Øving 8 – Traversering av grafer

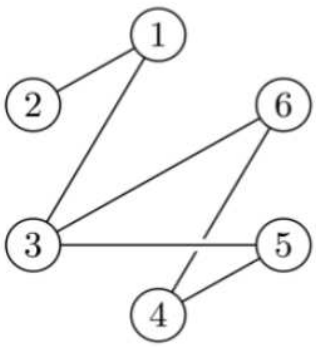
TDT4120 – Algoritmer og datastrukturer 2018

Question 1: Representasjon av grafer

Oppgaven ble godkjent! ×

Hvilke(n) påstand(er) stemmer om følgende figurer? Anta at grafen som beskrives kalles $G = (V, E)$.

a)



b)

1	→	2	3	
2	→	1		
3	→	6	1	5
4	→	6	5	
5	→	3	4	
6	→	3	4	

c)

	1	2	3	4	5	6
1		1	1			
2	1					
3	1				1	1
4					1	1
5			1	1		
6			1	1		

- [x] Graden til node 3 er 3
- [x] *b)* er en naboliste-representasjon for figuren i *a)*
- [x] *c)* er en nabomatise for figuren i *a)*
- [x] $|v| = 6$ og $|E| = 6$
- [x] Matisen i *c)* er symmertisk fordi G er urettet
- [x] *a)* er en urettet graf

Question 2: Representasjon av grafer

Oppgaven ble godkjent! ×

Gitt at $|V| = |E|^2$, hvordan kan grafen G lagres mest plasseffektivt i denne situasjonen?

- ( ) Nabolister og nabomatrise er like plasseffektivt
- (•) Nabolister
- ( ) En lenket liste
- ( ) En nabomatrise
- ( ) Nabolister, nabomatrise og lenket liste er like plasseffektivt

Man kan se dette fordi mange av rutene i nabomatrisen vil være null når |V| < |E|^2. Hvis derimot |V|^2 = |E|, ville nabomatrise og naboliste vært like effektivt asymptotisk (men siden nabolister ofte bruker pekere ville nabolister da ta konstant mer plass i det tilfellet. Se Cormen et al. side 589).

Question 3: Representasjon av grafer

Oppgaven ble godkjent!

Anta at du har to noder, $u, v \in V$. Hvor lang tid vil det ta å sjekke om det finnes en kant, $e \in E$, som går fra u til v gitt at grafen G er representert ved hjelp en nabomatrise? Anta at du ikke vet noe om hvor mange kanter eller hvor mange noder det finnes i G.

- O($|E|$)
- O($|E| + |V|$)
- O(1)
- O($|V|$)

Question 4: Representasjon av grafer

Oppgaven ble godkjent!

Hvor lang til vil tilsvarende oppslag ta hvis G er en naboliste-representasjon og det går minst én kant ut fra hver node?

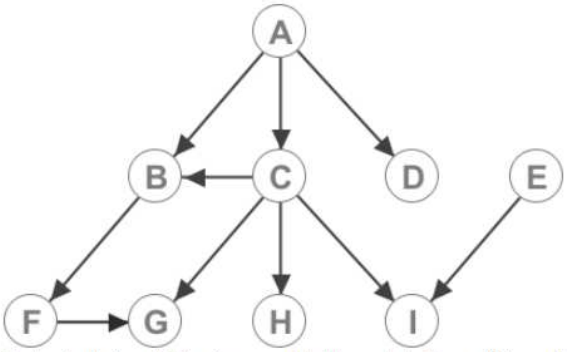
- O($|V|$)
- O($|E| + |V|$)
- O(1)
- O($|E|$)

Det verste tilfellet er når grafen er komplett. Da har hver node |V| naboer. Siden naboene til hver node er representert som en lenket liste, må man gjøre et lineærsøk i denne listen. Det tar O(|V|) tid.

Question 5: Bredde-først-søk

Oppgaven ble godkjent!

BFS blir kjørt med påfølgende graf med A som rotnode. I hvilken rekkefølge blir de fire første nodene farget svart?



Anta at alle konflikter løses ved hjelp av leksikografisk ordning (ved eventuelle konflikter velges den noden med bokstav tidligst i alfabetet, altså A før B, B før C osv.)

- B, C, D, E
- A, B, D, C
- A, C, D, B
- A, B, F, G
- A, B, C, D
- A, C, B, D
- B, C, D, F

Se animasjon.

Question 6: Bredde-først–søk

Oppgaven ble godkjent!

Hvilke(n) påstand(er) stemmer om BFS?

- [ ] Implementeres vanligvis rekursivt
- [ ] Implementeres vanligvis med en stakk
- [ ] Implementeres vanligvis med en heap
- [ ] Ingen av påstandene stemmer
- [x] Implementeres vanligvis med en kø

Question 7: Bredde-først-søk

Oppgaven ble godkjent!

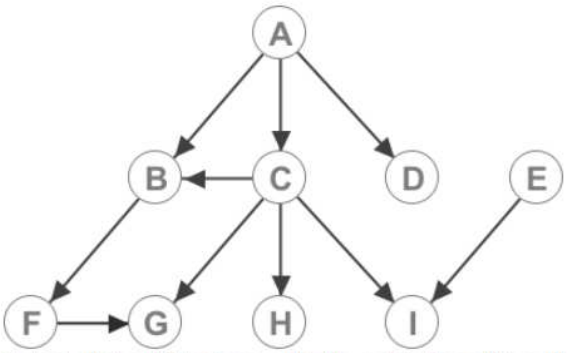
For hvilket av alternativene under er vi garantert at bredde-først-søk finner korteste vei i en vilkårlig sammenhengende graf?

- Alle kantene har lik vekt
- Alle kantene har lik ikke-negativ vekt
- Ingen negative kanter

Question 8: Dybde-først-søk

Oppgaven ble godkjent!

DFS blir kjørt med påfølgende graf med A som rotnode. I hvilken rekkefølge blir de fire første nodene farget svart?



Anta at alle konflikter løses ved hjelp av leksikografisk ordning (ved eventuelle konflikter velges den noden med bokstav tidligst i alfabetet, altså A før B, B før C osv.)

- (•) G, F, B, H
- ( ) G, H, B, F
- ( ) A, B, C, D
- ( ) A, B, F, G
- ( ) A, C, B, D
- ( ) A, B, D, C
- ( ) B, C, D, E
- ( ) B, C, D, F
- ( ) A, C, D, B

Se animasjon.

Question 9: Dybde-først-søk

Oppgaven ble godkjent!

Hvilke(n) påstand(er) stemmer om DFS?

- [ ] Det er svært unaturlig å implementere DFS med rekursjon
- [x] Det er svært unaturlig å implementere DFS med heap
- [ ] Det er naturlig å implementere DFS med kø, stakk, rekursjon og heap
- [x] Det er svært unaturlig å implementere DFS med kø
- [ ] Det er svært unaturlig å implementere DFS med stakk

DFS implementeres vanligvis med en stakk eller rekursjon.

Question 10: Dypde-først-søk

Oppgaven ble godkjent!

Et dypde-først-søk kan brukes til å klassifisere kantene i en graf. Hvilken av følgende kanttyper betegner en kant som går fra en forgjenger (ancestor) til en etterkommer (descendant)?

- Cross edge
- Back edge
- Tree edge

Question 11: Dybde-først-søk

Oppgaven ble godkjent!

Hva slags type kant kan vi ha kommet til når vi kommer til en node som allerede er farget svart i et dypde-først-søk?

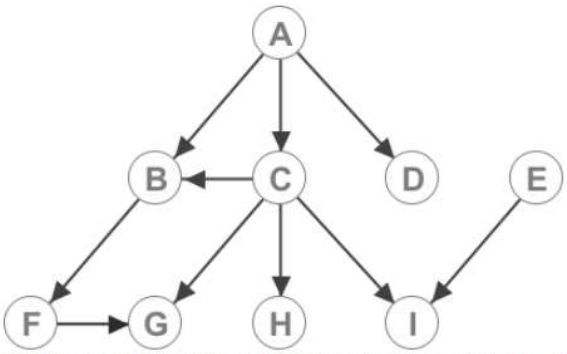
- [x] Cross edge
- [ ] Tree egde
- [x] Forward edge
- [ ] Back edge

Se Cormen et al. side 609 for klassifisering av kantene.

Question 12: Topologisk sortering

Oppgaven ble godkjent!

Hvilke(n) av følgende alternativ er en gyldig topologisk sortering?



Hint: En graf kan ha flere mulige topologiske sorteringer. I stedet for å lage en topologisk sortering av grafen, bør du heller sjekke hvilke av alternativene som overholder kravene til en topologisk sortering

- [x] $A, E, D, C, I, H, B, F, G$
- [ ] $A, B, F, G, C, H, I, D, E$
- [ ] $E, A, I, D, C, H, B, F, G$
- [ ] $E, A, D, C, I, H, B, G, F$

I denne oppgaven gjaldt det å finne noder som ikke tilfredstiller kravene til en topologisk sortering. Man kan f.eks utelukke det nederste alternativet her fordi G ikke kan komme før F i en gyldig topologisk sortering.

Question 13: Topologisk sortering

Oppgaven ble godkjent!

Du ønsker å lage en topologisk sortering av en graf $G = (V, E)$. Hvilke av følgende kriterier må være sanne (for grafen G) for at det skal finnes en topologisk sortering?

- (•) Den må være rettet og asyklisk (en DAG)
- ( ) Den må ha positive kantvekter
- ( ) Alle de andre alternative over må være riktige
- ( ) Alle kantvektene må være like

Question 14: Tidligere eksamensoppgave

Oppgaven ble godkjent!

Du prøver å implementere BFS for urettede grafer, men på grunn av en kodefeil, er rekkefølgen på nodene i køen din ikke lenger FIFO, men helt vilkårlig. Kan du nå være sikker på å besøke alle nodene?

- (•) Ja, dersom grafen er sammenhengende
- ( ) Nei
- ( ) Ja, for alle grafer
- ( ) Ja, dersom grafen er en skog

Dette er oppgave 6 eksamen august 2018. Se LF til denne oppgaven for utfyllende løsning.

Question 15: Best-case-kjøretid for BFS og DFS

Oppgaven ble godkjent!

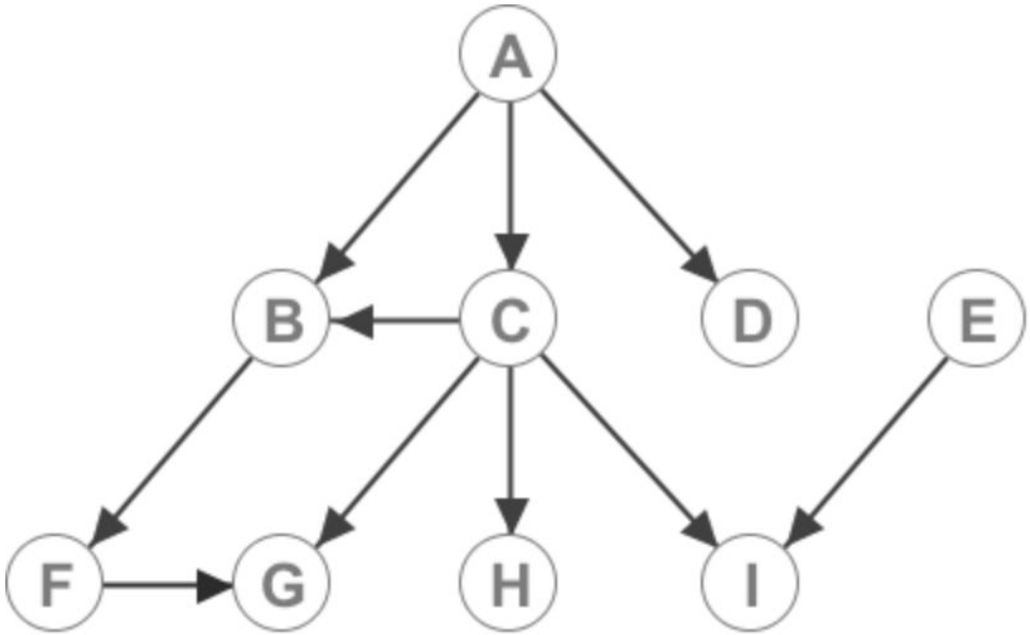
Hva er best-case-kjøretid for BFS og DFS gitt implementasjonen i læreboken?

- ( ) $O(|V| + |E|)$ for BFS og $O(1)$ for DFS
- ( ) $O(|V| + |E|)$ for begge
- (•) $O(1)$ for BFS og $O(|V| + |E|)$ for DFS
- ( ) $O(1)$ for begge

Denne oppgaven har vist seg å være gal. BFS tar ikke konstant tid i best-case gitt implementasjonen i læreboken. Siden implementasjonen i læreboken først går igjennom alle nodene for å sette `u.color`, `u.d` og `u.π`, tar algoritmen O(V) også i best-case. Hvis man ser bort fra denne initialiseringen, tar BFS O(1) tid i best-case.

Question 5: Bredde-først-søk

BFS blir kjørt med påfølgende graf med A som rotnode. I hvilken rekkefølge blir de fire første nodene farget svart?

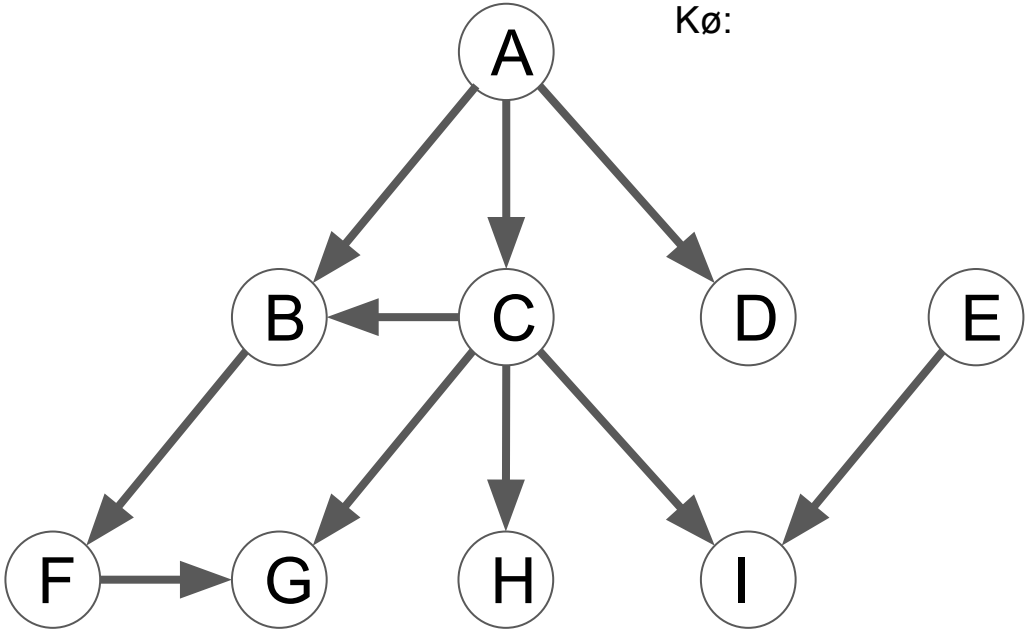


Anta at alle konflikter løses ved hjelp av leksikografisk ordning (ved eventuelle konflikter velges den noden med bokstav tidligst i alfabetet, altså A før B, B før C osv.)

Grå:

Svart:

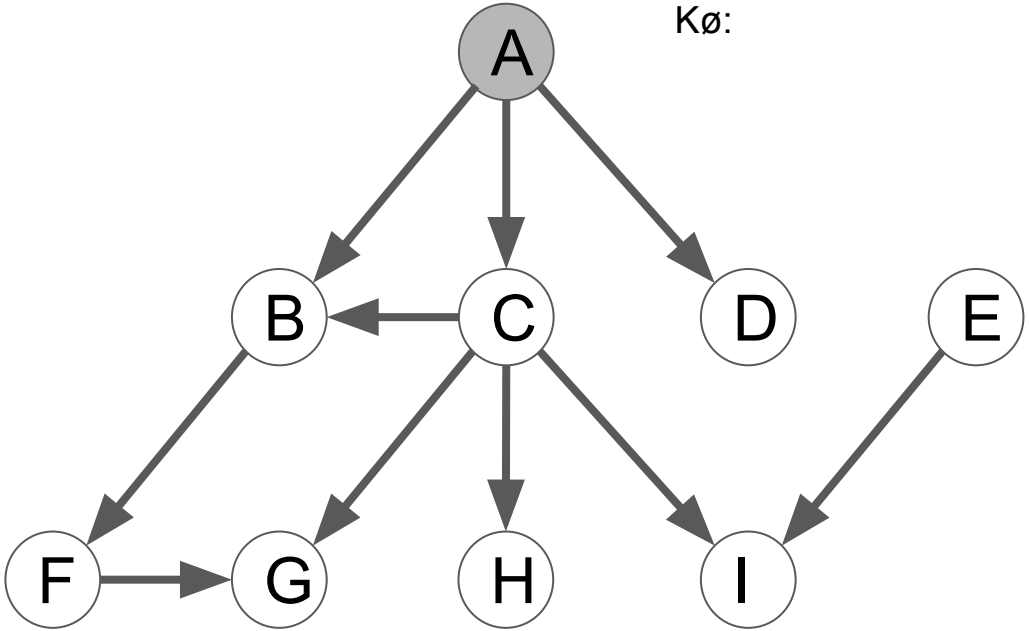
Kø:



Grå: A

Svart:

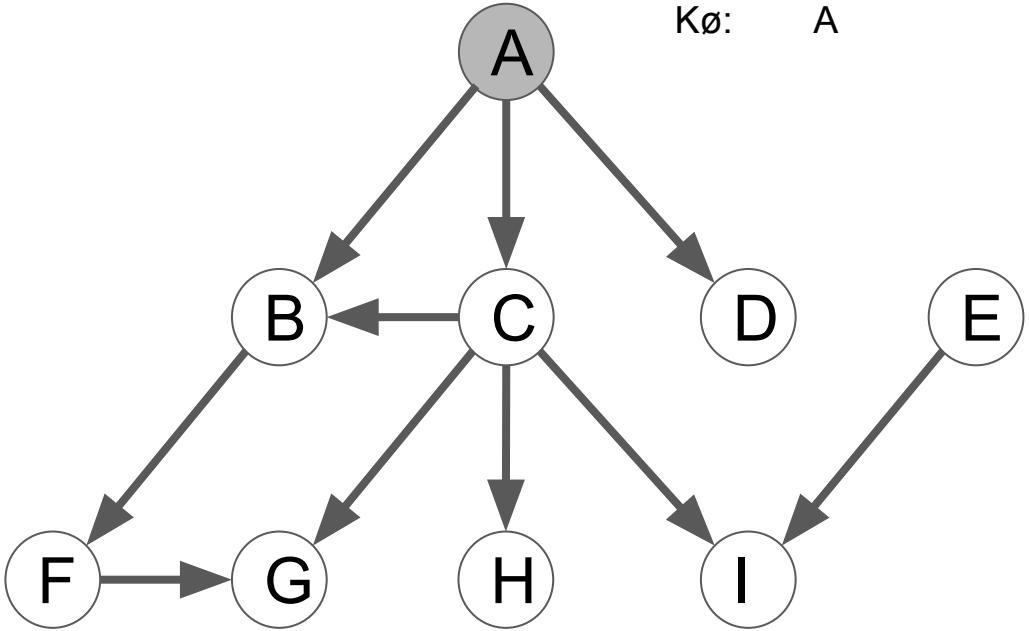
Kø:



Grå: A

Svart:

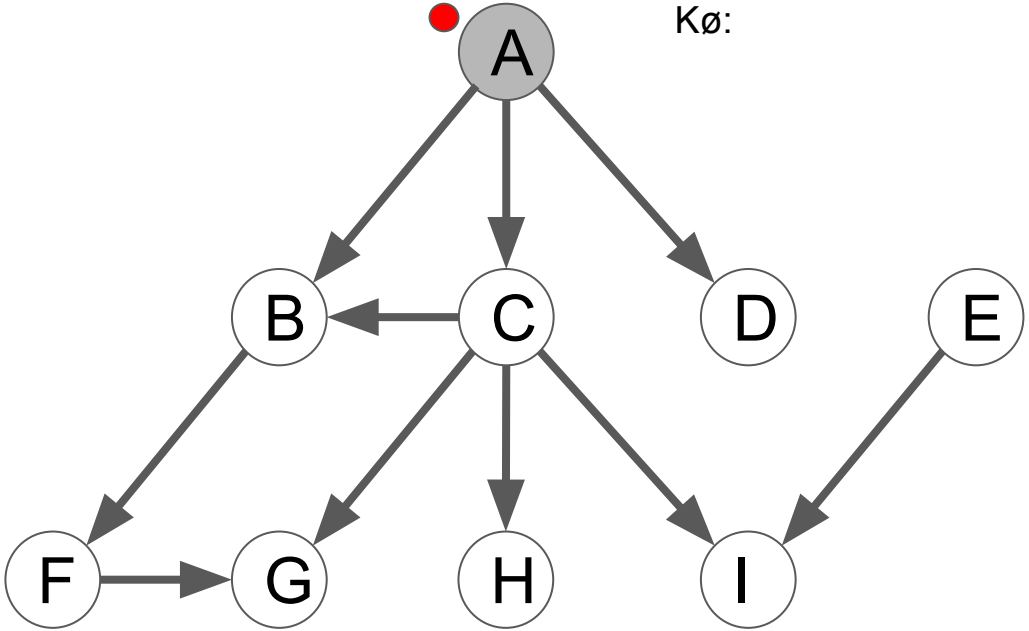
Kø: A



Grå: A

Svart:

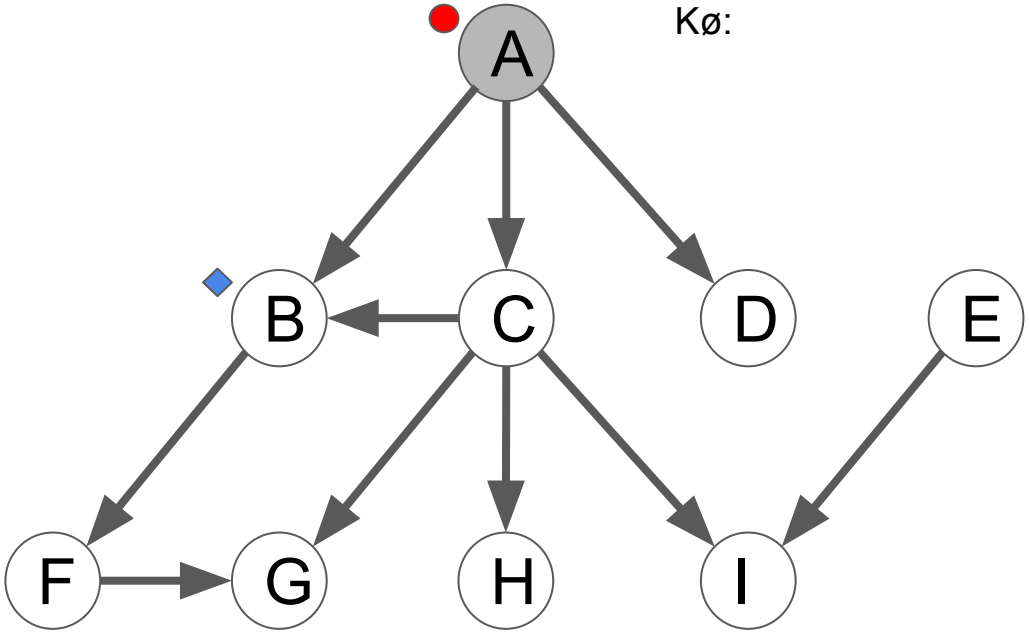
Kø:



Grå: A

Svart:

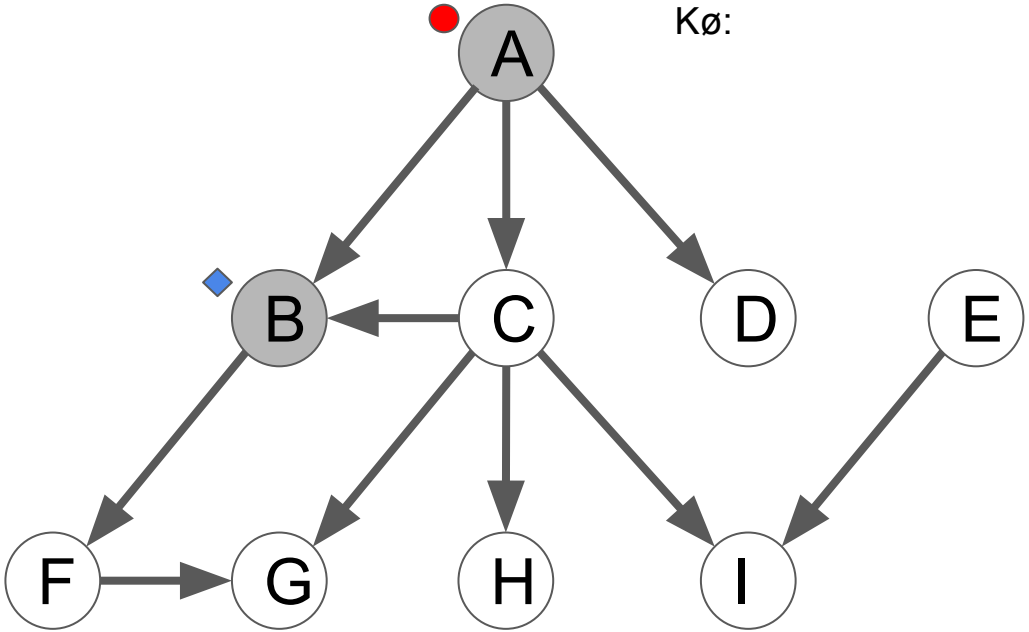
Kø:



Grå: A B

Svart:

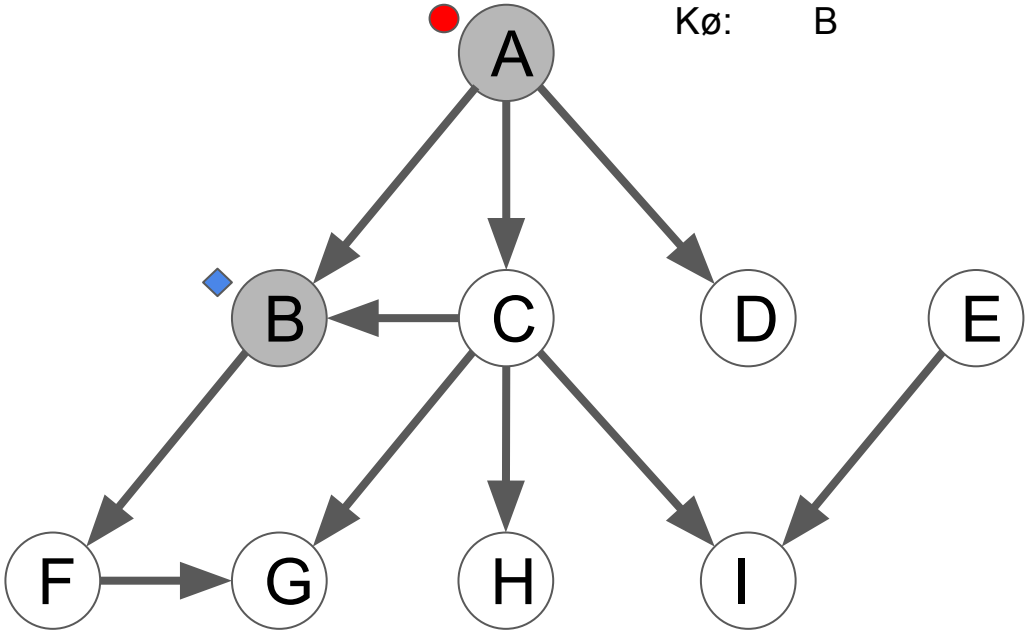
Kø:



Grå: A B

Svart:

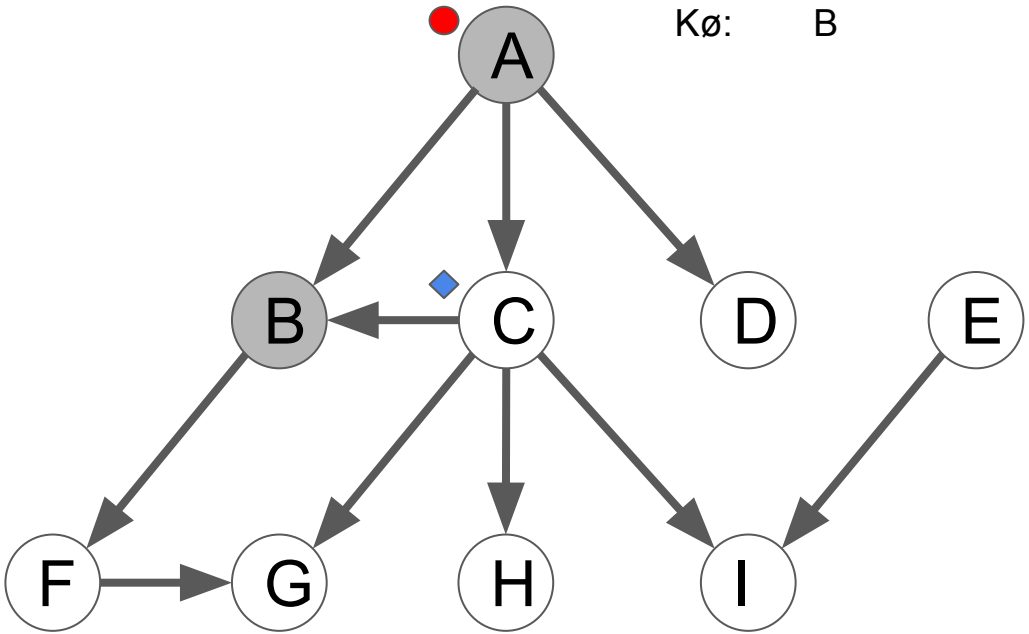
Kø: B



Grå: A B

Svart:

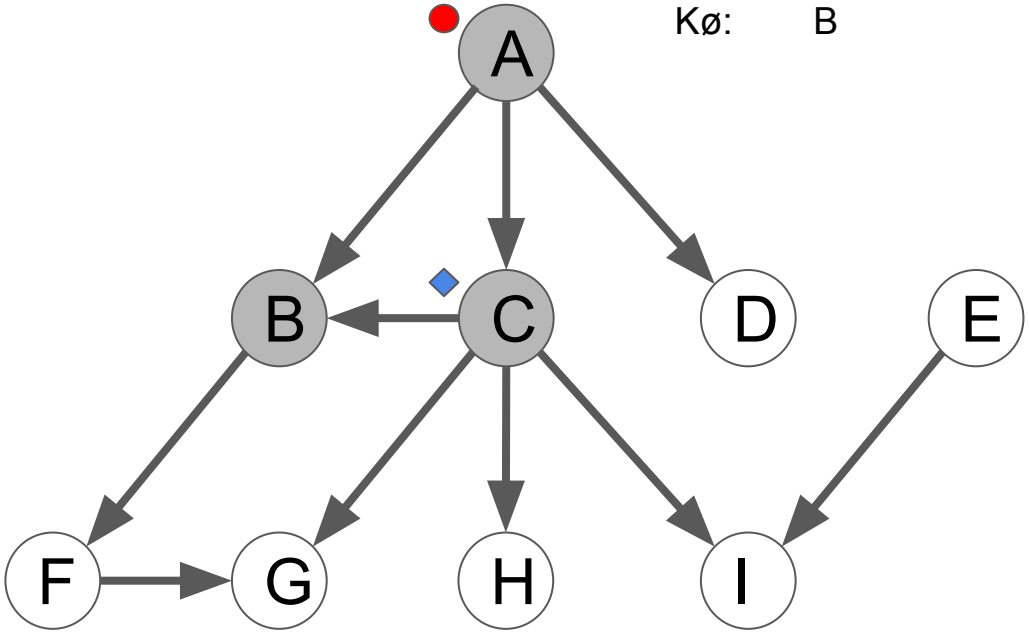
Kø: B



Grå: A B C

Svart:

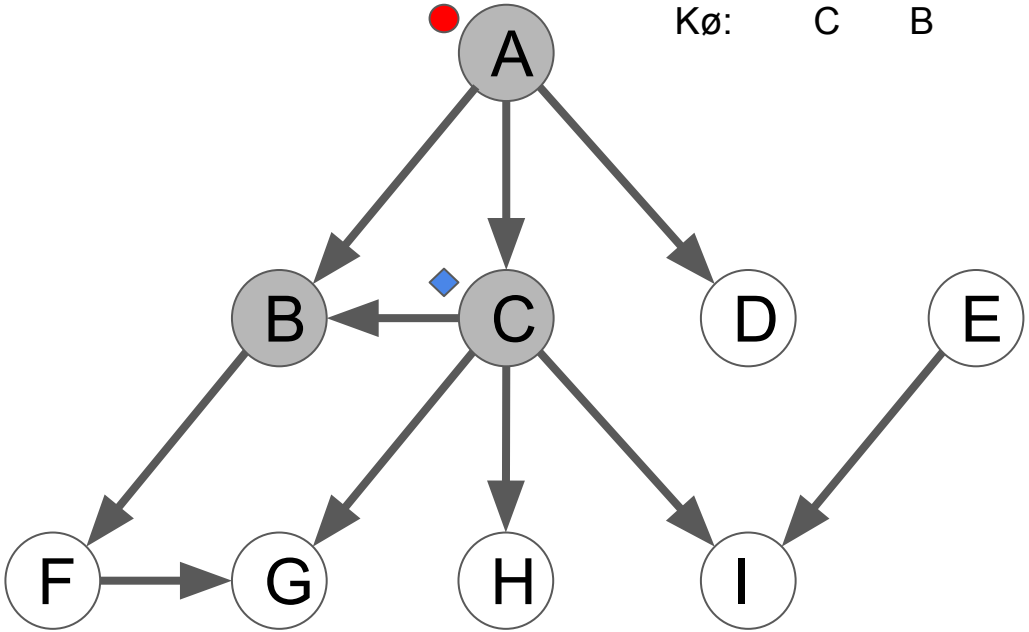
Kø: B



Grå: A B C

Svart:

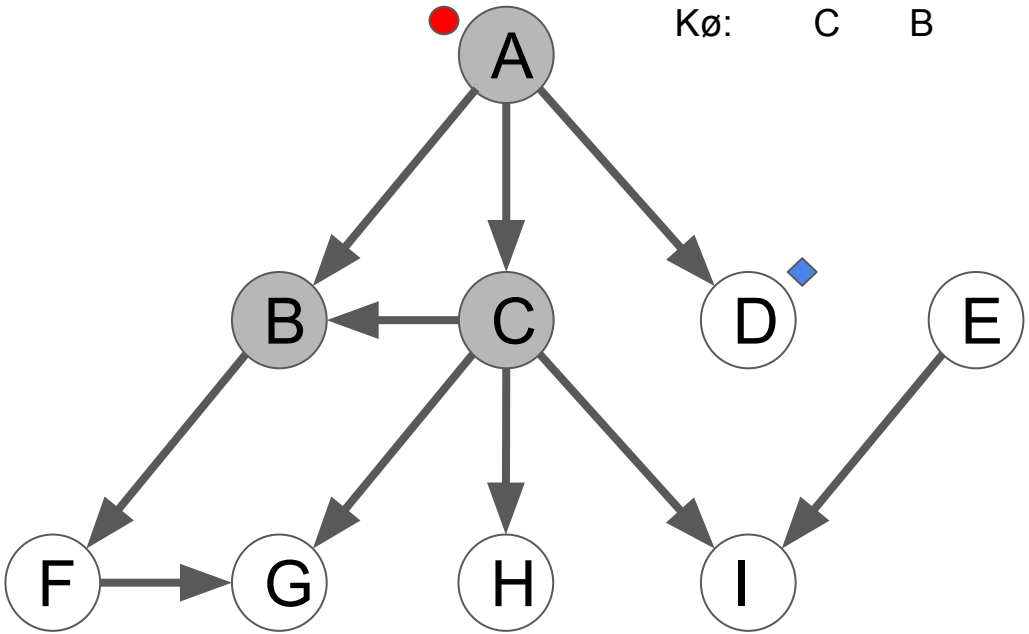
Kø: C B



Grå: A B C

Svart:

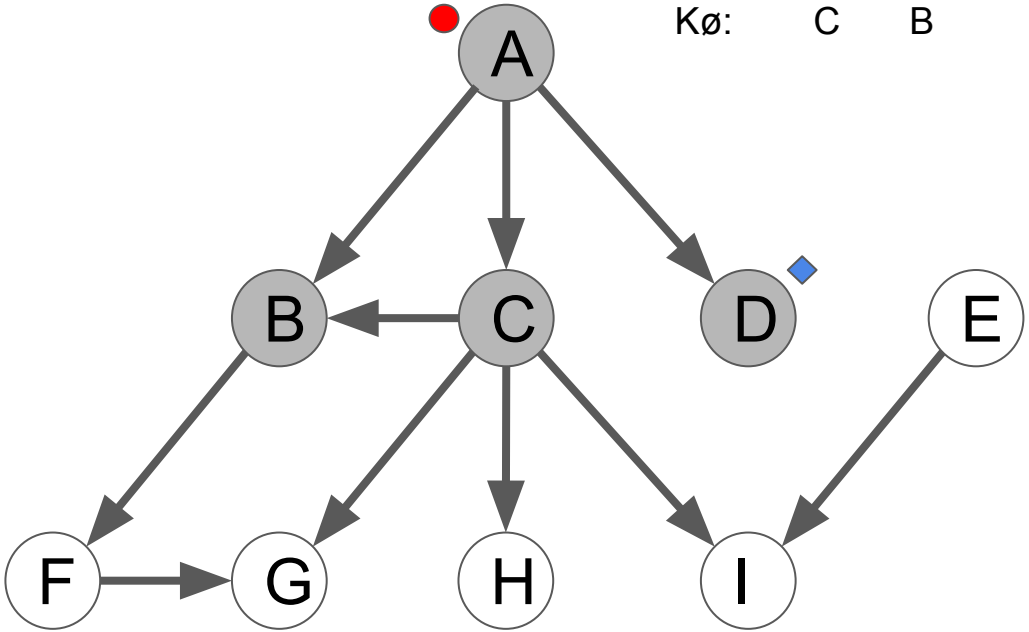
Kø: C B



Grå: A B C D

Svart:

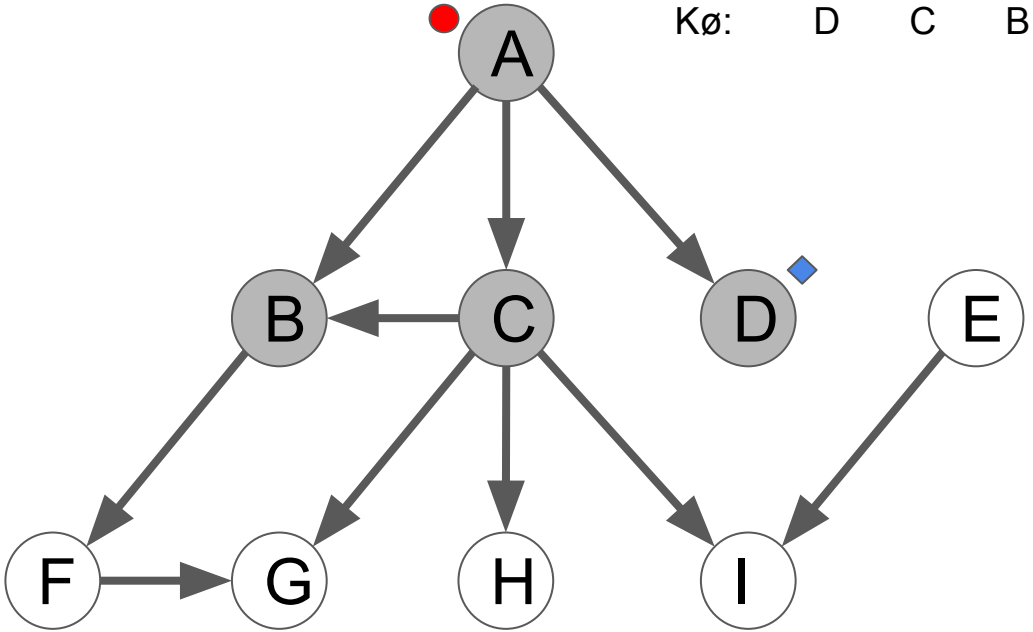
Kø: C B



Grå: A B C D

Svart:

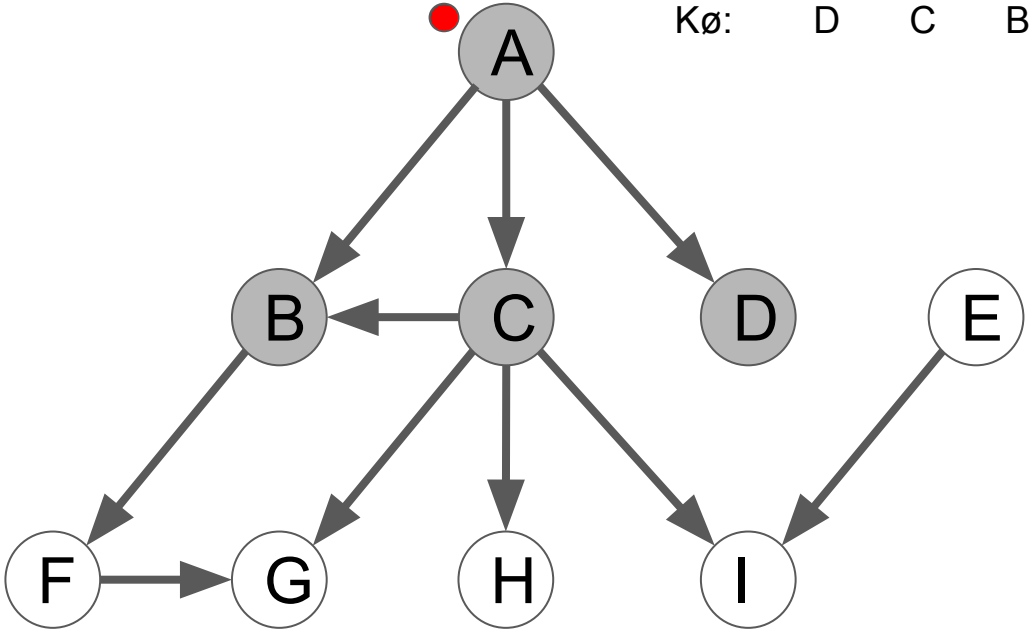
Kø: D C B



Grå: A B C D

Svart:

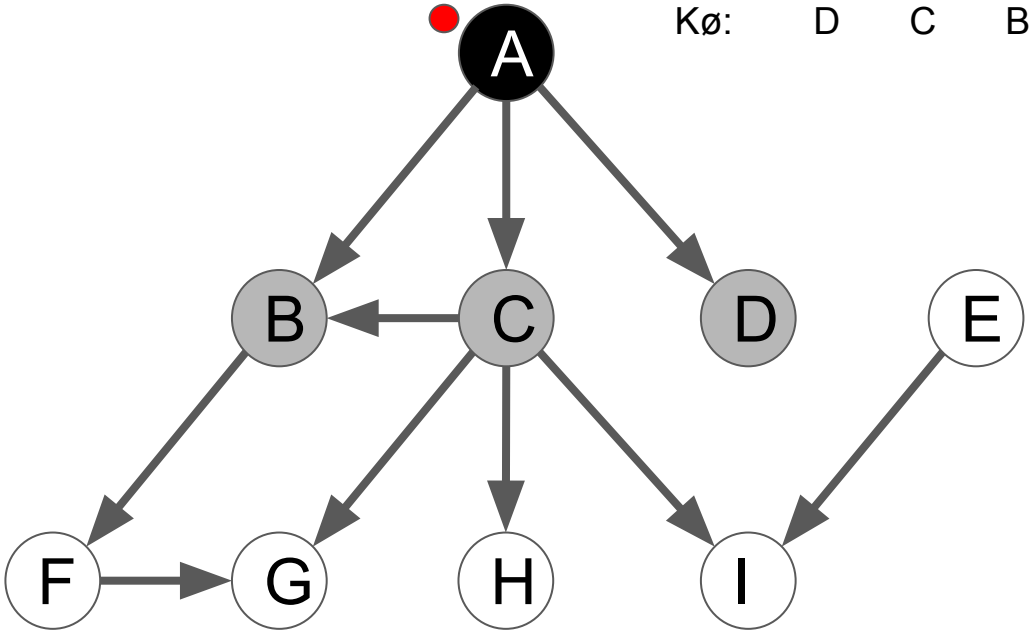
Kø: D C B



Grå: A B C D

Svart: A

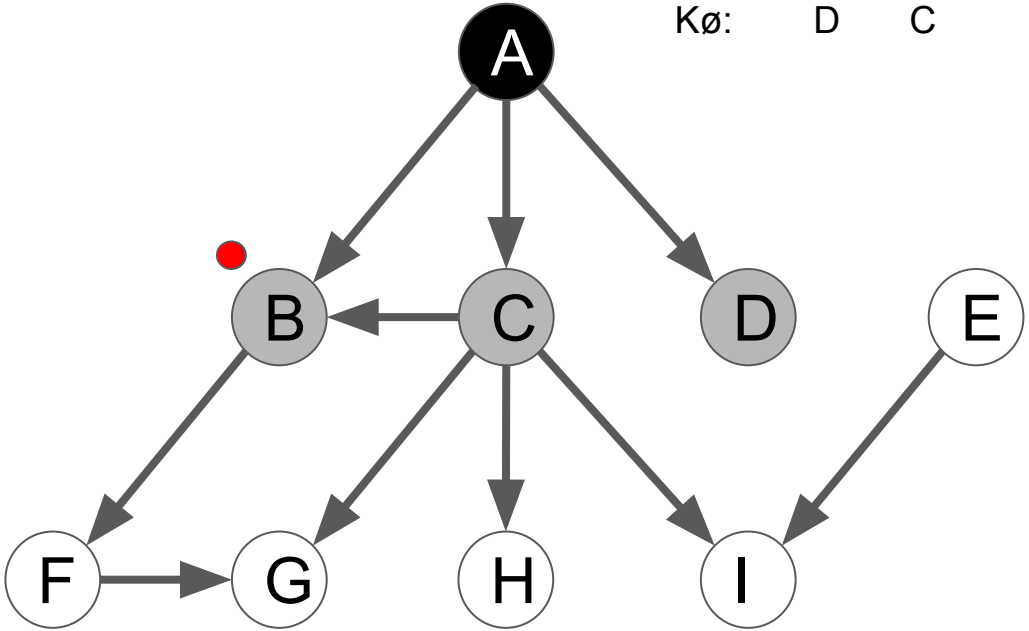
Kø: D C B



Grå: A B C D

Svart: A

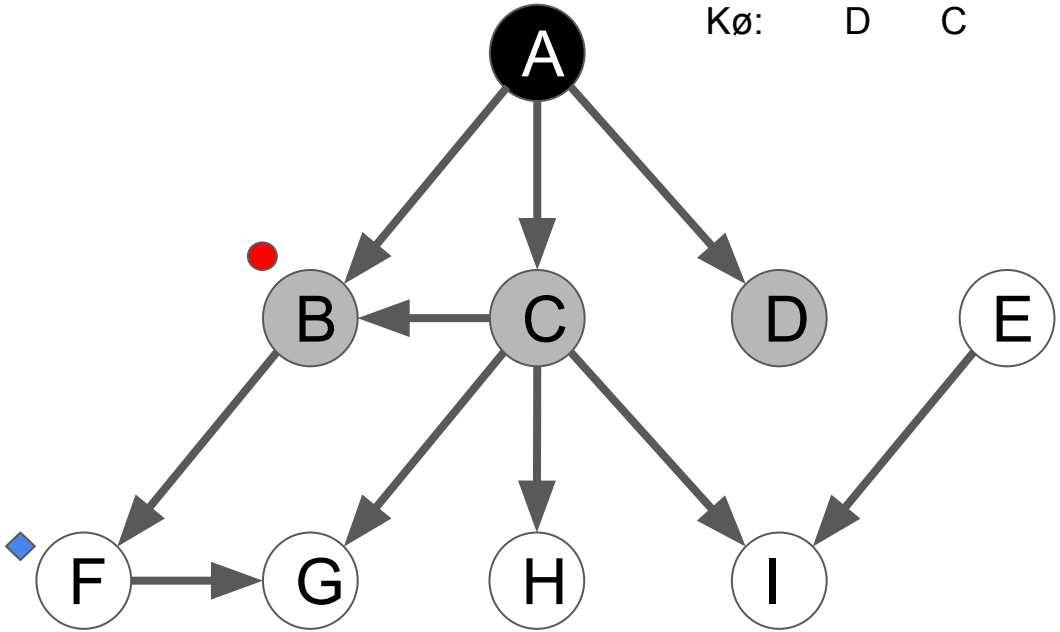
Kø: D C



Grå: A B C D

Svart: A

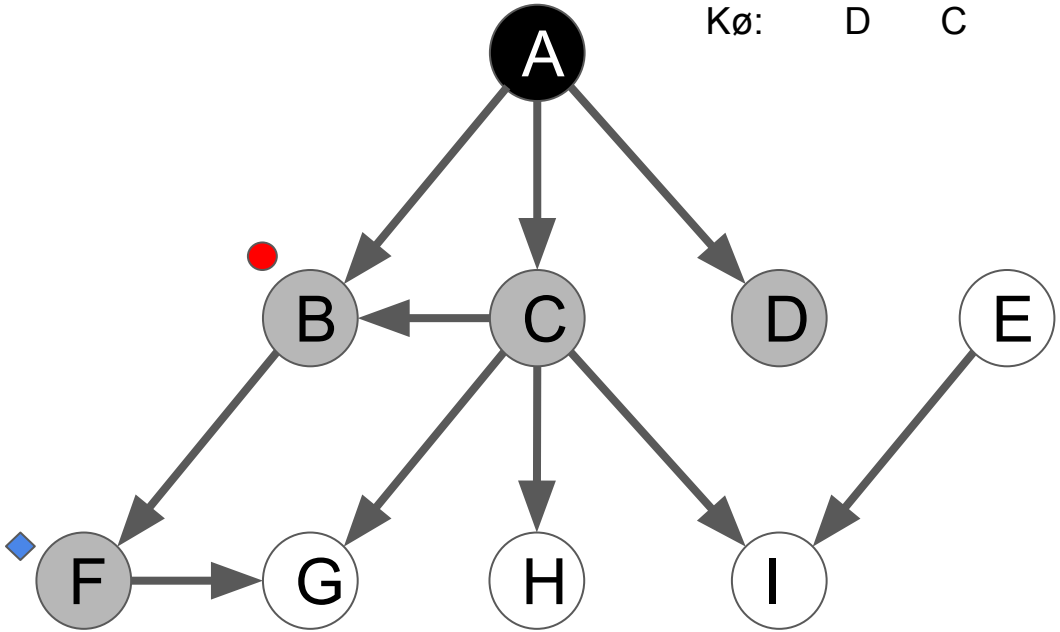
Kø: D C



Grå: A B C D F

Svart: A

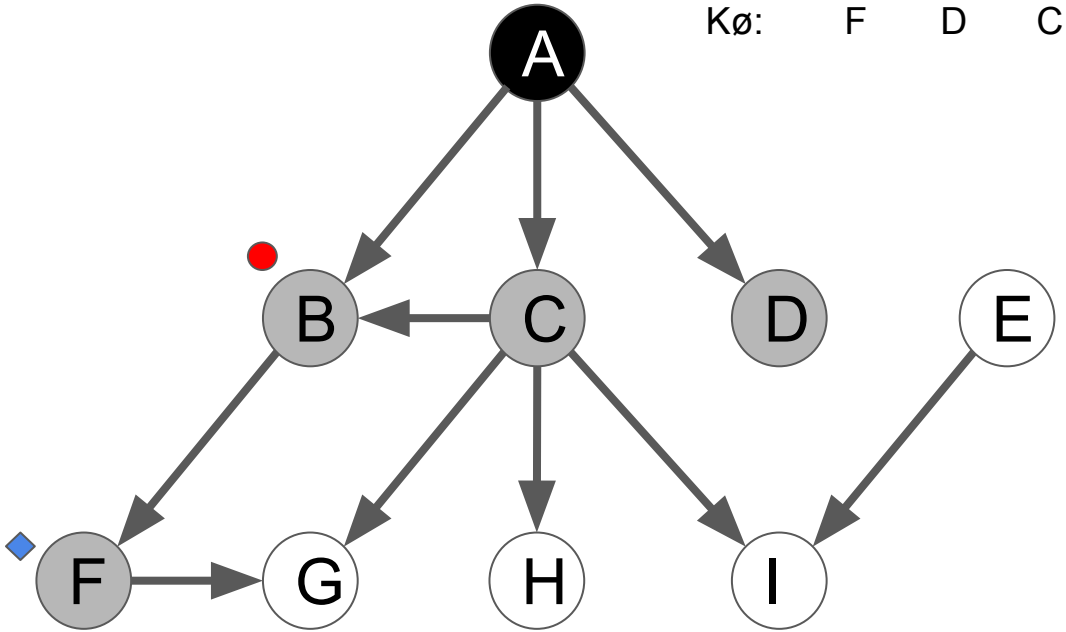
Kø: D C



Grå: A B C D F

Svart: A

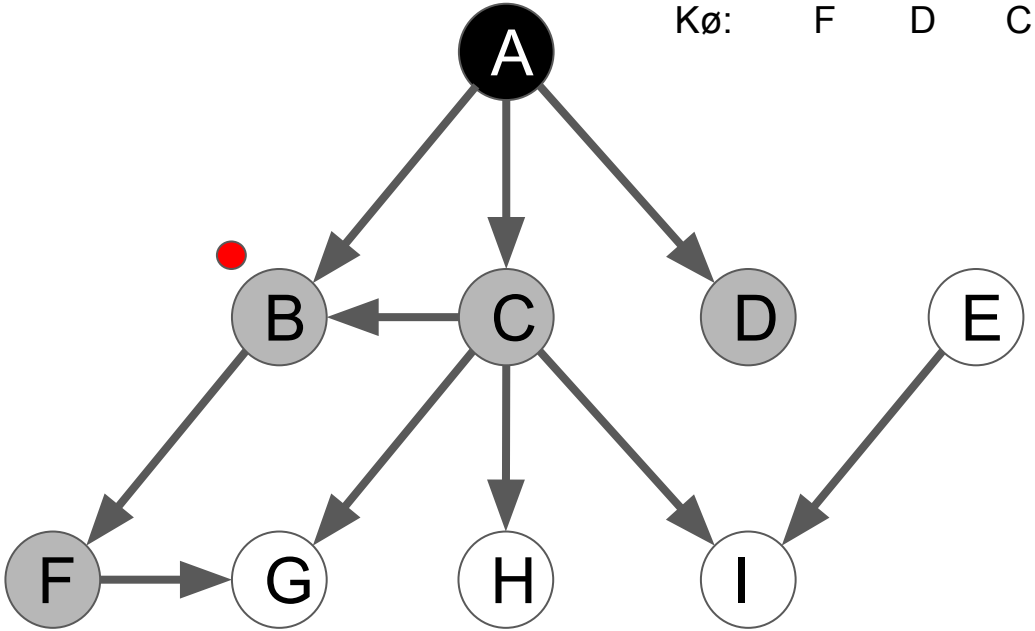
Kø: F D C



Grå: A B C D F

Svart: A

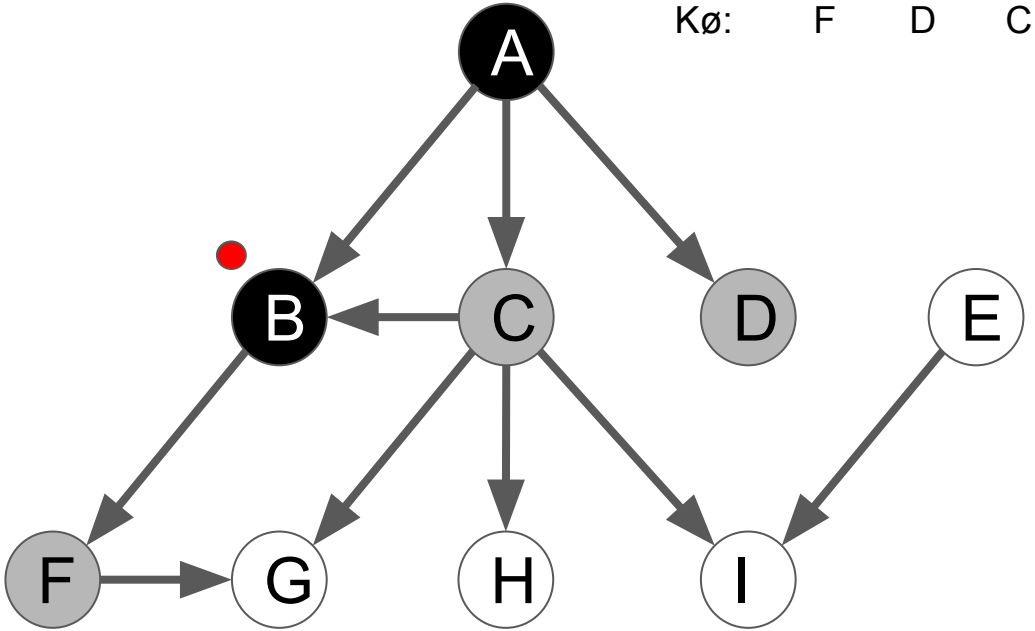
Kø: F D C



Grå: A B C D F

Svart: A B

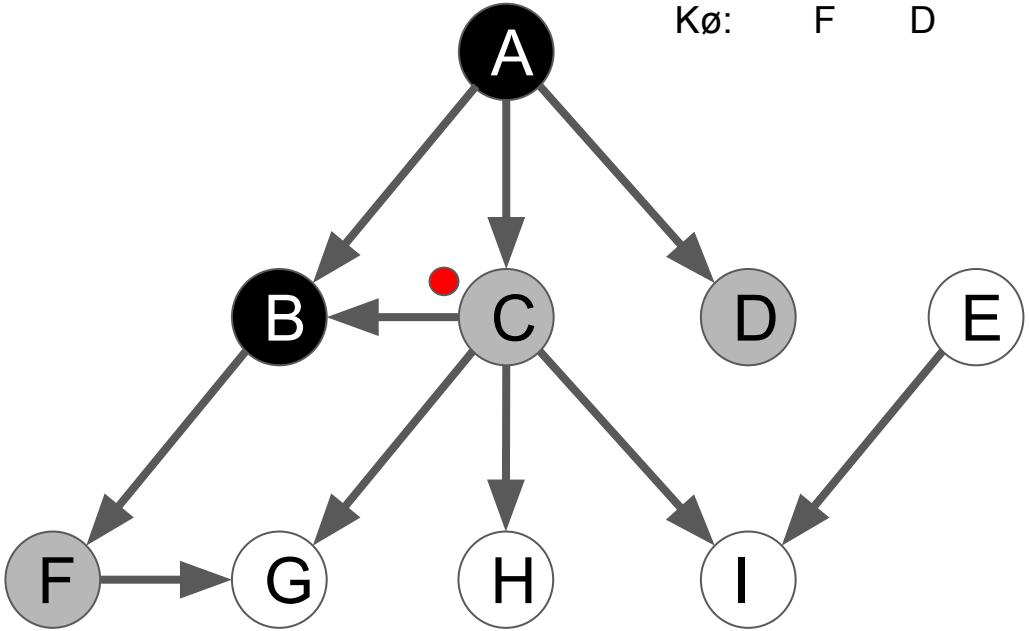
Kø: F D C



Grå: A B C D F

Svart: A B

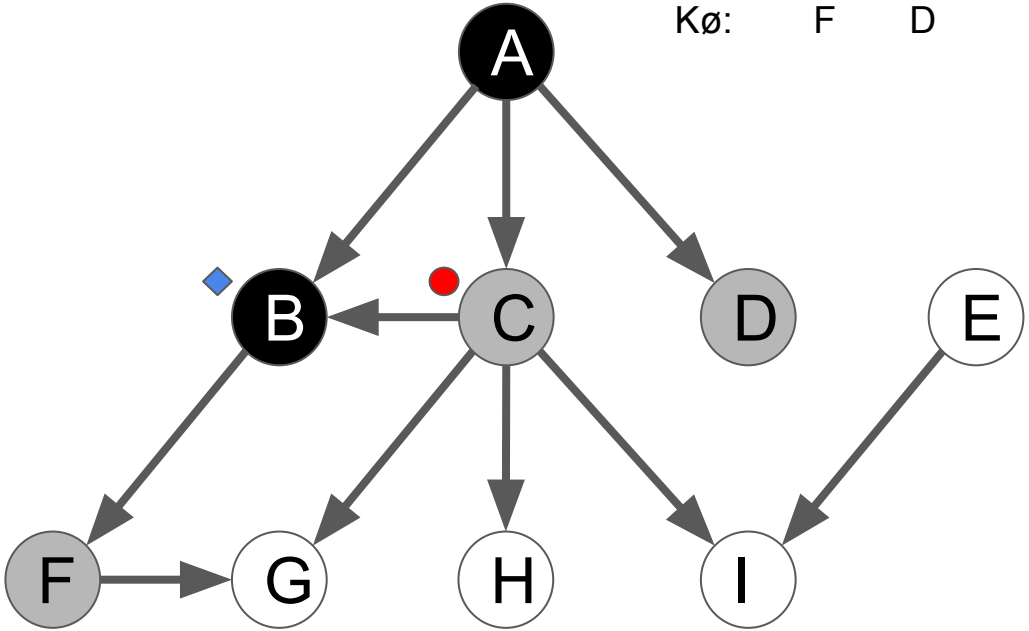
Kø: F D



Grå: A B C D F

Svart: A B

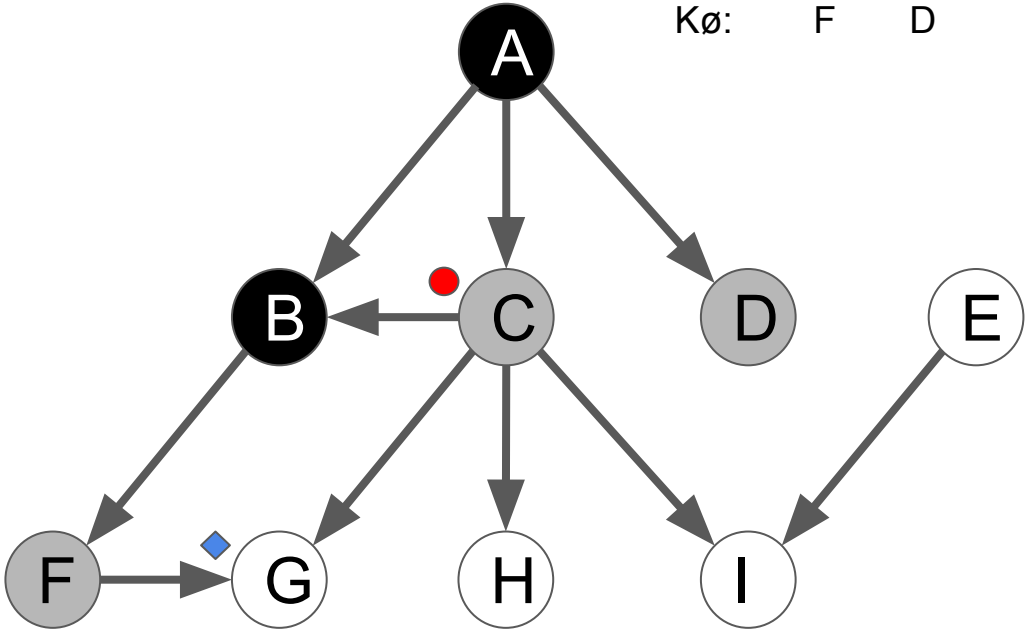
Kø: F D



Grå: A B C D F

Svart: A B

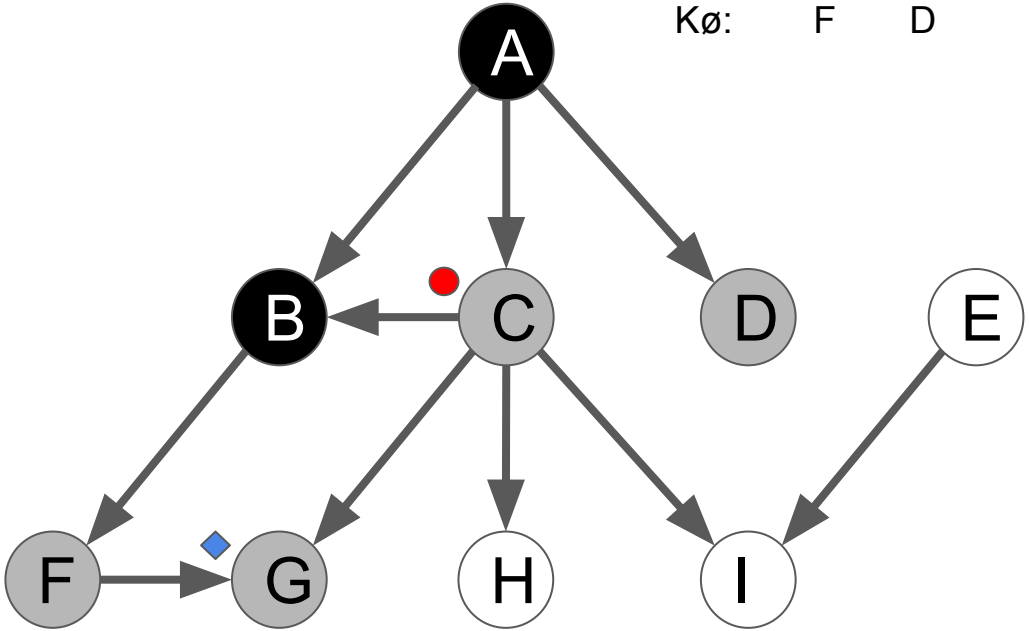
Kø: F D



Grå: A B C D F G

Svart: A B

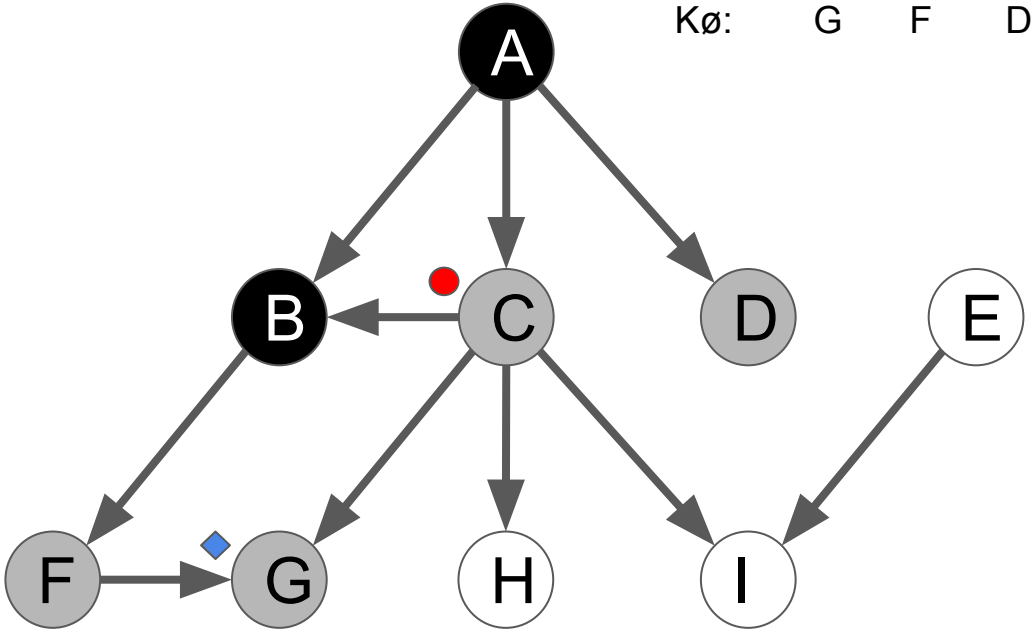
Kø: F D



Grå: A B C D F G

Svart: A B

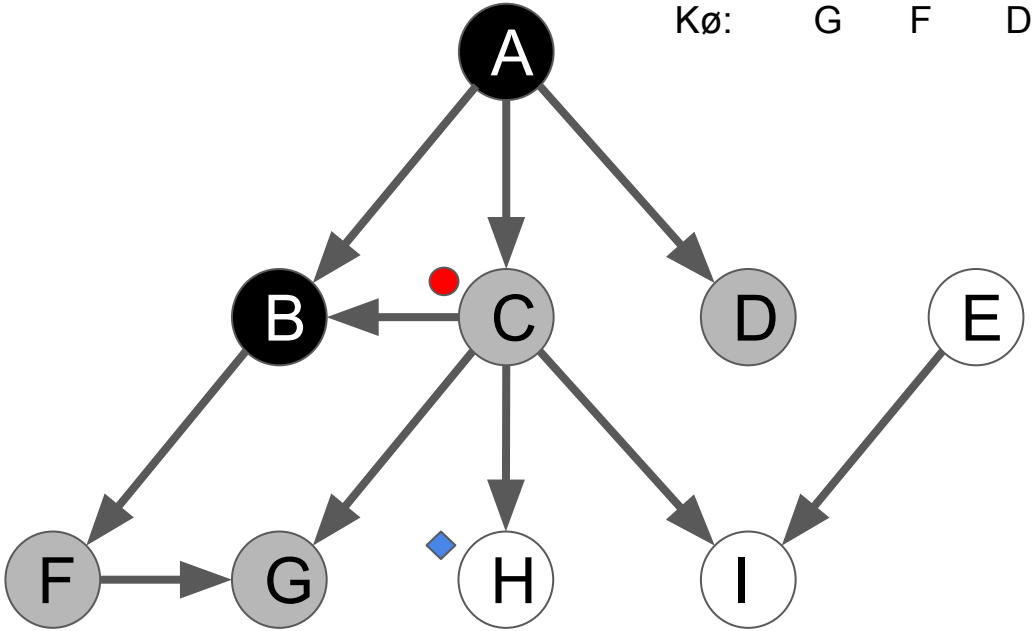
Kø: G F D



Grå: A B C D F G

Svart: A B

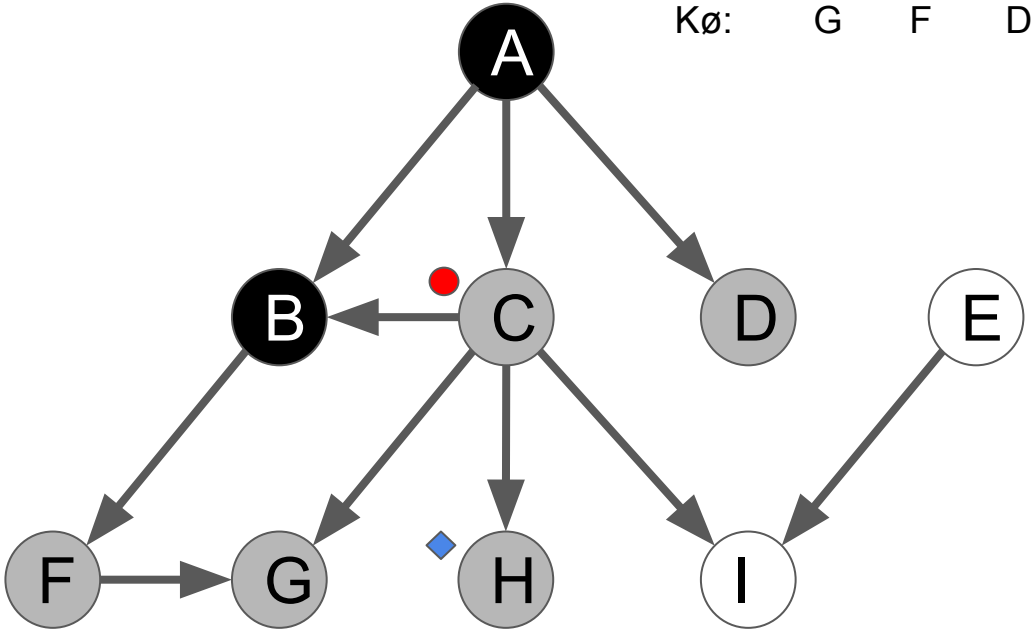
Kø: G F D



Grå: A B C D F G H

Svart: A B

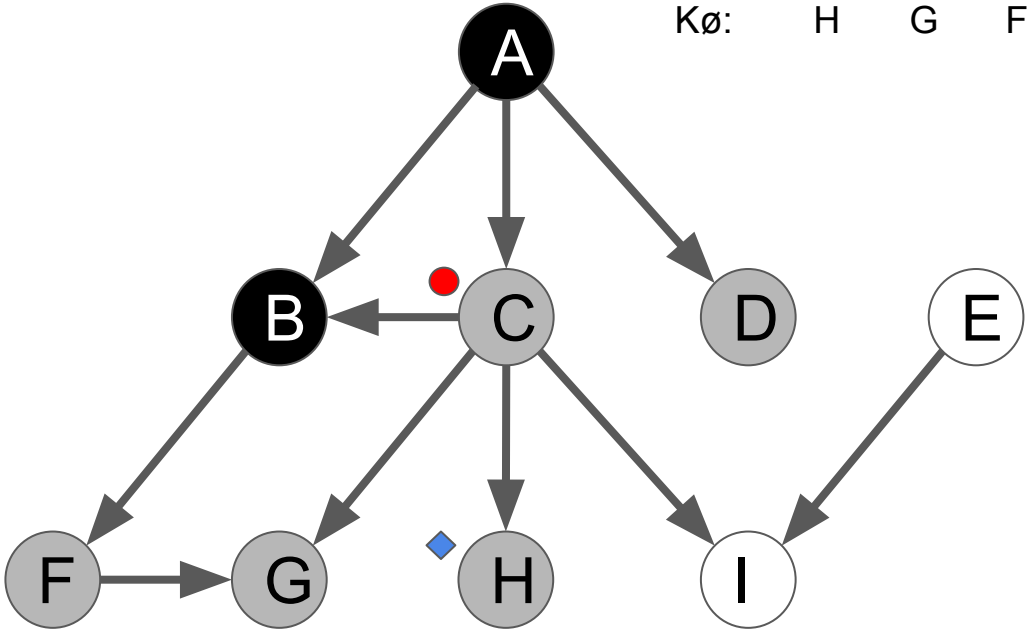
Kø: G F D



Grå: A B C D F G H

Svart: A B

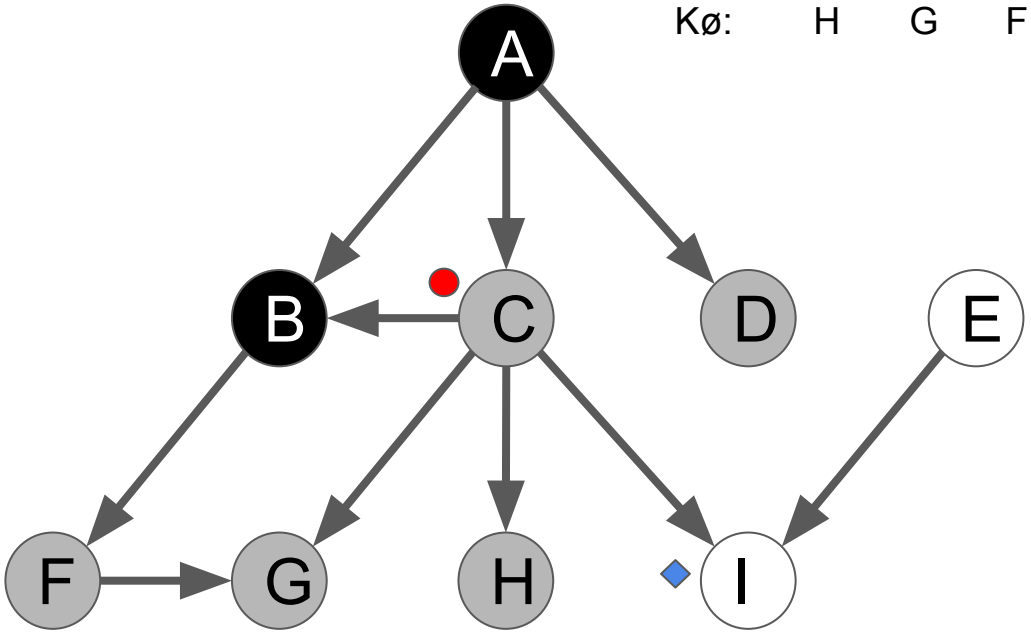
Kø: H G F D



Grå: A B C D F G H

Svart: A B

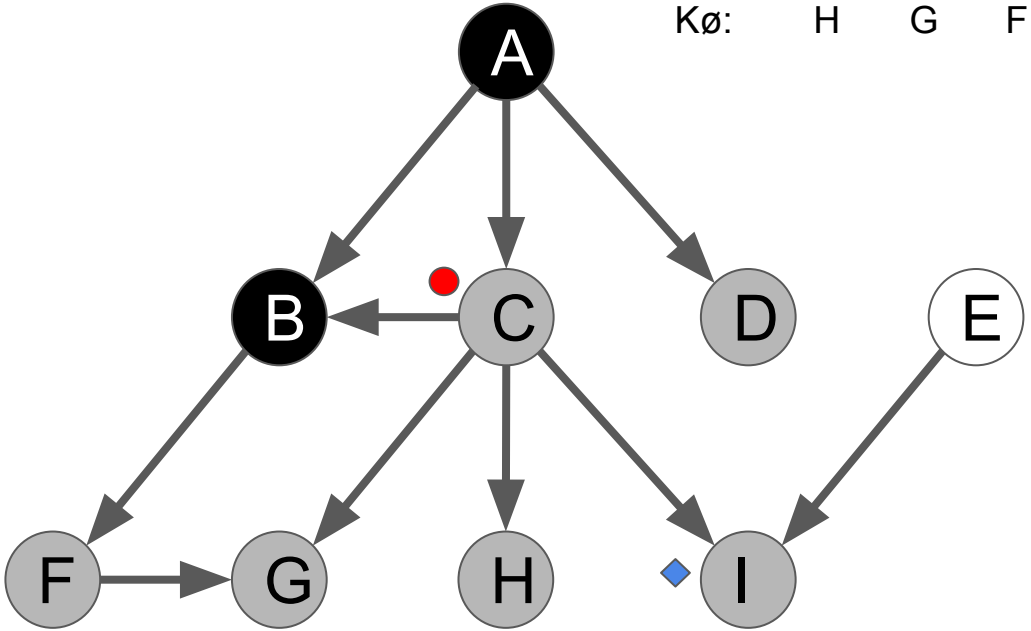
Kø: H G F D



Grå: A B C D F G H I

Svart: A B

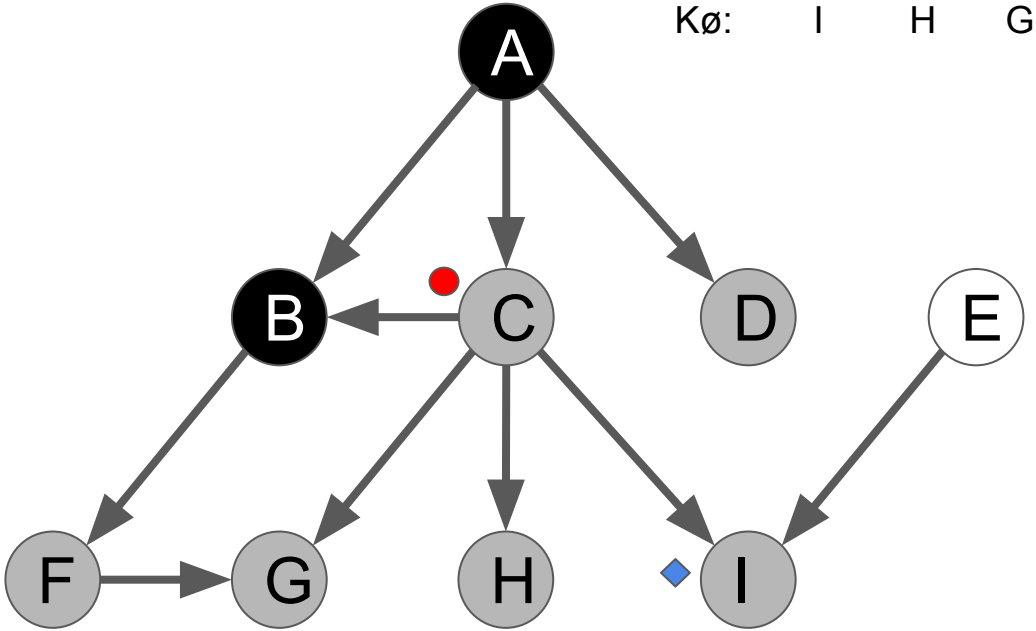
Kø: H G F D



Grå: A B C D F G H I

Svart: A B

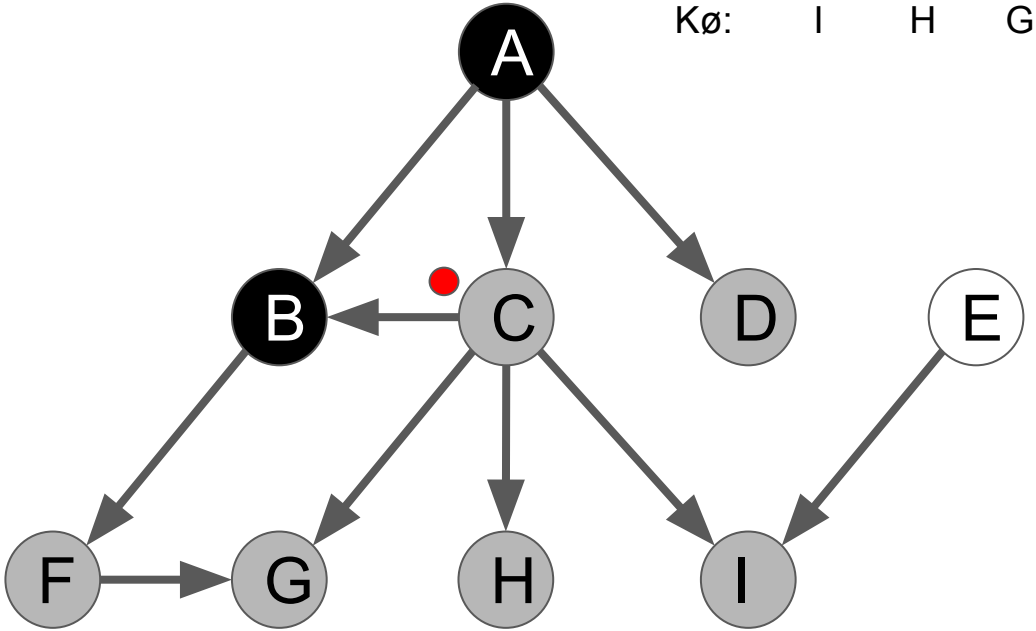
Kø: I H G F D



Grå: A B C D F G H I

Svart: A B

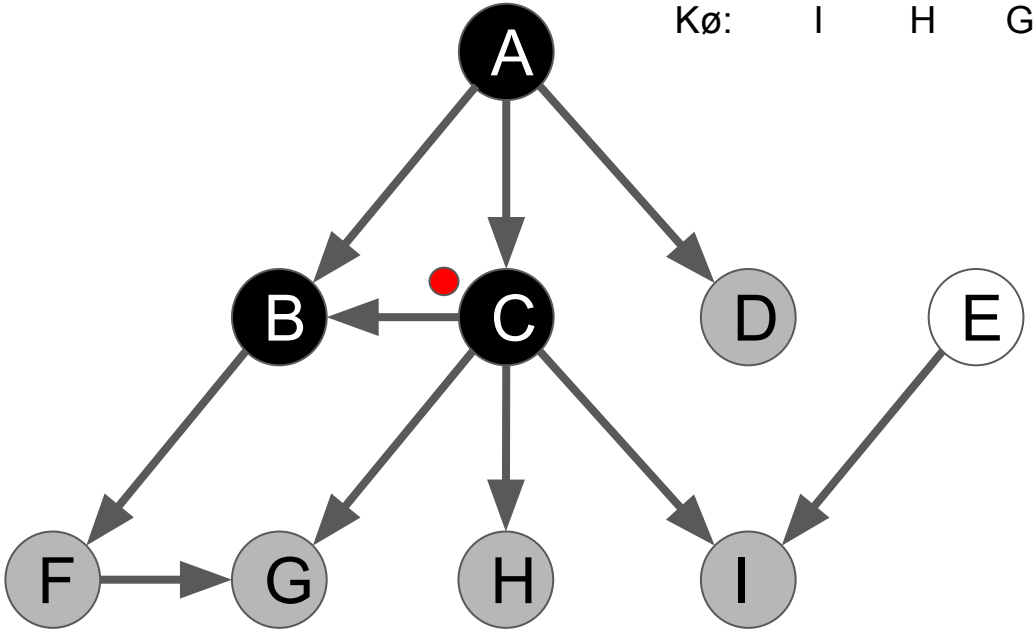
Kø: I H G F D



Grå: A B C D F G H I

Svart: A B C

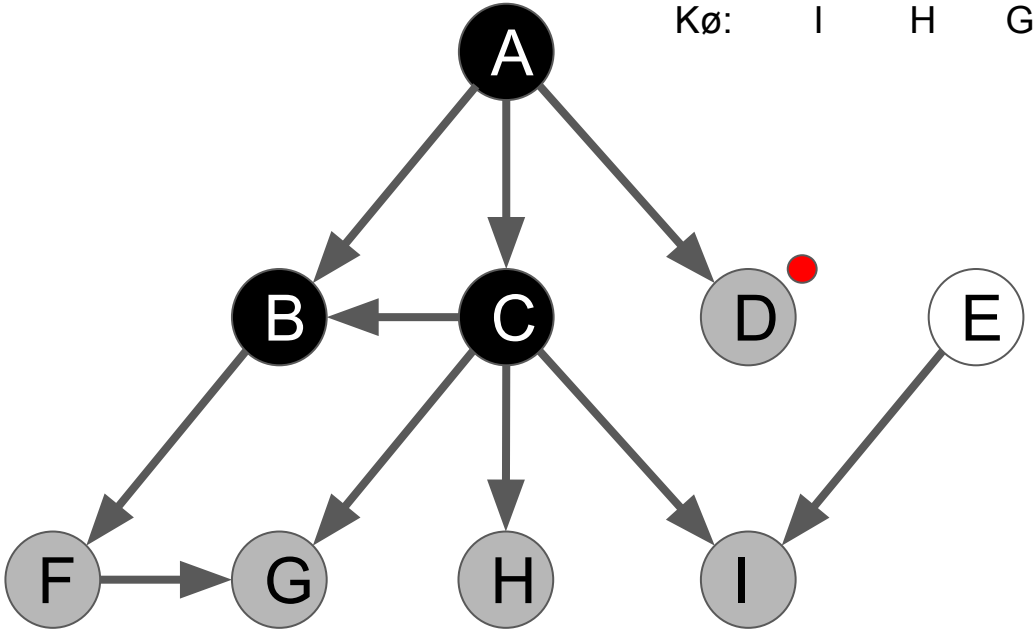
Kø: I H G F D



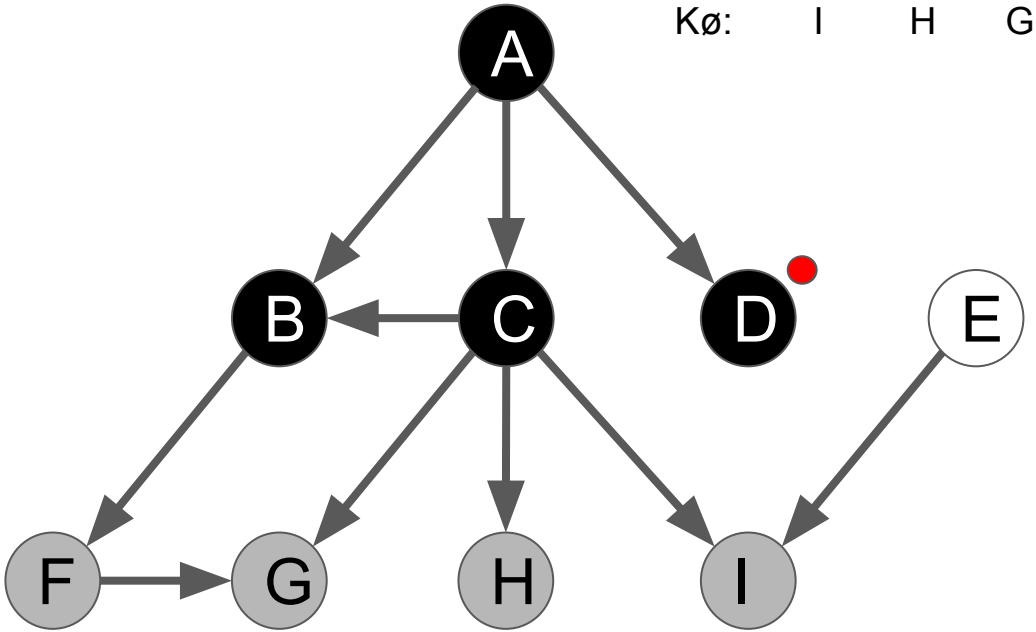
Grå: A B C D F G H I

Svart: A B C

Kø: I H G F



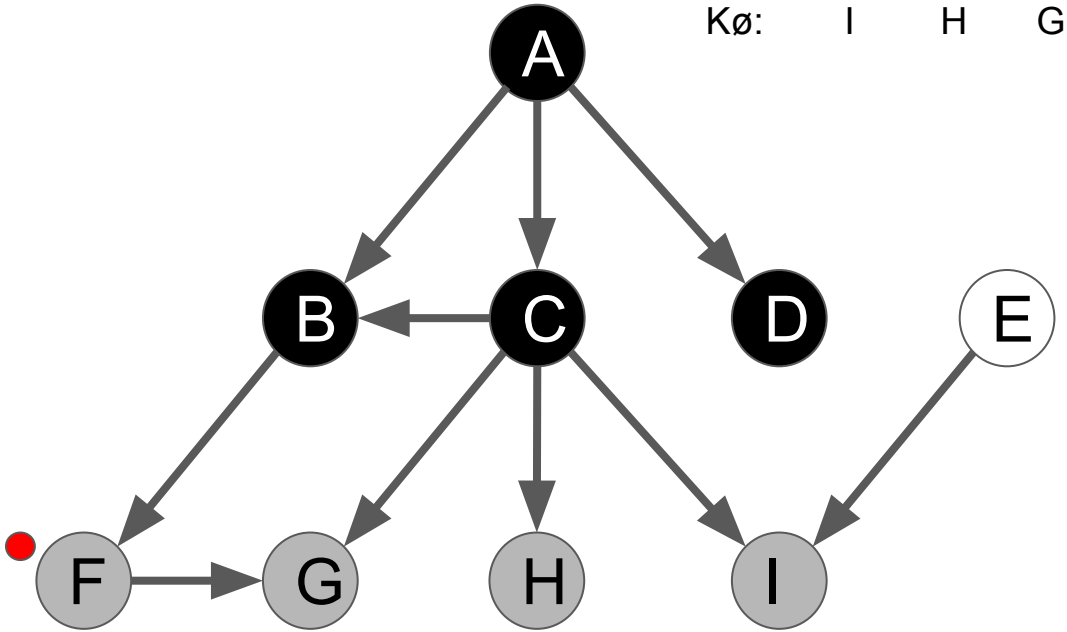
Grå:	A	B	C	D	F	G	H	I
Svart:	A	B	C	D				
Kø:	I	H	G	F				



Grå: A B C D F G H I

Svart: A B C D

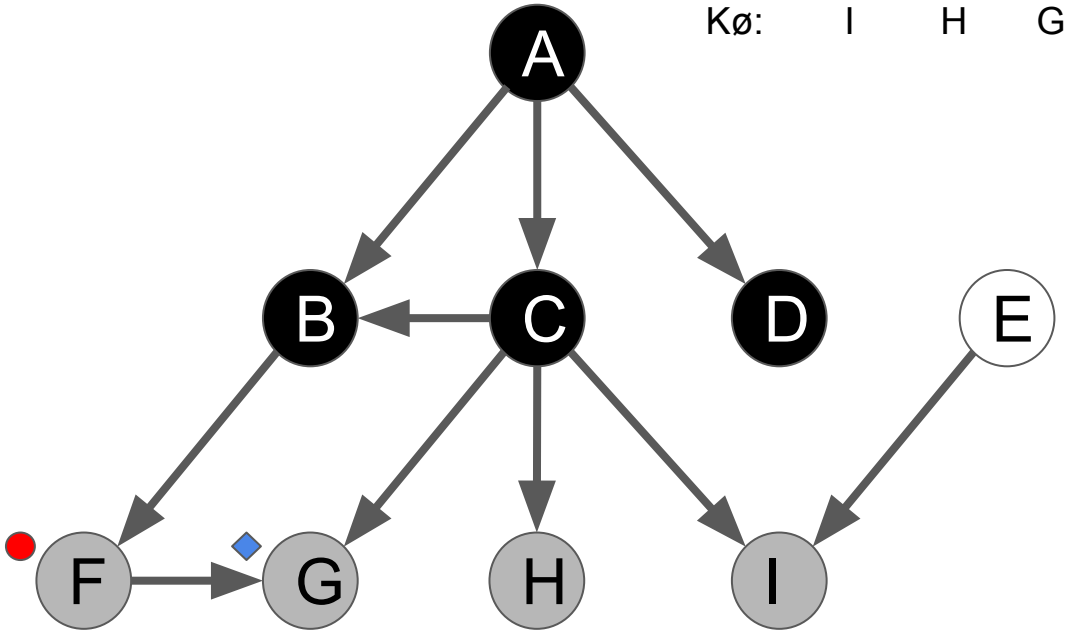
Kø: I H G



Grå: A B C D F G H I

Svart: A B C D

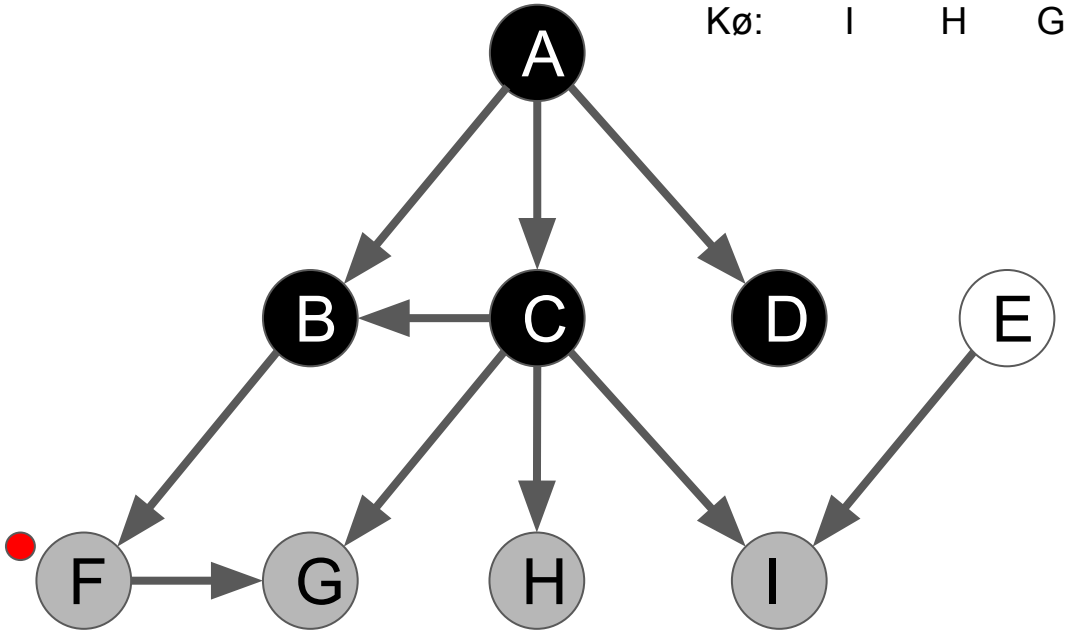
Kø: I H G



Grå: A B C D F G H I

Svart: A B C D

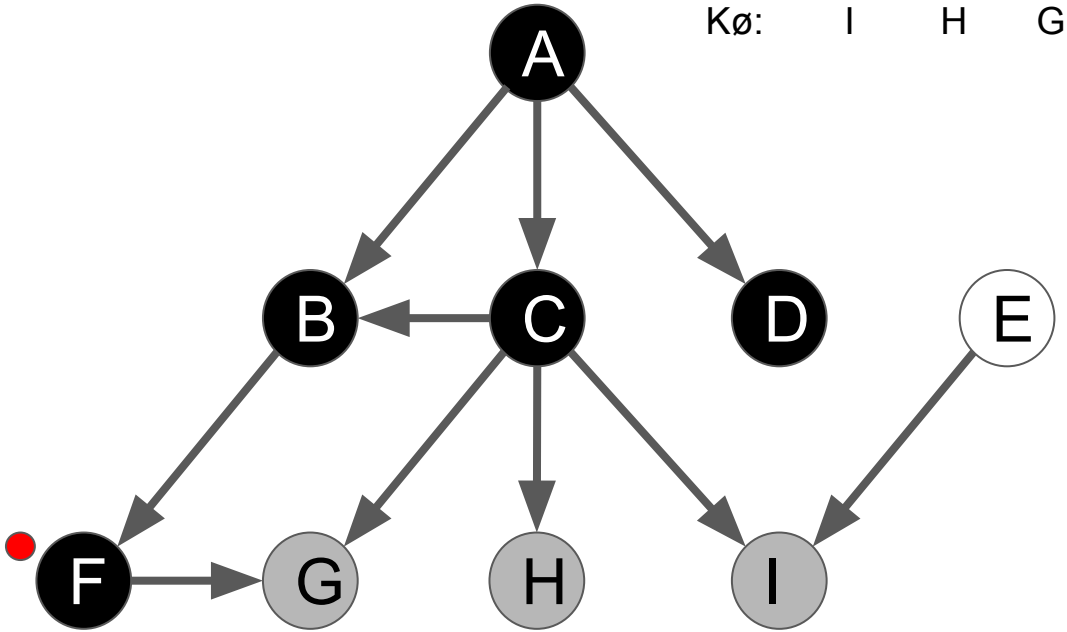
Kø: I H G



Grå: A B C D F G H I

Svart: A B C D F

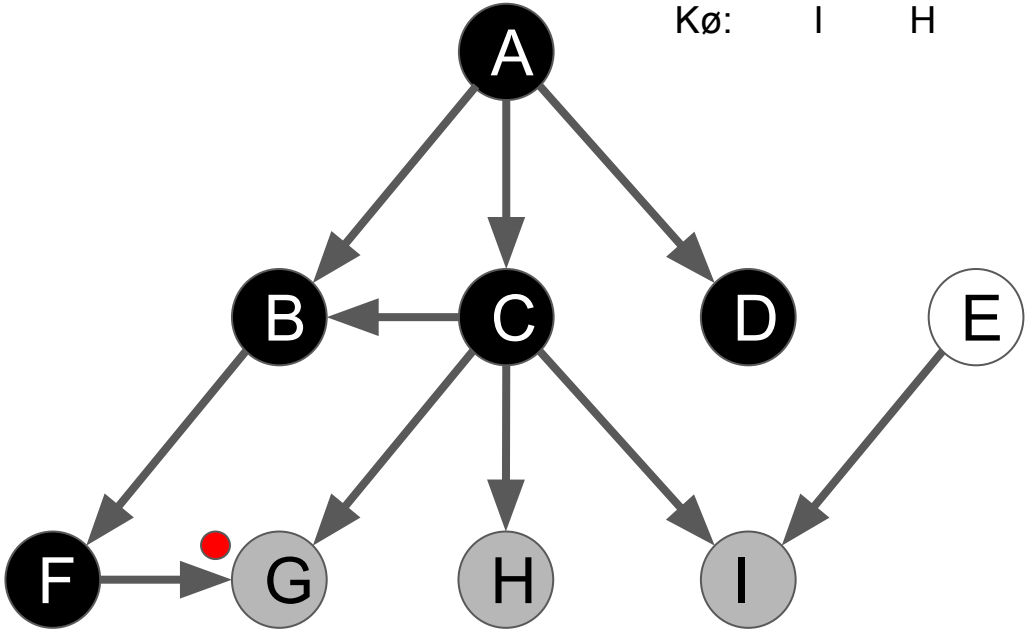
Kø: I H G



Grå: A B C D F G H I

Svart: A B C D F

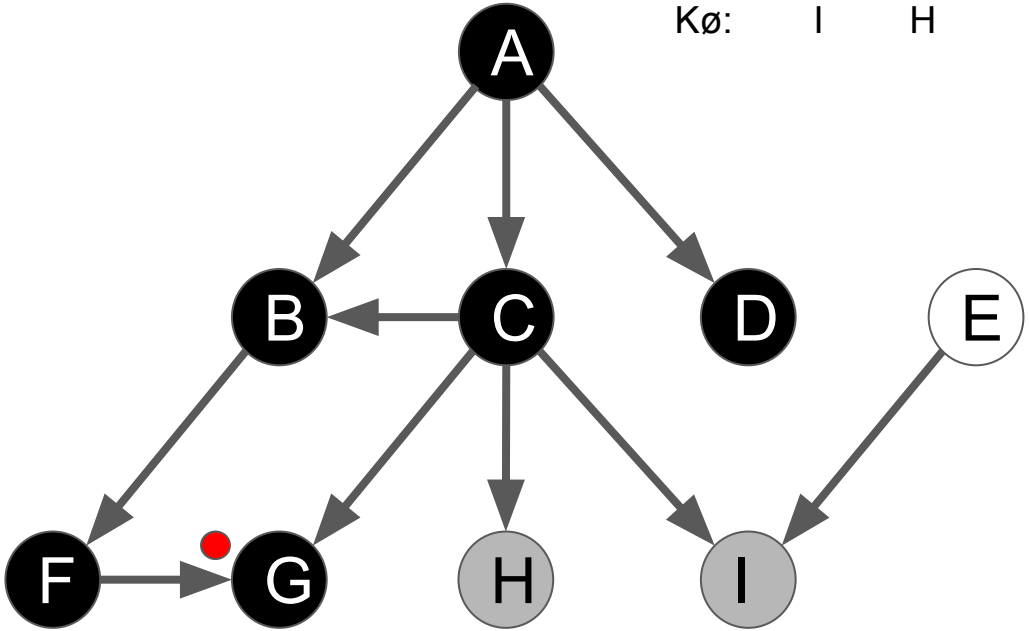
Kø: I H



Grå: A B C D F G H I

Svart: A B C D F G

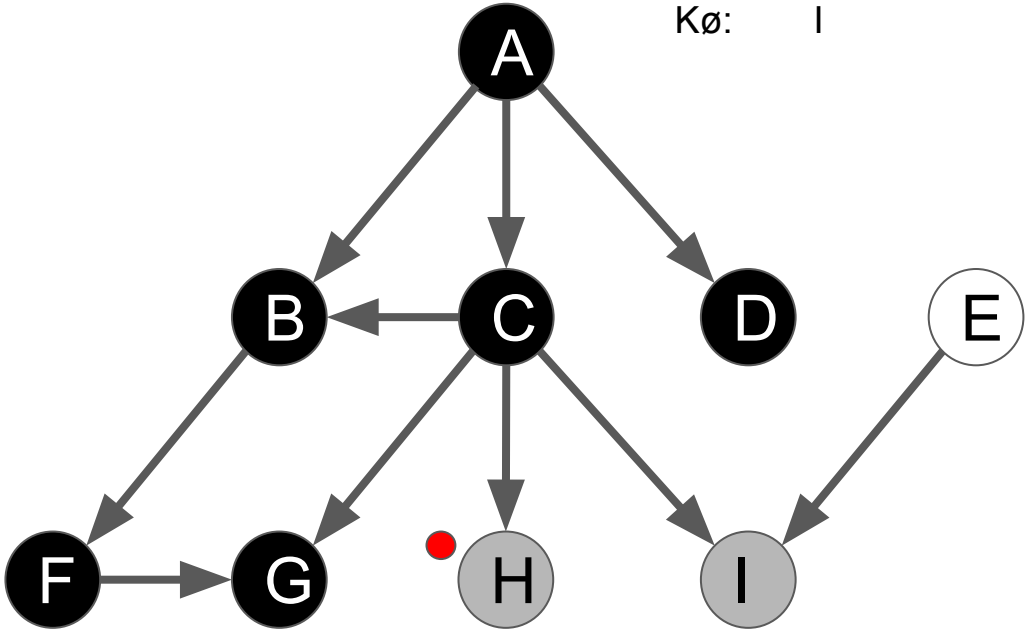
Kø: I H



Grå: A B C D F G H I

Svart: A B C D F G

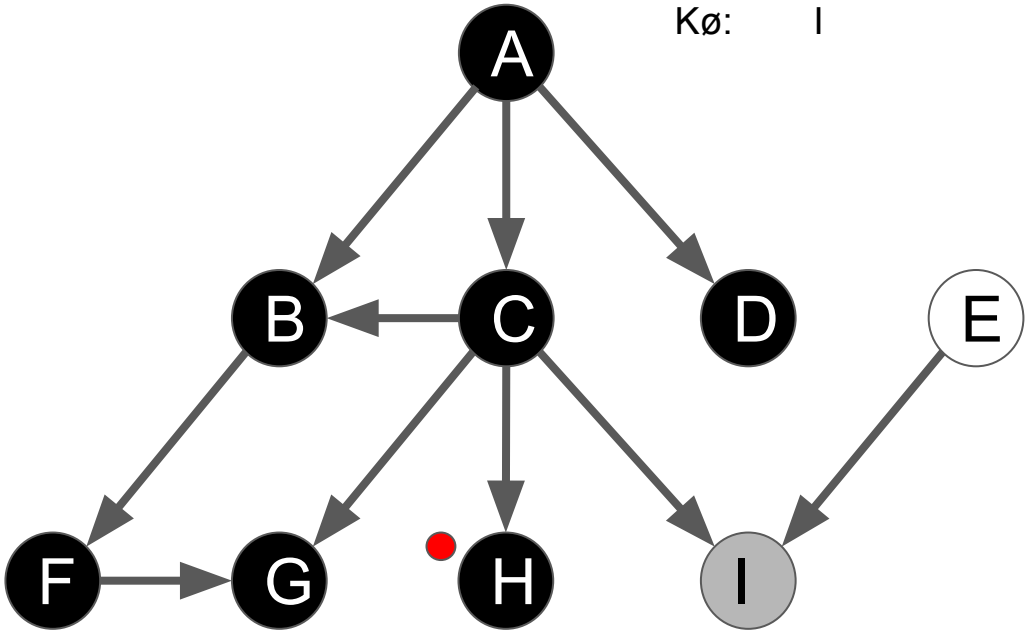
Kø: I



Grå: A B C D F G H I

Svart: A B C D F G H

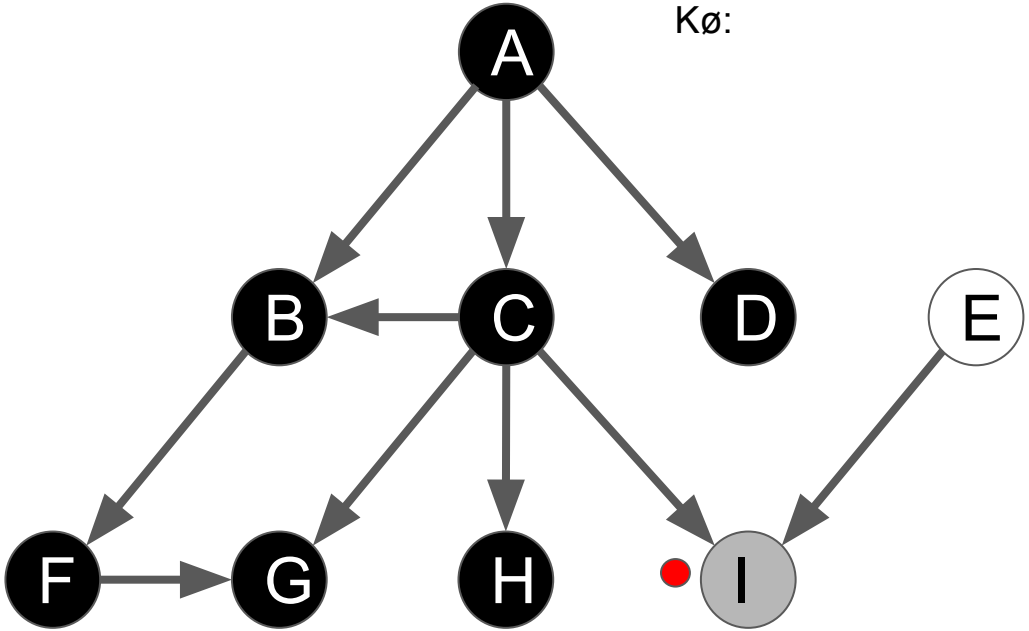
Kø: I



Grå: A B C D F G H I

Svart: A B C D F G H

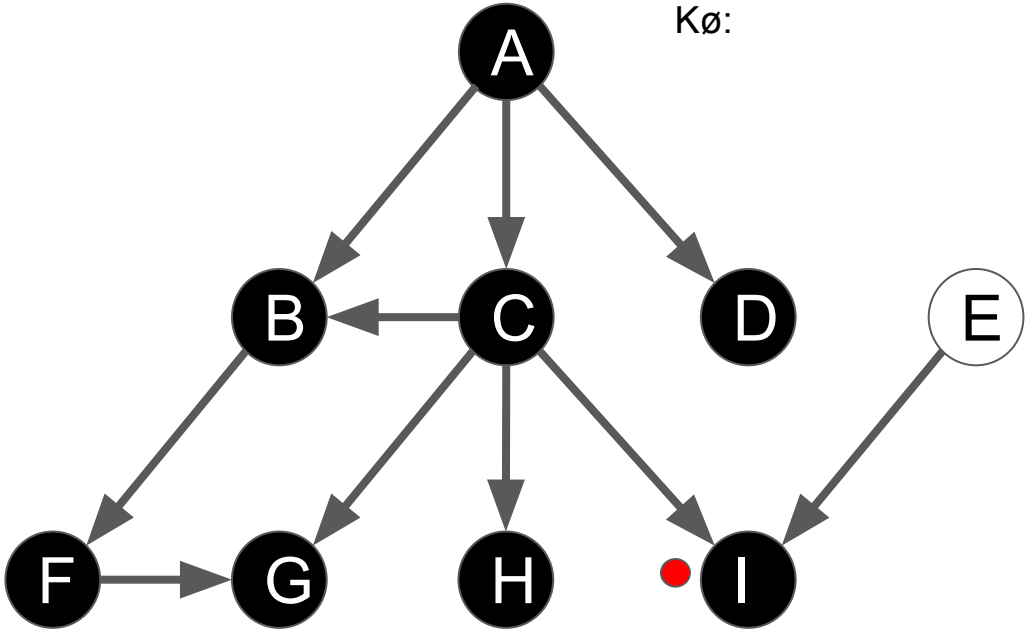
Kø:



Grå: A B C D F G H I

Svart: A B C D F G H I

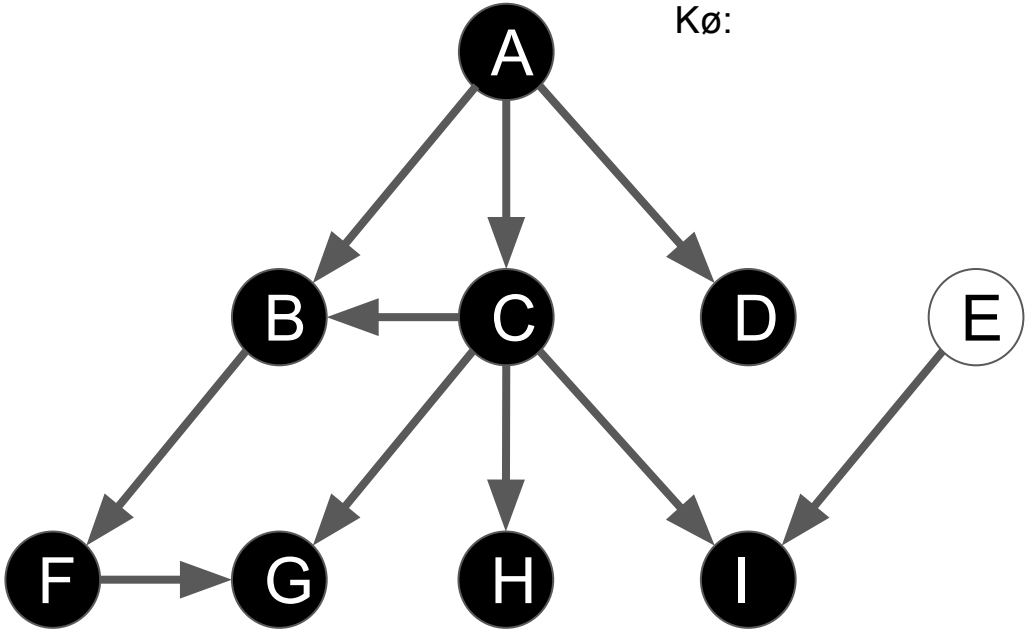
Kø:



Grå: A B C D F G H I

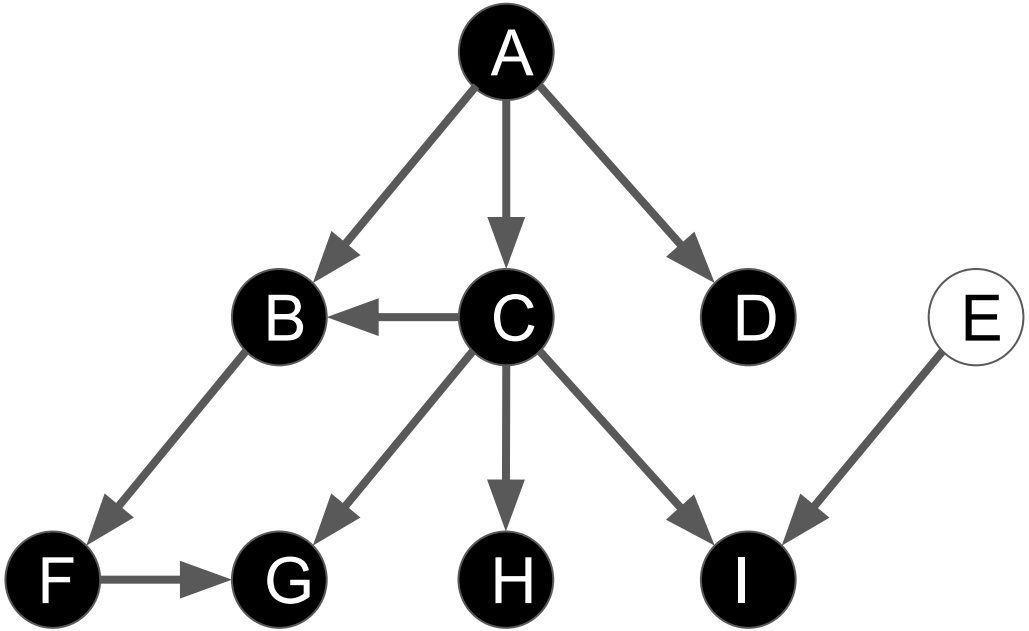
Svart: A B C D F G H I

Kø:



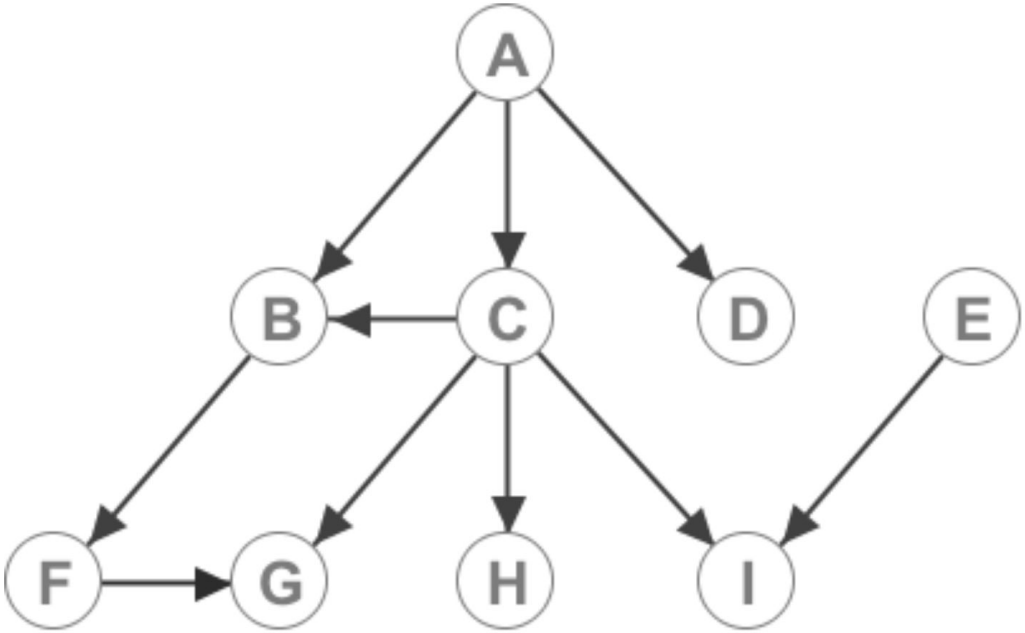
Grå: A B C D F G H I

Svart: **A** **B** **C** **D** F G H I



Question 8: Dybde-først-søk

DFS blir kjørt med påfølgende graf med A som rotnode. I hvilken rekkefølge blir de fire første nodene farget svart?

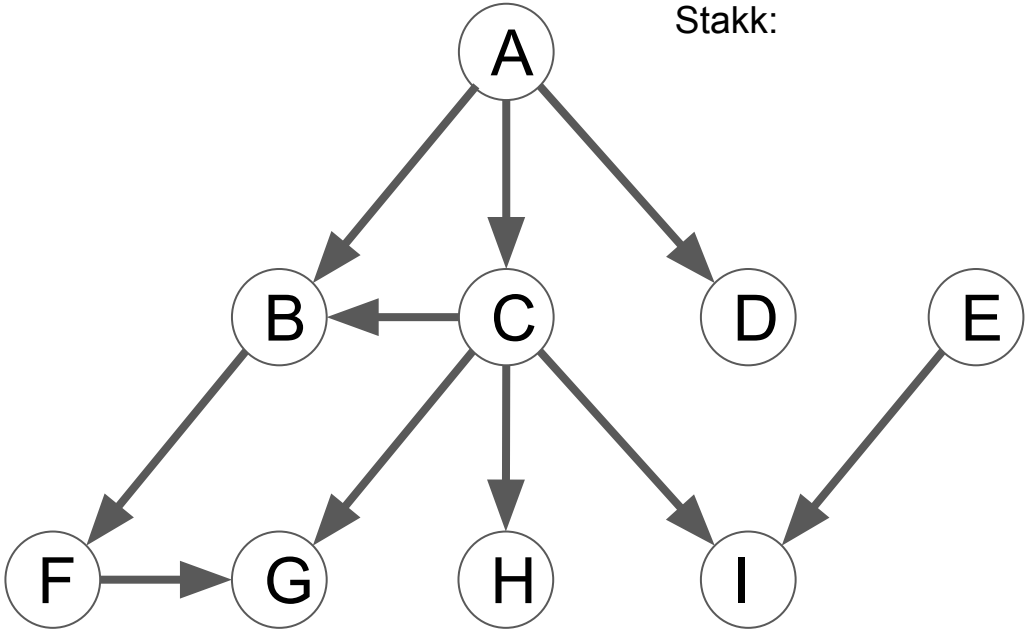


Anta at alle konflikter løses ved hjelp av leksikografisk ordning (ved eventuelle konflikter velges den noden med bokstav tidligst i alfabetet, altså A før B, B før C osv.)

Grå:

Svart:

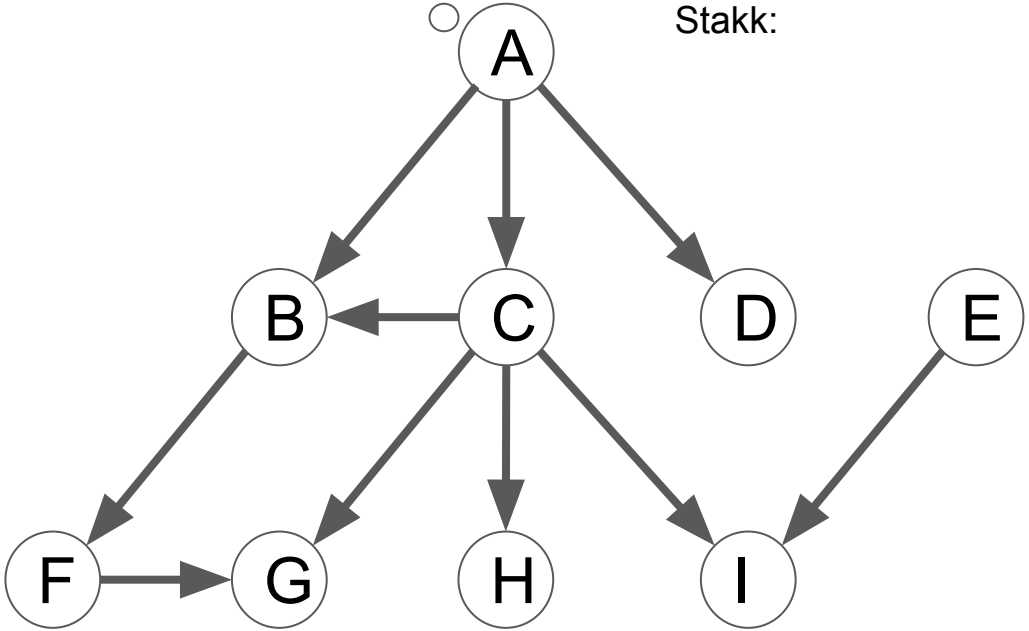
Stakk:



Grå:

Svart:

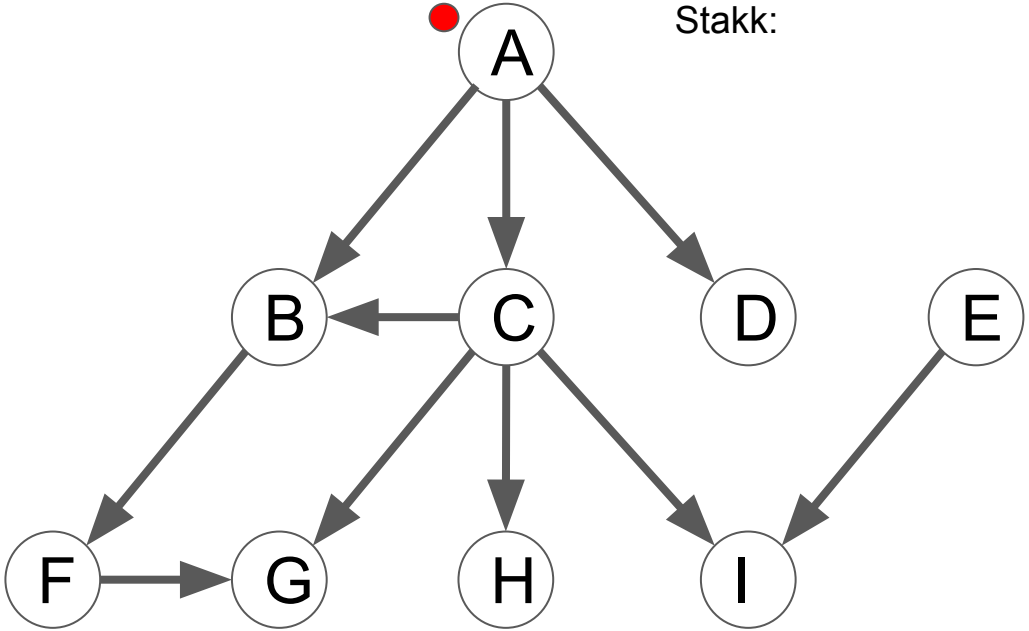
Stakk:



Grå:

Svart:

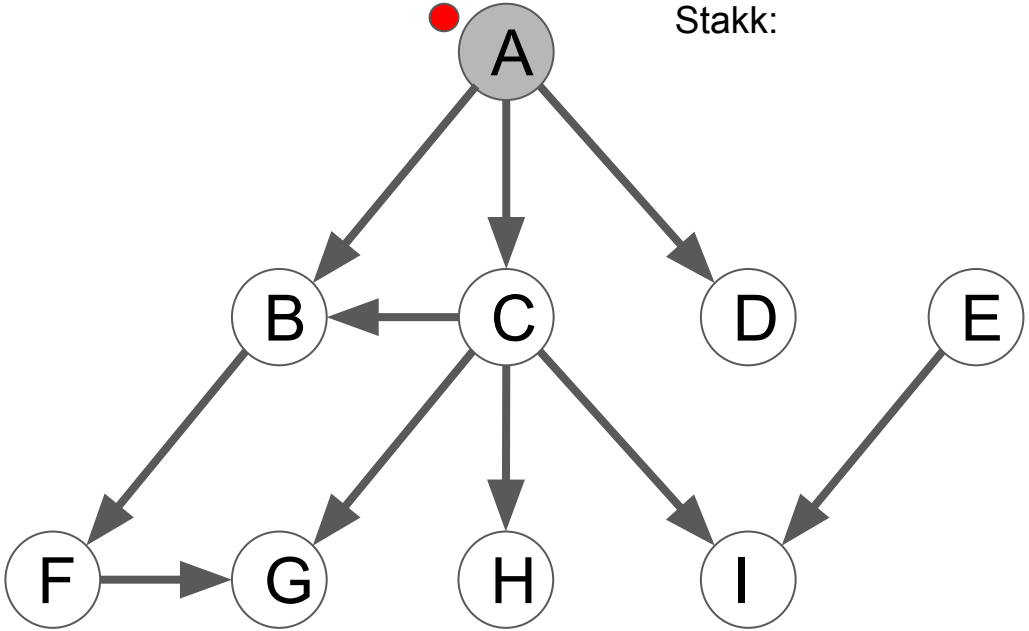
Stakk:



Grå: A

Svart:

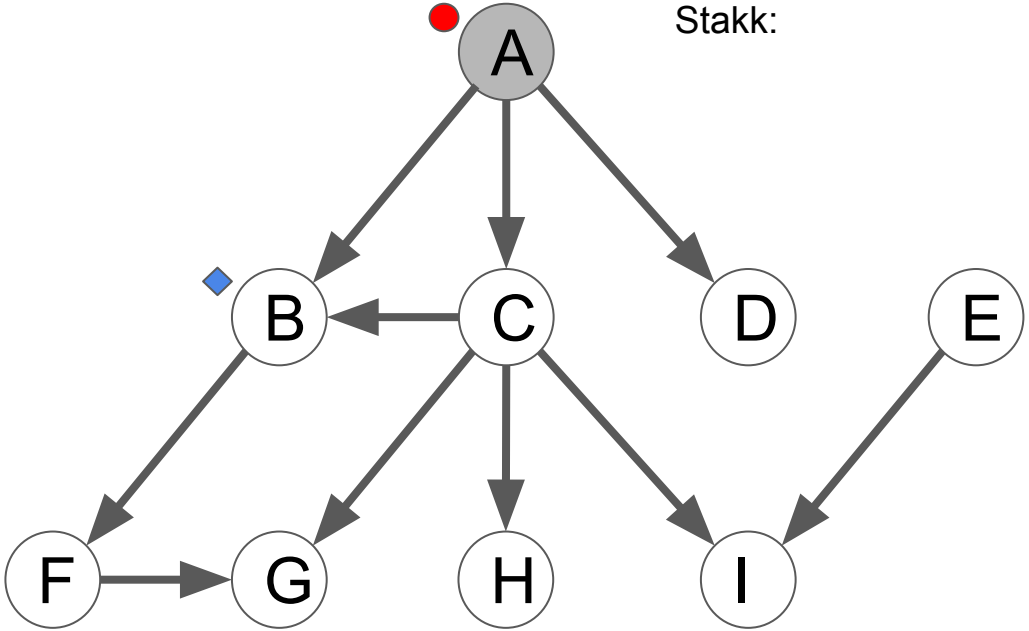
Stakk:



Grå: A

Svart:

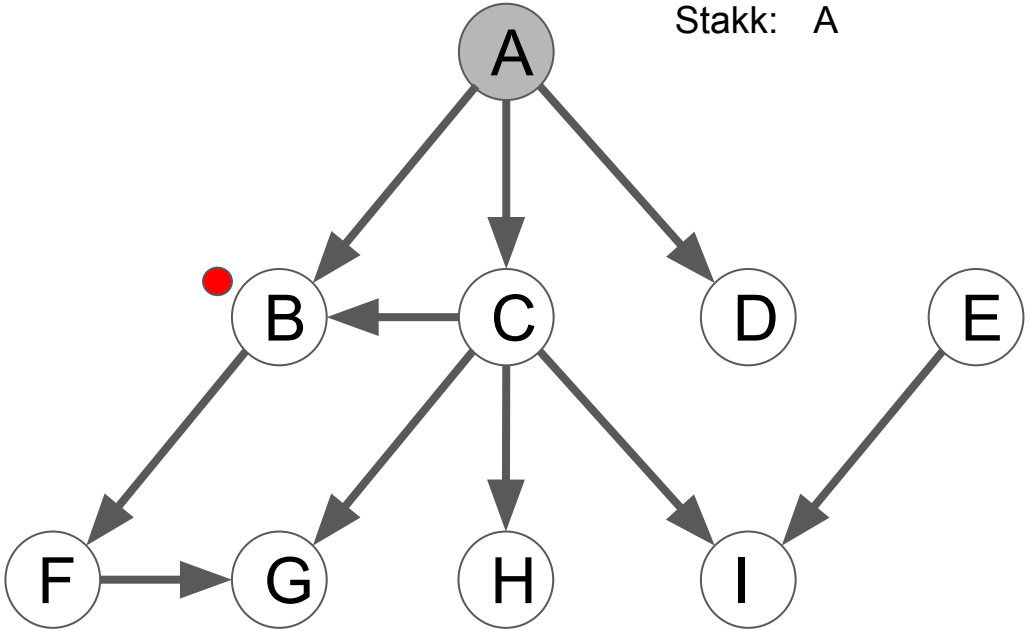
Stakk:



Grå: A

Svart:

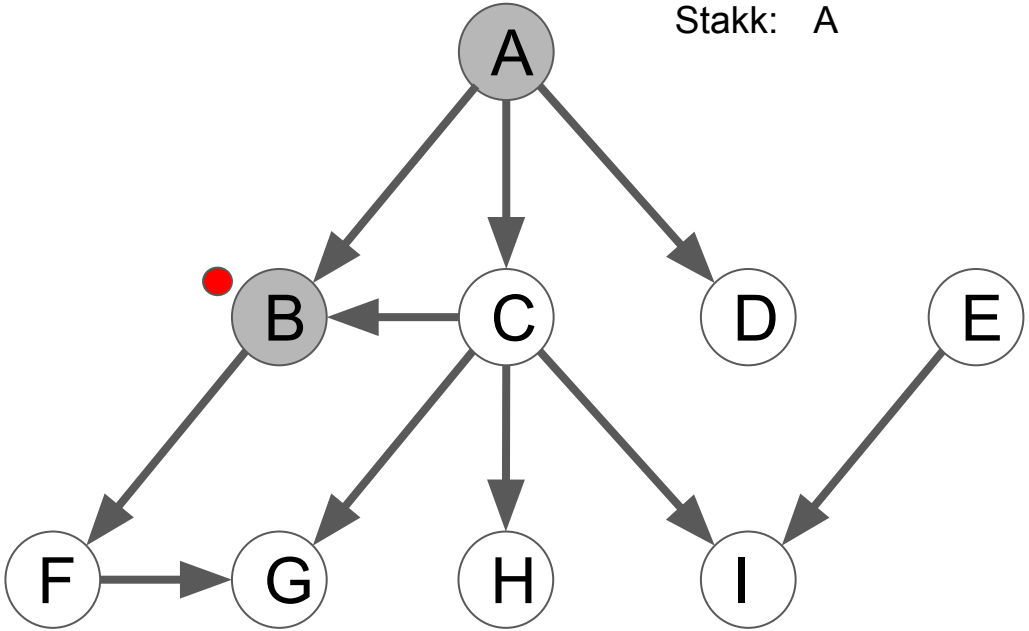
Stakk: A



Grå: A B

Svart:

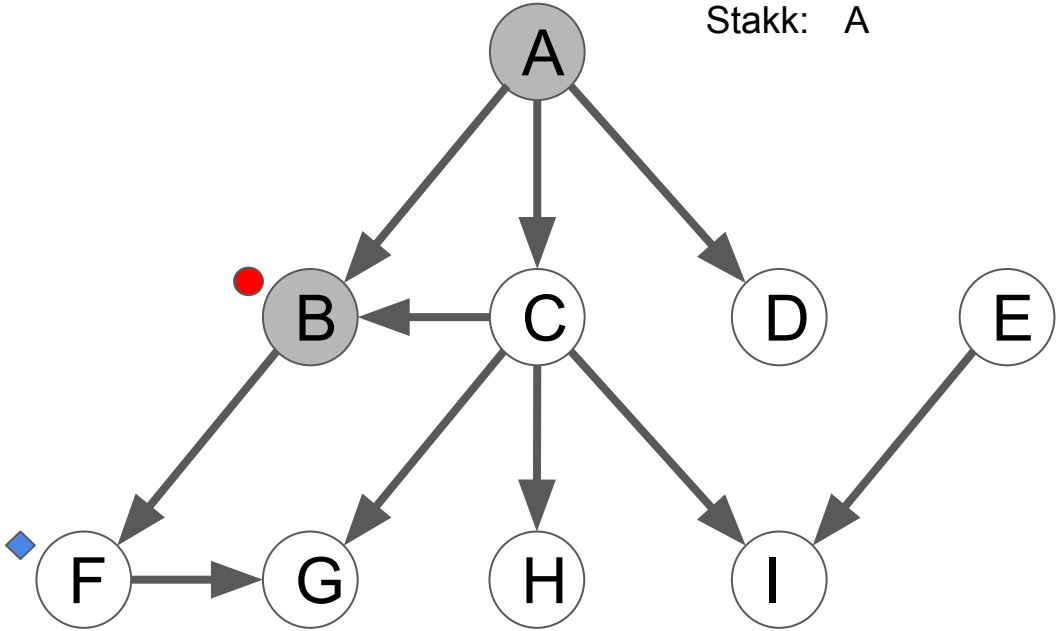
Stakk: A



Grå: A B

Svart:

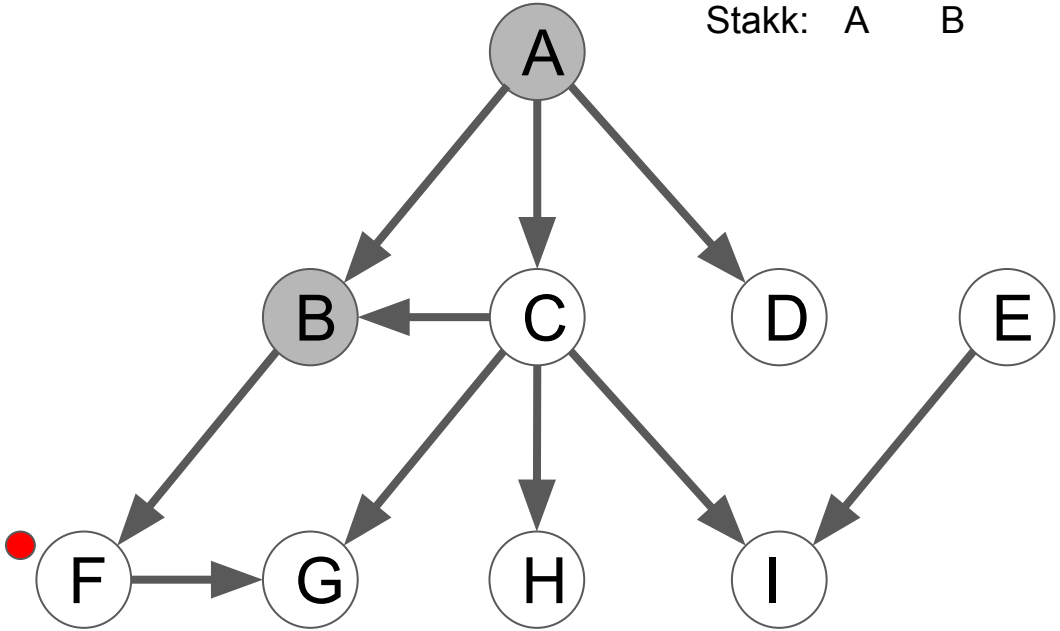
Stakk: A



Grå: A B

Svart:

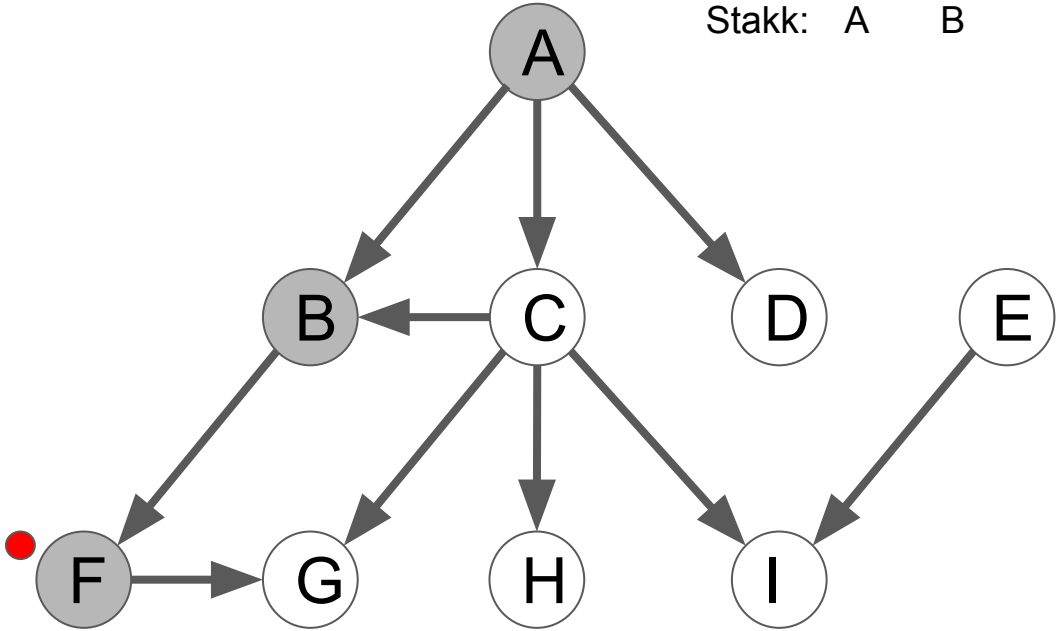
Stakk: A B



Grå: A B F

Svart:

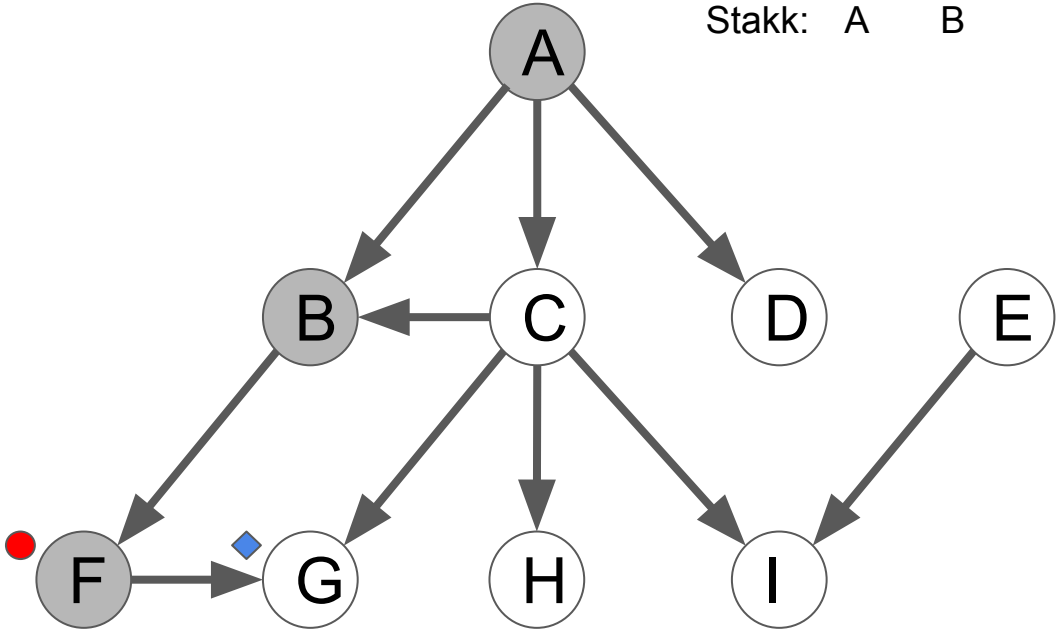
Stakk: A B



Grå: A B F

Svart:

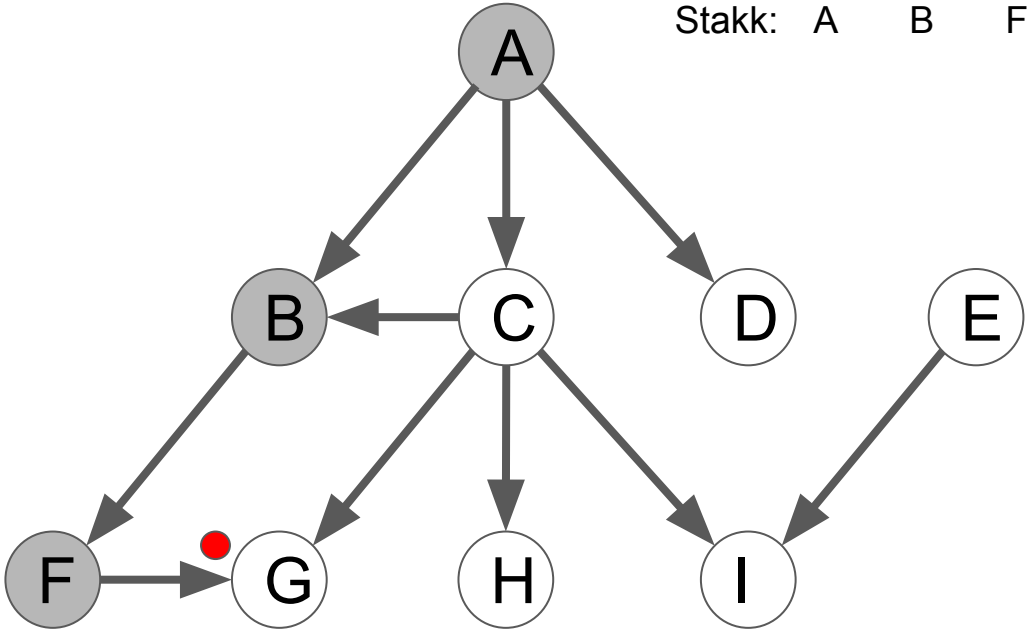
Stakk: A B



Grå: A B F

Svart:

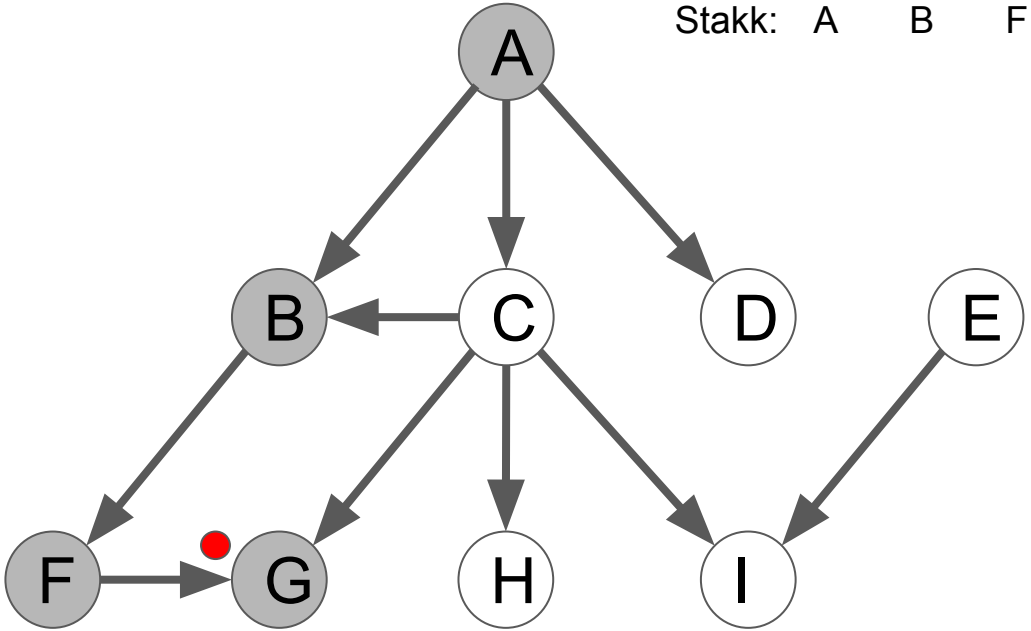
Stakk: A B F



Grå: A B F G

Svart:

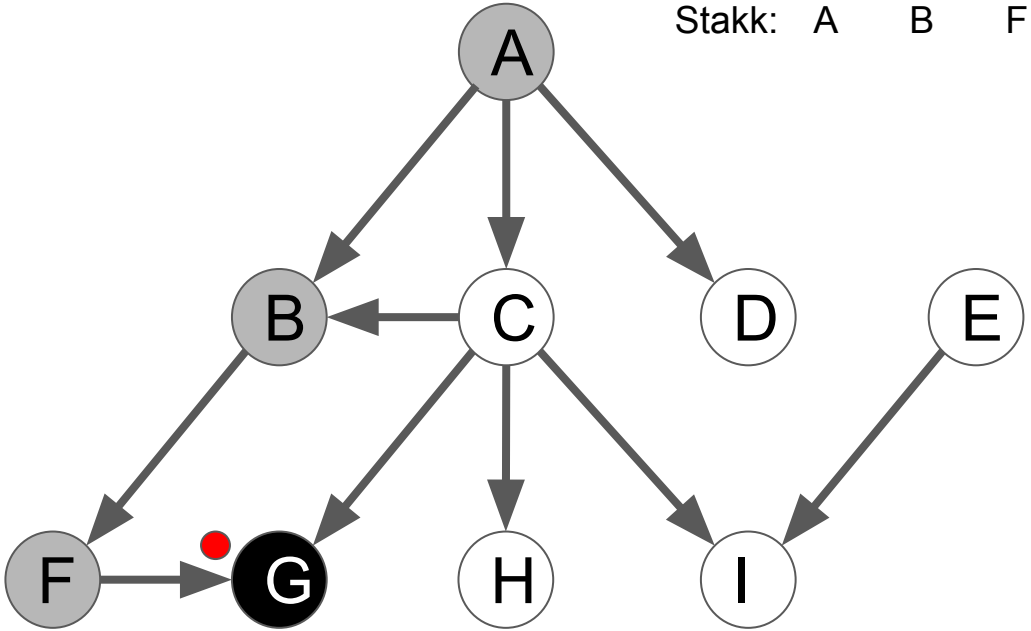
Stakk: A B F



Grå: A B F G

Svart: G

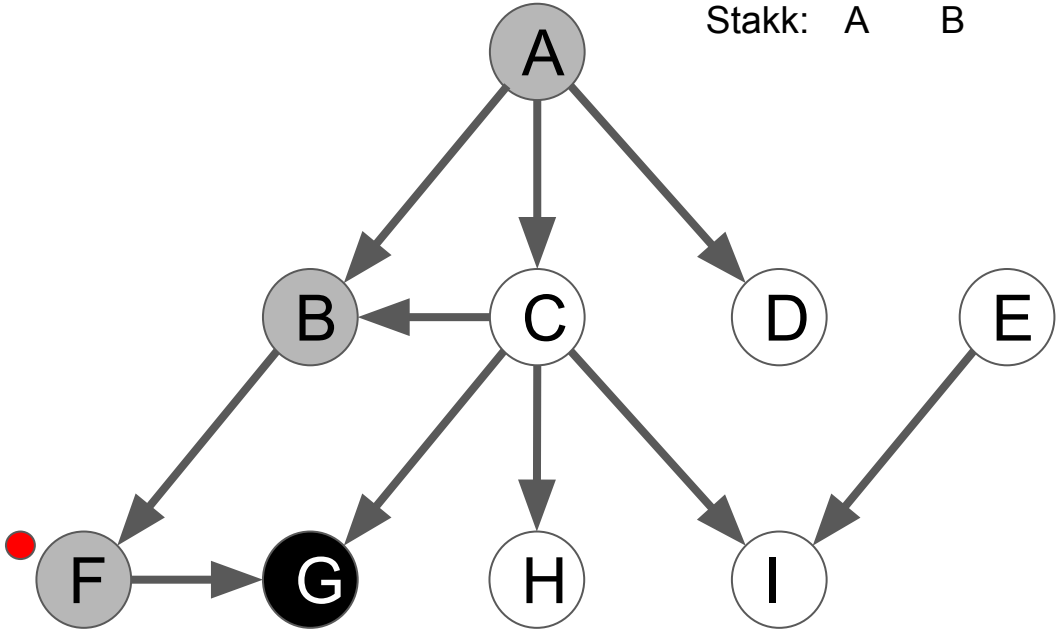
Stakk: A B F



Grå: A B F G

Svart: G

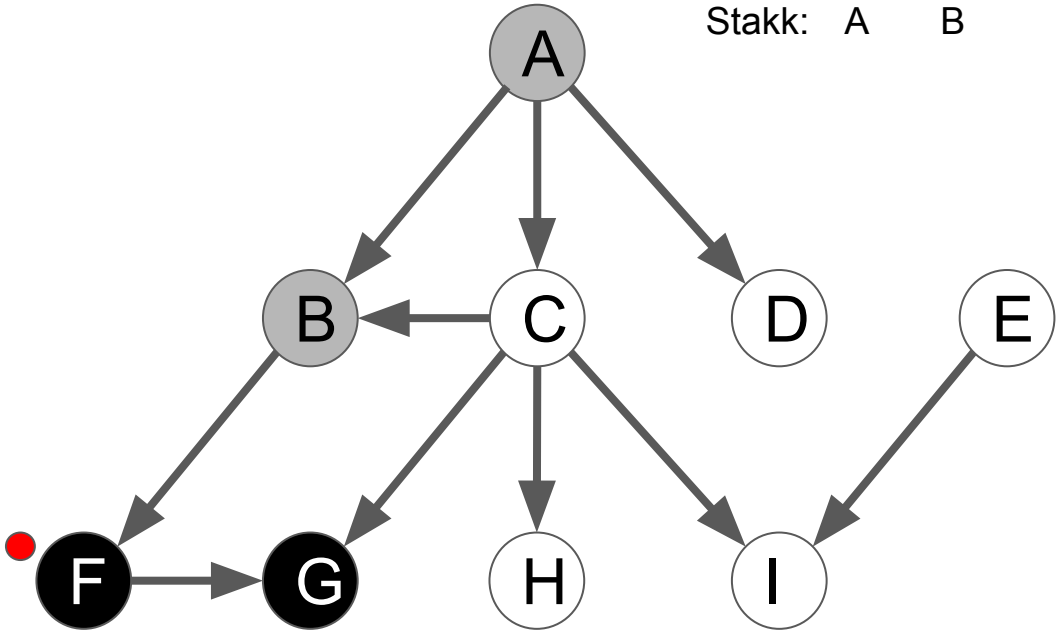
Stakk: A B



Grå: A B F G

Svart: G F

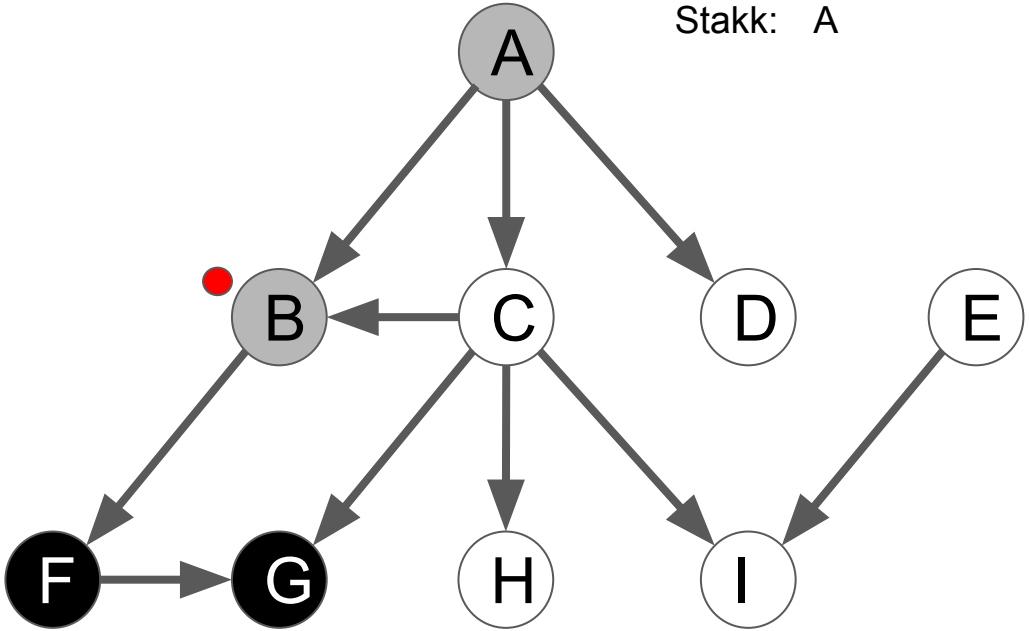
Stakk: A B



Grå: A B F G

Svart: G F

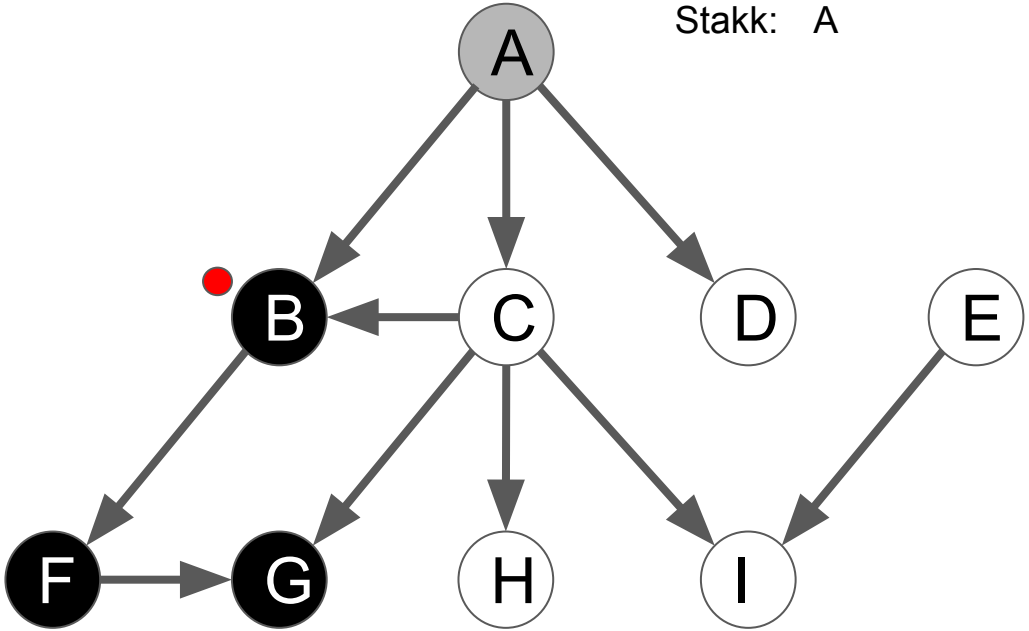
Stakk: A



Grå: A B F G

Svart: G F B

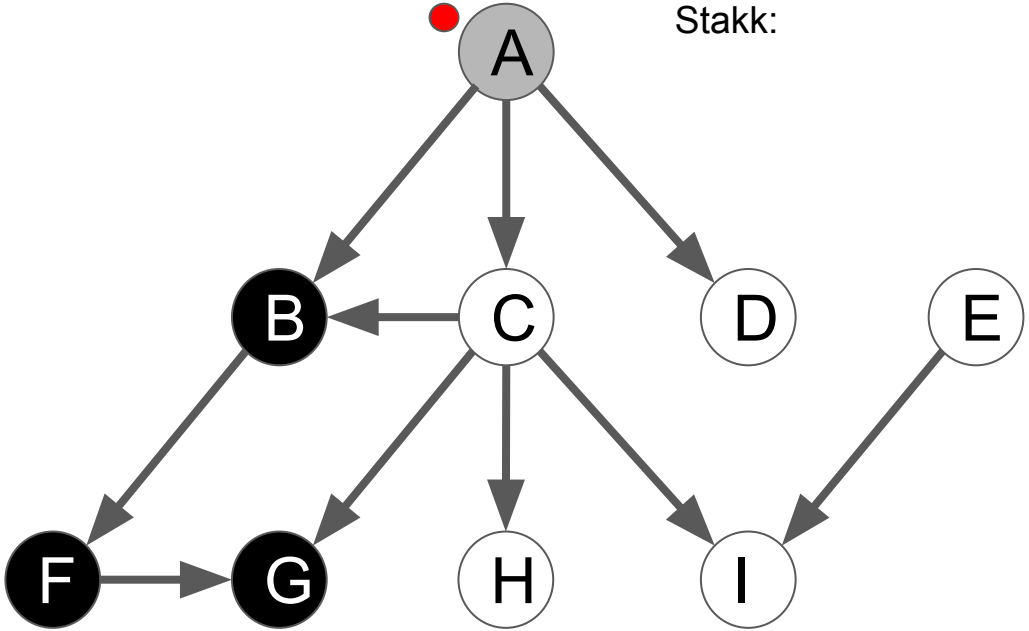
Stakk: A



Grå: A B F G

Svart: G F B

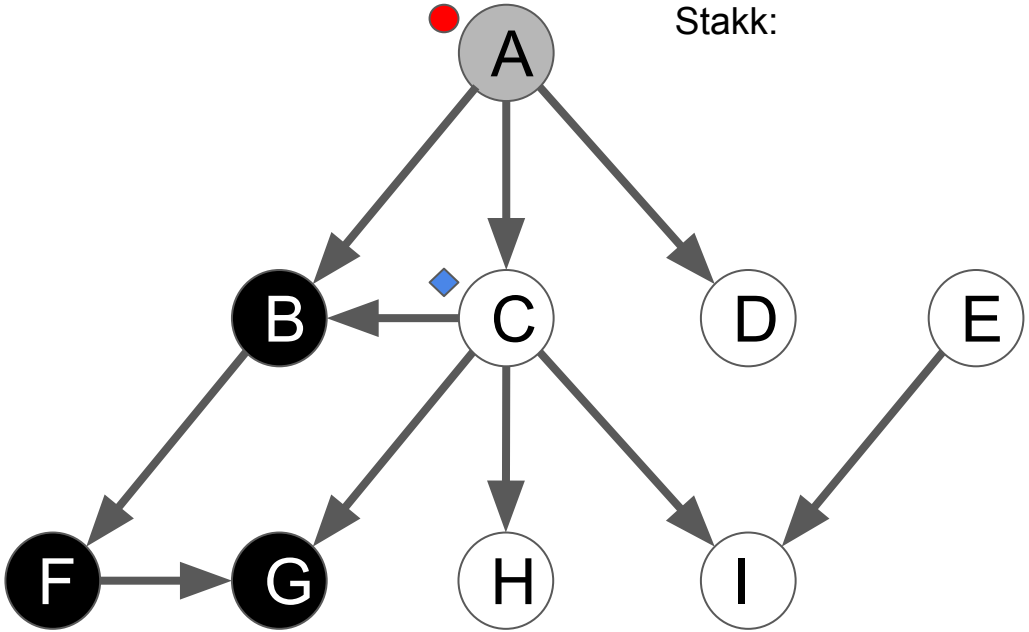
Stakk:



Grå: A B F G

Svart: G F B

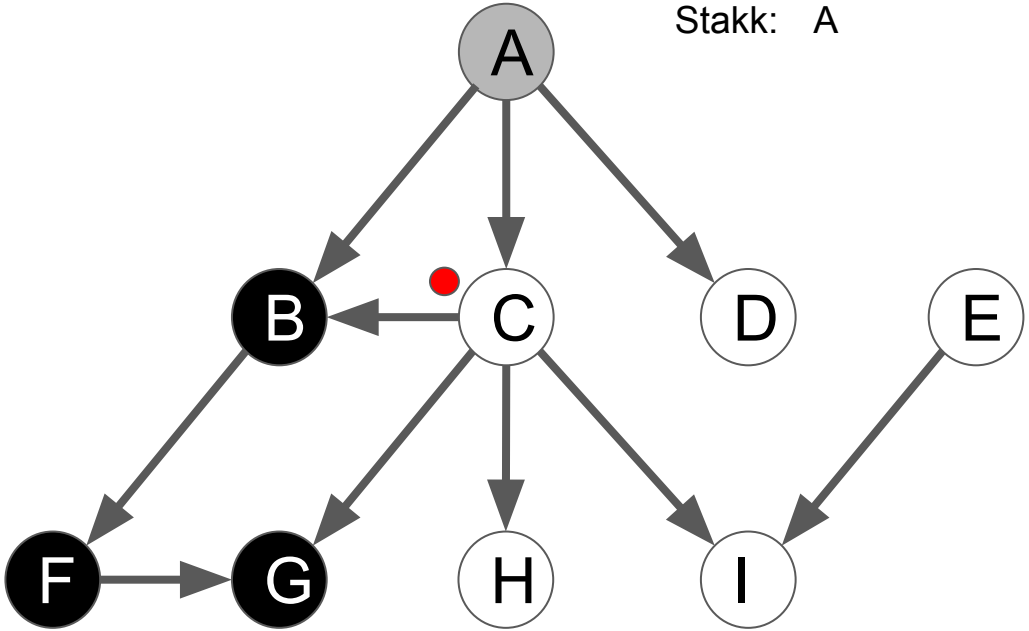
Stakk:



Grå: A B F G

Svart: G F B

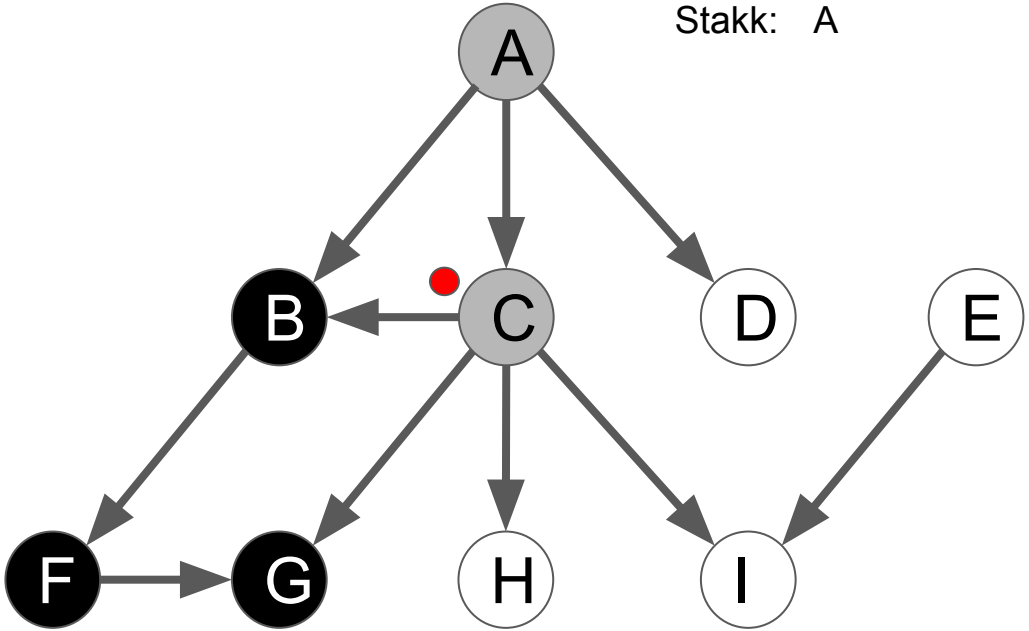
Stakk: A



Grå: A B F G C

Svart: G F B

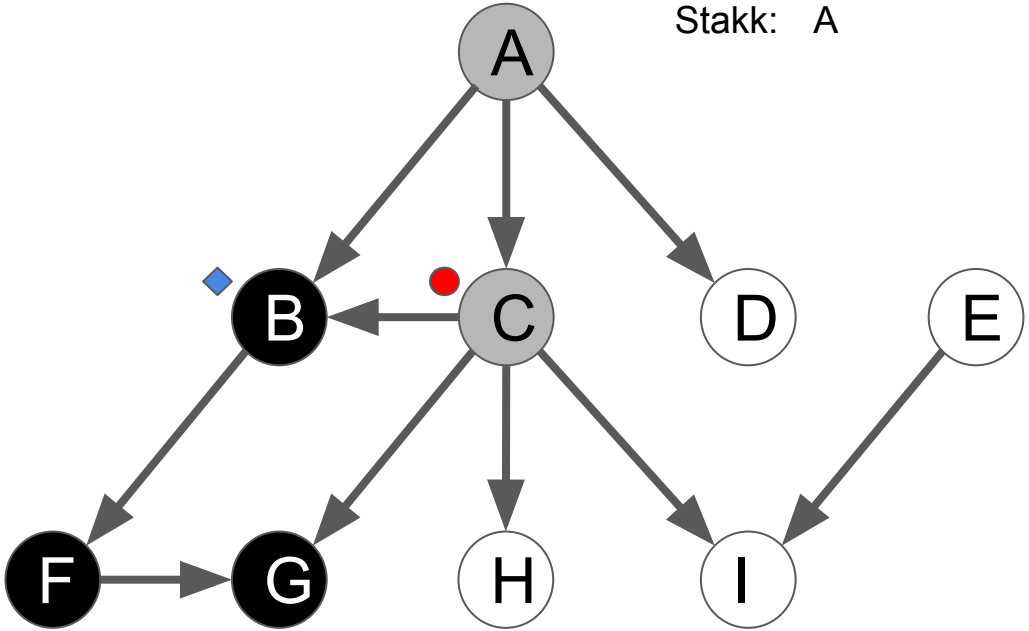
Stakk: A



Grå: A B F G C

Svart: G F B

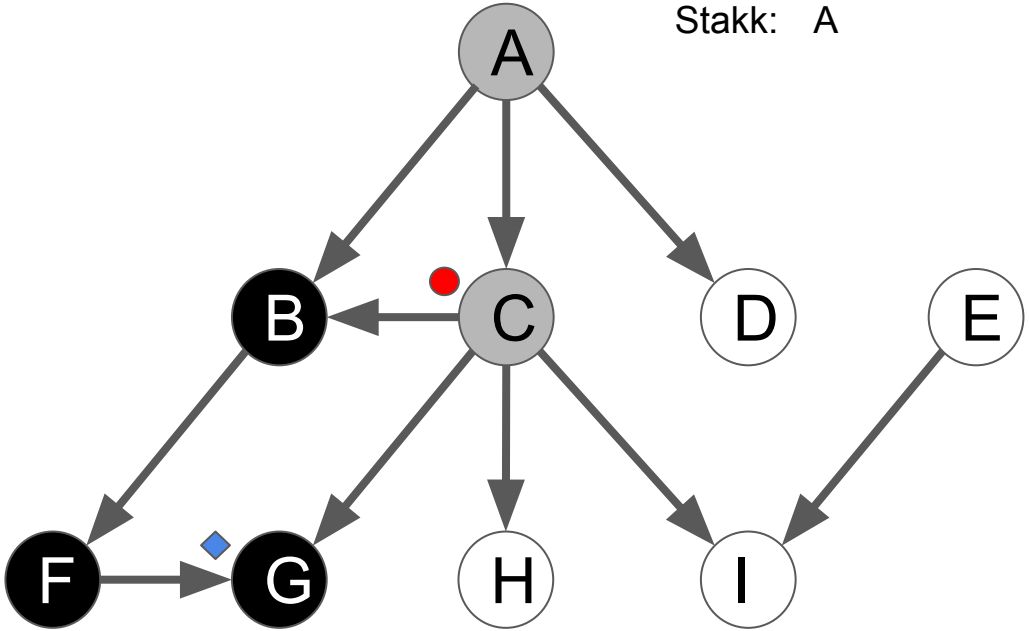
Stakk: A



Grå: A B F G C

Svart: G F B

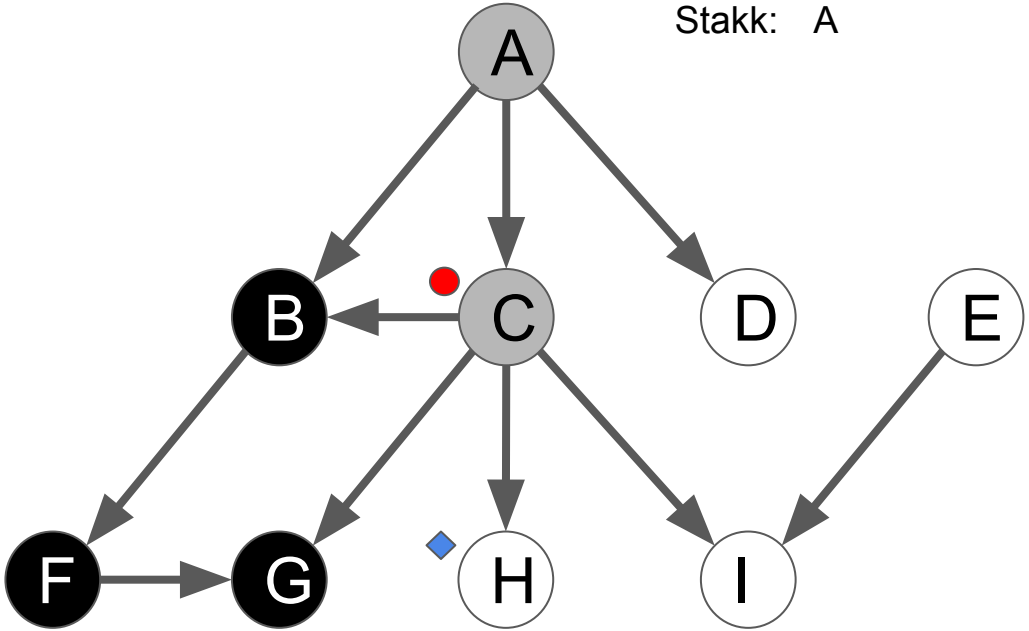
Stakk: A



Grå: A B F G C

Svart: G F B

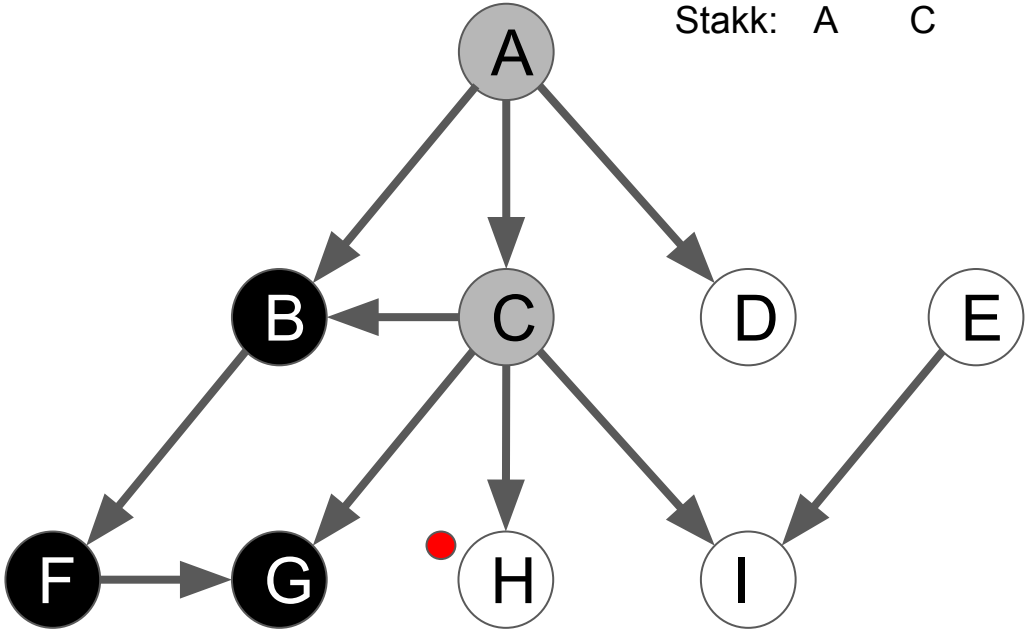
Stakk: A



Grå: A B F G C

Svart: G F B

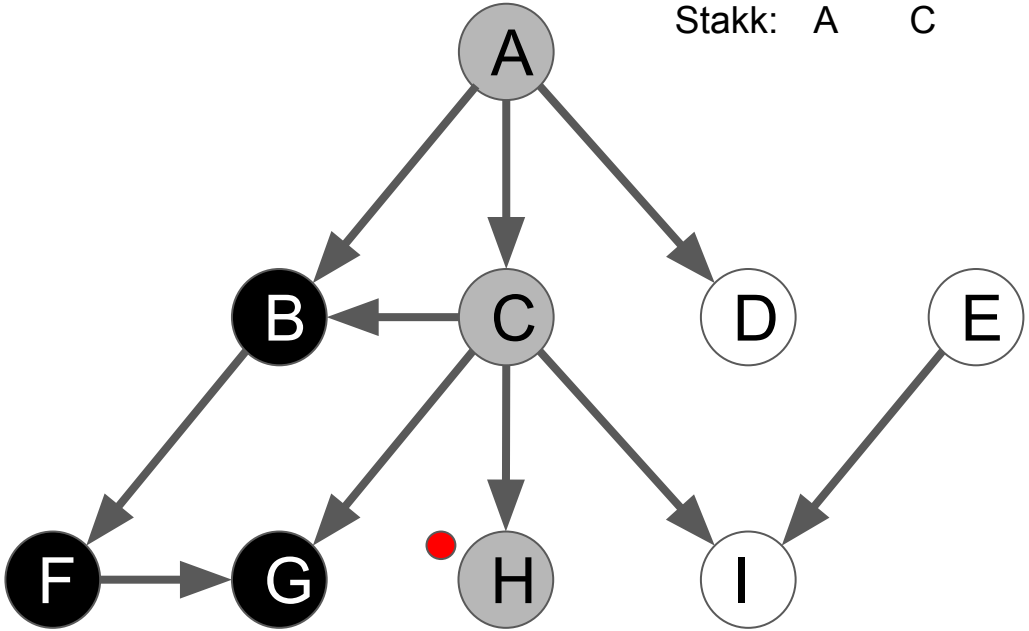
Stakk: A C



Grå: A B F G C H

Svart: G F B

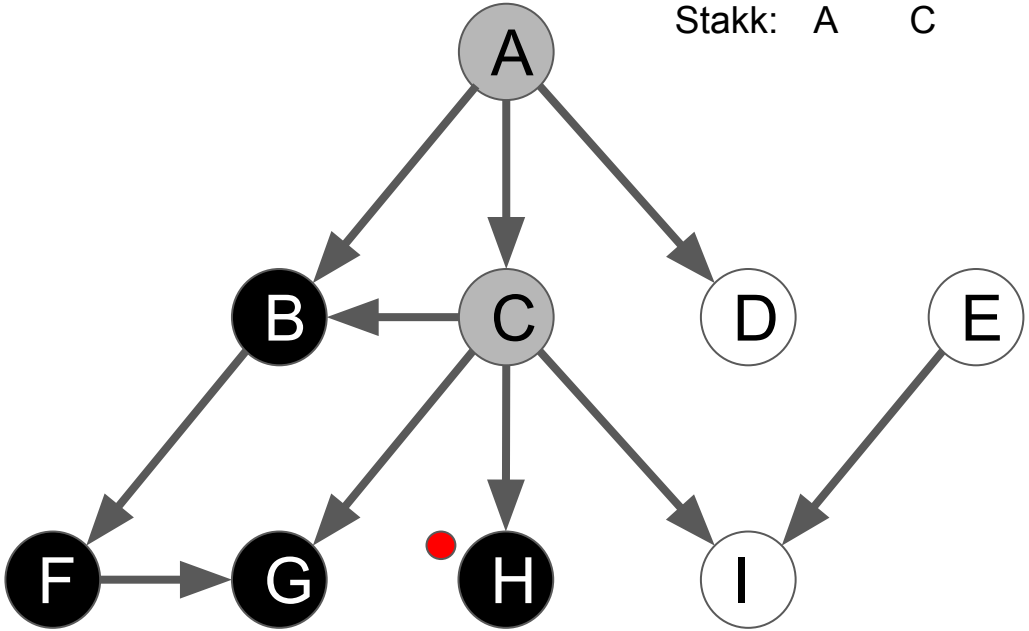
Stakk: A C



Grå: A B F G C H

Svart: G F B H

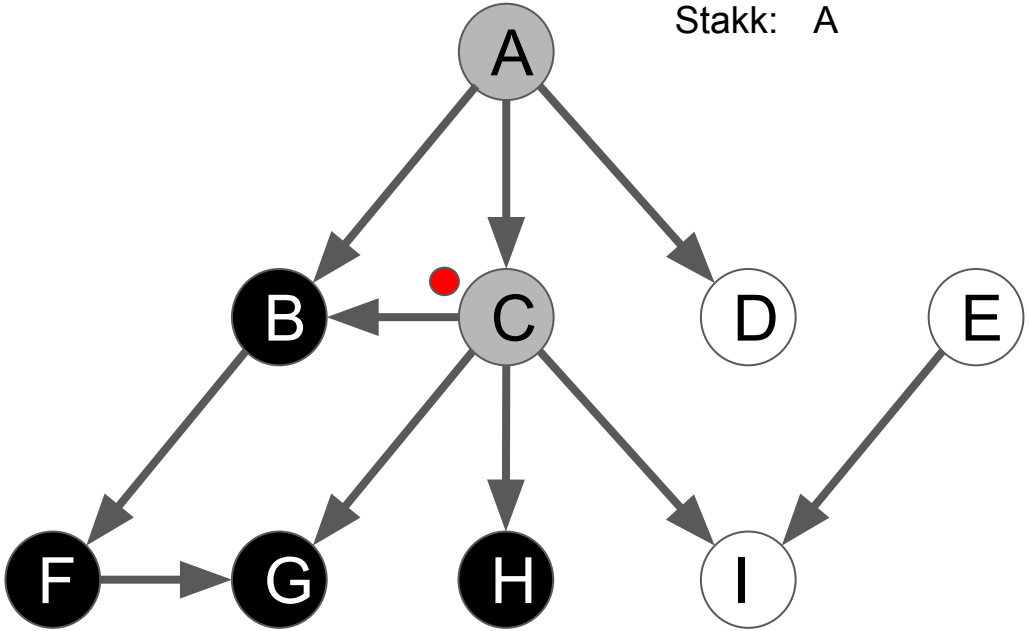
Stakk: A C



Grå: A B F G C H

Svart: G F B H

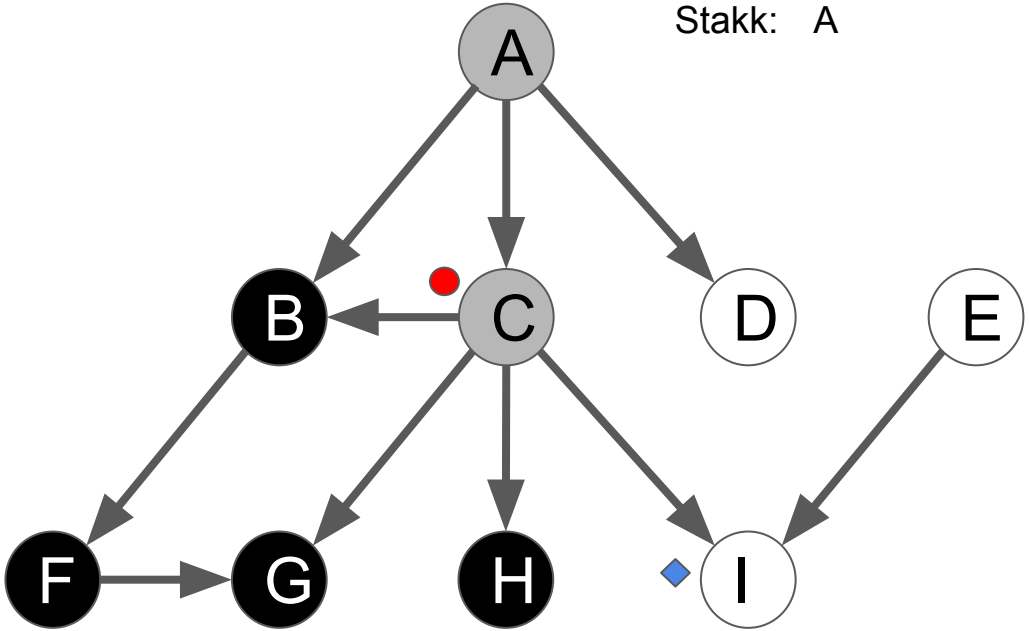
Stakk: A



Grå: A B F G C H

Svart: G F B H

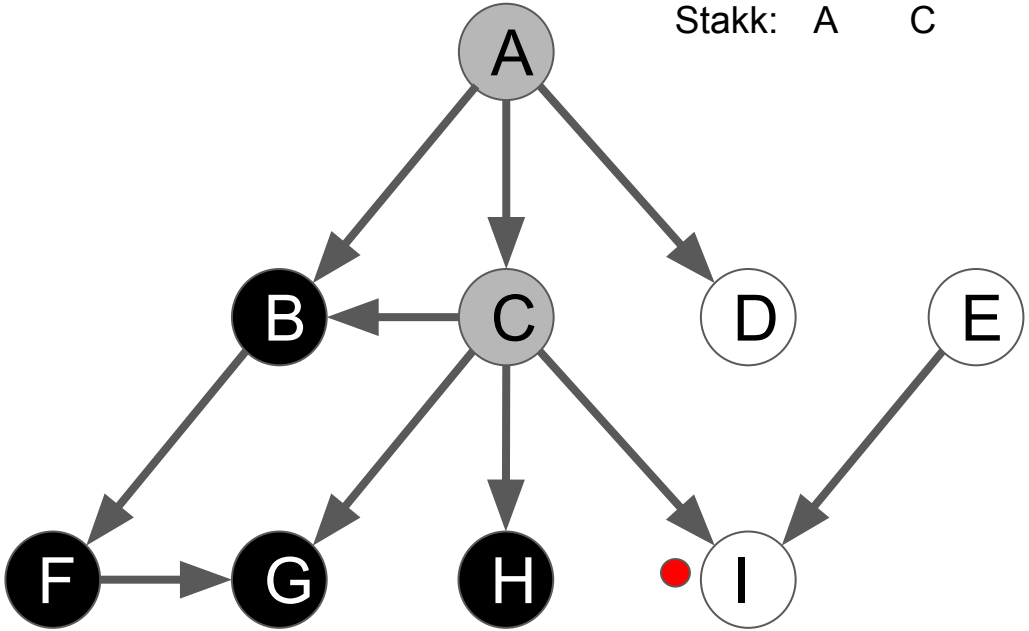
Stakk: A



Grå: A B F G C H

Svart: G F B H

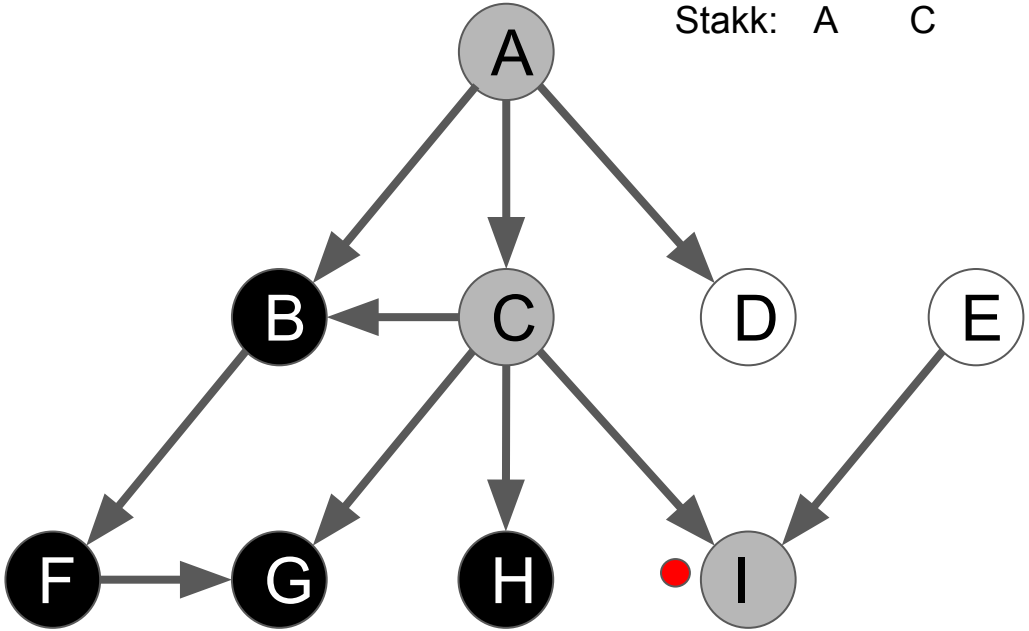
Stakk: A C



Grå: A B F G C H I

Svart: G F B H

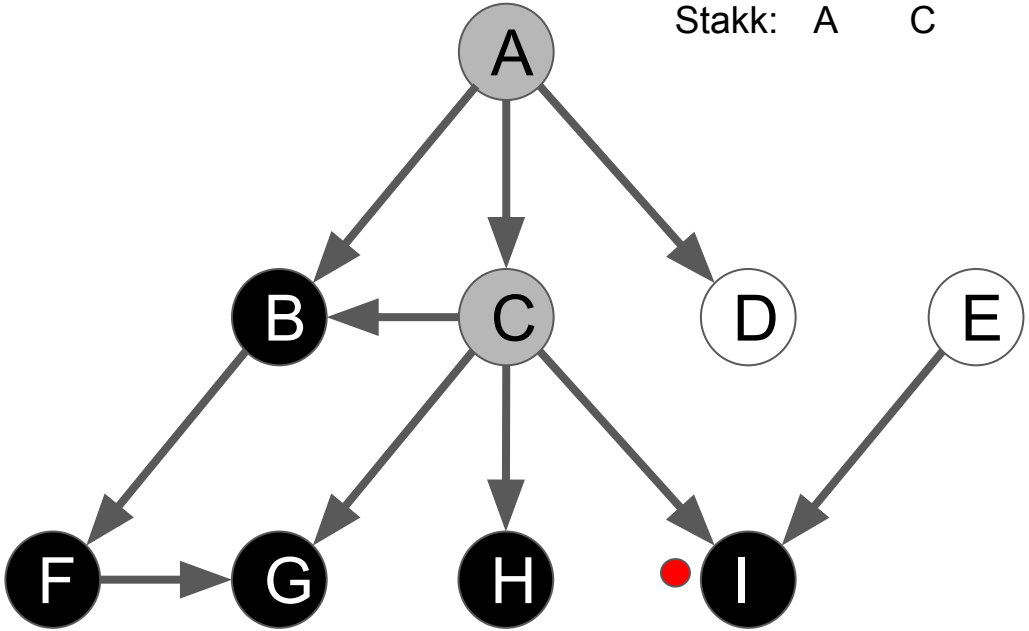
Stakk: A C



Grå: A B F G C H I

Svart: G F B H I

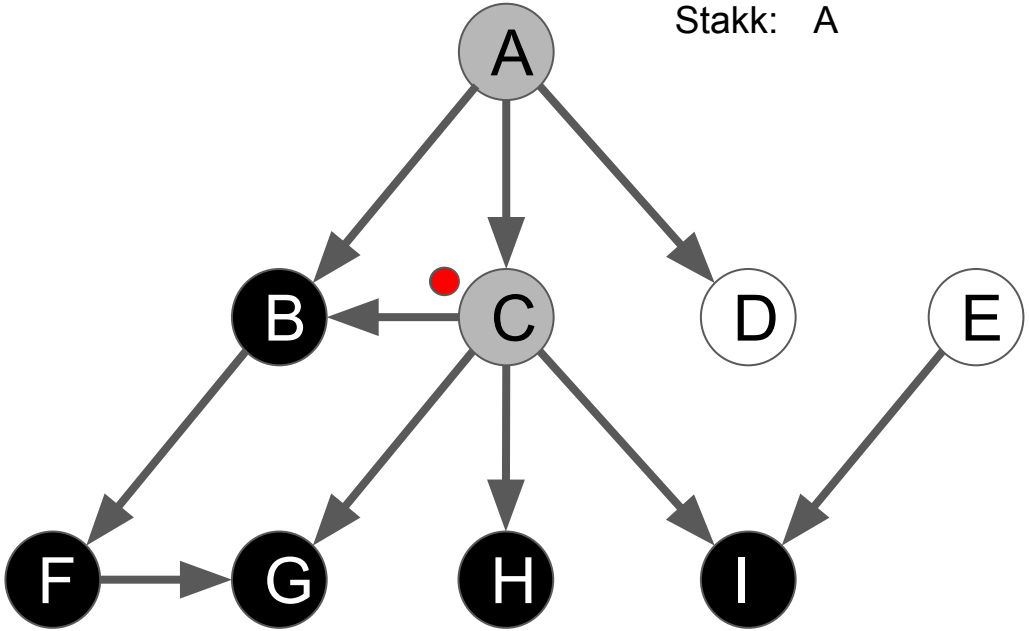
Stakk: A C



Grå: A B F G C H I

Svart: G F B H I

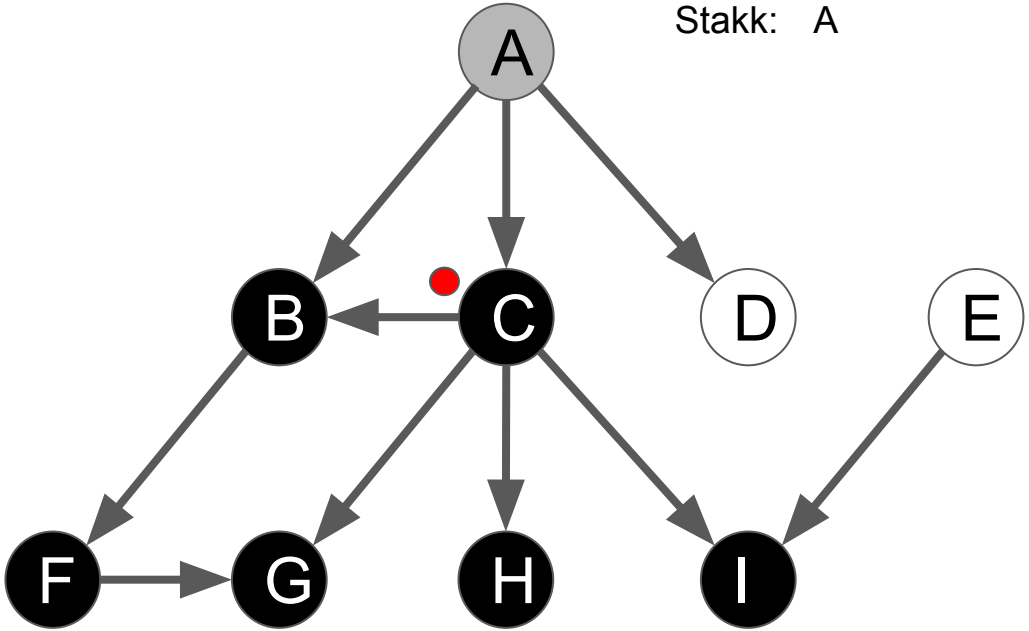
Stakk: A



Grå: A B F G C H I

Svart: G F B H I C

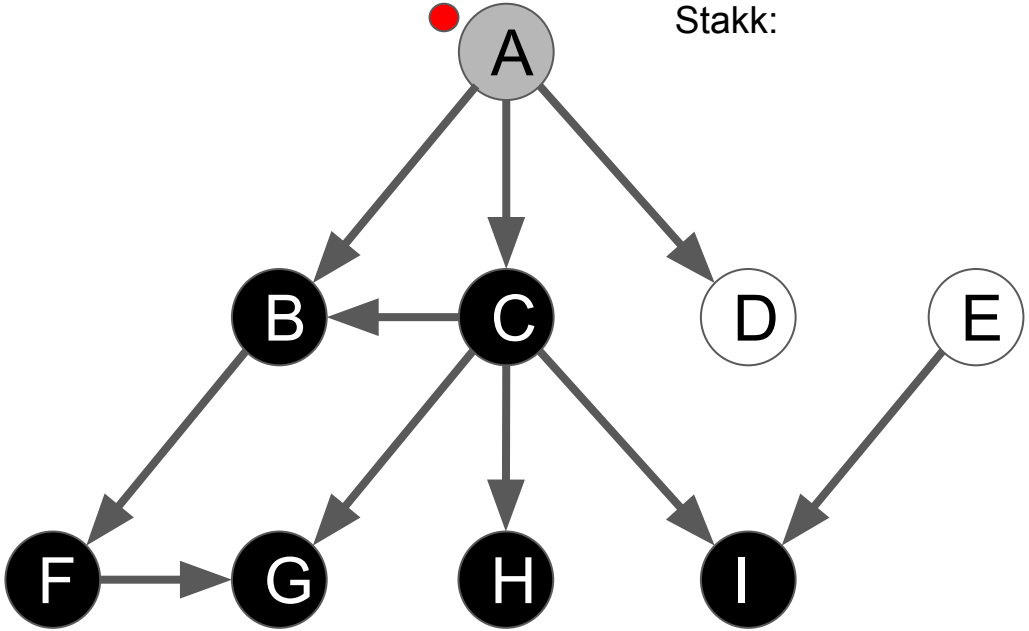
Stakk: A



Grå: A B F G C H I

Svart: G F B H I C

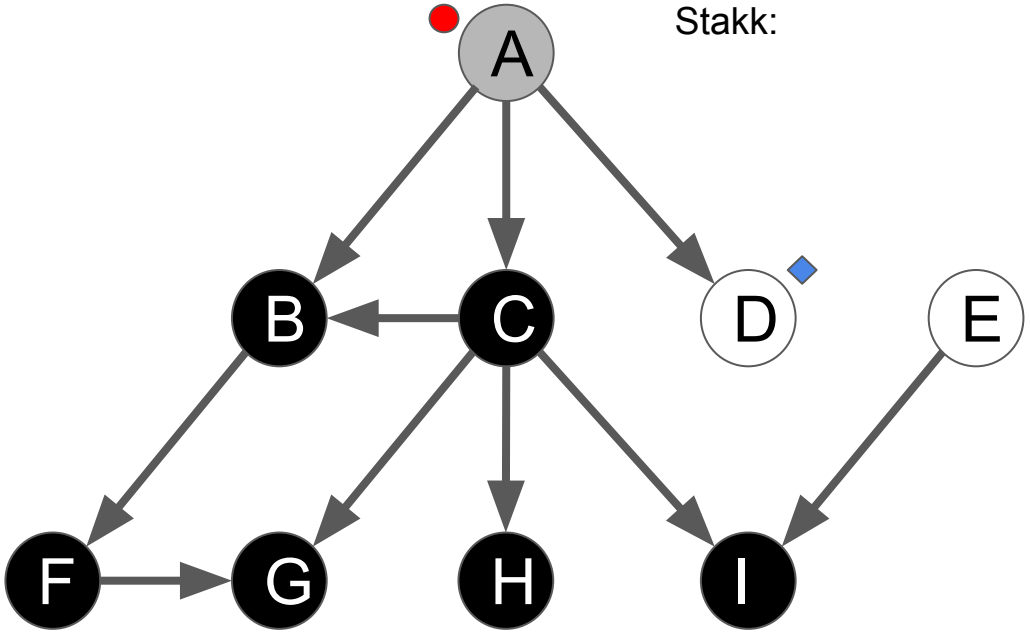
Stakk:



Grå: A B F G C H I

Svart: G F B H I C

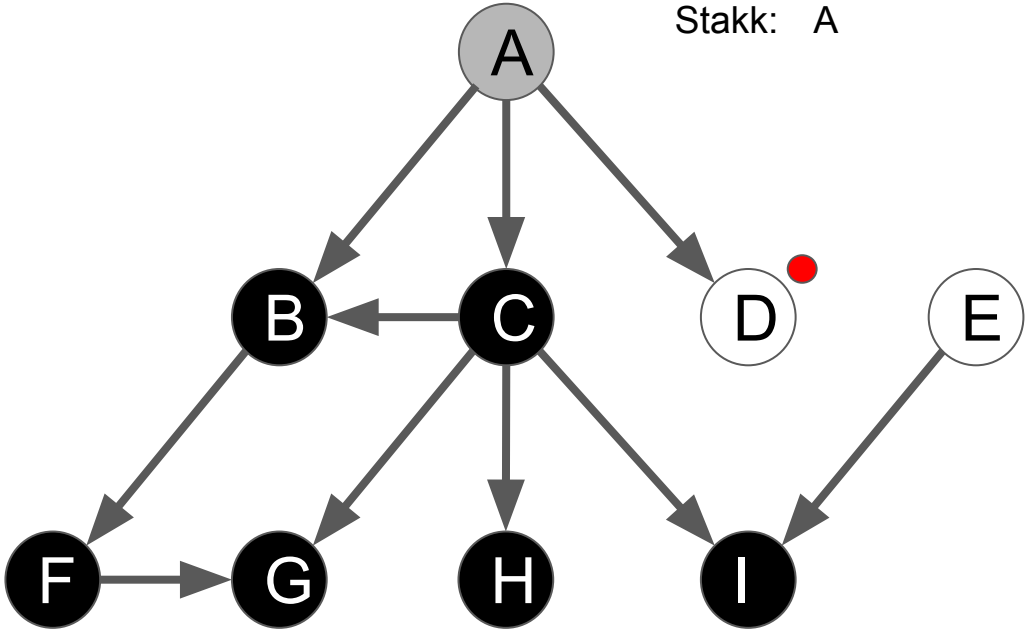
Stakk:



Grå: A B F G C H I

Svart: G F B H I C

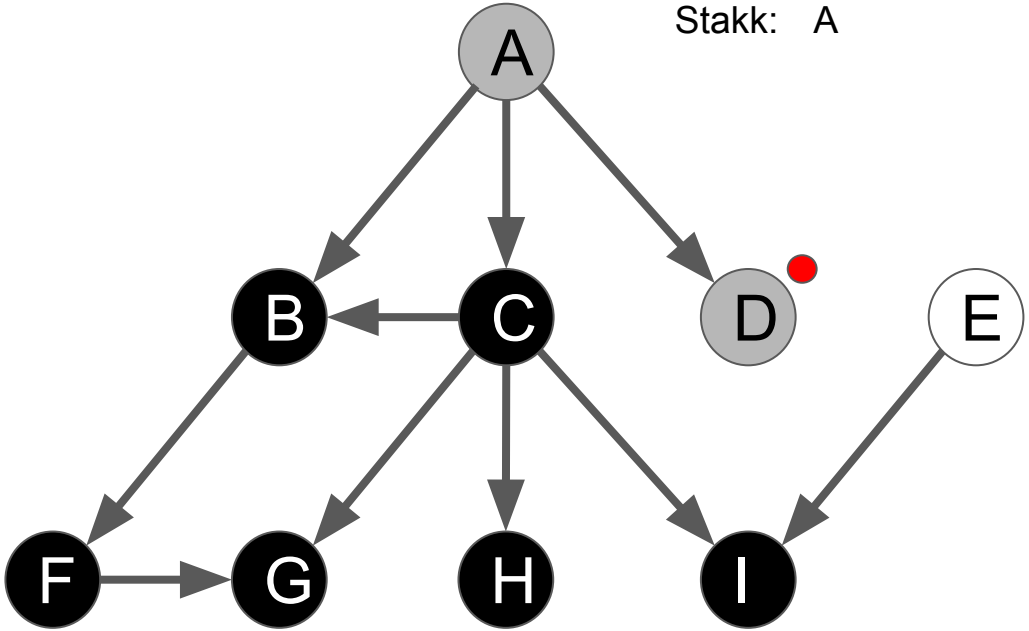
Stakk: A



Grå: A B F G C H I D

Svart: G F B H I C

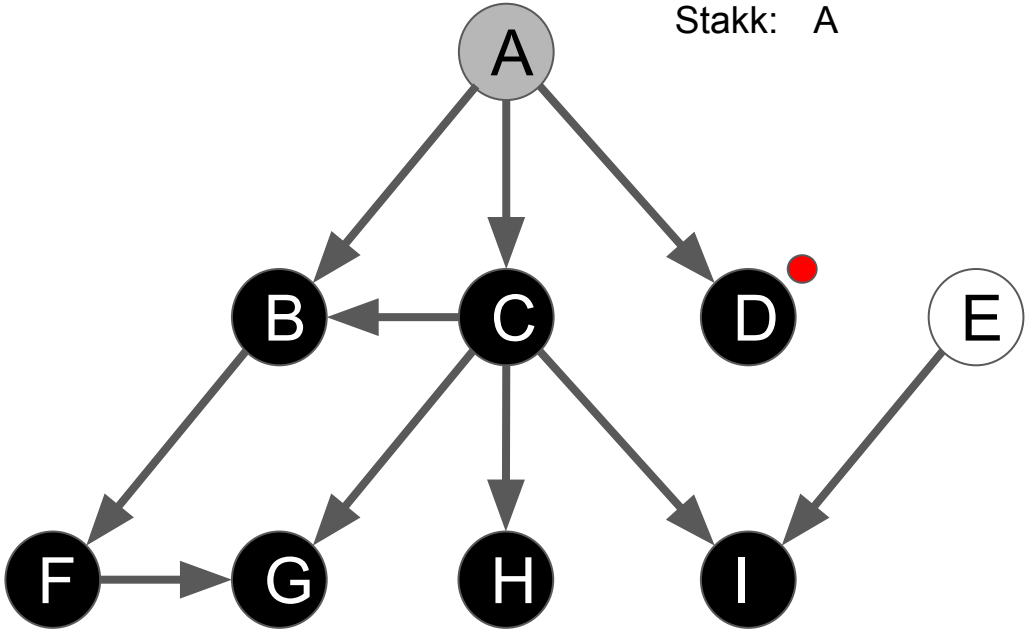
Stakk: A



Grå: A B F G C H I D

Svart: G F B H I C D

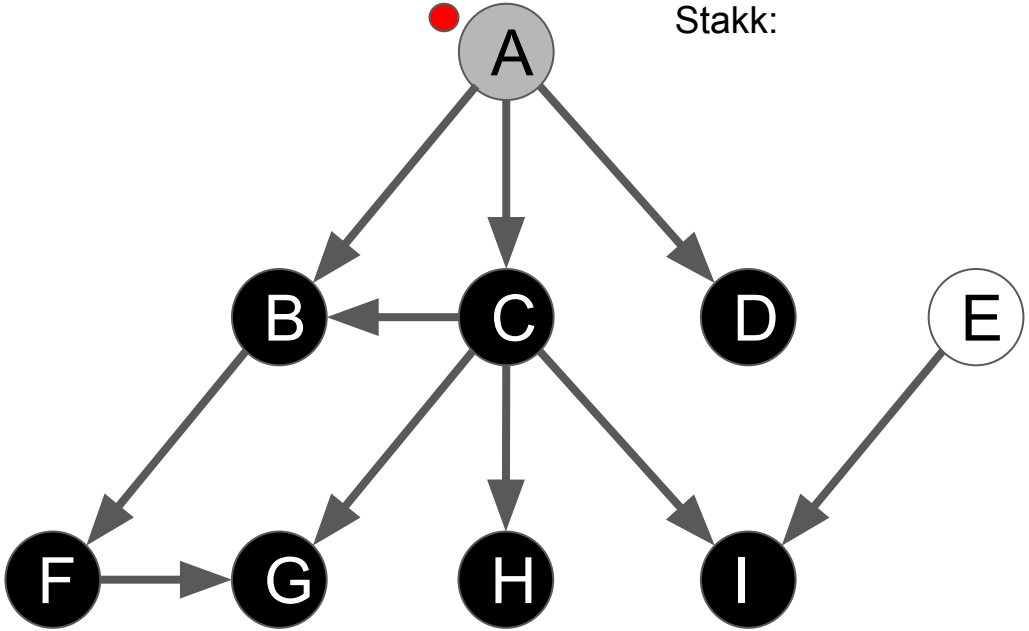
Stakk: A



Grå: A B F G C H I D

Svart: G F B H I C D

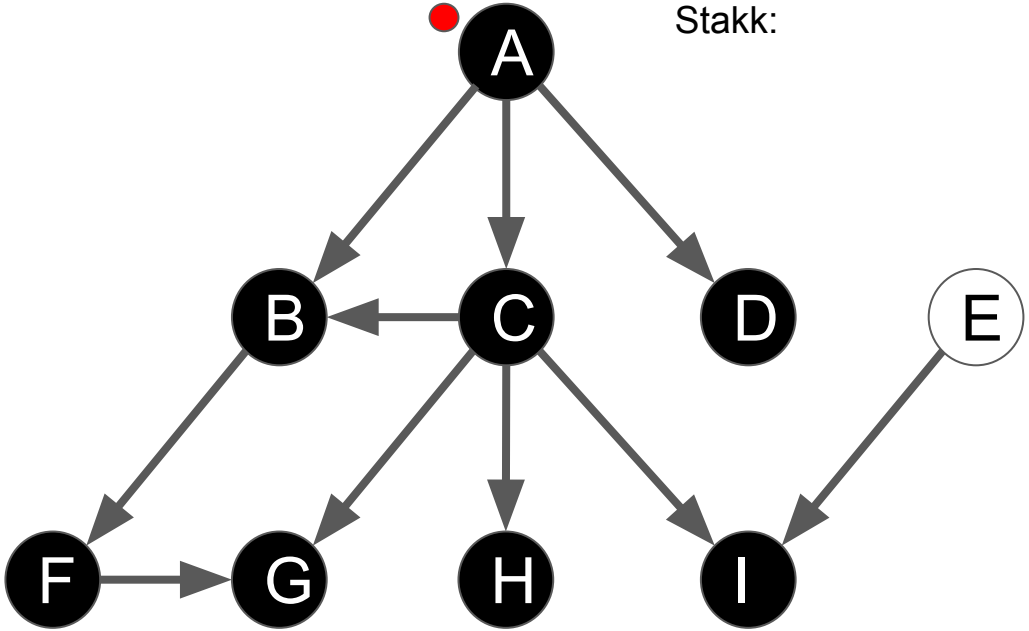
Stakk:



Grå: A B F G C H I D

Svart: G F B H I C D A

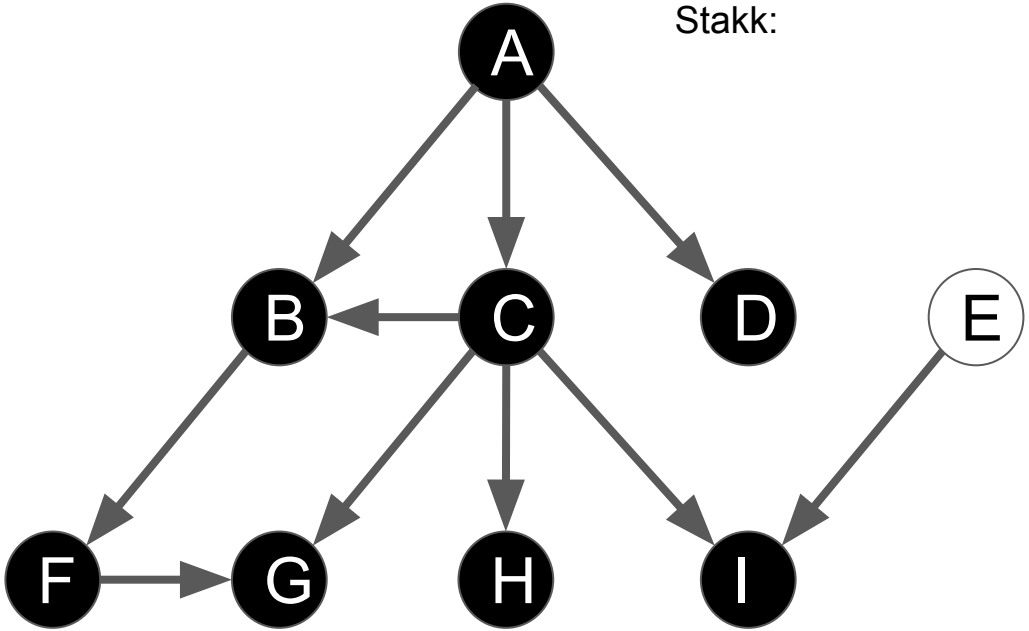
Stakk:



Grå: A B F G C H I D

Svart: G F B H I C D A

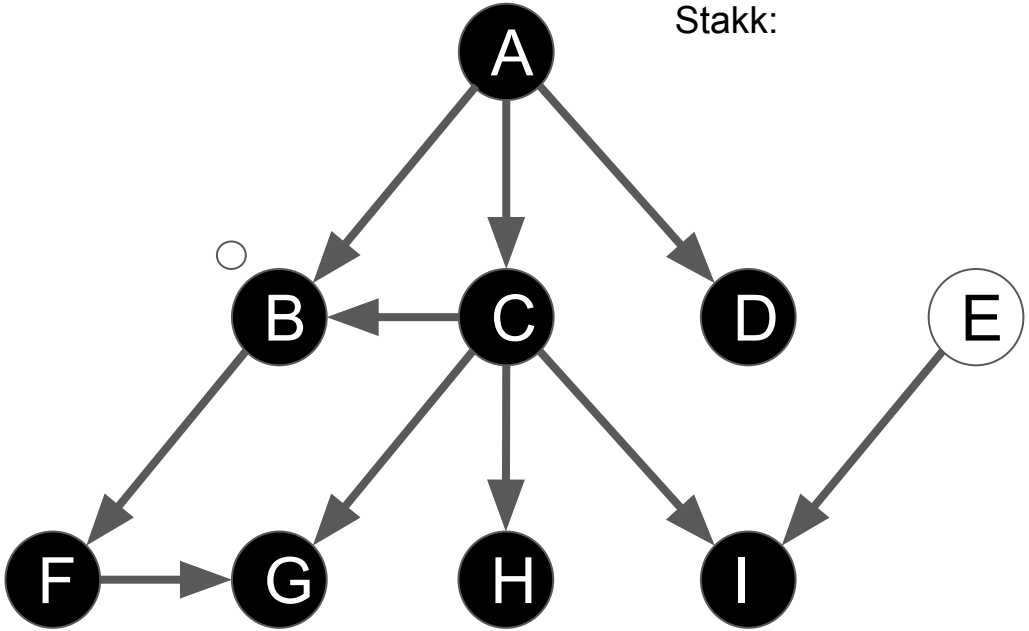
Stakk:



Grå: A B F G C H I D

Svart: G F B H I C D A

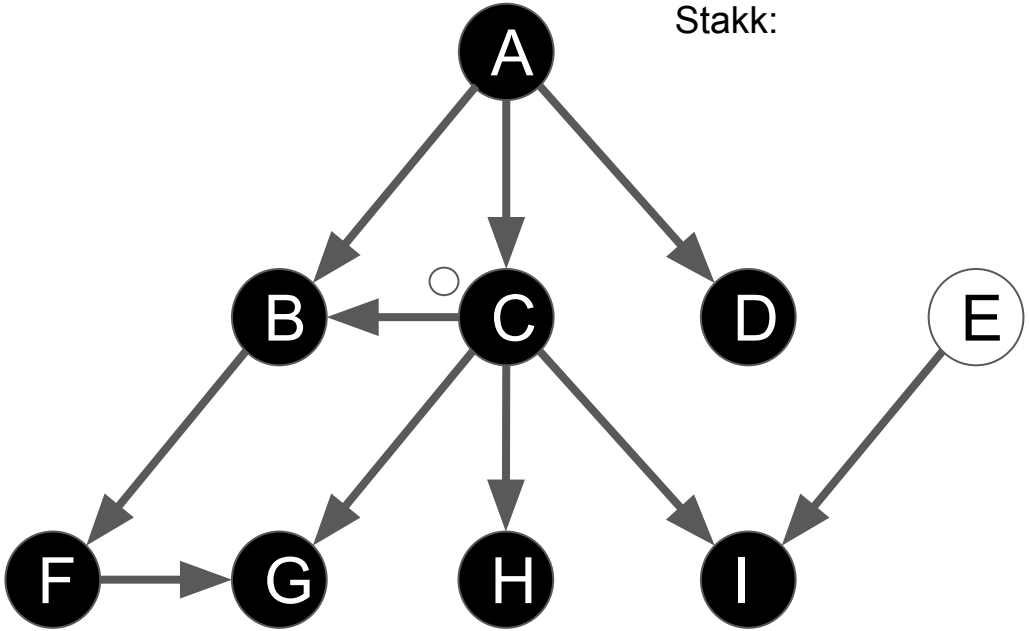
Stakk:



Grå: A B F G C H I D

Svart: G F B H I C D A

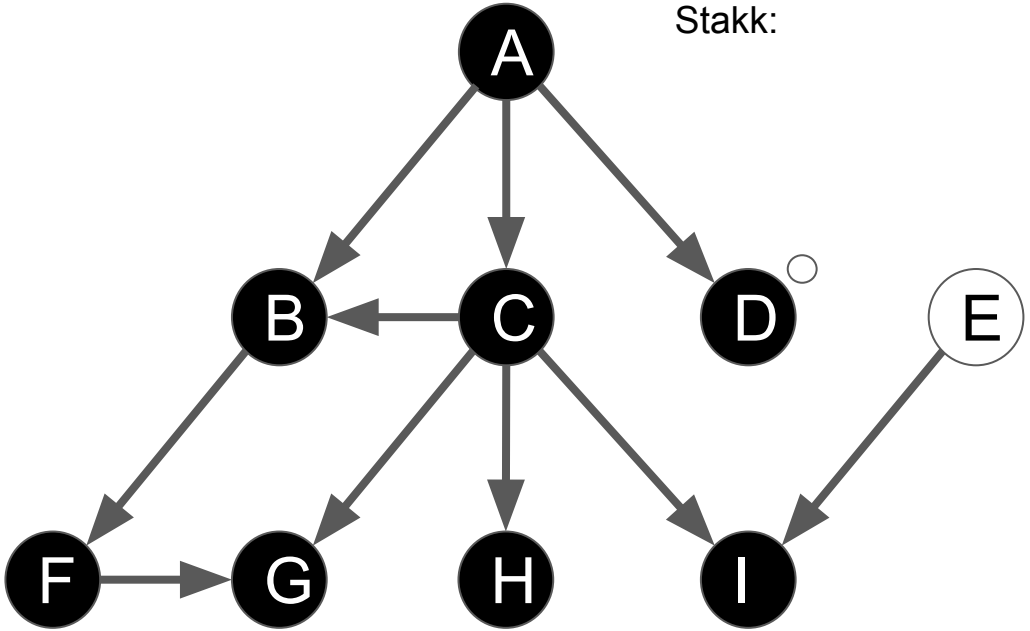
Stakk:



Grå: A B F G C H I D

Svart: G F B H I C D A

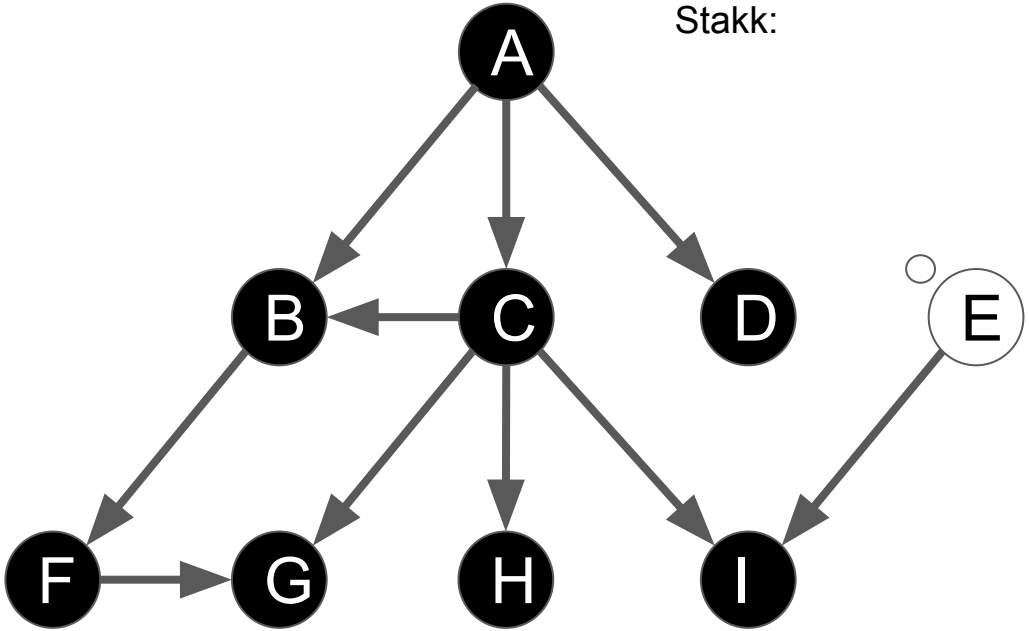
Stakk:



Grå: A B F G C H I D

Svart: G F B H I C D A

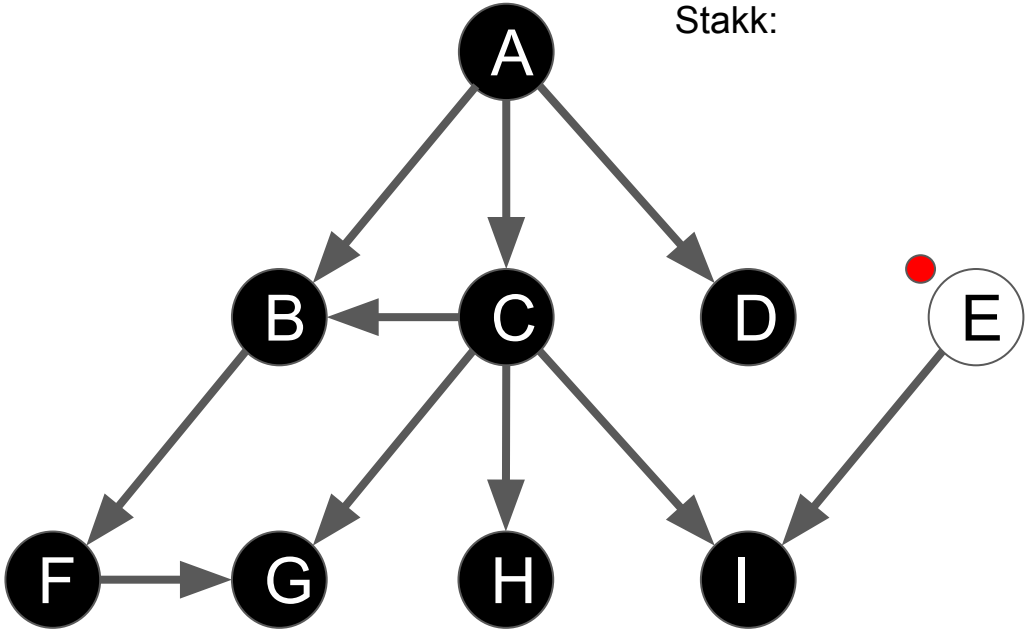
Stakk:



Grå: A B F G C H I D

Svart: G F B H I C D A

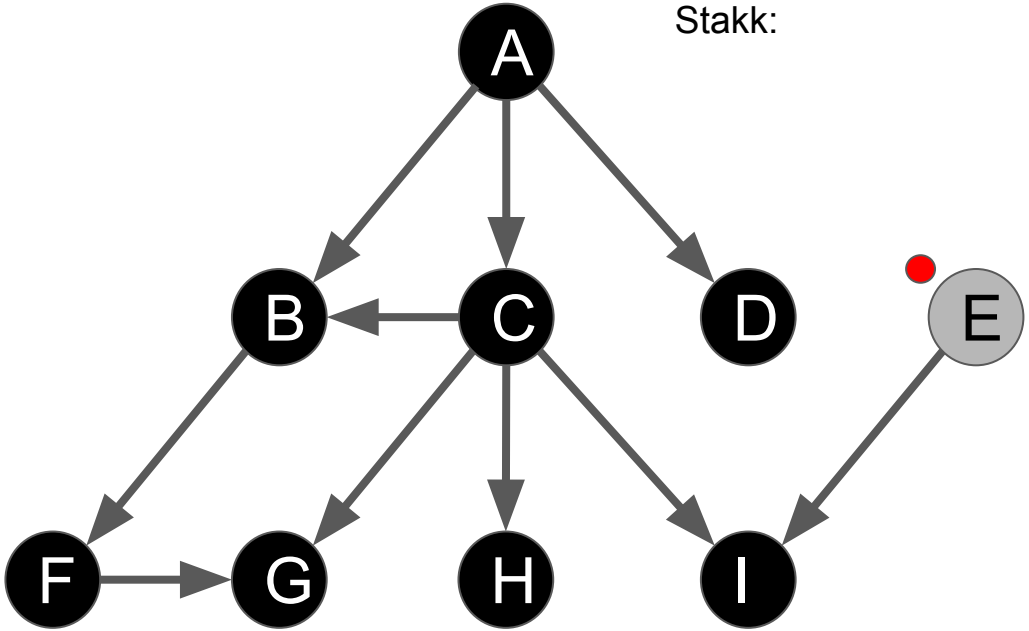
Stakk:



Grå: A B F G C H I D E

Svart: G F B H I C D A

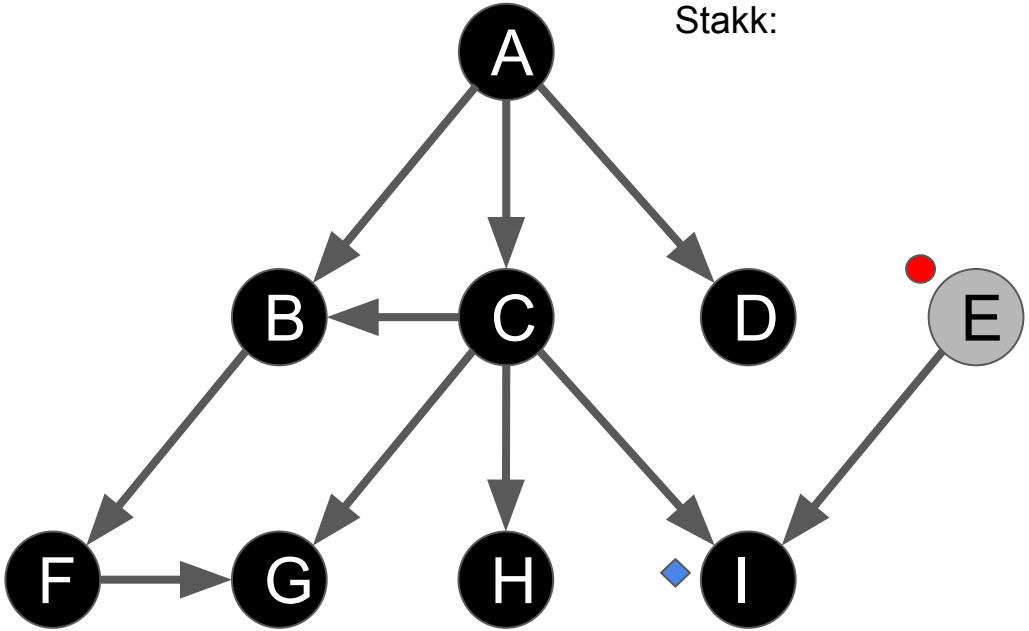
Stakk:



Grå: A B F G C H I D E

Svart: G F B H I C D A

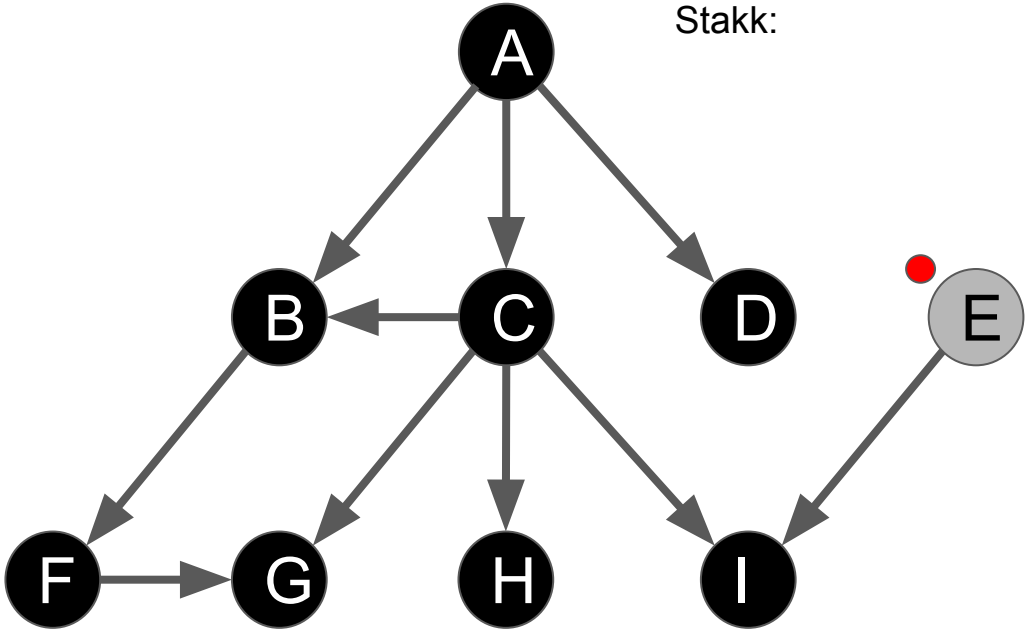
Stakk:



Grå: A B F G C H I D E

Svart: G F B H I C D A

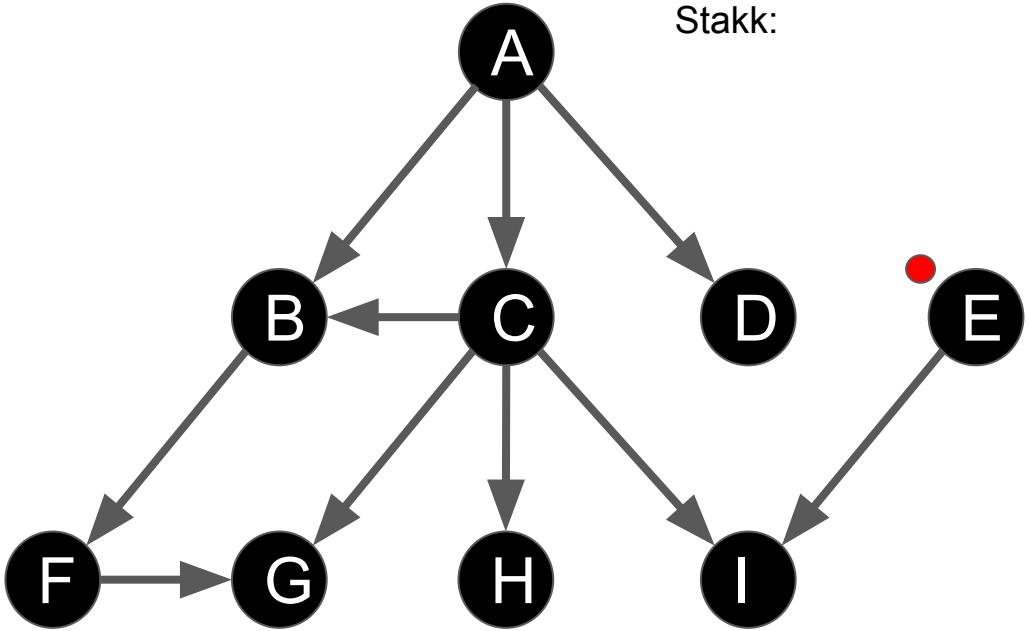
Stakk:



Grå: A B F G C H I D E

Svart: G F B H I C D A E

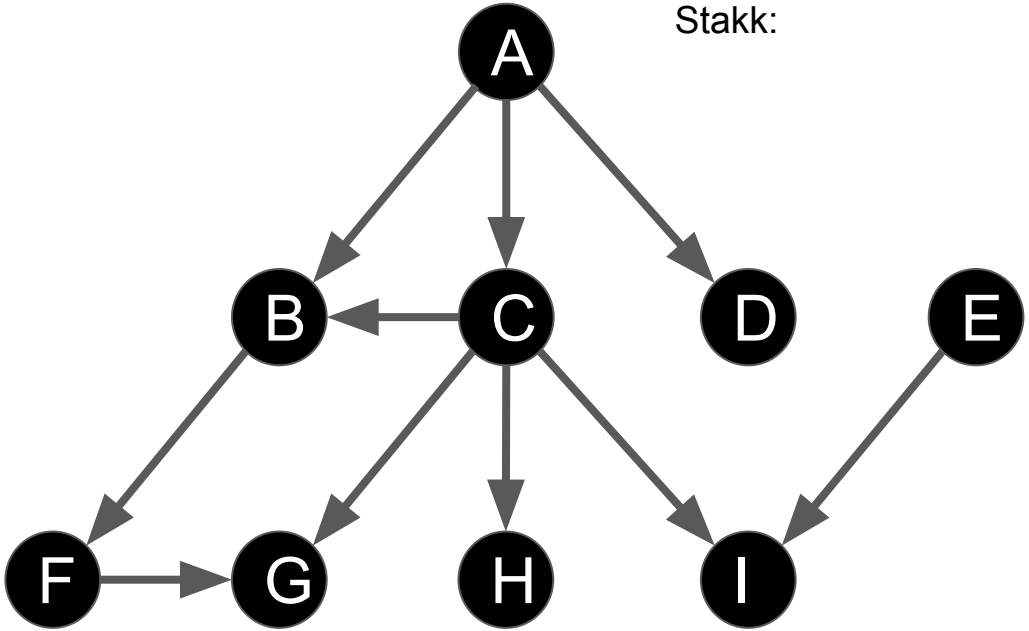
Stakk:



Grå: A B F G C H I D E

Svart: G F B H I C D A E

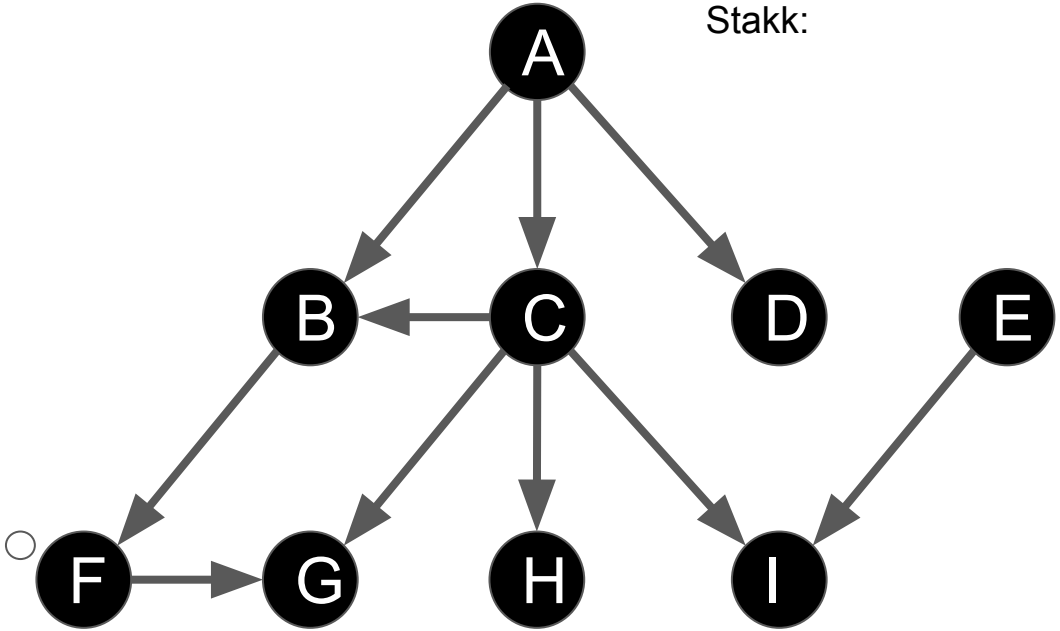
Stakk:



Grå: A B F G C H I D E

Svart: G F B H I C D A E

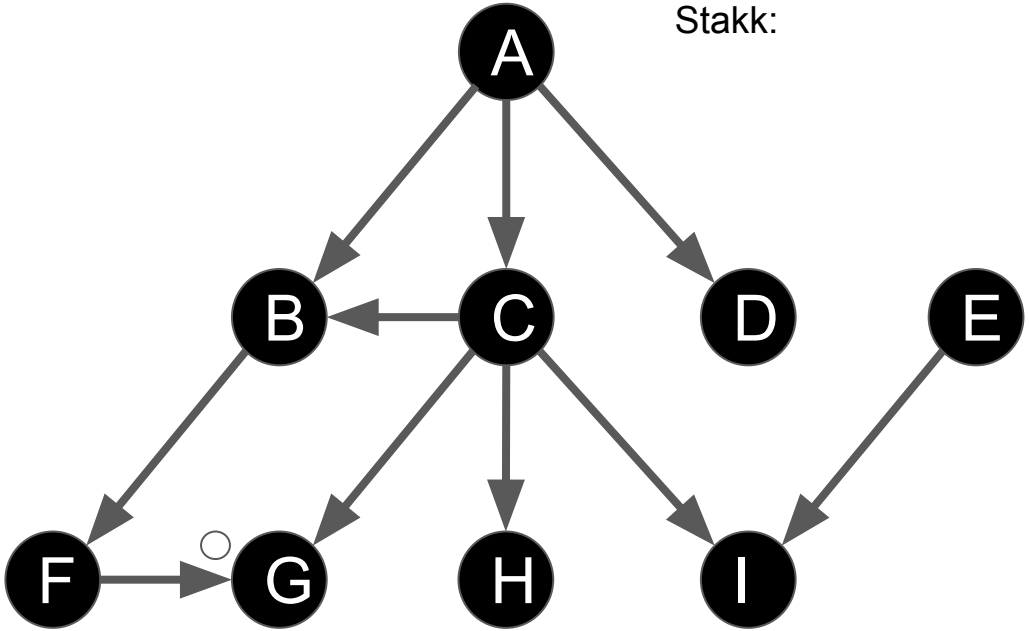
Stakk:



Grå: A B F G C H I D E

Svart: G F B H I C D A E

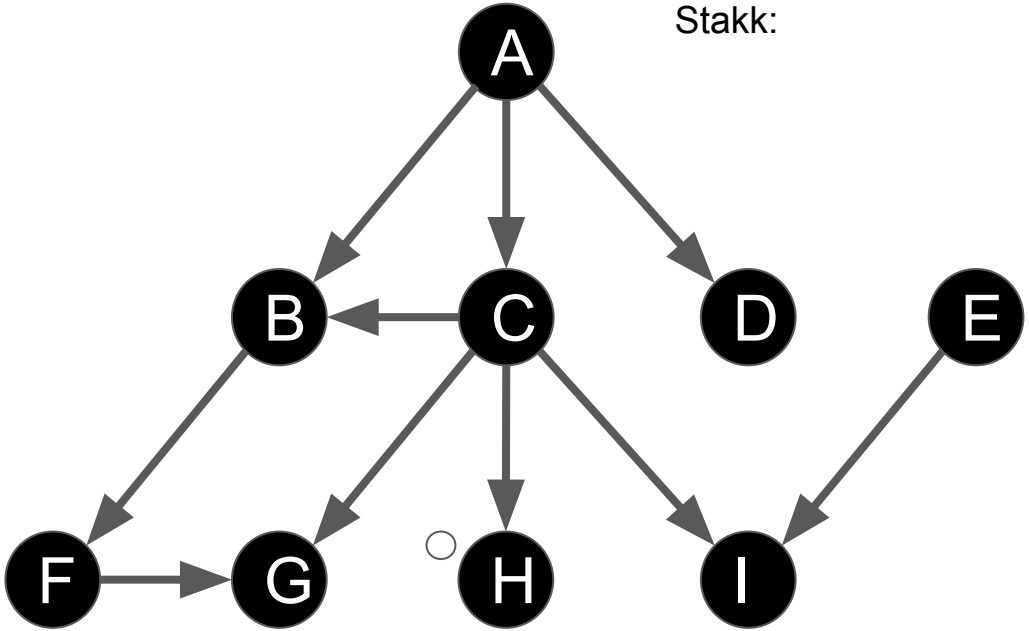
Stakk:



Grå: A B F G C H I D E

Svart: G F B H I C D A E

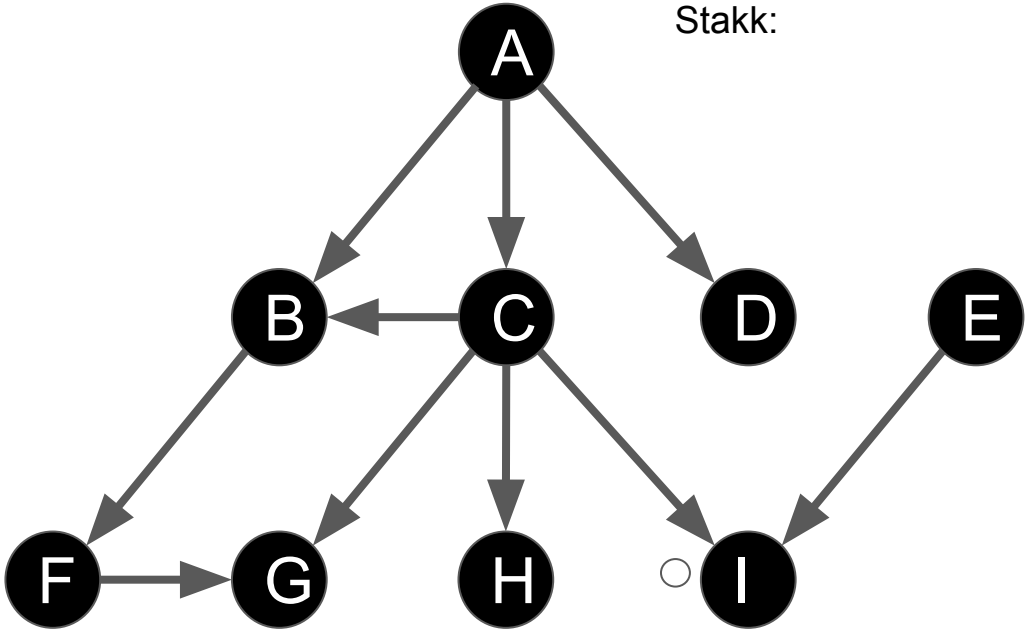
Stakk:



Grå: A B F G C H I D E

Svart: G F B H I C D A E

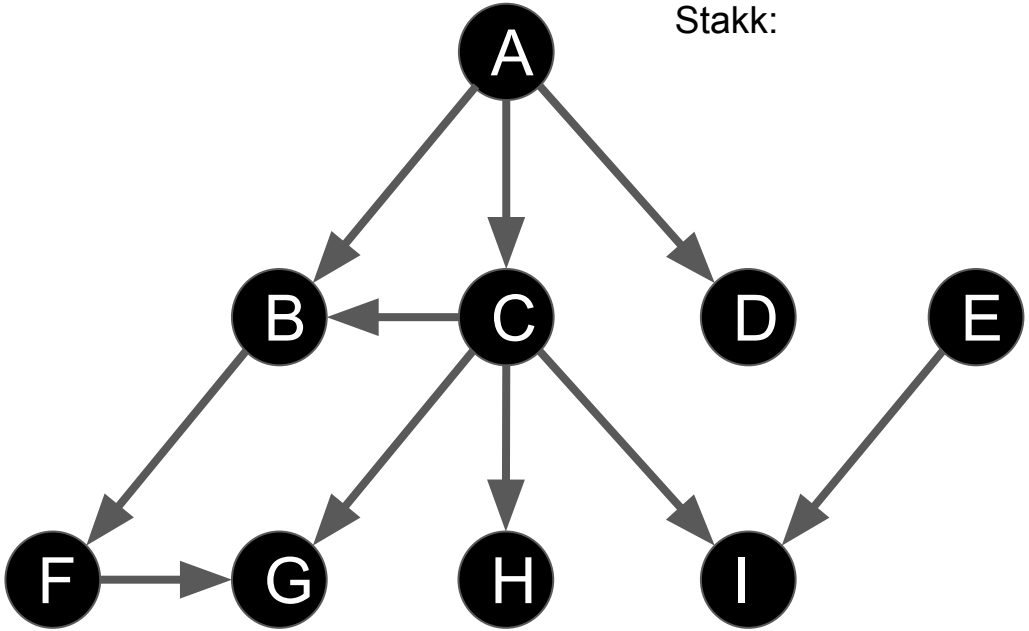
Stakk:



Grå: A B F G C H I D E

Svart: G F B H I C D A E

Stakk:



Grå: A B F G C H I D E

Svart: **G** **F** **B** **H** I C D A E