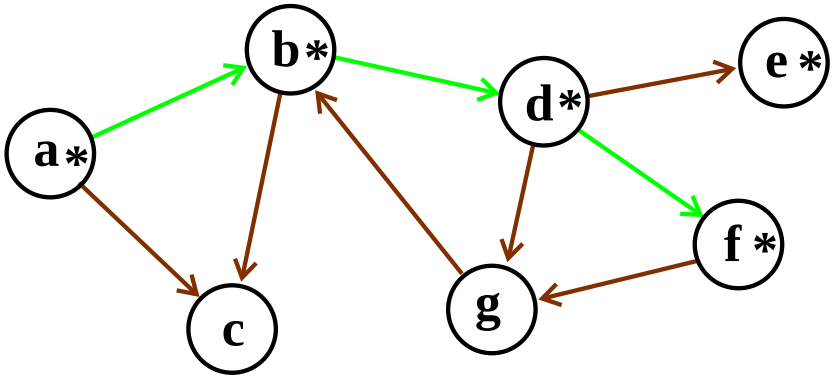
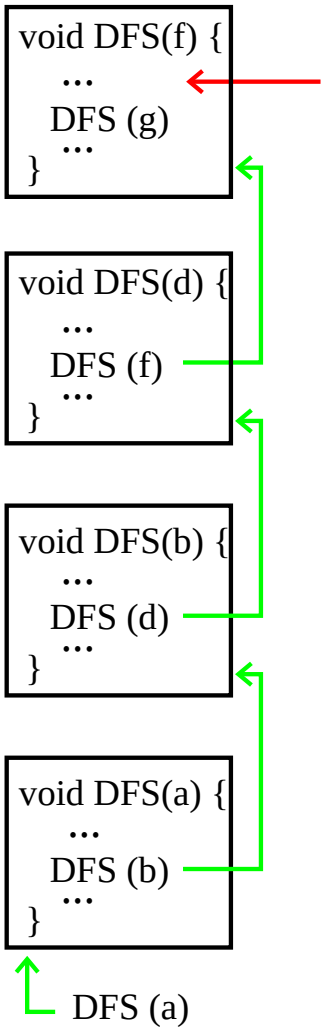
Dybde-først søk

- Generalisering av prefiks traversering for trær.
- Vi starter i en node v og traverserer rekursivt alle nabonodene.
- Rekursjonen gjør at vi undersøker alle noder som kan nåes fra første etterfølger til v, før vi undersøker neste etterfølger til v.
- For en vilkårlig graf må vi passe på å unngå løkker:
  - Markerer nodene som besøkt etterhvert som de behandles, og kaller rekursivt videre bare for umerkede noder.

```
void dybdeFørstSøk(Node v) {
    v.merke = true;
    for < hver nabo w til v > {
        if (!w.merke) {
            dybdeFørstSøk(w);
        }
    }
}
```

Midtveis i en dybde-først traversering ser kanskje kjeden av rekursive metodekall og besøkt-merkene i nodene slik ut:



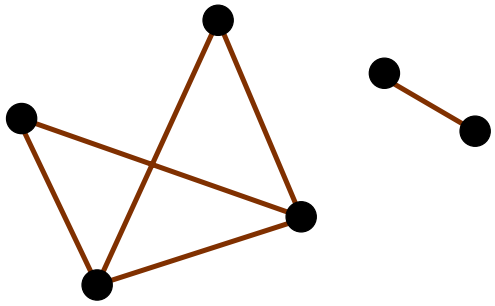


- Noder merket '*' er besøkte.
- Node e er allerede behandlet ferdig.
- Nå behandles node f
- Den kommer til å kalle DFS(g)
- DFS(g) har ingen ikke-besøkte etterfølgere.
- Dermed må algoritmen "trekke seg tilbake".
- Først i DFS(b) finnes det en ikke-besøkt etterfølger.

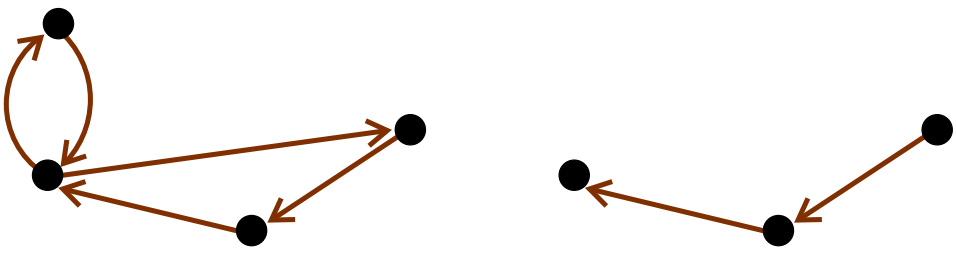
Er grafen sammenhengende?

Hvis grafen ikke er sammenhengende, kan vi foreta nye dybde-først søk fra noder som ikke er besøkte, inntil alle nodene er behandlet.

- En urettet graf er sammenhengende hvis og bare hvis et dybde-først søk som starter i en tilfeldig node, besøker alle nodene i grafen.



- En rettet graf er (sterkt) sammenhengende hvis og bare hvis vi fra hver eneste node v klarer å besøke alle de andre nodene i grafen ved et dybde-først søk fra v.



Løkkeleting

- Vi kan bruke dybde-først søk til å sjekke om en graf har løkker.
- Vi trenger da tre verdier til tilstandsvariablen:
  usett, igang og ferdig (besøkt).

```
void løkkeLet(Node v) {
    if (v.tilstand == igang) {
        < Løkke er funnet >
    } else if (v.tilstand == usett) {
        v.tilstand = igang;
        for < hver nabo w til v > {
            løkkeLet(w);
        }
        v.tilstand = ferdig;
    }
}
```

- Metoden bygger på at de nodene der kall er i gang, alltid ligger på en rett vei fra startnoden.
- Må passe på å gjøre nye startkall inntil metoden er kalt i alle nodene.
- Vil alltid finne en løkke dersom det eksisterer en!

- Metoden for løkkeleting kan også gi oss en topologisk sortering (med nodene skrevet ut i omvendt rekkefølge) dersom grafen ikke har løkker.
- Det får vi til ved å skrive ut noden like før vi trekker oss tilbake (idet vi er ferdig med kallet).
- Dette er riktig tidspunkt å skrive ut noden fordi:
  - Vi har sjekket alle etterfølgerne til noden.
  - Disse var enten usette (og da har vi skrevet dem ut i det vi trakk oss tilbake fra dem), eller ferdige (og da var de skrevet ut tidligere).

  Dersom vi fant en node som var igang, har vi funnet en løkke ...