

MAT1030 – Diskret Matematikk

Forelesning 23: Grafteori

Dag Normann

Matematisk Institutt, Universitetet i Oslo

20. april 2010

(Sist oppdatert: 2010-04-20 14:14)



Grafteori

Repetisjon og mer motivasjon

Repetisjon og mer motivasjon

- Først litt repetisjon

Repetisjon og mer motivasjon

- Først litt repetisjon
- En graf består av *noder* og *kanter*

Repetisjon og mer motivasjon

- Først litt repetisjon
- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.

Repetisjon og mer motivasjon

- Først litt repetisjon
- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om

Repetisjon og mer motivasjon

- Først litt repetisjon
- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer

Repetisjon og mer motivasjon

- Først litt repetisjon
- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer

Repetisjon og mer motivasjon

- Først litt repetisjon
- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer, *løkker*

Repetisjon og mer motivasjon

- Først litt repetisjon
- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer, *løkker*, *parallelle* kanter

Repetisjon og mer motivasjon

- Først litt repetisjon
- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer, *løkker*, *parallelle* kanter, *enkle* grafer

Repetisjon og mer motivasjon

- Først litt repetisjon
- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer, *løkker*, *parallelle* kanter, *enkle* grafer og *komplette* grafer.

Repetisjon og mer motivasjon

- Først litt repetisjon
- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer, *løkker*, *parallelle* kanter, *enkke* grafer og *komplette* grafer.
- Hver node har en *grad*.

Repetisjon og mer motivasjon

- Først litt repetisjon
- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer, *løkker*, *parallelle* kanter, *enkle* grafer og *komplette* grafer.
- Hver node har en *grad*.
 - Summen av gradene til alle nodene i en graf er lik 2 ganger antallet kanter.

Repetisjon og mer motivasjon

- Først litt repetisjon
- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer, *løkker*, *parallelle* kanter, *enkle* grafer og *komplette* grafer.
- Hver node har en *grad*.
 - Summen av gradene til alle nodene i en graf er lik 2 ganger antallet kanter.
 - Håndhilselemmaet: Det er alltid et partall antall noder av odde grad i en graf.

Repetisjon og mer motivasjon

- Først litt repetisjon
- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer, *løkker*, *parallelle* kanter, *enkle* grafer og *komplette* grafer.
- Hver node har en *grad*.
 - Summen av gradene til alle nodene i en graf er lik 2 ganger antallet kanter.
 - Håndhilselemmaet: Det er alltid et partall antall noder av odde grad i en graf.
- Vi skal kjenne til *komplementet* av en graf og *matriserepresentasjoner*.

Repetisjon og mer motivasjon

Repetisjon og mer motivasjon

- Grafer kan brukes til å representere omtrent alt som fins av relasjoner.

Repetisjon og mer motivasjon

- Grafer kan brukes til å representere omtrent alt som fins av relasjoner.
- Mange algoritmer/egenskaper kan forstås bedre ved å bruke grafer.

Repetisjon og mer motivasjon

- Grafer kan brukes til å representere omtrent alt som fins av relasjoner.
- Mange algoritmer/egenskaper kan forstås bedre ved å bruke grafer.
- Ofte er løsningen å kunne identifisere et problem som et *grafteoretisk* problem.

Grafisomorfier

Grafisomorfier

- Vi skal nå møte begrepet *isomorfi* for første gang i dette kurset.

Grafisomorfier

- Vi skal nå møte begrepet *isomorfi* for første gang i dette kurset.
- Isomorfibegrepet er veldig generelt og brukes overalt i matematikk.

Grafisomorfier

- Vi skal nå møte begrepet *isomorfi* for første gang i dette kurset.
- Isomorfibegrepet er veldig generelt og brukes overalt i matematikk.
- Ordet kommer fra gresk og betyr formlik (iso = lik, morf = form).

Grafisomorfier

- Vi skal nå møte begrepet *isomorfi* for første gang i dette kurset.
- Isomorfibegrepet er veldig generelt og brukes overalt i matematikk.
- Ordet kommer fra gresk og betyr formlik (iso = lik, morf = form).
- Isomorfe objekter skal ha samme *form*, men kan ha ulikt innhold.

Grafisomorfier

- Vi skal nå møte begrepet *isomorfi* for første gang i dette kurset.
- Isomorfibegrepet er veldig generelt og brukes overalt i matematikk.
- Ordet kommer fra gresk og betyr formlik (iso = lik, morf = form).
- Isomorfe objekter skal ha samme *form*, men kan ha ulikt innhold.
- En isomorfi er en funksjon som er injektiv og surjektiv

Grafisomorfier

- Vi skal nå møte begrepet *isomorfi* for første gang i dette kurset.
- Isomorfibegrepet er veldig generelt og brukes overalt i matematikk.
- Ordet kommer fra gresk og betyr formlik (iso = lik, morf = form).
- Isomorfe objekter skal ha samme *form*, men kan ha ulikt innhold.
- En isomorfi er en funksjon som er injektiv og surjektiv – det er altså en en-til-en-korrespondanse

Grafisomorfier

- Vi skal nå møte begrepet *isomorfi* for første gang i dette kurset.
- Isomorfibegrepet er veldig generelt og brukes overalt i matematikk.
- Ordet kommer fra gresk og betyr formlik (iso = lik, morf = form).
- Isomorfe objekter skal ha samme *form*, men kan ha ulikt innhold.
- En isomorfi er en funksjon som er injektiv og surjektiv – det er altså en en-til-en-korrespondanse – og som bevarer bestemte egenskaper.

Grafisomorfier

- Vi skal nå møte begrepet *isomorfi* for første gang i dette kurset.
- Isomorfibegrepet er veldig generelt og brukes overalt i matematikk.
- Ordet kommer fra gresk og betyr formlik (iso = lik, morf = form).
- Isomorfe objekter skal ha samme *form*, men kan ha ulikt innhold.
- En isomorfi er en funksjon som er injektiv og surjektiv – det er altså en en-til-en-korrespondanse – og som bevarer bestemte egenskaper.
- Intuitivt, så sier vi at to matematiske objekter er *isomorfe* hvis de er “strukturelt like”.

Grafisomorfier

- Vi skal nå møte begrepet *isomorfi* for første gang i dette kurset.
- Isomorfibegrepet er veldig generelt og brukes overalt i matematikk.
- Ordet kommer fra gresk og betyr formlik (iso = lik, morf = form).
- Isomorfe objekter skal ha samme *form*, men kan ha ulikt innhold.
- En isomorfi er en funksjon som er injektiv og surjektiv – det er altså en en-til-en-korrespondanse – og som bevarer bestemte egenskaper.
- Intuitivt, så sier vi at to matematiske objekter er *isomorfe* hvis de er “strukturelt like”.
- Hvis vi har tegnet opp to grafer og kan komme fra den ene grafen til den andre ved kun å flytte på noder og endre på lengdene på kantene, så er grafene *isomorfe*.

Grafisomorfier

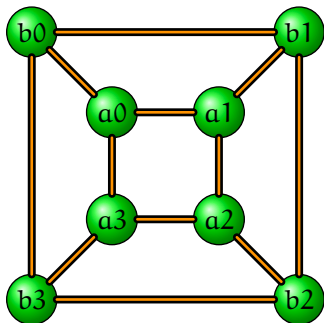
- Vi skal nå møte begrepet *isomorfi* for første gang i dette kurset.
- Isomorfibegrepet er veldig generelt og brukes overalt i matematikk.
- Ordet kommer fra gresk og betyr formlik (iso = lik, morf = form).
- Isomorfe objekter skal ha samme *form*, men kan ha ulikt innhold.
- En isomorfi er en funksjon som er injektiv og surjektiv – det er altså en en-til-en-korrespondanse – og som bevarer bestemte egenskaper.
- Intuitivt, så sier vi at to matematiske objekter er *isomorfe* hvis de er “strukturelt like”.
- Hvis vi har tegnet opp to grafer og kan komme fra den ene grafen til den andre ved kun å flytte på noder og endre på lengdene på kantene, så er grafene *isomorfe*.
- Vi har ikke lov til å legge til eller ta bort noder eller kanter, eller dele kanter eller noder i to...

Grafisomorfier

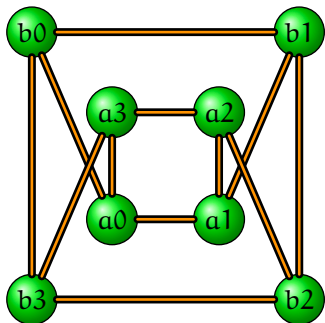
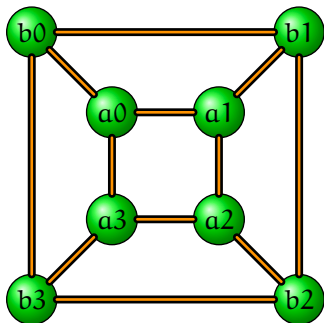
- Vi skal nå møte begrepet *isomorfi* for første gang i dette kurset.
- Isomorfibegrepet er veldig generelt og brukes overalt i matematikk.
- Ordet kommer fra gresk og betyr formlik (iso = lik, morf = form).
- Isomorfe objekter skal ha samme *form*, men kan ha ulikt innhold.
- En isomorfi er en funksjon som er injektiv og surjektiv – det er altså en en-til-en-korrespondanse – og som bevarer bestemte egenskaper.
- Intuitivt, så sier vi at to matematiske objekter er *isomorfe* hvis de er “strukturelt like”.
- Hvis vi har tegnet opp to grafer og kan komme fra den ene grafen til den andre ved kun å flytte på noder og endre på lengdene på kantene, så er grafene *isomorfe*.
- Vi har ikke lov til å legge til eller ta bort noder eller kanter, eller dele kanter eller noder i to...
- Følgende to grafer er isomorfe.

Grafisomorfier

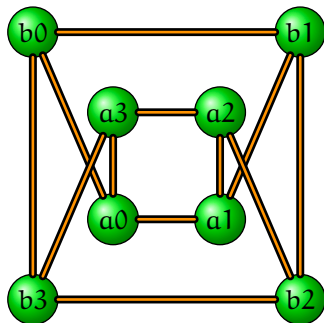
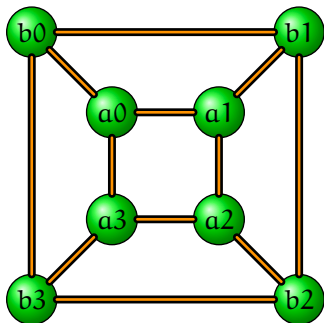
Grafisomorfier



Grafisomorfier



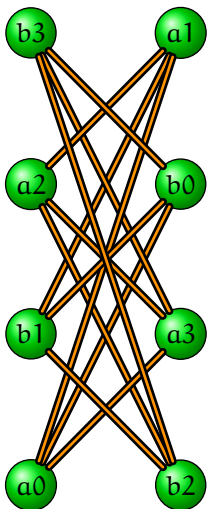
Grafisomorfier



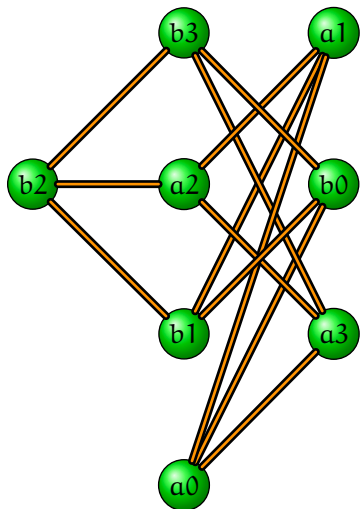
- Disse to grafene er også isomorfe med følgende grafer.

Grafisomorfier

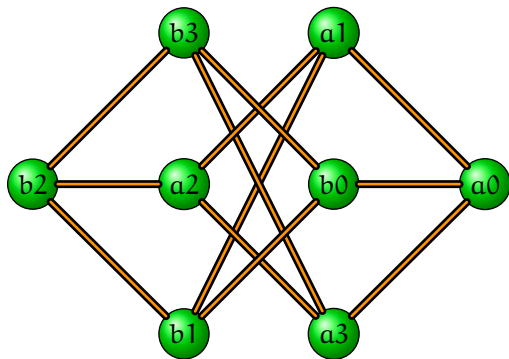
Grafisomorfier



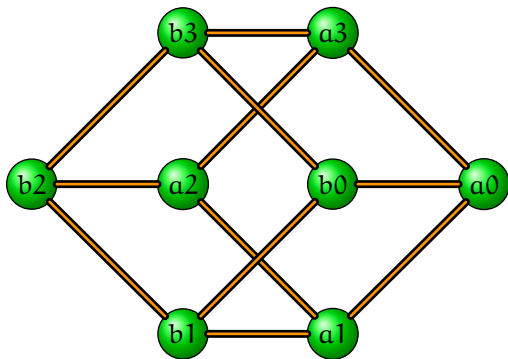
Grafisomorfier



Grafisomorfier



Grafisomorfier



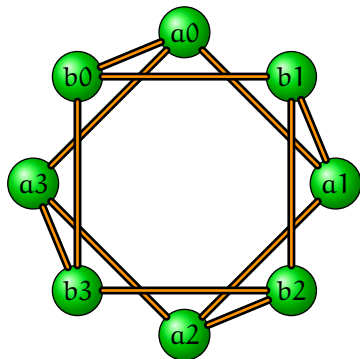
Grafisomorfier

Grafisomorfier

- De neste par grafene er også er isomorfe med disse.

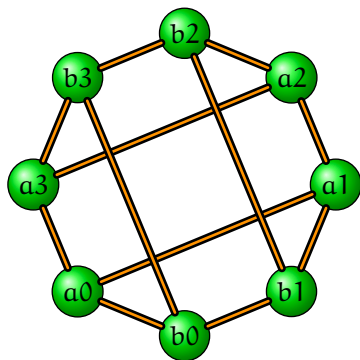
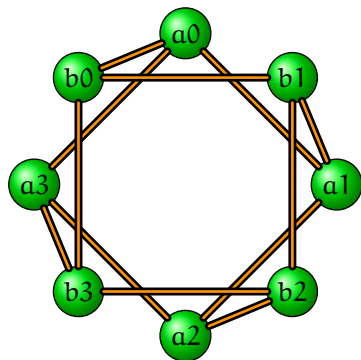
Grafisomorfier

- De neste par grafene er også er isomorfe med disse.



Grafisomorfier

- De neste par grafene er også er isomorfe med disse.



Grafisomorfier

Grafisomorfier

- Vi skal nå gjøre isomorfibegrepet helt presist.

Grafisomorfier

- Vi skal nå gjøre isomorfibegrepet helt presist.
- Hvis G og H er to grafer, så er en isomorfi mellom grafene en funksjon fra nodene i G til nodene i H som oppfyller bestemte egenskaper.

Grafisomorfier

- Vi skal nå gjøre isomorfibegrepet helt presist.
- Hvis G og H er to grafer, så er en isomorfi mellom grafene en funksjon fra nodene i G til nodene i H som oppfyller bestemte egenskaper.

Definisjon (Isomorfi)

Grafisomorfier

- Vi skal nå gjøre isomorfibegrepet helt presist.
- Hvis G og H er to grafer, så er en isomorfi mellom grafene en funksjon fra nodene i G til nodene i H som oppfyller bestemte egenskaper.

Definisjon (Isomorfi)

La G og H være to enkle grafer slik at $V(G)$ er mengden av noder i G og $V(H)$ er mengden av noder i H .

Grafisomorfier

- Vi skal nå gjøre isomorfibegrepet helt presist.
- Hvis G og H er to grafer, så er en isomorfi mellom grafene en funksjon fra nodene i G til nodene i H som oppfyller bestemte egenskaper.

Definisjon (Isomorfi)

La G og H være to enkle grafer slik at $V(G)$ er mengden av noder i G og $V(H)$ er mengden av noder i H . En *isomorfi* fra G til H er en funksjon $f : V(G) \rightarrow V(H)$ med følgende egenskaper:

Grafisomorfier

- Vi skal nå gjøre isomorfibegrepet helt presist.
- Hvis G og H er to grafer, så er en isomorfi mellom grafene en funksjon fra nodene i G til nodene i H som oppfyller bestemte egenskaper.

Definisjon (Isomorfi)

La G og H være to enkle grafer slik at $V(G)$ er mengden av noder i G og $V(H)$ er mengden av noder i H . En *isomorfi* fra G til H er en funksjon $f : V(G) \rightarrow V(H)$ med følgende egenskaper:

- f er surjektiv og injektiv

Grafisomorfier

- Vi skal nå gjøre isomorfibegrepet helt presist.
- Hvis G og H er to grafer, så er en isomorfi mellom grafene en funksjon fra nodene i G til nodene i H som oppfyller bestemte egenskaper.

Definisjon (Isomorfi)

La G og H være to enkle grafer slik at $V(G)$ er mengden av noder i G og $V(H)$ er mengden av noder i H . En *isomorfi* fra G til H er en funksjon $f : V(G) \rightarrow V(H)$ med følgende egenskaper:

- f er surjektiv og injektiv
- Nodene u og v er naboer i G hvis og bare hvis nodene $f(u)$ og $f(v)$ er naboer i H .

Grafisomorfier

Grafisomorfier

Eksempel

Eksempel

- La grafen G bestå av nodene $\{a, b, c, d\}$ og kantene $\{ab, bc, cd, da\}$.

Eksempel

- La grafen G bestå av nodene $\{a, b, c, d\}$ og kantene $\{ab, bc, cd, da\}$.
- La grafen H bestå av nodene $\{1, 2, 3, 4\}$ og kantene $\{14, 34, 12, 32\}$.

Eksempel

- La grafen G bestå av nodene $\{a, b, c, d\}$ og kantene $\{ab, bc, cd, da\}$.
- La grafen H bestå av nodene $\{1, 2, 3, 4\}$ og kantene $\{14, 34, 12, 32\}$.
- Funksjonen f slik at $f(a) = 1$, $f(b) = 2$, $f(c) = 3$ og $f(d) = 4$ er en isomorfi.

Eksempel

- La grafen G bestå av nodene $\{a, b, c, d\}$ og kantene $\{ab, bc, cd, da\}$.
- La grafen H bestå av nodene $\{1, 2, 3, 4\}$ og kantene $\{14, 34, 12, 32\}$.
- Funksjonen f slik at $f(a) = 1$, $f(b) = 2$, $f(c) = 3$ og $f(d) = 4$ er en isomorfi.
- F.eks. ser vi at a og b er naboer i G (siden ab er en kant).

Eksempel

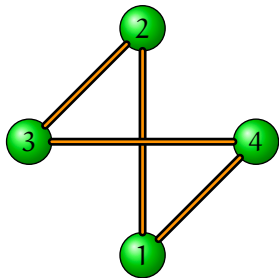
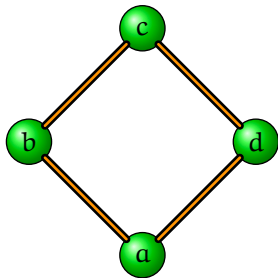
- La grafen G bestå av nodene $\{a, b, c, d\}$ og kantene $\{ab, bc, cd, da\}$.
- La grafen H bestå av nodene $\{1, 2, 3, 4\}$ og kantene $\{14, 34, 12, 32\}$.
- Funksjonen f slik at $f(a) = 1$, $f(b) = 2$, $f(c) = 3$ og $f(d) = 4$ er en isomorfi.
- F.eks. ser vi at a og b er naboer i G (siden ab er en kant).
- Da må $f(a)$ og $f(b)$, som er 1 og 2 , være naboer i H .

Eksempel

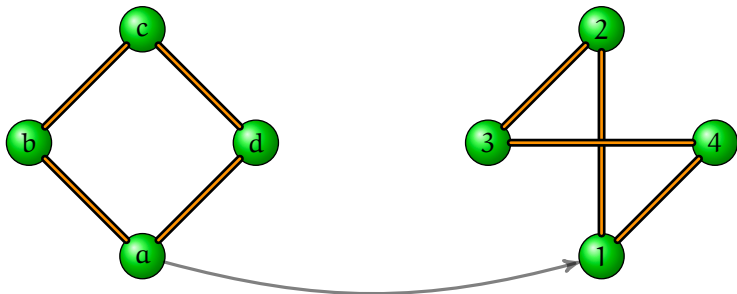
- La grafen G bestå av nodene $\{a, b, c, d\}$ og kantene $\{ab, bc, cd, da\}$.
- La grafen H bestå av nodene $\{1, 2, 3, 4\}$ og kantene $\{14, 34, 12, 32\}$.
- Funksjonen f slik at $f(a) = 1$, $f(b) = 2$, $f(c) = 3$ og $f(d) = 4$ er en isomorfi.
- F.eks. ser vi at a og b er naboer i G (siden ab er en kant).
- Da må $f(a)$ og $f(b)$, som er 1 og 2 , være naboer i H .
- Det stemmer, siden 12 er en kant i H .

Grafisomorfier

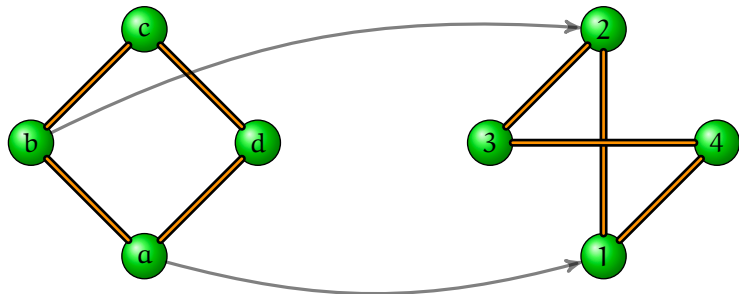
Grafisomorfier



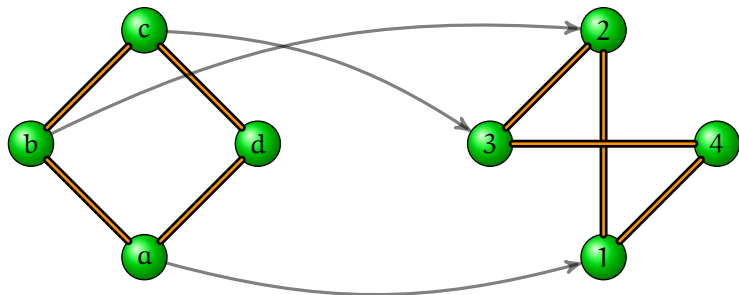
Grafisomorfier



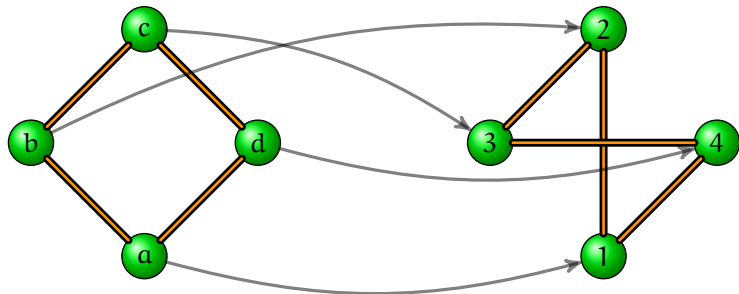
Grafisomorfier



Grafisomorfier



Grafisomorfier



Grafisomorfier

Grafisomorfier

Merk

Merk

- Det står “*hvis og bare hvis*” i definisjonen:

Merk

- Det står “*hvis og bare hvis*” i definisjonen:
- Nodene u og v er naboer i G **hvis og bare hvis** nodene $f(u)$ og $f(v)$ er naboer i H .

Merk

- Det står “*hvis og bare hvis*” i definisjonen:
- Nodene u og v er naboer i G **hvis og bare hvis** nodene $f(u)$ og $f(v)$ er naboer i H .
- *Hvis*-delen av definisjonen er denne påstanden:

Merk

- Det står “*hvis og bare hvis*” i definisjonen:
- Nodene u og v er naboer i G **hvis og bare hvis** nodene $f(u)$ og $f(v)$ er naboer i H .
- *Hvis*-delen av definisjonen er denne påstanden:
 - Nodene u og v er naboer i G **hvis** nodene $f(u)$ og $f(v)$ er naboer i H .

Merk

- Det står “*hvis og bare hvis*” i definisjonen:
- Nodene u og v er naboer i G **hvis og bare hvis** nodene $f(u)$ og $f(v)$ er naboer i H .
- *Hvis*-delen av definisjonen er denne påstanden:
 - Nodene u og v er naboer i G **hvis** nodene $f(u)$ og $f(v)$ er naboer i H .
 - Det betyr at hvis nodene u og v *ikke* er naboer, så er heller ikke nodene $f(u)$ og $f(v)$ naboer.

Merk

- Det står “*hvis og bare hvis*” i definisjonen:
- Nodene u og v er naboer i G **hvis og bare hvis** nodene $f(u)$ og $f(v)$ er naboer i H .
- *Hvis*-delen av definisjonen er denne påstanden:
 - Nodene u og v er naboer i G **hvis** nodene $f(u)$ og $f(v)$ er naboer i H .
 - Det betyr at hvis nodene u og v *ikke* er naboer, så er heller ikke nodene $f(u)$ og $f(v)$ naboer.
 - “Naboer i $H \rightarrow$ Naboer i G ”

Merk

- Det står “*hvis og bare hvis*” i definisjonen:
- Nodene u og v er naboer i G **hvis og bare hvis** nodene $f(u)$ og $f(v)$ er naboer i H .
- *Hvis*-delen av definisjonen er denne påstanden:
 - Nodene u og v er naboer i G **hvis** nodene $f(u)$ og $f(v)$ er naboer i H .
 - Det betyr at hvis nodene u og v *ikke* er naboer, så er heller ikke nodene $f(u)$ og $f(v)$ naboer.
 - “Naboer i $H \rightarrow$ Naboer i G ”
- *Bare hvis*-delen av definisjonen er denne påstanden:

Merk

- Det står “*hvis og bare hvis*” i definisjonen:
- Nodene u og v er naboer i G **hvis og bare hvis** nodene $f(u)$ og $f(v)$ er naboer i H .
- *Hvis*-delen av definisjonen er denne påstanden:
 - Nodene u og v er naboer i G **hvis** nodene $f(u)$ og $f(v)$ er naboer i H .
 - Det betyr at hvis nodene u og v *ikke* er naboer, så er heller ikke nodene $f(u)$ og $f(v)$ naboer.
 - “Naboer i $H \rightarrow$ Naboer i G ”
- *Bare hvis*-delen av definisjonen er denne påstanden:
 - Nodene u og v er naboer **bare hvis** nodene $f(u)$ og $f(v)$ naboer.

Merk

- Det står “*hvis og bare hvis*” i definisjonen:
- Nodene u og v er naboer i G **hvis og bare hvis** nodene $f(u)$ og $f(v)$ er naboer i H .
- *Hvis*-delen av definisjonen er denne påstanden:
 - Nodene u og v er naboer i G **hvis** nodene $f(u)$ og $f(v)$ er naboer i H .
 - Det betyr at hvis nodene u og v *ikke* er naboer, så er heller ikke nodene $f(u)$ og $f(v)$ naboer.
 - “Naboer i $H \rightarrow$ Naboer i G ”
- *Bare hvis*-delen av definisjonen er denne påstanden:
 - Nodene u og v er naboer **bare hvis** nodene $f(u)$ og $f(v)$ naboer.
 - Det betyr at hvis nodene u og v er naboer, så er også nodene $f(u)$ og $f(v)$ naboer.

Merk

- Det står “*hvis og bare hvis*” i definisjonen:
- Nodene u og v er naboer i G **hvis og bare hvis** nodene $f(u)$ og $f(v)$ er naboer i H .
- *Hvis*-delen av definisjonen er denne påstanden:
 - Nodene u og v er naboer i G **hvis** nodene $f(u)$ og $f(v)$ er naboer i H .
 - Det betyr at hvis nodene u og v *ikke* er naboer, så er heller ikke nodene $f(u)$ og $f(v)$ naboer.
 - “Naboer i $H \rightarrow$ Naboer i G ”
- *Bare hvis*-delen av definisjonen er denne påstanden:
 - Nodene u og v er naboer **bare hvis** nodene $f(u)$ og $f(v)$ naboer.
 - Det betyr at hvis nodene u og v er naboer, så er også nodene $f(u)$ og $f(v)$ naboer.
 - “Naboer i $G \rightarrow$ Naboer i H ”

Grafisomorfier

Grafisomorfier

- Vi har definert **isomorfi** mellom **enkle grafer**.

Grafisomorfier

- Vi har definert **isomorfi** mellom **enkle grafer**.
- Vi kan utvide definisjonen til å gjelde **endelige grafer** generelt.

Grafisomorfier

- Vi har definert **isomorfi** mellom **enkle grafer**.
- Vi kan utvide definisjonen til å gjelde **endelige grafer** generelt.
- Da krever vi at **antall kanter** mellom u og v i G skal være det samme som antall kanter mellom $f(u)$ og $f(v)$ i H .

Grafisomorfier

- Vi har definert **isomorfi** mellom **enkle grafer**.
- Vi kan utvide definisjonen til å gjelde **endelige grafer** generelt.
- Da krever vi at **antall kanter** mellom u og v i G skal være det samme som antall kanter mellom $f(u)$ og $f(v)$ i H .
- For enkle grafer kan dette antallet være 0 (om u og v ikke er naboer) og 1 (om u og v er naboer).

Grafisomorfier

Grafisomorfier

- For å vise at to grafer er isomorfe, må man gi en funksjon og argumentere for at funksjonen har egenskapene som er nødvendige.

Grafisomorfier

- For å vise at to grafer er isomorfe, må man gi en funksjon og argumentere for at funksjonen har egenskapene som er nødvendige.
- Det fins per i dag ingen effektiv algoritme for å avgjøre om to grafer er isomorfe.

Grafisomorfier

- For å vise at to grafer er isomorfe, må man gi en funksjon og argumentere for at funksjonen har egenskapene som er nødvendige.
- Det fins per i dag ingen effektiv algoritme for å avgjøre om to grafer er isomorfe.
- For å vise at to grafer *ikke* er isomorfe, så er det tilstrekkelig å finne en “grafteoretisk egenskap” som kun den ene av grafene har.

Grafisomorfier

- For å vise at to grafer er isomorfe, må man gi en funksjon og argumentere for at funksjonen har egenskapene som er nødvendige.
- Det fins per i dag ingen effektiv algoritme for å avgjøre om to grafer er isomorfe.
- For å vise at to grafer *ikke* er isomorfe, så er det tilstrekkelig å finne en “grafteoretisk egenskap” som kun den ene av grafene har.
- En grafteoretisk egenskap er en egenskap som bevares under “lovlige” transformasjoner, som å flytte rundt på nodene, gjøre kantene lengre/kortere, etc.

Grafisomorfier

- For å vise at to grafer er isomorfe, må man gi en funksjon og argumentere for at funksjonen har egenskapene som er nødvendige.
- Det fins per i dag ingen effektiv algoritme for å avgjøre om to grafer er isomorfe.
- For å vise at to grafer *ikke* er isomorfe, så er det tilstrekkelig å finne en “grafteoretisk egenskap” som kun den ene av grafene har.
- En grafteoretisk egenskap er en egenskap som bevares under “lovlige” transformasjoner, som å flytte rundt på nodene, gjøre kantene lengre/kortere, etc.
- Noen av de enkleste grafteoretiske egenskapene er f.eks.:

Grafisomorfier

- For å vise at to grafer er isomorfe, må man gi en funksjon og argumentere for at funksjonen har egenskapene som er nødvendige.
- Det fins per i dag ingen effektiv algoritme for å avgjøre om to grafer er isomorfe.
- For å vise at to grafer *ikke* er isomorfe, så er det tilstrekkelig å finne en “grafteoretisk egenskap” som kun den ene av grafene har.
- En grafteoretisk egenskap er en egenskap som bevares under “lovlige” transformasjoner, som å flytte rundt på nodene, gjøre kantene lengre/kortere, etc.
- Noen av de enkleste grafteoretiske egenskapene er f.eks.:
 - Hvor mange noder en graf har.

Grafisomorfier

- For å vise at to grafer er isomorfe, må man gi en funksjon og argumentere for at funksjonen har egenskapene som er nødvendige.
- Det fins per i dag ingen effektiv algoritme for å avgjøre om to grafer er isomorfe.
- For å vise at to grafer *ikke* er isomorfe, så er det tilstrekkelig å finne en “grafteoretisk egenskap” som kun den ene av grafene har.
- En grafteoretisk egenskap er en egenskap som bevares under “lovlige” transformasjoner, som å flytte rundt på nodene, gjøre kantene lengre/kortere, etc.
- Noen av de enkleste grafteoretiske egenskapene er f.eks.:
 - Hvor mange noder en graf har.
 - Hvor mange kanter en graf har.

Grafisomorfier

- For å vise at to grafer er isomorfe, må man gi en funksjon og argumentere for at funksjonen har egenskapene som er nødvendige.
- Det fins per i dag ingen effektiv algoritme for å avgjøre om to grafer er isomorfe.
- For å vise at to grafer *ikke* er isomorfe, så er det tilstrekkelig å finne en “grafteoretisk egenskap” som kun den ene av grafene har.
- En grafteoretisk egenskap er en egenskap som bevares under “lovlige” transformasjoner, som å flytte rundt på nodene, gjøre kantene lengre/kortere, etc.
- Noen av de enkleste grafteoretiske egenskapene er f.eks.:
 - Hvor mange noder en graf har.
 - Hvor mange kanter en graf har.
 - Hvor mange noder av en bestemt grad en graf har.

Grafisomorfier

- For å vise at to grafer er isomorfe, må man gi en funksjon og argumentere for at funksjonen har egenskapene som er nødvendige.
- Det fins per i dag ingen effektiv algoritme for å avgjøre om to grafer er isomorfe.
- For å vise at to grafer *ikke* er isomorfe, så er det tilstrekkelig å finne en “grafteoretisk egenskap” som kun den ene av grafene har.
- En grafteoretisk egenskap er en egenskap som bevares under “lovlige” transformasjoner, som å flytte rundt på nodene, gjøre kantene lengre/kortere, etc.
- Noen av de enkleste grafteoretiske egenskapene er f.eks.:
 - Hvor mange noder en graf har.
 - Hvor mange kanter en graf har.
 - Hvor mange noder av en bestemt grad en graf har.
 - Korteste avstand mellom to noder.

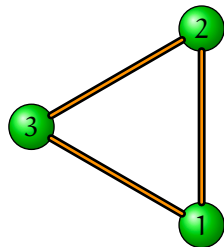
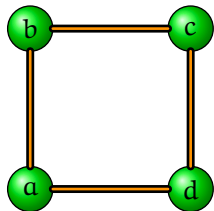
Grafisomorfier

Grafisomorfier

- Er følgende grafer isomorfe?

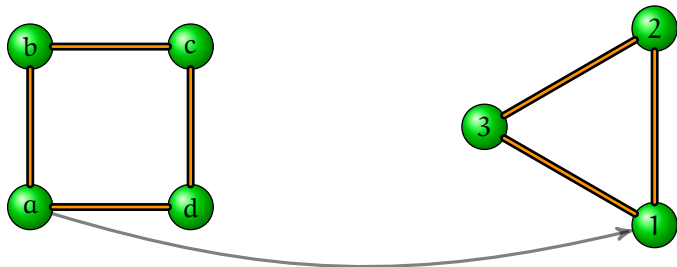
Grafisomorfier

- Er følgende grafer isomorfe?



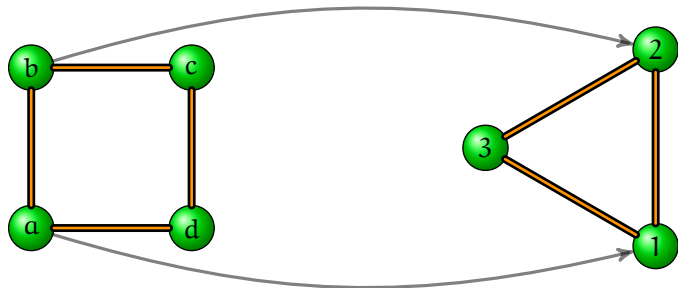
Grafisomorfier

- Er følgende grafer isomorfe?



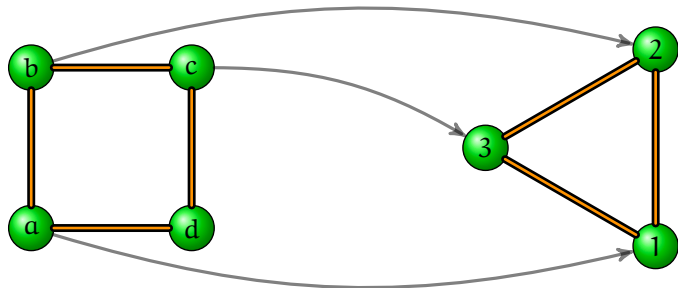
Grafisomorfier

- Er følgende grafer isomorfe?



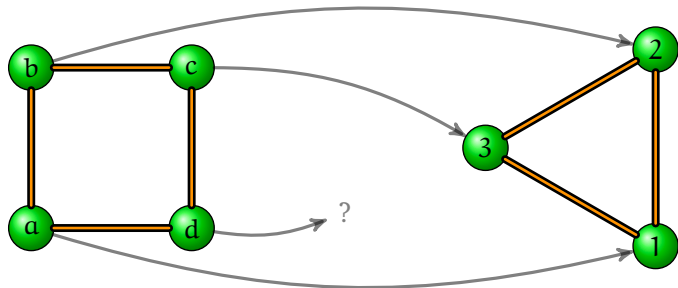
Grafisomorfier

- Er følgende grafer isomorfe?



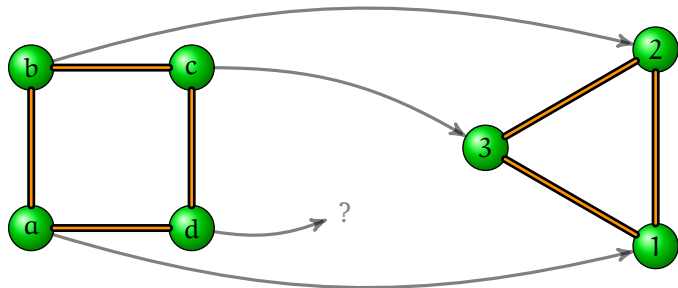
Grafisomorfier

- Er følgende grafer isomorfe?



Grafisomorfier

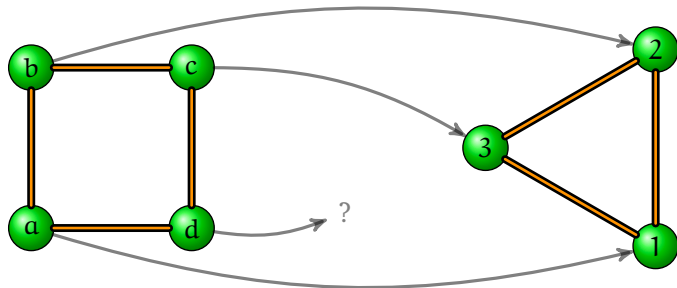
- Er følgende grafer isomorfe?



- Nei, de er ikke isomorfe.

Grafisomorfier

- Er følgende grafer isomorfe?



- Nei, de er ikke isomorfe.
- Grafen til høyre har *færre* noder enn grafen til venstre, så ingen funksjon fra den venstre grafen til den høyre kan være *injektiv* eller *en-til-en*.

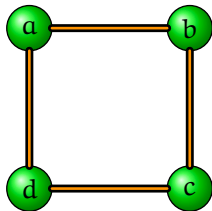
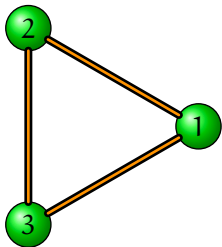
Grafisomorfier

Grafisomorfier

- Er følgende grafer isomorfe?

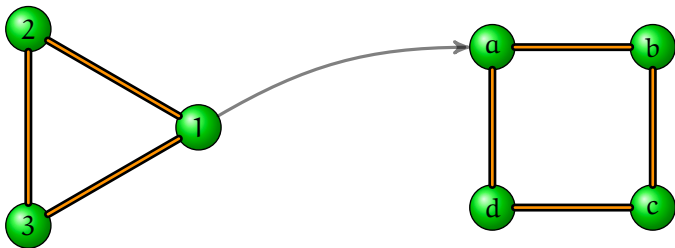
Grafisomorfier

- Er følgende grafer isomorfe?



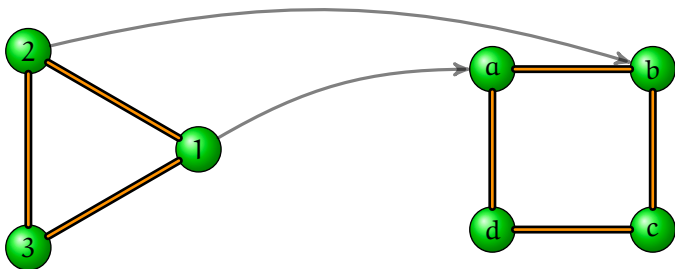
Grafisomorfier

- Er følgende grafer isomorfe?



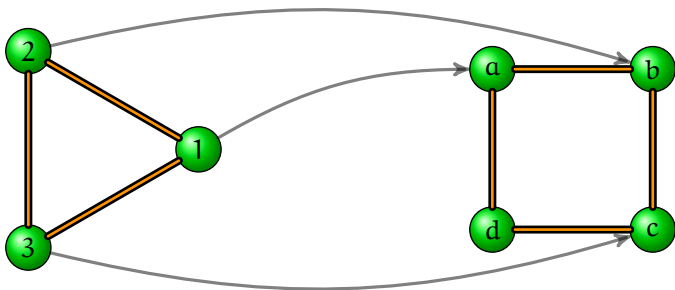
Grafisomorfier

- Er følgende grafer isomorfe?



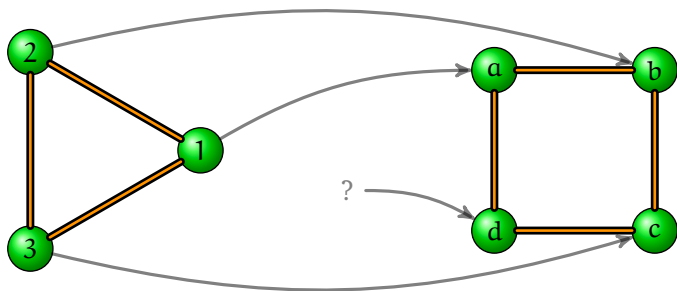
Grafisomorfier

- Er følgende grafer isomorfe?



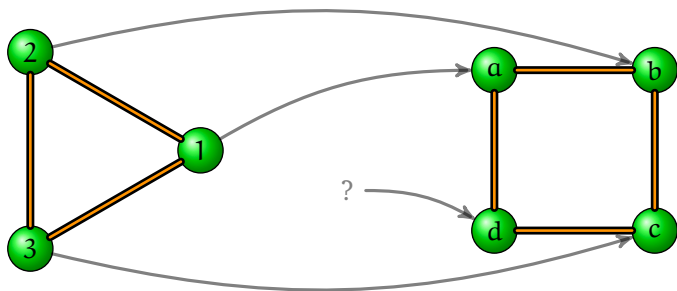
Grafisomorfier

- Er følgende grafer isomorfe?



Grafisomorfier

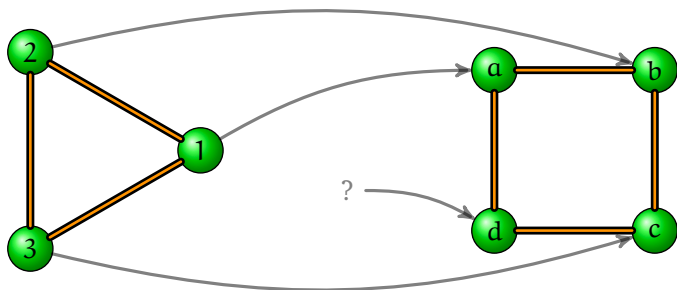
- Er følgende grafer isomorfe?



- Nei, de er ikke isomorfe.

Grafisomorfier

- Er følgende grafer isomorfe?



- Nei, de er ikke isomorfe.
- Grafen til høyre har *flere* noder enn grafen til venstre, så ingen funksjon fra den venstre grafen til den høyre kan være *surjektiv* eller *på*.

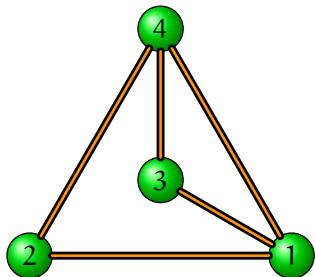
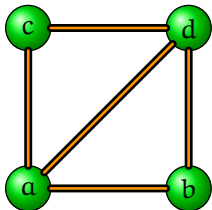
Grafisomorfier

Grafisomorfier

- Er følgende grafer, G og H, isomorfe?

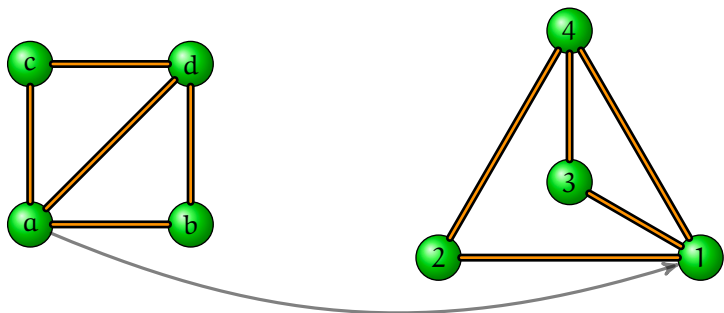
Grafisomorfier

- Er følgende grafer, G og H, isomorfe?



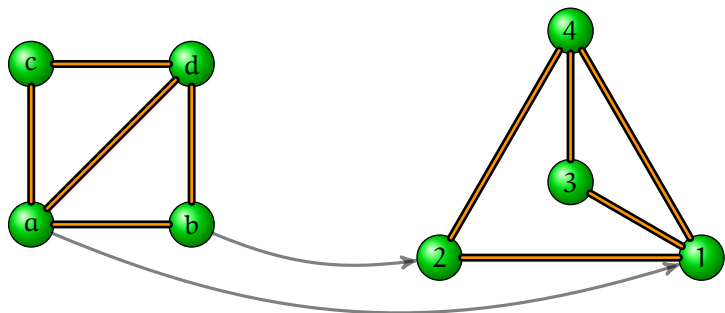
Grafisomorfier

- Er følgende grafer, G og H, isomorfe?



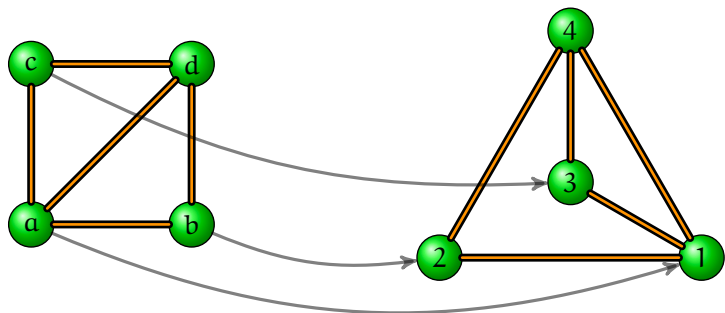
Grafisomorfier

- Er følgende grafer, G og H, isomorfe?



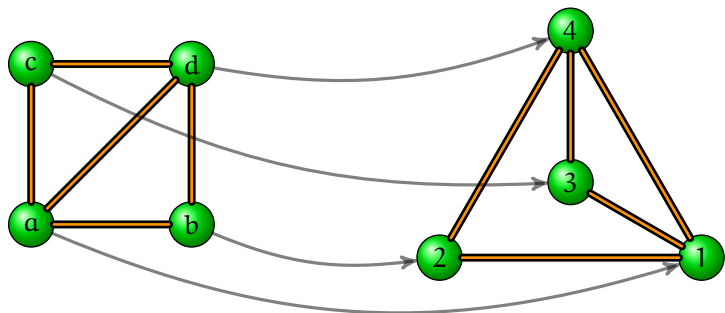
Grafisomorfier

- Er følgende grafer, G og H, isomorfe?



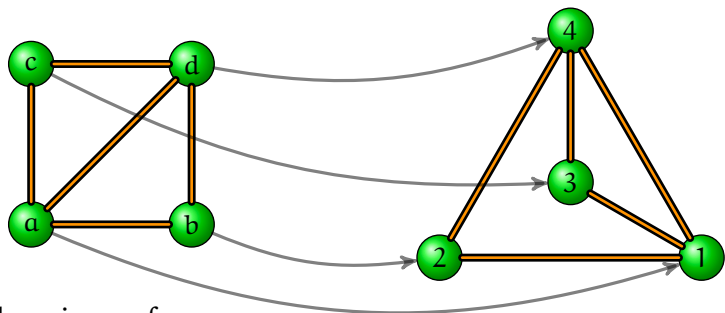
Grafisomorfier

- Er følgende grafer, G og H, isomorfe?



Grafisomorfier

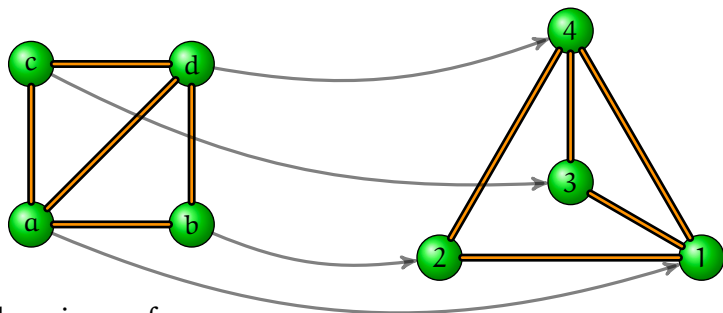
- Er følgende grafer, G og H, isomorfe?



- Ja, de er isomorfe.

Grafisomorfier

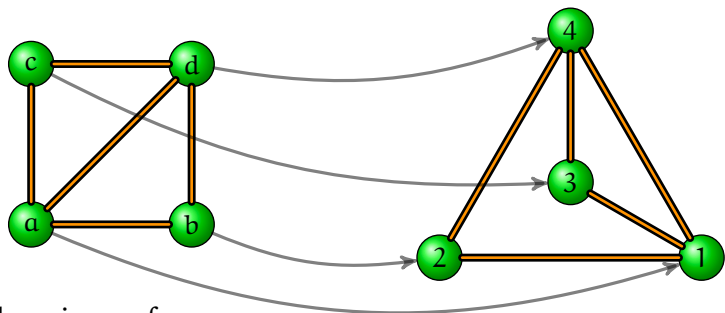
- Er følgende grafer, G og H, isomorfe?



- Ja, de er isomorfe.
- Funksjonen $f : V(G) \rightarrow V(H)$ gitt ved $f(a) = 1$, $f(b) = 2$, $f(c) = 3$ og $f(d) = 4$ er en isomorfi.

Grafisomorfier

- Er følgende grafer, G og H, isomorfe?



- Ja, de er isomorfe.
- Funksjonen $f : V(G) \rightarrow V(H)$ gitt ved $f(a) = 1$, $f(b) = 2$, $f(c) = 3$ og $f(d) = 4$ er en isomorfi.
- Vi ser at u og v er naboer i G hvis og bare hvis $f(u)$ og $f(v)$ er naboer i H.

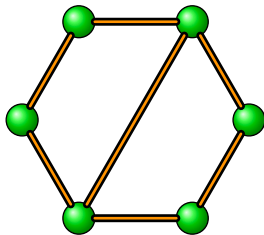
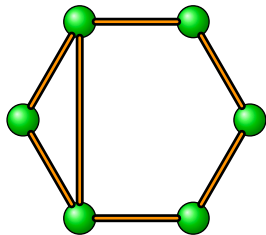
Grafisomorfier

Grafisomorfier

- Er følgende grafer isomorfe?

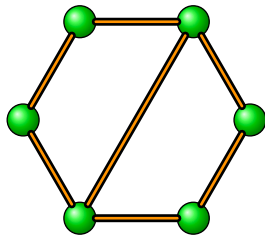
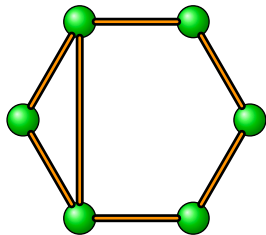
Grafisomorfier

- Er følgende grafer isomorfe?



Grafisomorfier

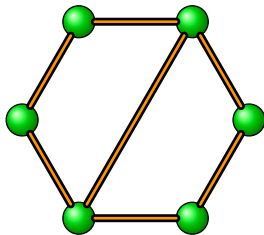
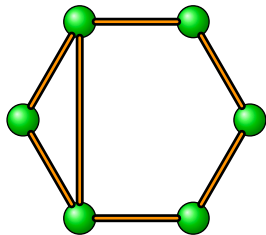
- Er følgende grafer isomorfe?



- Nei, de er ikke isomorfe.

Grafisomorfier

- Er følgende grafer isomorfe?



- Nei, de er ikke isomorfe.
- Grafen til venstre inneholder tre noder som alle er relatert til hverandre; det gjør ikke grafen til høyre.

Noen kommentarer

Noen kommentarer

- Å avgjøre om to grafer er isomorfe er i bunn og grunn å finne ut om det er den samme grafen man har å gjøre med.

Noen kommentarer

- Å avgjøre om to grafer er isomorfe er i bunn og grunn å finne ut om det er den samme grafen man har å gjøre med.
- Hvis en rekke grafer er gitt, så ønsker vi å finne ut om noen av dem er isomorfe for å unngå å gjøre overflødig arbeid.

Noen kommentarer

- Å avgjøre om to grafer er isomorfe er i bunn og grunn å finne ut om det er den samme grafen man har å gjøre med.
- Hvis en rekke grafer er gitt, så ønsker vi å finne ut om noen av dem er isomorfe for å unngå å gjøre overflødig arbeid.
- Det er ingen som har klart å lage en *effektiv* algoritme (polynomiell tid) for å avgjøre om grafer er isomorfe (i det generelle tilfellet).

Noen kommentarer

- Å avgjøre om to grafer er isomorfe er i bunn og grunn å finne ut om det er den samme grafen man har å gjøre med.
- Hvis en rekke grafer er gitt, så ønsker vi å finne ut om noen av dem er isomorfe for å unngå å gjøre overflødig arbeid.
- Det er ingen som har klart å lage en *effektiv* algoritme (polynomiell tid) for å avgjøre om grafer er isomorfe (i det generelle tilfellet).
- Mange spesialtilfeller, f.eks. trær, vet man mye om.

Noen kommentarer

- Å avgjøre om to grafer er isomorfe er i bunn og grunn å finne ut om det er den samme grafen man har å gjøre med.
- Hvis en rekke grafer er gitt, så ønsker vi å finne ut om noen av dem er isomorfe for å unngå å gjøre overflødig arbeid.
- Det er ingen som har klart å lage en *effektiv* algoritme (polynomiell tid) for å avgjøre om grafer er isomorfe (i det generelle tilfellet).
- Mange spesialtilfeller, f.eks. trær, vet man mye om.
- I praksis så klarer man å lage ganske effektive algoritmer allikevel, men i *verste* tilfelle må man backtracke over alle $n!$ mulige omdøpinger av nodene.

Noen kommentarer

- Å avgjøre om to grafer er isomorfe er i bunn og grunn å finne ut om det er den samme grafen man har å gjøre med.
- Hvis en rekke grafer er gitt, så ønsker vi å finne ut om noen av dem er isomorfe for å unngå å gjøre overflødig arbeid.
- Det er ingen som har klart å lage en *effektiv* algoritme (polynomiell tid) for å avgjøre om grafer er isomorfe (i det generelle tilfellet).
- Mange spesialtilfeller, f.eks. trær, vet man mye om.
- I praksis så klarer man å lage ganske effektive algoritmer allikevel, men i *verste* tilfelle må man backtracke over alle $n!$ mulige omdøpinger av nodene.
- Å finne isomorfier fra en graf til seg selv er en måte å avdekke *symmetrier* på.

Noen kommentarer

- Å avgjøre om to grafer er isomorfe er i bunn og grunn å finne ut om det er den samme grafen man har å gjøre med.
- Hvis en rekke grafer er gitt, så ønsker vi å finne ut om noen av dem er isomorfe for å unngå å gjøre overflødig arbeid.
- Det er ingen som har klart å lage en *effektiv* algoritme (polynomiell tid) for å avgjøre om grafer er isomorfe (i det generelle tilfellet).
- Mange spesialtilfeller, f.eks. trær, vet man mye om.
- I praksis så klarer man å lage ganske effektive algoritmer allikevel, men i *verste* tilfelle må man backtracke over alle $n!$ mulige omdøpinger av nodene.
- Å finne isomorfier fra en graf til seg selv er en måte å avdekke *symmetrier* på.
- Å finne ut om en graf er en *delgraf* av en annen er et annet, men relatert, problem. (Ofte vanskeligere.)

Stier og kretser

Stier og kretser

- Vårt neste tema er *stier* (engelsk: *path*) og *kretser* (engelsk: *circuit*).

Stier og kretser

- Vårt neste tema er *stier* (engelsk: *path*) og *kretser* (engelsk: *circuit*).
- Vi skal begynne med det klassiske eksemplet om *Königsbergs broer*.

Stier og kretser

- Vårt neste tema er *stier* (engelsk: *path*) og *kretser* (engelsk: *circuit*).
- Vi skal begynne med det klassiske eksemplet om *Königsbergs broer*.
- Kort fortalt har vi sju broer som forbinder fire landområder.

Stier og kretser

- Vårt neste tema er *stier* (engelsk: *path*) og *kretser* (engelsk: *circuit*).
- Vi skal begynne med det klassiske eksemplet om *Königsbergs broer*.
- Kort fortalt har vi sju broer som forbinder fire landområder.
- Spørsmålet er om det går an å gå en tur i Königsberg slik at man går over hver av de sju broene *nøyaktig én gang*.

Stier og kretser

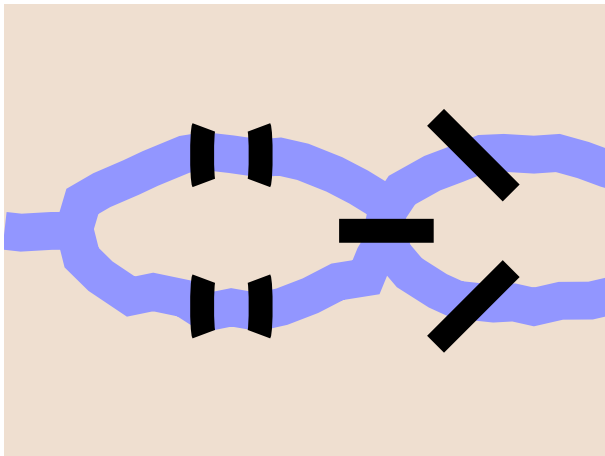
- Vårt neste tema er *stier* (engelsk: *path*) og *kretser* (engelsk: *circuit*).
- Vi skal begynne med det klassiske eksemplet om *Königsbergs broer*.
- Kort fortalt har vi sju broer som forbinder fire landområder.
- Spørsmålet er om det går an å gå en tur i Königsberg slik at man går over hver av de sju broene *nøyaktig én gang*.
- Dette er kjent for å ha blitt løst av Leonhard Euler omkring 1735.

Stier og kretser

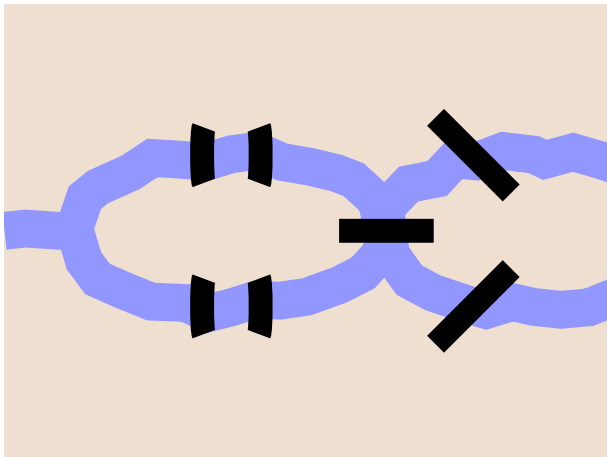
- Vårt neste tema er *stier* (engelsk: *path*) og *kretser* (engelsk: *circuit*).
- Vi skal begynne med det klassiske eksemplet om *Königsbergs broer*.
- Kort fortalt har vi sju broer som forbinder fire landområder.
- Spørsmålet er om det går an å gå en tur i Königsberg slik at man går over hver av de sju broene *nøyaktig én gang*.
- Dette er kjent for å ha blitt løst av Leonhard Euler omkring 1735.
- Vi skal se at oppgaven er den samme som å finne en *Eulersti* i grafen som representerer Königsberg.

Königsbergs broer

Königsbergs broer

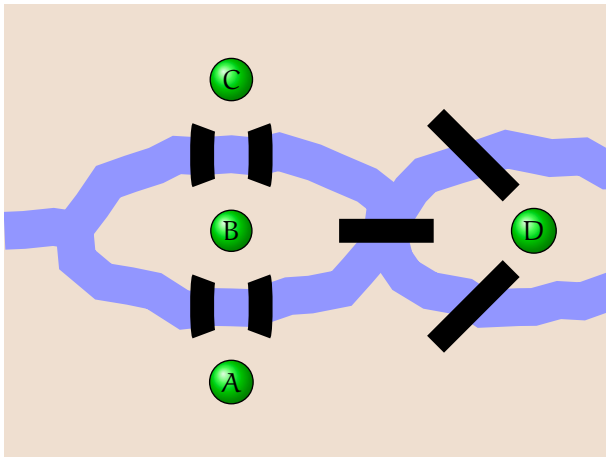


Königsbergs broer



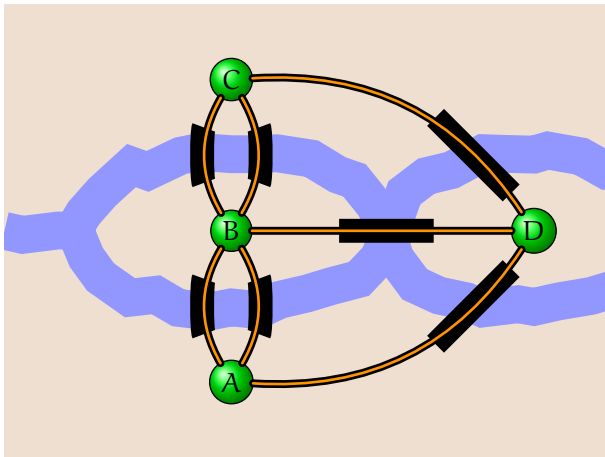
Vi representerer situasjonen med en graf.

Königsbergs broer



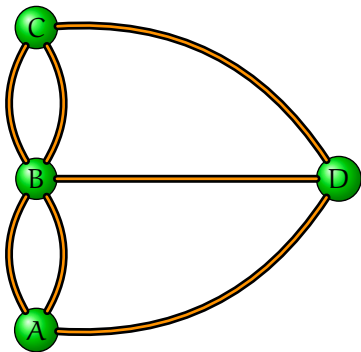
Vi representerer situasjonen med en graf.

Königsbergs broer



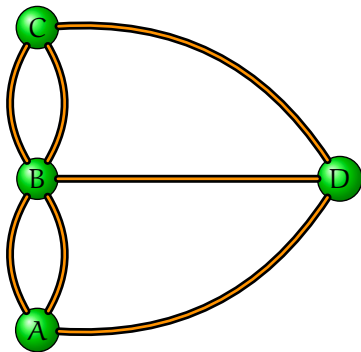
Vi representerer situasjonen med en graf.

Königsbergs broer



Vi representerer situasjonen med en graf.

Königsbergs broer



Vi representerer situasjonen med en graf. Spørsmålet blir nå om det er mulig å gå over alle kantene nøyaktig en gang.

Stier og kretser

Stier og kretser

Definisjon (Sti)

Stier og kretser

Definisjon (Sti)

En *sti* av lengde n i en graf er sekvens av noder og kanter på formen

Stier og kretser

Definisjon (Sti)

En *sti* av lengde n i en graf er sekvens av noder og kanter på formen

$$v_0 e_1 v_1 e_2 v_2 \dots e_n v_n$$

Stier og kretser

Definisjon (Sti)

En *sti* av lengde n i en graf er sekvens av noder og kanter på formen

$$v_0 e_1 v_1 e_2 v_2 \dots e_n v_n$$

hvor e_i er en kant som forbinder v_{i-1} og v_i for $i \in \{1, 2, \dots, n\}$.

Stier og kretser

Definisjon (Sti)

En *sti* av lengde n i en graf er sekvens av noder og kanter på formen

$$v_0 e_1 v_1 e_2 v_2 \dots e_n v_n$$

hvor e_i er en kant som forbinder v_{i-1} og v_i for $i \in \{1, 2, \dots, n\}$. En sti hvor $v_0 = v_n$ kalles en *krets*.

Stier og kretser

Definisjon (Sti)

En *sti* av lengde n i en graf er sekvens av noder og kanter på formen

$$v_0 e_1 v_1 e_2 v_2 \dots e_n v_n$$

hvor e_i er en kant som forbinder v_{i-1} og v_i for $i \in \{1, 2, \dots, n\}$. En sti hvor $v_0 = v_n$ kalles en *krets*. Vi sier at sekvensen er en sti fra v_0 til v_n .

Stier og kretser

Definisjon (Sti)

En *sti* av lengde n i en graf er sekvens av noder og kanter på formen

$$v_0 e_1 v_1 e_2 v_2 \dots e_n v_n$$

hvor e_i er en kant som forbinder v_{i-1} og v_i for $i \in \{1, 2, \dots, n\}$. En sti hvor $v_0 = v_n$ kalles en *krets*. Vi sier at sekvensen er en sti fra v_0 til v_n .

- En sti kalles også for en *vei*.

Stier og kretser

Definisjon (Sti)

En *sti* av lengde n i en graf er sekvens av noder og kanter på formen

$$v_0 e_1 v_1 e_2 v_2 \dots e_n v_n$$

hvor e_i er en kant som forbinder v_{i-1} og v_i for $i \in \{1, 2, \dots, n\}$. En sti hvor $v_0 = v_n$ kalles en *krets*. Vi sier at sekvensen er en sti fra v_0 til v_n .

- En sti kalles også for en *vei*.
- *Lengden* til en sti er det samme som antall kanter i stien.

Stier og kretser

Definisjon (Sti)

En *sti* av lengde n i en graf er sekvens av noder og kanter på formen

$$v_0 e_1 v_1 e_2 v_2 \dots e_n v_n$$

hvor e_i er en kant som forbinder v_{i-1} og v_i for $i \in \{1, 2, \dots, n\}$. En sti hvor $v_0 = v_n$ kalles en *krets*. Vi sier at sekvensen er en sti fra v_0 til v_n .

- En sti kalles også for en *vei*.
- *Lengden* til en sti er det samme som antall kanter i stien.
- En enkelt node er både en sti og en krets av lengde 0.

Stier og kretser

Definisjon (Sti)

En *sti* av lengde n i en graf er sekvens av noder og kanter på formen

$$v_0 e_1 v_1 e_2 v_2 \dots e_n v_n$$

hvor e_i er en kant som forbinder v_{i-1} og v_i for $i \in \{1, 2, \dots, n\}$. En sti hvor $v_0 = v_n$ kalles en *krets*. Vi sier at sekvensen er en sti fra v_0 til v_n .

- En sti kalles også for en *vei*.
- *Lengden* til en sti er det samme som antall kanter i stien.
- En enkelt node er både en sti og en krets av lengde 0.
- Hvis grafen er *enkel*, tillater vi oss å skrive $v_0 v_1 v_2 \dots v_n$ for stien.

Stier og kretser

Definisjon (Sti)

En *sti* av lengde n i en graf er sekvens av noder og kanter på formen

$$v_0 e_1 v_1 e_2 v_2 \dots e_n v_n$$

hvor e_i er en kant som forbinder v_{i-1} og v_i for $i \in \{1, 2, \dots, n\}$. En sti hvor $v_0 = v_n$ kalles en *krets*. Vi sier at sekvensen er en sti fra v_0 til v_n .

- En sti kalles også for en *vei*.
- *Lengden* til en sti er det samme som antall kanter i stien.
- En enkelt node er både en sti og en krets av lengde 0.
- Hvis grafen er *enkel*, tillater vi oss å skrive $v_0 v_1 v_2 \dots v_n$ for stien.
- Vær oppmerksom på at terminologien for grafteori varierer fra lærebok til lærebok.

Stier og kretser

Definisjon (Sti)

En *sti* av lengde n i en graf er sekvens av noder og kanter på formen

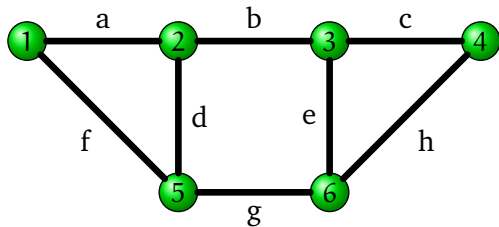
$$v_0 e_1 v_1 e_2 v_2 \dots e_n v_n$$

hvor e_i er en kant som forbinder v_{i-1} og v_i for $i \in \{1, 2, \dots, n\}$. En sti hvor $v_0 = v_n$ kalles en *krets*. Vi sier at sekvensen er en sti fra v_0 til v_n .

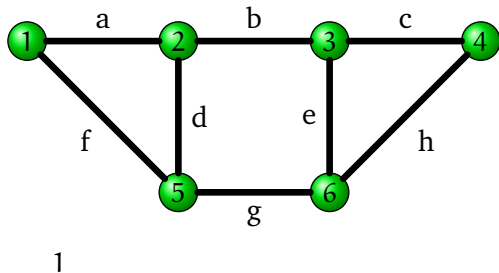
- En sti kalles også for en *vei*.
- *Lengden* til en sti er det samme som antall kanter i stien.
- En enkelt node er både en sti og en krets av lengde 0.
- Hvis grafen er *enkel*, tillater vi oss å skrive $v_0 v_1 v_2 \dots v_n$ for stien.
- Vær oppmerksom på at terminologien for grafteori varierer fra lærebok til lærebok.
- Vi ser på noen eksempler.

Stier og kretser

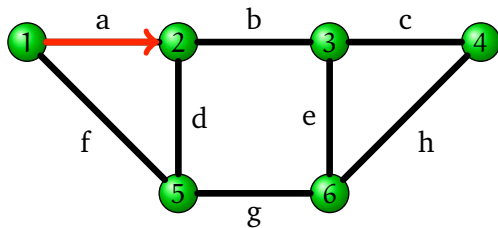
Stier og kretser



Stier og kretser

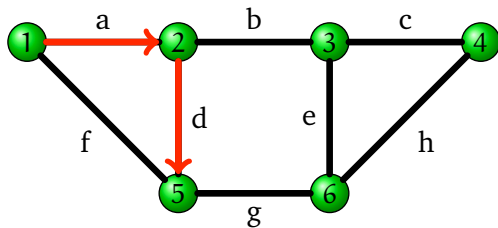


Stier og kretser



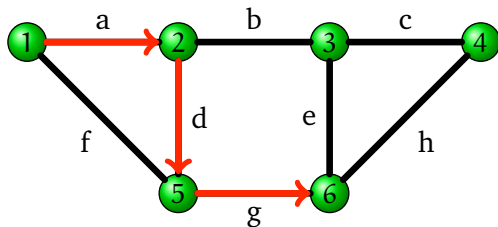
1a2

Stier og kretser



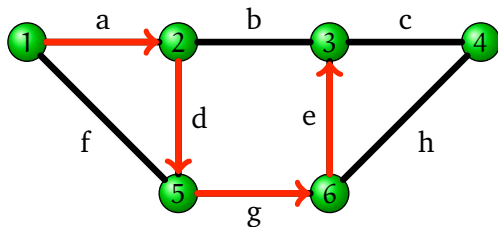
1a2d5

Stier og kretser



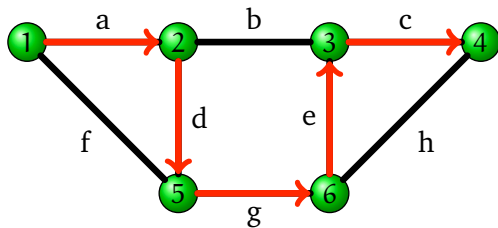
1a2d5g6

Stier og kretser



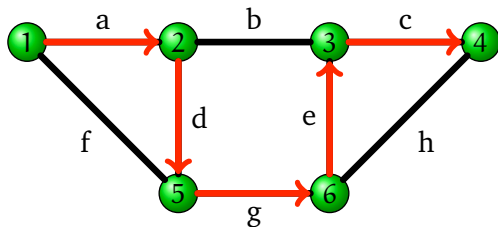
1a2d5g6e3

Stier og kretser



1a2d5g6e3c4

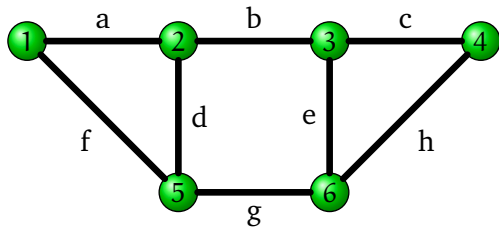
Stier og kretser



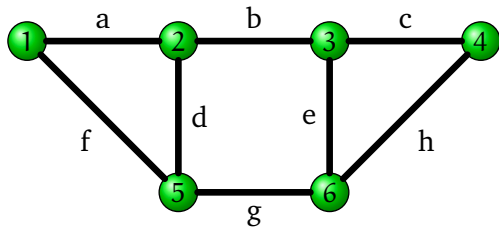
1a2d5g6e3c4 er en sti fra 1 til 4.

Stier og kretser

Stier og kretser

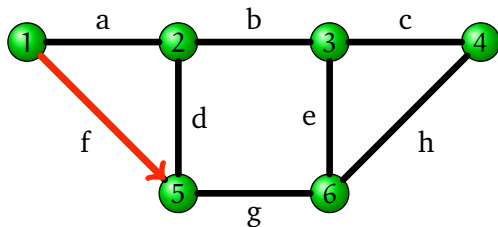


Stier og kretser



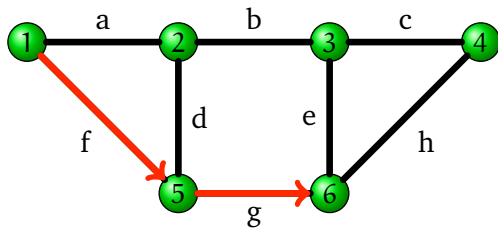
1

Stier og kretser



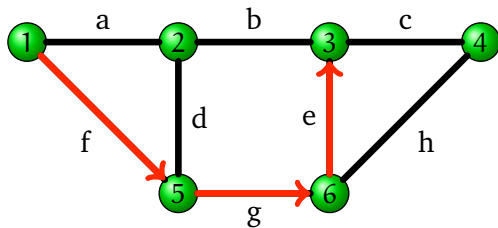
1f5

Stier og kretser



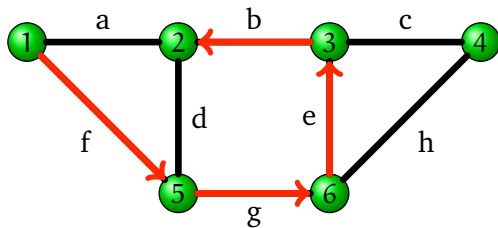
1f5g6

Stier og kretser



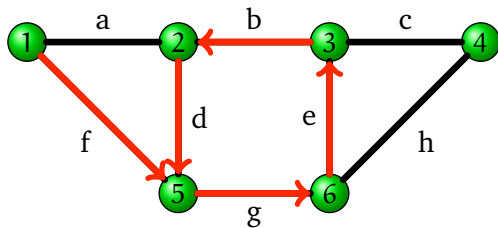
1f5g6e3

Stier og kretser



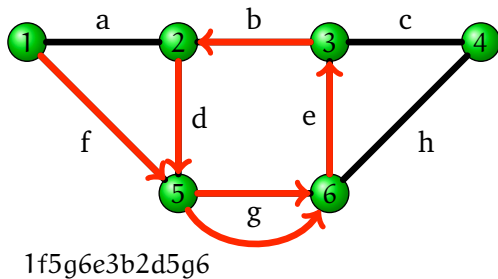
1f5g6e3b2

Stier og kretser

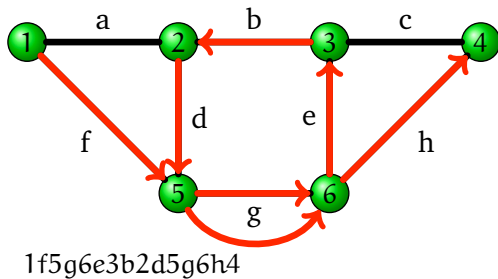


1f5g6e3b2d5

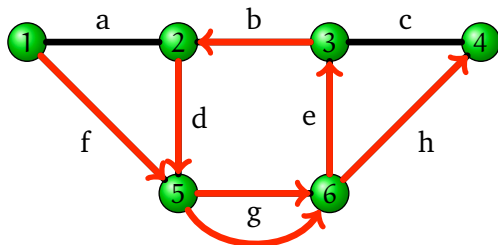
Stier og kretser



Stier og kretser



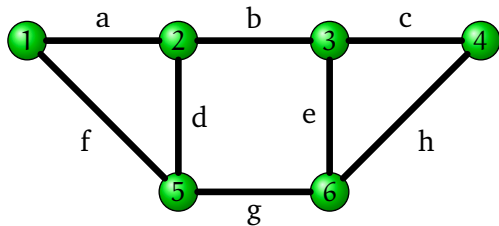
Stier og kretser



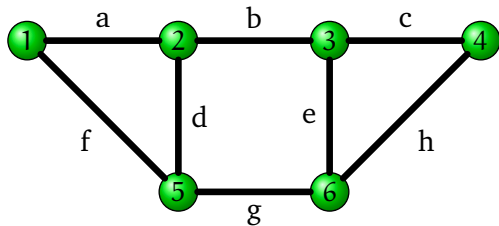
1f5g6e3b2d5g6h4 er en sti fra 1 til 4.

Stier og kretser

Stier og kretser

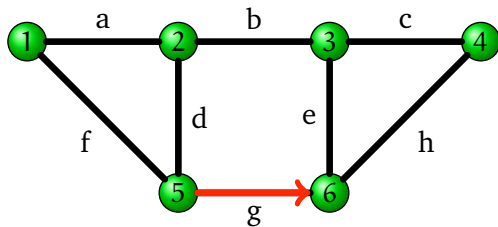


Stier og kretser



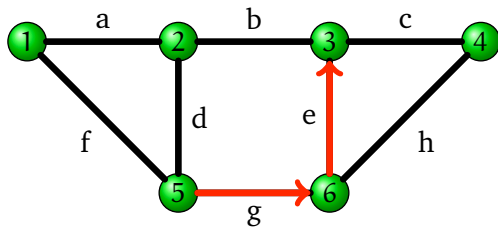
5

Stier og kretser



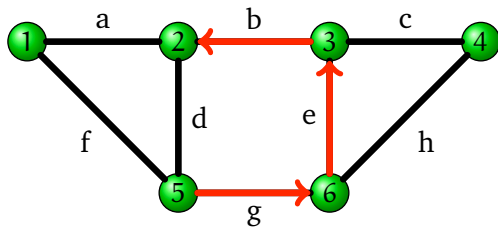
5g6

Stier og kretser



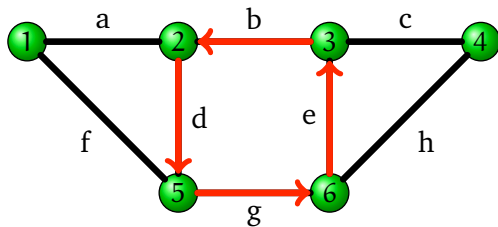
5g6e3

Stier og kretser



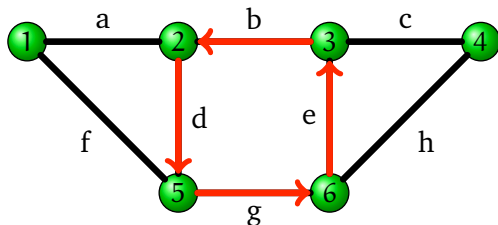
5g6e3b2

Stier og kretser



5g6e3b2d5

Stier og kretser



5g6e3b2d5 er en krets som begynner og slutter i 5.

Stier og kretser

Stier og kretser

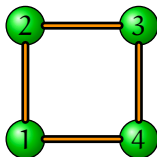
- Hvis en graf er enkel fins det ingen parallelle kanter, og da kan vi betegne en sti som en sekvens av noder.

Stier og kretser

- Hvis en graf er enkel fins det ingen parallelle kanter, og da kan vi betegne en sti som en sekvens av noder.
- Det er vanlig å identifisere kretser som kun er forskjellige med hensyn på startnode eller rekkefølge.

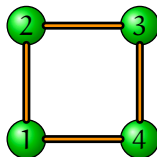
Stier og kretser

- Hvis en graf er enkel fins det ingen parallelle kanter, og da kan vi betegne en sti som en sekvens av noder.
- Det er vanlig å identifisere kretser som kun er forskjellige med hensyn på startnode eller rekkefølge.



Stier og kretser

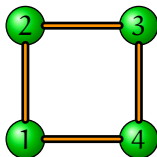
- Hvis en graf er enkel fins det ingen parallelle kanter, og da kan vi betegne en sti som en sekvens av noder.
- Det er vanlig å identifisere kretser som kun er forskjellige med hensyn på startnode eller rekkefølge.



- Her er 12341 en krets.

Stier og kretser

- Hvis en graf er enkel fins det ingen parallelle kanter, og da kan vi betegne en sti som en sekvens av noder.
- Det er vanlig å identifisere kretser som kun er forskjellige med hensyn på startnode eller rekkefølge.



- Her er 12341 en krets.
- Vi identifiserer denne med kretsene 23412, 34123, etc., og 43214, 32143, etc.

Stier og kretser

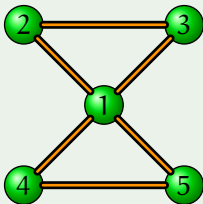
Oppgave

Oppgave

Hvor mange *forskjellige* kretser som inneholder alle kantene nøyaktig én gang fins i følgende graf?

Oppgave

Hvor mange *forskjellige* kretser som inneholder alle kantene nøyaktig én gang fins i følgende graf?



Stier og kretser

Stier og kretser

- Vi kan definere mengder av stier *induktivt* på følgende måte.

Stier og kretser

- Vi kan definere mengder av stier *induktivt* på følgende måte.

Definisjon (Mengden av stier - induktivt)

Stier og kretser

- Vi kan definere mengder av stier *induktivt* på følgende måte.

Definisjon (Mengden av stier - induktivt)

- En sekvens som består av en node v er en sti.

Stier og kretser

- Vi kan definere mengder av stier *induktivt* på følgende måte.

Definisjon (Mengden av stier - induktivt)

- En sekvens som består av en node v er en sti.
- Hvis p er en sti, og det siste elementet i p er noden u

Stier og kretser

- Vi kan definere mengder av stier *induktivt* på følgende måte.

Definisjon (Mengden av stier - induktivt)

- En sekvens som består av en node v er en sti.
- Hvis p er en sti, og det siste elementet i p er noden u , og det går en kant fra u til v

Stier og kretser

- Vi kan definere mengder av stier *induktivt* på følgende måte.

Definisjon (Mengden av stier - induktivt)

- En sekvens som består av en node v er en sti.
- Hvis p er en sti, og det siste elementet i p er noden u , og det går en kant fra u til v , så er sekvensen pev en sti.

Stier og kretser

- Vi kan definere mengder av stier *induktivt* på følgende måte.

Definisjon (Mengden av stier - induktivt)

- En sekvens som består av en node v er en sti.
- Hvis p er en sti, og det siste elementet i p er noden u , og det går en kant fra u til v , så er sekvensen pev en sti.
- Mengden av stier er den minste mengden som oppfyller disse to kravene.

Stier og kretser

- Vi kan definere mengder av stier *induktivt* på følgende måte.

Definisjon (Mengden av stier - induktivt)

- En sekvens som består av en node v er en sti.
 - Hvis p er en sti, og det siste elementet i p er noden u , og det går en kant fra u til v , så er sekvensen pev en sti.
 - Mengden av stier er den minste mengden som oppfyller disse to kravene.
-
- Når vi nå har begrepet om en sti kan vi definere *sammenhengende grafer* mer presist.

Stier og kretser

- Vi kan definere mengder av stier *induktivt* på følgende måte.

Definisjon (Mengden av stier - induktivt)

- En sekvens som består av en node v er en sti.
 - Hvis p er en sti, og det siste elementet i p er noden u , og det går en kant fra u til v , så er sekvensen pev en sti.
 - Mengden av stier er den minste mengden som oppfyller disse to kravene.
-
- Når vi nå har begrepet om en sti kan vi definere *sammenhengende grafer* mer presist.

Definisjon (Sammenhengende)

En graf er *sammenhengende* hvis det for hvert par av noder u og v fins en sti fra u til v .

Stier og kretser

Stier og kretser

- Vi kan definere en ekvivalensrelasjon R på mengden av noder i en graf ved å si at uRv skal holde hvis det fins en sti fra u til v .

Stier og kretser

- Vi kan definere en ekvivalensrelasjon R på mengden av noder i en graf ved å si at uRv skal holde hvis det fins en sti fra u til v .
- Det er lett å sjekke at R er transitiv, refleksiv og symmetrisk.

Stier og kretser

- Vi kan definere en ekvivalensrelasjon R på mengden av noder i en graf ved å si at uRv skal holde hvis det fins en sti fra u til v .
- Det er lett å sjekke at R er transitiv, refleksiv og symmetrisk.
(Vi tegner og forklarer på tavla.)

Stier og kretser

- Vi kan definere en ekvivalensrelasjon R på mengden av noder i en graf ved å si at uRv skal holde hvis det fins en sti fra u til v .
- Det er lett å sjekke at R er transitiv, refleksiv og symmetrisk. (Vi tegner og forklarer på tavla.)
- Ekvivalensklassene definert av R kalles for *komponentene* til grafen.

Stier og kretser

- Vi kan definere en ekvivalensrelasjon R på mengden av noder i en graf ved å si at uRv skal holde hvis det fins en sti fra u til v .
- Det er lett å sjekke at R er transitiv, refleksiv og symmetrisk.
(Vi tegner og forklarer på tavla.)
- Ekvivalensklassene definert av R kalles for *komponentene* til grafen.
- En graf kan deles opp i “sammenhengende delgrafer” på en slik måte.

Stier og kretser

- Vi kan definere en ekvivalensrelasjon R på mengden av noder i en graf ved å si at uRv skal holde hvis det fins en sti fra u til v .
- Det er lett å sjekke at R er transitiv, refleksiv og symmetrisk.
(Vi tegner og forklarer på tavla.)
- Ekvivalensklassene definert av R kalles for *komponentene* til grafen.
- En graf kan deles opp i “sammenhengende delgrafer” på en slik måte.
- En sammenhengende graf er en graf som består av en komponent.

Stier og kretser

Definisjon (Eulersti/Eulerkrets)

Definisjon (Eulersti/Eulerkrets)

La G være en sammenhengende graf.

Definisjon (Eulersti/Eulerkrets)

La G være en sammenhengende graf. En *Eulersti* er en sti som inneholder hver kant fra G nøyaktig én gang.

Definisjon (Eulersti/Eulerkrets)

La G være en sammenhengende graf. En *Eulersti* er en sti som inneholder hver kant fra G nøyaktig én gang. En *Eulerkrets* er en Eulersti hvor den første og den siste noden sammenfaller.

Stier og kretser

Definisjon (Eulersti/Eulerkrets)

La G være en sammenhengende graf. En *Eulersti* er en sti som inneholder hver kant fra G nøyaktig én gang. En *Eulerkrets* er en Eulersti hvor den første og den siste noden sammenfaller. En sammenhengende graf som har en Eulerkrets kalles *Eulersk*.

Definisjon (Eulersti/Eulerkrets)

La G være en sammenhengende graf. En *Eulersti* er en sti som inneholder hver kant fra G nøyaktig én gang. En *Eulerkrets* er en Eulersti hvor den første og den siste noden sammenfaller. En sammenhengende graf som har en Eulerkrets kalles *Eulersk*. En sammenhengende graf som har en Eulersti, men *ikke* en Eulerkrets, kalles *semi-Eulersk*.

Definisjon (Eulersti/Eulerkrets)

La G være en sammenhengende graf. En *Eulersti* er en sti som inneholder hver kant fra G nøyaktig én gang. En *Eulerkrets* er en Eulersti hvor den første og den siste noden sammenfaller. En sammenhengende graf som har en Eulerkrets kalles *Eulersk*. En sammenhengende graf som har en Eulersti, men *ikke* en Eulerkrets, kalles *semi-Eulersk*.

- Noen kommentarer er på sin plass.

Definisjon (Eulersti/Eulerkrets)

La G være en sammenhengende graf. En *Eulersti* er en sti som inneholder hver kant fra G nøyaktig én gang. En *Eulerkrets* er en Eulersti hvor den første og den siste noden sammenfaller. En sammenhengende graf som har en Eulerkrets kalles *Eulersk*. En sammenhengende graf som har en Eulersti, men *ikke* en Eulerkrets, kalles *semi-Eulersk*.

- Noen kommentarer er på sin plass.
- I en Eulersti har vi lov til å gjenta noder, men ikke kanter.

Definisjon (Eulersti/Eulerkrets)

La G være en sammenhengende graf. En *Eulersti* er en sti som inneholder hver kant fra G nøyaktig én gang. En *Eulerkrets* er en Eulersti hvor den første og den siste noden sammenfaller. En sammenhengende graf som har en Eulerkrets kalles *Eulersk*. En sammenhengende graf som har en Eulersti, men *ikke* en Eulerkrets, kalles *semi-Eulersk*.

- Noen kommentarer er på sin plass.
- I en Eulersti har vi lov til å gjenta noder, men ikke kanter.
- Eulerstier kalles også for *Eulerveier*.

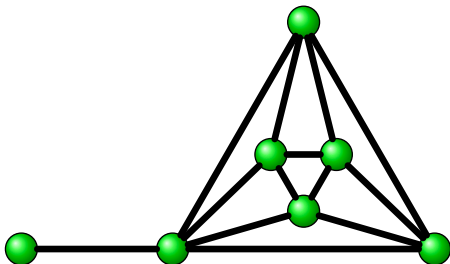
Stier og kretser

Stier og kretser

- Finner vi en Eulersti i denne grafen?

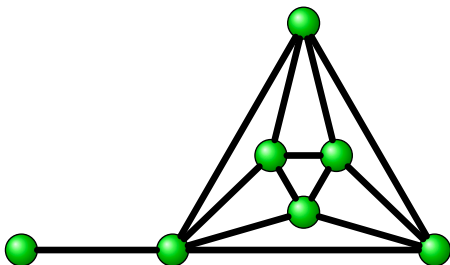
Stier og kretser

- Finner vi en Eulersti i denne grafen?



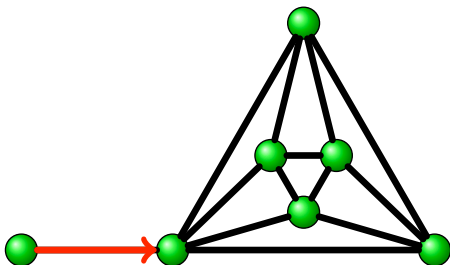
Stier og kretser

- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



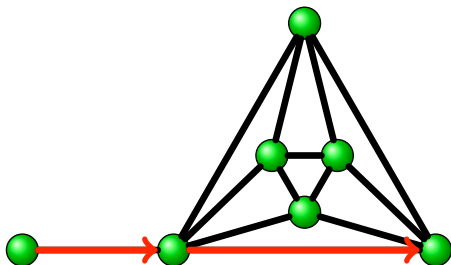
Stier og kretser

- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



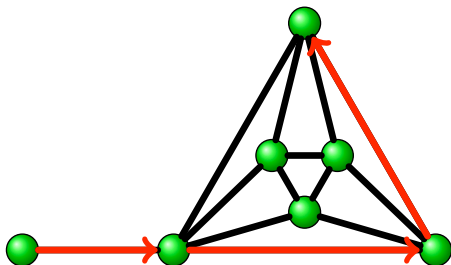
Stier og kretser

- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



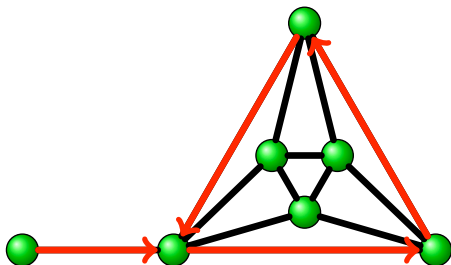
Stier og kretser

- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



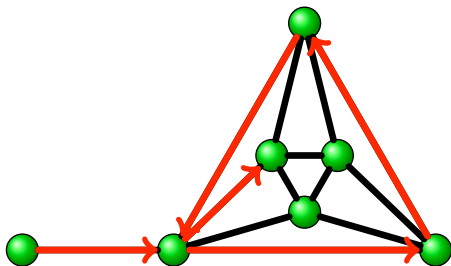
Stier og kretser

- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



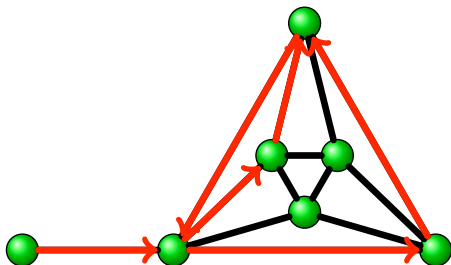
Stier og kretser

- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



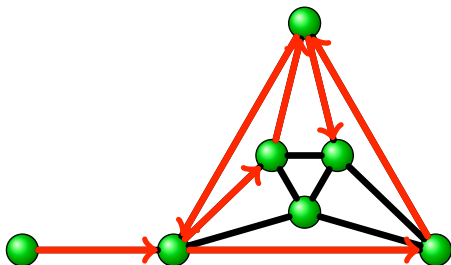
Stier og kretser

- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



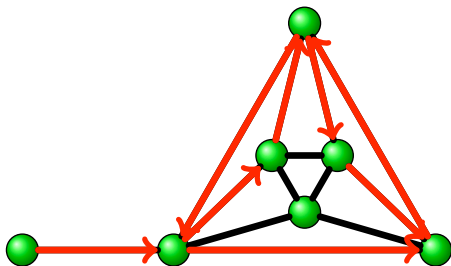
Stier og kretser

- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



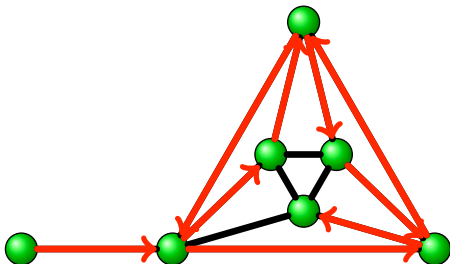
Stier og kretser

- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



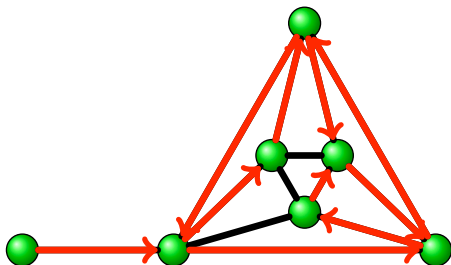
Stier og kretser

- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



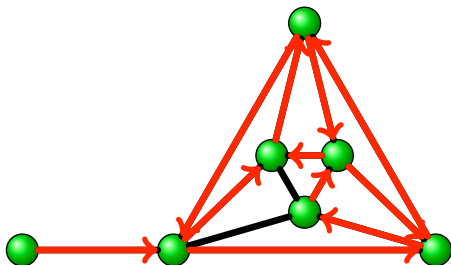
Stier og kretser

- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



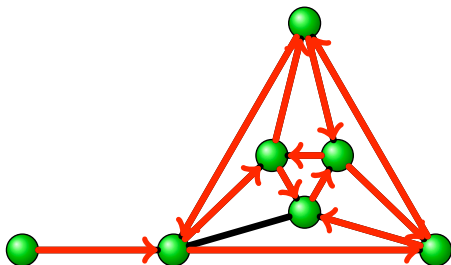
Stier og kretser

- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



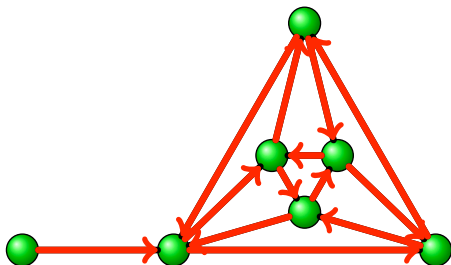
Stier og kretser

- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



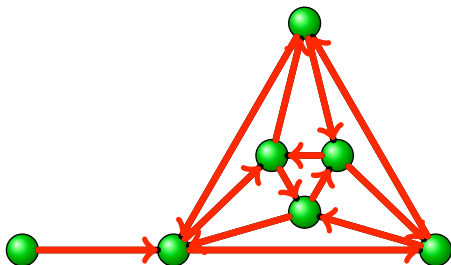
Stier og kretser

- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



Stier og kretser

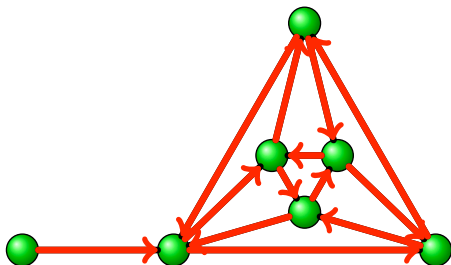
- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



- Vi fant en Eulersti.

Stier og kretser

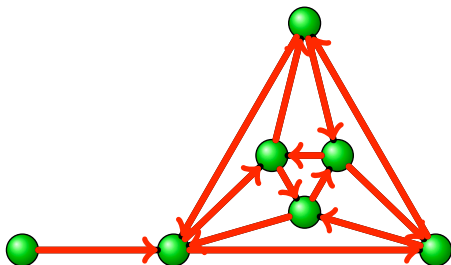
- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



- Vi fant en Eulersti.
- En Eulerkrets er riktignok umulig, på grunn av noden til venstre.

Stier og kretser

- Finner vi en Eulersti i denne grafen?
- Vi må i hvert fall begynne eller slutte i noden helt til venstre, siden den har grad 1.



- Vi fant en Eulersti.
- En Eulerkrets er riktignok umulig, på grunn av noden til venstre.
- Ved å ta bort denne, så får vi en Eulerkrets.

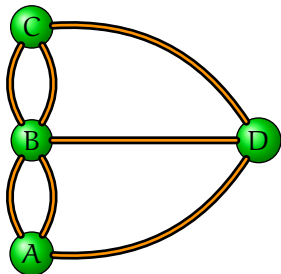
Tilbake til Königsberg

Tilbake til Königsberg

- Problemet om Königsbergs broer blir nå: Har grafen over Königsberg en Eulersti?

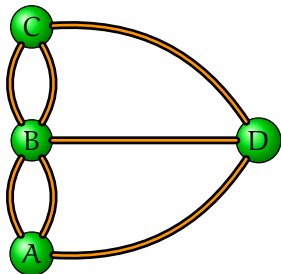
Tilbake til Königsberg

- Problemet om Königsbergs broer blir nå: Har grafen over Königsberg en Eulersti?
- En sti må gå innom noden A minst en gang.



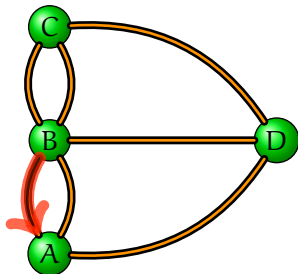
Tilbake til Königsberg

- Problemet om Königsbergs broer blir nå: Har grafen over Königsberg en Eulersti?
- En sti må gå innom noden A minst en gang.
- Hvis vi går inn i A og ut igjen



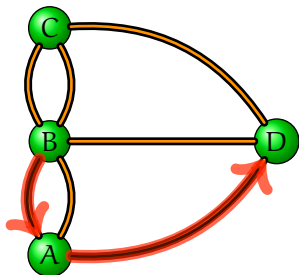
Tilbake til Königsberg

- Problemet om Königsbergs broer blir nå: Har grafen over Königsberg en Eulersti?
- En sti må gå innom noden A minst en gang.
- Hvis vi går inn i A og ut igjen



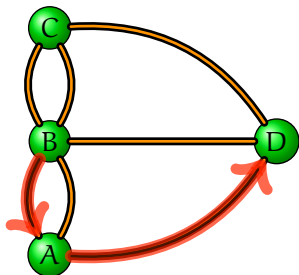
Tilbake til Königsberg

- Problemet om Königsbergs broer blir nå: Har grafen over Königsberg en Eulersti?
- En sti må gå innom noden A minst en gang.
- Hvis vi går inn i A og ut igjen



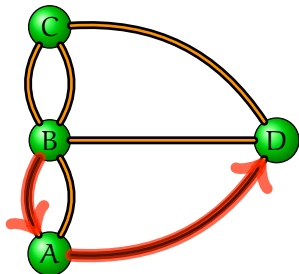
Tilbake til Königsberg

- Problemet om Königsbergs broer blir nå: Har grafen over Königsberg en Eulersti?
- En sti må gå innom noden A minst en gang.
- Hvis vi går inn i A og ut igjen, så gjenstår én kant.



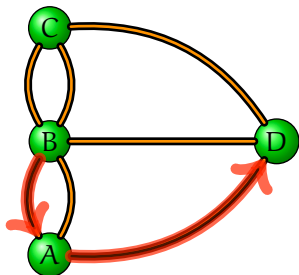
Tilbake til Königsberg

- Problemet om Königsbergs broer blir nå: Har grafen over Königsberg en Eulersti?
- En sti må gå innom noden A minst en gang.
- Hvis vi går inn i A og ut igjen, så gjenstår én kant.
- For å gå over alle tre kantene som ligger inntil A, så må man enten begynne eller slutte i A.



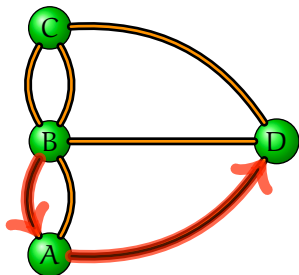
Tilbake til Königsberg

- Problemet om Königsbergs broer blir nå: Har grafen over Königsberg en Eulersti?
- En sti må gå innom noden A minst en gang.
- Hvis vi går inn i A og ut igjen, så gjenstår én kant.
- For å gå over alle tre kantene som ligger inntil A, så må man enten begynne eller slutte i A.
- Det samme gjelder for C



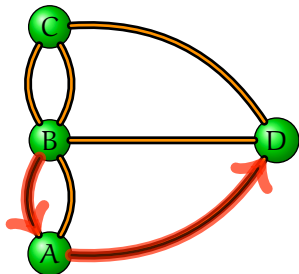
Tilbake til Königsberg

- Problemet om Königsbergs broer blir nå: Har grafen over Königsberg en Eulersti?
- En sti må gå innom noden A minst en gang.
- Hvis vi går inn i A og ut igjen, så gjenstår én kant.
- For å gå over alle tre kantene som ligger inntil A, så må man enten begynne eller slutte i A.
- Det samme gjelder for C, og D



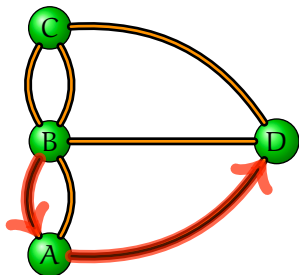
Tilbake til Königsberg

- Problemet om Königsbergs broer blir nå: Har grafen over Königsberg en Eulersti?
- En sti må gå innom noden A minst en gang.
- Hvis vi går inn i A og ut igjen, så gjenstår én kant.
- For å gå over alle tre kantene som ligger inntil A, så må man enten begynne eller slutte i A.
- Det samme gjelder for C, og D og B



Tilbake til Königsberg

- Problemet om Königsbergs broer blir nå: Har grafen over Königsberg en Eulersti?
- En sti må gå innom noden A minst en gang.
- Hvis vi går inn i A og ut igjen, så gjenstår én kant.
- For å gå over alle tre kantene som ligger inntil A, så må man enten begynne eller slutte i A.
- Det samme gjelder for C, og D og B, siden de alle har *odde grad*.



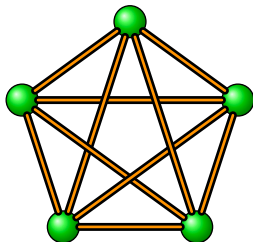
Stier og kretser

Stier og kretser

- Vi ser om vi finner en Eulerkrets i den komplette grafen med fem noder.

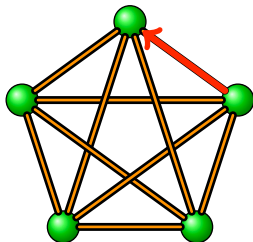
Stier og kretser

- Vi ser om vi finner en Eulerkrets i den komplette grafen med fem noder.



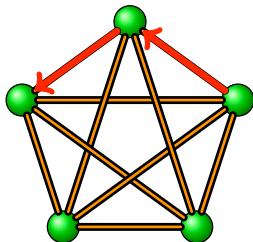
Stier og kretser

- Vi ser om vi finner en Eulerkrets i den komplette grafen med fem noder.



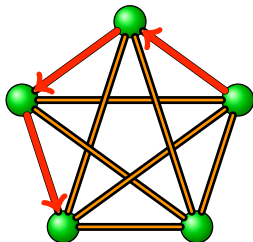
Stier og kretser

- Vi ser om vi finner en Eulerkrets i den komplette grafen med fem noder.



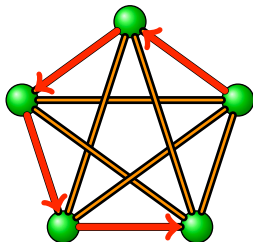
Stier og kretser

- Vi ser om vi finner en Eulerkrets i den komplette grafen med fem noder.



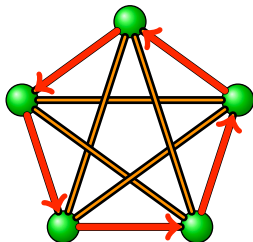
Stier og kretser

- Vi ser om vi finner en Eulerkrets i den komplette grafen med fem noder.



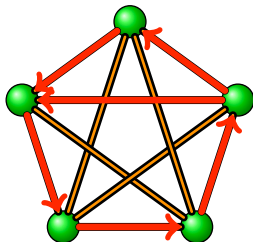
Stier og kretser

- Vi ser om vi finner en Eulerkrets i den komplette grafen med fem noder.



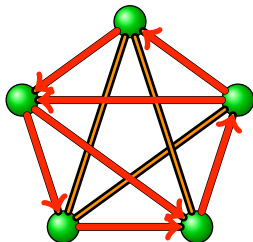
Stier og kretser

- Vi ser om vi finner en Eulerkrets i den komplette grafen med fem noder.



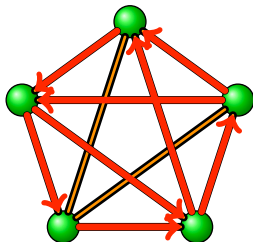
Stier og kretser

- Vi ser om vi finner en Eulerkrets i den komplette grafen med fem noder.



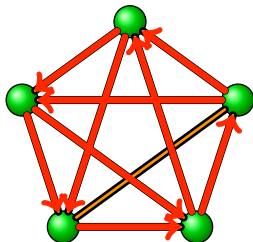
Stier og kretser

- Vi ser om vi finner en Eulerkrets i den komplette grafen med fem noder.



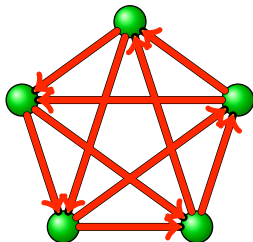
Stier og kretser

- Vi ser om vi finner en Eulerkrets i den komplette grafen med fem noder.



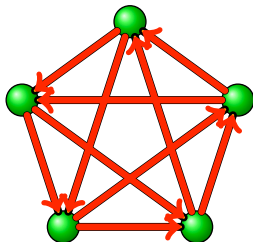
Stier og kretser

- Vi ser om vi finner en Eulerkrets i den komplette grafen med fem noder.



Stier og kretser

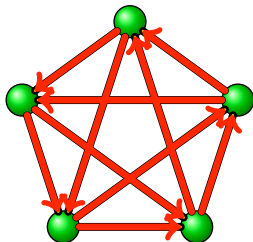
- Vi ser om vi finner en Eulerkrets i den komplette grafen med fem noder.



- Det var greit.

Stier og kretser

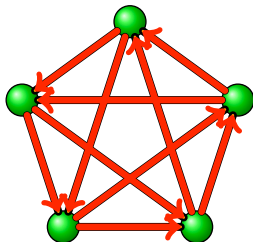
- Vi ser om vi finner en Eulerkrets i den komplette grafen med fem noder.



- Det var greit.
- Vi observerer at hver node har grad 4, et partall.

Stier og kretser

- Vi ser om vi finner en Eulerkrets i den komplette grafen med fem noder.



- Det var greit.
- Vi observerer at hver node har grad 4, et partall.
- Vi skal nå se at det er en sammenheng mellom eksistensen av Eulerstier/-kretser og hvorvidt gradene til nodene er partall.

Stier og kretser

Stier og kretser

- Anta at vi har en graf med en Eulerkrets.

Stier og kretser

- Anta at vi har en graf med en Eulerkrets.
- Hva er da gradene til nodene i grafen?

Stier og kretser

- Anta at vi har en graf med en Eulerkrets.
- Hva er da gradene til nodene i grafen?
- Vi kan skrive Eulerkretsen som sekvensen $v_0 e_1 v_1 e_2 v_2 \dots e_n v_n$, hvor $v_n = v_0$.

Stier og kretser

- Anta at vi har en graf med en Eulerkrets.
- Hva er da gradene til nodene i grafen?
- Vi kan skrive Eulerkretsen som sekvensen $v_0 e_1 v_1 e_2 v_2 \dots e_n v_n$, hvor $v_n = v_0$.
- Graden til en node u må være 2 ganger antall ganger den forekommer i sekvensen.

Stier og kretser

- Anta at vi har en graf med en Eulerkrets.
- Hva er da gradene til nodene i grafen?
- Vi kan skrive Eulerkretsen som sekvensen $v_0 e_1 v_1 e_2 v_2 \dots e_n v_n$, hvor $v_n = v_0$.
- Graden til en node u må være 2 ganger antall ganger den forekommer i sekvensen.
- Enhver node i grafen må da ha grad lik et partall.

Stier og kretser

- Anta at vi har en graf med en Eulerkrets.
- Hva er da gradene til nodene i grafen?
- Vi kan skrive Eulerkretsen som sekvensen $v_0 e_1 v_1 e_2 v_2 \dots e_n v_n$, hvor $v_n = v_0$.
- Graden til en node u må være 2 ganger antall ganger den forekommer i sekvensen.
- Enhver node i grafen må da ha grad lik et partall.
- Vårt neste teorem sier det omvendte, nemlig at hvis hver node i en graf har grad som er et partall, så må grafen inneholde en Eulerkrets.

Stier og kretser

Teorem

Teorem

La G være en sammenhengende graf.

Teorem

La G være en sammenhengende graf.

1. Hvis graden til enhver node i G er et partall, så inneholder G en Eulerkrets.

Teorem

La G være en sammenhengende graf.

1. Hvis graden til enhver node i G er et partall, så inneholder G en Eulerkrets.
2. Hvis nøyaktig to noder i G har *odde* grad, så inneholder G en Eulersti

Teorem

La G være en sammenhengende graf.

1. Hvis graden til enhver node i G er et partall, så inneholder G en Eulerkrets.
2. Hvis nøyaktig to noder i G har *odde* grad, så inneholder G en Eulersti som begynner i en node av odde grad og som slutter i en node av odde grad.

Teorem

La G være en sammenhengende graf.

1. Hvis graden til enhver node i G er et partall, så inneholder G en Eulerkrets.
2. Hvis nøyaktig to noder i G har *odde* grad, så inneholder G en Eulersti som begynner i en node av odde grad og som slutter i en node av odde grad.
3. Hvis G har mer enn to noder av odde grad, så inneholder grafen *ikke* en Eulersti.

Stier og kretser

Stier og kretser

- Boka beviser påstand 1 ved å gi en algoritme for å konstruere en Eulerkrets for en graf hvor hver node har et partall som grad.

Stier og kretser

- Boka beviser påstand 1 ved å gi en algoritme for å konstruere en Eulerkrets for en graf hvor hver node har et partall som grad.
- Vi må da argumentere for at algoritmen er *korrekt*, at den alltid vil finne en Eulerkrets.

Stier og kretser

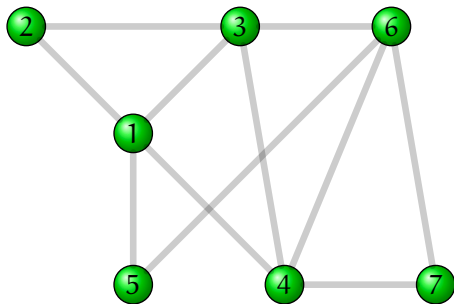
- Boka beviser påstand 1 ved å gi en algoritme for å konstruere en Eulerkrets for en graf hvor hver node har et partall som grad.
- Vi må da argumentere for at algoritmen er *korrekt*, at den alltid vil finne en Eulerkrets.
- I dette tilfellet er det ganske greit å se.

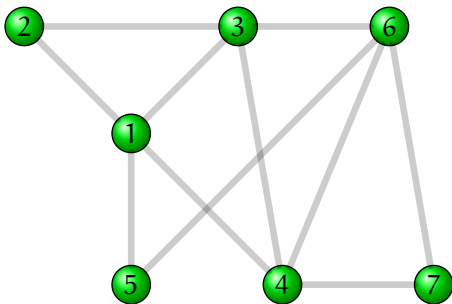
Stier og kretser

- Boka beviser påstand 1 ved å gi en algoritme for å konstruere en Eulerkrets for en graf hvor hver node har et partall som grad.
- Vi må da argumentere for at algoritmen er *korrekt*, at den alltid vil finne en Eulerkrets.
- I dette tilfellet er det ganske greit å se.
- Vi skal se at påstand 1 medfører påstand 2, at en graf som inneholder nøyaktig to noder av odde grad inneholder en Eulersti.

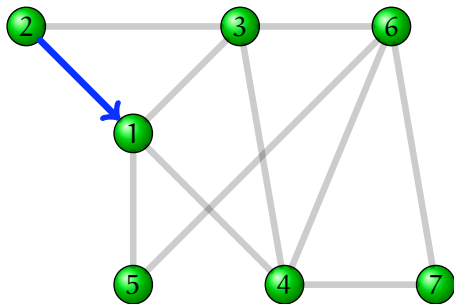
Stier og kretser

- Boka beviser påstand 1 ved å gi en algoritme for å konstruere en Eulerkrets for en graf hvor hver node har et partall som grad.
- Vi må da argumentere for at algoritmen er *korrekt*, at den alltid vil finne en Eulerkrets.
- I dette tilfellet er det ganske greit å se.
- Vi skal se at påstand 1 medfører påstand 2, at en graf som inneholder nøyaktig to noder av odde grad inneholder en Eulersti.
- La oss se på intuisjonen bak algoritmen.

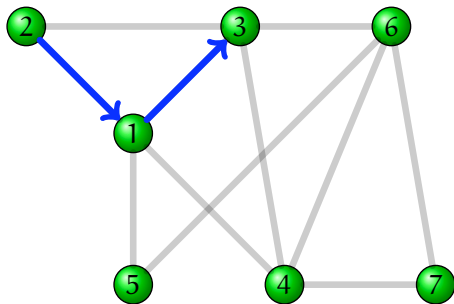




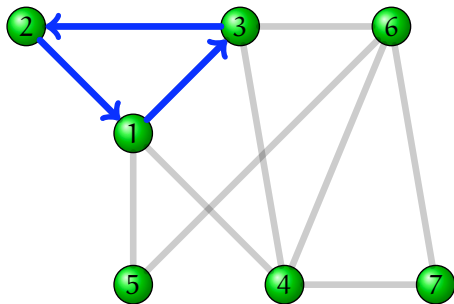
- Hvis vi begynner i node 2, så finner vi kretsen 2132.



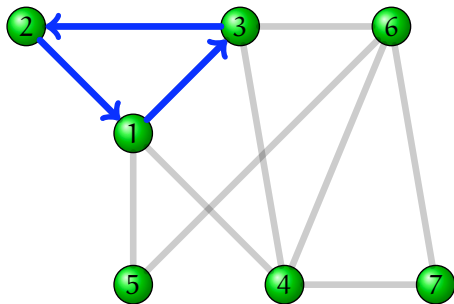
- Hvis vi begynner i node 2, så finner vi kretsen 2132.



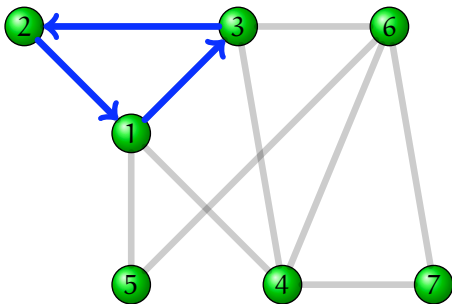
- Hvis vi begynner i node 2, så finner vi kretsen 2132.



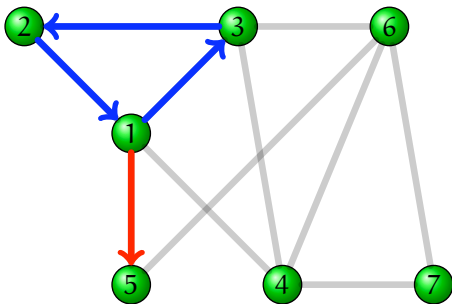
- Hvis vi begynner i node 2, så finner vi kretsen 2132.



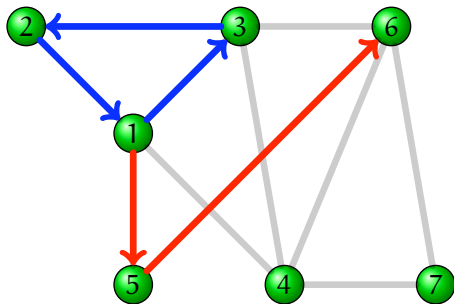
- Hvis vi begynner i node 2, så finner vi kretsen 2132.
- Siden ubrukte kanter ligger inntil både node 1 og 3, forsøker vi å *utvide* vår nåværende krets ved å legge til nye kretser.



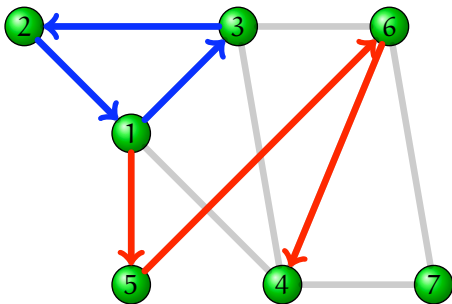
- Hvis vi begynner i node 2, så finner vi kretsen **2132**.
- Siden ubrukte kanter ligger inntil både node 1 og 3, forsøker vi å *utvide* vår nåværende krets ved å legge til nye kretser.
- Hvis vi begynner i node 1, så finner vi kretsen **15641**.



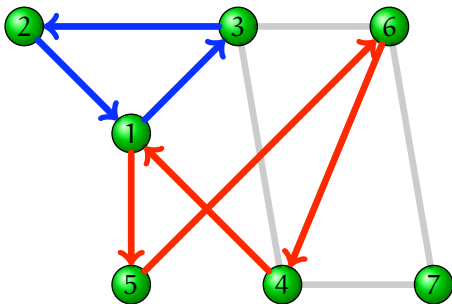
- Hvis vi begynner i node 2, så finner vi kretsen **2132**.
- Siden ubrukte kanter ligger inntil både node 1 og 3, forsøker vi å *utvide* vår nåværende krets ved å legge til nye kretser.
- Hvis vi begynner i node 1, så finner vi kretsen **15641**.



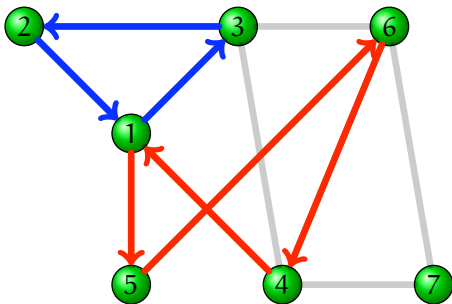
- Hvis vi begynner i node 2, så finner vi kretsen **2132**.
- Siden ubrukte kanter ligger inntil både node 1 og 3, forsøker vi å *utvide* vår nåværende krets ved å legge til nye kretser.
- Hvis vi begynner i node 1, så finner vi kretsen **15641**.



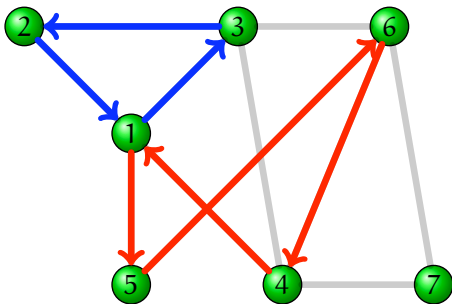
- Hvis vi begynner i node 2, så finner vi kretsen **2132**.
- Siden ubrukte kanter ligger inntil både node 1 og 3, forsøker vi å *utvide* vår nåværende krets ved å legge til nye kretser.
- Hvis vi begynner i node 1, så finner vi kretsen **15641**.



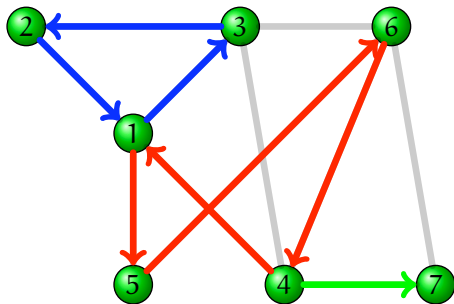
- Hvis vi begynner i node 2, så finner vi kretsen **2132**.
- Siden ubrukte kanter ligger inntil både node 1 og 3, forsøker vi å *utvide* vår nåværende krets ved å legge til nye kretser.
- Hvis vi begynner i node 1, så finner vi kretsen **15641**.



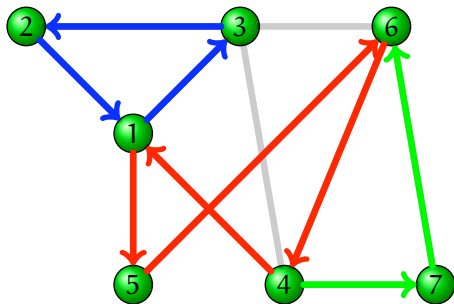
- Hvis vi begynner i node 2, så finner vi kretsen **2132**.
- Siden ubrukte kanter ligger inntil både node 1 og 3, forsøker vi å *utvide* vår nåværende krets ved å legge til nye kretser.
- Hvis vi begynner i node 1, så finner vi kretsen **15641**.
- Vi kan *sette sammen* disse kretsene og få kretsen **21564132**.



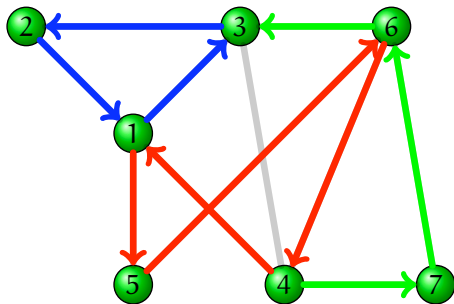
- Hvis vi begynner i node 2, så finner vi kretsen **2132**.
- Siden ubrukte kanter ligger inntil både node 1 og 3, forsøker vi å *utvide* vår nåværende krets ved å legge til nye kretser.
- Hvis vi begynner i node 1, så finner vi kretsen **15641**.
- Vi kan *sette sammen* disse kretsene og få kretsen **21564132**.
- Vi finner en krets til, **47634**.



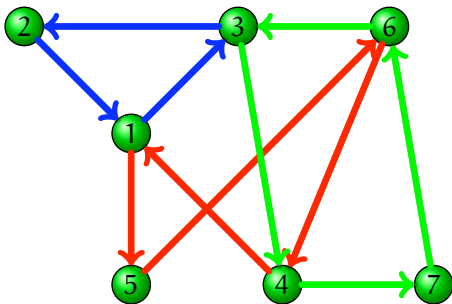
- Hvis vi begynner i node 2, så finner vi kretsen **2132**.
- Siden ubrukte kanter ligger inntil både node 1 og 3, forsøker vi å *utvide* vår nåværende krets ved å legge til nye kretser.
- Hvis vi begynner i node 1, så finner vi kretsen **15641**.
- Vi kan *sette sammen* disse kretsene og få kretsen **21564132**.
- Vi finner en krets til, **47634**.



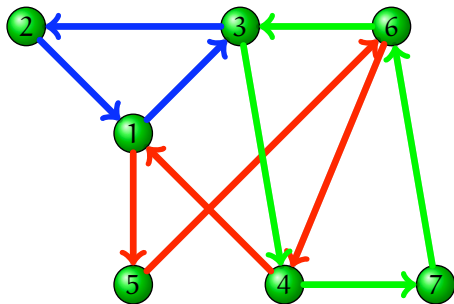
- Hvis vi begynner i node 2, så finner vi kretsen **2132**.
- Siden ubrukte kanter ligger inntil både node 1 og 3, forsøker vi å *utvide* vår nåværende krets ved å legge til nye kretser.
- Hvis vi begynner i node 1, så finner vi kretsen **15641**.
- Vi kan *sette sammen* disse kretsene og få kretsen **21564132**.
- Vi finner en krets til, **47634**.



- Hvis vi begynner i node 2, så finner vi kretsen **2132**.
- Siden ubrukte kanter ligger inntil både node 1 og 3, forsøker vi å *utvide* vår nåværende krets ved å legge til nye kretser.
- Hvis vi begynner i node 1, så finner vi kretsen **15641**.
- Vi kan *sette sammen* disse kretsene og få kretsen **21564132**.
- Vi finner en krets til, **47634**.



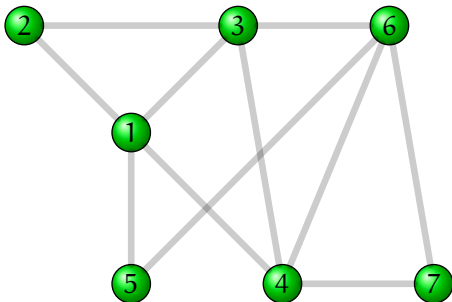
- Hvis vi begynner i node 2, så finner vi kretsen **2132**.
- Siden ubrukte kanter ligger inntil både node 1 og 3, forsøker vi å *utvide* vår nåværende krets ved å legge til nye kretser.
- Hvis vi begynner i node 1, så finner vi kretsen **15641**.
- Vi kan *sette sammen* disse kretsene og få kretsen **21564132**.
- Vi finner en krets til, **47634**.



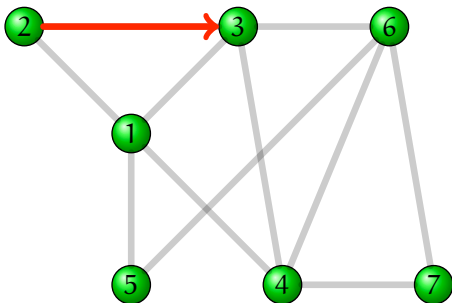
- Hvis vi begynner i node 2, så finner vi kretsen **2132**.
- Siden ubrukte kanter ligger inntil både node 1 og 3, forsøker vi å *utvide* vår nåværende krets ved å legge til nye kretser.
- Hvis vi begynner i node 1, så finner vi kretsen **15641**.
- Vi kan *sette sammen* disse kretsene og få kretsen **21564132**.
- Vi finner en krets til, **47634**.
- Vi får da kretsen **215647634132** som er en Eulerkrets.

- Her er en annen måte å sette sammen flere kretser til en Eulerkrets på, med den samme grafen.

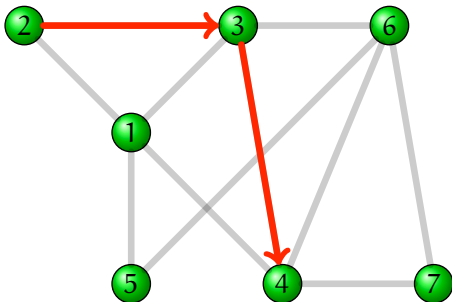
- Her er en annen måte å sette sammen flere kretser til en Eulerkrets på, med den samme grafen.



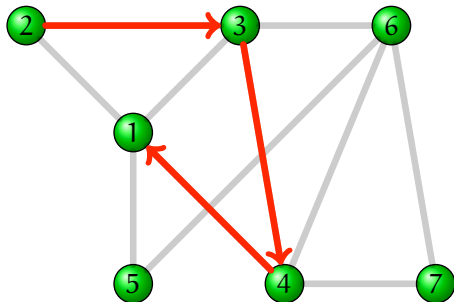
- Her er en annen måte å sette sammen flere kretser til en Eulerkrets på, med den samme grafen.



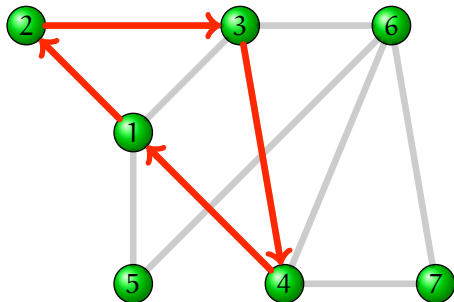
- Her er en annen måte å sette sammen flere kretser til en Eulerkrets på, med den samme grafen.



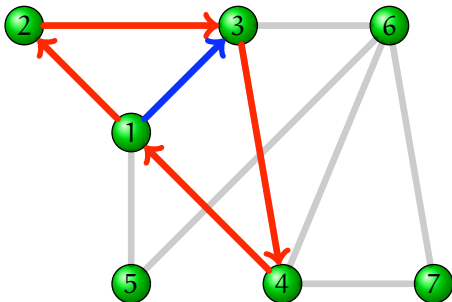
- Her er en annen måte å sette sammen flere kretser til en Eulerkrets på, med den samme grafen.



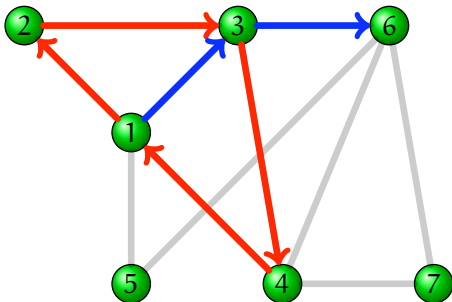
- Her er en annen måte å sette sammen flere kretser til en Eulerkrets på, med den samme grafen.



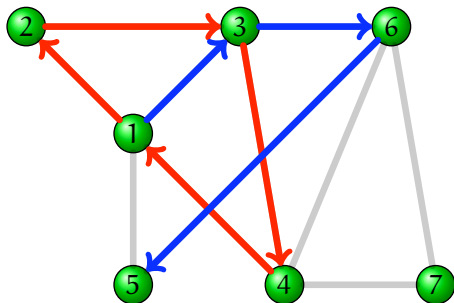
- Her er en annen måte å sette sammen flere kretser til en Eulerkrets på, med den samme grafen.



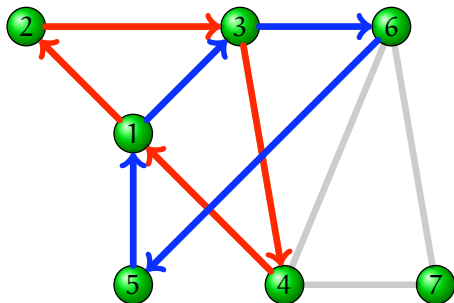
- Her er en annen måte å sette sammen flere kretser til en Eulerkrets på, med den samme grafen.



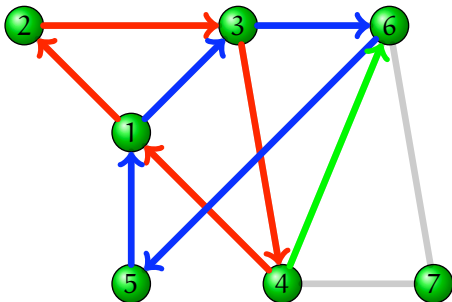
- Her er en annen måte å sette sammen flere kretser til en Eulerkrets på, med den samme grafen.



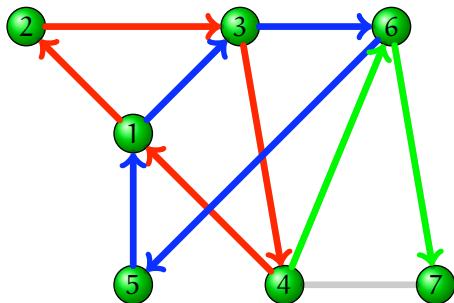
- Her er en annen måte å sette sammen flere kretser til en Eulerkrets på, med den samme grafen.



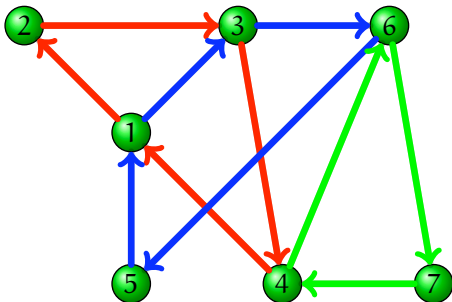
- Her er en annen måte å sette sammen flere kretser til en Eulerkrets på, med den samme grafen.



- Her er en annen måte å sette sammen flere kretser til en Eulerkrets på, med den samme grafen.



- Her er en annen måte å sette sammen flere kretser til en Eulerkrets på, med den samme grafen.



Stier og kretser

Stier og kretser

- Hvis vi begynner med en sti som kun inneholder én node og utvider stien stegvis ved å legge til kanter og stier

Stier og kretser

- Hvis vi begynner med en sti som kun inneholder én node og utvider stien stegvis ved å legge til kanter og stier, så må vi før eller siden komme tilbake til den første noden i stien

Stier og kretser

- Hvis vi begynner med en sti som kun inneholder én node og utvider stien stegvis ved å legge til kanter og stier, så må vi før eller siden komme tilbake til den første noden i stien, slik at vi får en *krets*.

Stier og kretser

- Hvis vi begynner med en sti som kun inneholder én node og utvider stien stegvis ved å legge til kanter og stier, så må vi før eller siden komme tilbake til den første noden i stien, slik at vi får en *krets*.
- Grunnen er at hver node har grad som er et partall.

Stier og kretser

- Hvis vi begynner med en sti som kun inneholder én node og utvider stien stegvis ved å legge til kanter og stier, så må vi før eller siden komme tilbake til den første noden i stien, slik at vi får en *krets*.
- Grunnen er at hver node har grad som er et partall.
- Hver gang vi “går inn i” en node i stien, som ikke er startnoden, så må finnes en ubrukt kant slik at vi kan “gå ut” igjen.

Stier og kretser

- Hvis vi begynner med en sti som kun inneholder én node og utvider stien stegvis ved å legge til kanter og stier, så må vi før eller siden komme tilbake til den første noden i stien, slik at vi får en *krets*.
- Grunnen er at hver node har grad som er et partall.
- Hver gang vi “går inn i” en node i stien, som ikke er startnoden, så må finnes en ubrukt kant slik at vi kan “gå ut” igjen.
- Før eller siden må vi komme tilbake til startnoden.

Stier og kretser

Stier og kretser

1. Input en Eulergraf G med noder V og kanter E

Stier og kretser

1. Input en Eulergraf G med noder V og kanter E
2. krets \leftarrow en node fra V

Stier og kretser

1. Input en Eulergraf G med noder V og kanter E
2. krets \leftarrow en node fra V
3. **While** $E \neq \emptyset$ **do**

Stier og kretser

1. Input en Eulergraf G med noder V og kanter E
2. krets \leftarrow en node fra V
3. **While** $E \neq \emptyset$ **do**
 - 3.1. $i \leftarrow$ den første noden i krets med en kant fra E som ligger inntil i

Stier og kretser

1. Input en Eulergraf G med noder V og kanter E
2. krets \leftarrow en node fra V
3. **While** $E \neq \emptyset$ **do**
 - 3.1. $i \leftarrow$ den første noden i krets med en kant fra E som ligger inntil i
 - 3.2. $v \leftarrow i$; nykrets $\leftarrow i$

Stier og kretser

1. Input en Eulergraf G med noder V og kanter E
2. krets \leftarrow en node fra V
3. **While** $E \neq \emptyset$ **do**
 - 3.1. $i \leftarrow$ den første noden i krets med en kant fra E som ligger inntil i
 - 3.2. $v \leftarrow i$; nykrets $\leftarrow i$
 - 3.3. **Repeat**

until ingen kant fra E ligger inntil v

Stier og kretser

1. Input en Eulergraf G med noder V og kanter E
 2. krets \leftarrow en node fra V
 3. **While** $E \neq \emptyset$ **do**
 - 3.1. $i \leftarrow$ den første noden i krets med en kant fra E som ligger inntil i
 - 3.2. $v \leftarrow i$; nykrets $\leftarrow i$
 - 3.3. **Repeat**
 - 3.3.1. $e \leftarrow$ en kant fra E som ligger inntil v
- until** ingen kant fra E ligger inntil v

Stier og kretser

1. Input en Eulergraf G med noder V og kanter E
 2. krets \leftarrow en node fra V
 3. **While** $E \neq \emptyset$ **do**
 - 3.1. $i \leftarrow$ den første noden i krets med en kant fra E som ligger inntil i
 - 3.2. $v \leftarrow i$; nykrets $\leftarrow i$
 - 3.3. **Repeat**
 - 3.3.1. $e \leftarrow$ en kant fra E som ligger inntil v
 - 3.3.2. $v \leftarrow$ noden som er nabo med v via e
- until** ingen kant fra E ligger inntil v

Stier og kretser

1. Input en Eulergraf G med noder V og kanter E
 2. krets \leftarrow en node fra V
 3. **While** $E \neq \emptyset$ **do**
 - 3.1. $i \leftarrow$ den første noden i krets med en kant fra E som ligger inntil i
 - 3.2. $v \leftarrow i$; nykrets $\leftarrow i$
 - 3.3. **Repeat**
 - 3.3.1. $e \leftarrow$ en kant fra E som ligger inntil v
 - 3.3.2. $v \leftarrow$ noden som er nabo med v via e
 - 3.3.3. nykrets \leftarrow sammensetningen av nykrets og e og v
- until** ingen kant fra E ligger inntil v

Stier og kretser

1. Input en Eulergraf G med noder V og kanter E
2. krets \leftarrow en node fra V
3. **While** $E \neq \emptyset$ **do**
 - 3.1. $i \leftarrow$ den første noden i krets med en kant fra E som ligger inntil i
 - 3.2. $v \leftarrow i$; nykrets $\leftarrow i$
 - 3.3. **Repeat**
 - 3.3.1. $e \leftarrow$ en kant fra E som ligger inntil v
 - 3.3.2. $v \leftarrow$ noden som er nabo med v via e
 - 3.3.3. nykrets \leftarrow sammensetningen av nykrets og e og v
 - 3.3.4. $E \leftarrow E - \{e\}$
- until** ingen kant fra E ligger inntil v

Stier og kretser

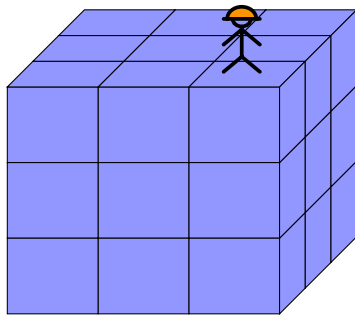
1. Input en Eulergraf G med noder V og kanter E
2. krets \leftarrow en node fra V
3. **While** $E \neq \emptyset$ **do**
 - 3.1. $i \leftarrow$ den første noden i krets med en kant fra E som ligger inntil i
 - 3.2. $v \leftarrow i$; nykrets $\leftarrow i$
 - 3.3. **Repeat**
 - 3.3.1. $e \leftarrow$ en kant fra E som ligger inntil v
 - 3.3.2. $v \leftarrow$ noden som er nabo med v via e
 - 3.3.3. nykrets \leftarrow sammensetningen av nykrets og e og v
 - 3.3.4. $E \leftarrow E - \{e\}$
 - until** ingen kant fra E ligger inntil v
- 3.4. krets \leftarrow sammensetningen av krets før i , nykrets og krets etter i

Stier og kretser

1. Input en Eulergraf G med noder V og kanter E
 2. krets \leftarrow en node fra V
 3. **While** $E \neq \emptyset$ **do**
 - 3.1. $i \leftarrow$ den første noden i krets med en kant fra E som ligger inntil i
 - 3.2. $v \leftarrow i$; nykrets $\leftarrow i$
 - 3.3. **Repeat**
 - 3.3.1. $e \leftarrow$ en kant fra E som ligger inntil v
 - 3.3.2. $v \leftarrow$ noden som er nabo med v via e
 - 3.3.3. nykrets \leftarrow sammensetningen av nykrets og e og v
 - 3.3.4. $E \leftarrow E - \{e\}$
 - until** ingen kant fra E ligger inntil v
 - 3.4. krets \leftarrow sammensetningen av krets før i , nykrets og krets etter i
4. Output krets

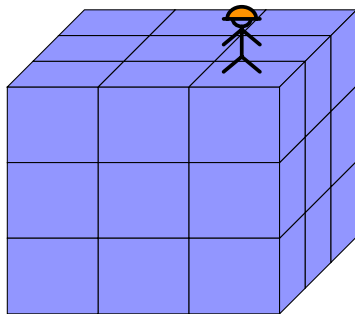
En nøtt om en maur og en kube

En nøtt om en maur og en kube



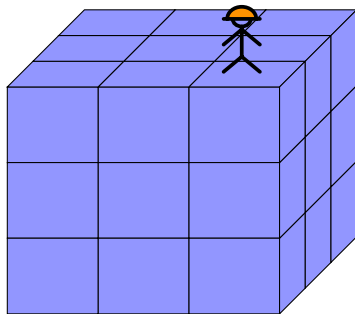
- Anta at en maur sitter på utsiden av en $3 \times 3 \times 3$ -kub.

En nøtt om en maur og en kube



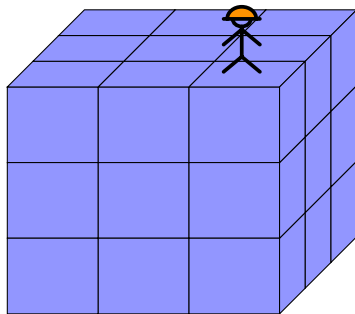
- Anta at en maur sitter på utsiden av en $3 \times 3 \times 3$ -kube.
- Mauren spiser seg inn i den første lille kuben (av i 27 kuber).

En nøtt om en maur og en kube



- Anta at en maur sitter på utsiden av en $3 \times 3 \times 3$ -kubee.
- Mauren spiser seg inn i den første lille kubee (av i 27 kubee).
- Er det mulig for mauren å spise seg gjennom alle kubee og så komme ut igjen samme sted?

En nøtt om en maur og en kube



- Anta at en maur sitter på utsiden av en $3 \times 3 \times 3$ -kube.
- Mauren spiser seg inn i den første lille kuben (av i 27 kuber).
- Er det mulig for mauren å spise seg gjennom alle kubene og så komme ut igjen samme sted?
- Mauren har kun lov til å gå en kube av gangen og ikke på skrå.