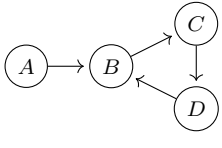
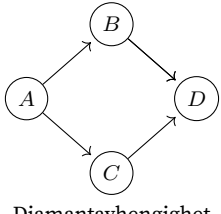
Dependency hell

I moderne programvareutvikling er det vanlig at en applikasjon er avhengig av mange *biblioteker*, som igjen kan avhenge av andre biblioteker. Et bibliotek kan ses på som en samling med kode som er designet for gjenbruk. Bibliotekene som brukes for å bygge en applikasjon kalles *avhengighetene* til applikasjonen. Avhengighetene mellom biblioteker er gitt som en rettet graf $G = (V, E)$, der en node representerer et bibliotek, og hvor det går en kant fra u til v dersom u bruker v som et bibliotek.

Eksemplene under er situasjonener ofte omtalt som *dependency hell*.



Sirkulær avhengighet



Diamantavhengighet

Sykliske avhengigheter er som regel uønsket fordi en oppdatering av et av bibliotekene i en sykel vil kunne påvirke alle de andre bibliotekene i sykelen. *Diamantavhengighet* er også som regel uønsket fordi en oppdatering kan føre til at man blir avhengig av forskjellige versjoner av det samme biblioteket samtidig.

Du skal utvikle algoritmer som kan hjelpe med å identifisere slike uønskede situasjoner.

(a) Mer presist sier vi at grafen inneholder en *syklisk avhengighet* dersom det finnes en sti fra en node v til den samme noden v som består av minst to noder.

Skriv en prosedyre som, gitt en rettet graf $G = (V, E)$, returner **`true`** dersom grafen inneholder *sykliske avhengigheter*, og **`false`** ellers. Dersom prosedyren din benytter seg av algoritmer kjent fra pensum må du redgjøre kort for hvordan de fungerer.

(b) I denne deloppgaven kan du anta at grafen ikke inneholder sykliske avhengigheter.

Mer presist sier vi at grafen inneholder en *diamantavhengighet* dersom det finnes to distinkte stier fra u til v. Husk at to stier er distinkte dersom de ikke deler noen noder eller kanter utenom u og v.

Skriv en prosedyre som, gitt en rettet asyklisk graf $G = (V, E)$, returnerer **`true`** dersom grafen inneholder *diamantavhengigheter*, og **`false`** ellers.