IN2010

Vektede grafer, korteste stier og minimale spenntrær

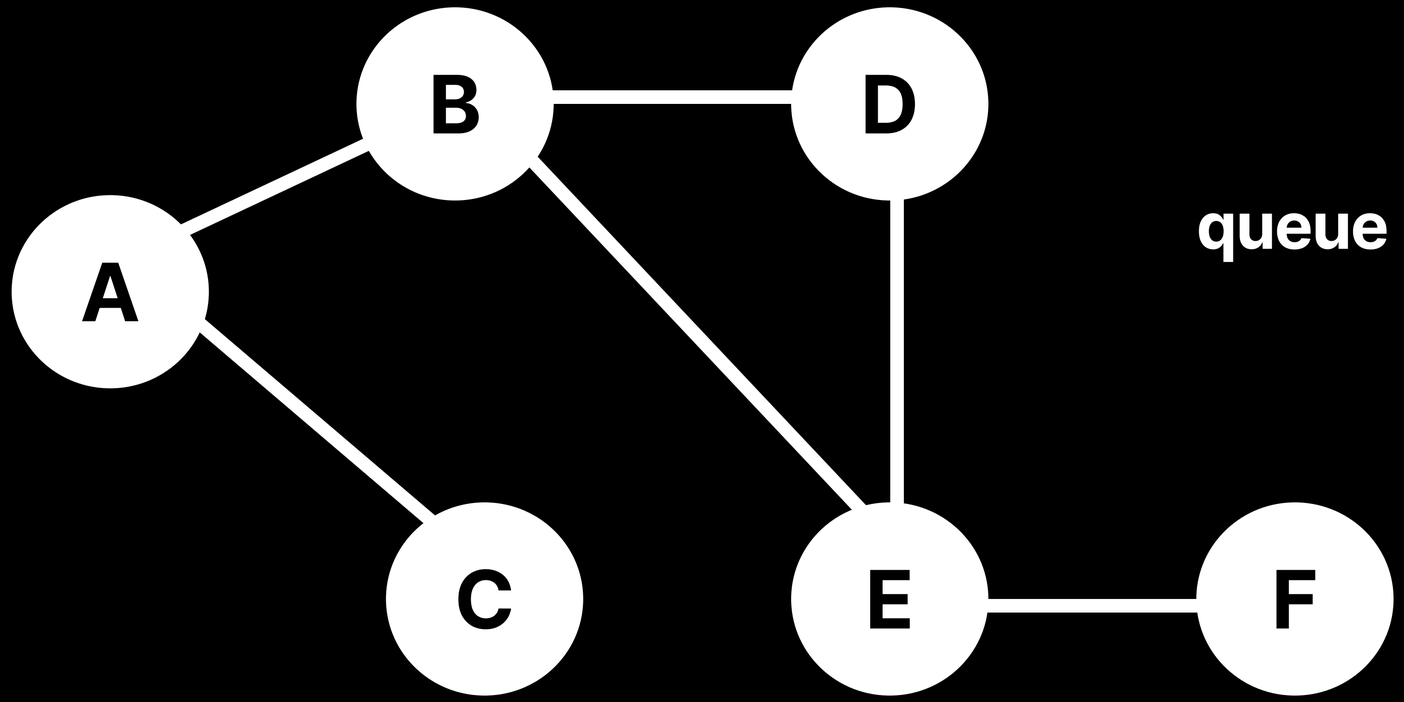
Korteste sti

Hva er en sti?

Sekvens av noder, forbundet av kanter, der ingen noder gjentas

Korteste sti i en uvekta graf

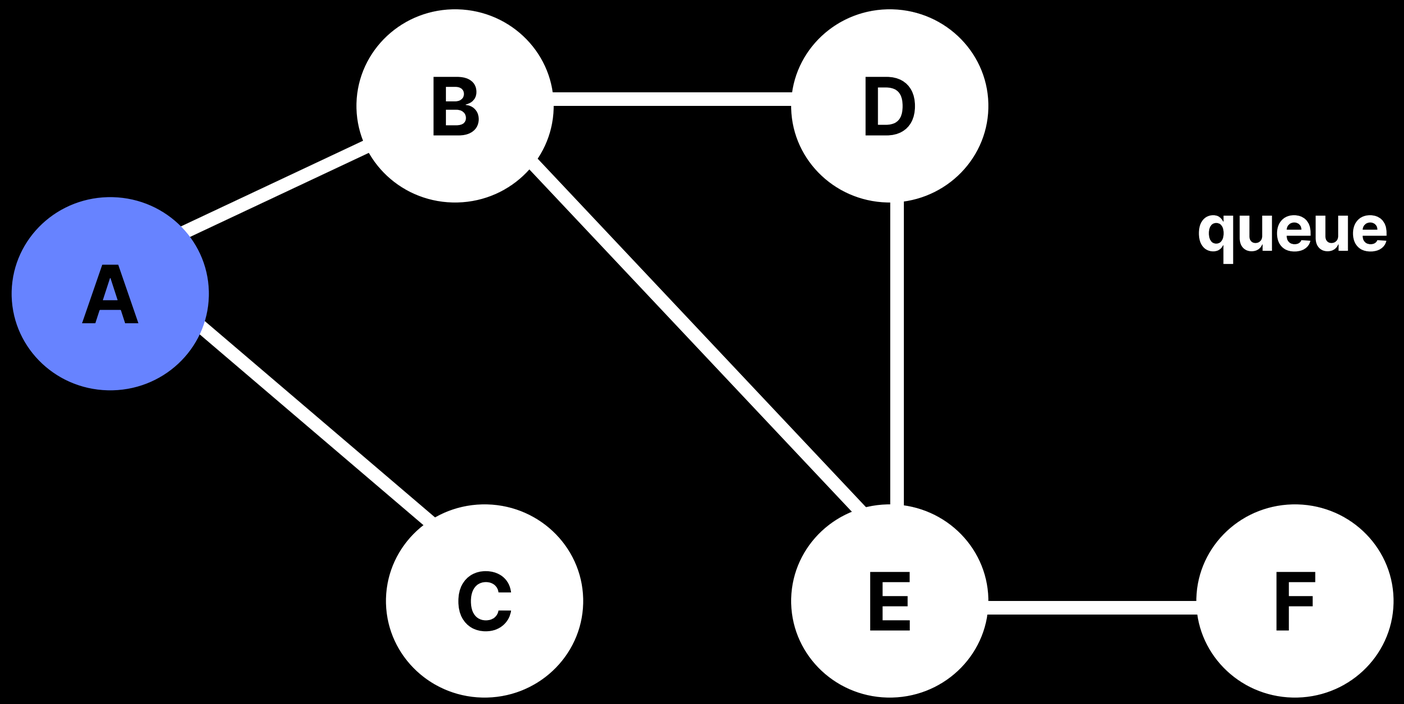
Bredde først søk



queue

Korteste sti i en uvekta graf

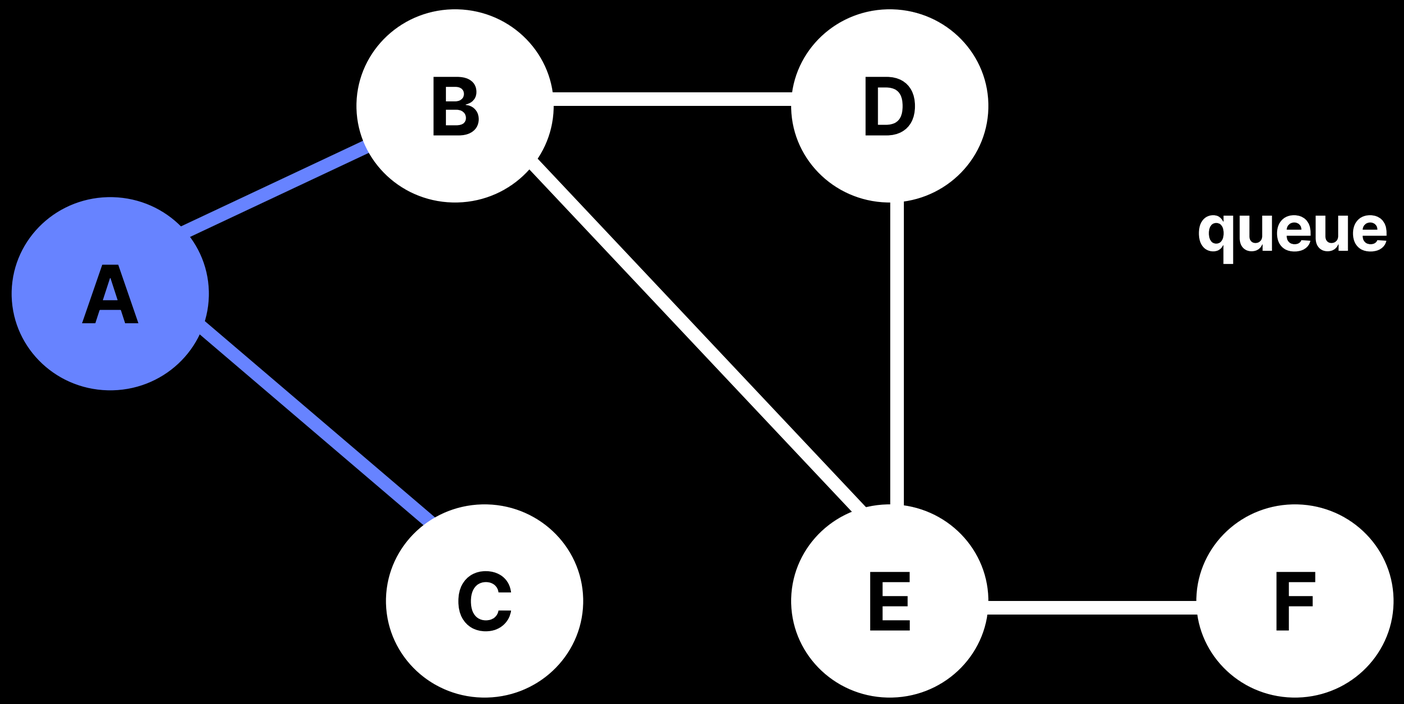
Bredde først søk



queue A

Korteste sti i en uvekta graf

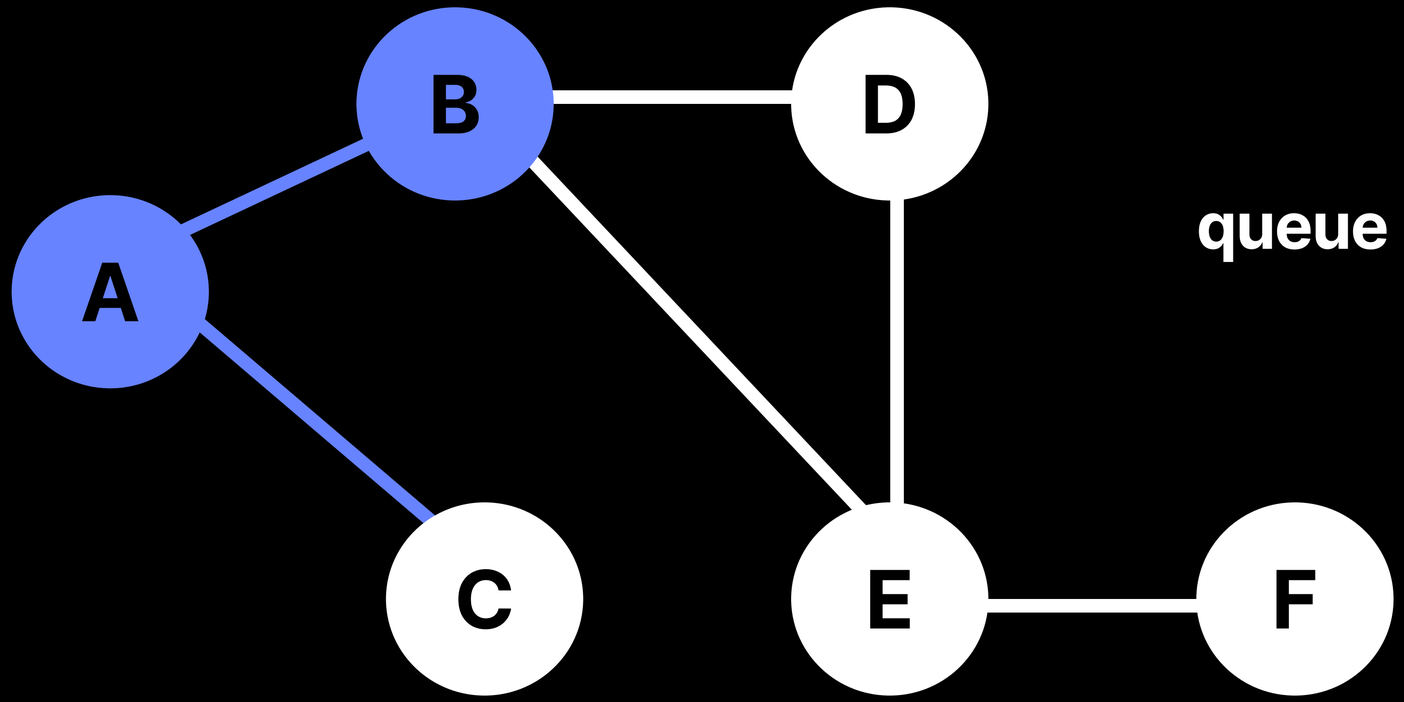
Bredde først søk



queue **A** B C

Korteste sti i en uvekta graf

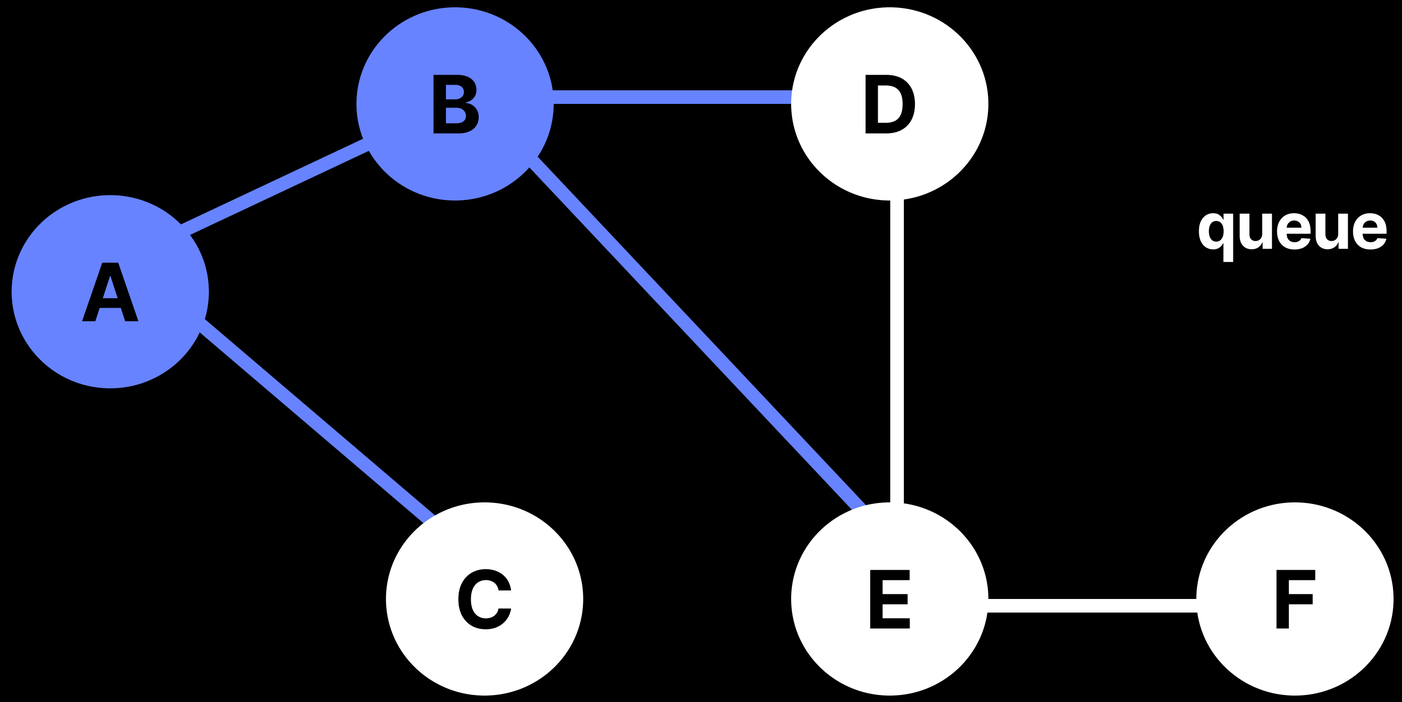
Bredde først søk



queue A **B** C

Korteste sti i en uvekta graf

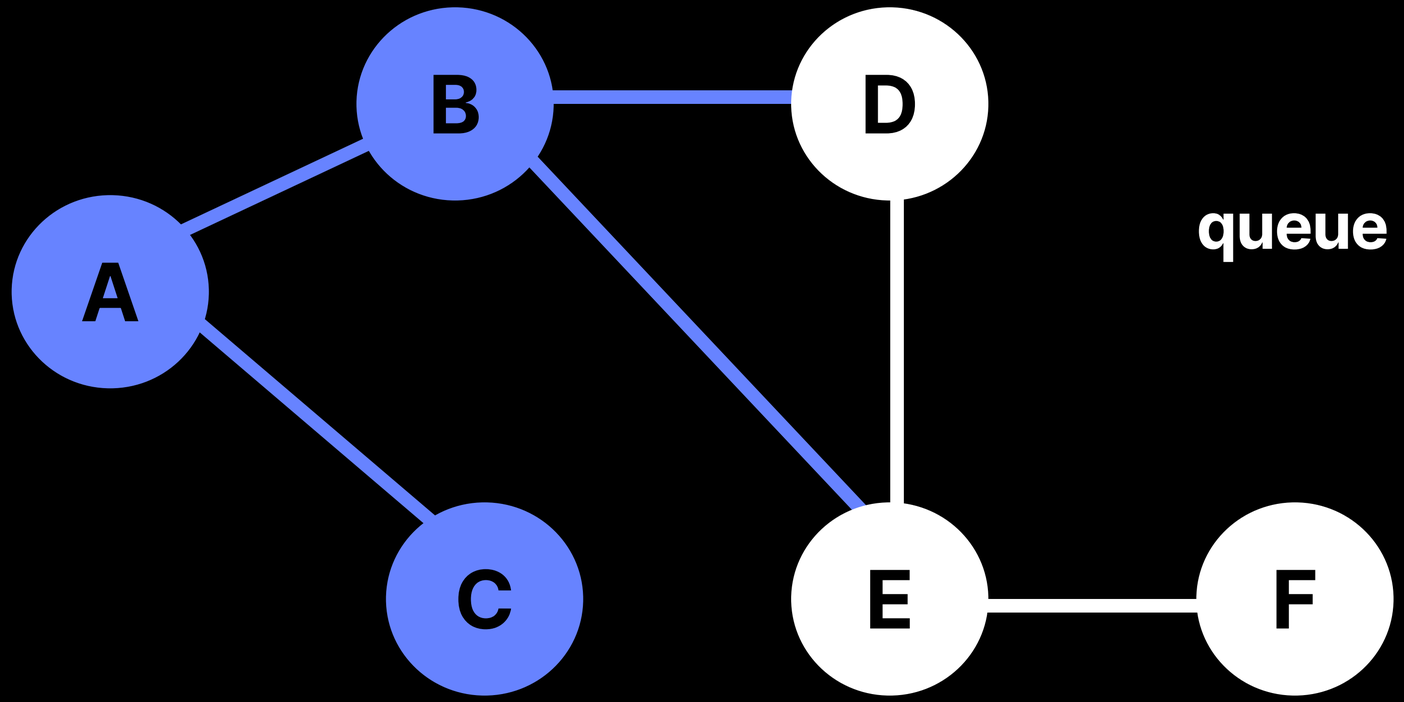
Bredde først søk



queue A **B** C D E

Korteste sti i en uvekta graf

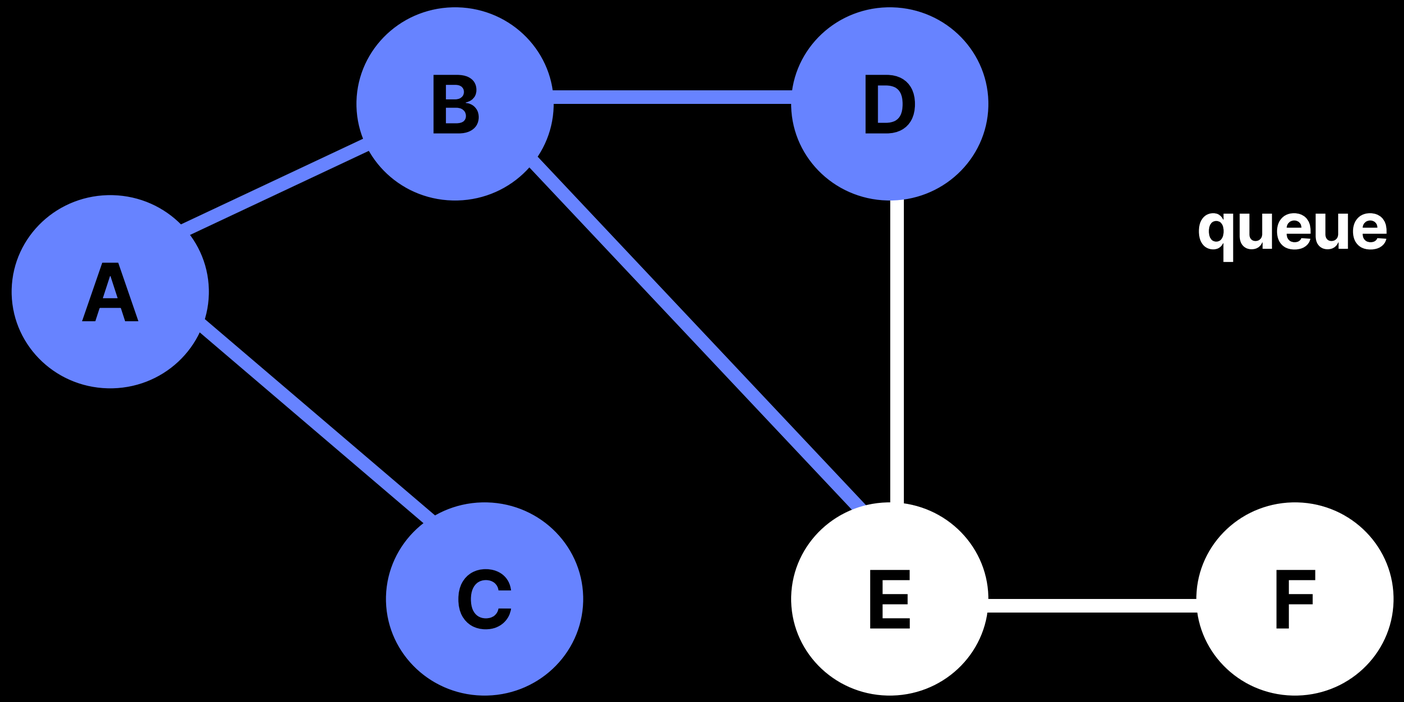
Bredde først søk



queue A B **C** D E

Korteste sti i en uvekta graf

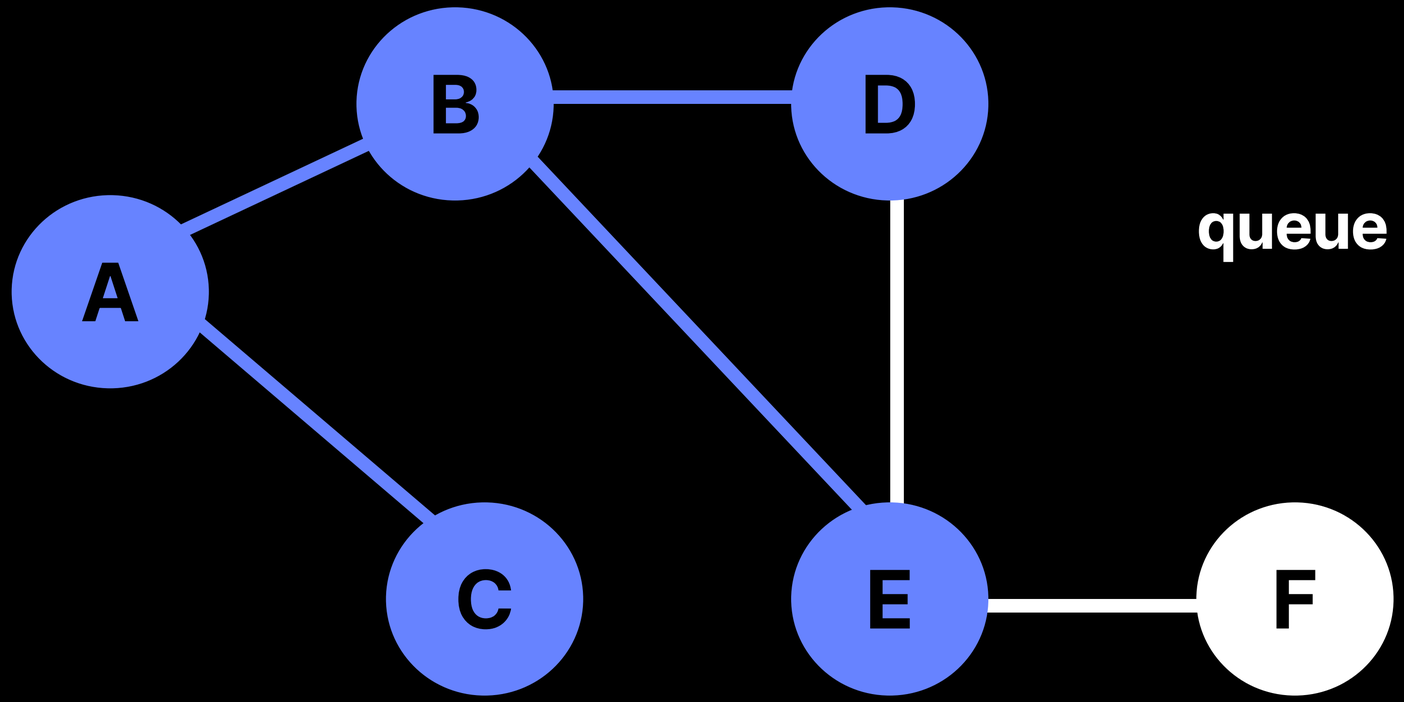
Bredde først søk



queue A B C **D** E

Korteste sti i en uvekta graf

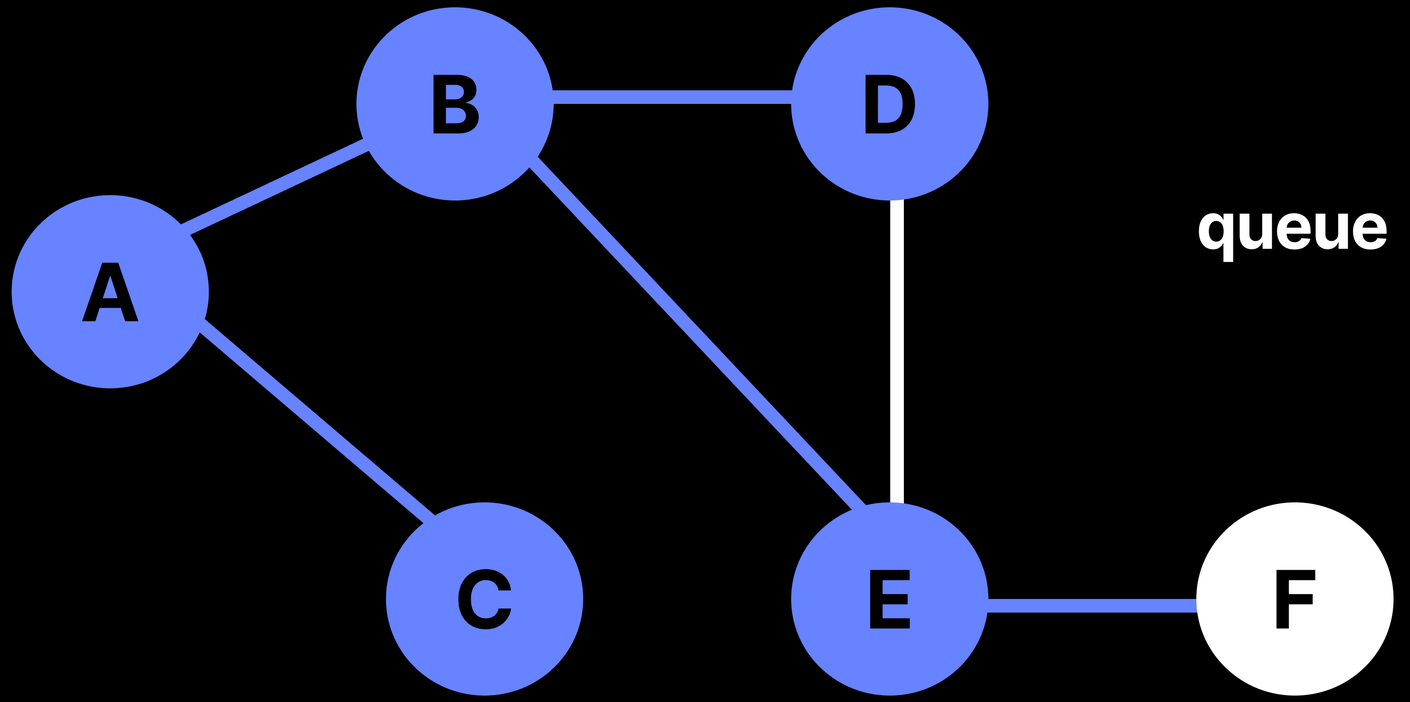
Bredde først søk



queue A B C D **E**

Korteste sti i en uvekta graf

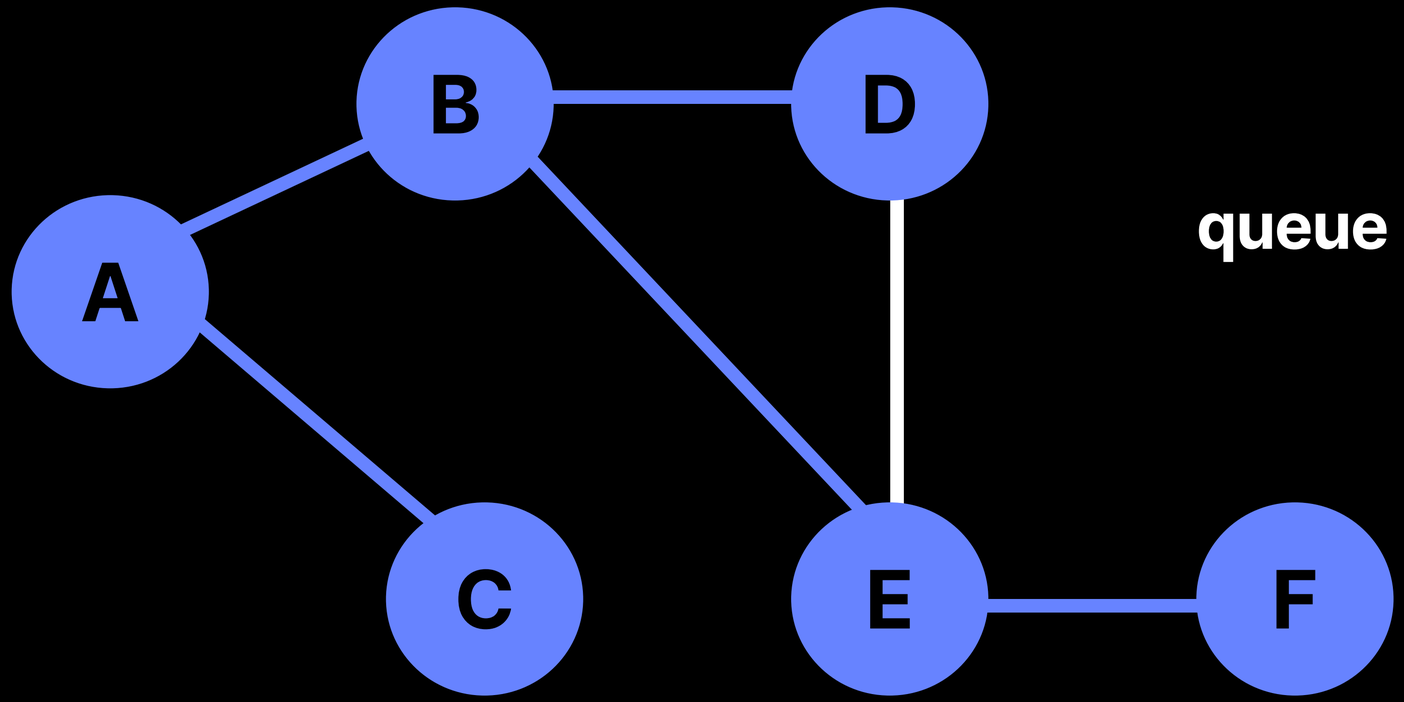
Bredde først søk



queue A B C D **E** F

Korteste sti i en uvekta graf

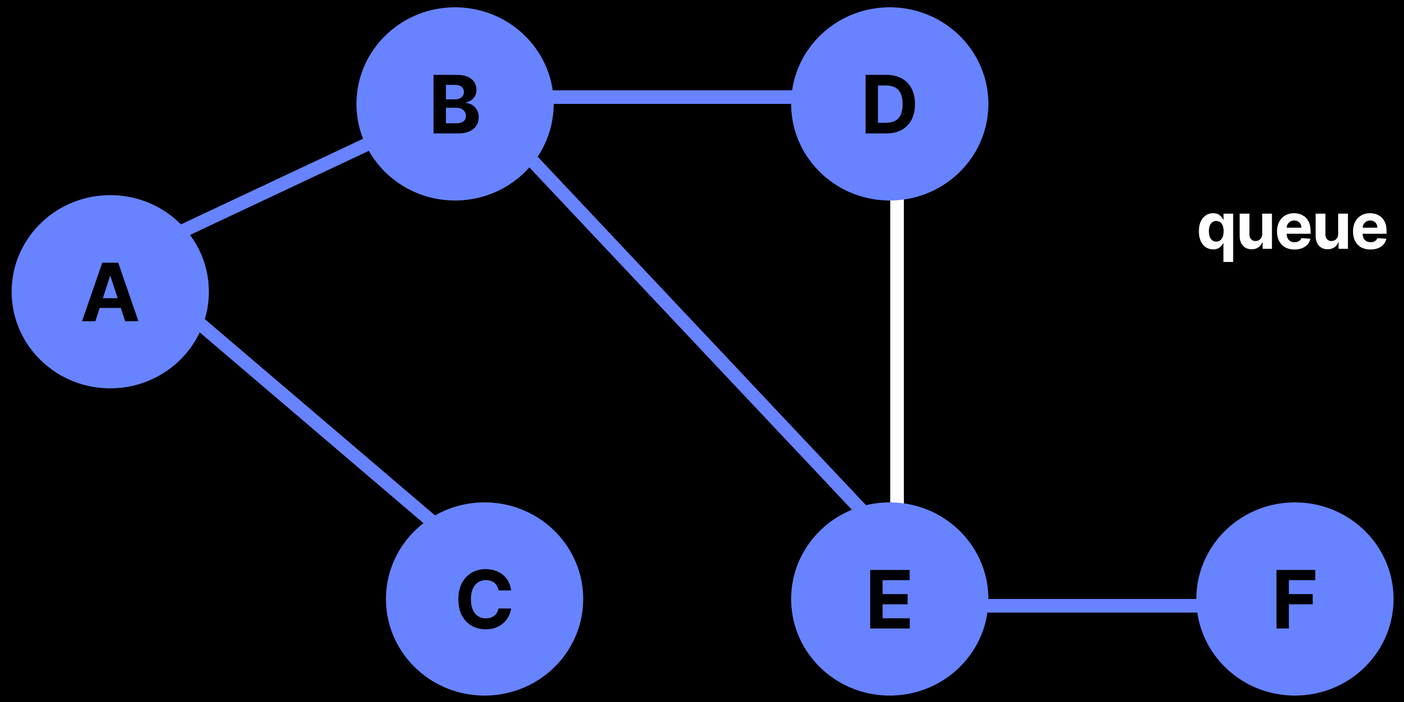
Bredde først søk



queue A B C D E **F**

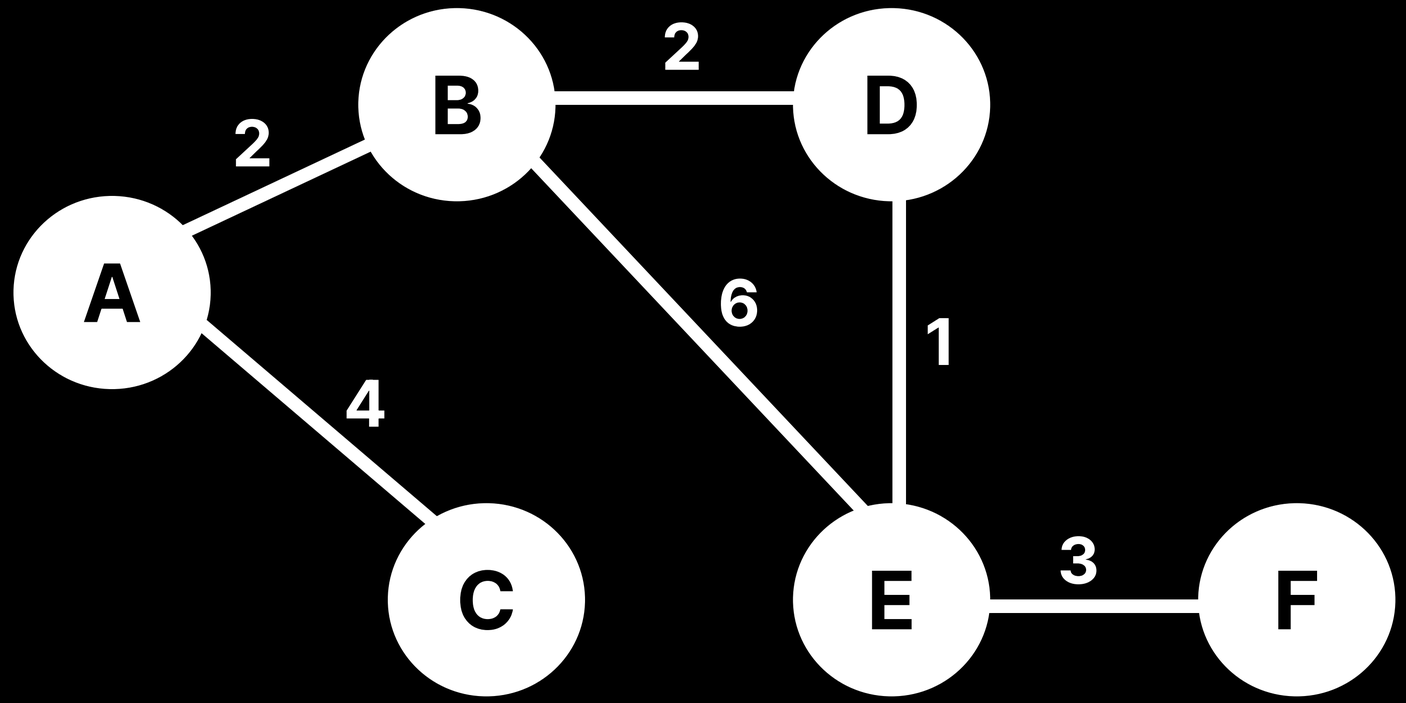
Korteste sti i en uvekta graf

Bredde først søk



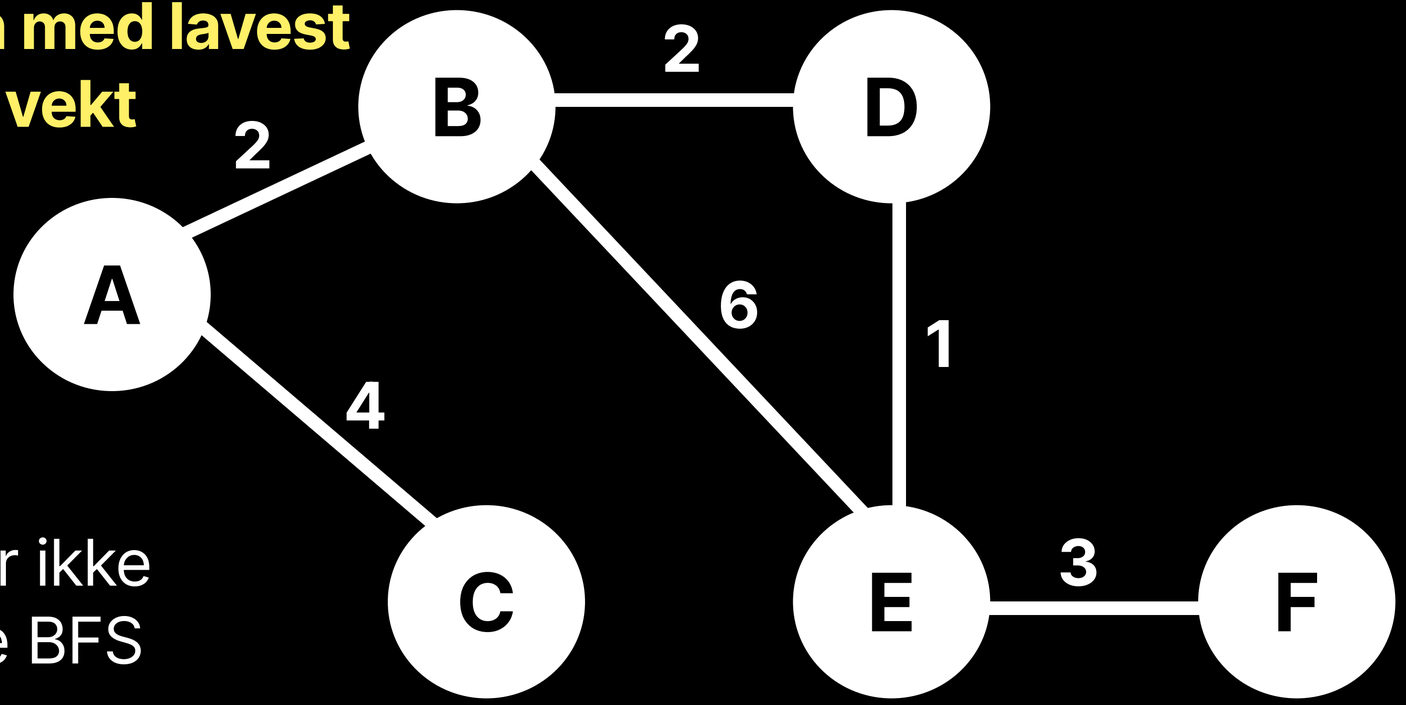
queue A B C D E F

Vektede grafer



Korteste sti i vektede grafer

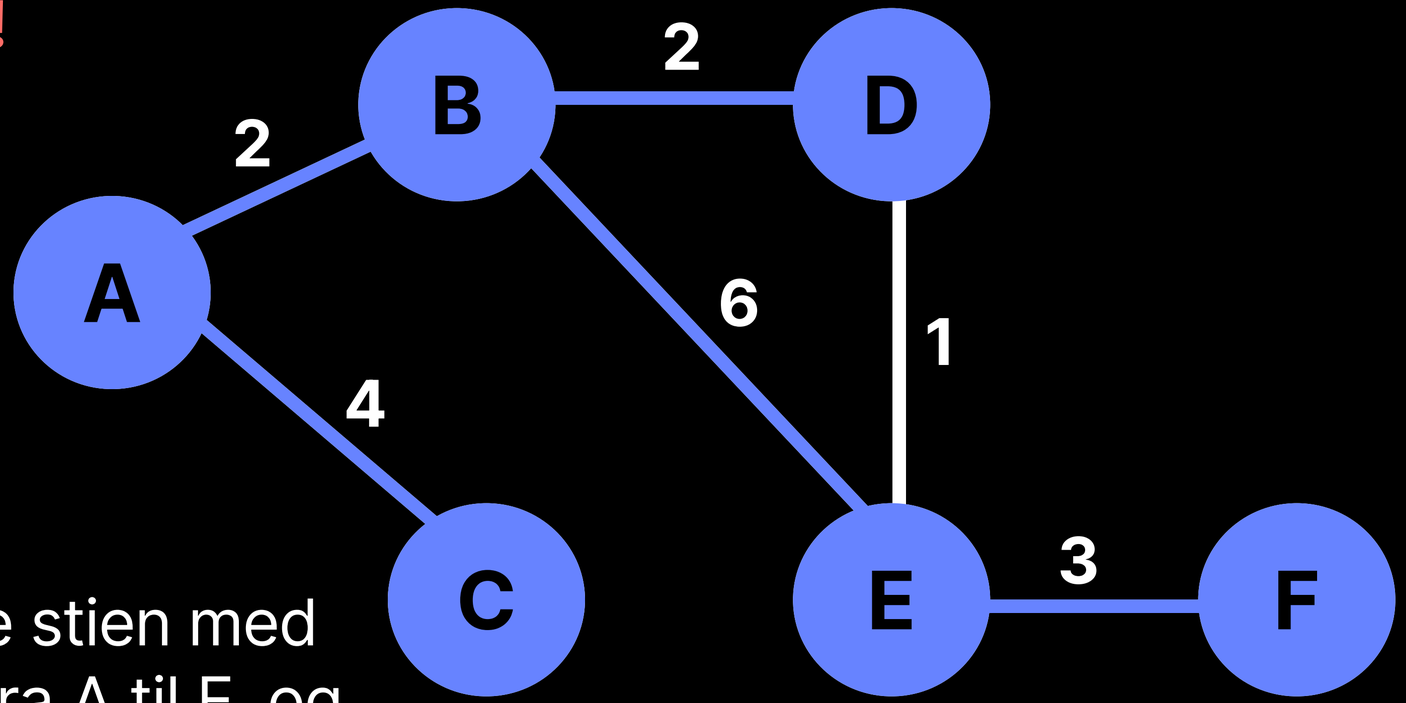
Korteste sti i en vekted graf er stien med lavest akkumulert vekt



Vi kan derfor ikke lenger bruke BFS

Korteste sti i vektede grafer

BFS ville ikke gitt oss korteste sti!



Dette er ikke stien med lavest vekt fra A til E, og dermed ikke korteste sti

Korteste sti i vektede grafer

Dijkstra

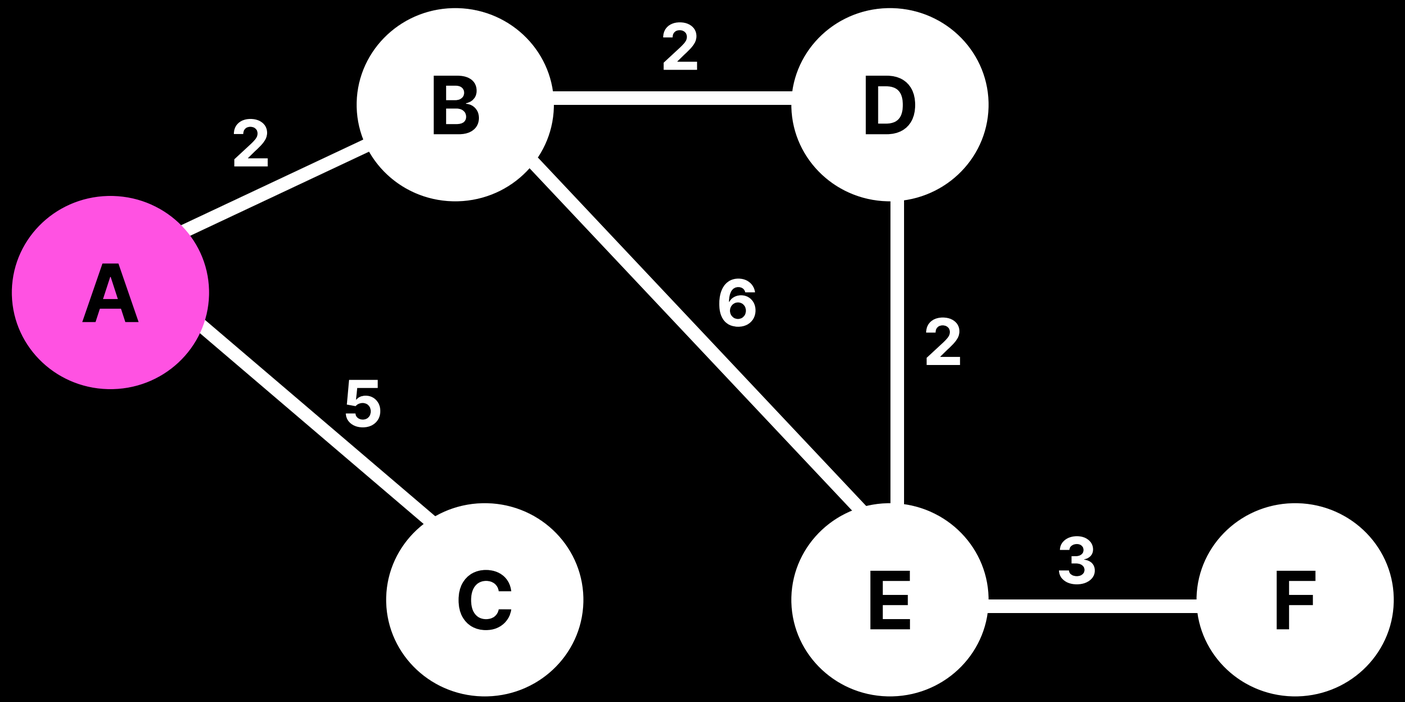
Når vi vet eller kan anta at det **ikke** finnes kanter med negativ vekt

Bellman-Ford

Når det finnes kanter med negativ vekt

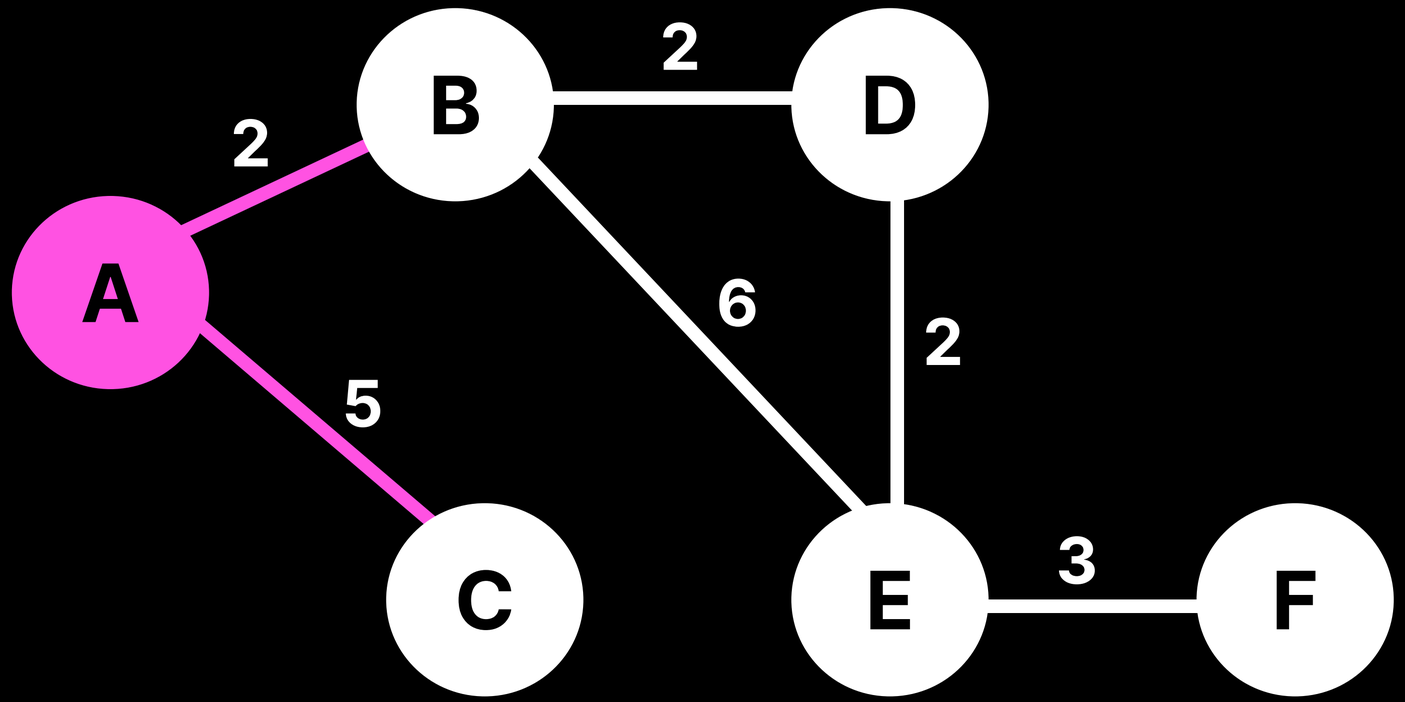
Begge algoritmene fungerer både for rettede og urettede grafer

Dijkstra



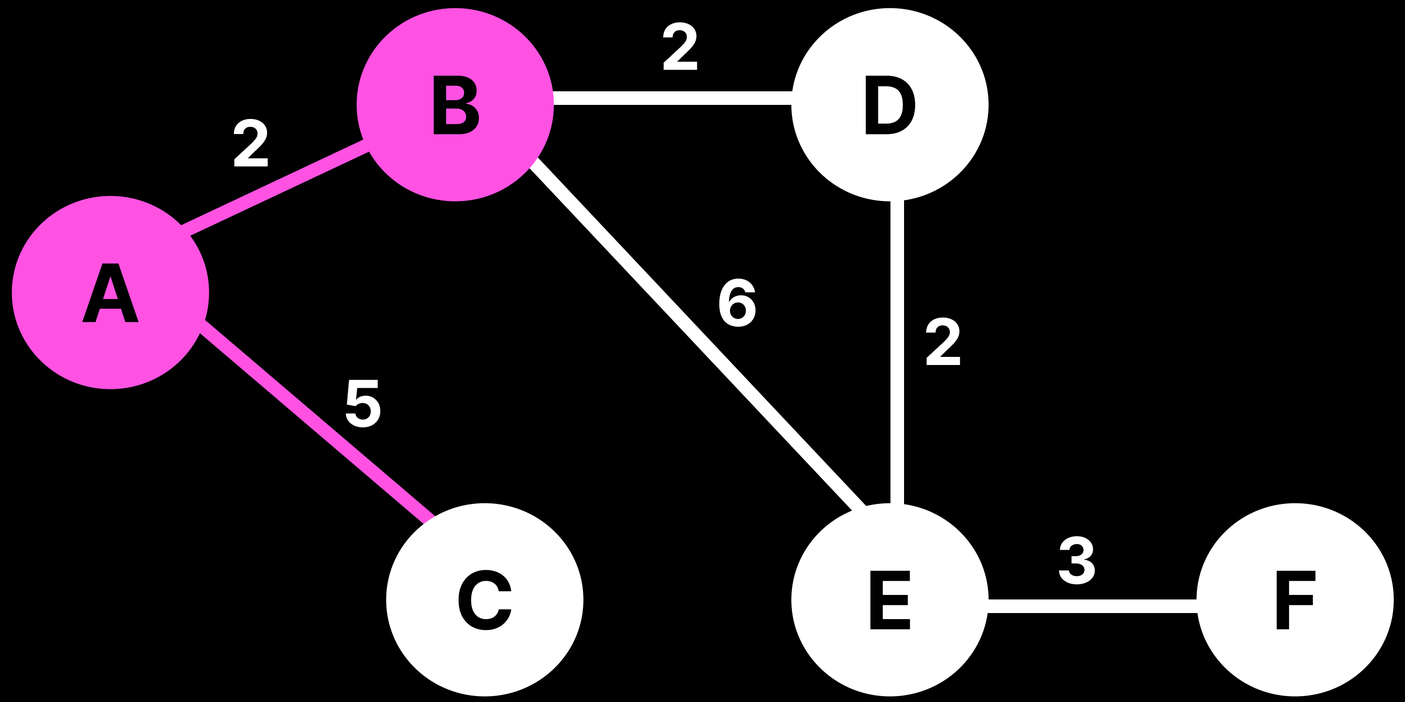
queue	distance
A	0
B	∞
C	∞
D	∞
E	∞
F	∞

Dijkstra



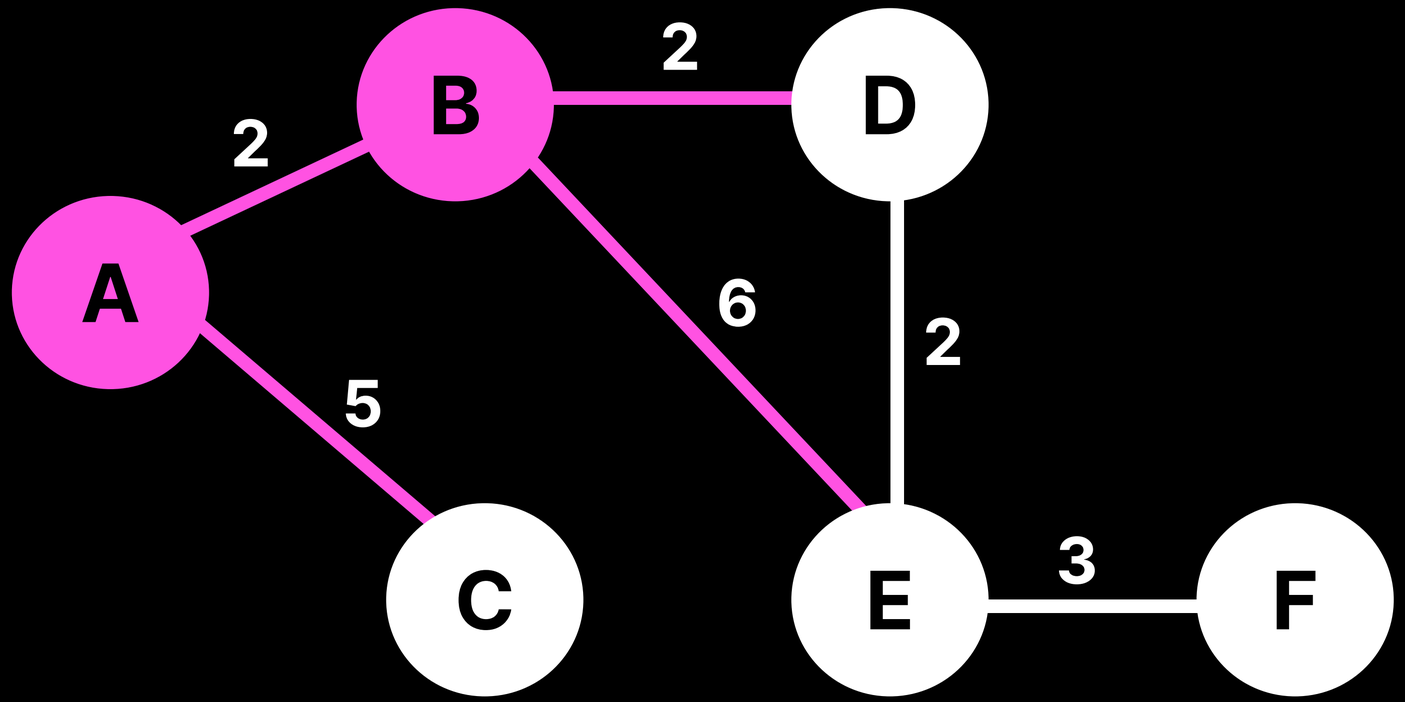
queue	distance
A	0
B	2
C	5
D	∞
E	∞
F	∞

Dijkstra



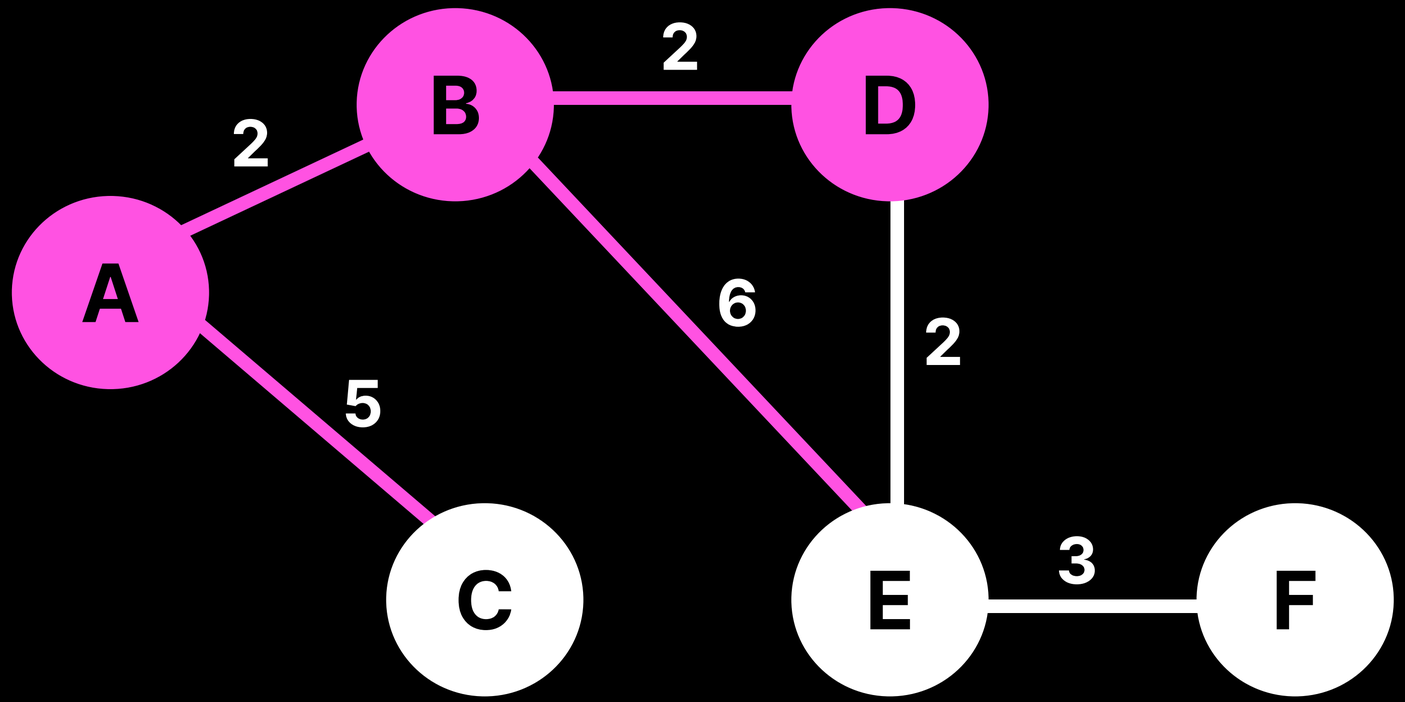
queue	distance
A	0
B	2 den minste
C	5
D	∞
E	∞
F	∞

Dijkstra



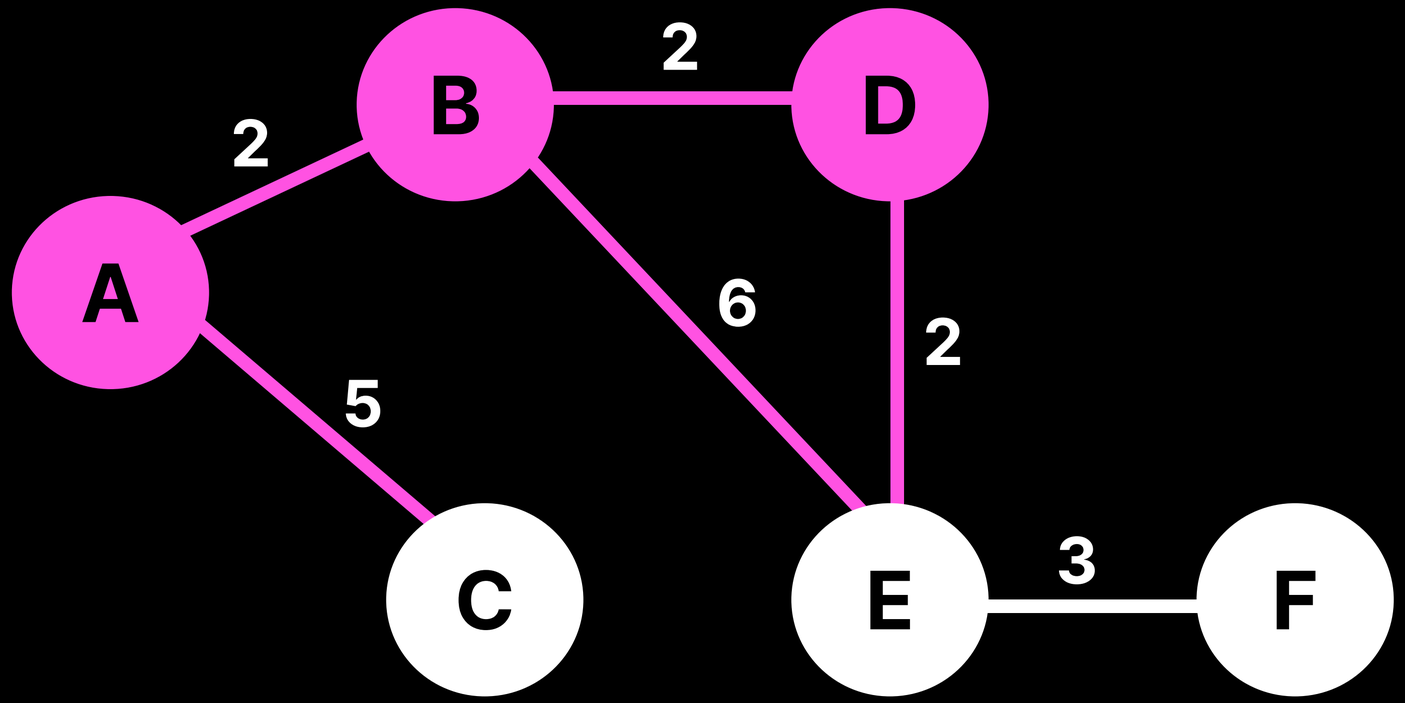
queue	distance
A	0
B	2
C	5
D	2+2
E	2+6
F	∞

Dijkstra



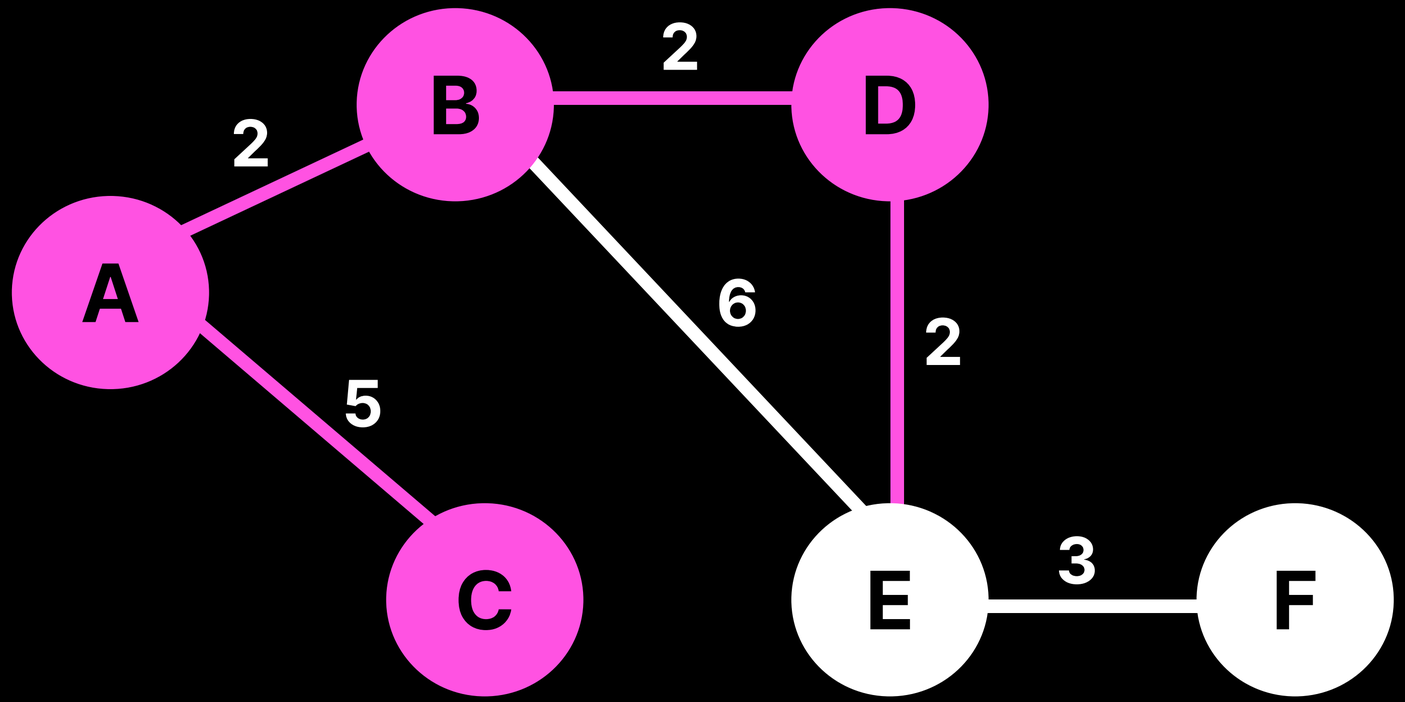
queue	distance
A	0
B	2
C	5
D	4 den minste
E	8
F	∞

Dijkstra



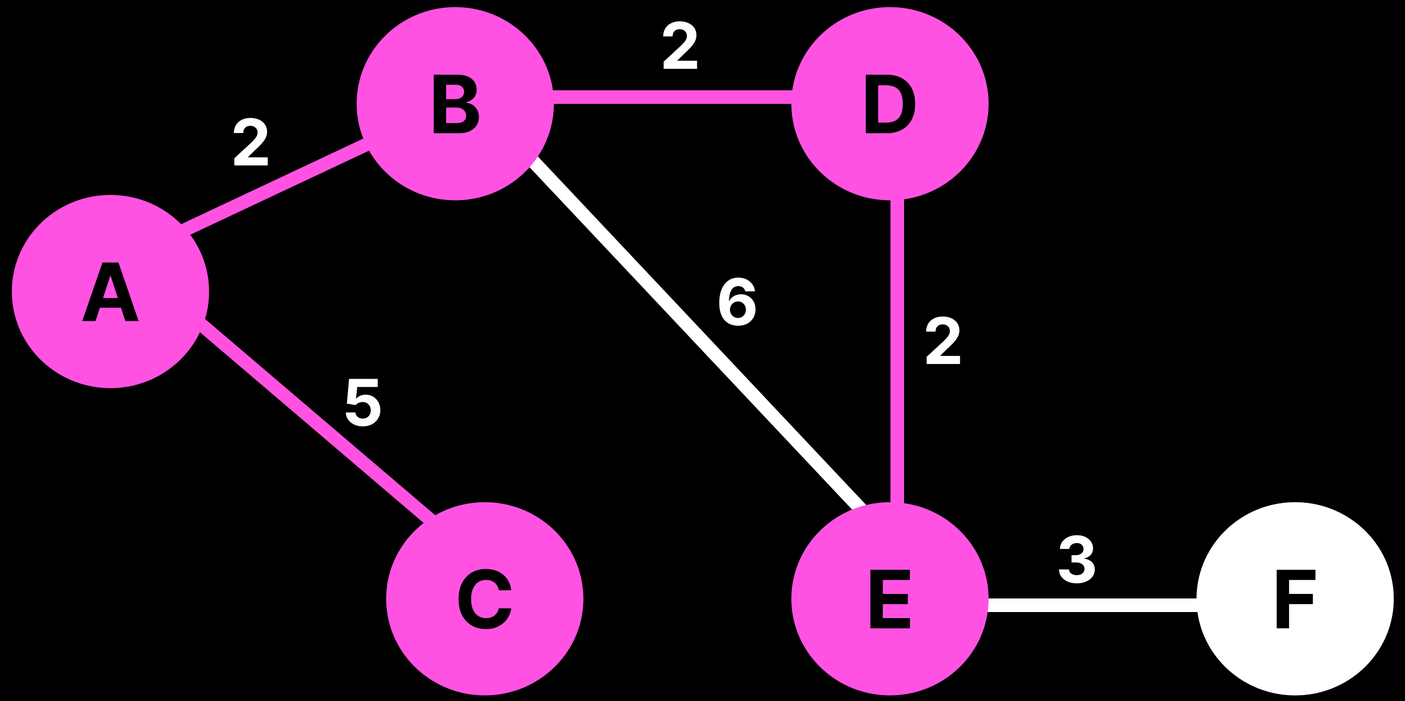
queue	distance
A	0
B	2
C	5
D	4
E	8>6 denne oppdateres!
F	∞

Dijkstra



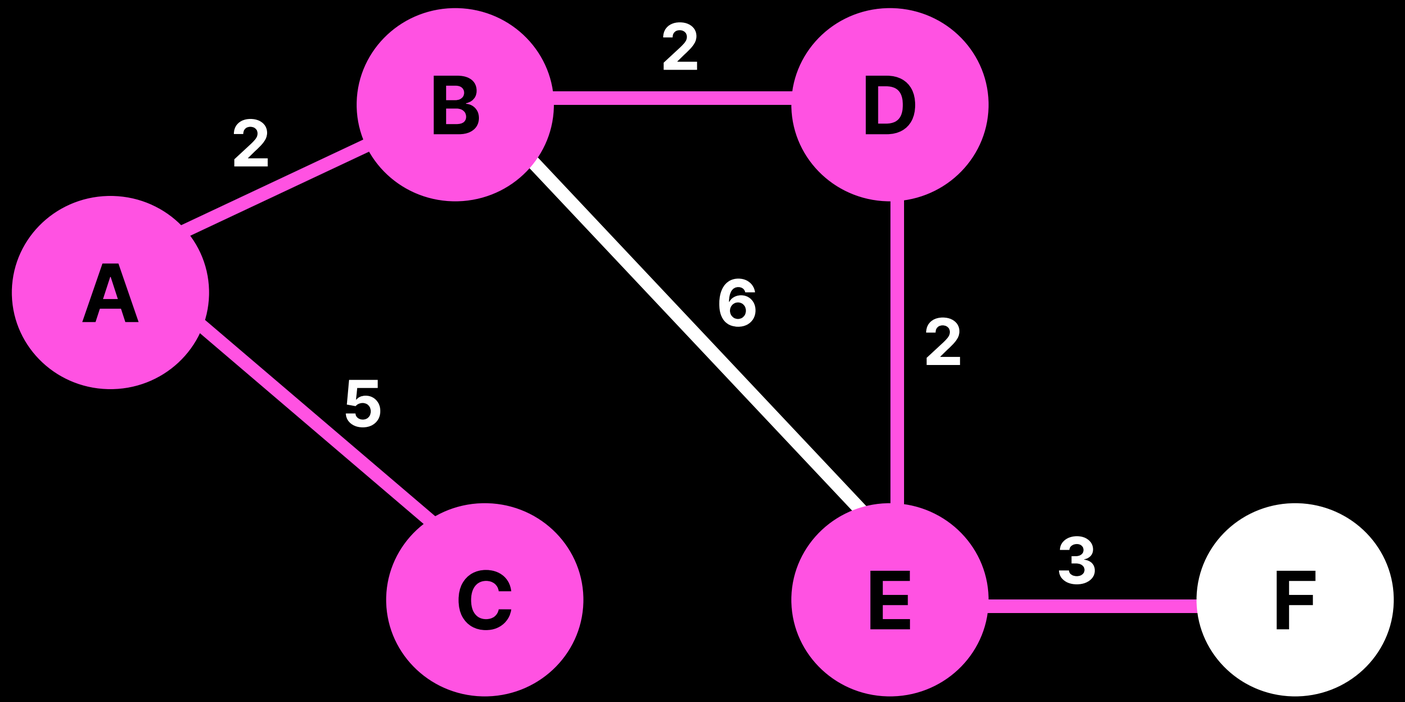
queue	distance
A	0
B	2
C	5 den minste
D	4
E	6
F	∞

Dijkstra



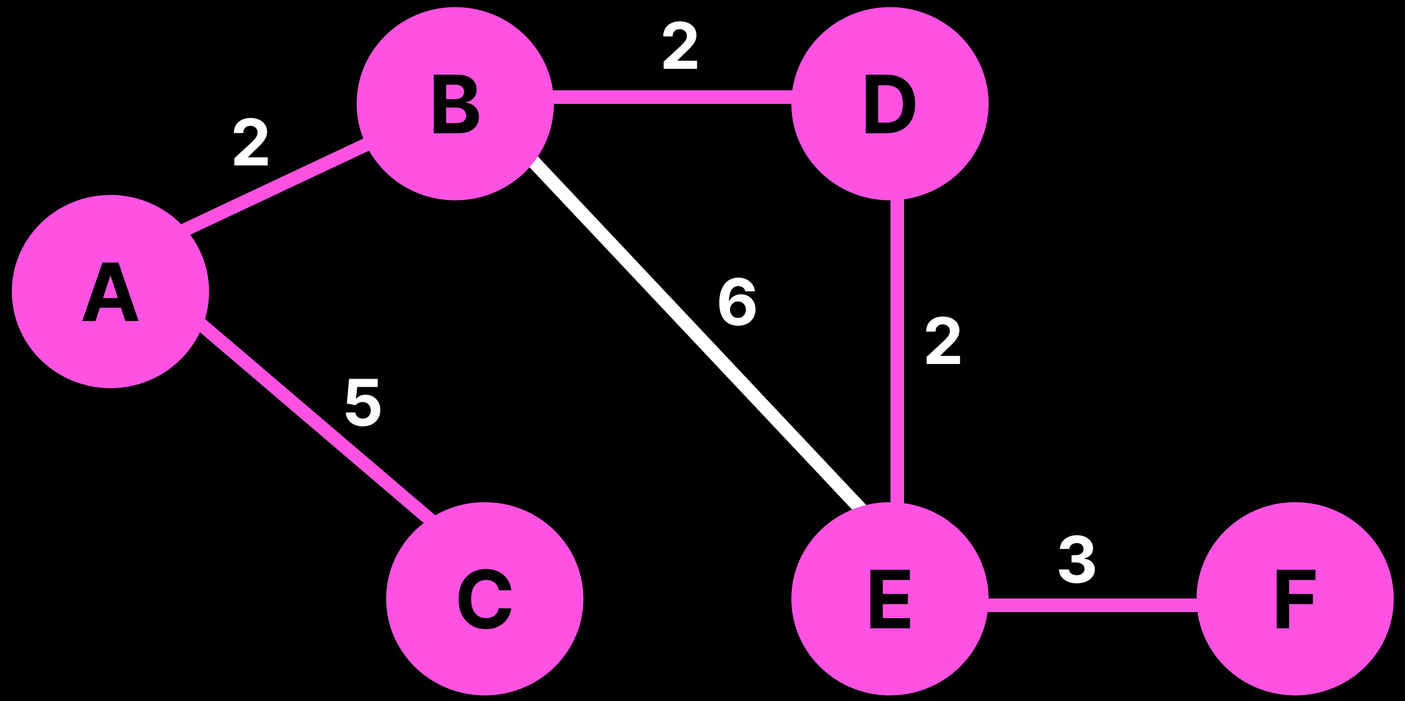
queue	distance
A	0
B	2
C	5
D	4
E	6 den minste
F	∞

Dijkstra



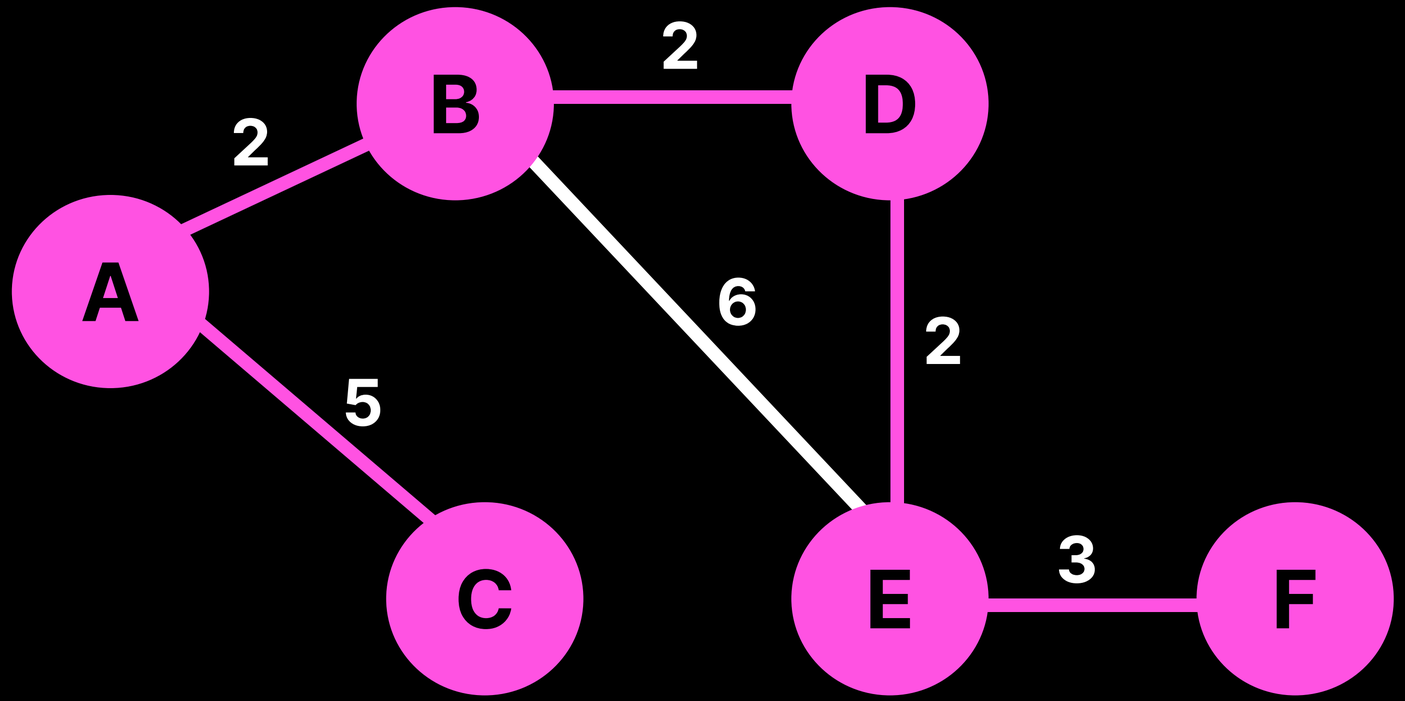
queue	distance
A	0
B	2
C	5
D	4
E	6
F	2+2+2+3

Dijkstra



queue	distance
A	0
B	2
C	5
D	4
E	6
F	9

Dijkstra



queue	distance
A	0
B	2
C	5
D	4
E	6
F	9

Kjøretidsanalysen til dijkstra

```
1  Procedure Dijkstra(G, s)
2      queue ← empty priority queue
3      dist ← empty map
4      for v ∈ V do
5          dist[v] ← ∞
6          Insert(queue, v) with priority ∞
7      dist[s] ← 0
8      DecreasePriority(queue, s, 0)
9
10     while queue is not empty do
11         u ← RemoveMin(queue)
12         for (u, v) ∈ E do
13             c ← dist[u] + w(u, v)
14             if c < dist[v] then
15                 dist[v] ← c
16                 DecreasePriority(queue, v, c)
17     return dist
```

Legger til alle nodene i queue

O(|V|)

Denne er fra forelesning gitt at er logaritmisk O(log(|V|))

For alle nodene gjør vi RemoveMin,
og for alle kantene gjør vi DecreasePriority.
Både RemoveMin og DecreasePriority er
logaritmiske. Vi gjør altså noe logaritmisk for
alle noder og kanter.

O((|E|+|V|) * log(|V|)

Kjøretidsanalysen til dijkstra

```
1  Procedure Dijkstra(G, s)
2      queue ← empty priority queue
3      dist ← empty map
4      for v ∈ V do
5          dist[v] ← ∞
6          Insert(queue, v) with priority ∞
7      dist[s] ← 0
8      DecreasePriority(queue, s, 0)
9
10     while queue is not empty do
11         u ← RemoveMin(queue)
12         for (u, v) ∈ E do
13             c ← dist[u] + w(u, v)
14             if c < dist[v] then
15                 dist[v] ← c
16                 DecreasePriority(queue, v, c)
17     return dist
```

O(|V|)

O(log(|V|))

O((|E|+|V|) * log(|V|)

O(|V| + log(|V|) + ((|E|+|V|) * log(|V|) + 4)

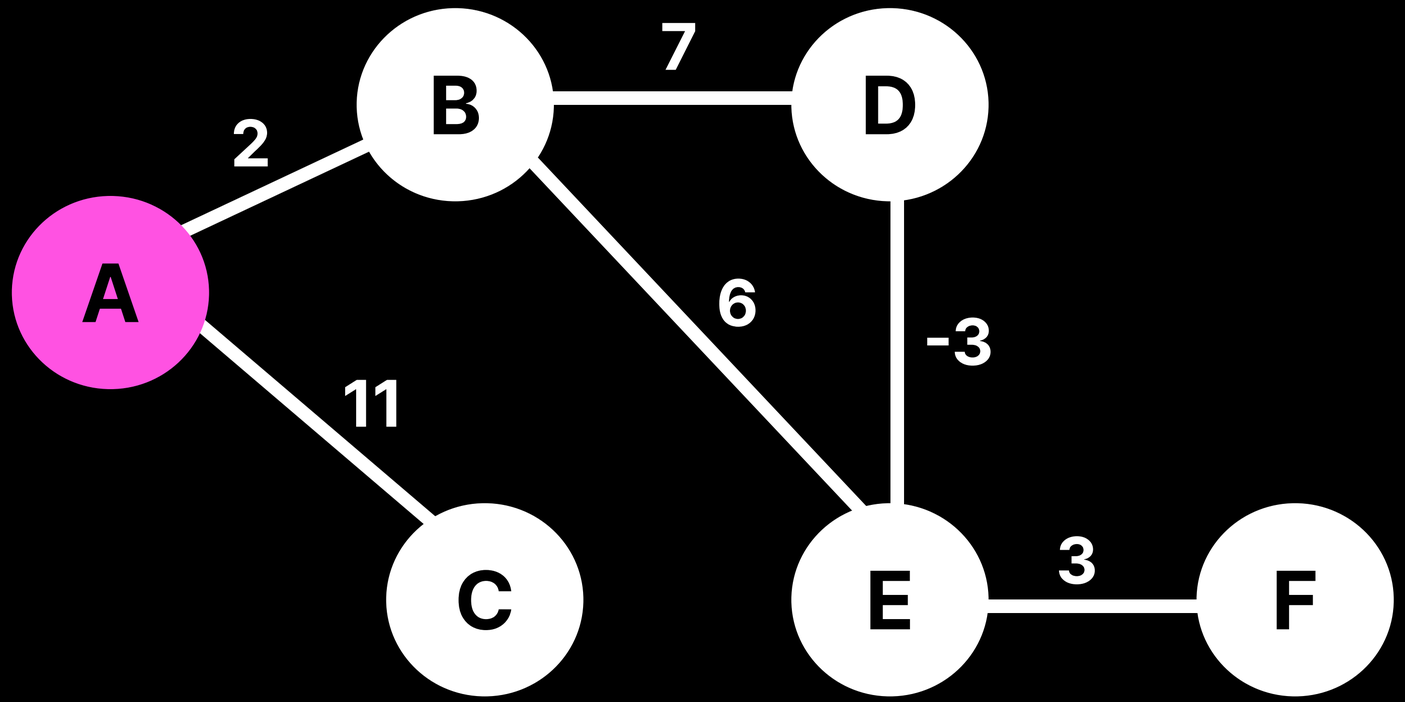
= O((|E|+|V|) * log(|V|)

Fordi ((|E|+|V|) * log(|V|) tar lengst tid av de tre, og store O konverterer fra addisjon (+) til maks (max)

Kan forenkles til $O(|E| \cdot \log(|V|))$ hvis grafen er sammenhengende, ved å implementere dijkstra uten DecreasePriority. Se forelesning.

Bellman-Ford

Negative vekter

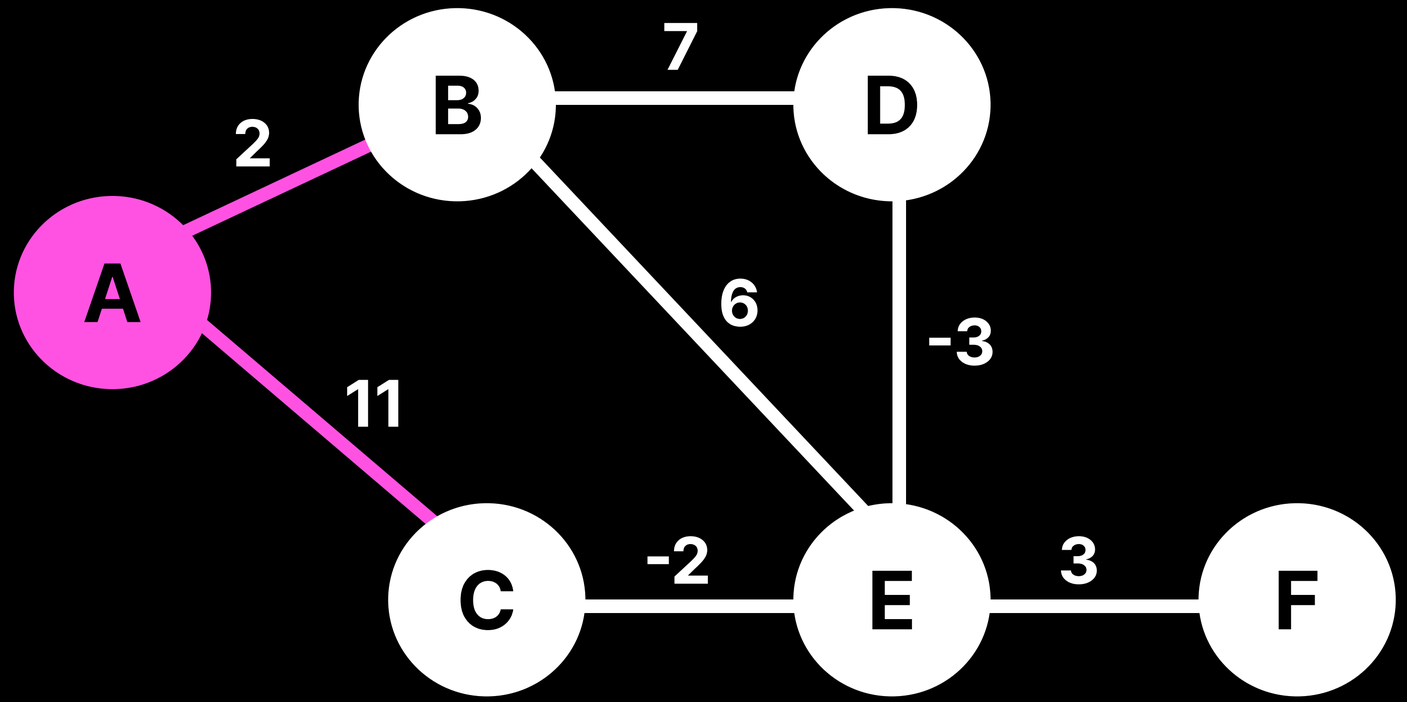


queue	distance
A	0
B	∞
C	∞
D	∞
E	∞
F	∞

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter

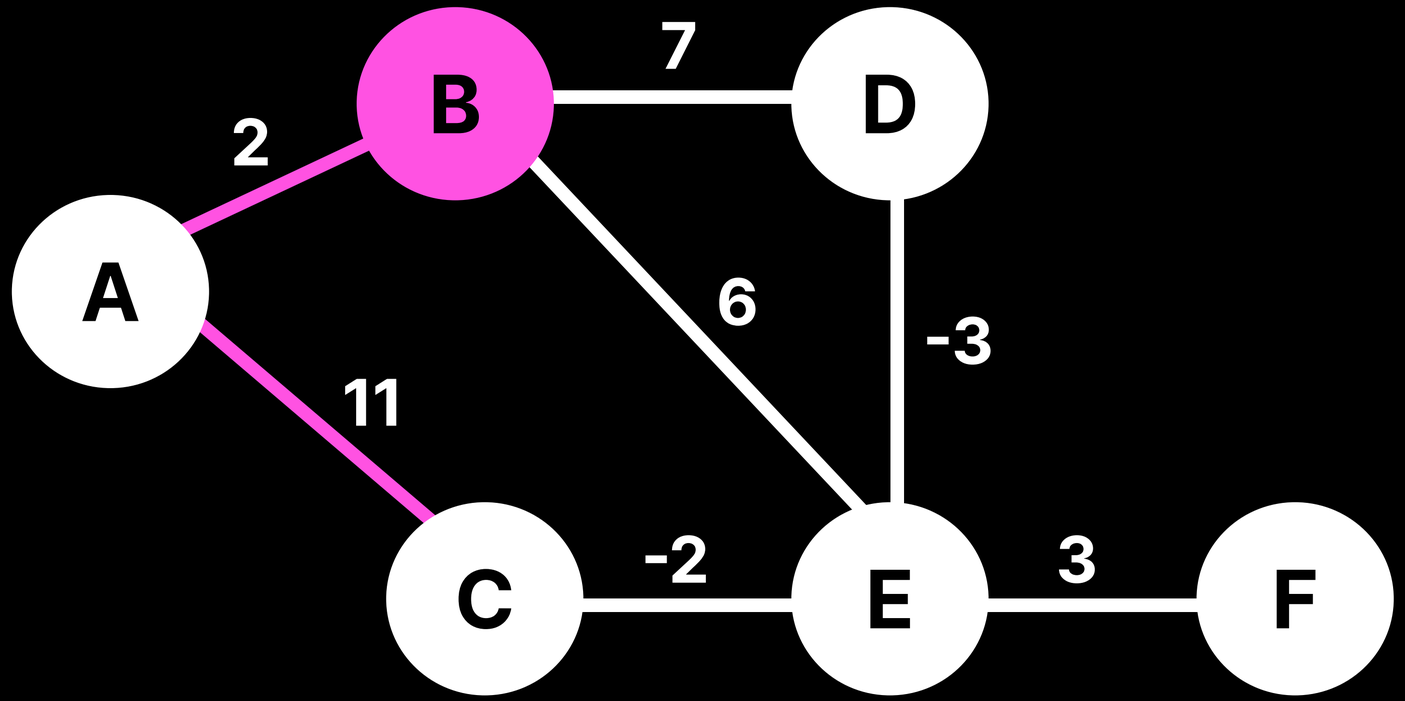


queue	distance
A	0
B	2
C	11
D	∞
E	∞
F	∞

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter

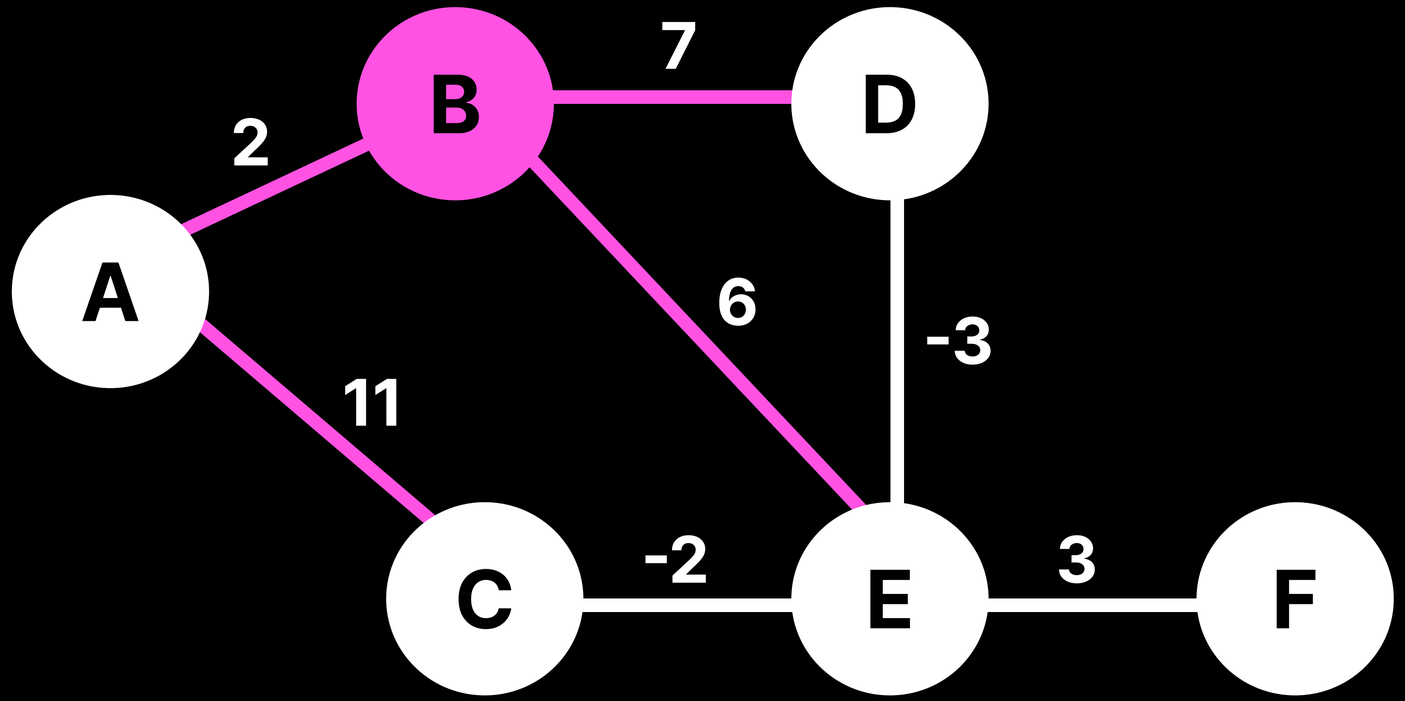


queue	distance
A	0
B	2
C	11
D	∞
E	∞
F	∞

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter

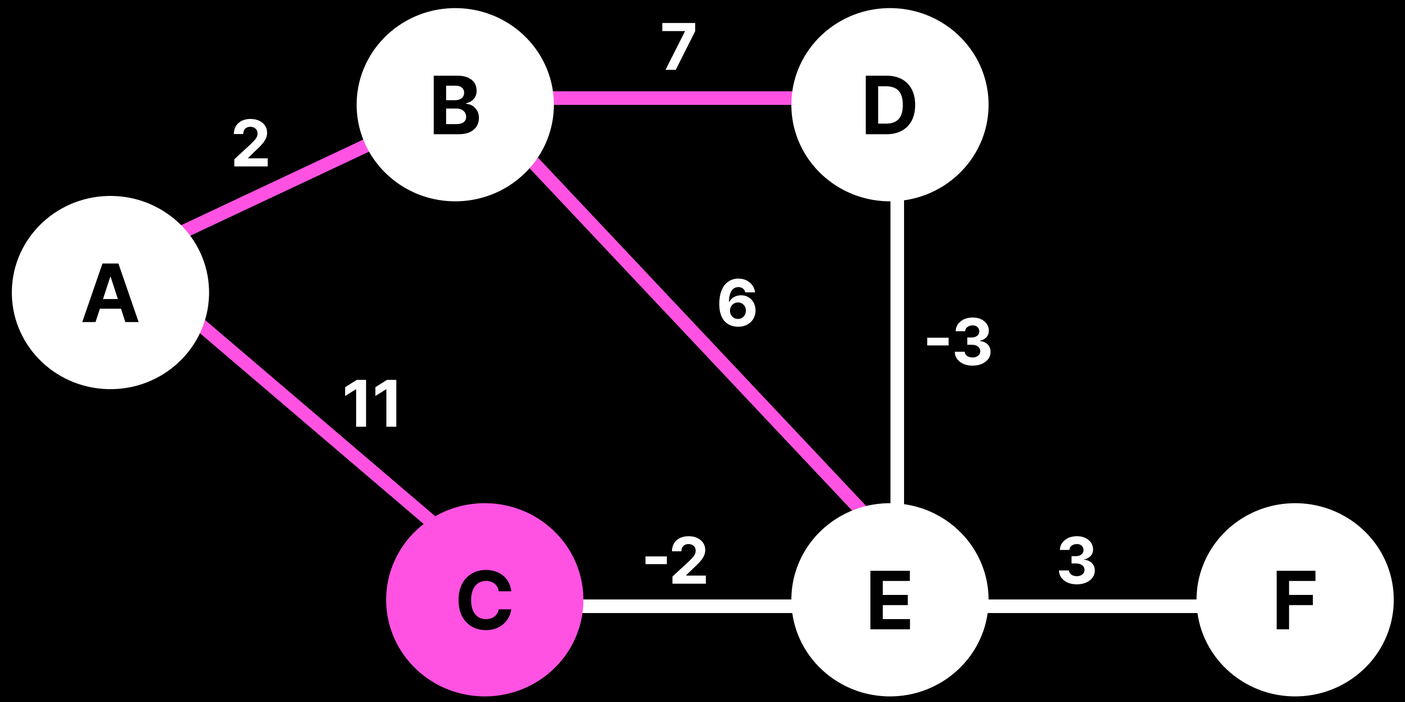


queue	distance
A	0
B	2
C	11
D	9
E	8
F	∞

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter



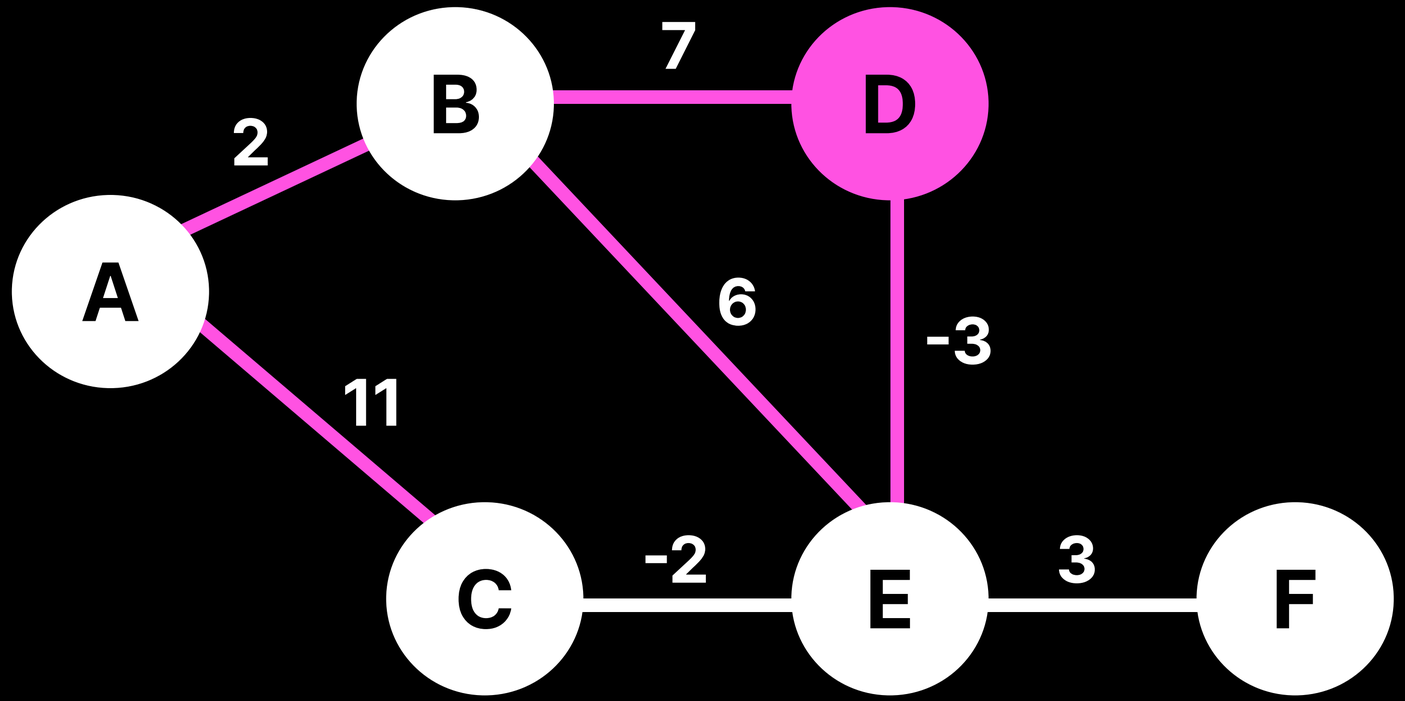
queue	distance
A	0
B	2
C	11
D	9
E	8<9
F	∞

Oppdateres ikke

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter



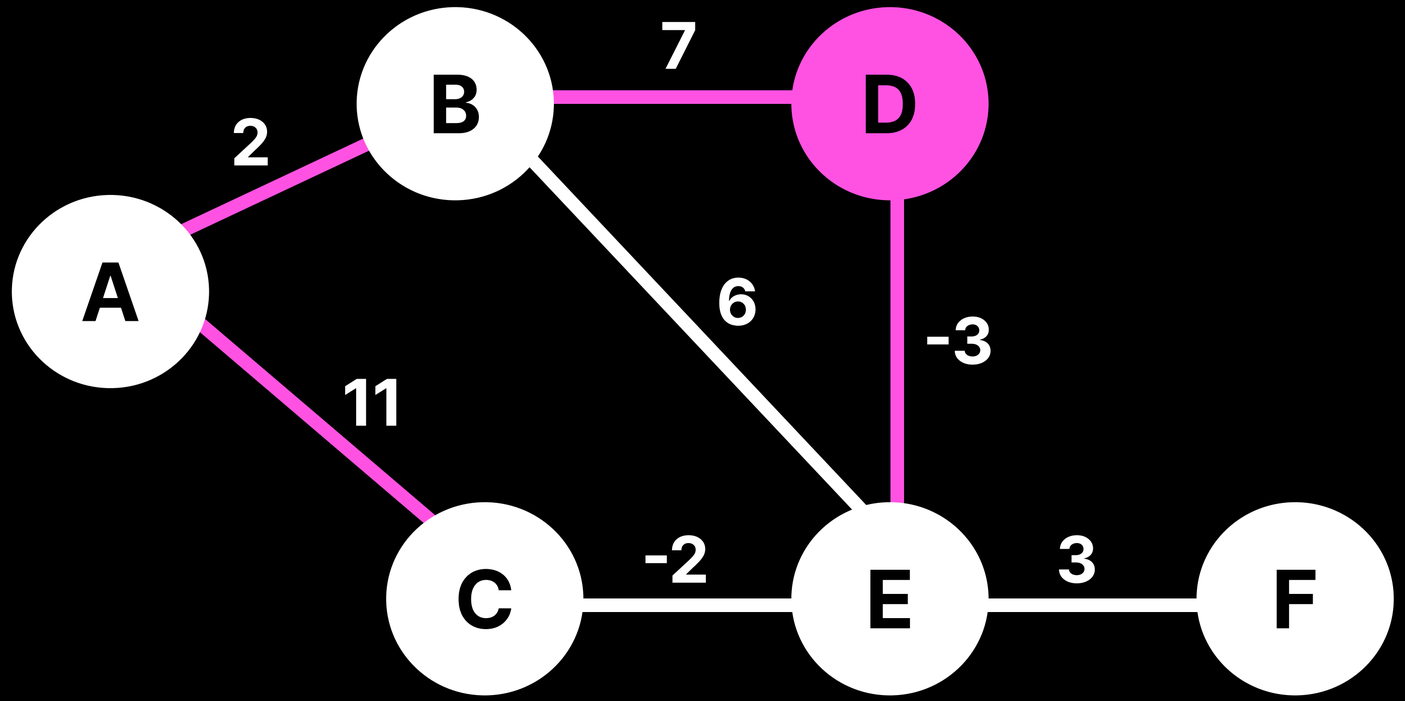
queue	distance
A	0
B	2
C	11
D	9
E	8>6
F	∞

Oppdateres!

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter

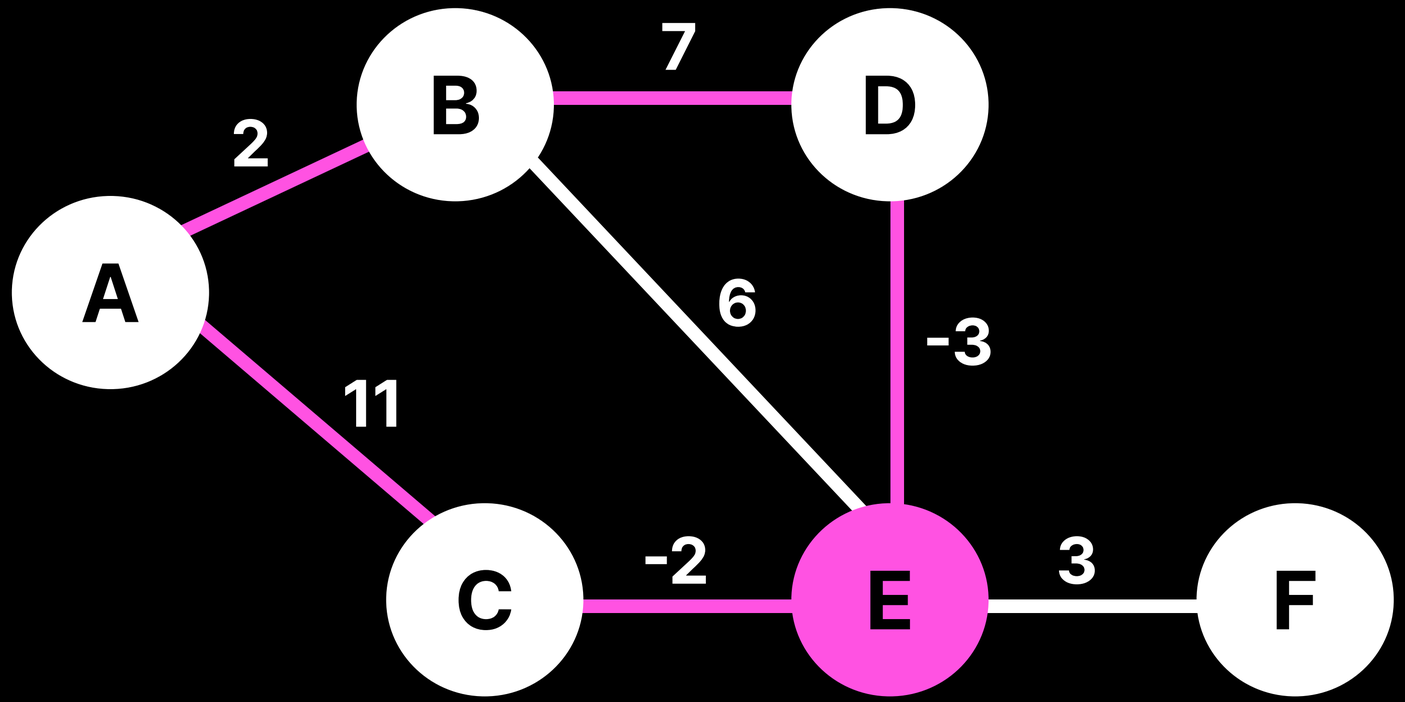


queue	distance
A	0
B	2
C	11
D	9
E	6
F	∞

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter



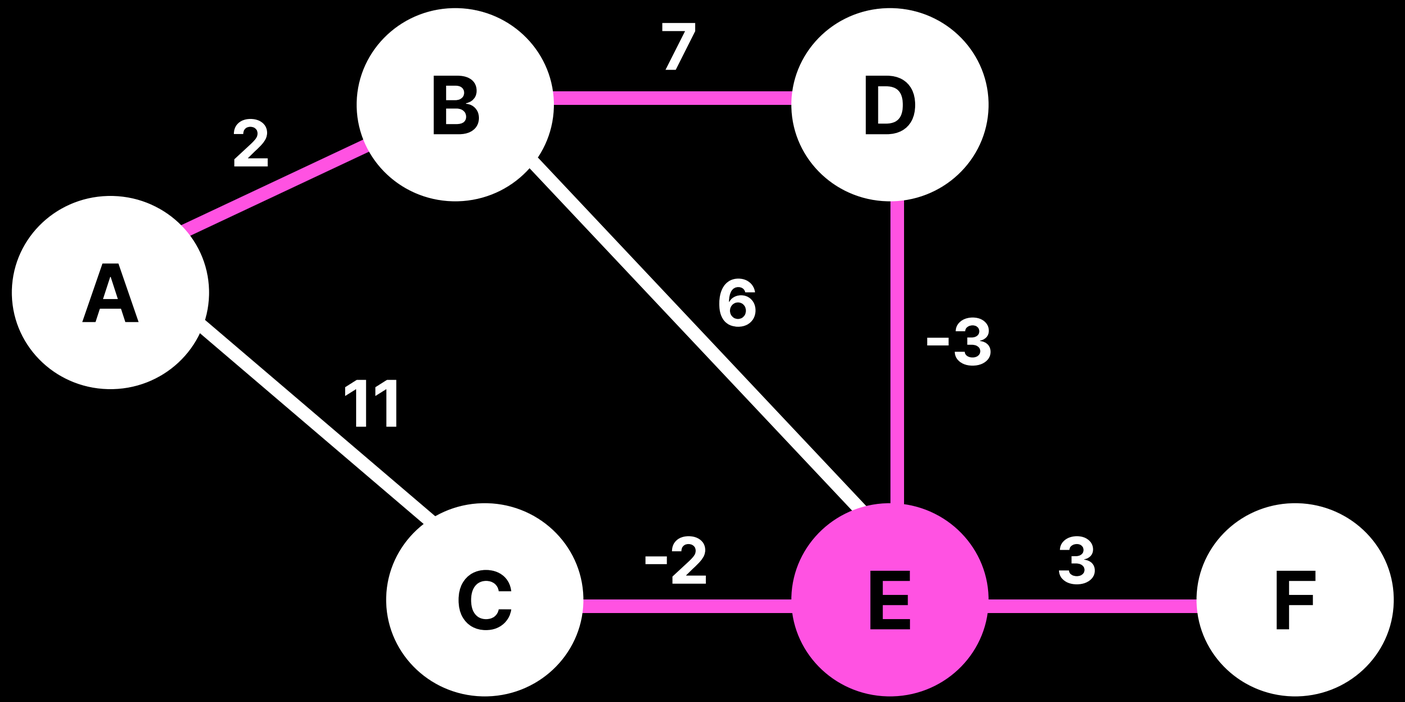
queue	distance
A	0
B	2
C	11>4
D	9
E	6
F	∞

Oppdateres!

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter

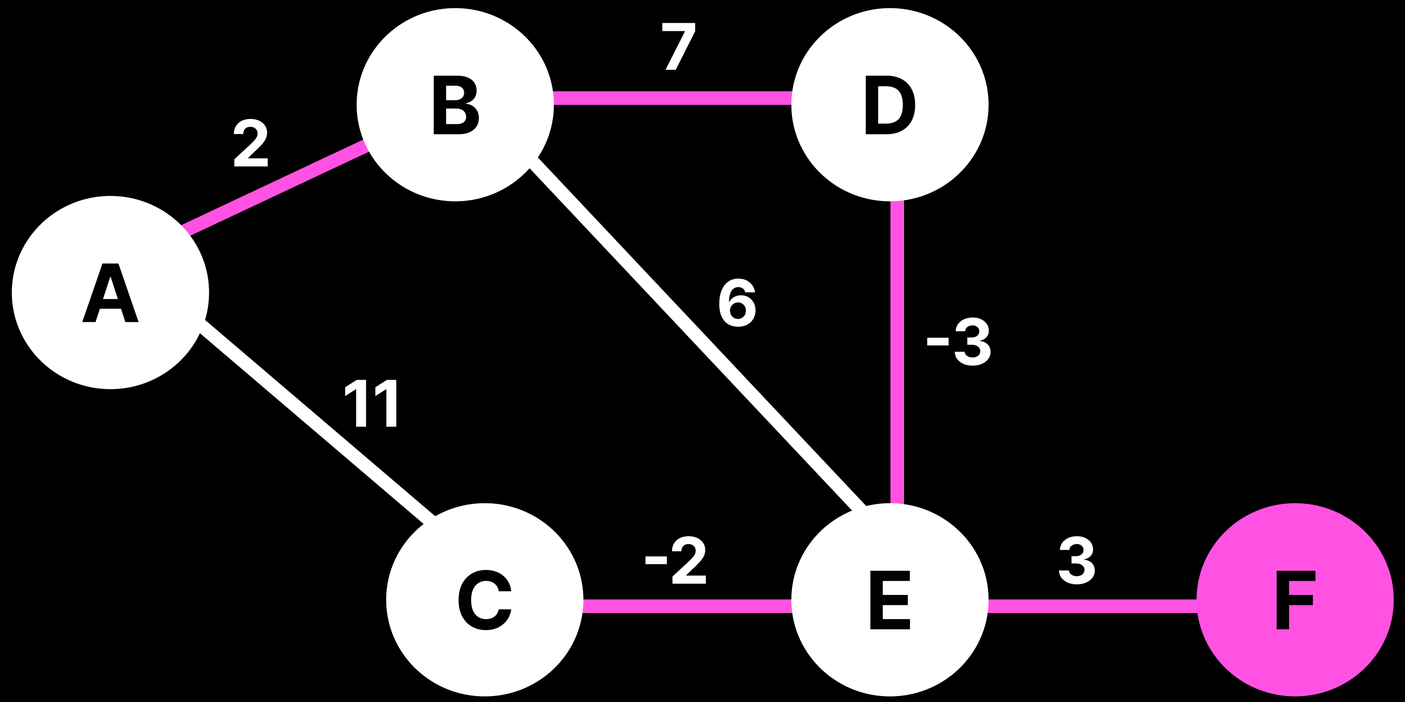


queue	distance
A	0
B	2
C	4
D	9
E	6
F	9

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter

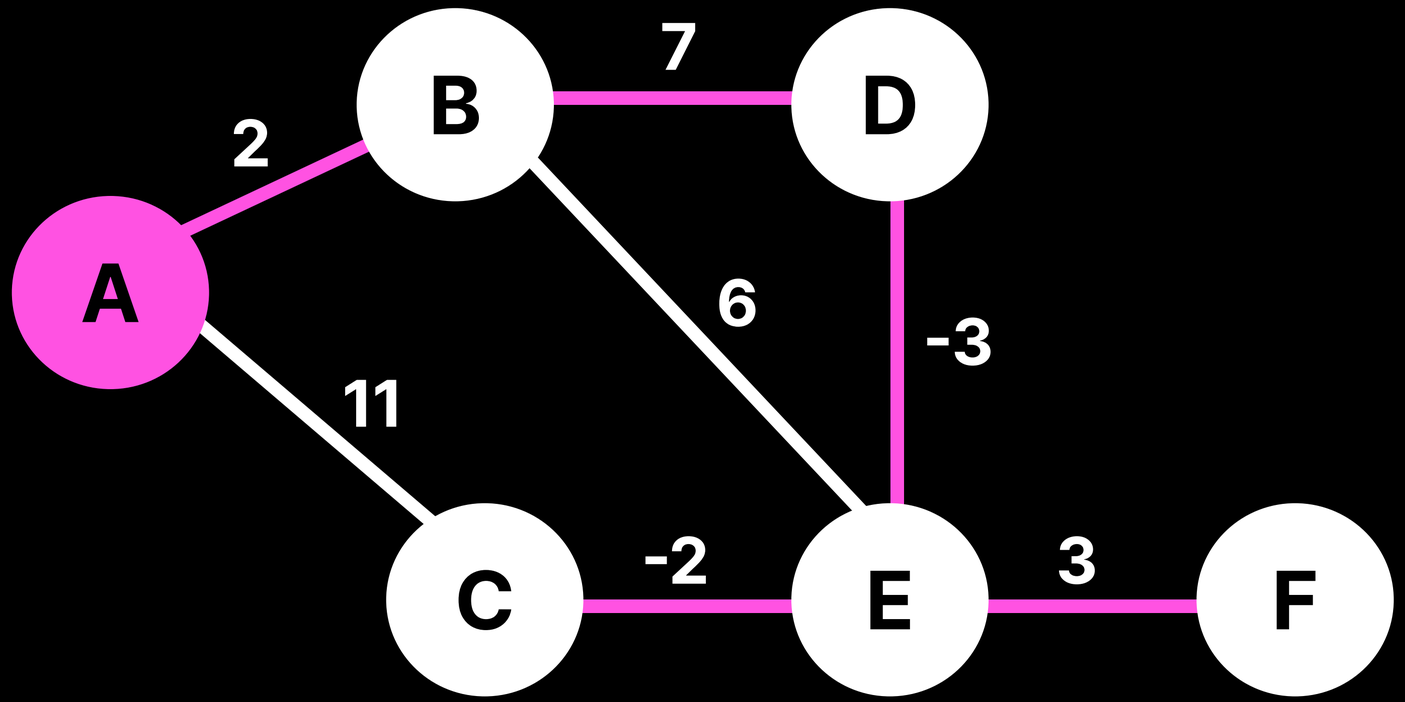


queue	distance
A	0
B	2
C	4
D	9
E	6
F	9

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter

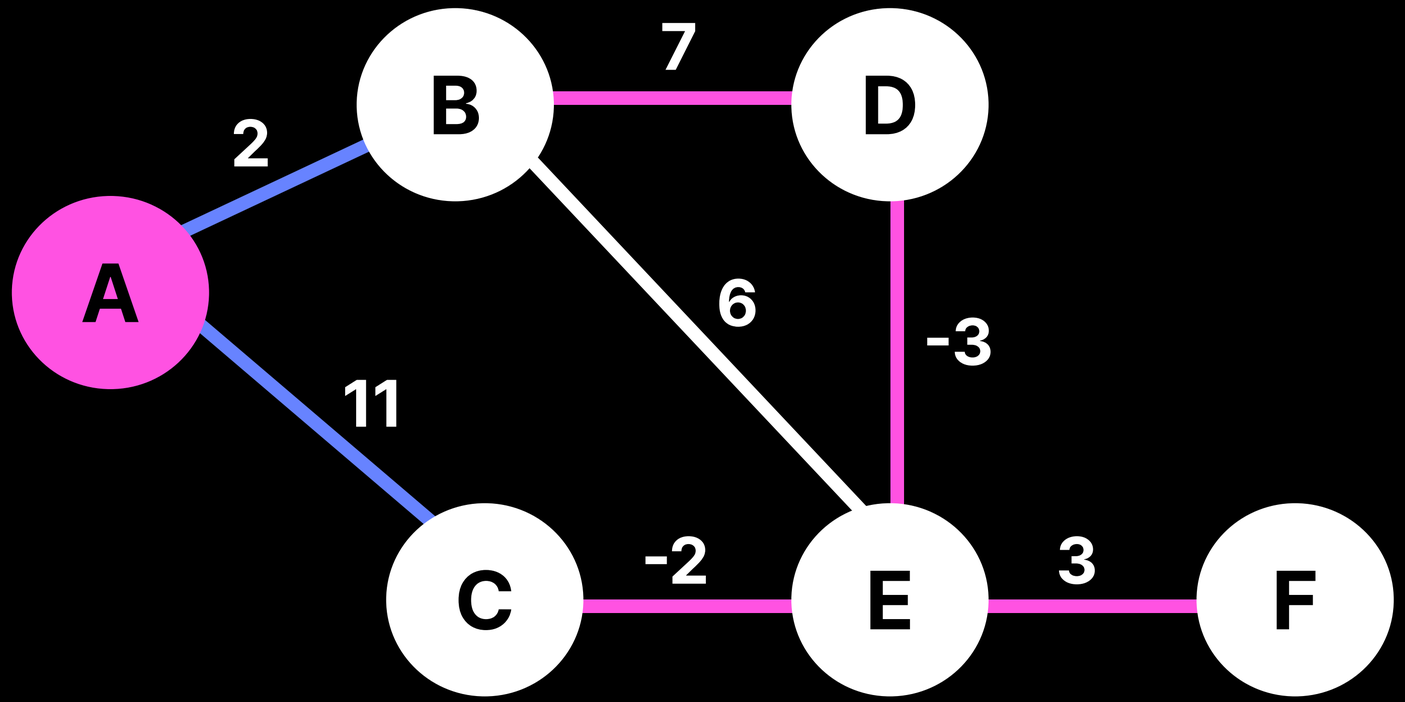


queue	distance
A	0
B	2
C	4
D	9
E	6
F	9

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter

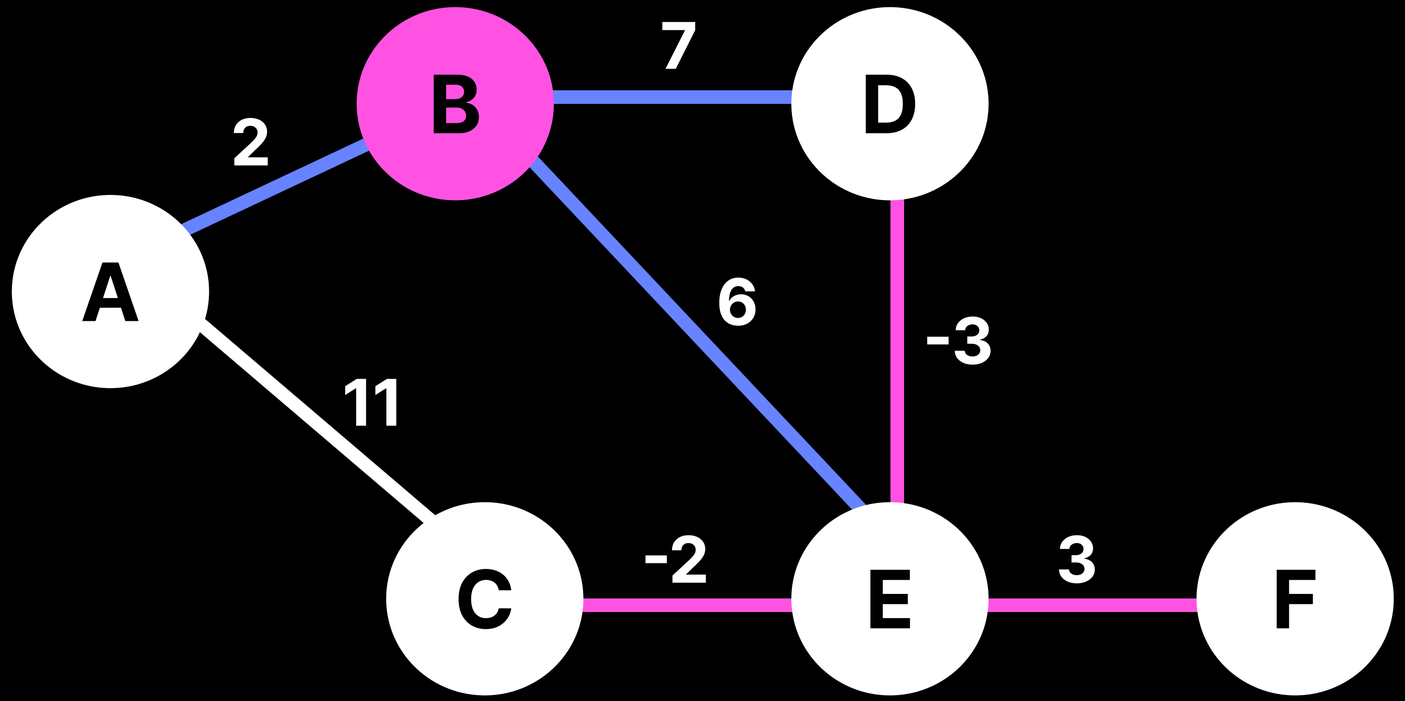


queue	distance
A	0
	Oppdaterer ingen
B	2=2
C	4<11
D	9
E	6
F	9

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter

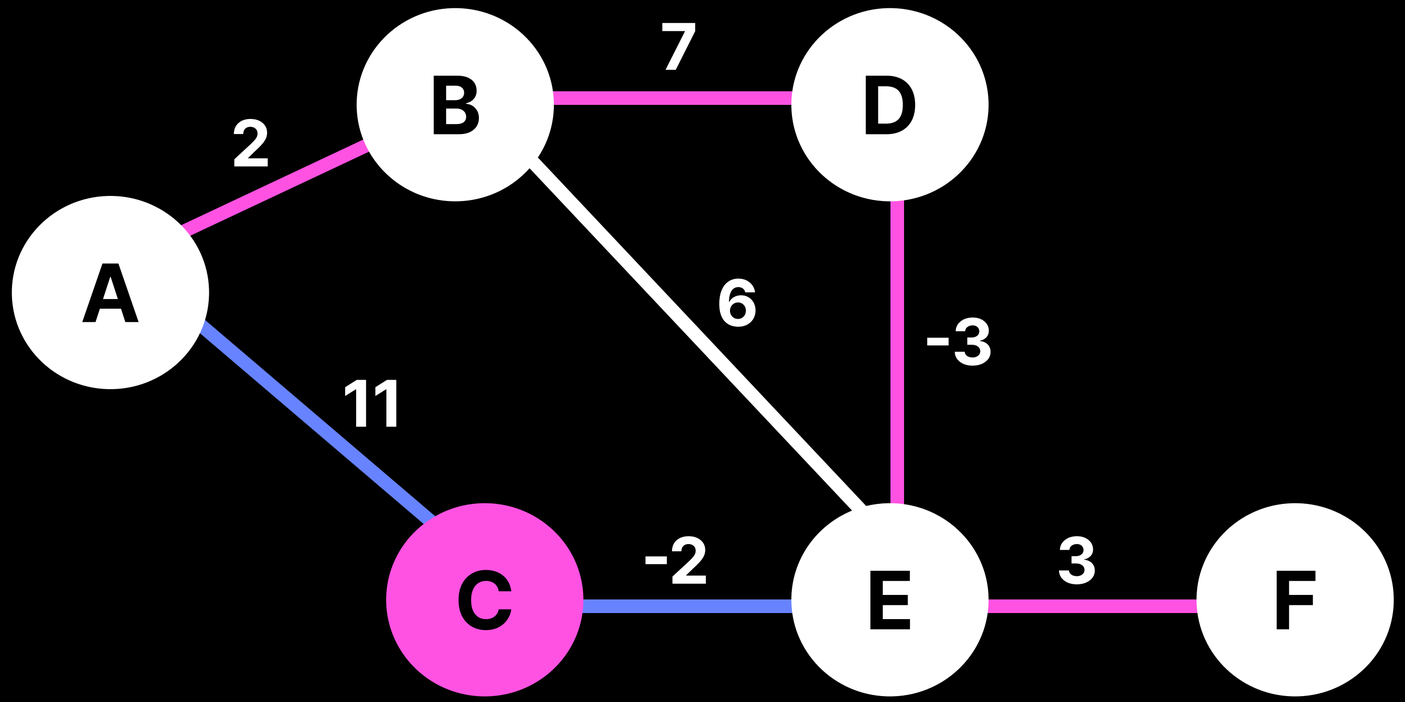


queue	distance
	Oppdaterer ingen
A	0
B	2
C	4
D	9=9
E	6<8
F	9

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter

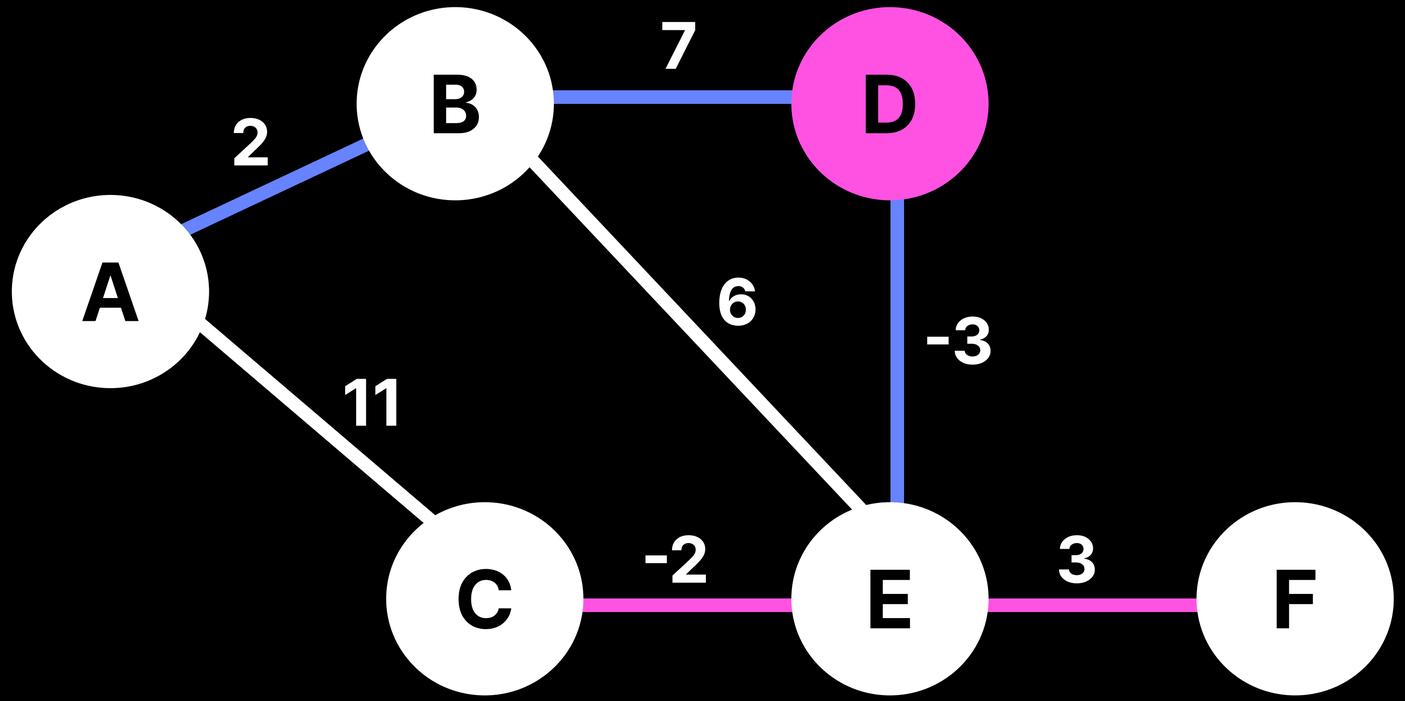


queue	distance
	Oppdaterer ingen
A	0
B	2
C	4
D	9
E	6<9
F	9

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter

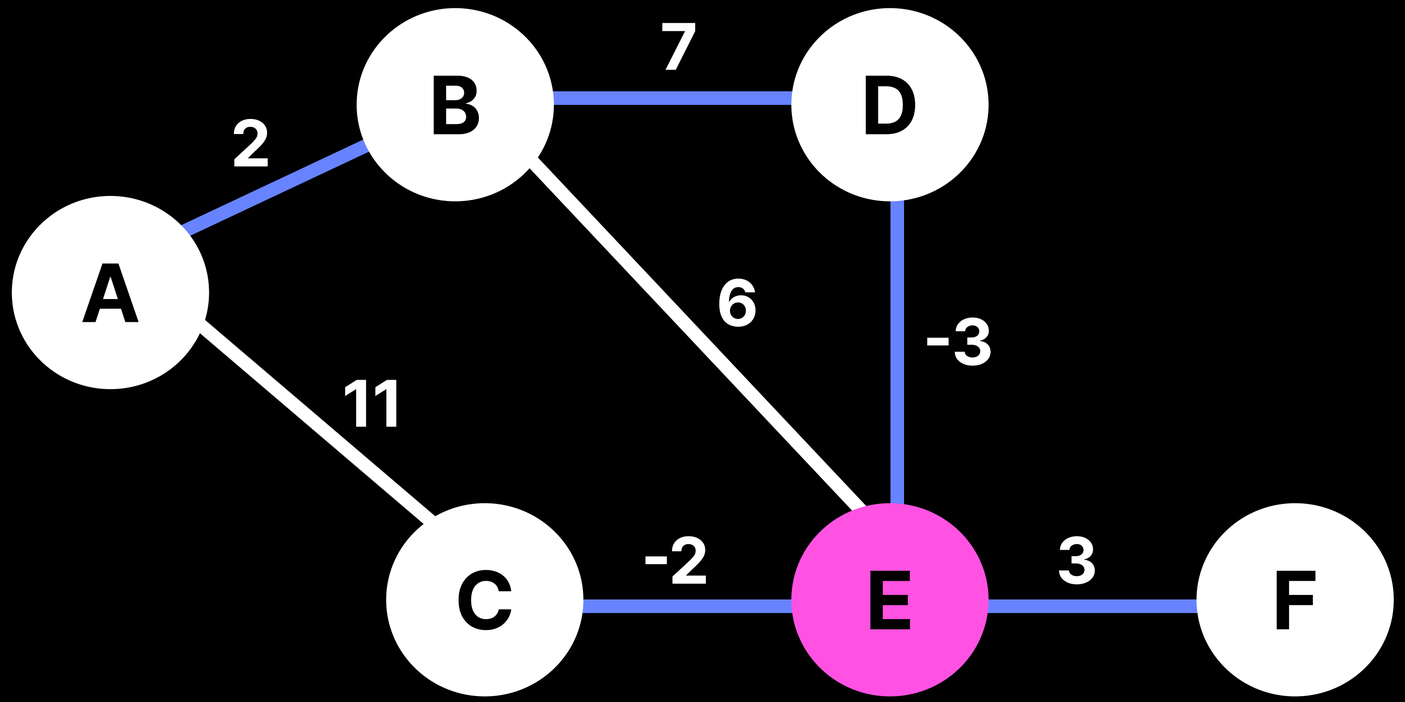


queue	distance
	Oppdaterer ingen
A	0
B	2
C	4
D	9
E	6=6
F	9

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter

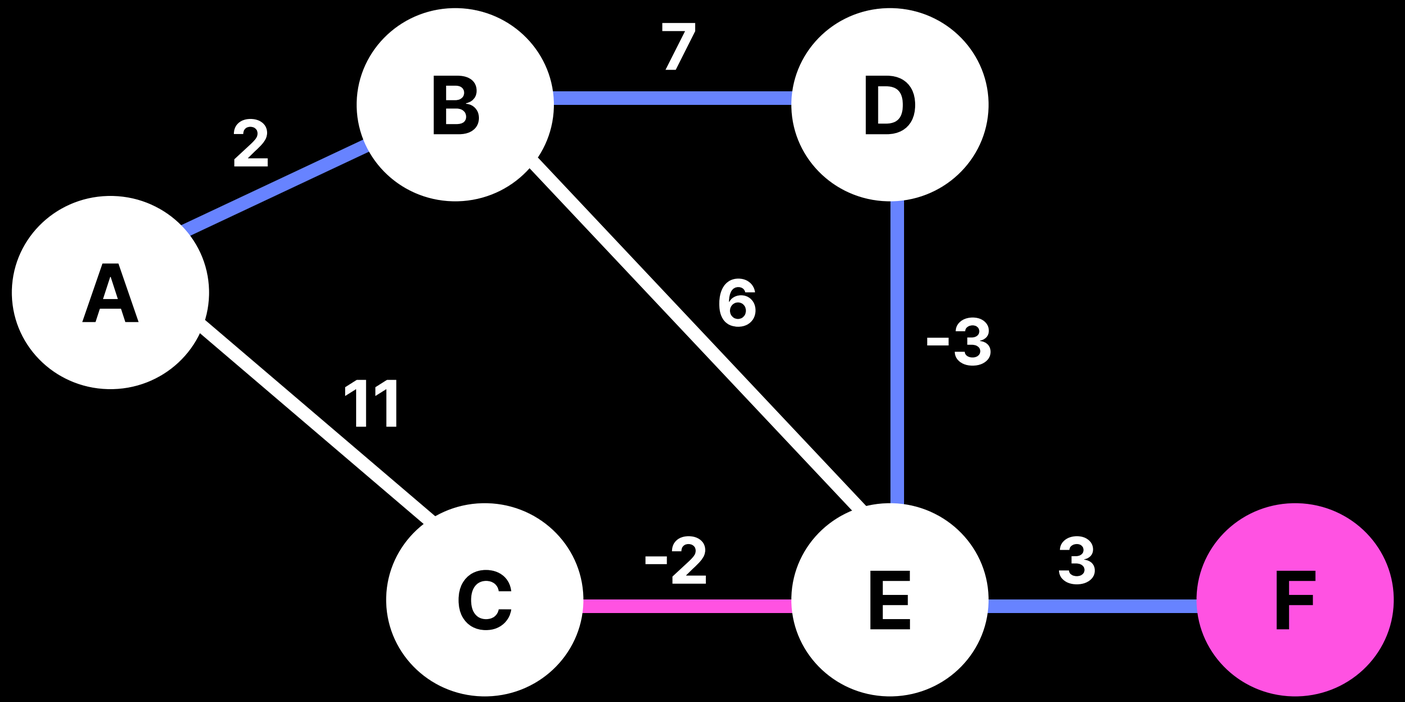


queue	distance
	Oppdaterer ingen
A	0
B	2
C	4=4
D	9
E	6
F	9=9

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter

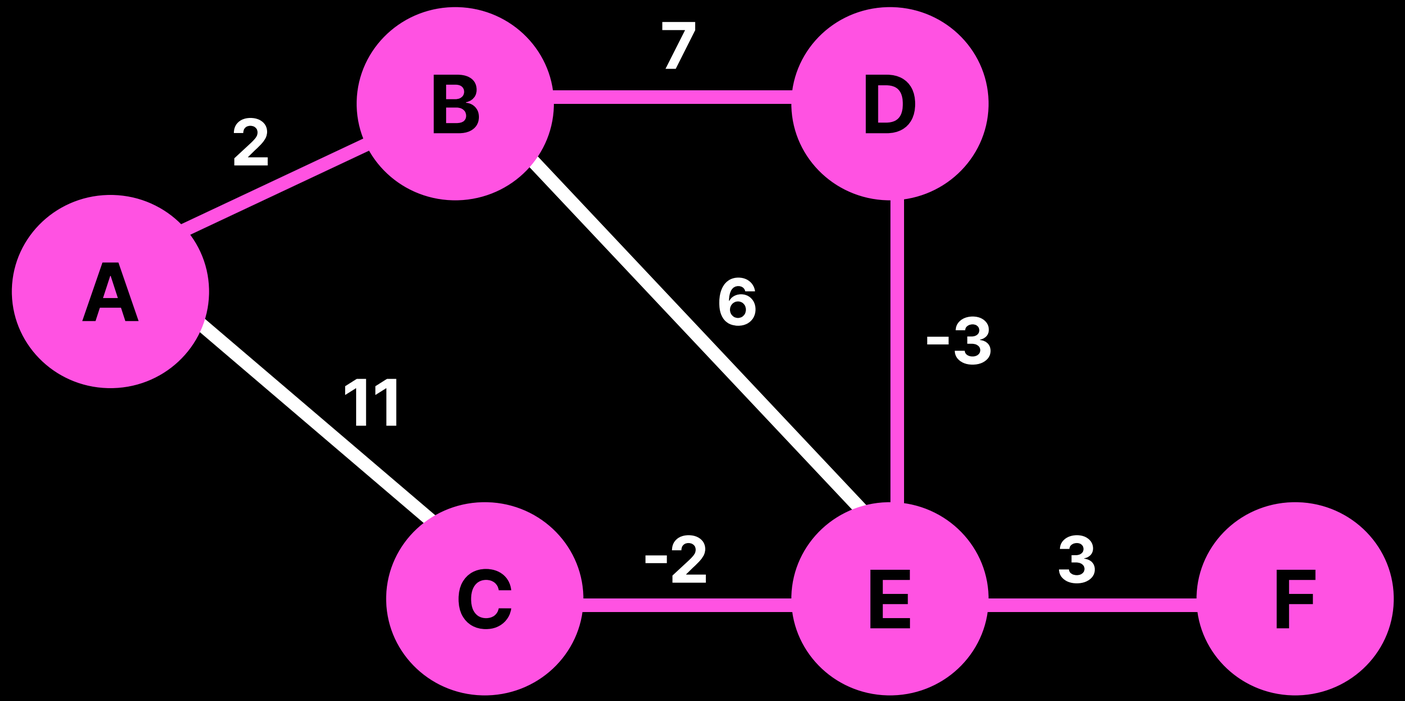


queue	distance
	Oppdaterer ingen
A	0
B	2
C	4
D	9
E	6
F	9

Obs: negative sykler vil gi en error

Bellman-Ford

Negative vekter

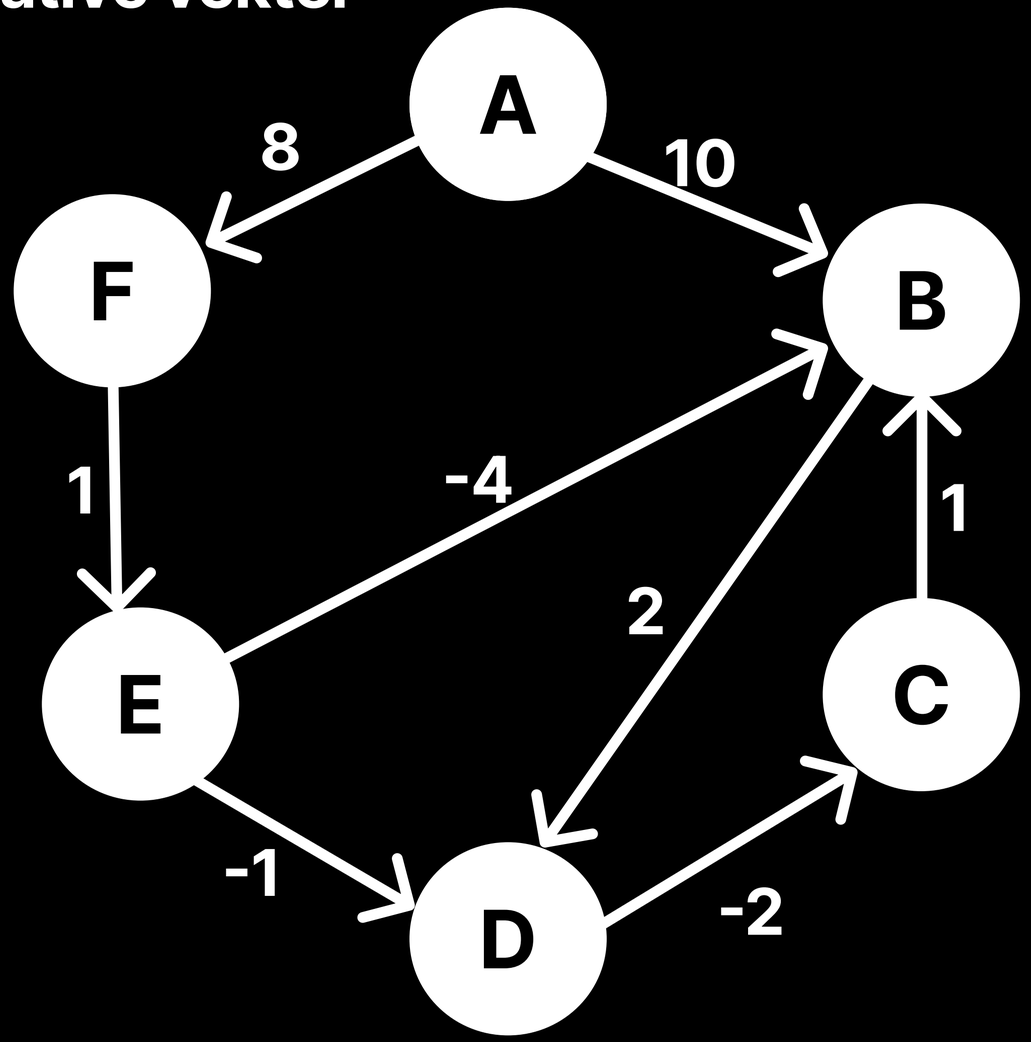


queue	distance
A	0
B	2
C	4
D	9
E	6
F	9

Obs: negative sykler vil gi en error

Bellman-Ford

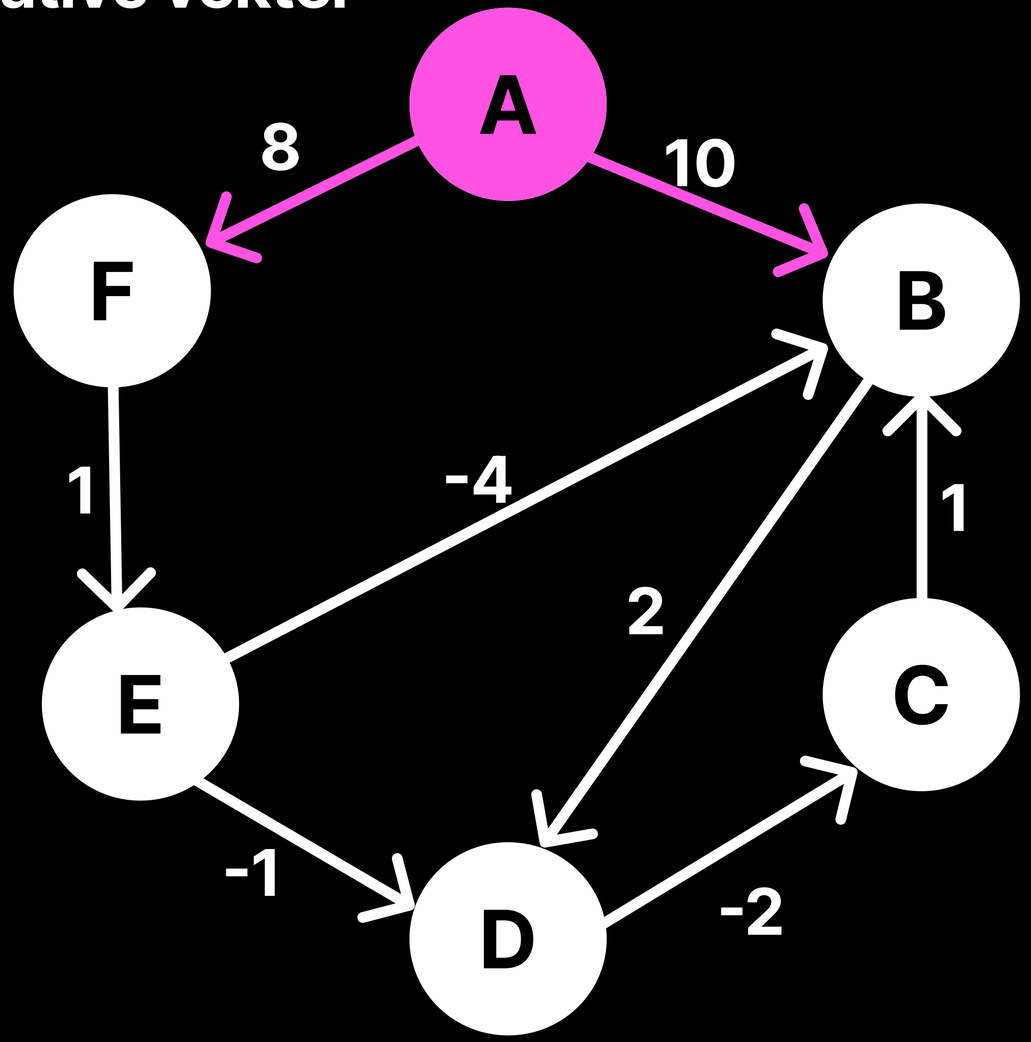
Negative vekter



queue	distance
A	0
B	∞
C	∞
D	∞
E	∞
F	∞

Bellman-Ford

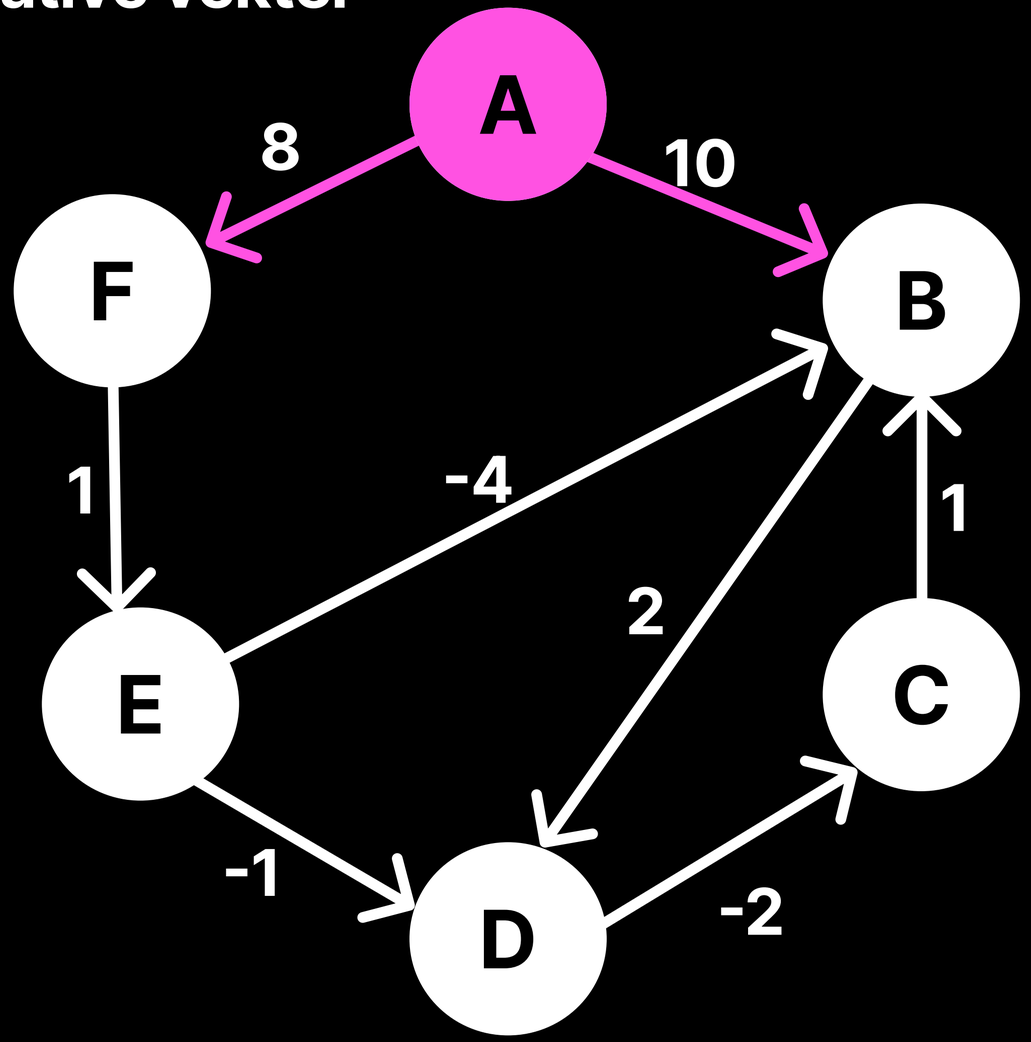
Negative vekter



queue	distance
A	0
B	∞
C	∞
D	∞
E	∞
F	∞

Bellman-Ford

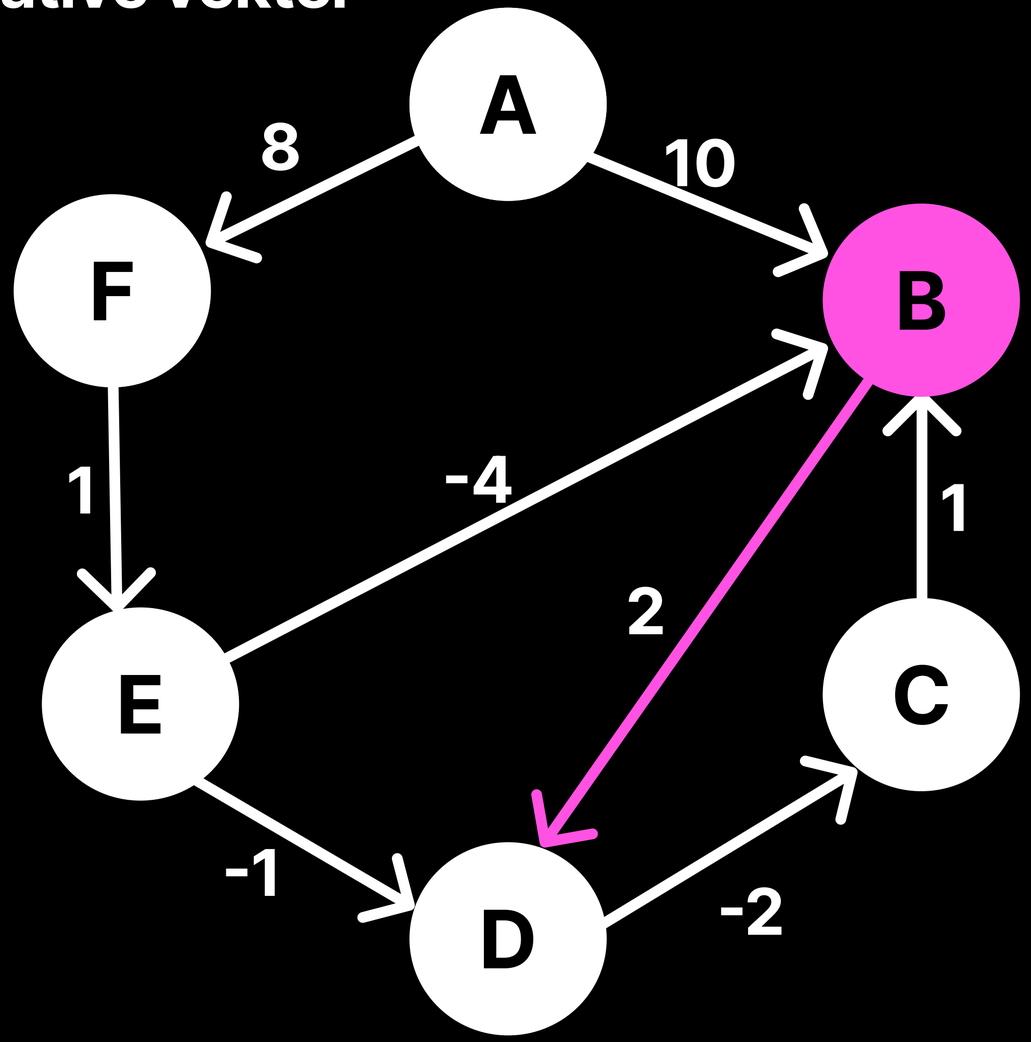
Negative vekter



queue	distance
A	0
B	10
C	∞
D	∞
E	∞
F	8

Bellman-Ford

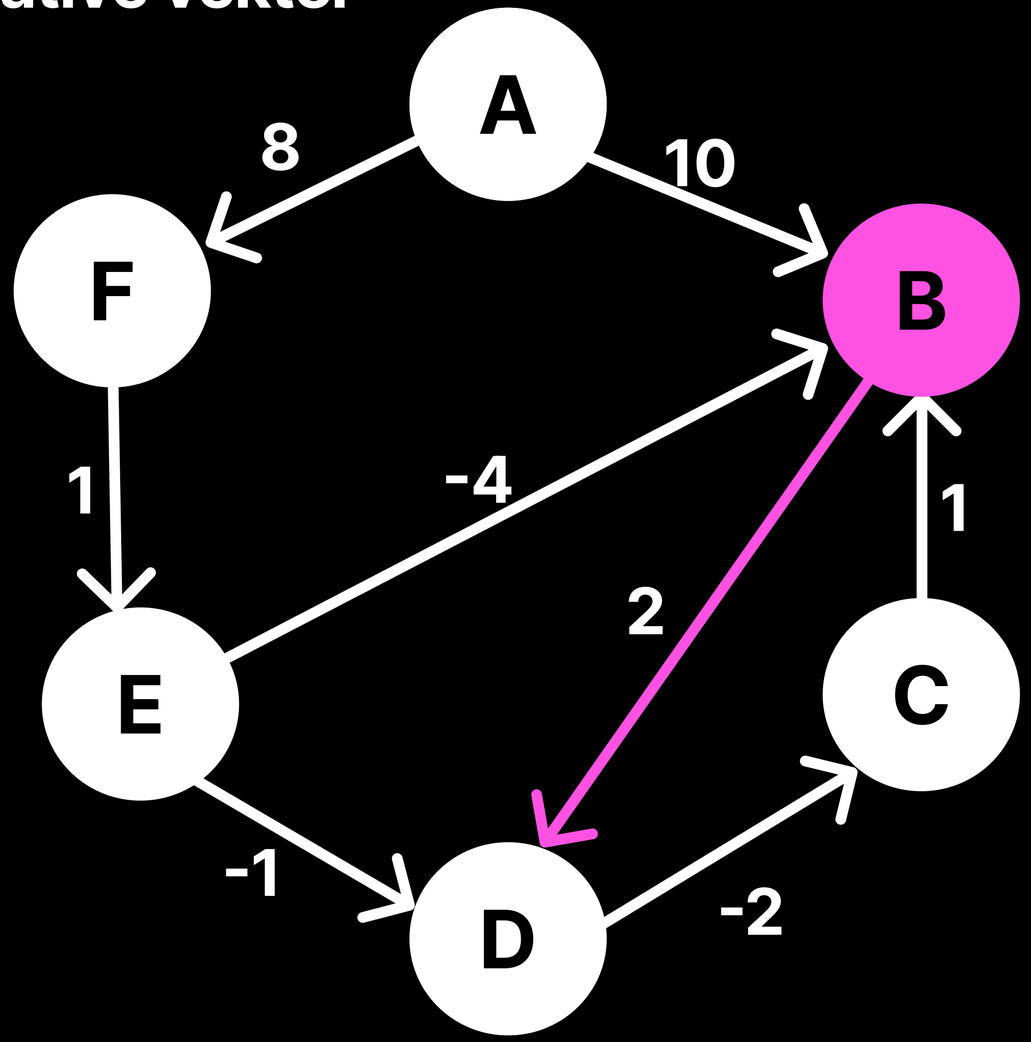
Negative vekter



queue	distance
A	0
B	10
C	∞
D	∞
E	∞
F	8

Bellman-Ford

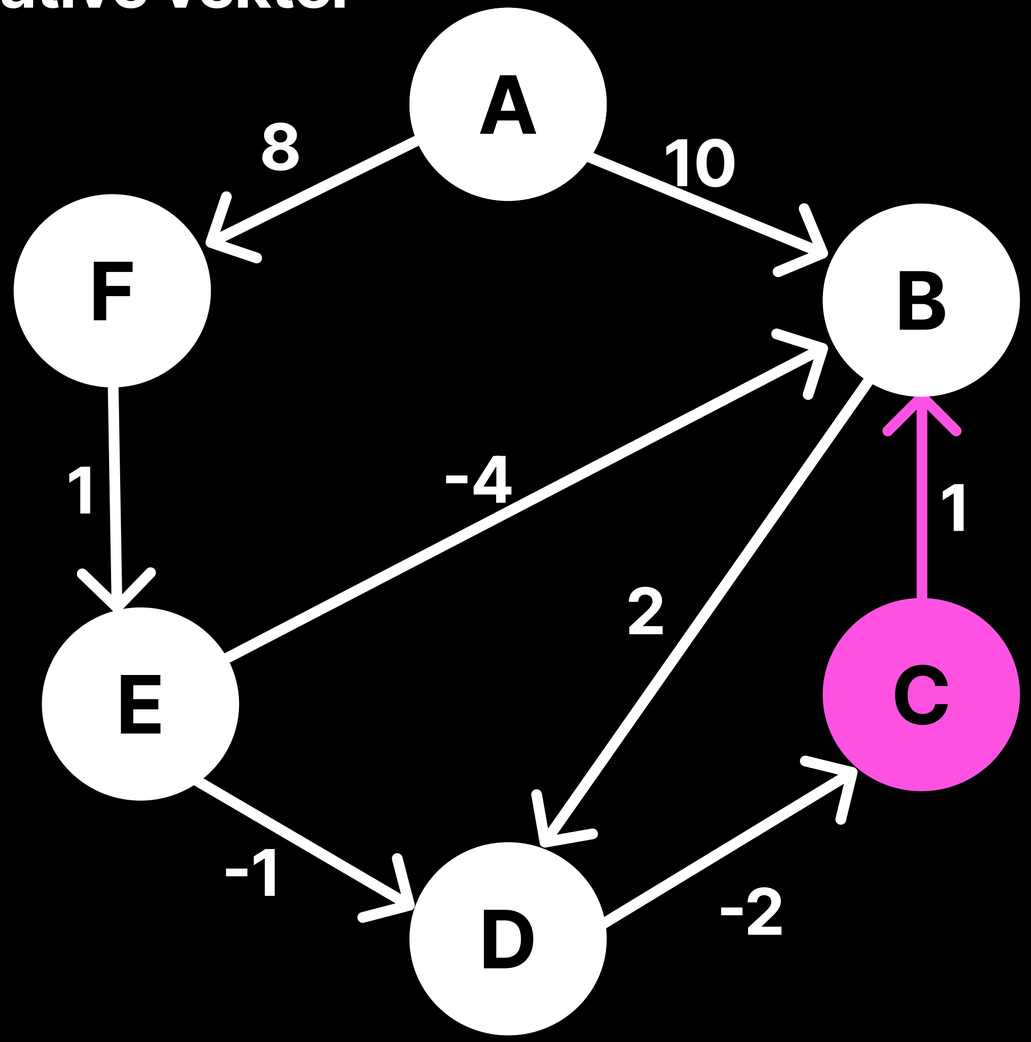
Negative vekter



queue	distance
A	0
B	10
C	∞
D	12
E	∞
F	8

Bellman-Ford

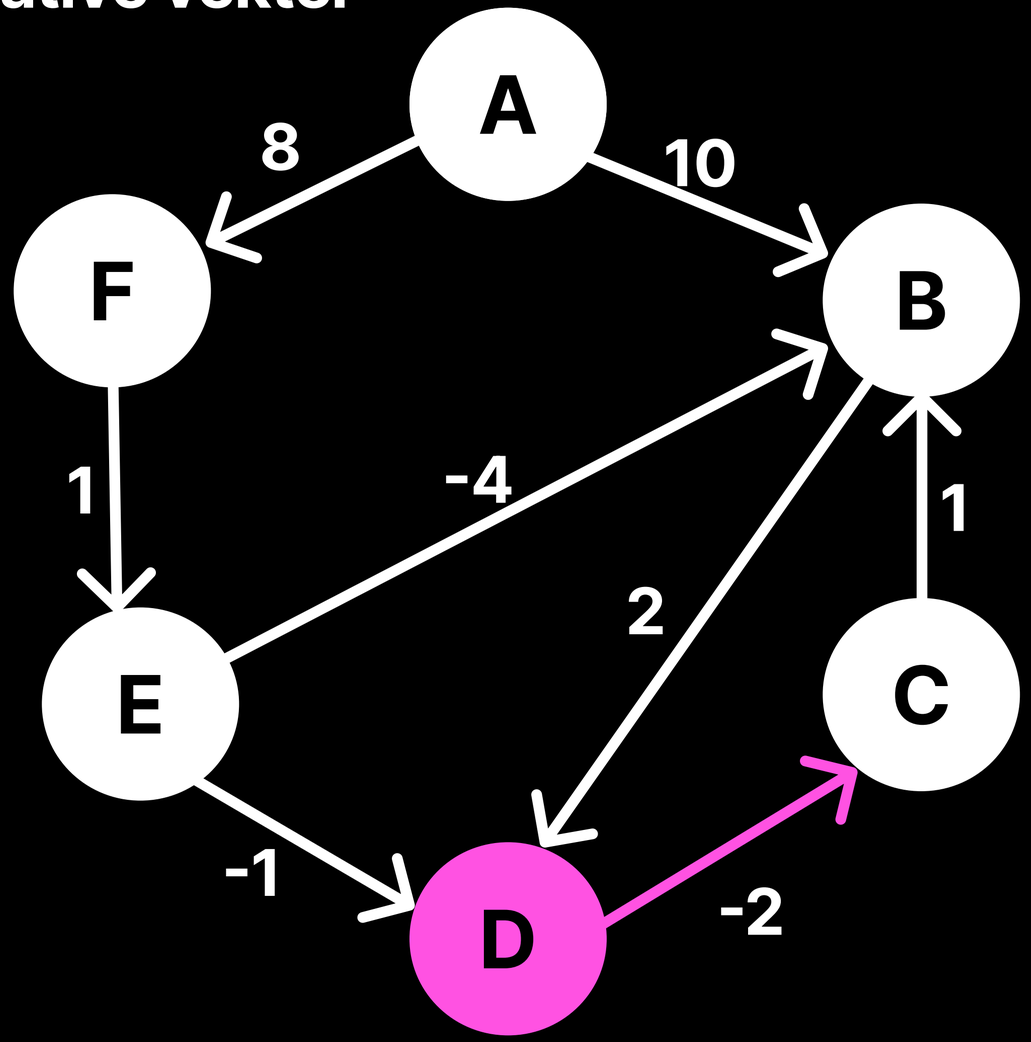
Negative vekter



queue	distance
A	0
B	10
C	∞
D	12
E	∞
F	8

Bellman-Ford

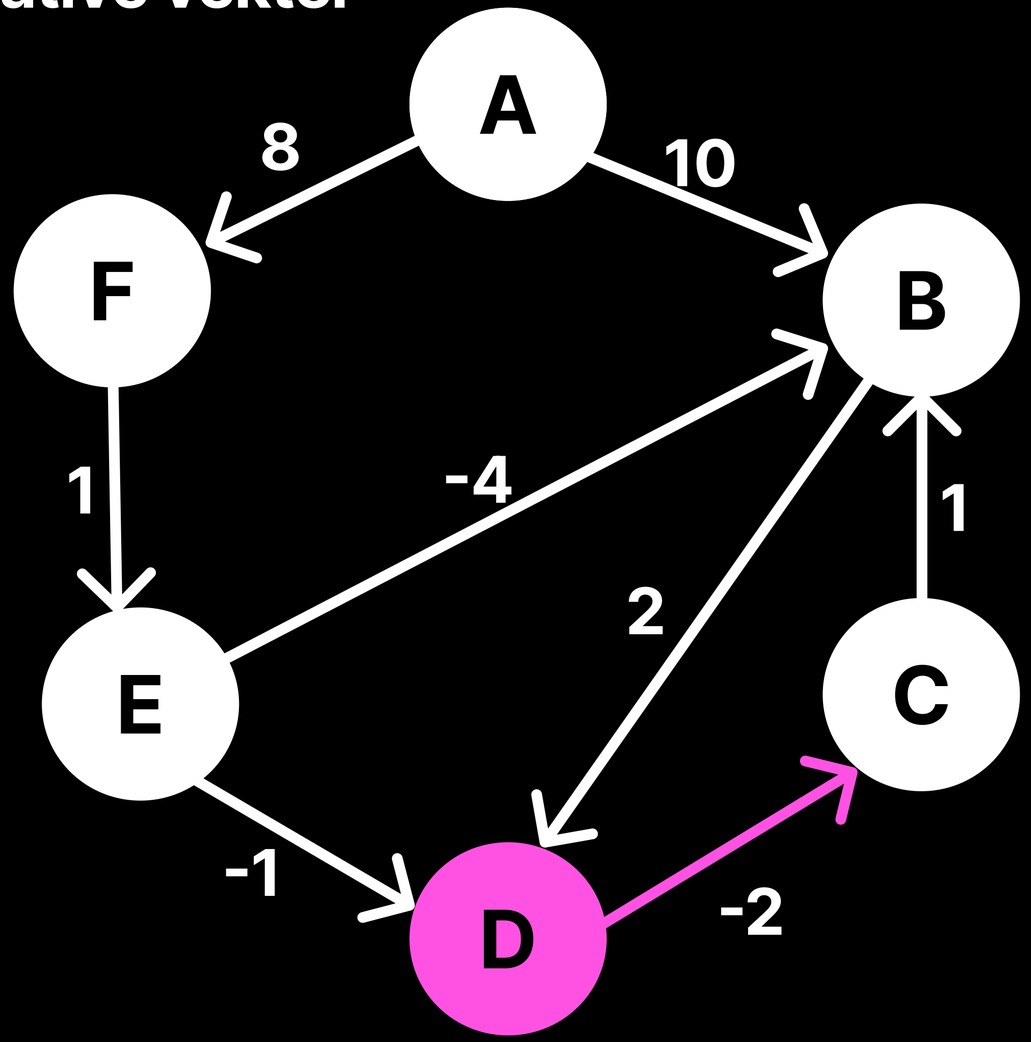
Negative vekter



queue	distance
A	0
B	10
C	∞
D	12
E	∞
F	8

Bellman-Ford

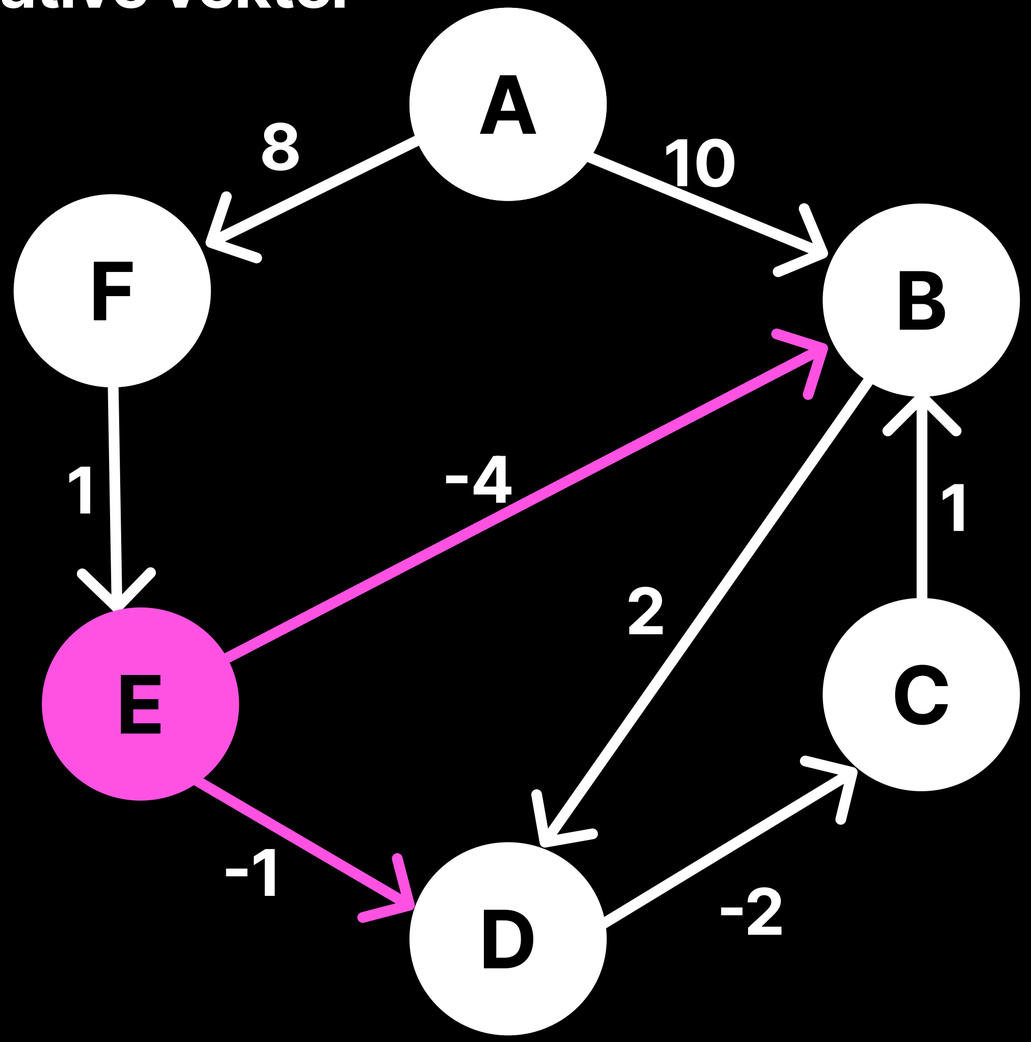
Negative vekter



queue	distance
A	0
B	10
C	10
D	12
E	∞
F	8

Bellman-Ford

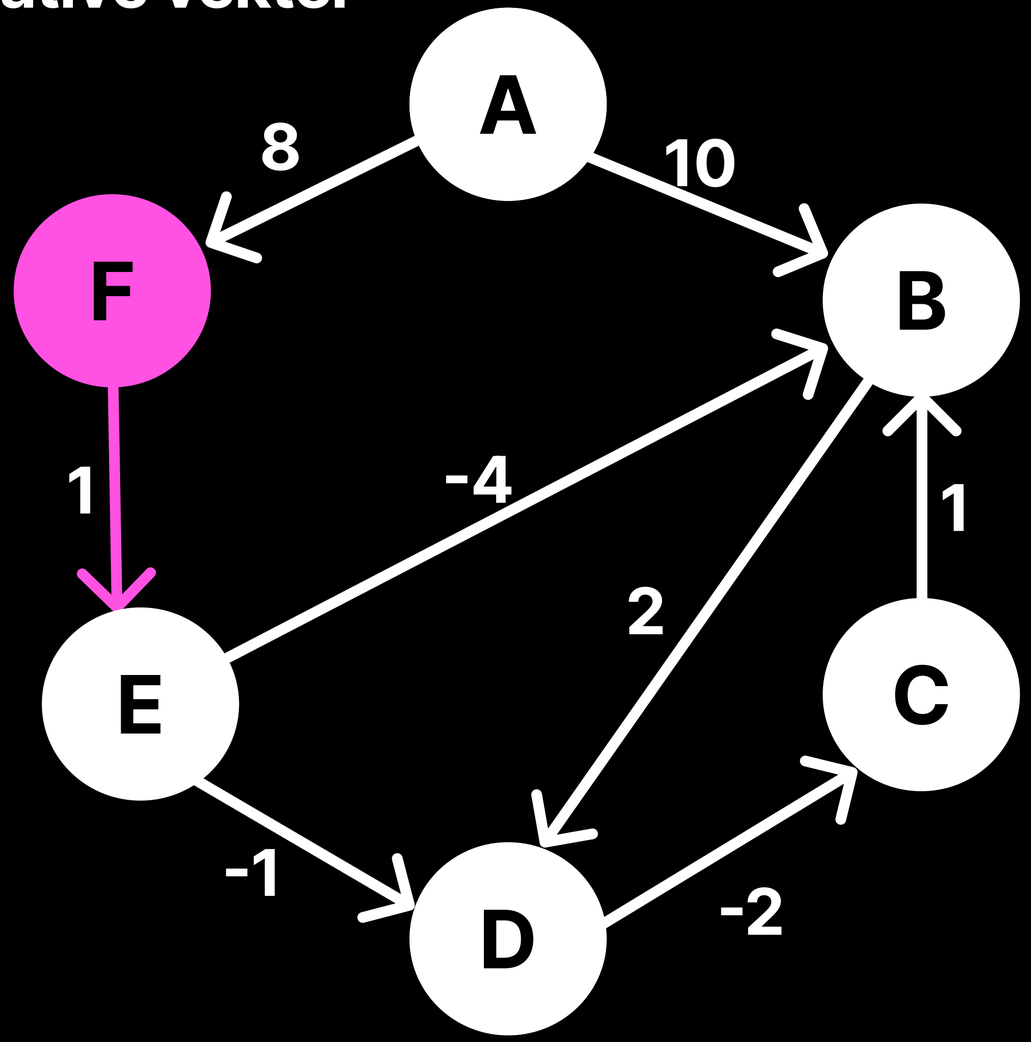
Negative vekter



queue	distance
A	0
B	10
C	10
D	12
E	∞
F	8

Bellman-Ford

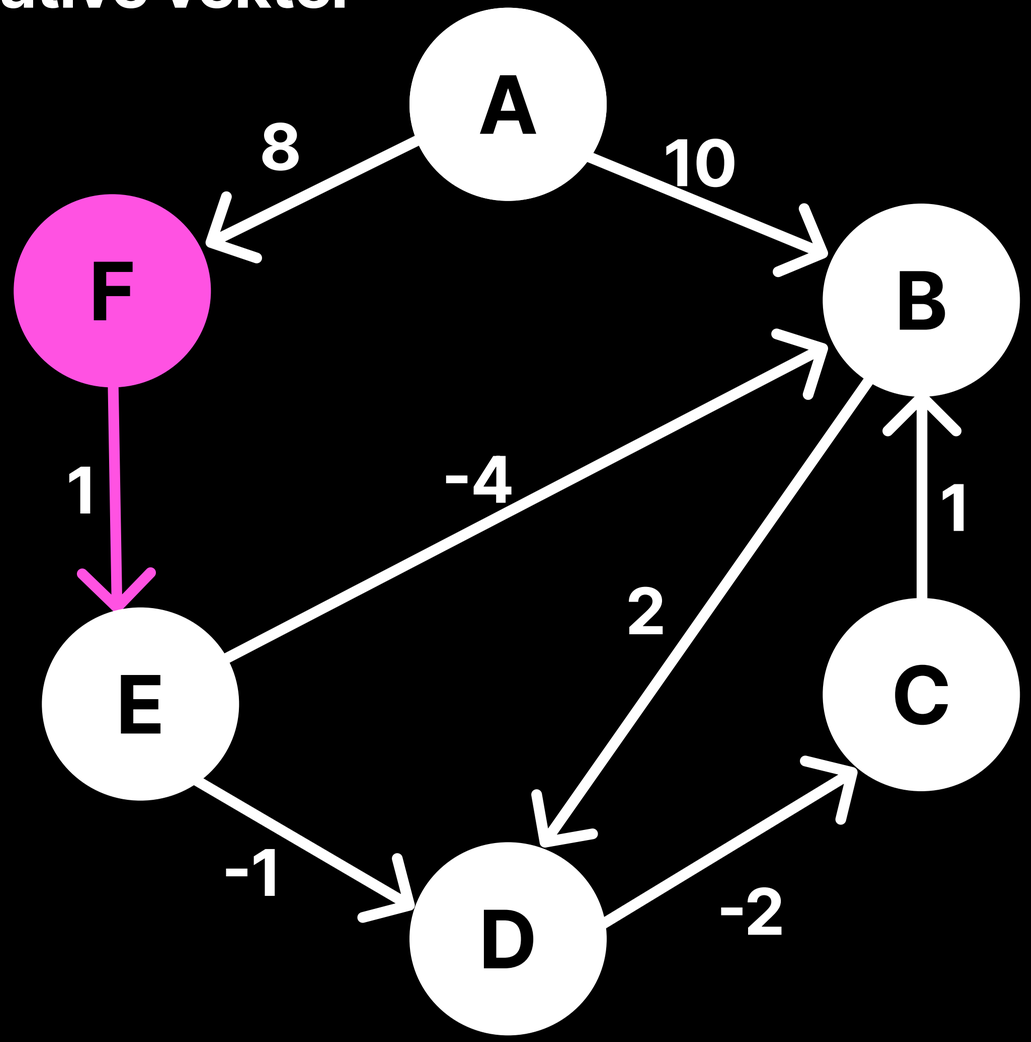
Negative vekter



queue	distance
A	0
B	10
C	10
D	12
E	∞
F	8

Bellman-Ford

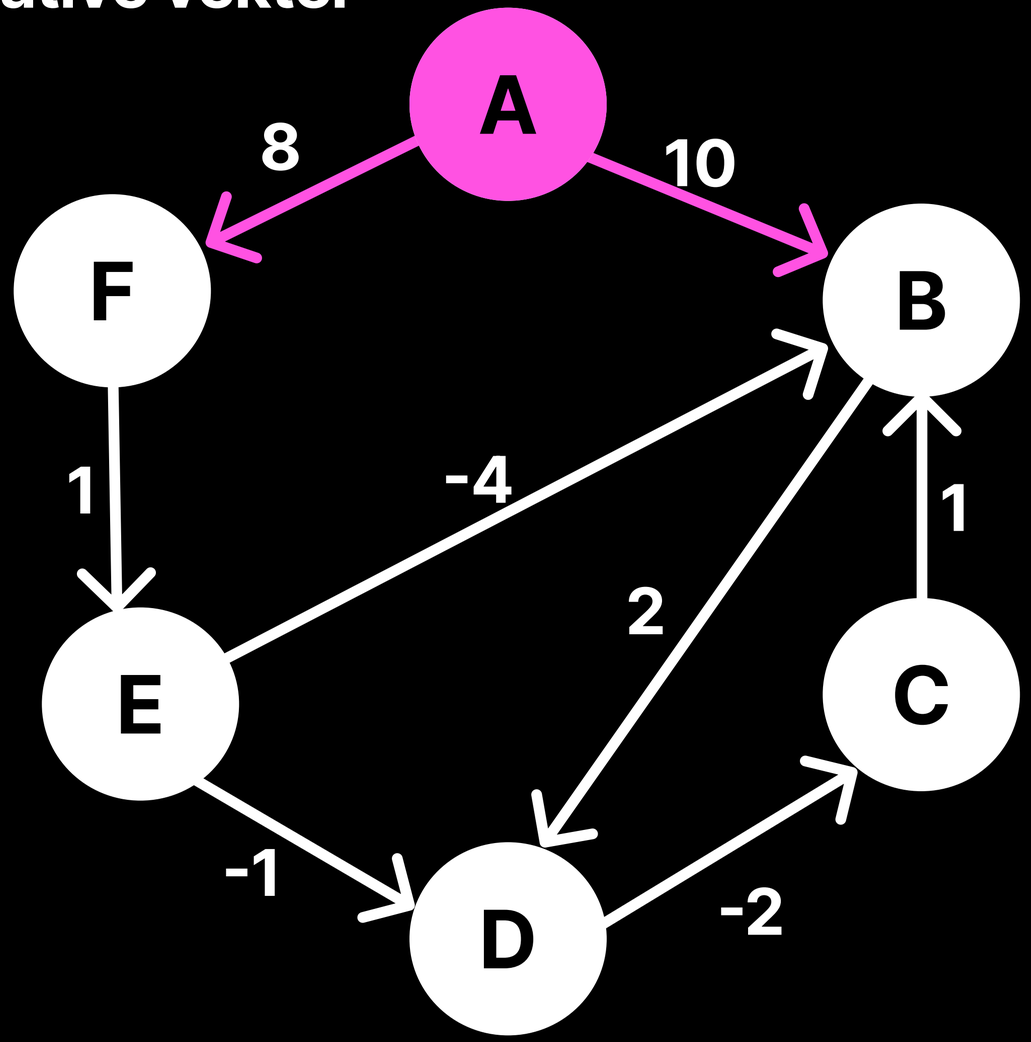
Negative vekter



queue	distance
A	0
B	10
C	10
D	12
E	9
F	8

Bellman-Ford

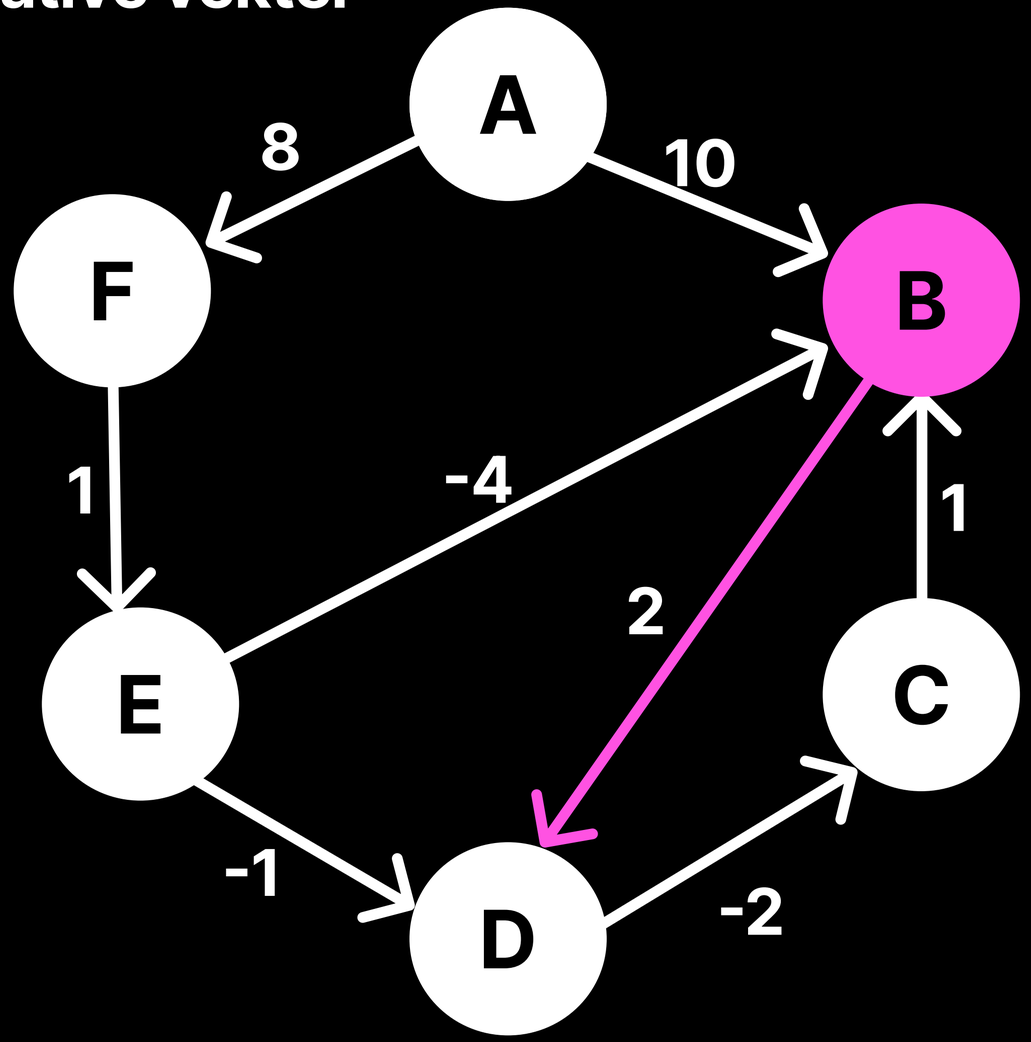
Negative vekter



queue	distance
A	0
B	10=10
C	10
D	12
E	9
F	8=10

Bellman-Ford

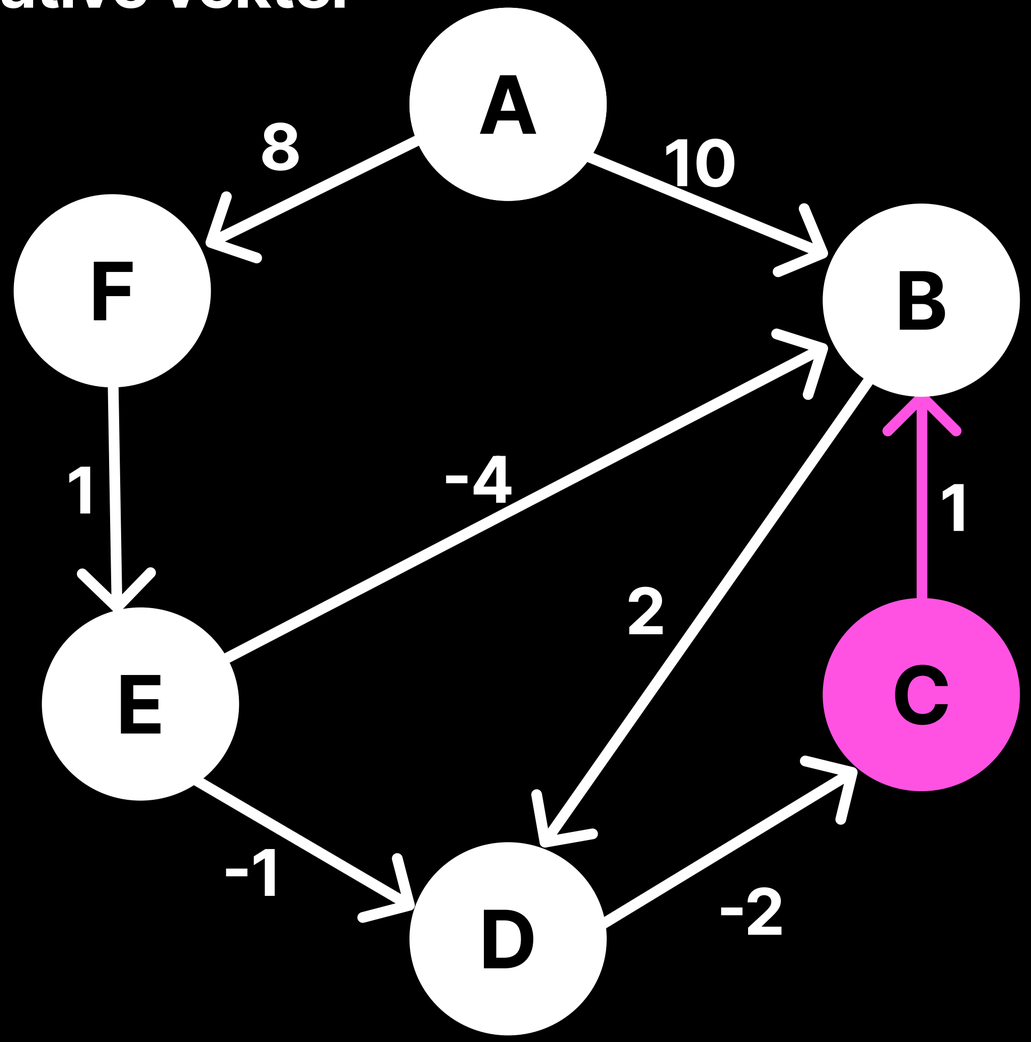
Negative vekter



queue	distance
A	0
B	10
C	10
D	12=12
E	9
F	8

Bellman-Ford

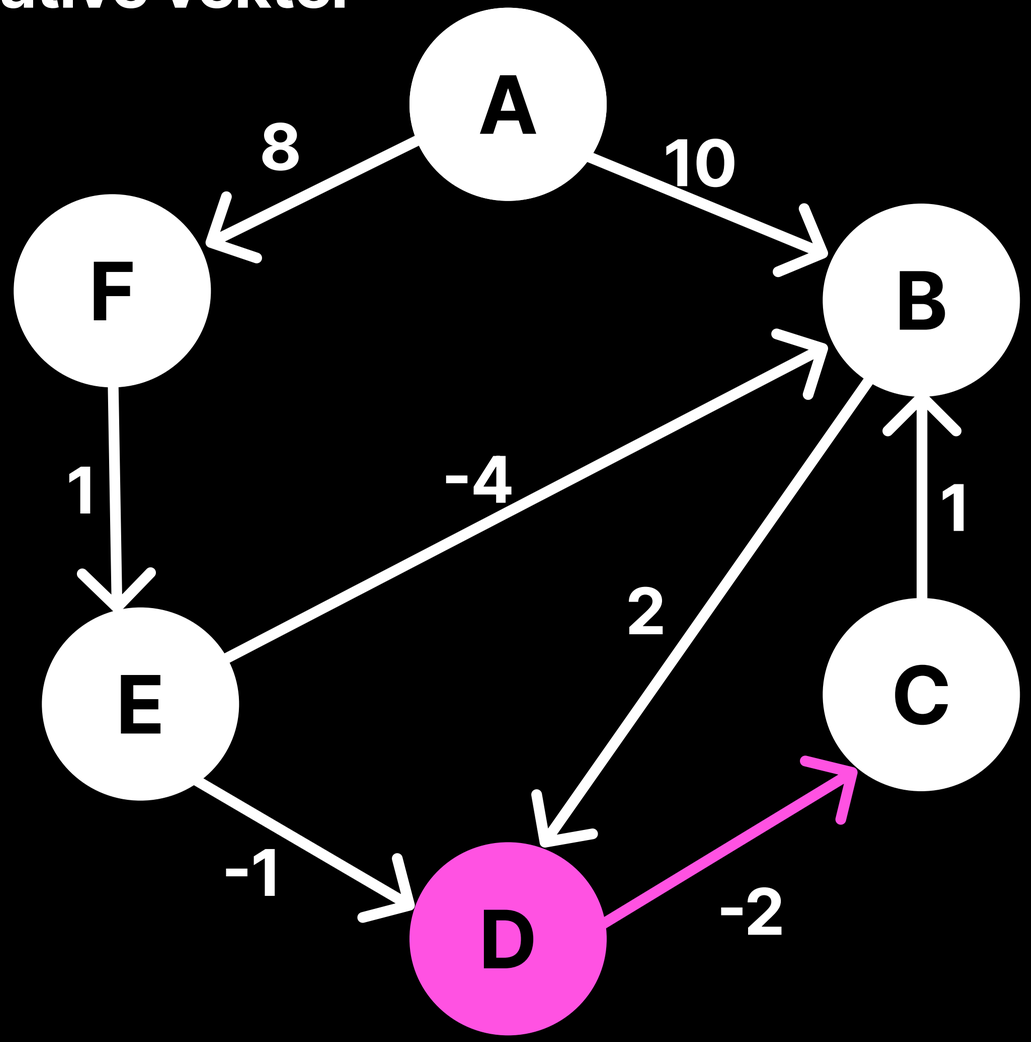
Negative vekter



queue	distance
A	0
B	10<11
C	13
D	12
E	9
F	8

Bellman-Ford

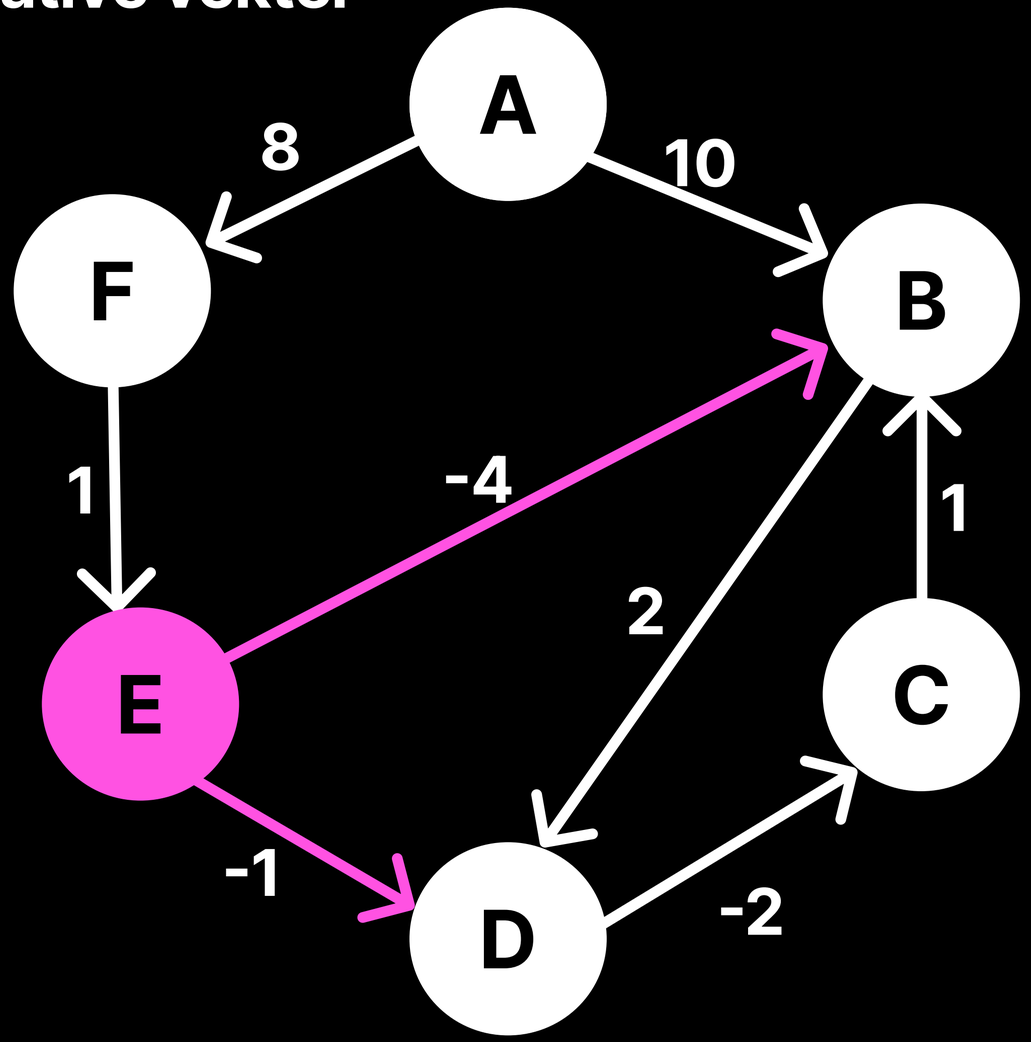
Negative vekter



queue	distance
A	0
B	10
C	10=10
D	12
E	9
F	8

Bellman-Ford

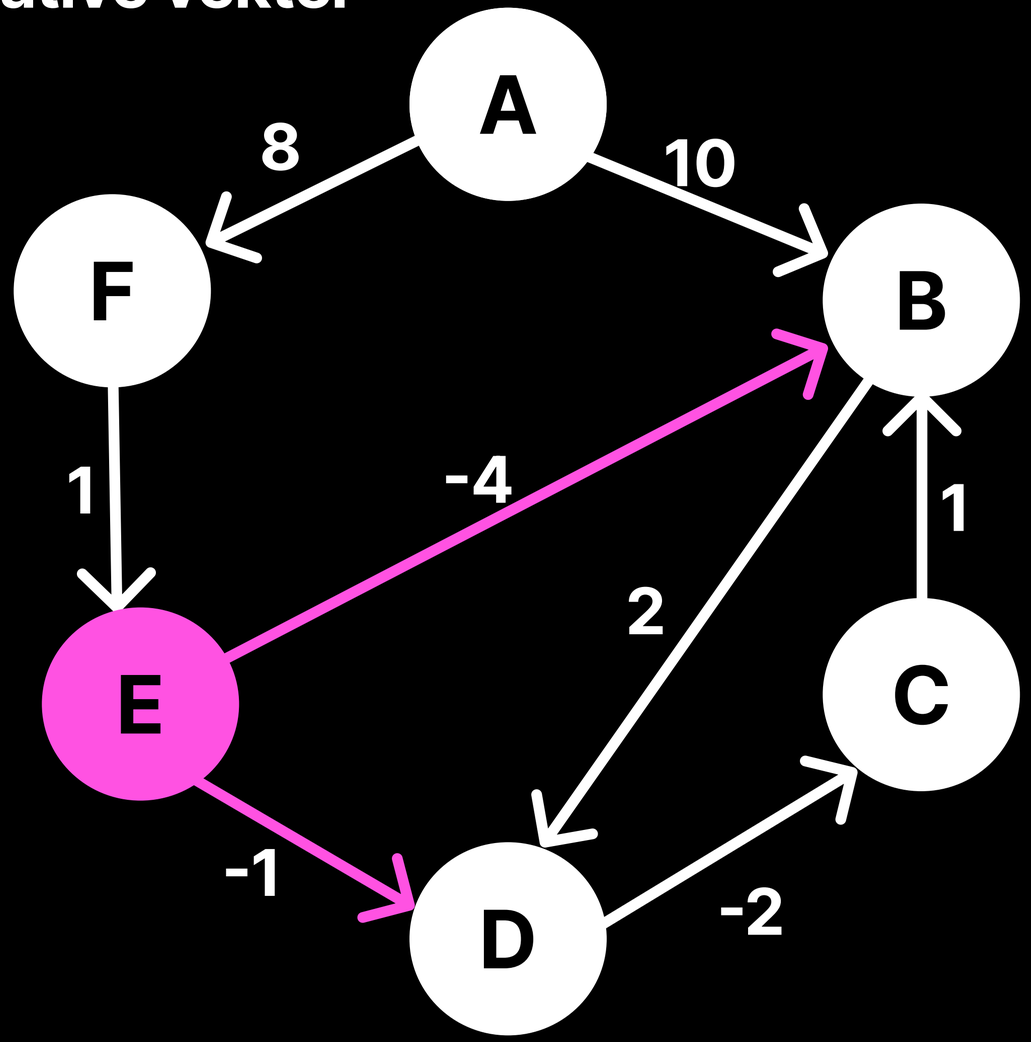
Negative vekter



queue	distance
A	0
B	10>5
C	10
D	12>8
E	9
F	8

Bellman-Ford

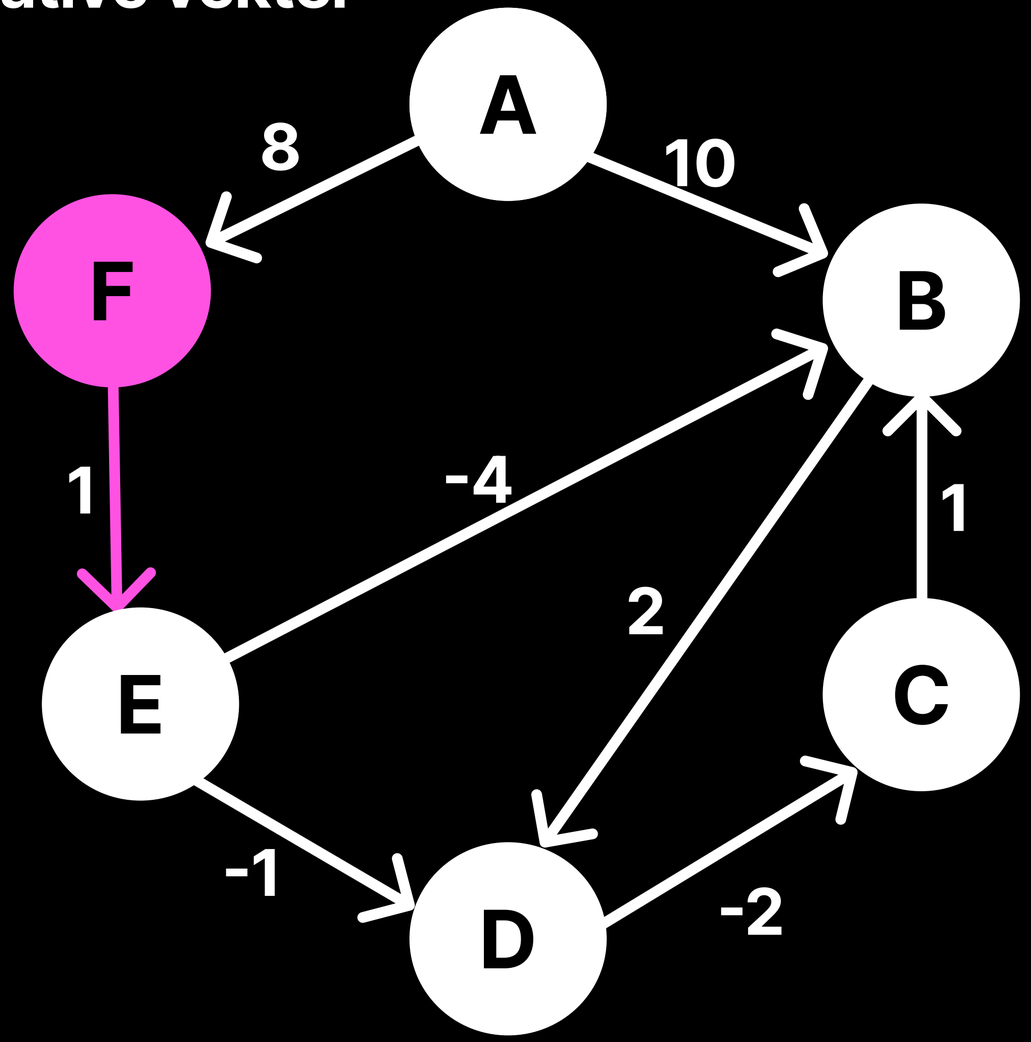
Negative vekter



queue	distance
A	0
B	5
C	10
D	8
E	9
F	8

Bellman-Ford

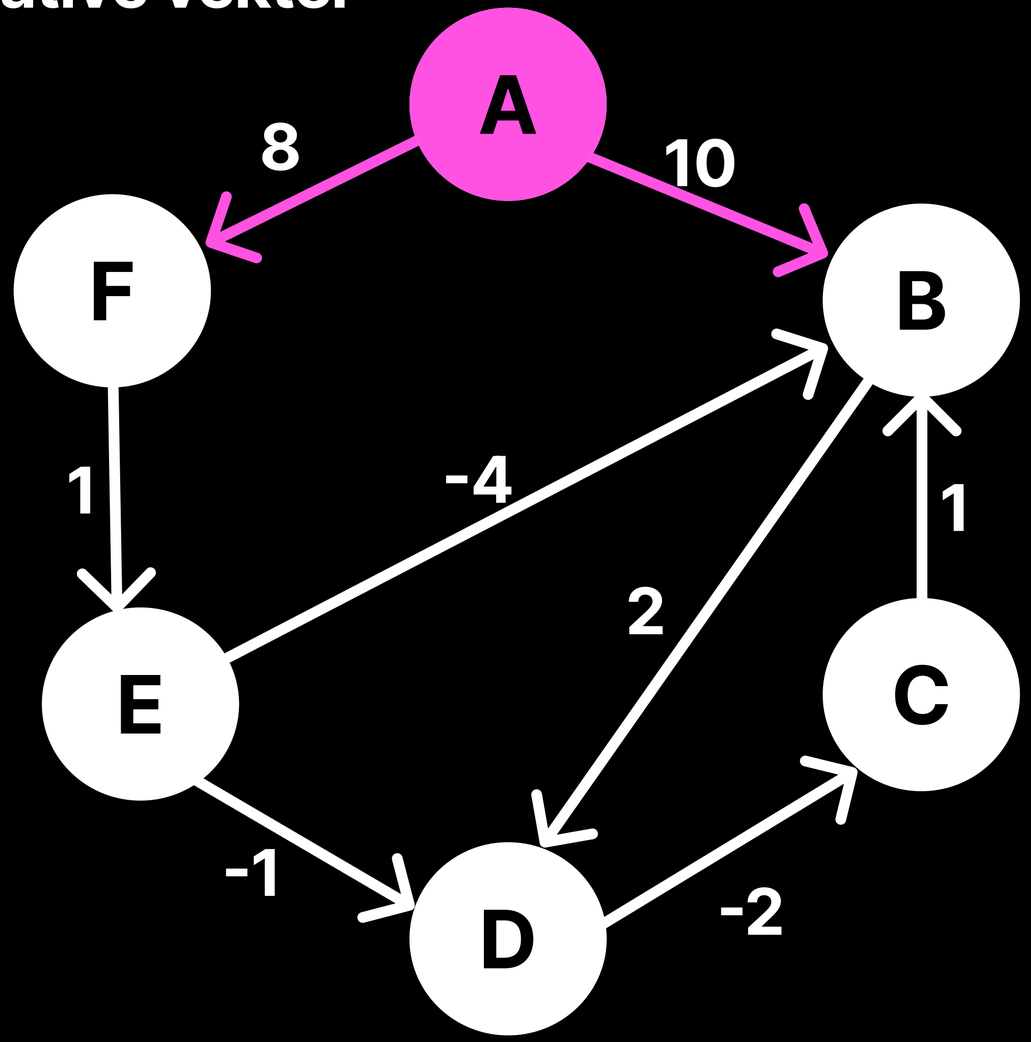
Negative vekter



queue	distance
A	0
B	5
C	10
D	8
E	9=9
F	8

Bellman-Ford

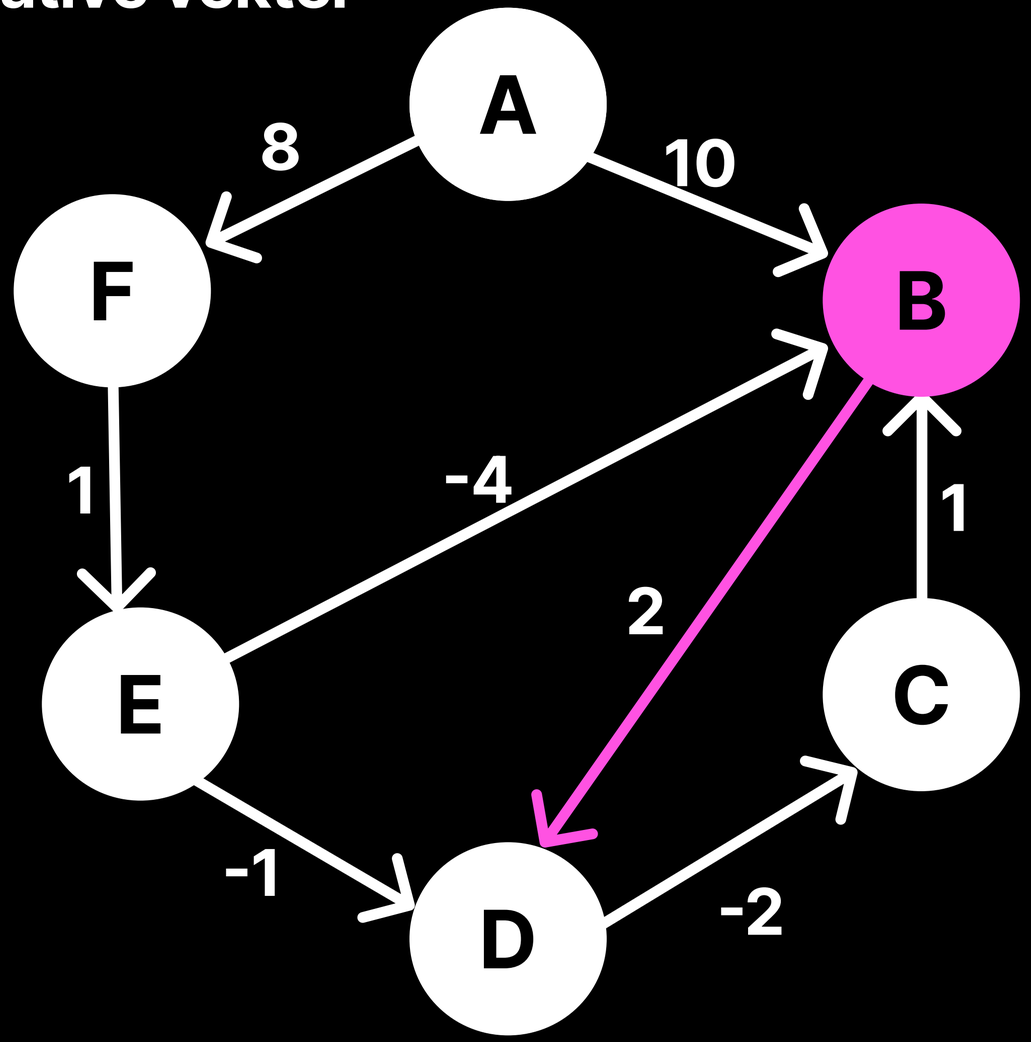
Negative vekter



queue	distance
A	0
B	5<10
C	10
D	8
E	9
F	8=8

Bellman-Ford

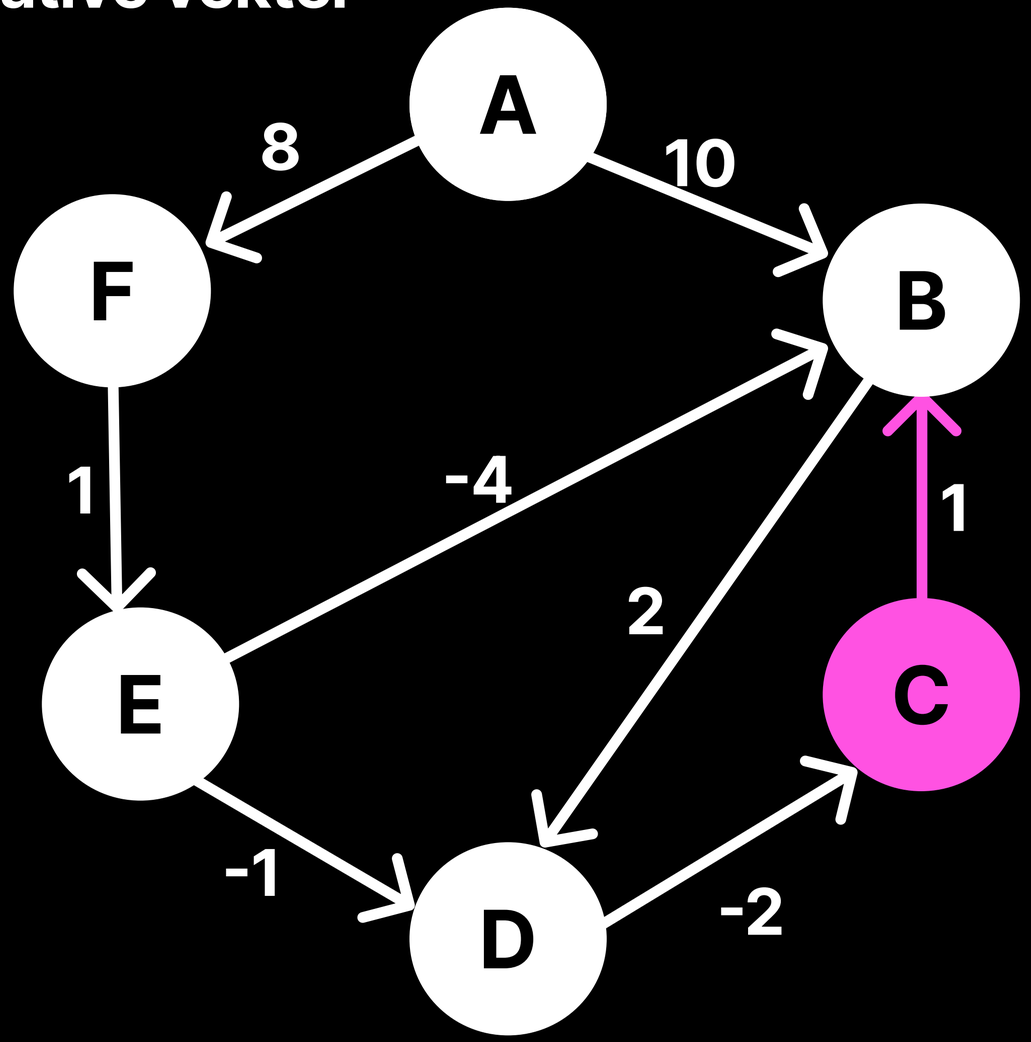
Negative vekter



queue	distance
A	0
B	5
C	10
D	8<12
E	9
F	8

Bellman-Ford

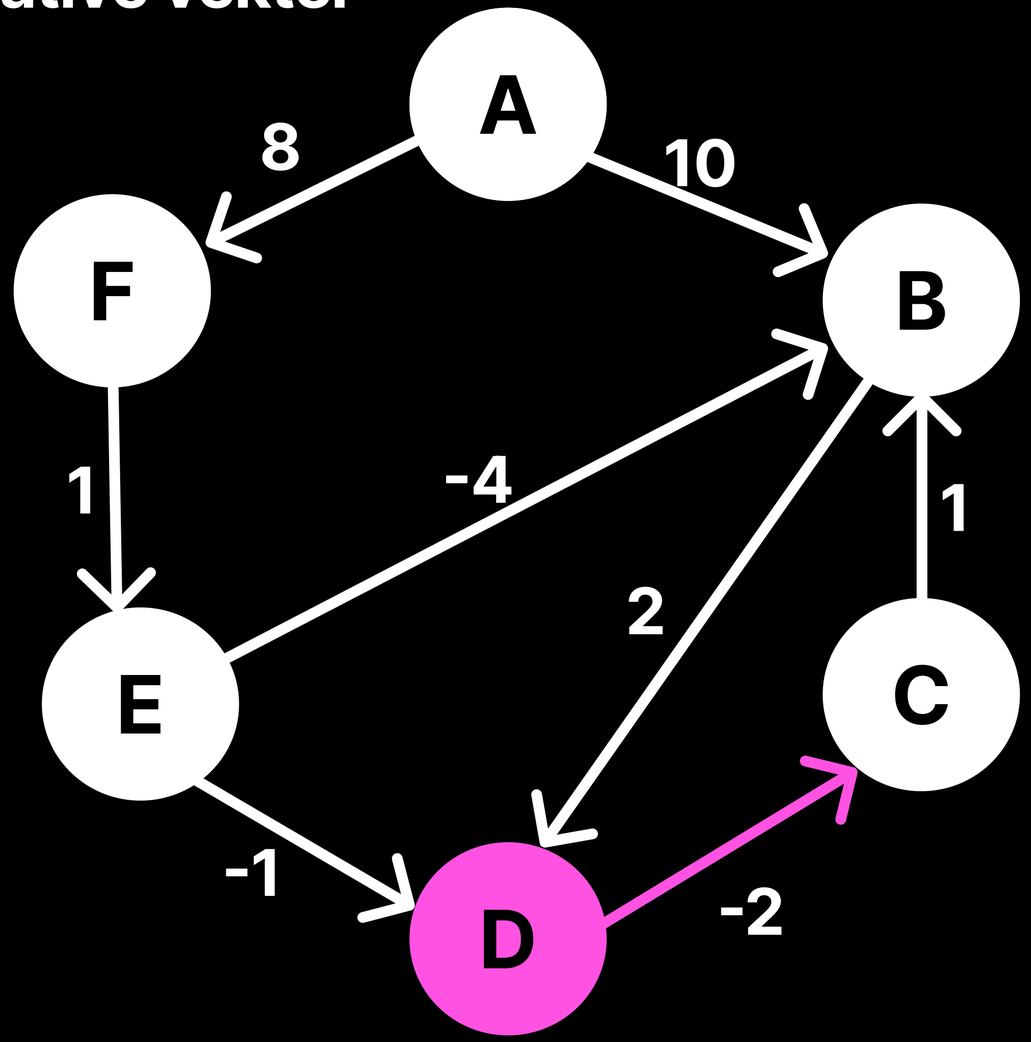
Negative vekter



queue	distance
A	0
B	5<11
C	10
D	8
E	9
F	8

Bellman-Ford

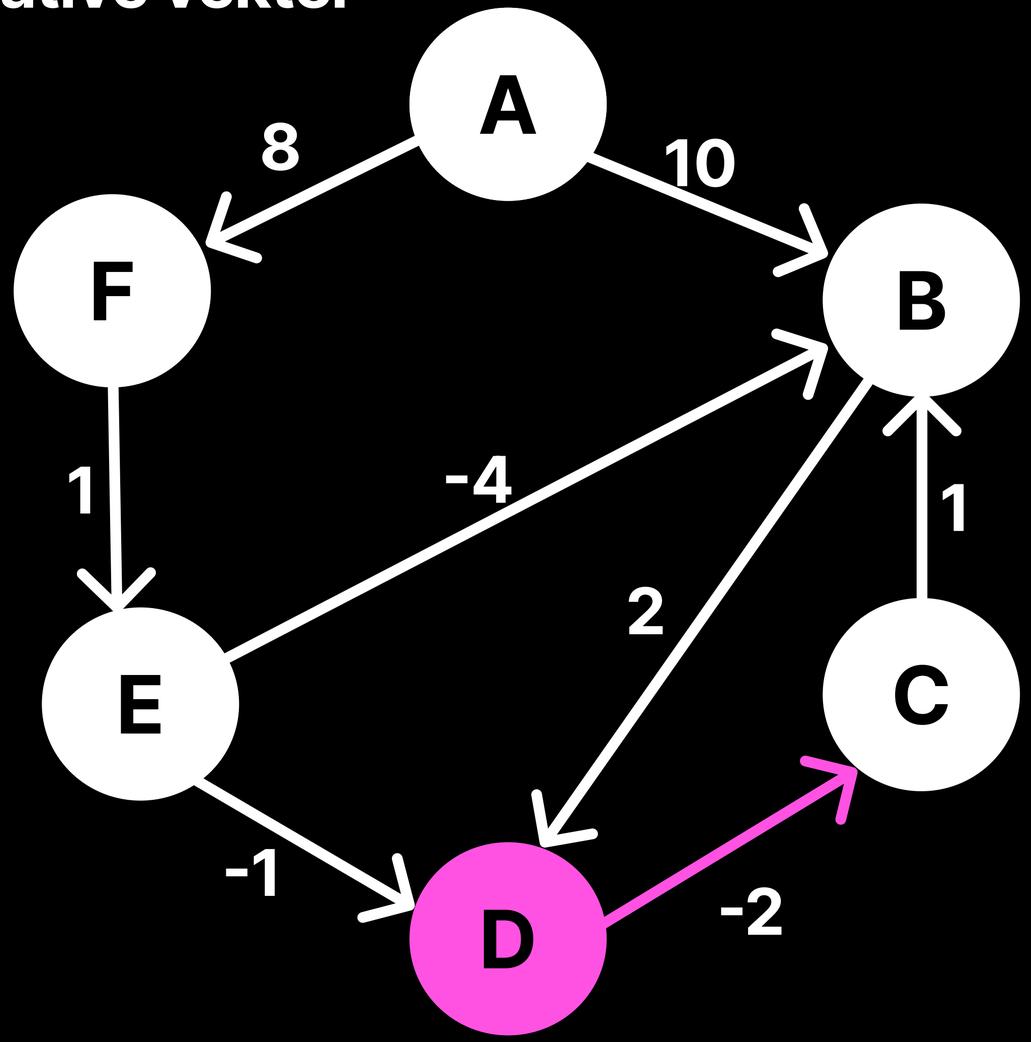
Negative vekter



queue	distance
A	0
B	5
C	10>6
D	8
E	9
F	8

Bellman-Ford

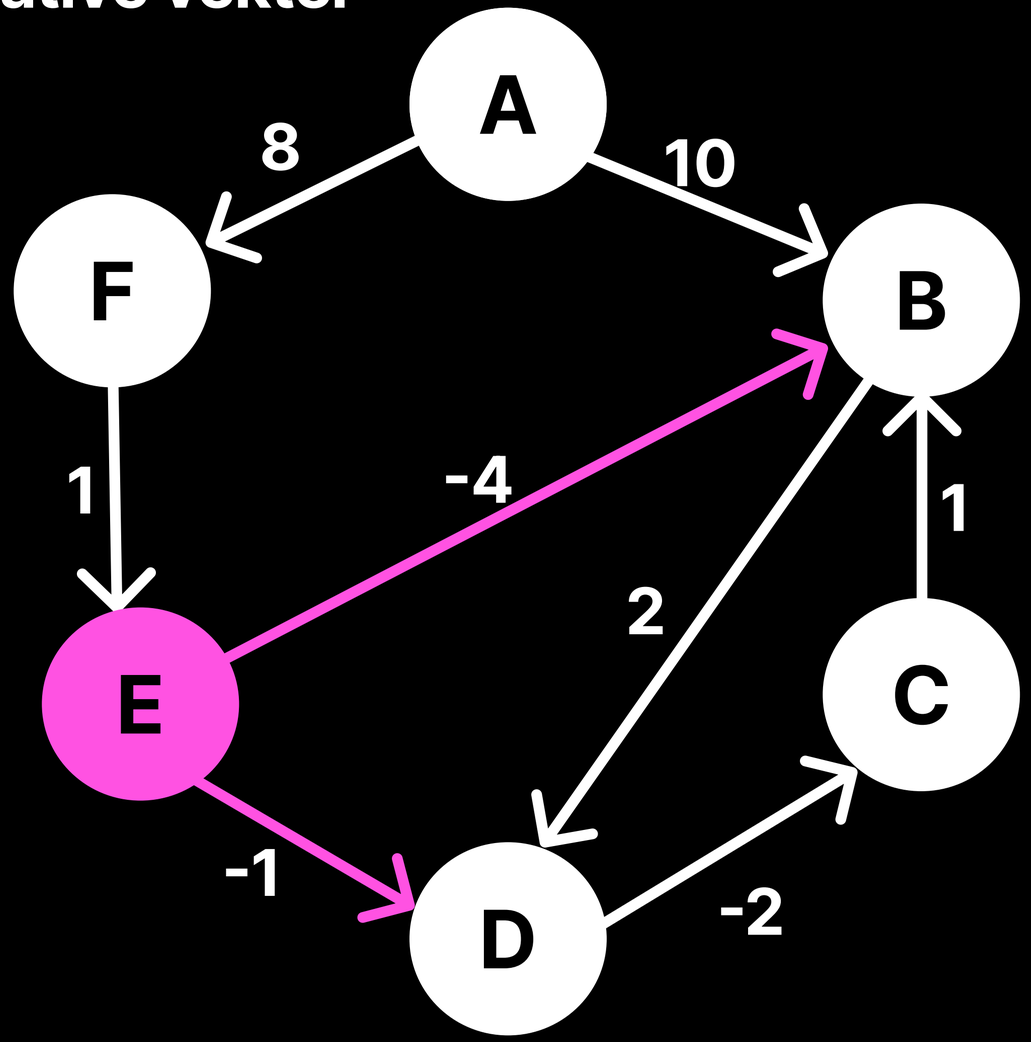
Negative vekter



queue	distance
A	0
B	5
C	6
D	8
E	9
F	8

Bellman-Ford

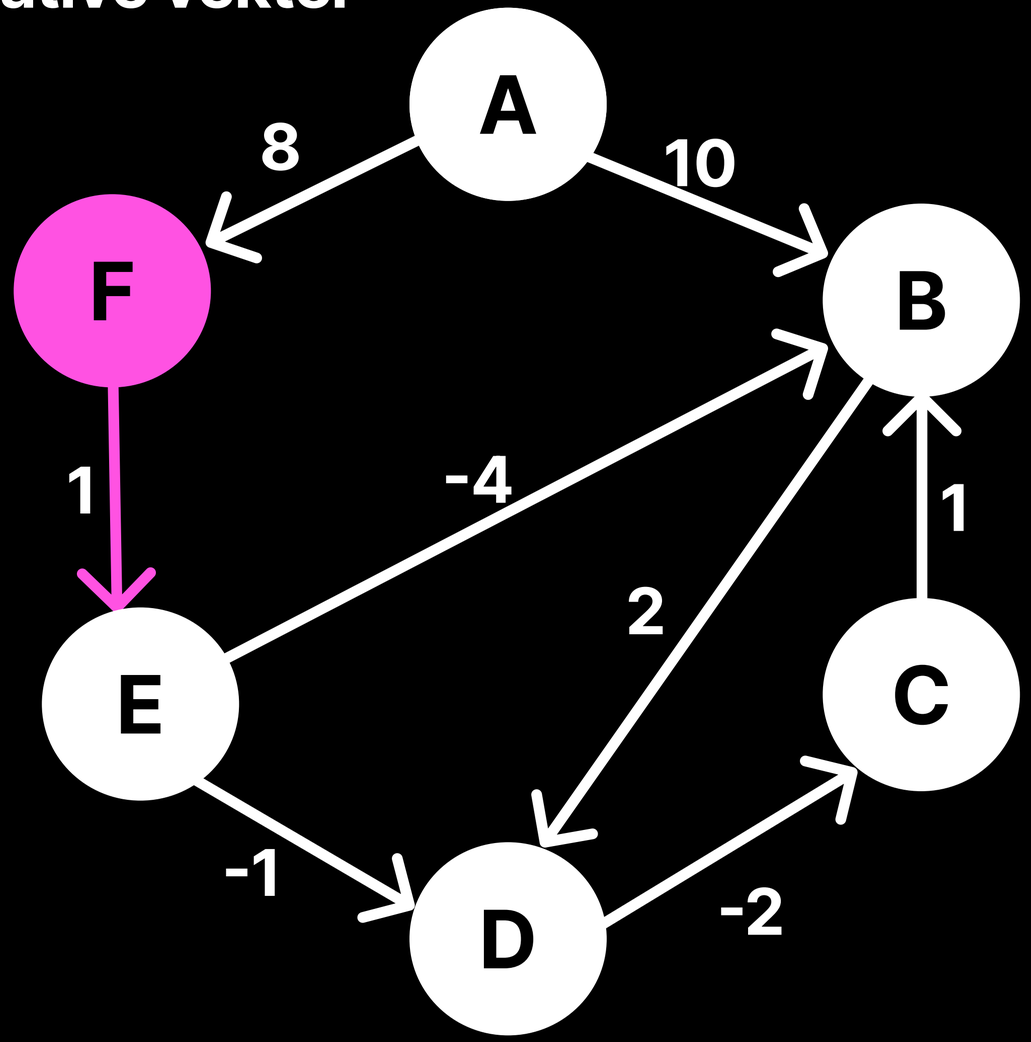
Negative vekter



queue	distance
A	0
B	5=5
C	6
D	8=8
E	9
F	8

Bellman-Ford

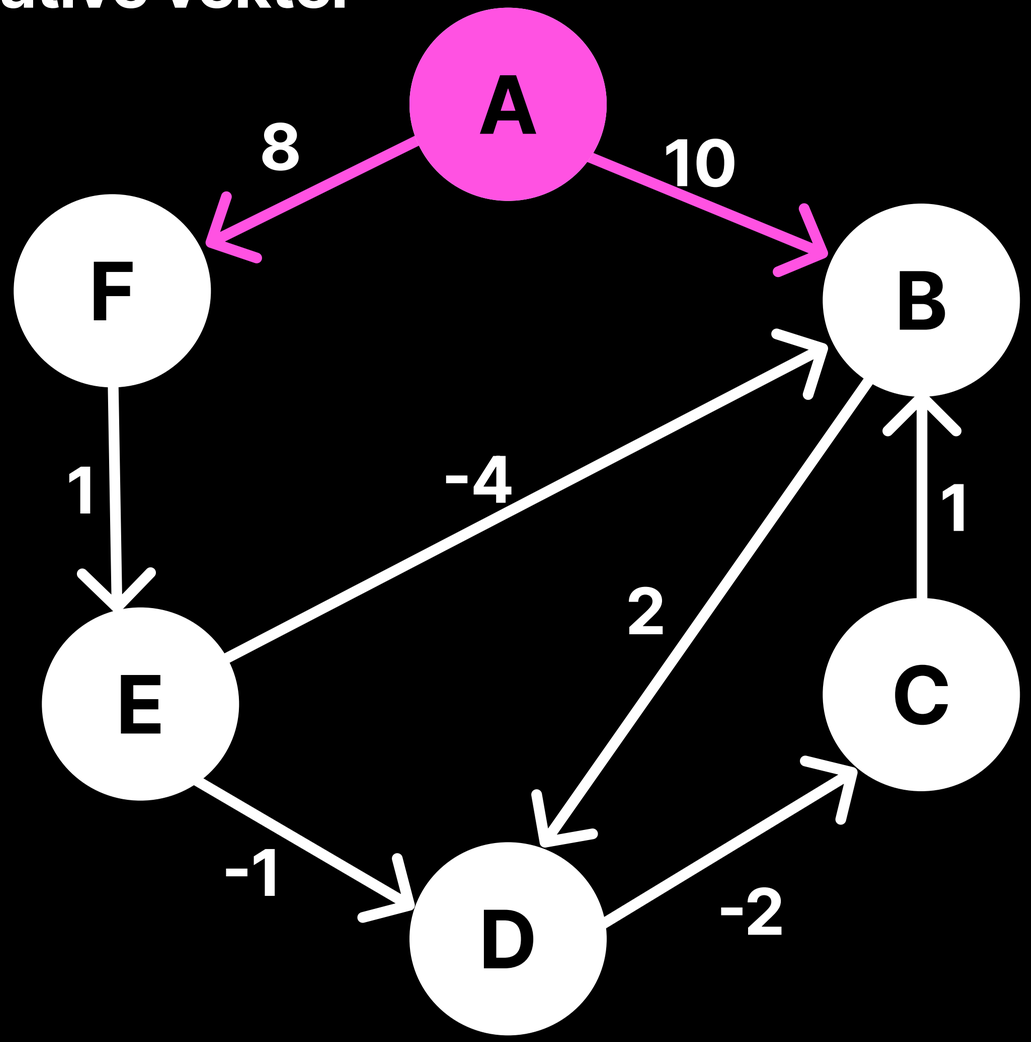
Negative vekter



queue	distance
A	0
B	5
C	6
D	8
E	9=9
F	8

Bellman-Ford

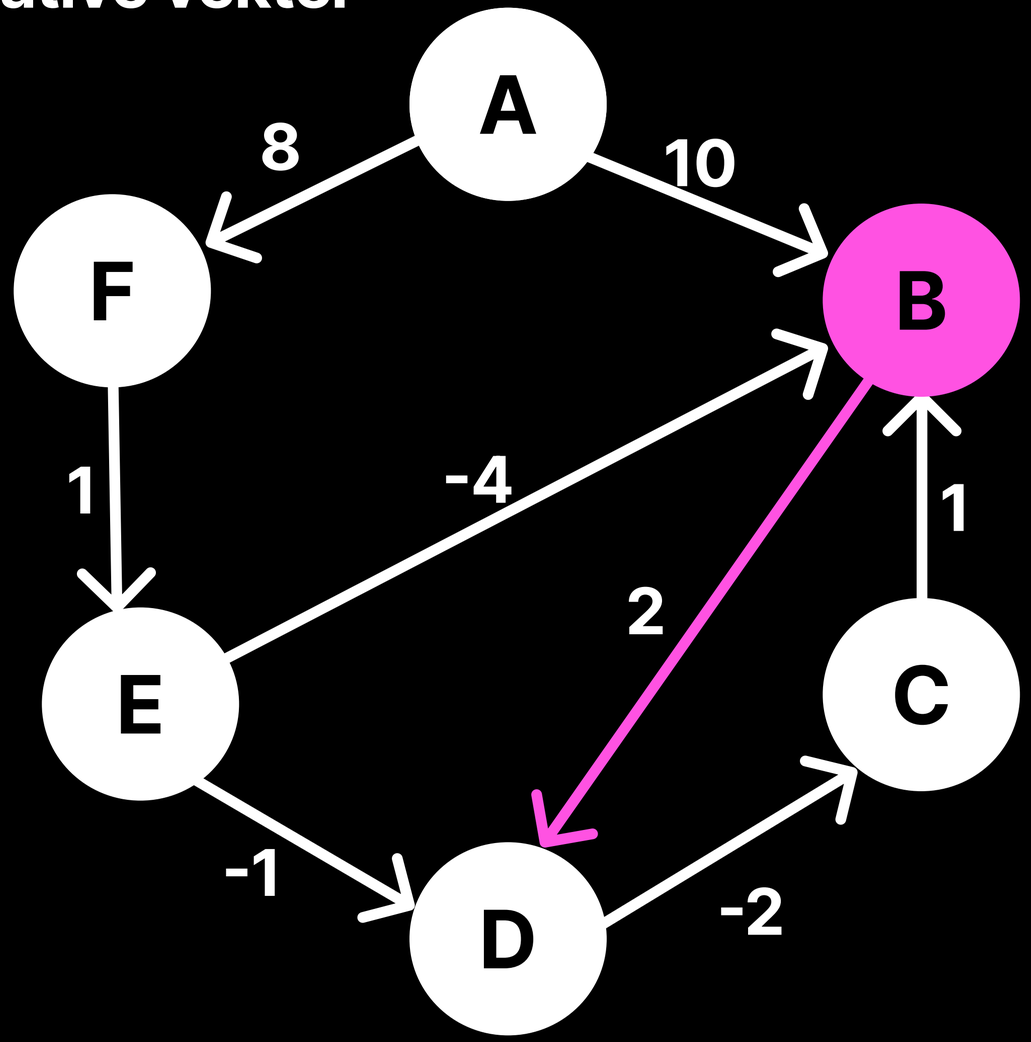
Negative vekter



queue	distance
A	0
B	5<10
C	6
D	8
E	9
F	8=8

Bellman-Ford

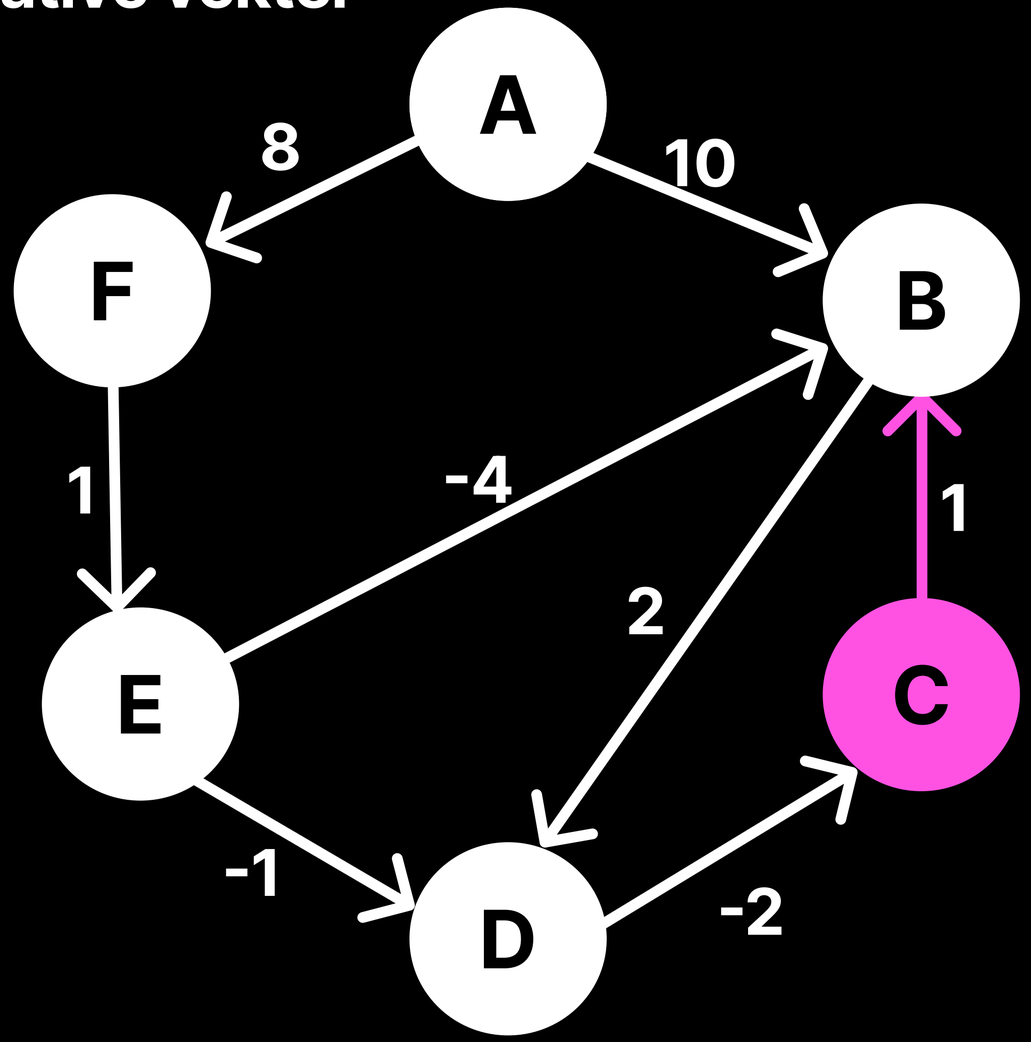
Negative vekter



queue	distance
A	0
B	5
C	6
D	8<12
E	9
F	8

Bellman-Ford

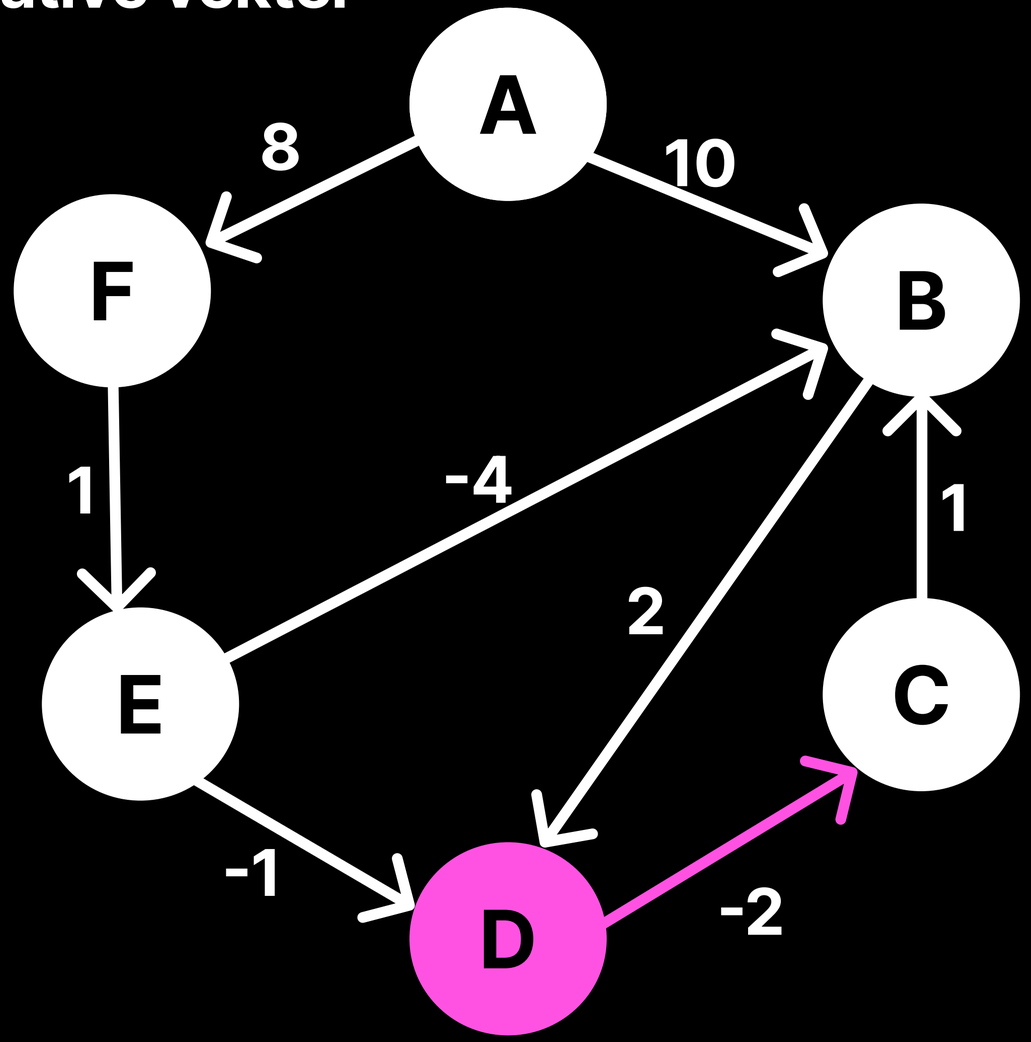
Negative vekter



queue	distance
A	0
B	5<7
C	6
D	8
E	9
F	8

Bellman-Ford

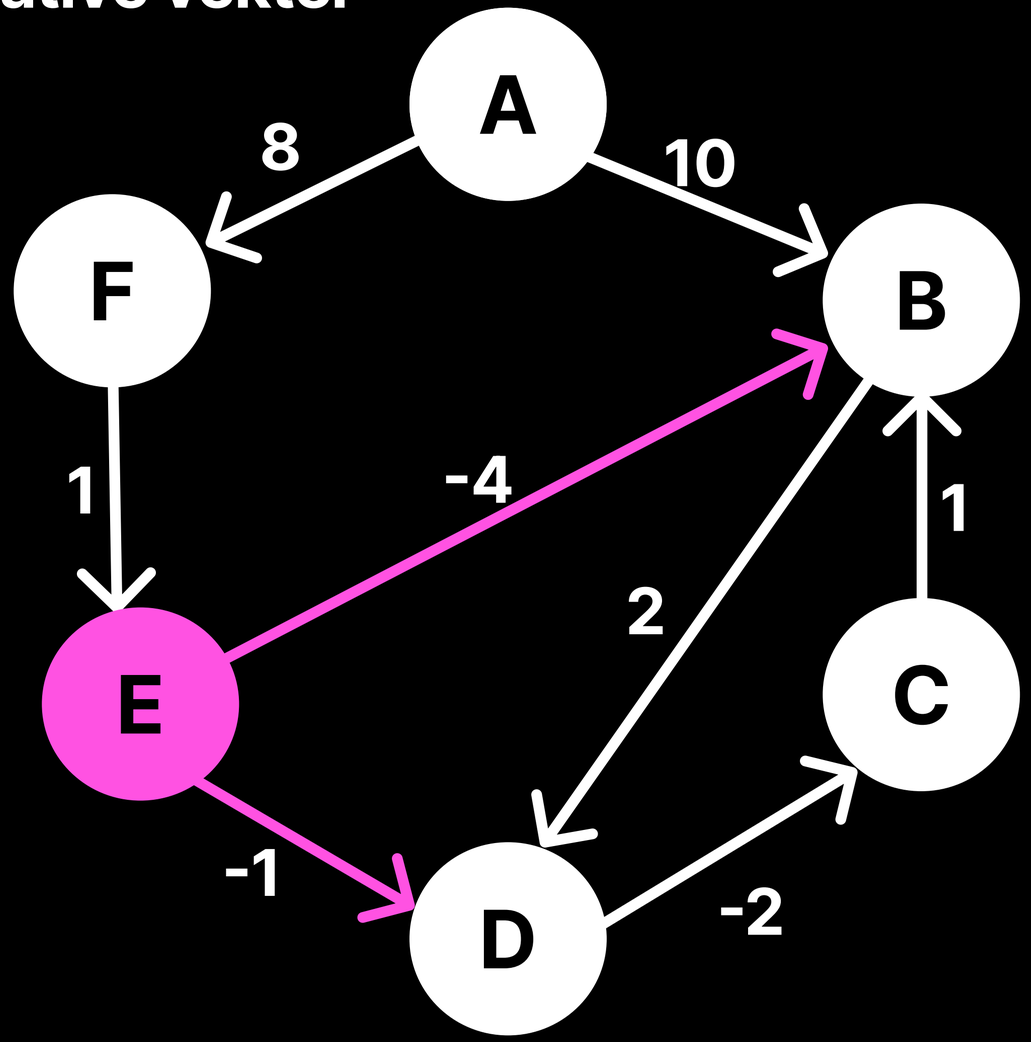
Negative vekter



queue	distance
A	0
B	5
C	6=6
D	8
E	9
F	8

Bellman-Ford

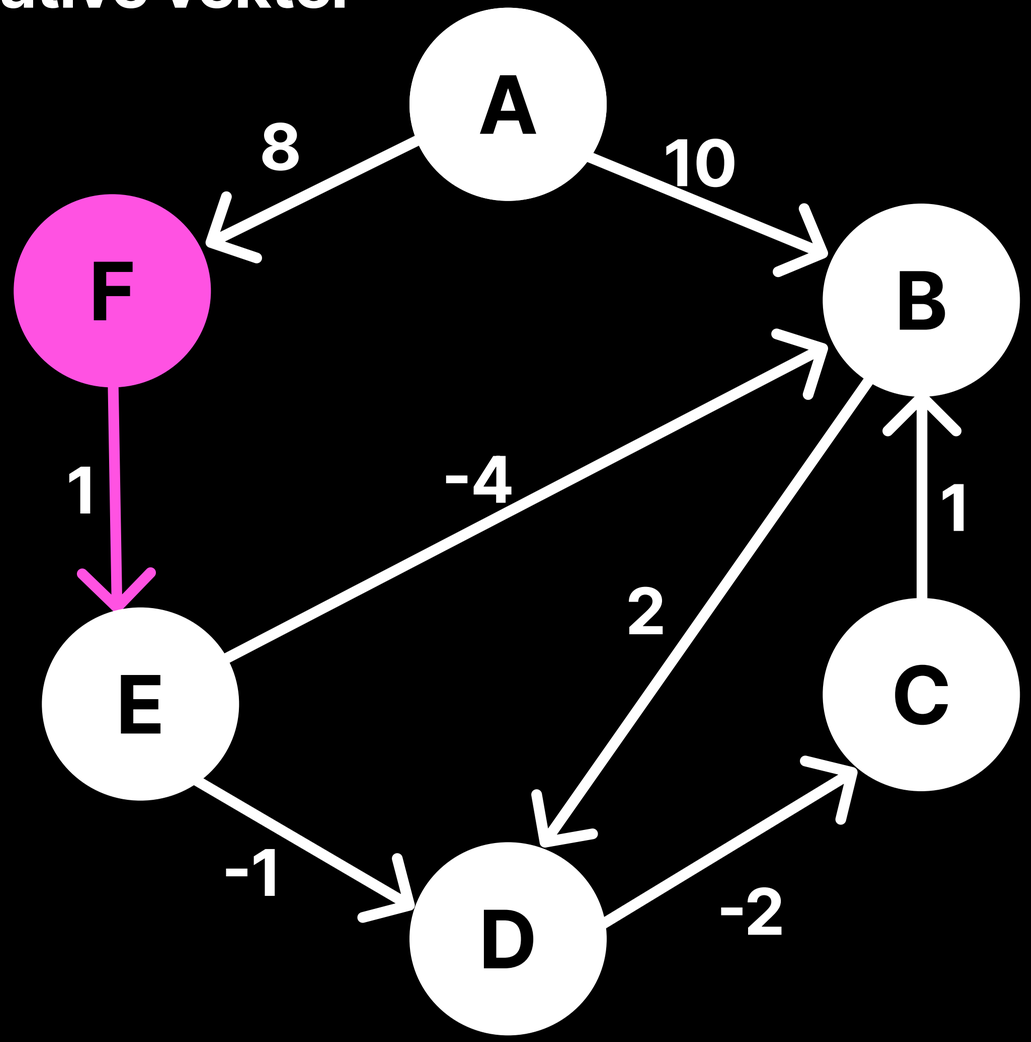
Negative vekter



queue	distance
A	0
B	5=5
C	6
D	8=8
E	9
F	8

Bellman-Ford

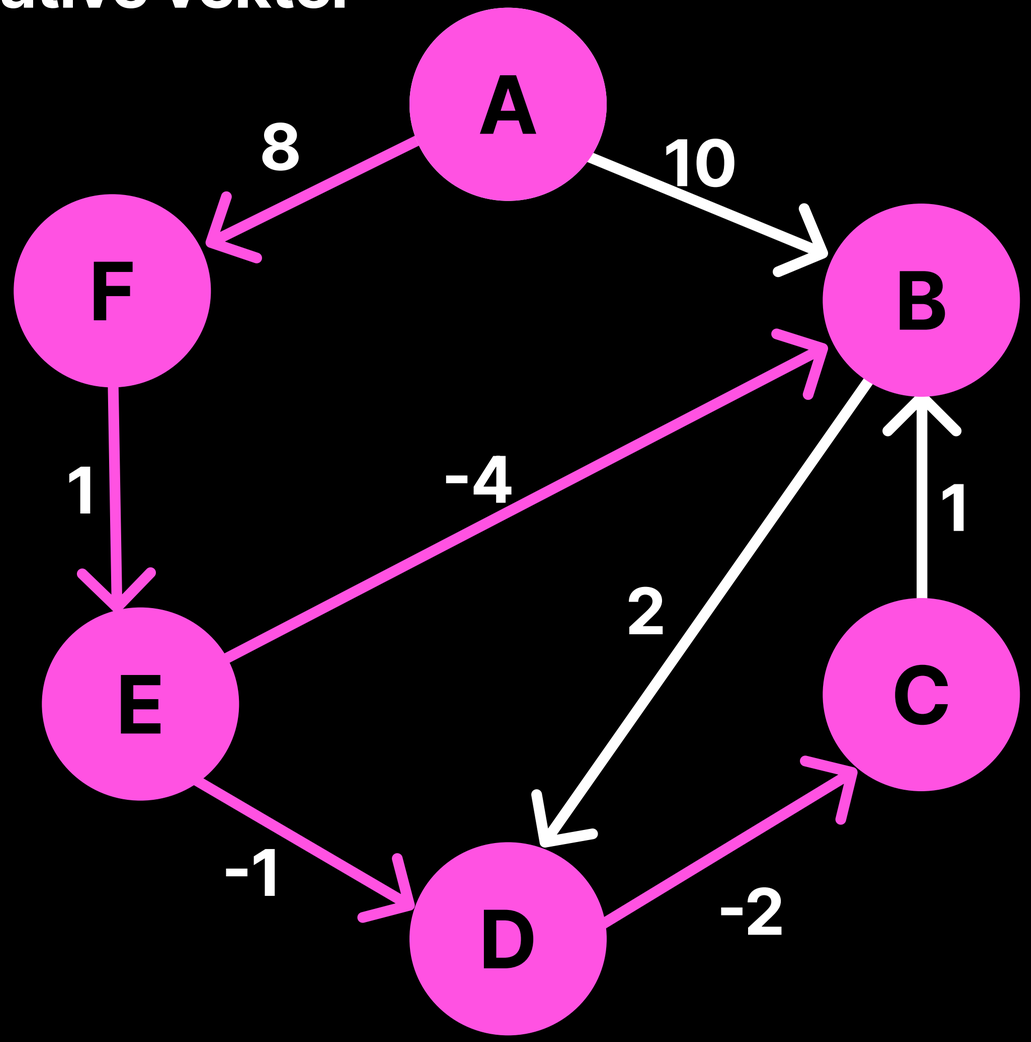
Negative vekter



queue	distance
A	0
B	5
C	6
D	8
E	9=9
F	8

Bellman-Ford

Negative vekter



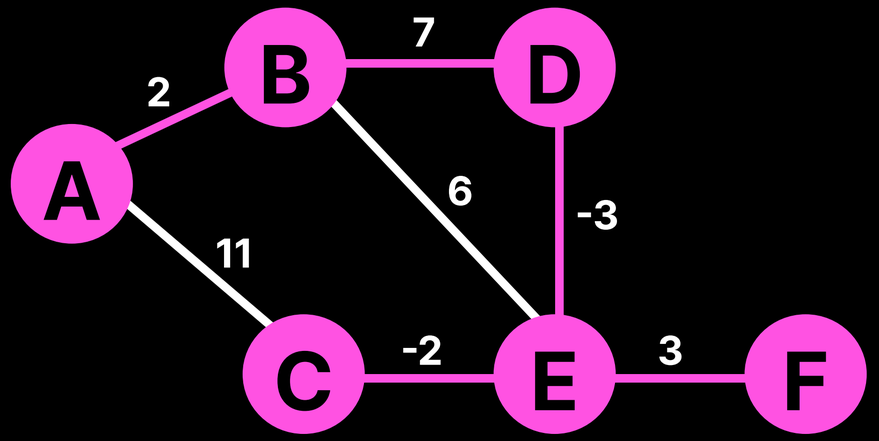
queue	distance
A	0
B	5
C	6
D	8
E	9
F	8

Spenntrær

Trær: En sammenhengende, urettet og asyklisk graf er et **tre**.

- nøyaktig $|V| - 1$ kanter

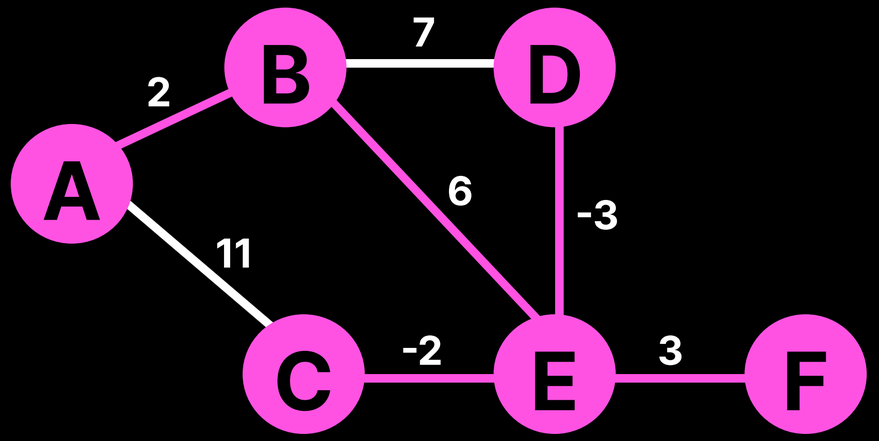
Det i rosa er et spenntre, men ikke et minimalt spenntre!



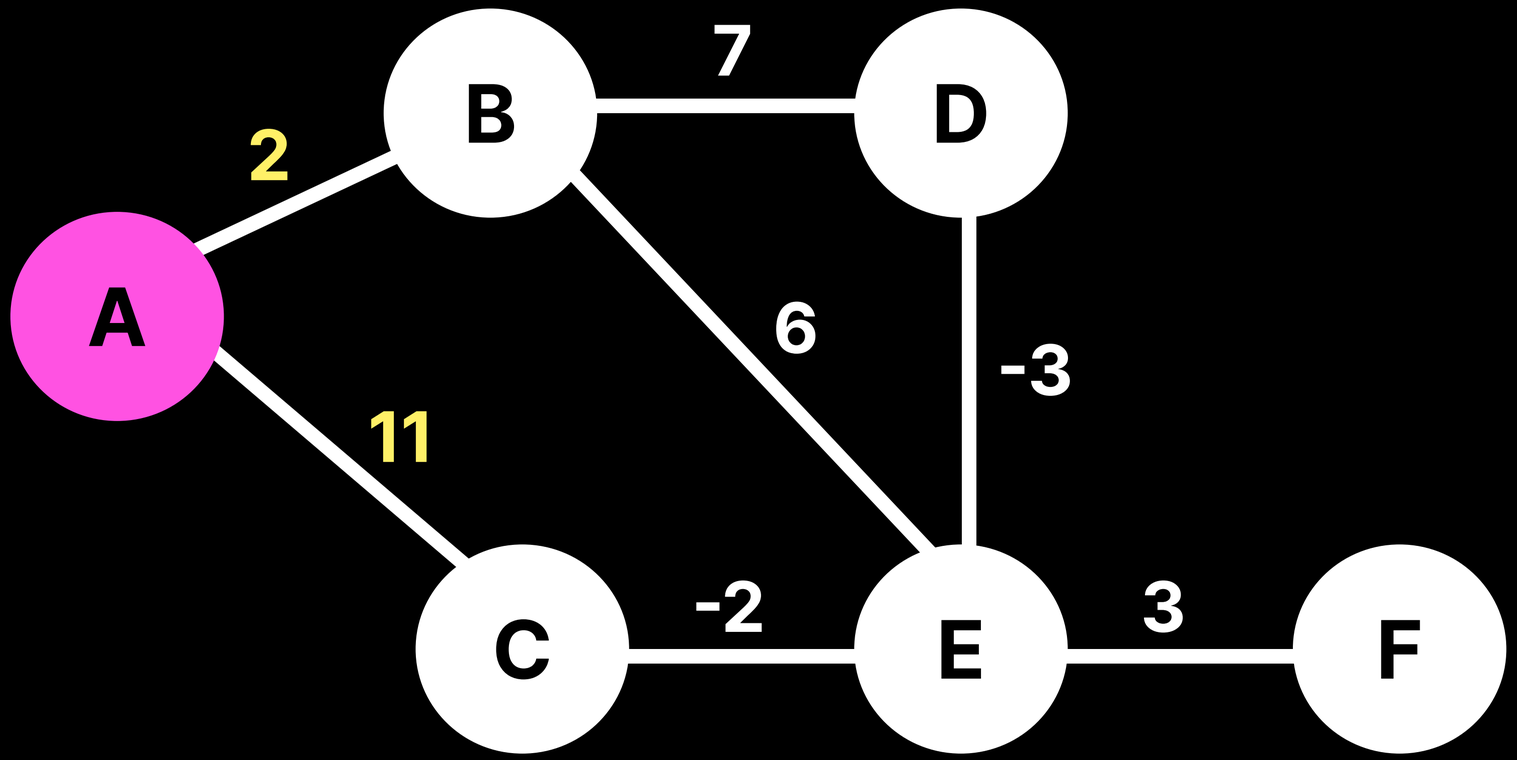
Minimale spenntrær

I et minimalt spenntre leter vi etter kantene med minst vekt, fram til vi har knyttet sammen alle nodene i grafen (uten sykler! Det skal være et tre)

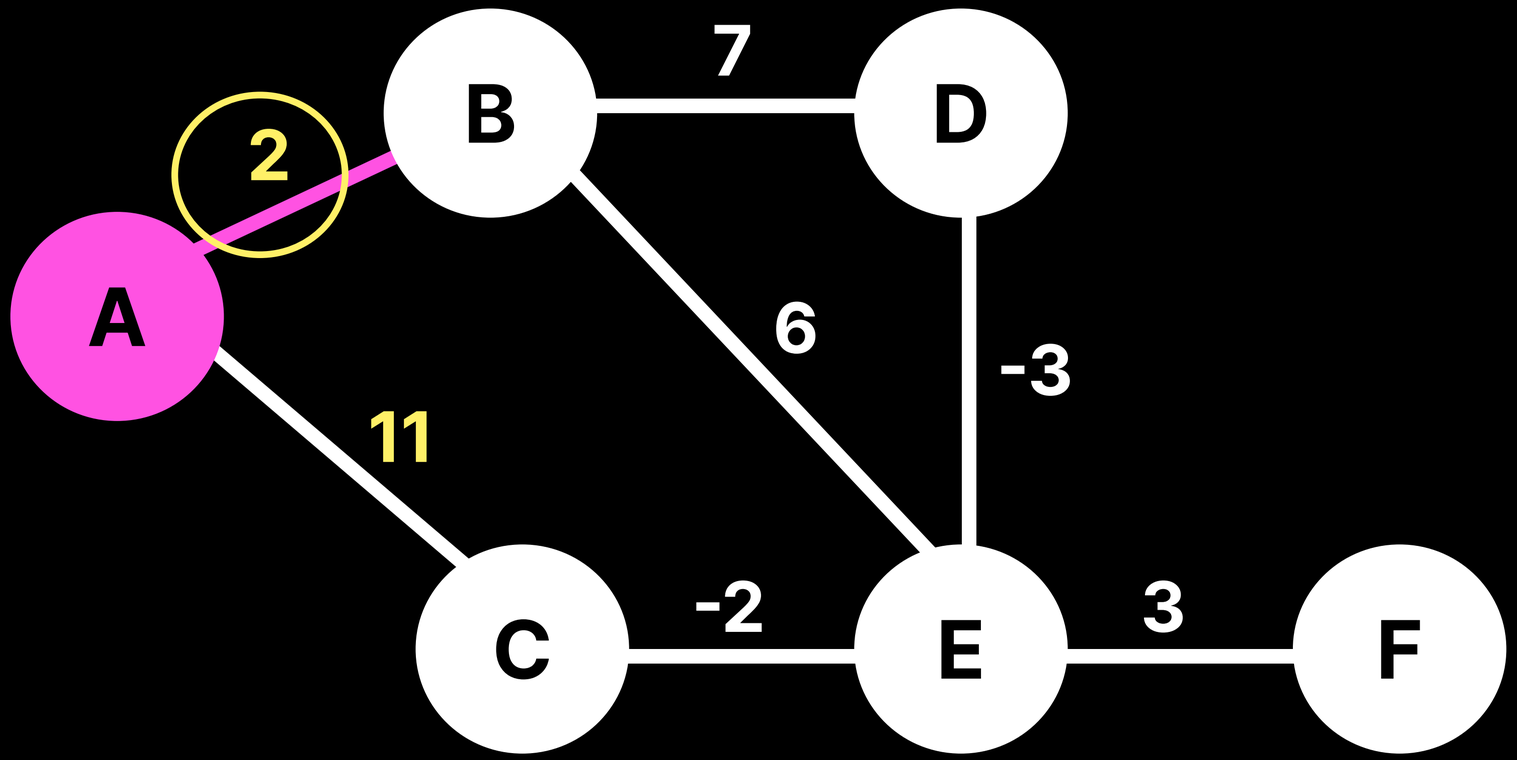
Dette er altså et minimalt spenntre.
Merk at vi isteden for kanten (B,D) nå bruker kanten (B,E) fordi denne har lavere vekt.



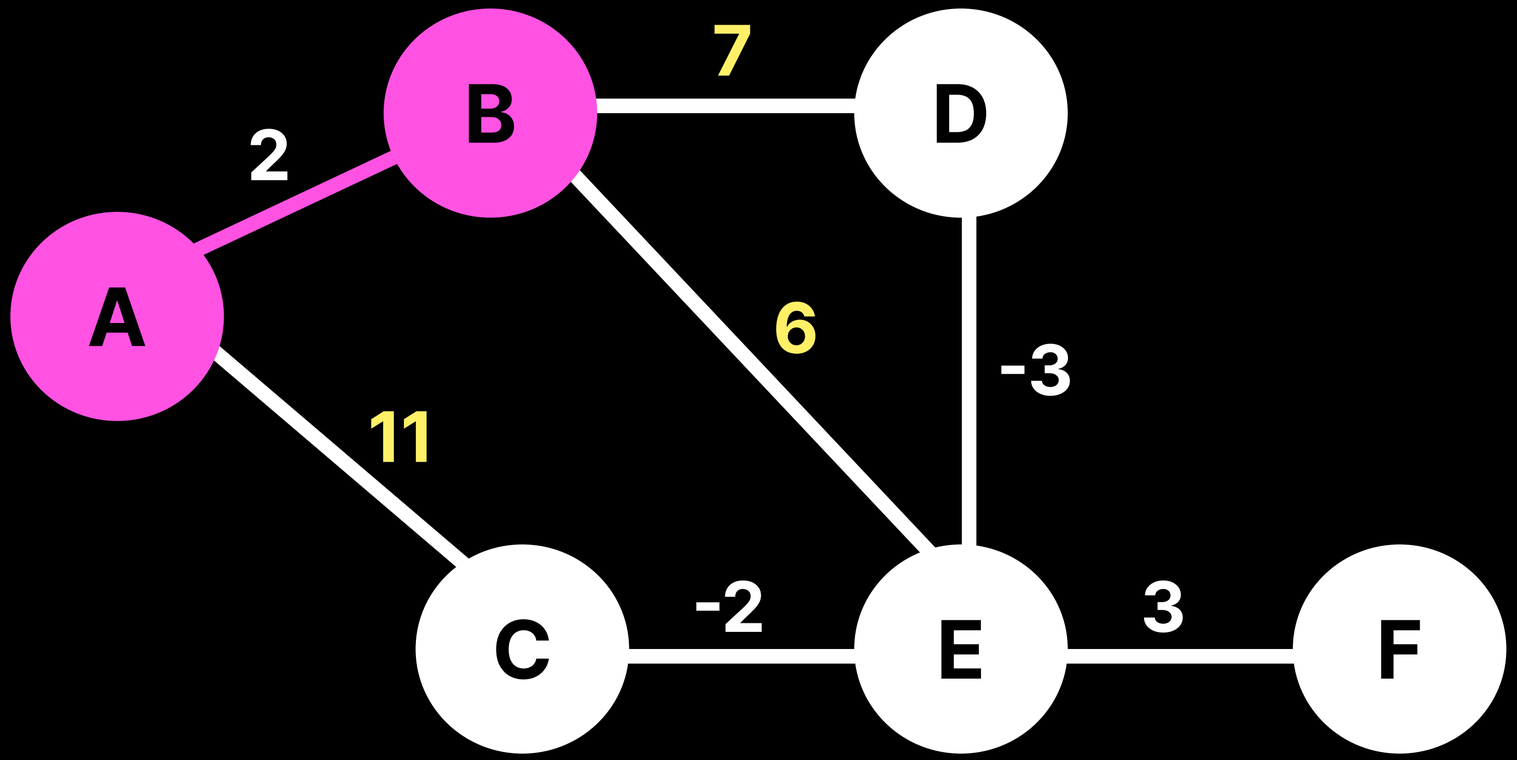
Prims



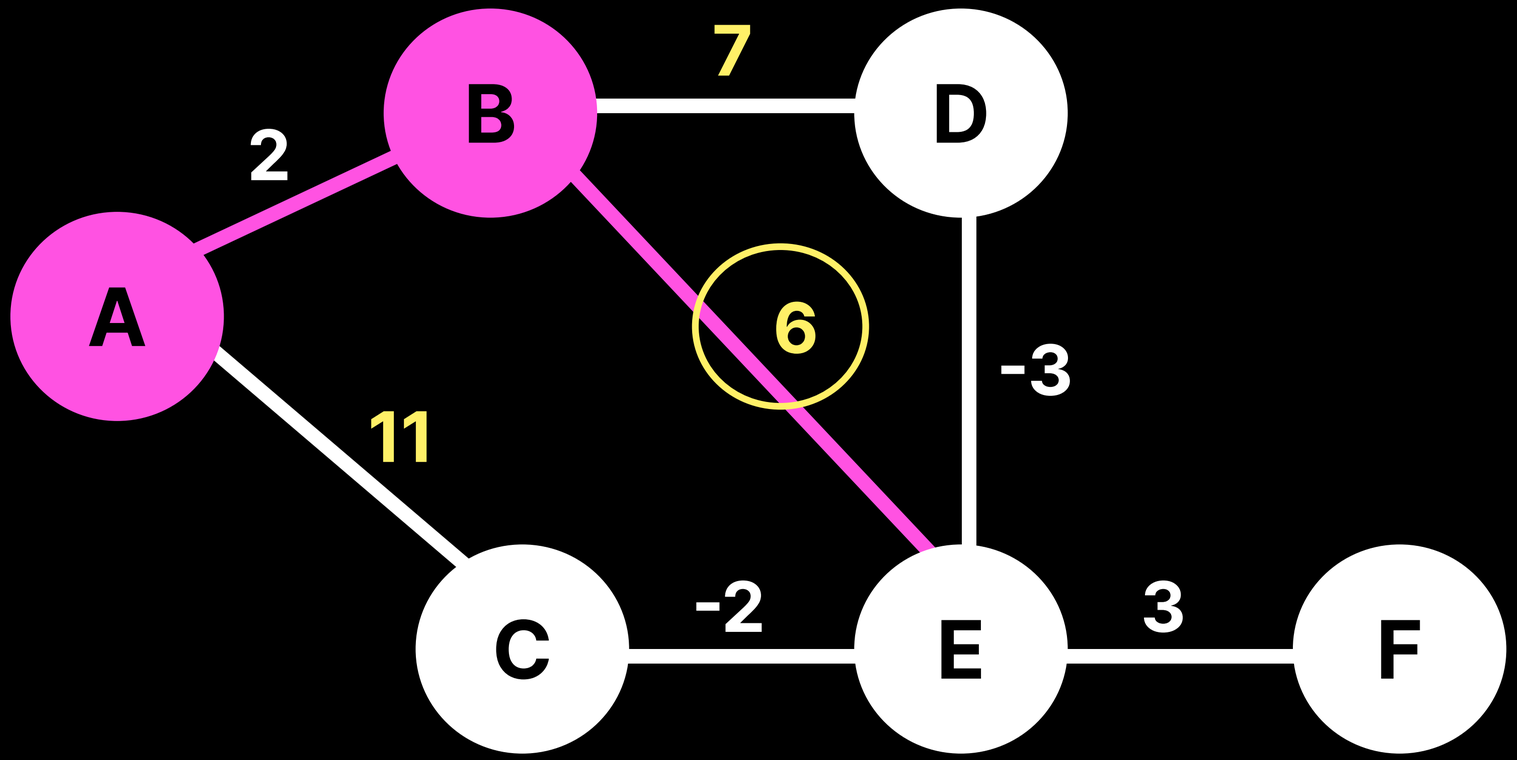
Prims



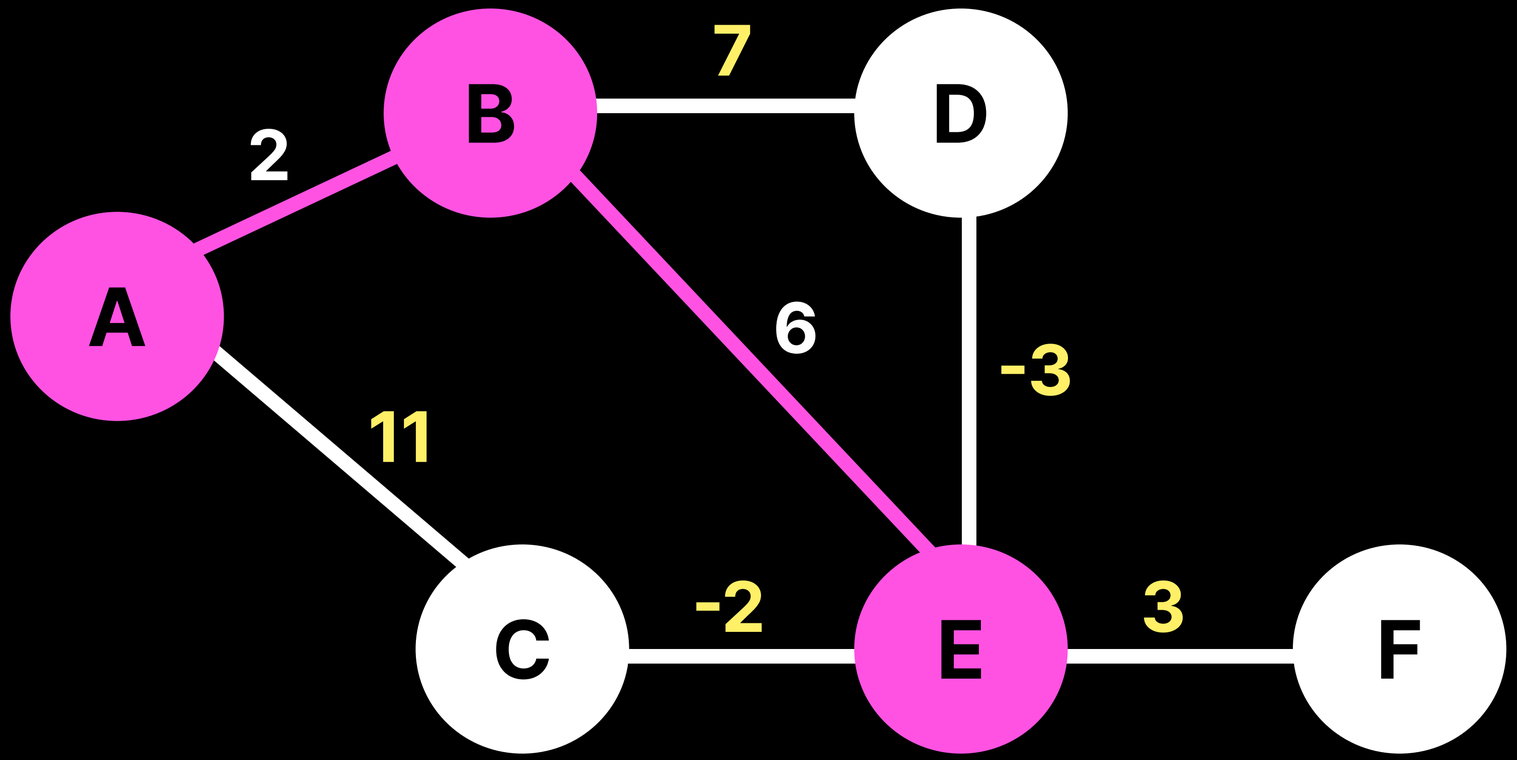
Prims



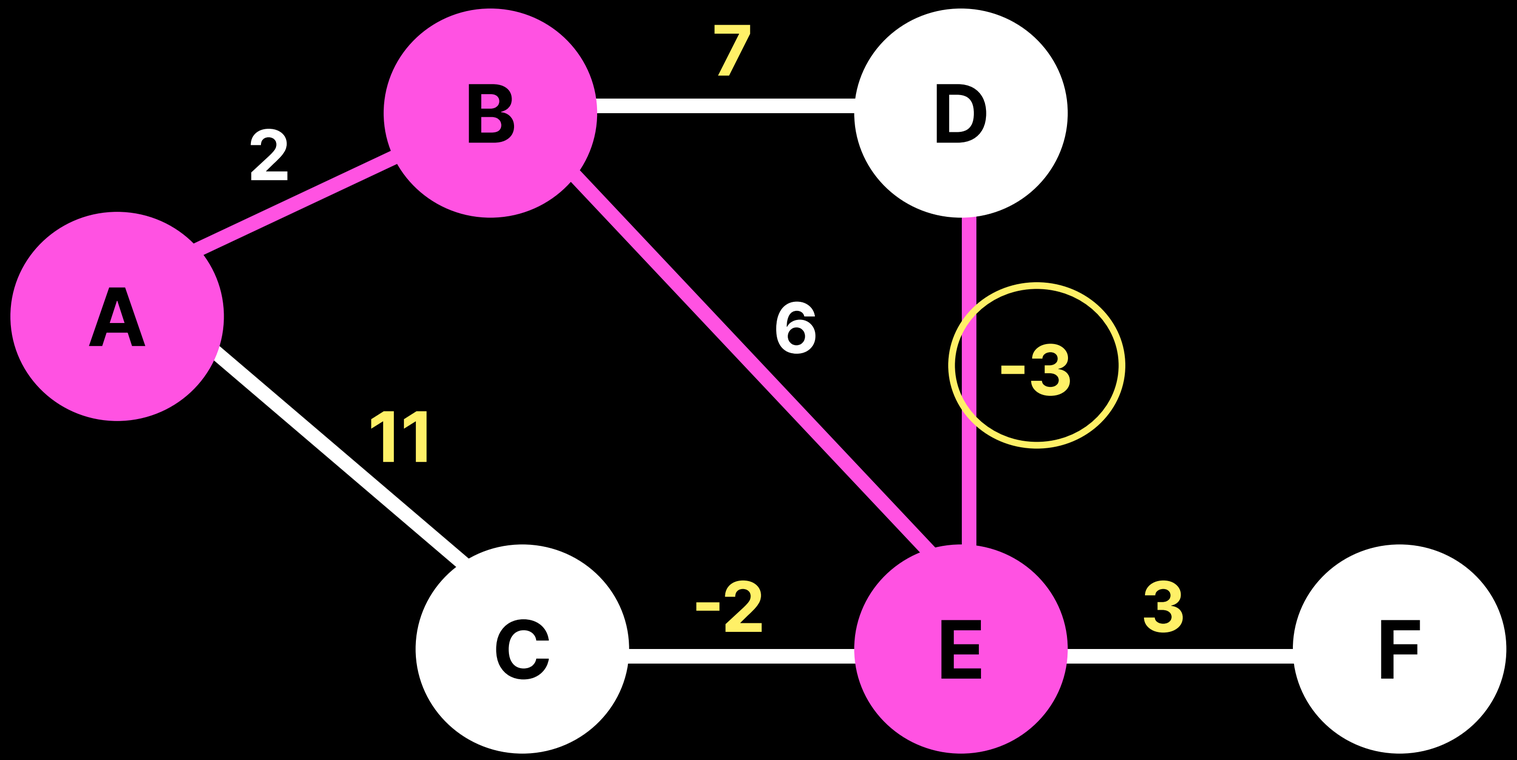
Prims



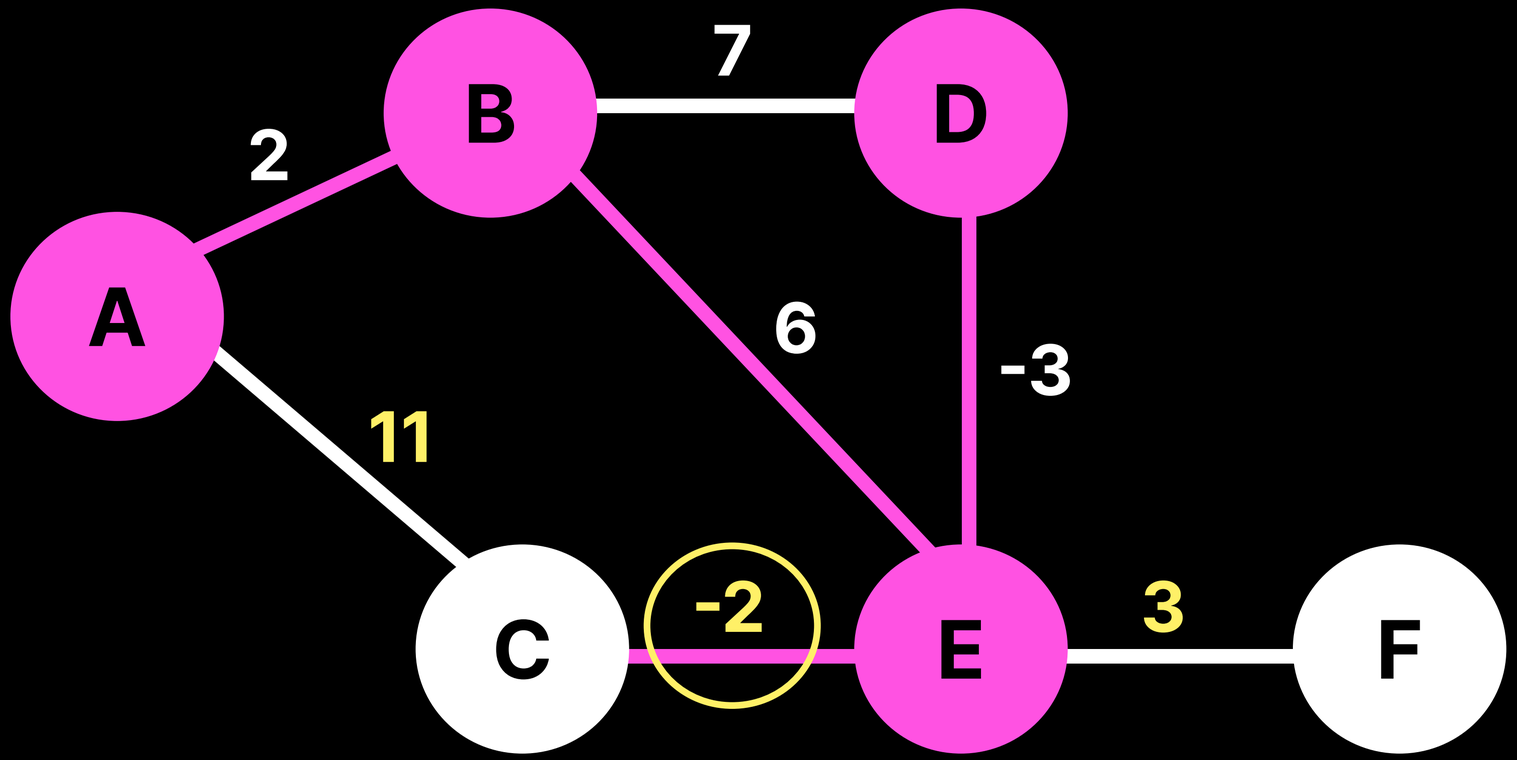
Prims



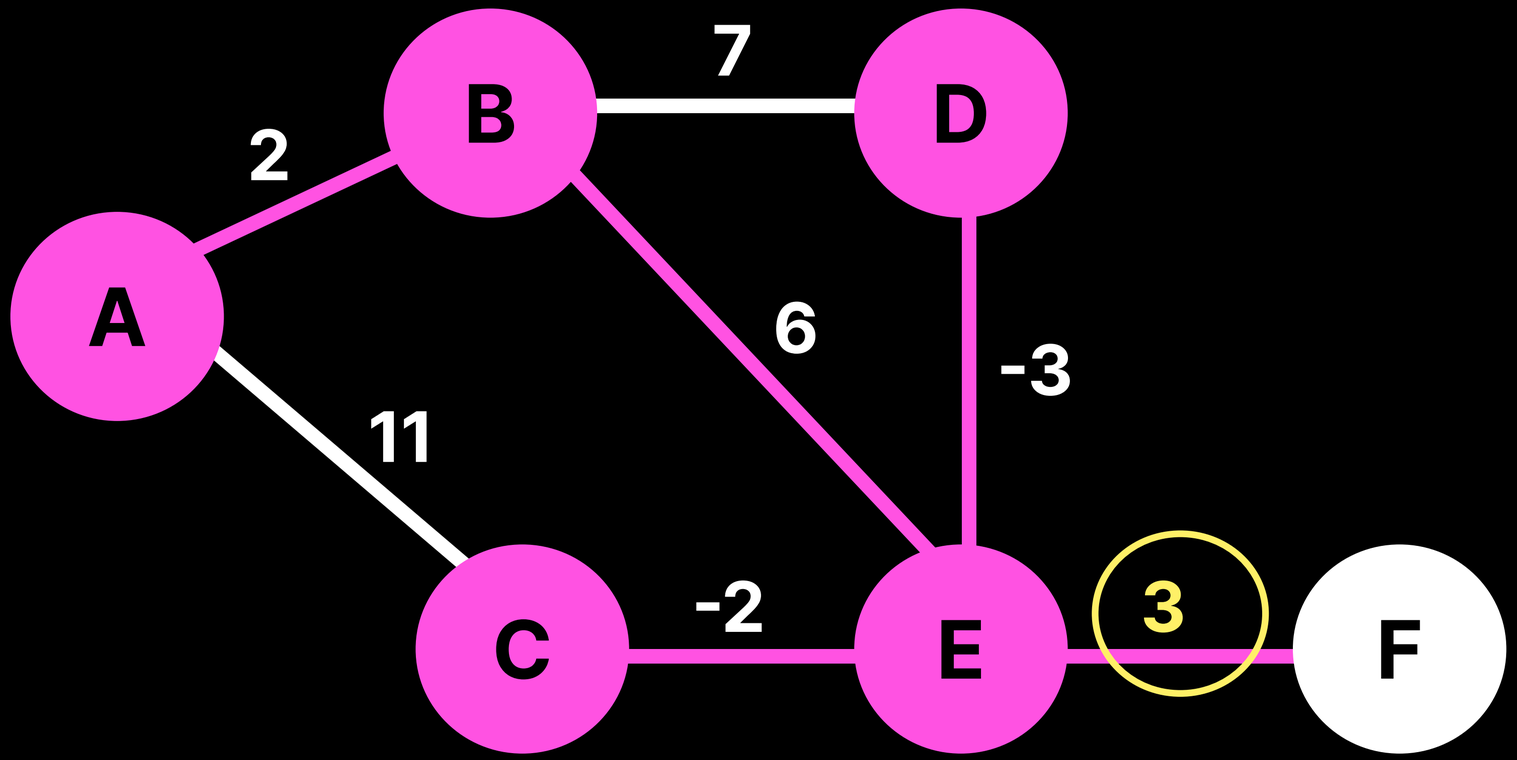
Prims



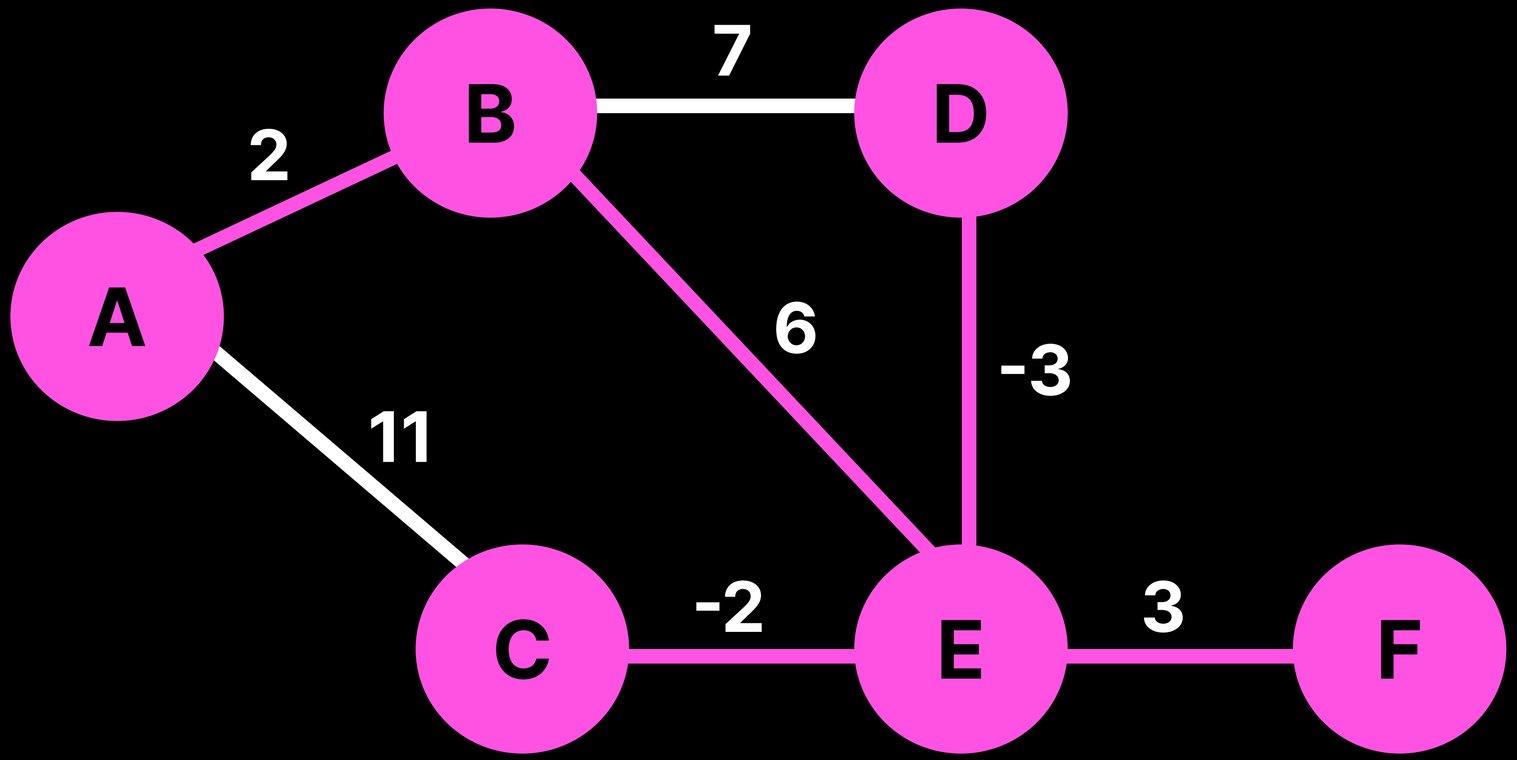
Prims



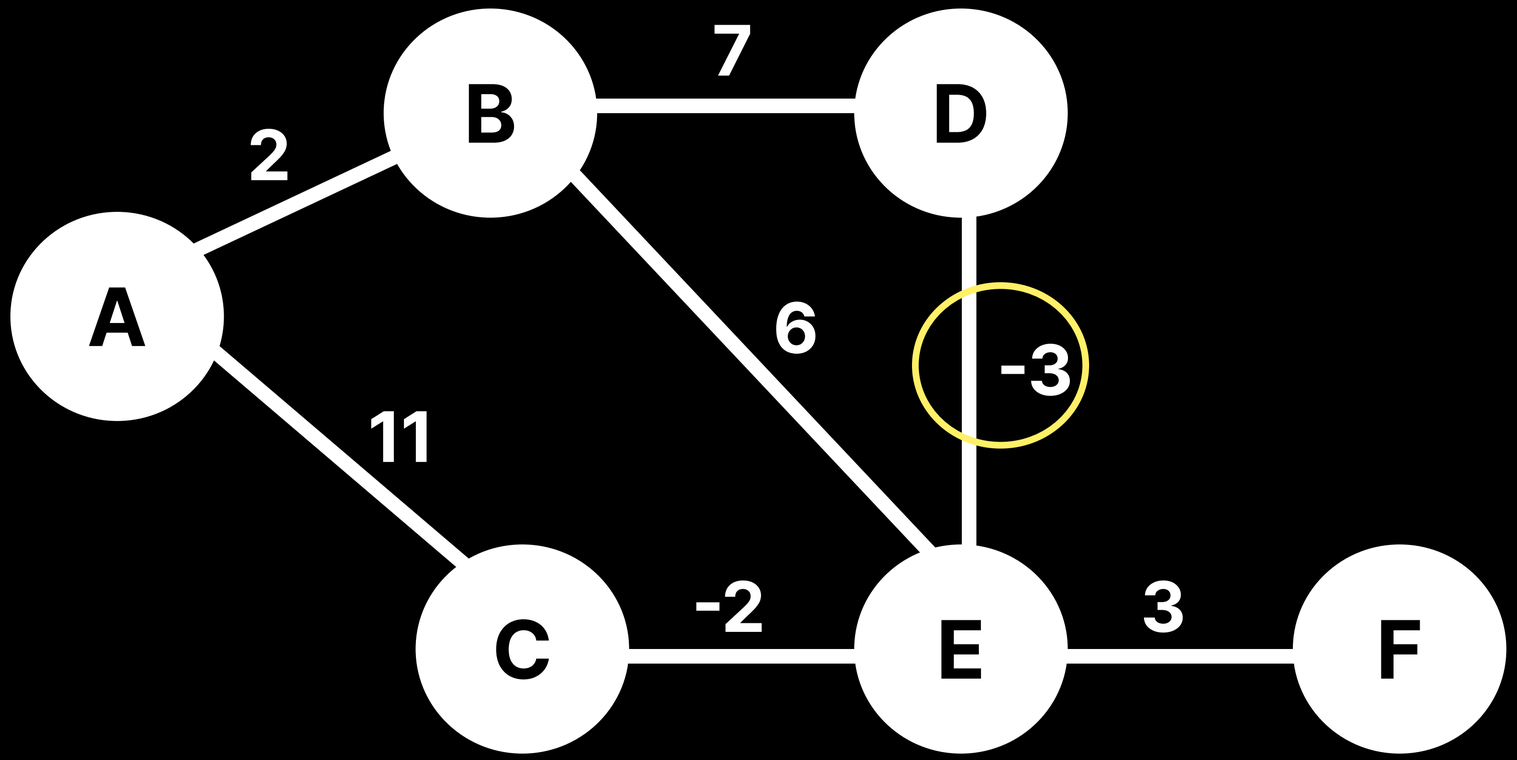
Prims



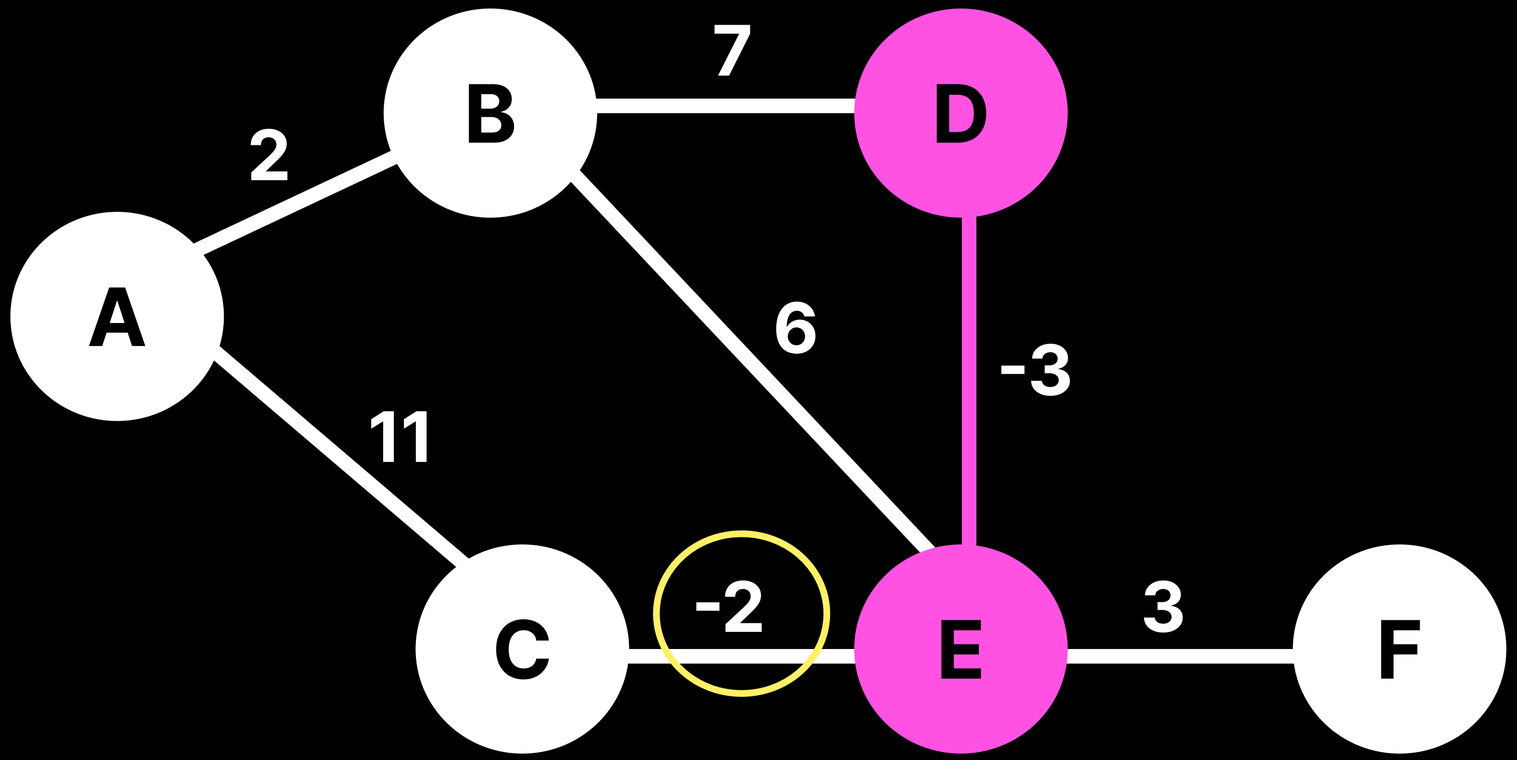
Prims



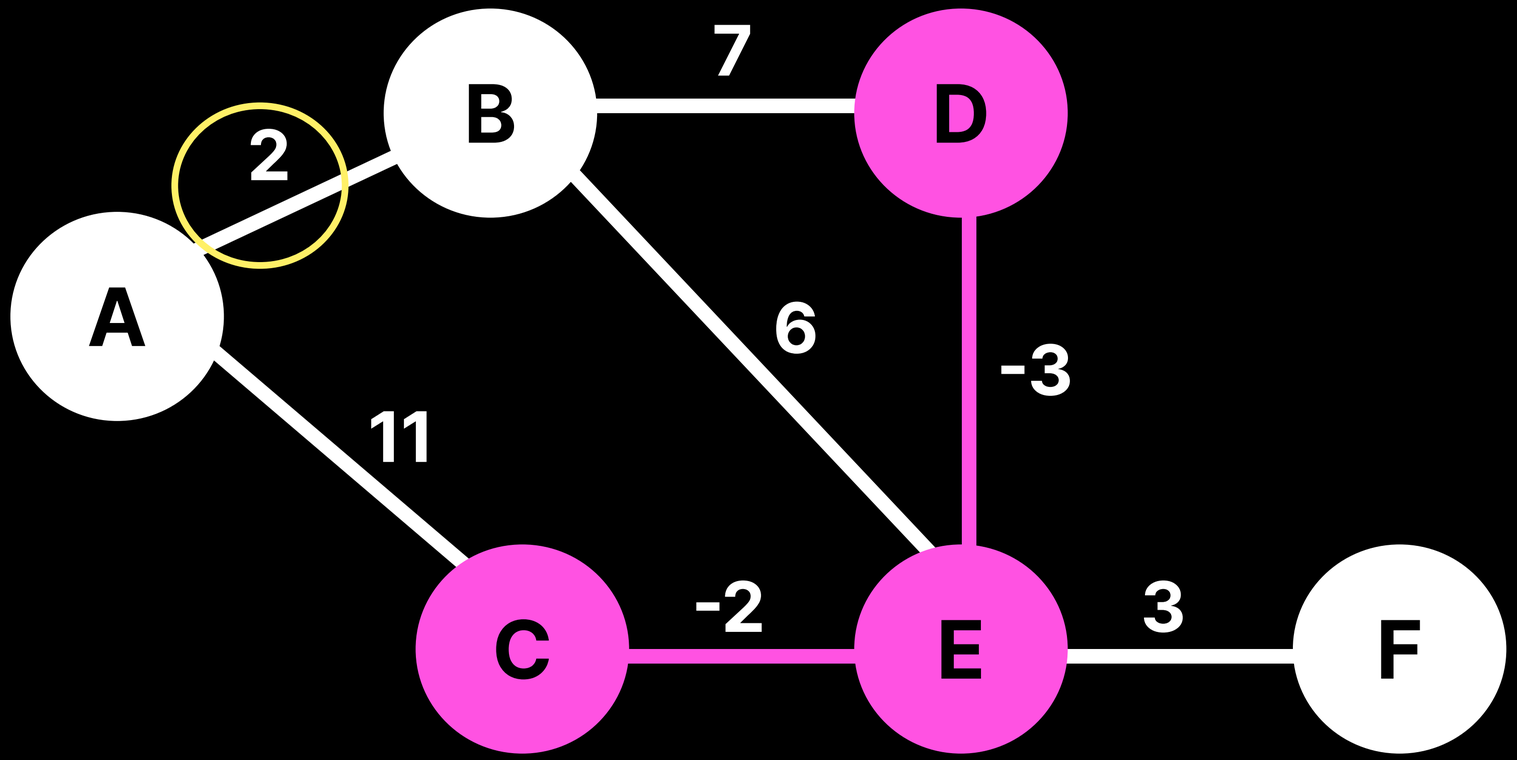
Kruskals



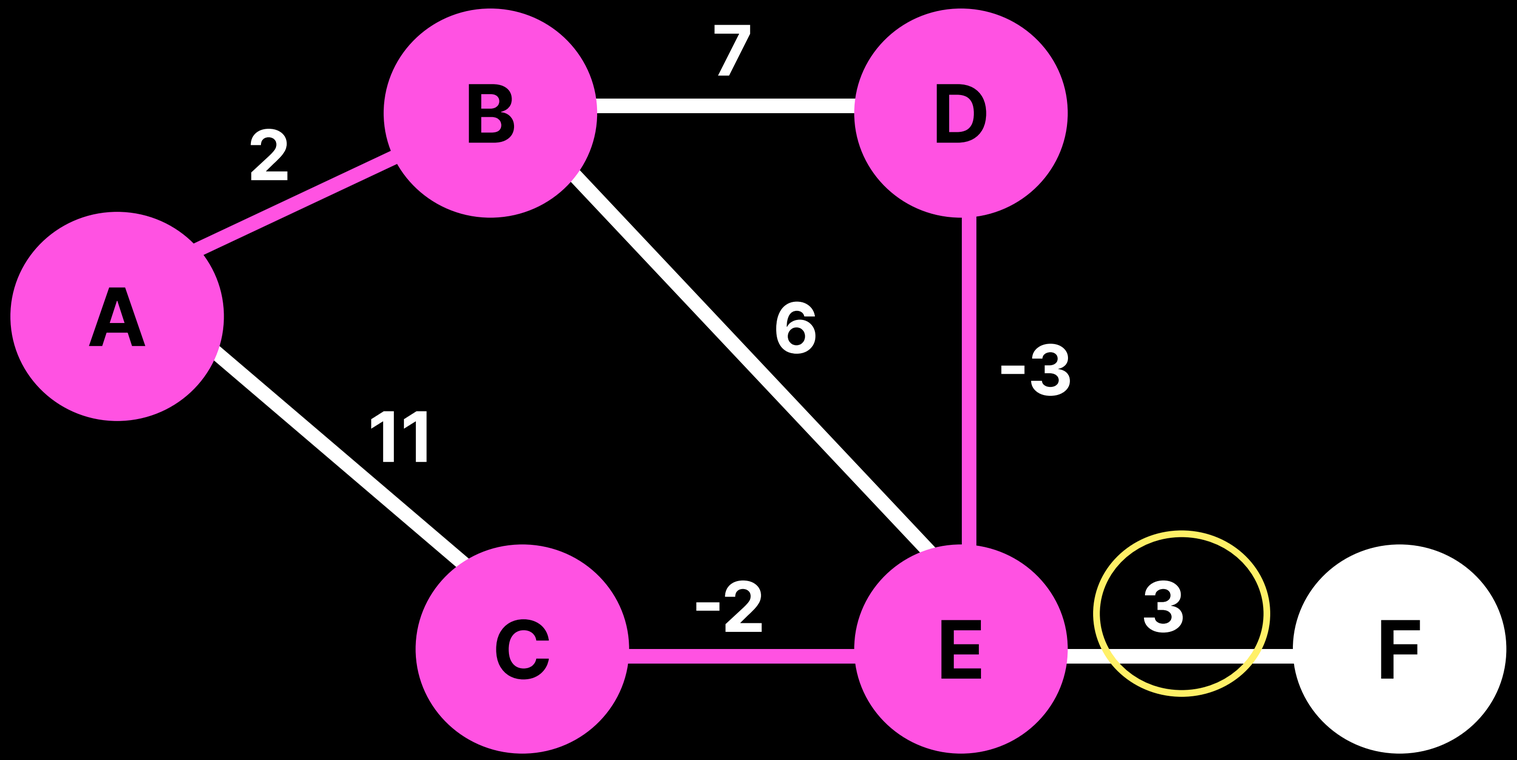
Kruskals



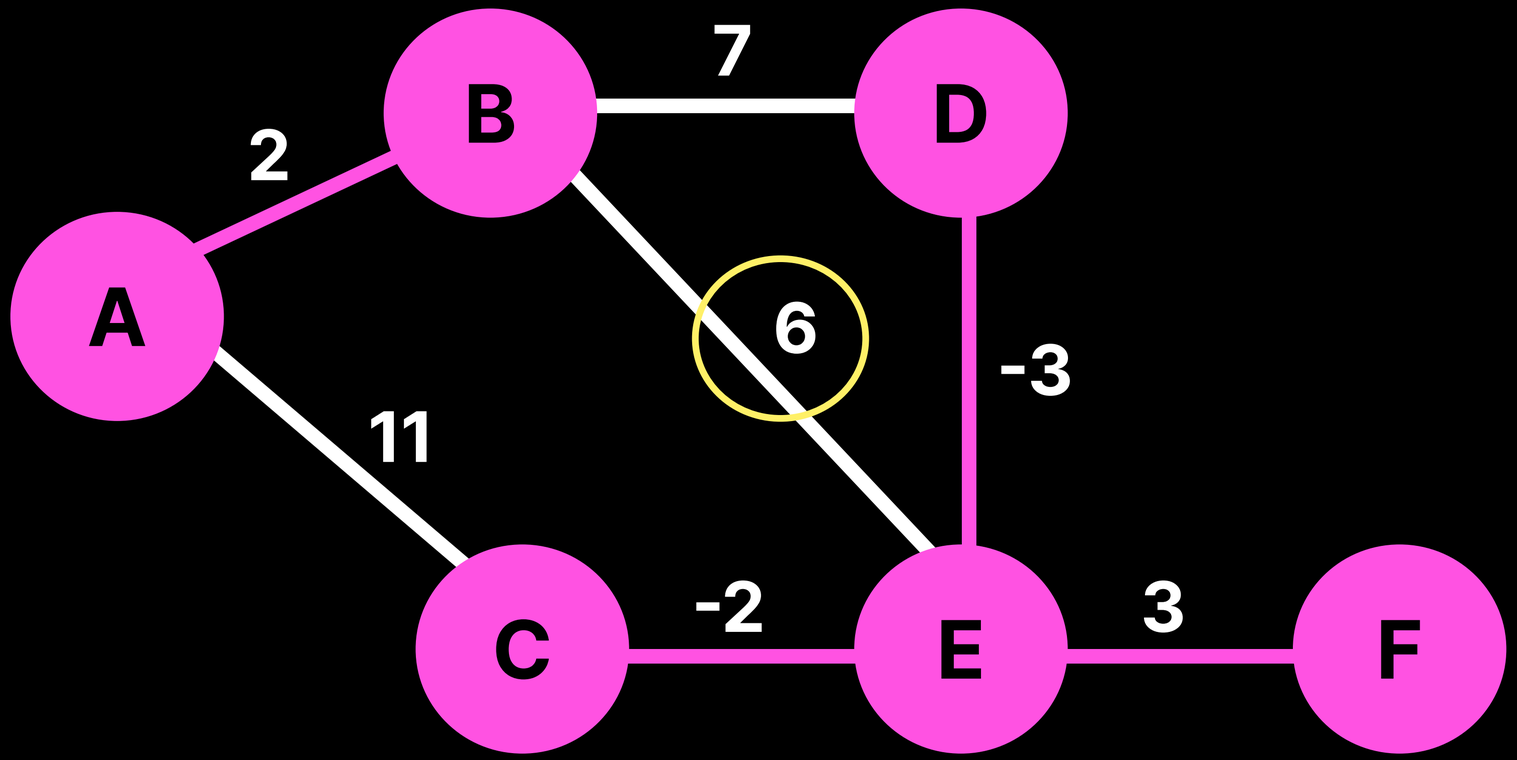
Kruskals



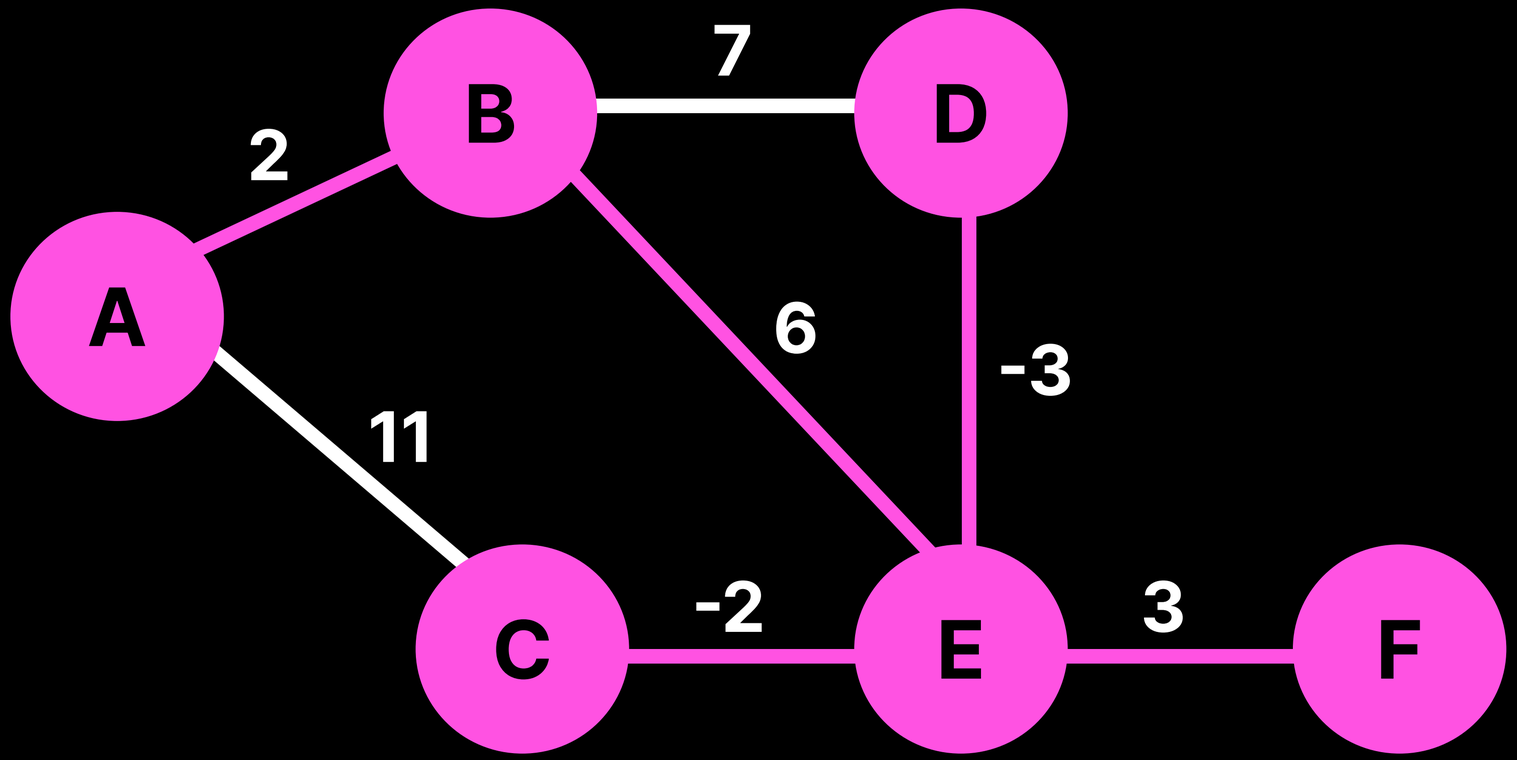
Kruskals



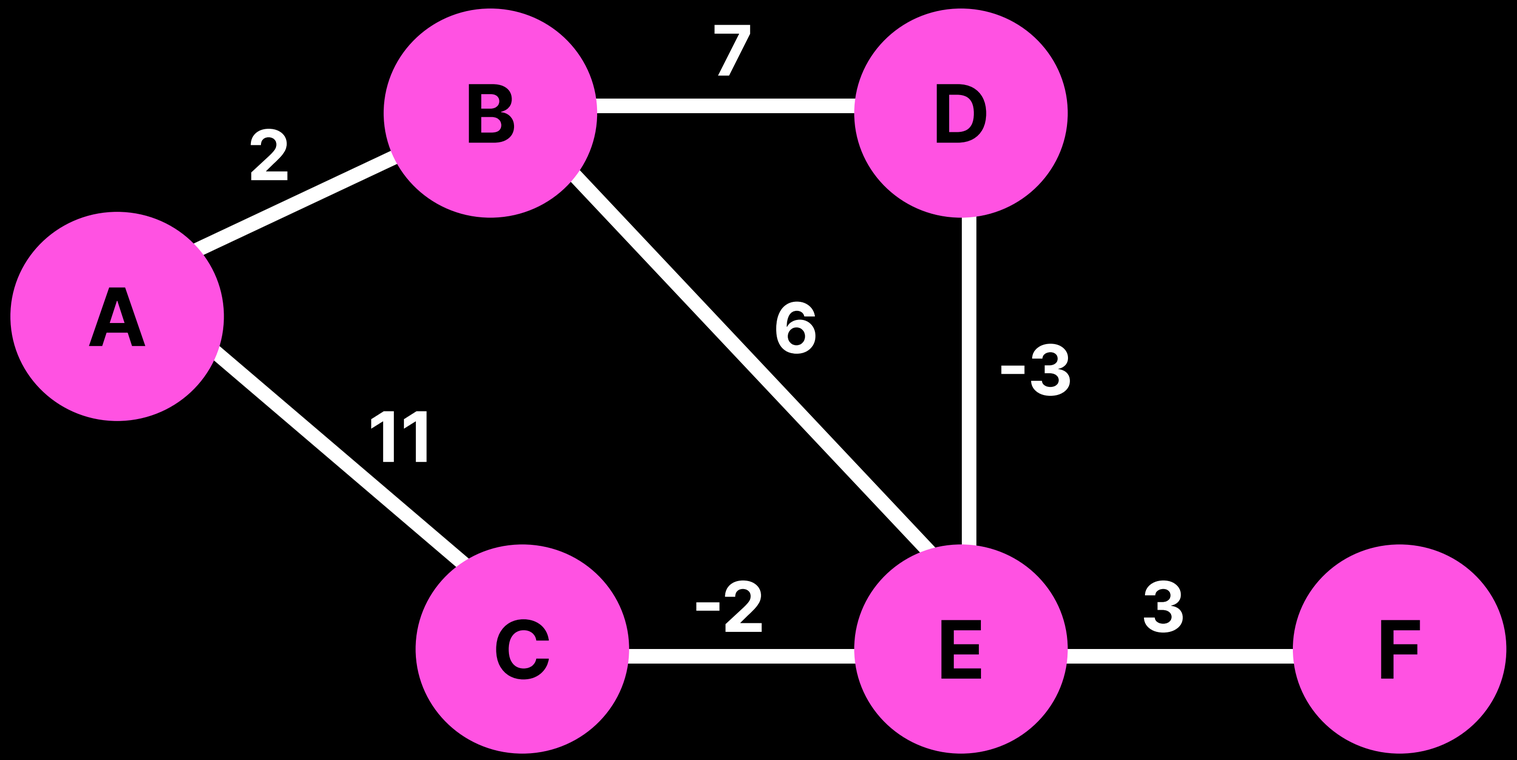
Kruskals



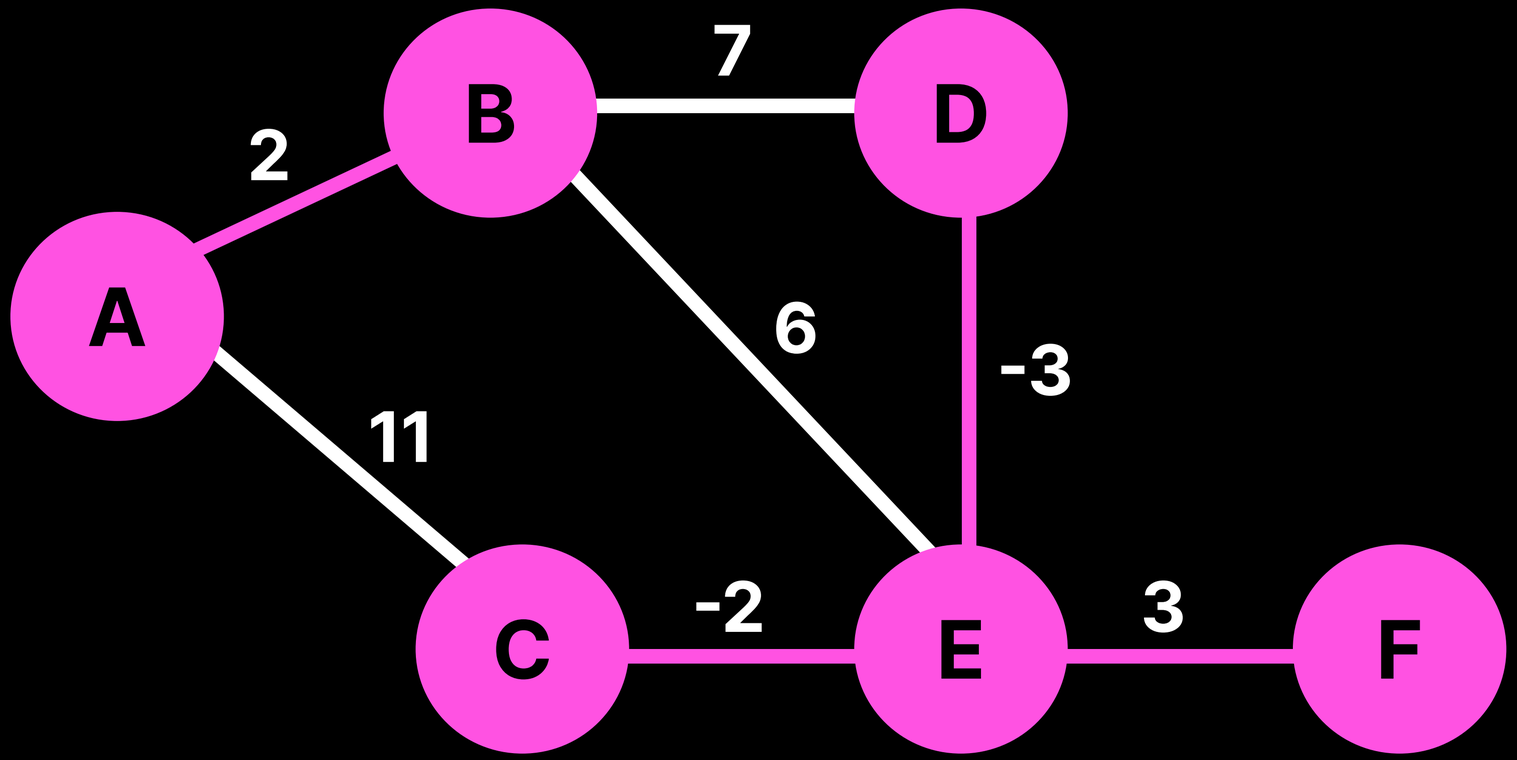
Kruskals



Borůvkas



Borůvkas



Borůvkas

