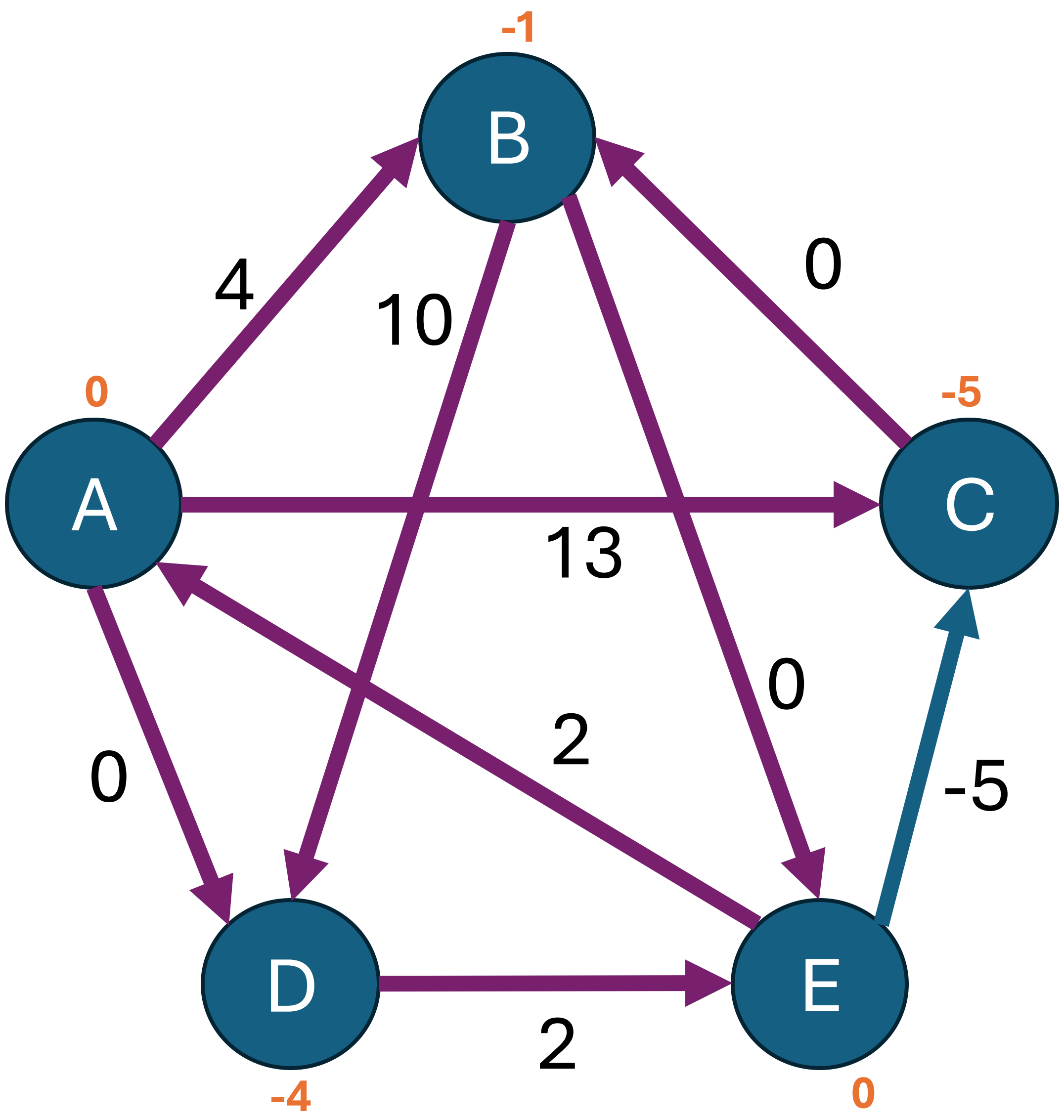
AB	AC	AD	BD	BE	CB	DE	EA	EC

3



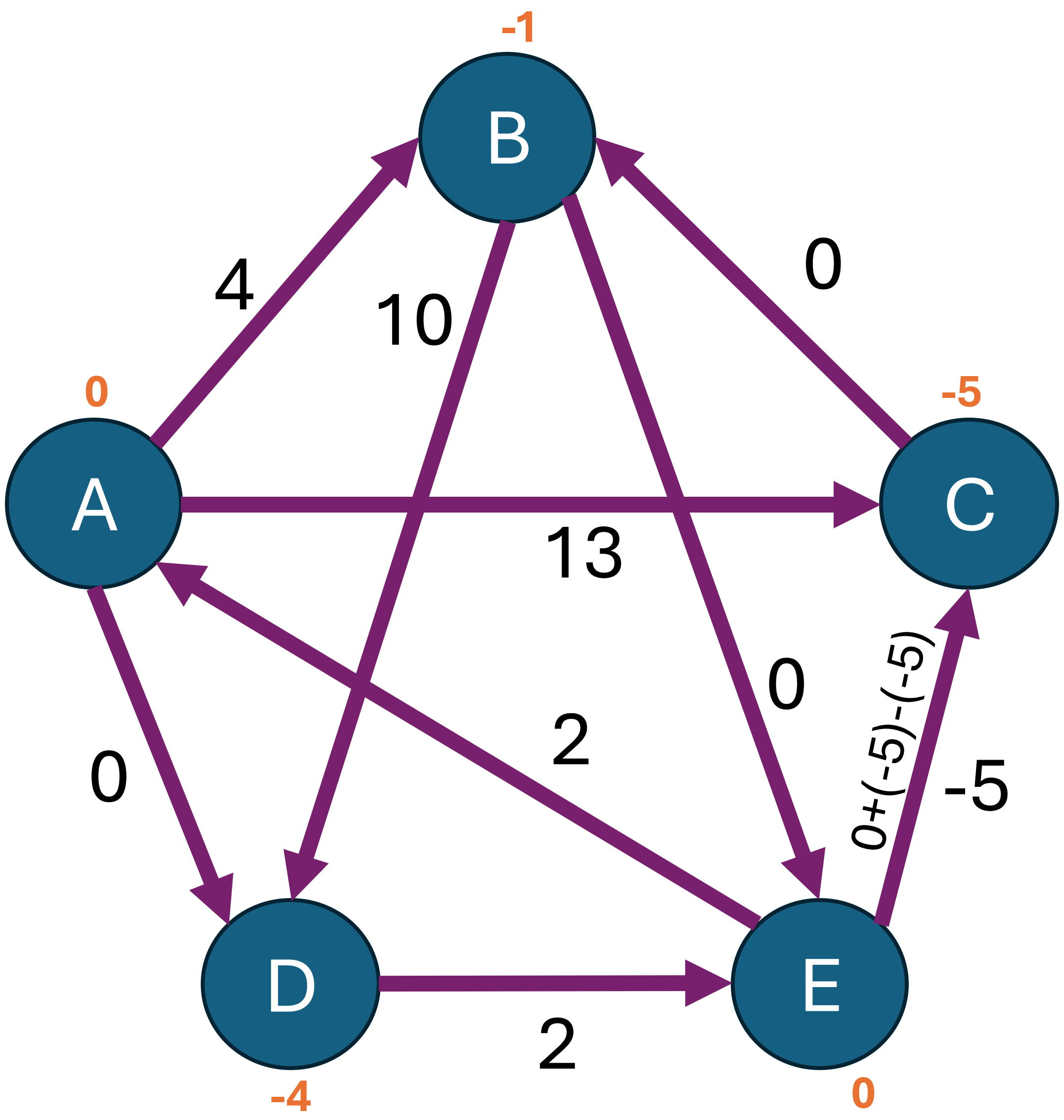
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

AB	AC	AD	BD	BE	CB	DE	EA	EC



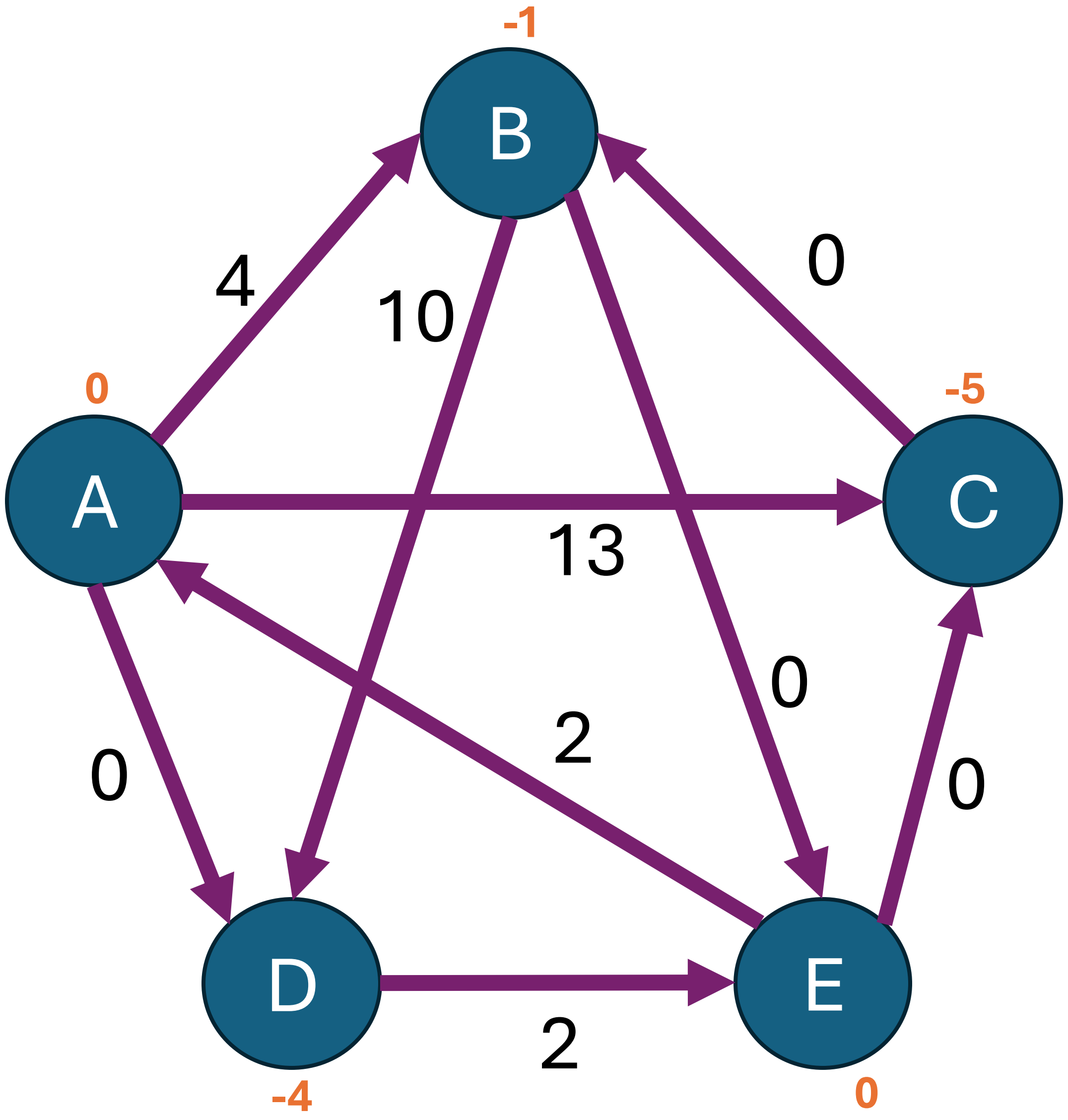
AB	AC	AD	BD	BE	CB	DE	EA	EC

	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S



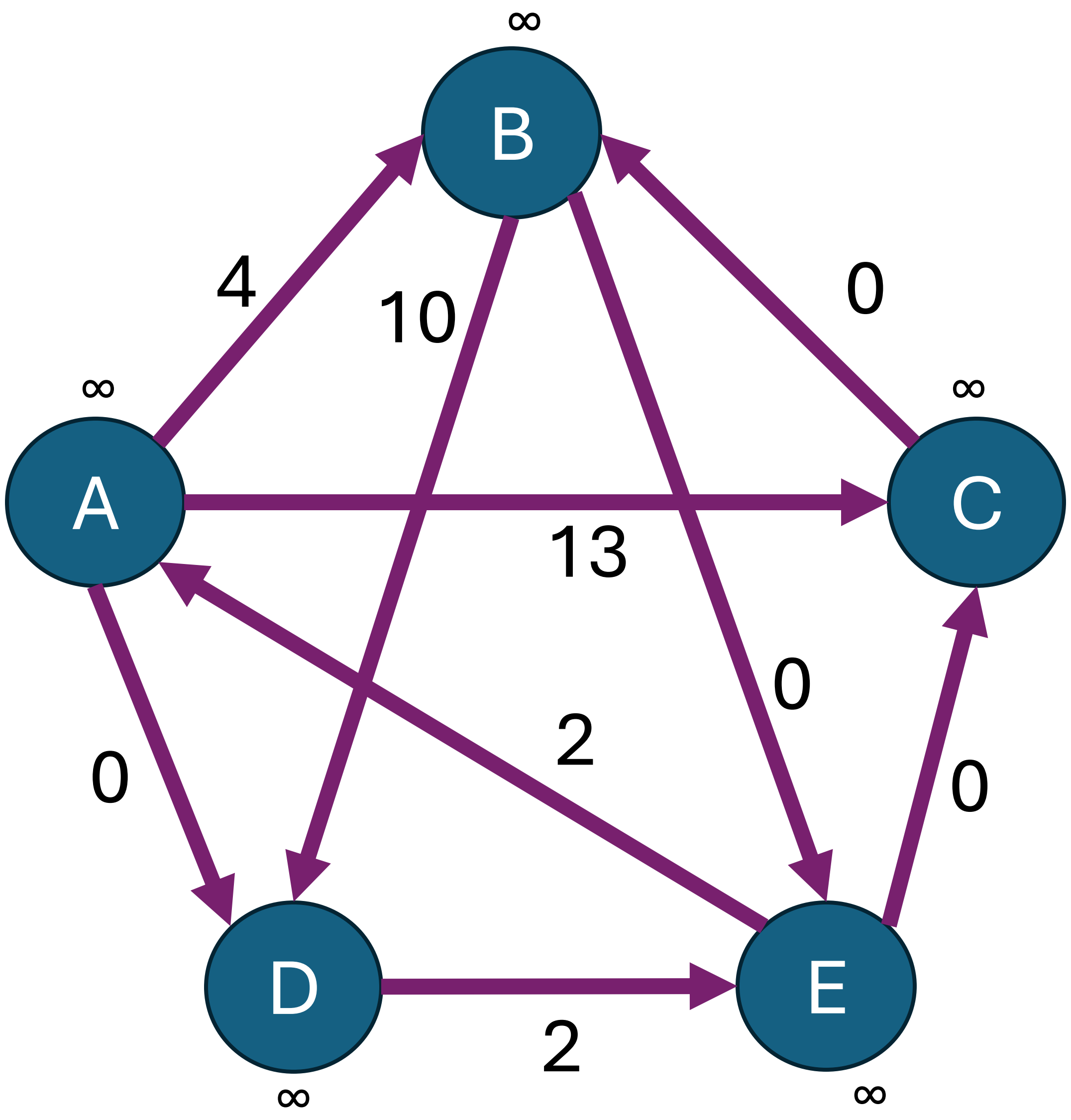
	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

AB	AC	AD	BD	BE	CB	DE	EA	EC

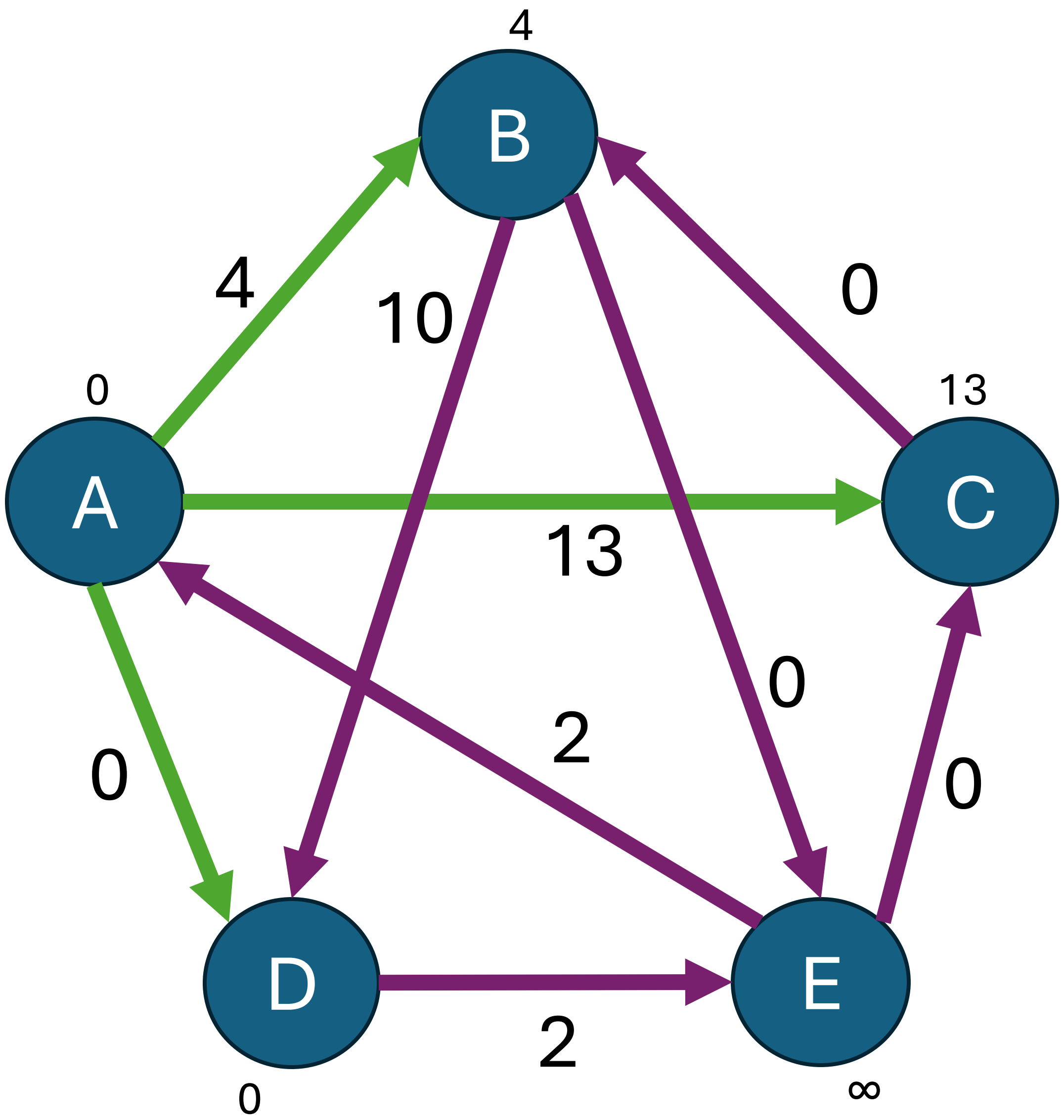


	Cost	Prev
S	0	-
A	0	S
B	-1	C
C	-5	E
D	-4	A
E	0	S

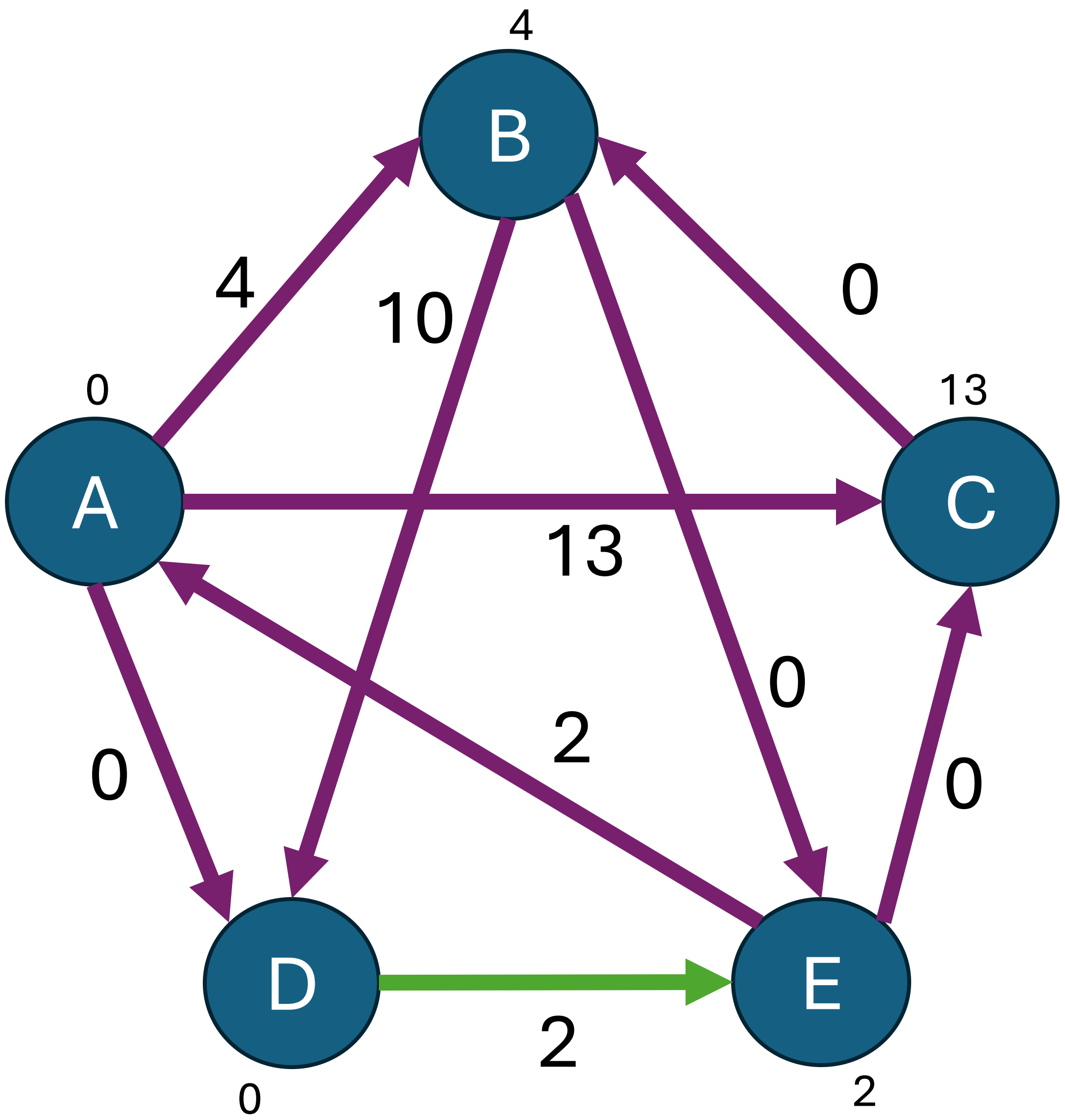
AB	AC	AD	BD	BE	CB	DE	EA	EC



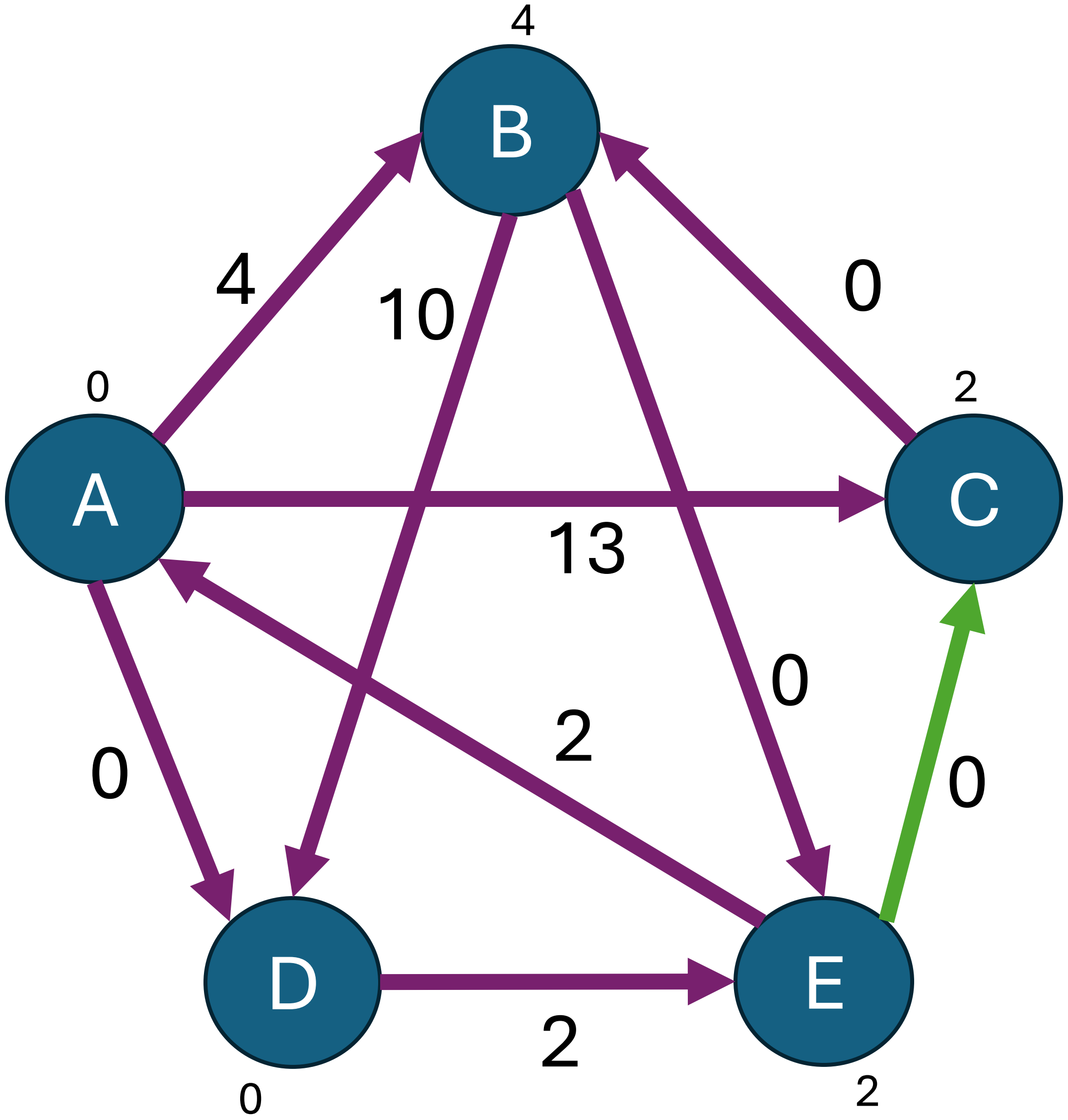
	A	B	C	D	E
A					



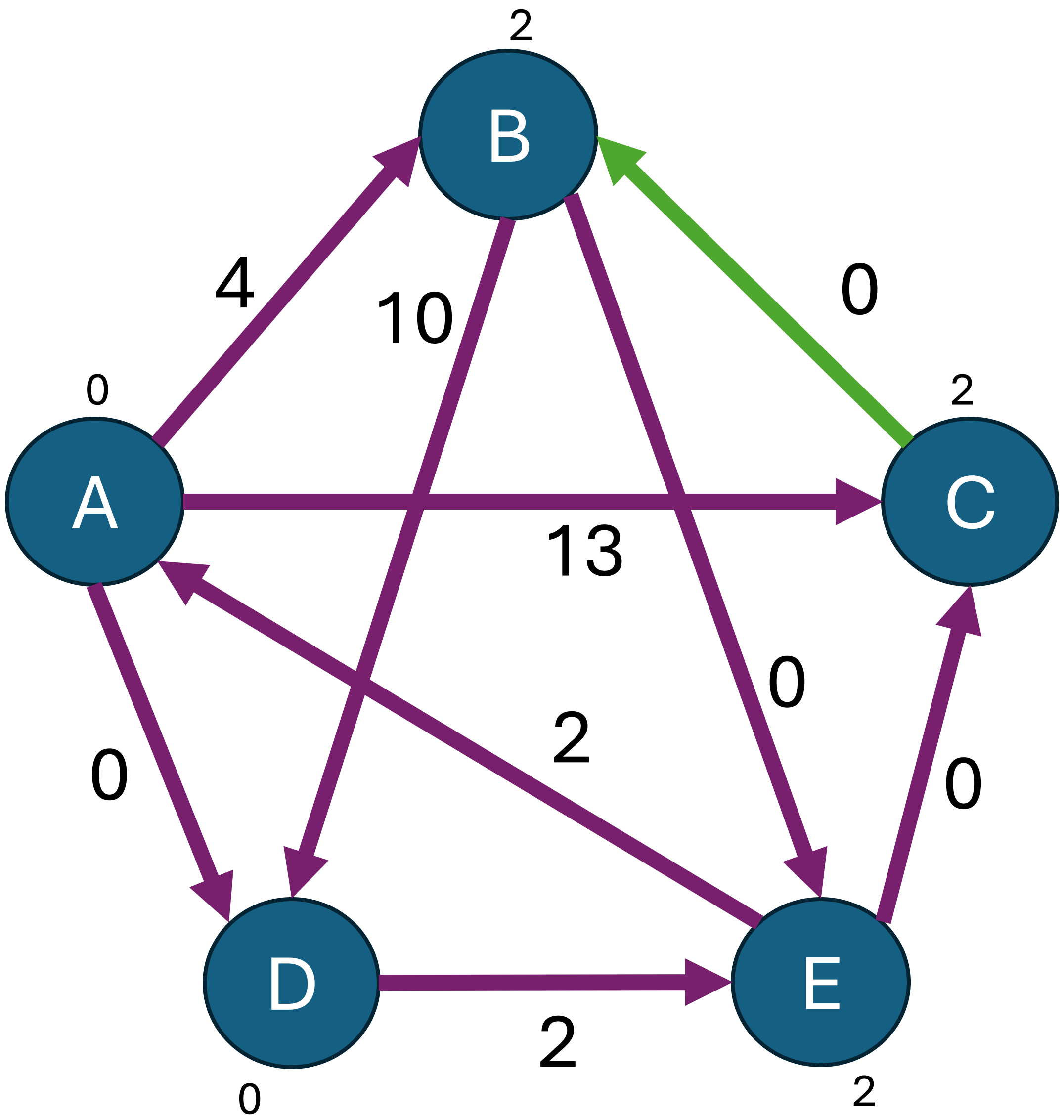
	A	B	C	D	E
A	0	4 A	13 A	0 A	



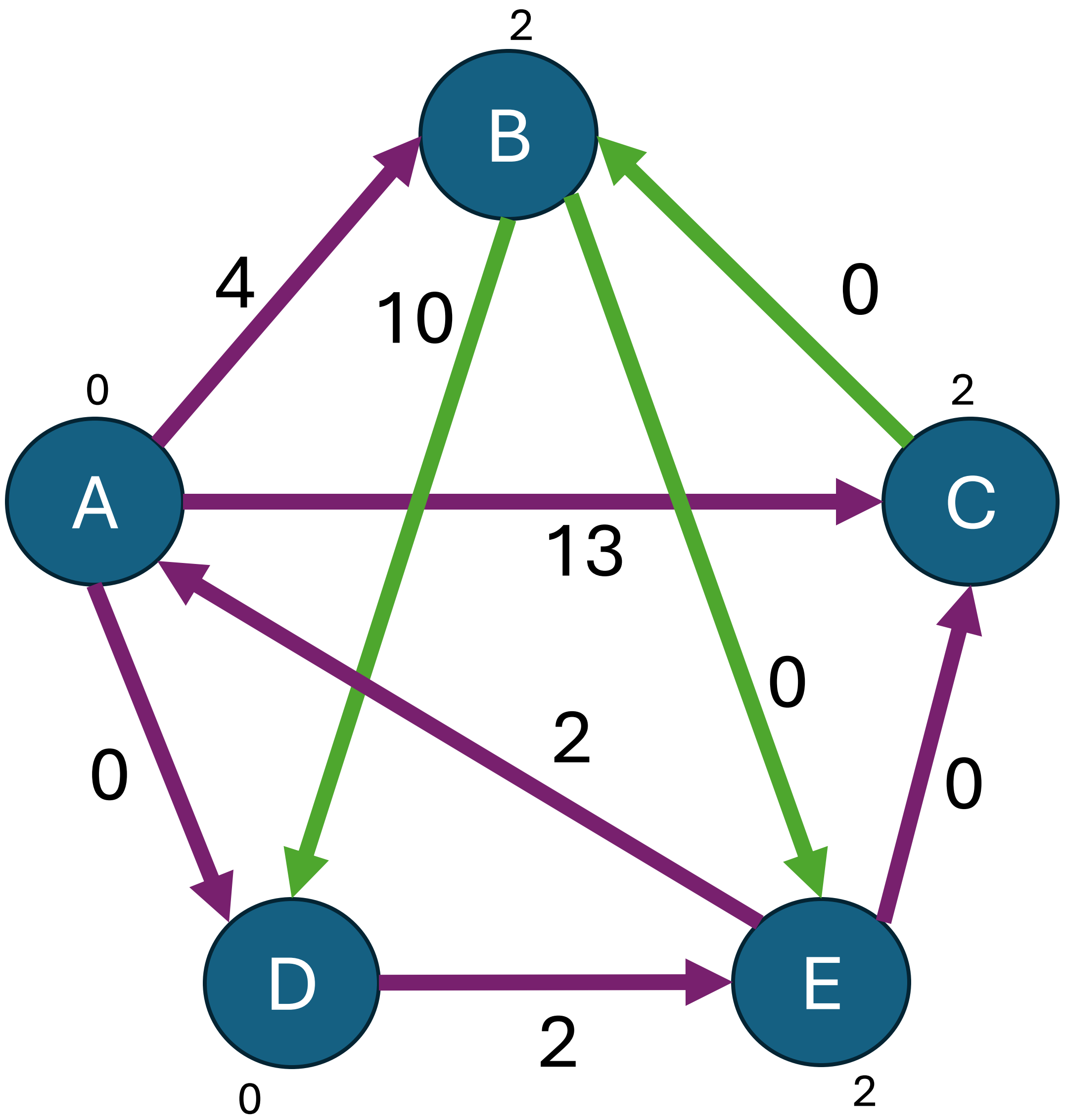
	A	B	C	D	E
A	0	4 A	13 A	0 A	
D	0	4 A	13 A	0 A	2 D



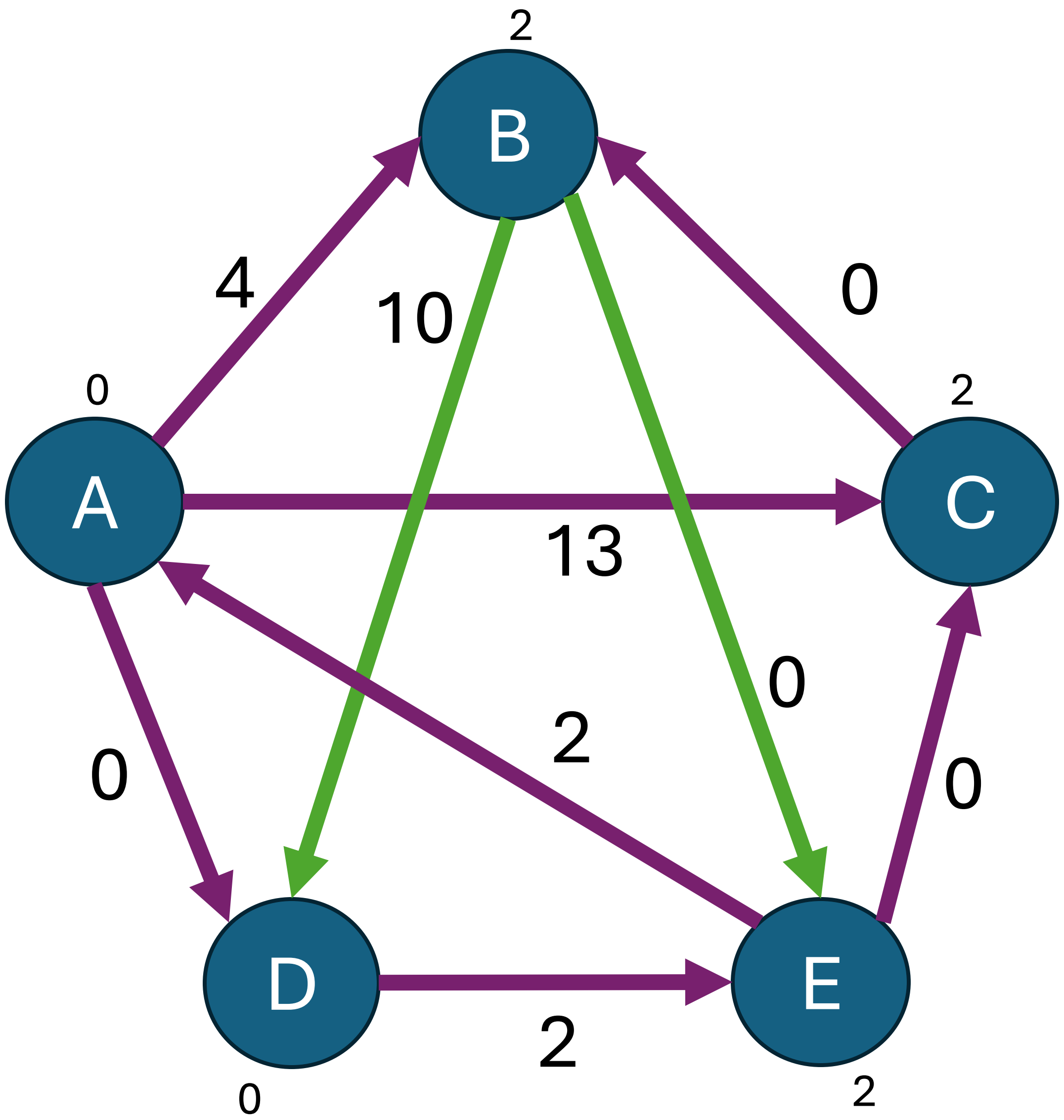
	A	B	C	D	E
A	0	4 A	13 A	0 A	
D	0	4 A	13 A	0 A	2 D
E	0	4 A	2 E	0 A	2 D



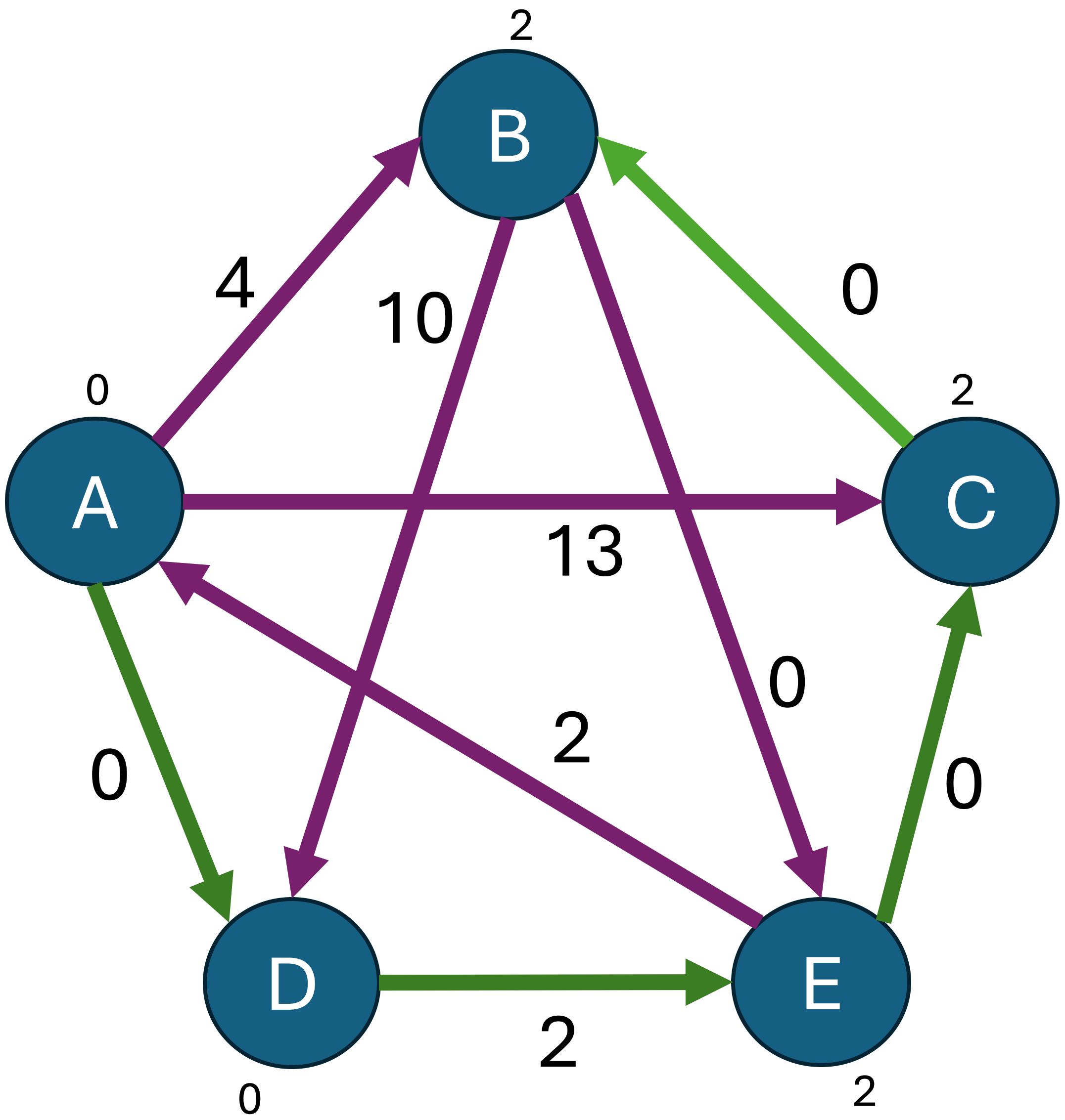
	A	B	C	D	E
A	0	4 A	13 A	0 A	
D	0	4 A	13 A	0 A	2 D
E	0	4 A	2 E	0 A	2 D
C	0	2 C	2 E	0 A	2 D
B	0	2 C	2 E	0 A	2 D



	A	B	C	D	E
A	0	4 A	13 A	0 A	
D	0	4 A	13 A	0 A	2 D
E	0	4 A	2 E	0 A	2 D
C	0	2 C	2 E	0 A	2 D
B	0	2 C	2 E	0 A	2 D



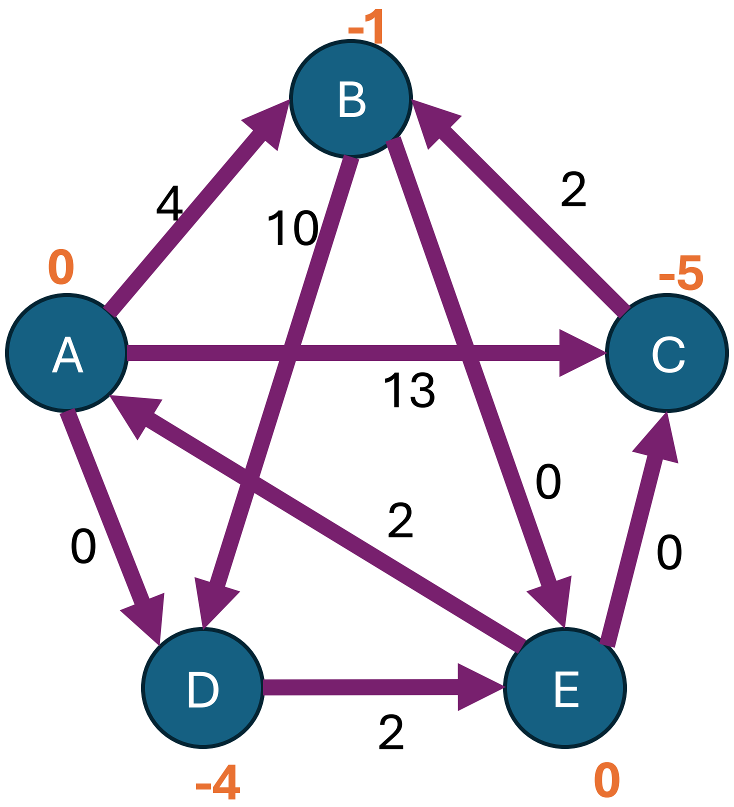
	A	B	C	D	E
A	0	4 A	13 A	0 A	
D	0	4 A	13 A	0 A	2 D
E	0	4 A	2 E	0 A	2 D
C	0	2 C	2 E	0 A	2 D
B	0	2 C	2 E	0 A	2 D



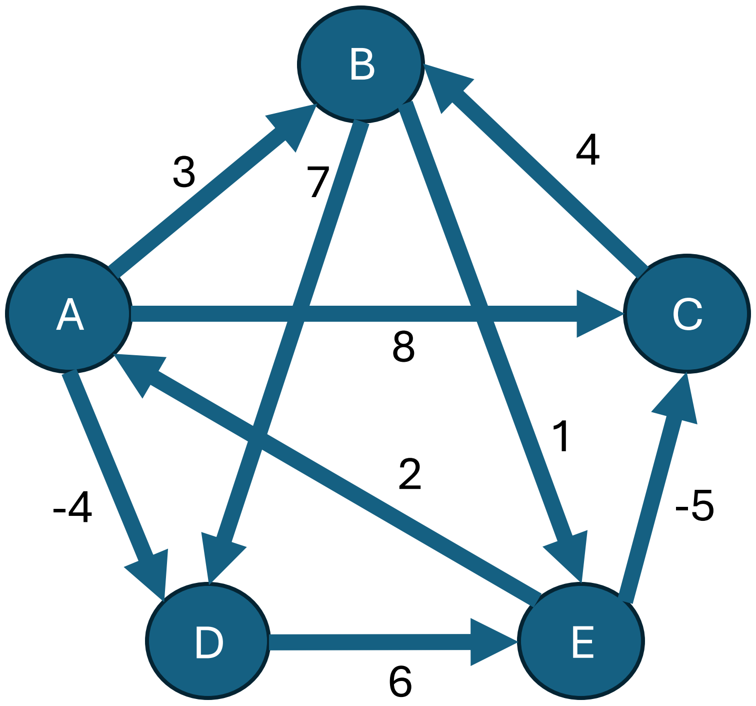
	A	B	C	D	E
A	0	4 A	13 A	0 A	
D	0	4 A	13 A	0 A	2 D
E	0	4 A	2 E	0 A	2 D
C	0	2 C	2 E	0 A	2 D
B	0	2 C	2 E	0 A	2 D

Let us reweight the edges to retrieve the original weights

Graph computed by Bellman-Ford and reweighting



Original Graph



Distances from A to every other vertex

	A	B	C	D	E
A	0	4 A	13 A	0 A	
D	0	4 A	13 A	0 A	2 D
E	0	4 A	2 E	0 A	2 D
C	0	2 C	2 E	0 A	2 D
B	0	2 C	2 E	0 A	2 D

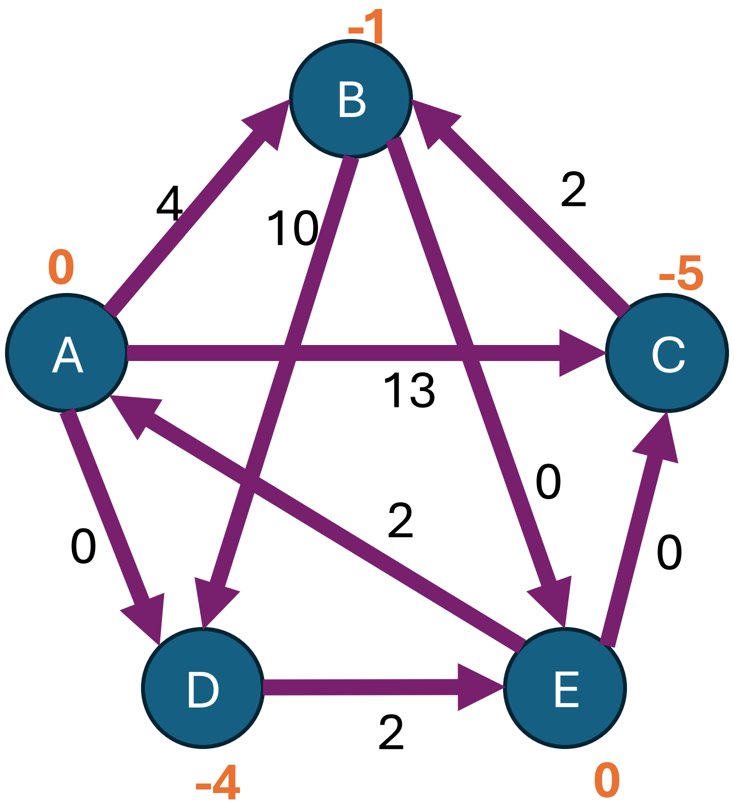
$d(u,v)=d'(u,v)-h(u)+h(v)$

$d(a,b) = 2-0+(-1) = 1$

$d(a,d) = 0-0+(-4) = -4$

$d(a,c) = 2-0+(-5) = -3$

Graph computed by Bellman-Ford and reweighting



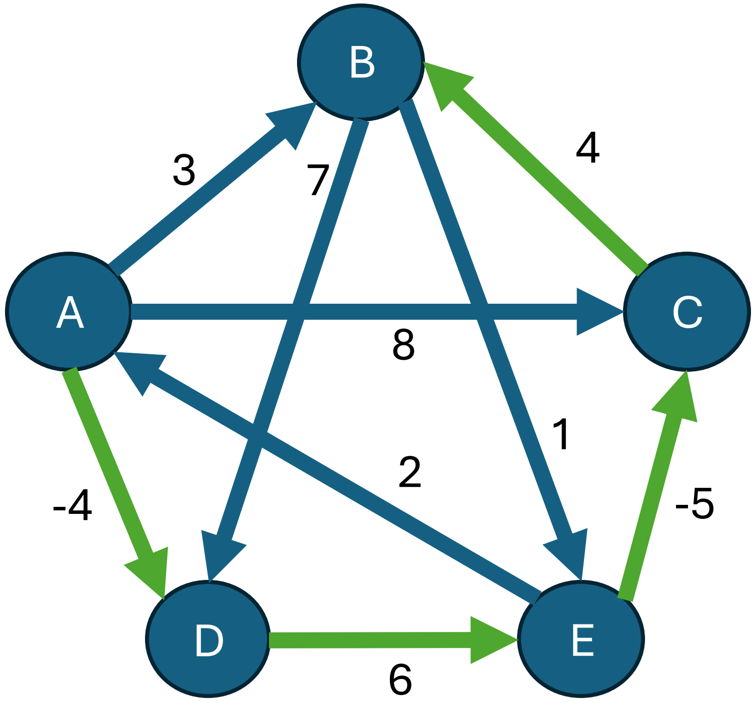
$d(u,v)=d'(u,v)-h(u)+h(v)$

$d(a,b) = 2-0+(-1) = 1$

$d(a,d) = 0-0+(-4) = -4$

$d(a,c) = 2-0+(-5) = -3$

Original Graph

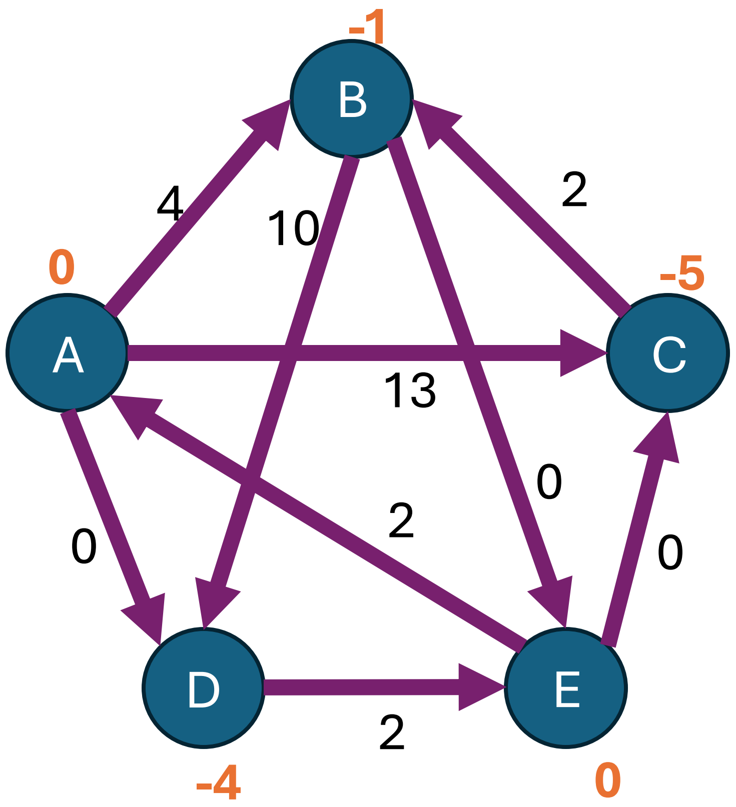


Total is 1!

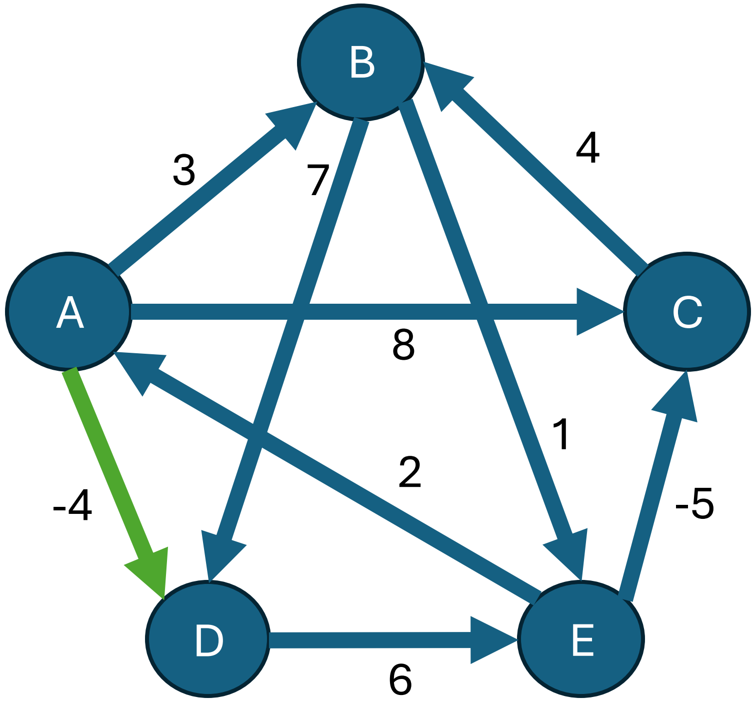
Distances from A to every other vertex

	A	B	C	D	E
A	0	4 A	13 A	0 A	
D	0	4 A	13 A	0 A	2 D
E	0	4 A	2 E	0 A	2 D
C	0	2 C	2 E	0 A	2 D
B	0	2 C	2 E	0 A	2 D

Graph computed by Bellman-Ford and reweighting



Original Graph



Total is -4!

Distances from A to every other vertex

	A	B	C	D	E
A	0	4 A	13 A	0 A	
D	0	4 A	13 A	0 A	2 D
E	0	4 A	2 E	0 A	2 D
C	0	2 C	2 E	0 A	2 D
B	0	2 C	2 E	0 A	2 D

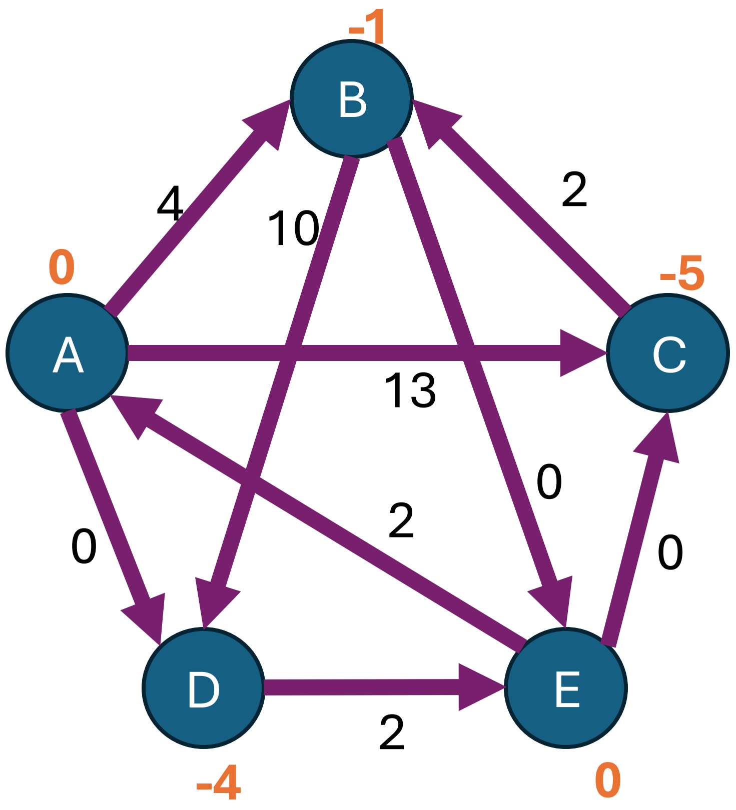
d(u,v)=d'(u,v)−h(u)+h(v)

d(a,b) = 2-0+(-1) = 1

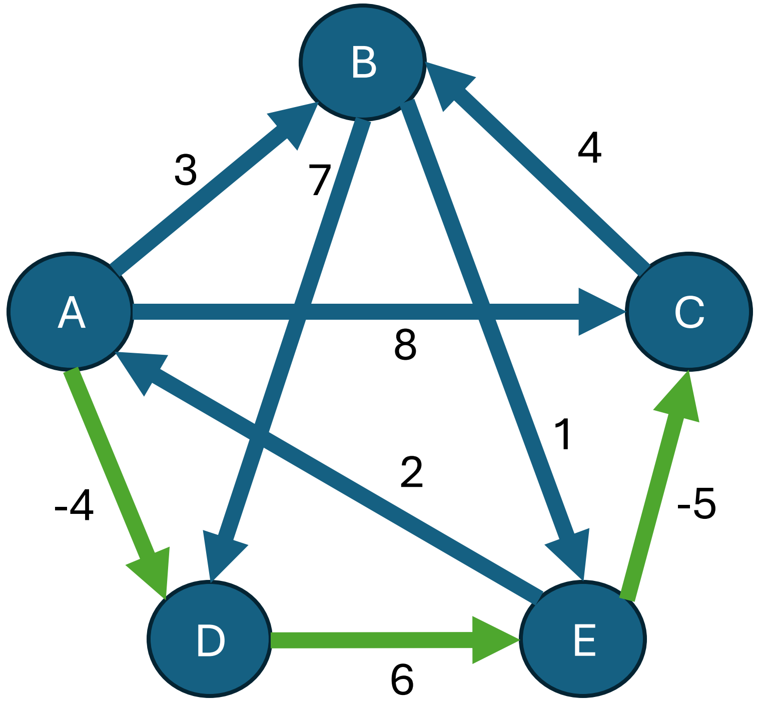
d(a,d) = 0-0+(-4) = -4

d(a,c) = 2-0+(-5) = -3

Graph computed by Bellman-Ford and reweighting



Original Graph



Total is -3!

Distances from A to every other vertex

	A	B	C	D	E
A	0	4 A	13 A	0 A	
D	0	4 A	13 A	0 A	2 D
E	0	4 A	2 E	0 A	2 D
C	0	2 C	2 E	0 A	2 D
B	0	2 C	2 E	0 A	2 D

$d(u,v)=d'(u,v)-h(u)+h(v)$

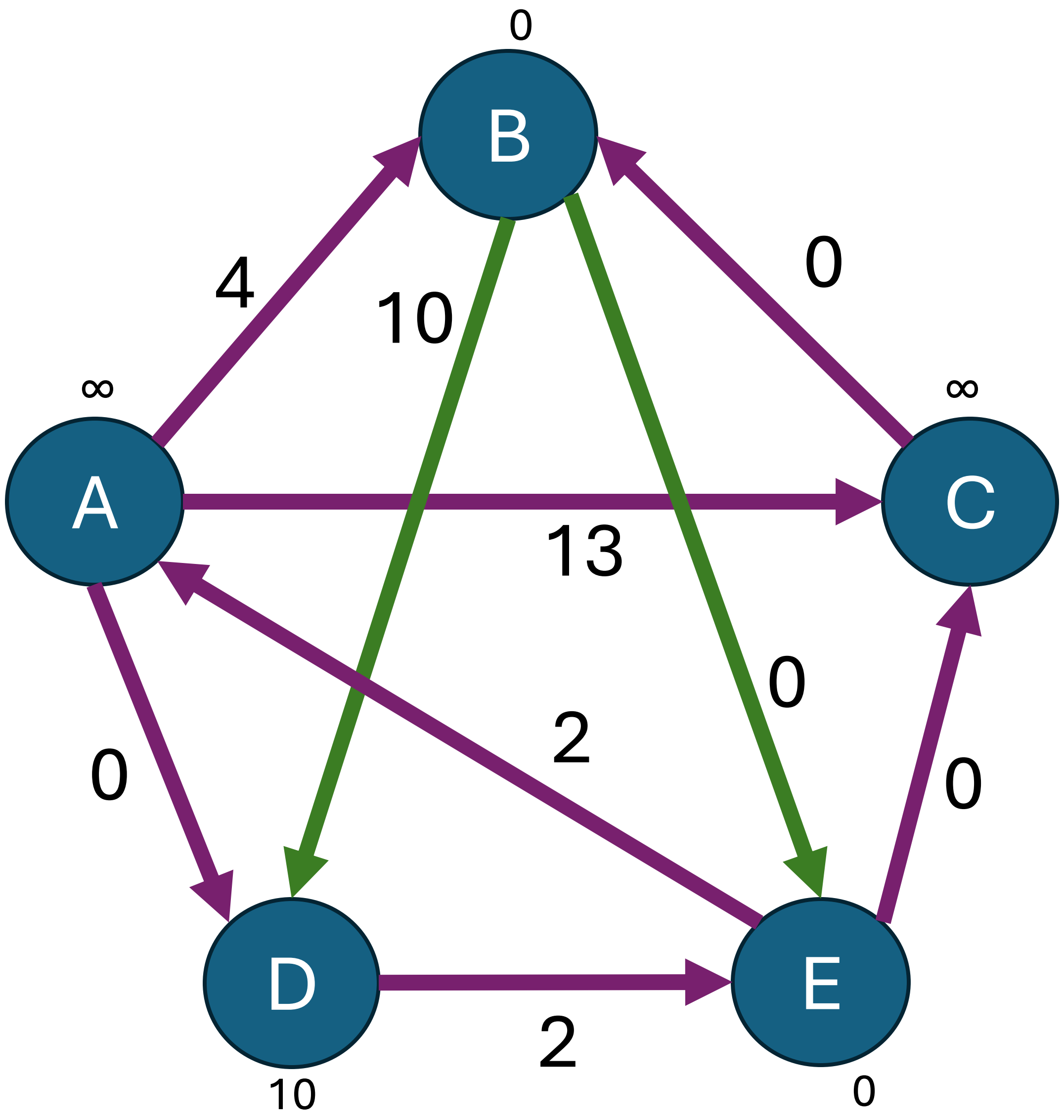
$d(a,b) = 2-0+(-1) = 1$

$d(a,d) = 0-0+(-4) = -4$

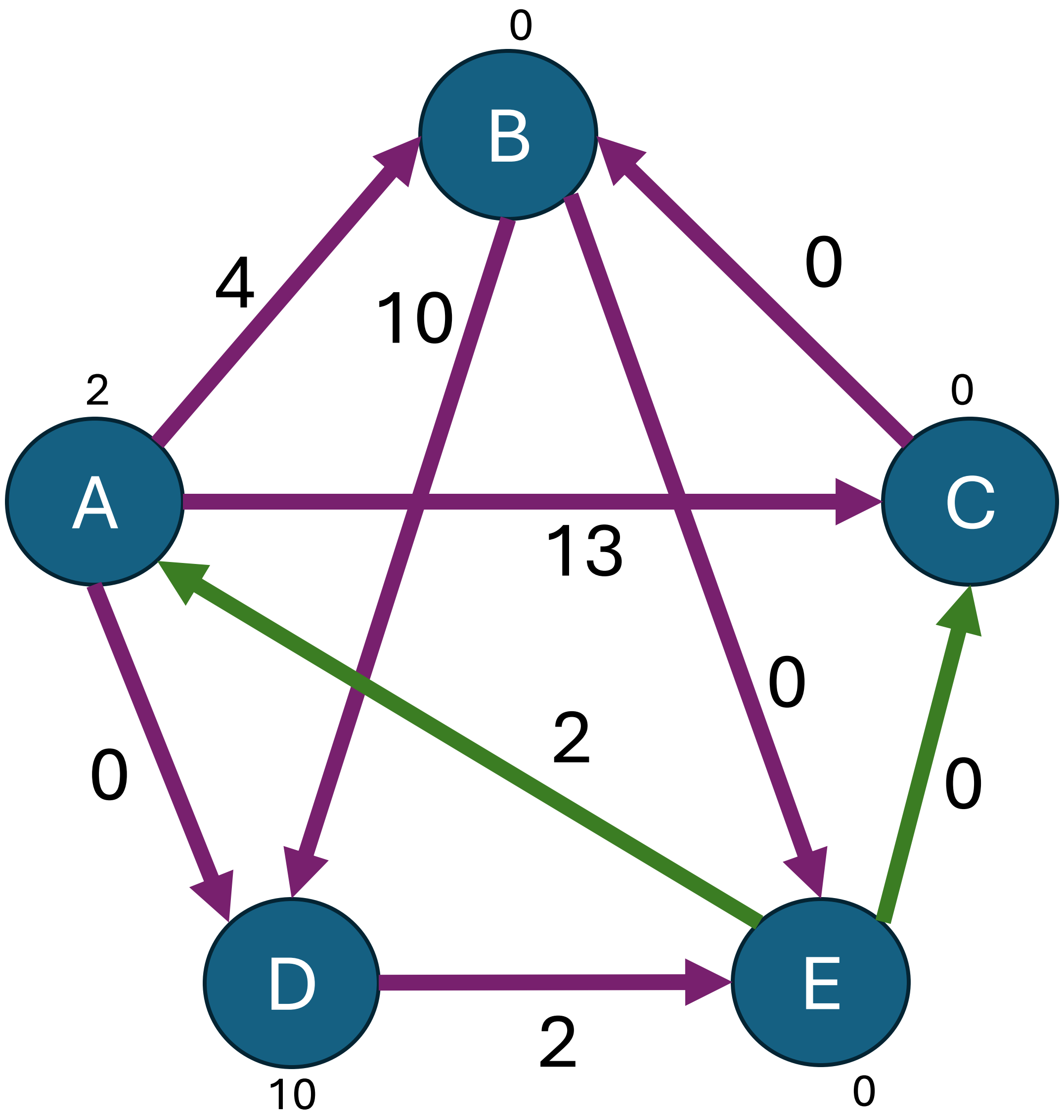
$\mathbf{d(a,c) = 2-0+(-5) = -3}$

Perform Dijkstra on every other vertex in the graph

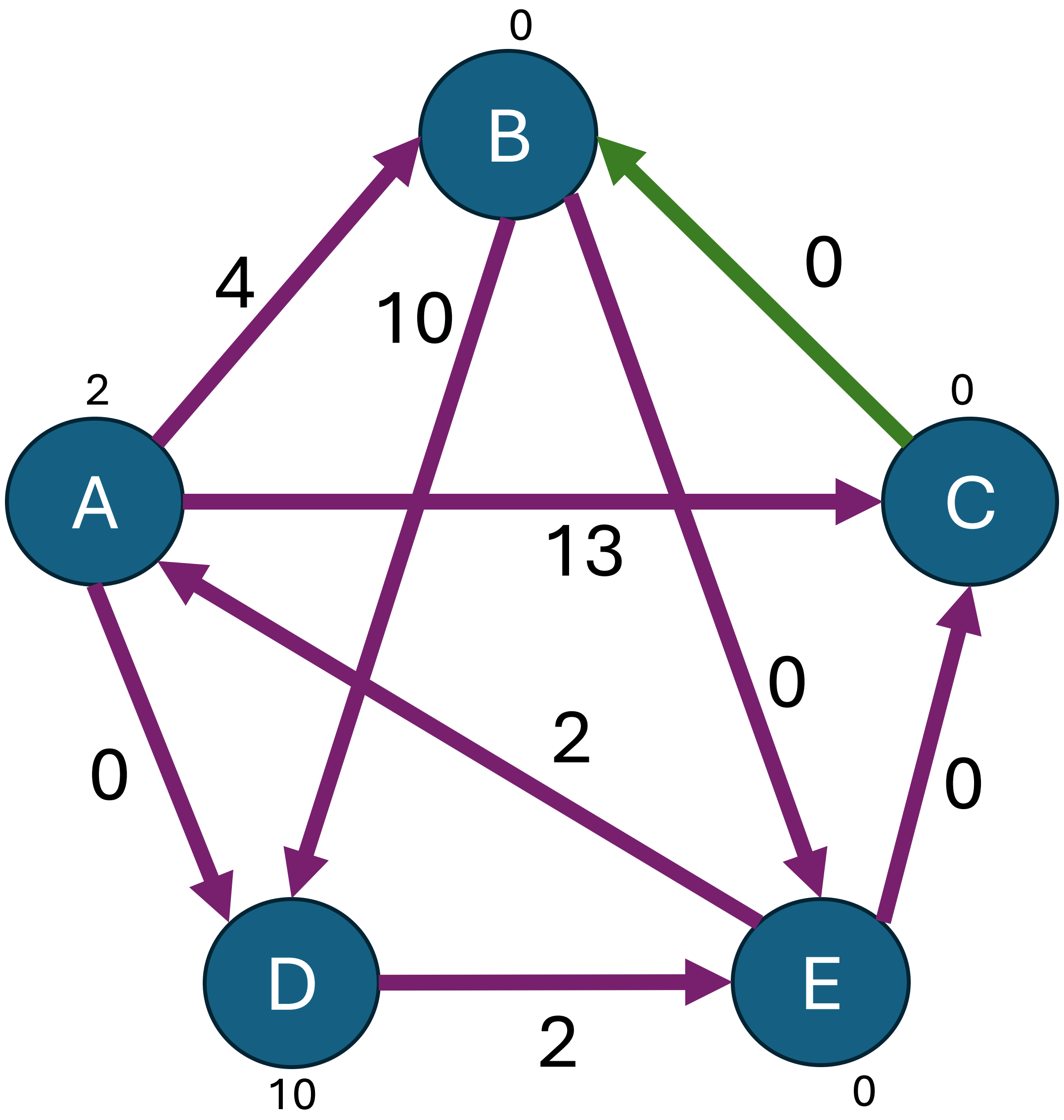
Perform Dijkstra on vertex B



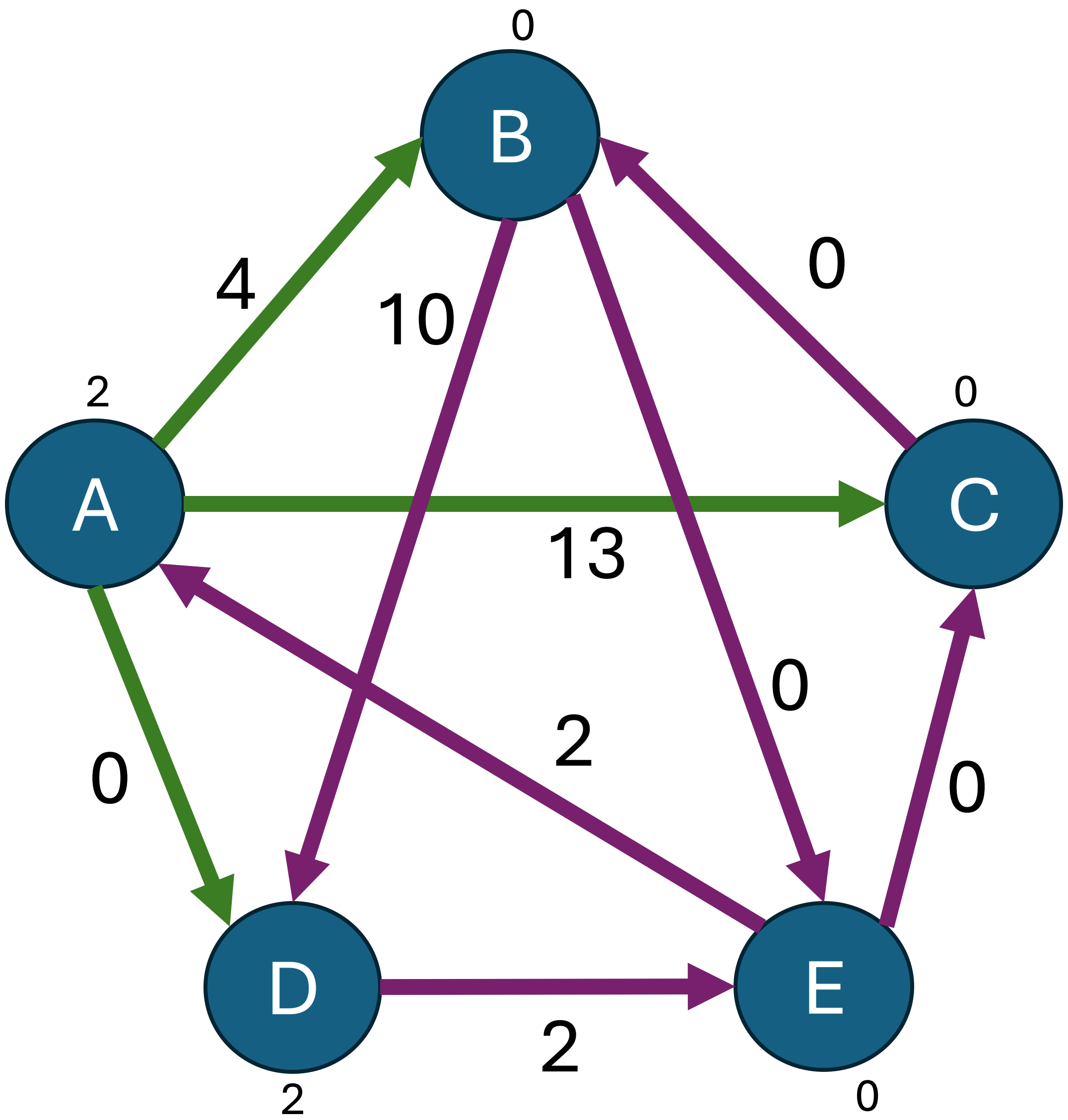
	B	A	C	D	E
B	0			10 B	0 B



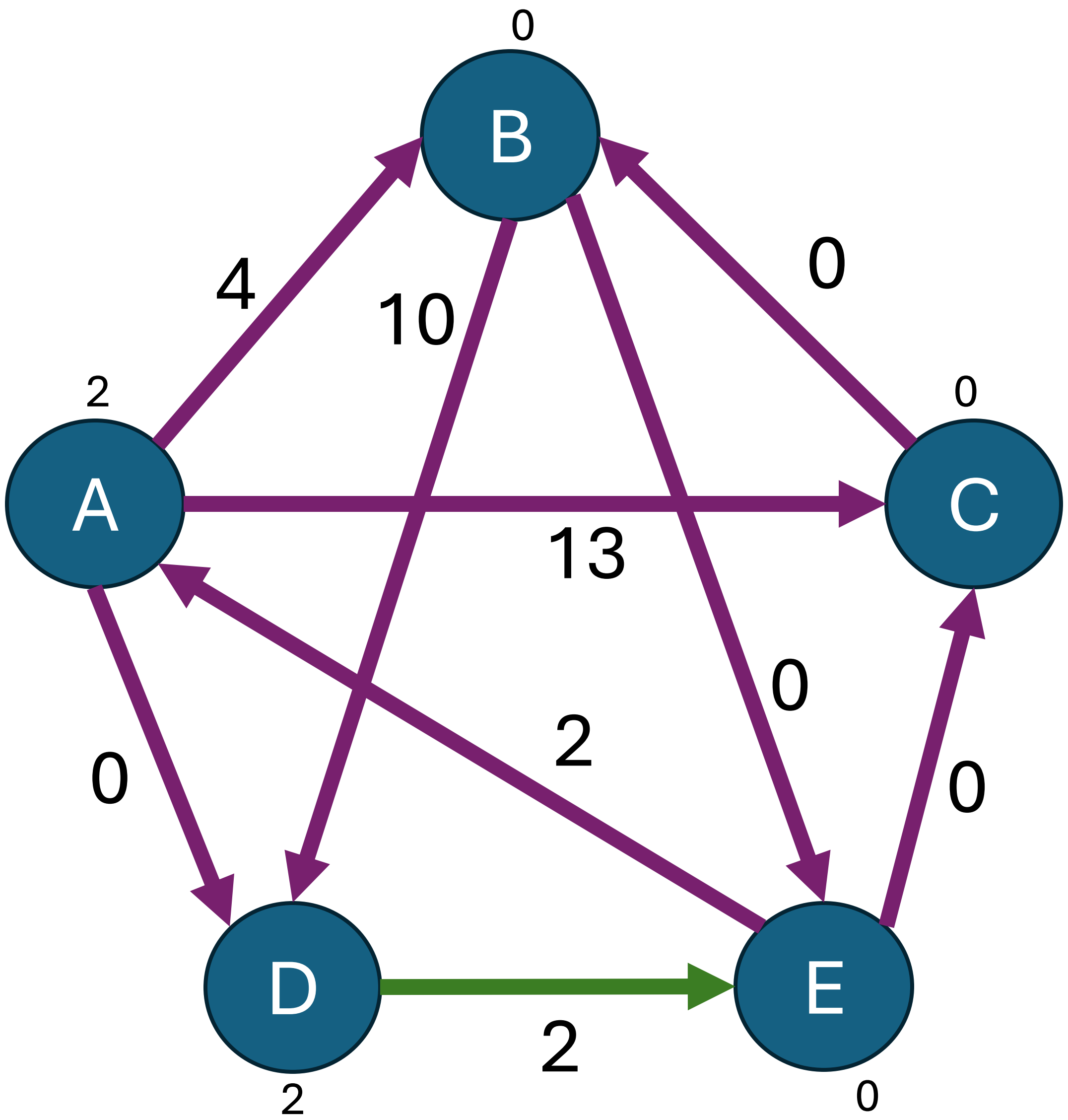
	B	A	C	D	E
B	0			10 B	0 B
E	0	2 E	0 E	10 B	0 B



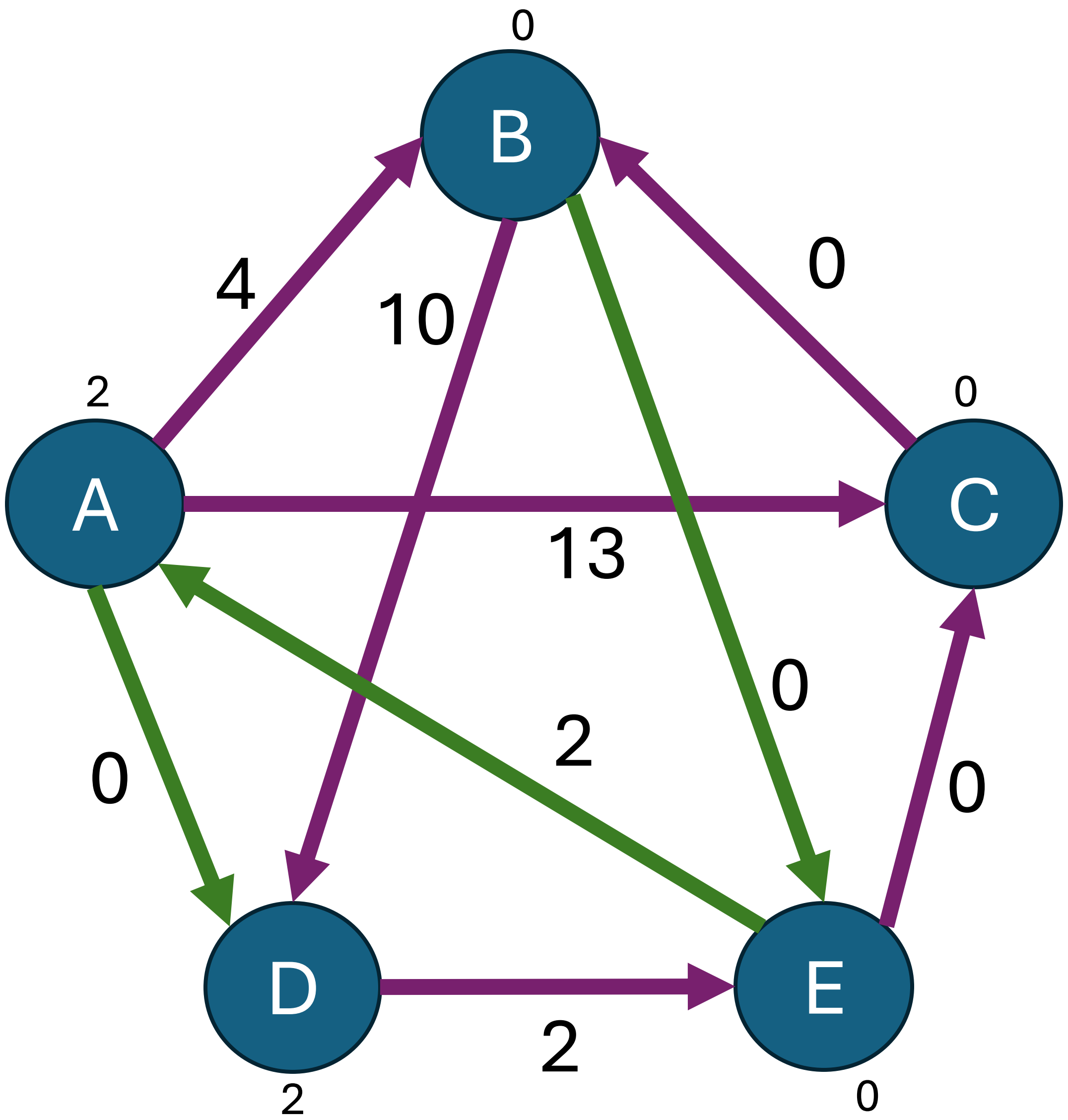
	B	A	C	D	E
B	0			10 B	0 B
E	0	2 E	0 E	10 B	0 B
C	0	2 E	0 E	10 B	0 B



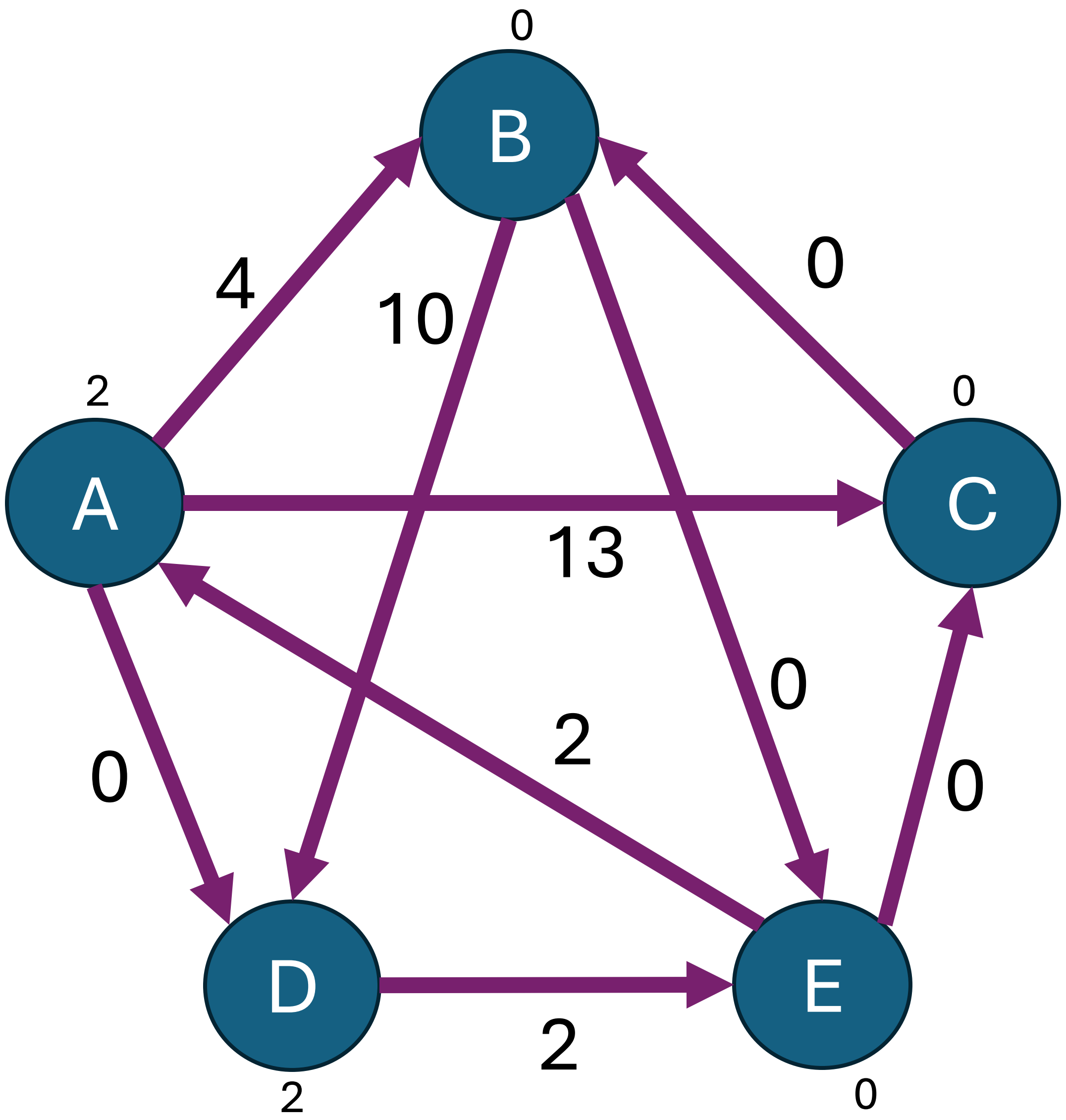
	B	A	C	D	E
B	0			10 B	0 B
E	0	2 E	0 E	10 B	0 B
C	0	2 E	0 E	10 B	0 B
A	0	2 E	0 E	2 A	0 B



	B	A	C	D	E
B	0			10 B	0 B
E	0	2 E	0 E	10 B	0 B
C	0	2 E	0 E	10 B	0 B
A	0	2 E	0 E	2 A	0 B
D	0	2 E	0 E	2 A	0 B

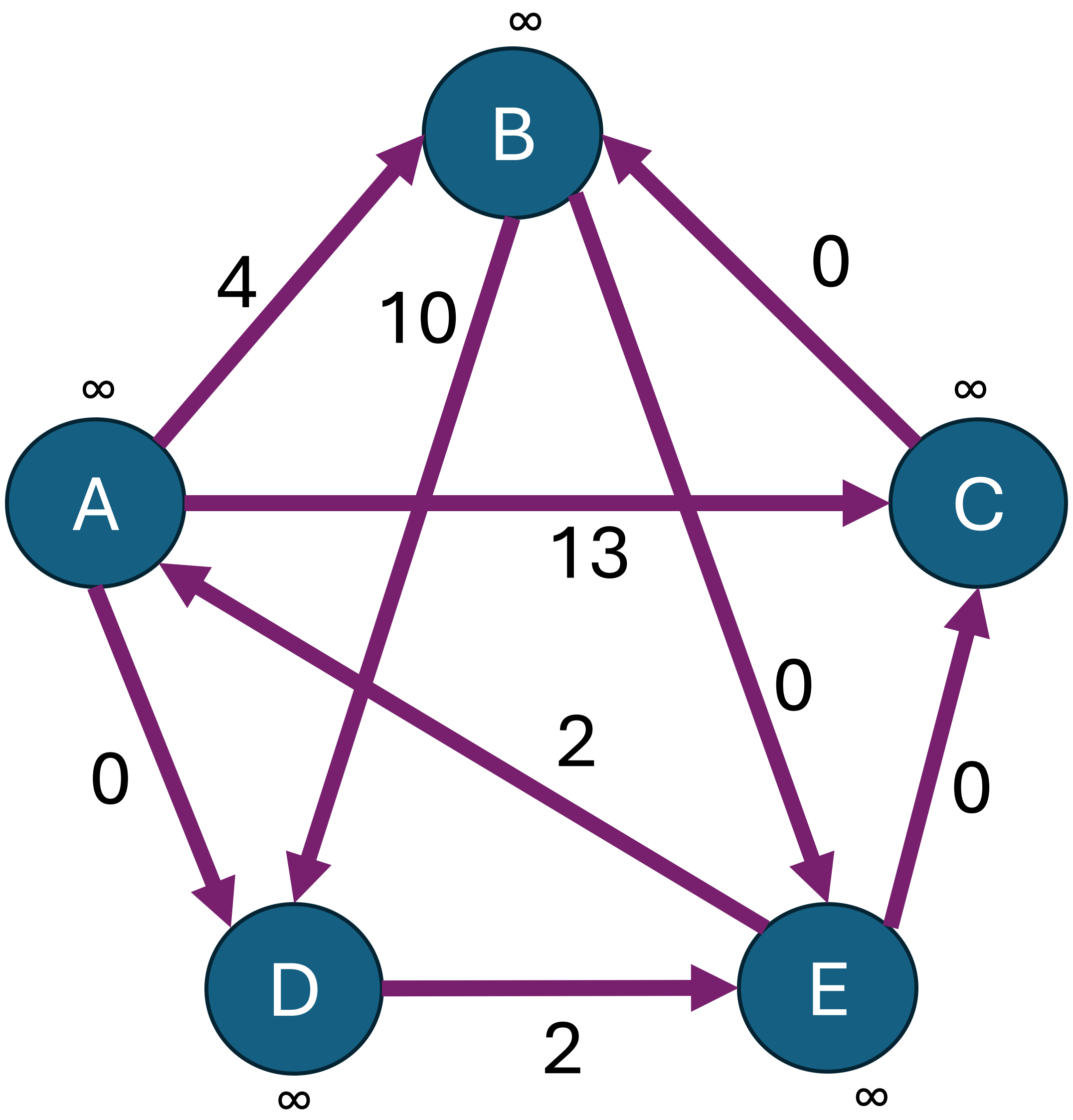


	B	A	C	D	E
B	0			10 B	0 B
E	0	2 E	0 E	10 B	0 B
C	0	2 E	0 E	10 B	0 B
A	0	2 E	0 E	2 A	0 B
D	0	2 E	0 E	2 A	0 B

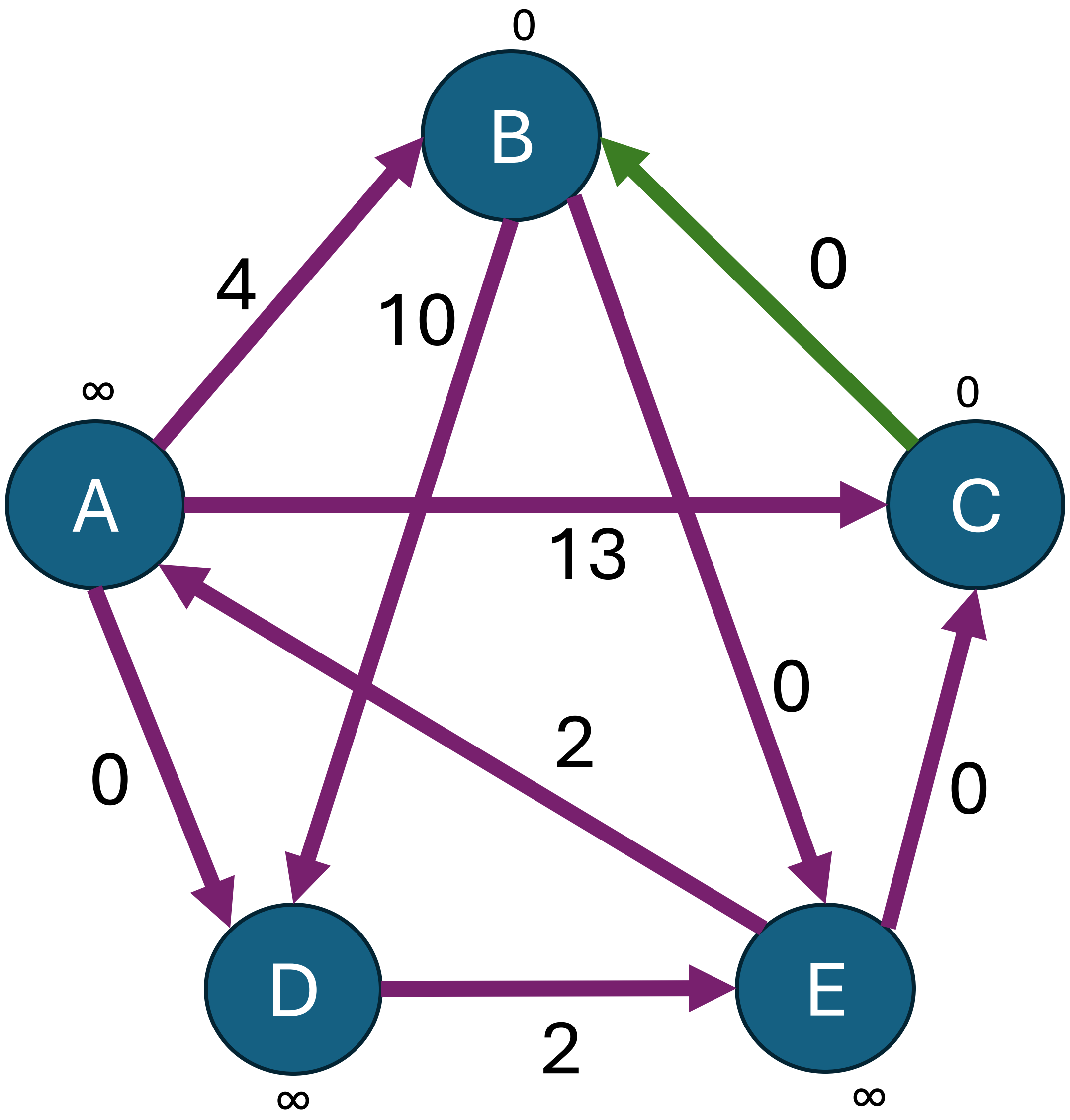


	B	A	C	D	E
B	0			10 B	0 B
E	0	2 E	0 E	10 B	0 B
C	0	2 E	0 E	10 B	0 B
A	0	2 E	0 E	2 A	0 B
D	0	2 E	0 E	2 A	0 B

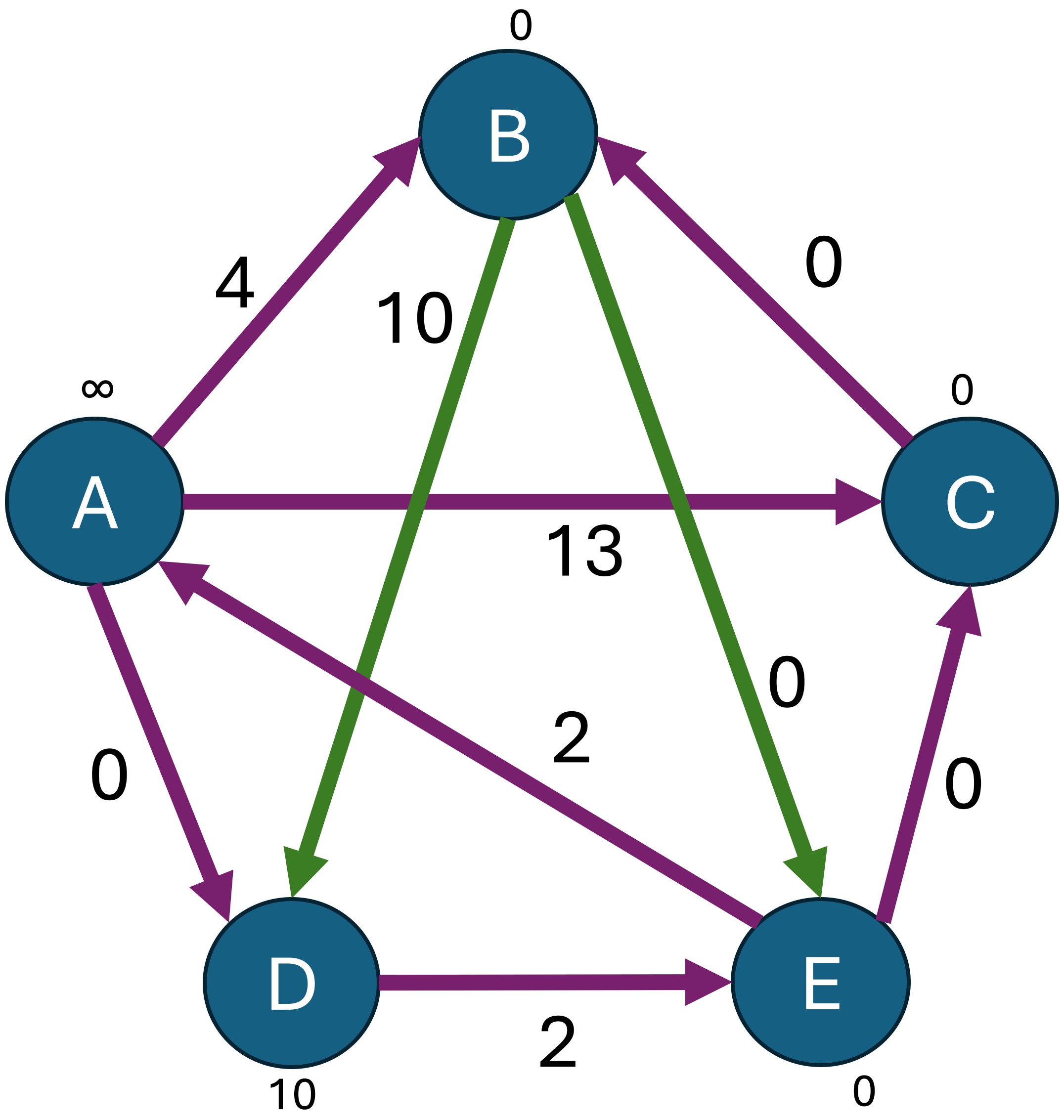
Perform Dijkstra on vertex C



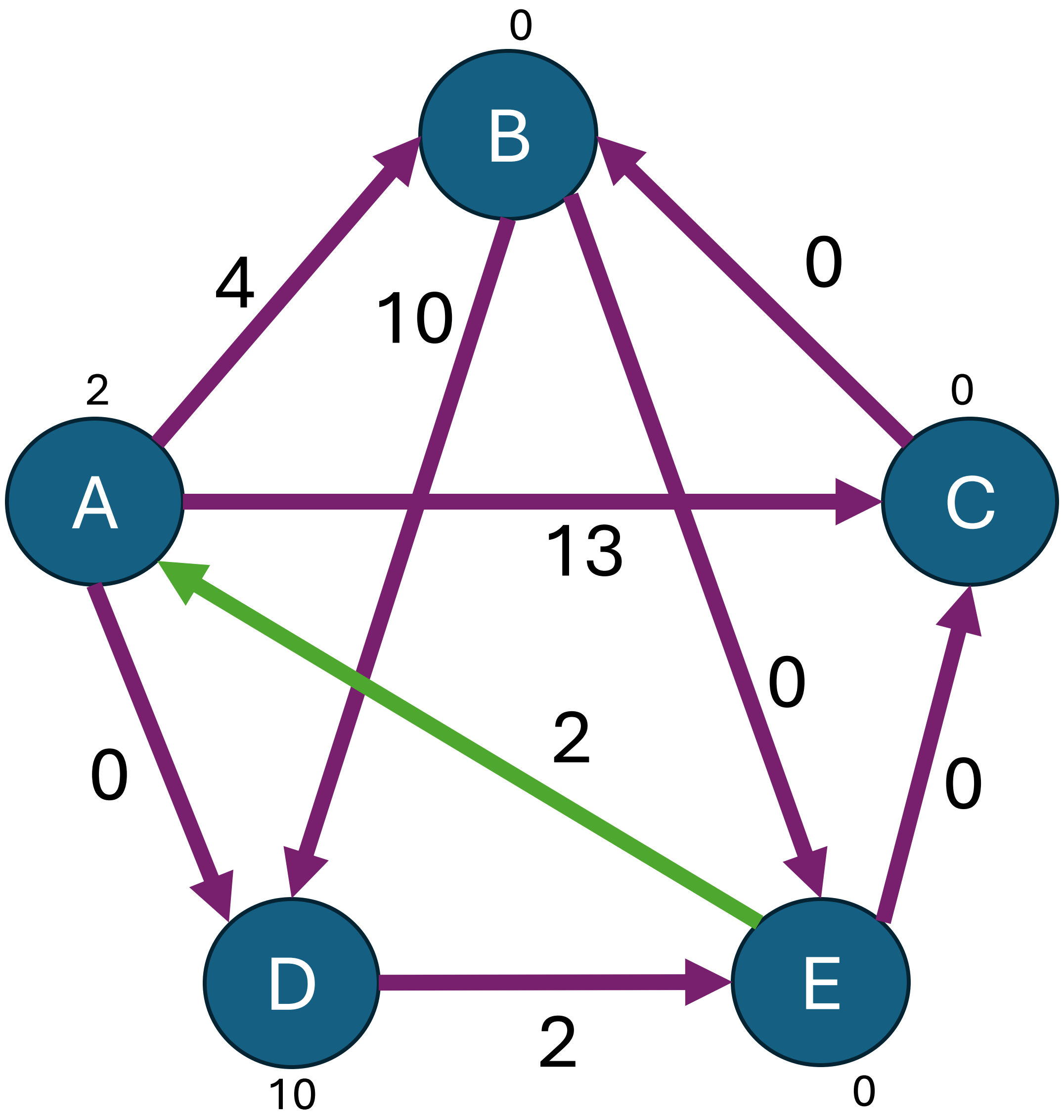
	C	A	B	D	E
C					



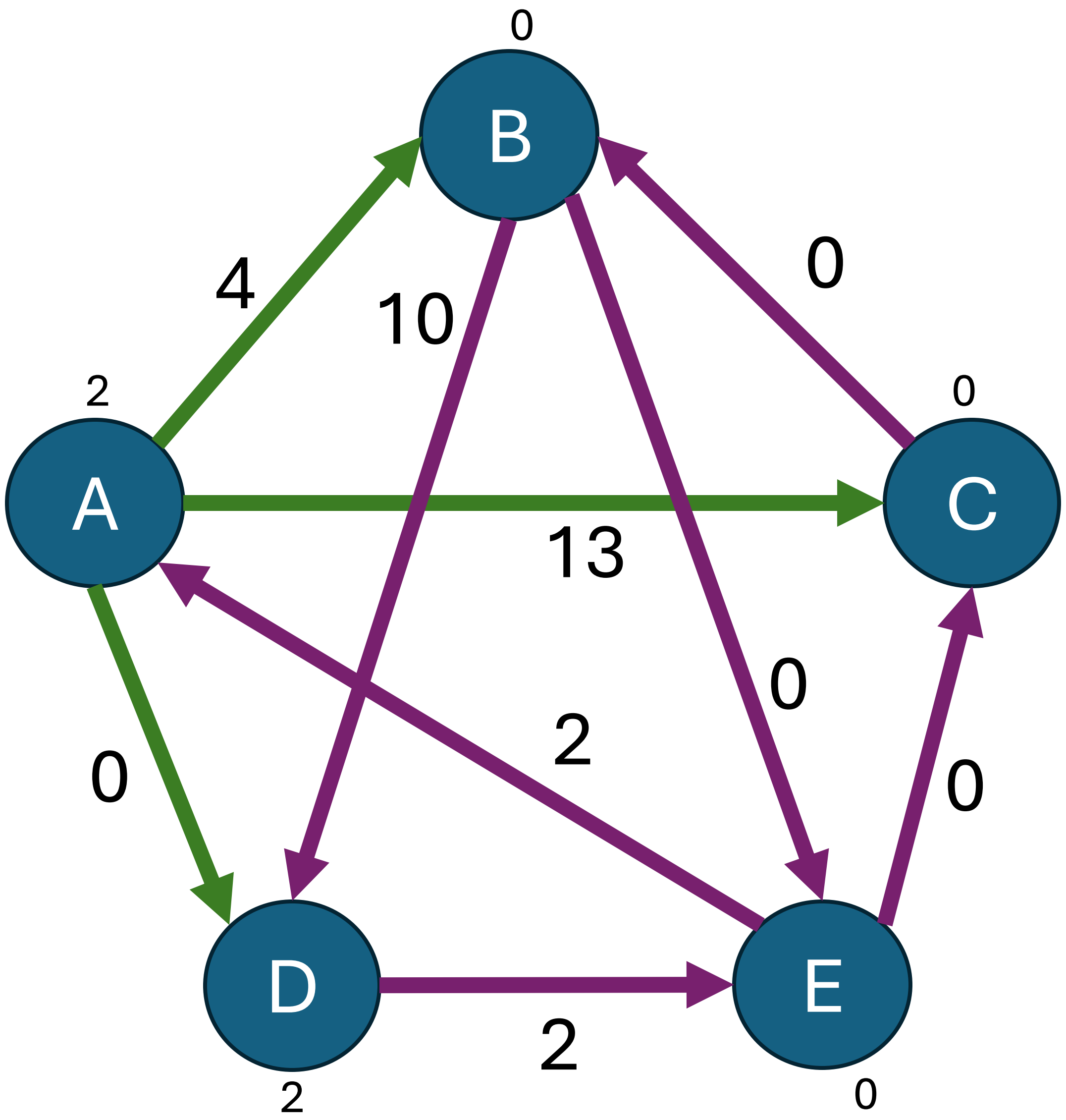
	C	A	B	D	E
C	0		0 C		



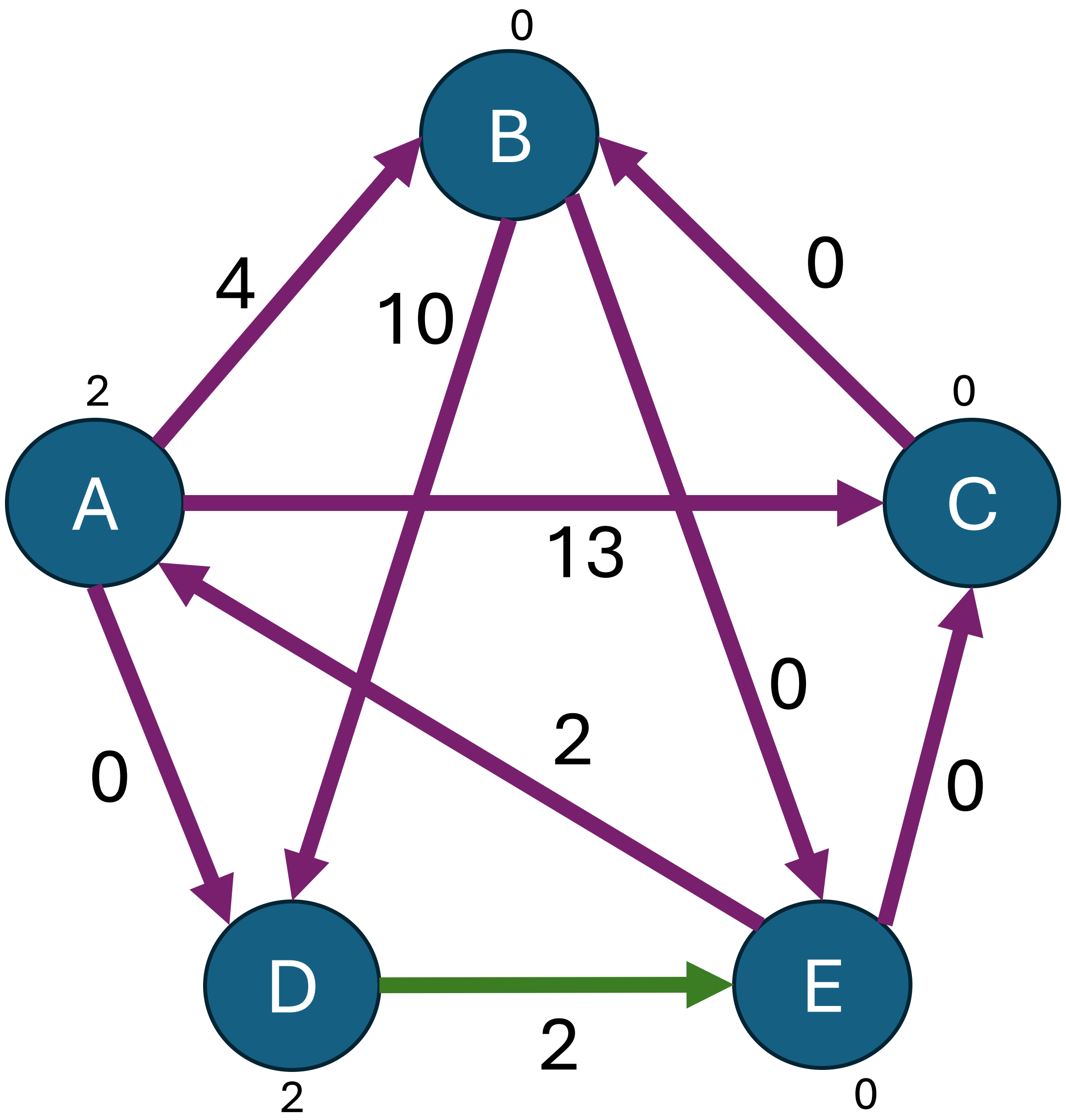
	C	A	B	D	E
C	0		0 C		
B	0		0 C	10 B	0 B



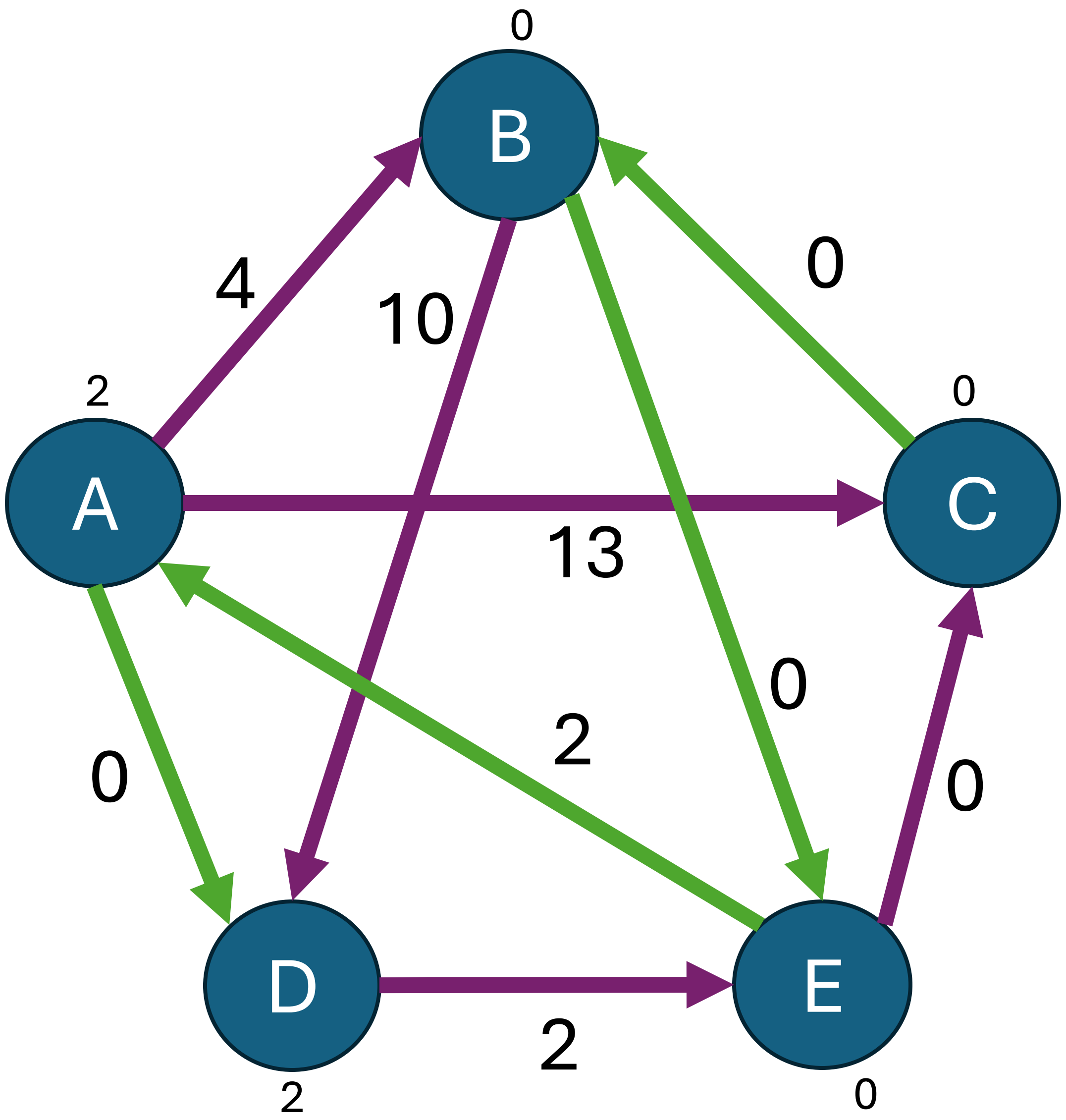
	C	A	B	D	E
C	0		0 C		
B	0		0 C	10 B	0 B
E	0	2 E	0 C	10 B	0 B



	C	A	B	D	E
C	0		0 C		
B	0		0 C	10 B	0 B
E	0	2 E	0 C	10 B	0 B
A	0	2 E	0 C	2 A	0 B



	C	A	B	D	E
C	0		0 C		
B	0		0 C	10 B	0 B
E	0	2 E	0 C	10 B	0 B
A	0	2 E	0 C	2 A	0 B
D	0	2 E	0 C	2 A	0 B



	C	A	B	D	E
C	0		0 C		
B	0		0 C	10 B	0 B
E	0	2 E	0 C	10 B	0 B
A	0	2 E	0 C	2 A	0 B
D	0	2 E	0 C	2 A	0 B