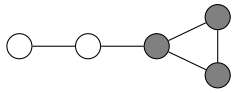
Sykler i grafer 11 poeng

(a) Anta at grafen $G = (V, E)$ er en *urettet* graf. Grafen inneholder en sykel hvis det finnes en sti fra en node u tilbake til samme node u som går innom minst én annen node v.



Skriv en prosedyre som avgjør om grafen inneholder en sykel.

```
Input: En urettet graf G = (V, E)
Output: Returnerer true hvis grafen inneholder en sykel, og false ellers
Procedure isCyclicUndirected(G)
    // ...
```

Det finnes flere gode løsninger i $\mathcal{O}(|V|+|E|)$. Den mest nærliggende er kanskje et dybde-først søk.

```
Input: En urettet graf G = (V, E)
Output: Returnerer true hvis grafen inneholder en sykel, og false ellers
Procedure isCyclicUndirected(G)
    visited ← empty set
    for u ∈ V do
        if u ∈ visited then
            continue
        if isCyclicComponent(G, null, u, visited) then
            return true
    return false

Procedure isCyclicComponent(G, p, u, visited)
    add u to visited
    for (u, v) ∈ E do
        if v = p then
            continue
        if v ∈ visited then
            return true
        if isCyclicComponent(G, u, v, visited) then
            return true
    return false
```

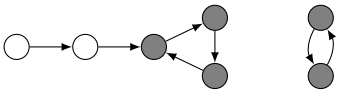
I løsningen over fastslår vi at det finnes en sykel dersom dybde-først søket har en kant til en node som allerede er besøkt, *med mindre* dette er den samme noden som DFS ble kalt fra (altså foreldrenoden i spenntreet). Uten denne begrensningen ville `isCyclicUndirected` svart **`true`** på alle grafer som inneholder en kant.

En annen løsning er å finne komponentene til grafen, og sjekke at hver av komponentene inneholder nøyaktig $|E_c| = |V_c| - 1$, der V_c og E_c henholdsvis er nodene og kantene i komponenten c.

En noe mer innsiktsfull løsning er å se at grafen ikke inneholder sykler hvis og bare hvis $|E| = |V| - |C|$, der C er mengden av komponenter. Altså inneholder grafen ingen sykler hvis antall kanter svarer til antall noder minus antall komponenter.

- Inntil 5 poeng for en korrekt løsning i $\mathcal{O}(|V| + |E|)$ som er godt formidlet med pseudokode.
- Trekk 2 poeng hvis man ikke tar høyde for at noden må være ulik foreldrenoden i spenntreet.
- Inntil 3 poeng dersom løsningen er i $\mathcal{O}(|V| \cdot (|V| + |E|))$.

(b) Anta at grafen $G = (V, E)$ er en *rettet* graf. Grafen inneholder en sykel hvis det finnes en sti fra en node u tilbake til samme node u.



Skriv en prosedyre som avgjør om grafen inneholder en sykel.

```
Input: En rettet graf G = (V, E)
Output: Returnerer true hvis grafen inneholder en sykel, og false ellers
Procedure isCyclicDirected(G)
    // ...
```

Dette bør løses på lineær tid. Mest naturlig er å bruke topologisk sortering (som oppdager sykler). Algoritmen for topologisk sortering plasserer først alle noder med inngrad 0 på en stack. Noder plukkes av stacken så lenge den ikke er tom, og for hver node som plukkes av fjernes alle dens utgående kanter (som fører til at dens naboer får lavere inngrad). Hvis en node får inngrad 0 plasseres den på stacken. Grafen er asyklisk hvis og bare hvis alle nodene blir prosessert.

```
Procedure isCyclicDirected(G)
    stack ← empty stack
    processed ← 0
    for v ∈ V do
        if indegree of v is 0 then
            stack.push(v)
    while stack is not empty do
        u ← stack.pop()
        processed ← processed + 1
        for (u, v) ∈ E do
            remove u from the incoming edges of v
            if indegree of v is 0 then
                add v to stack
    return processed < |V|
```

En annen mulighet er å finne de sterkt sammenhengende komponentene av grafen, og sjekke om det finnes komponenter av størrelse større enn én.

Det er også fullt mulig å skrive et modifisert dybde-først søk som oppdager sykler. I en slik løsning vil man typisk skille mellom en node er *uoppdaget*, *under prosessering* eller *ferdig prosessert*. Dersom man under et søk finner en kant til en node som er *under prosessering* kan man konkludere med at grafen inneholder en sykel. En annen lignende (og nærliggende) løsning er å sjekke om en node er på *stacken* (fordi da er den under prosessering); med en vanlig stack vil dette gi $\mathcal{O}(|V| \cdot (|V| + |E|))$.

- Inntil 6 poeng for en korrekt løsning i $\mathcal{O}(|V| + |E|)$ som er godt formidlet med pseudokode.
- Inntil 3 poeng dersom løsningen er i $\mathcal{O}(|V| \cdot (|V| + |E|))$.