DM503

Forelæsning 8

Indhold

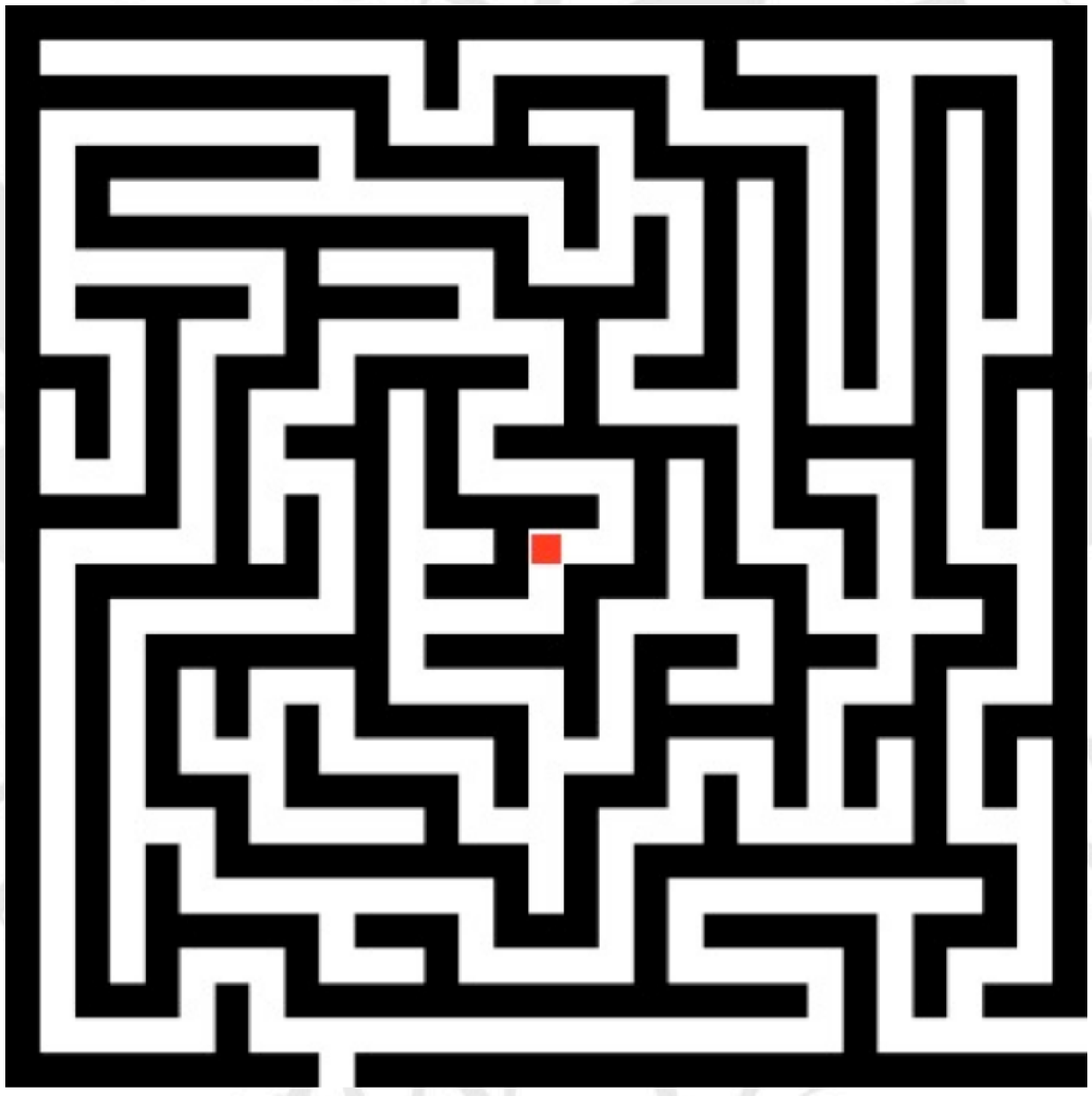
- Grafer
  - Labyrint
  - Brede-først-søgning
  - Labyrint genbesøg
- 2. Eksamensdelprojektopgave

Eksaminatorietimer / Labøvelser

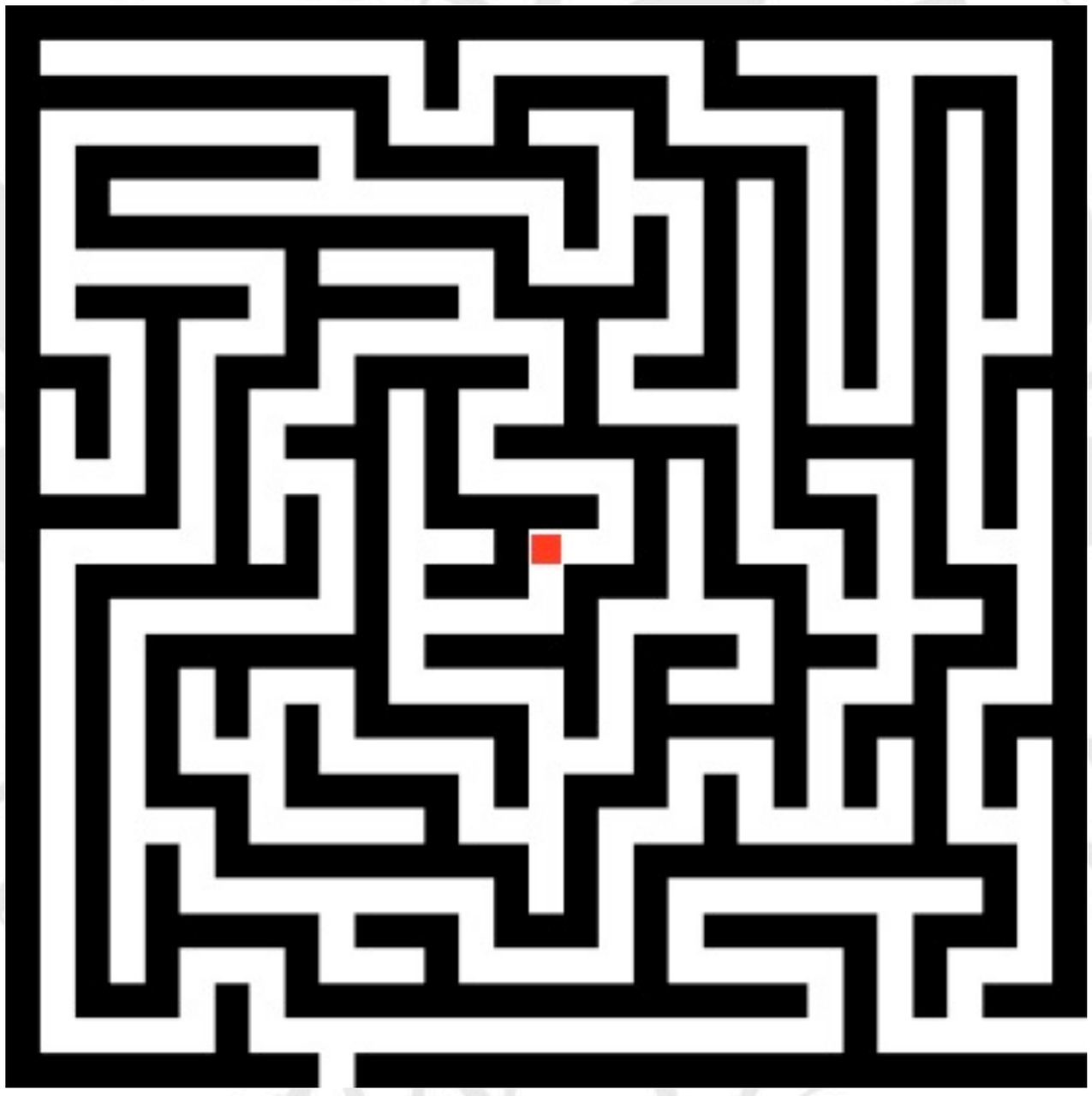
- Onsdag eksaminatorier for S1/M1 og S7 i U17/U26
- Labøvelser for S7 på torsdag
- Labøvelser for S1/M1 på fredag

Labyrint

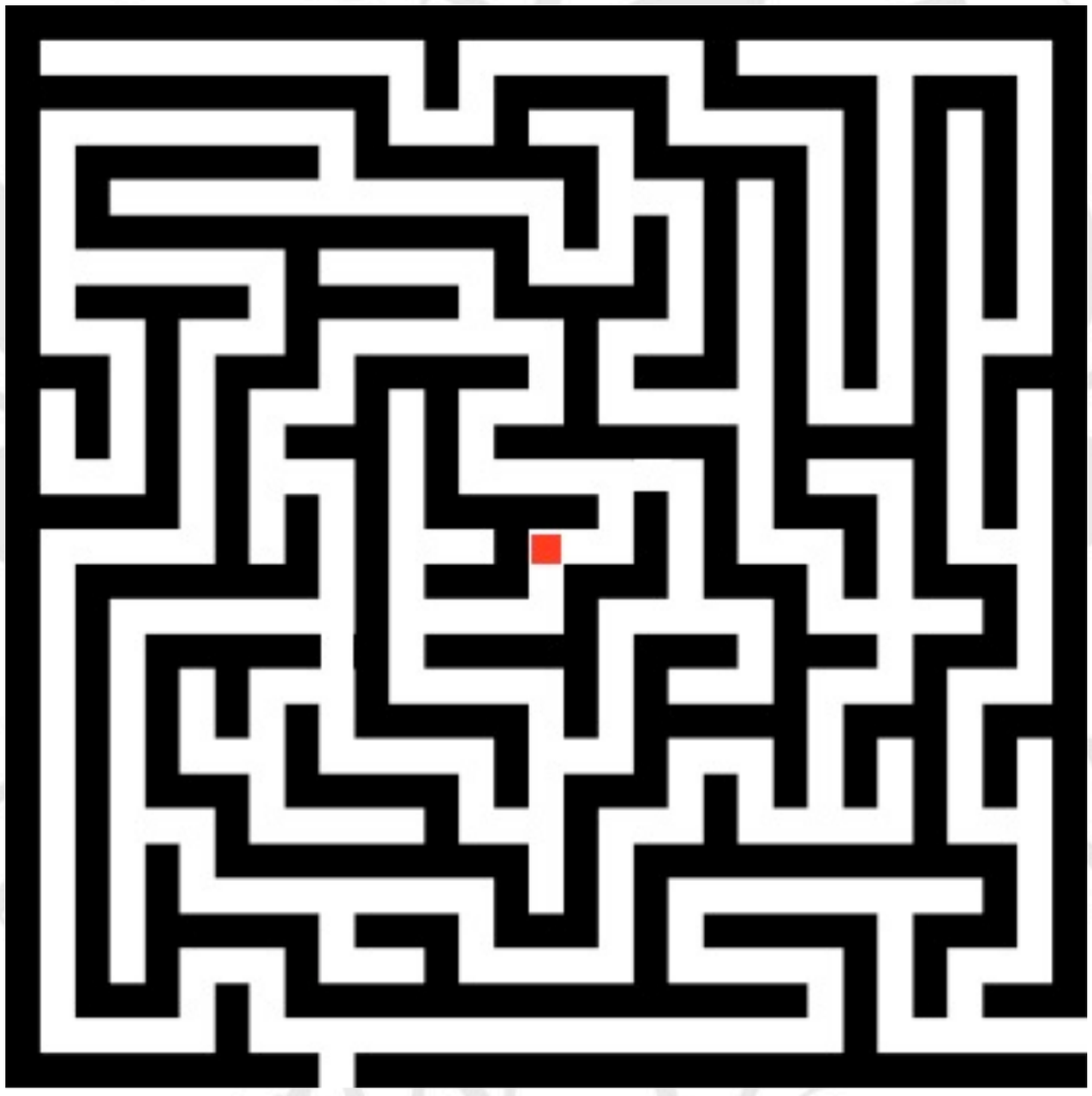
- helten bliver droppet i midten og skal finde en udgang
- helten kan kun se nabofeltene



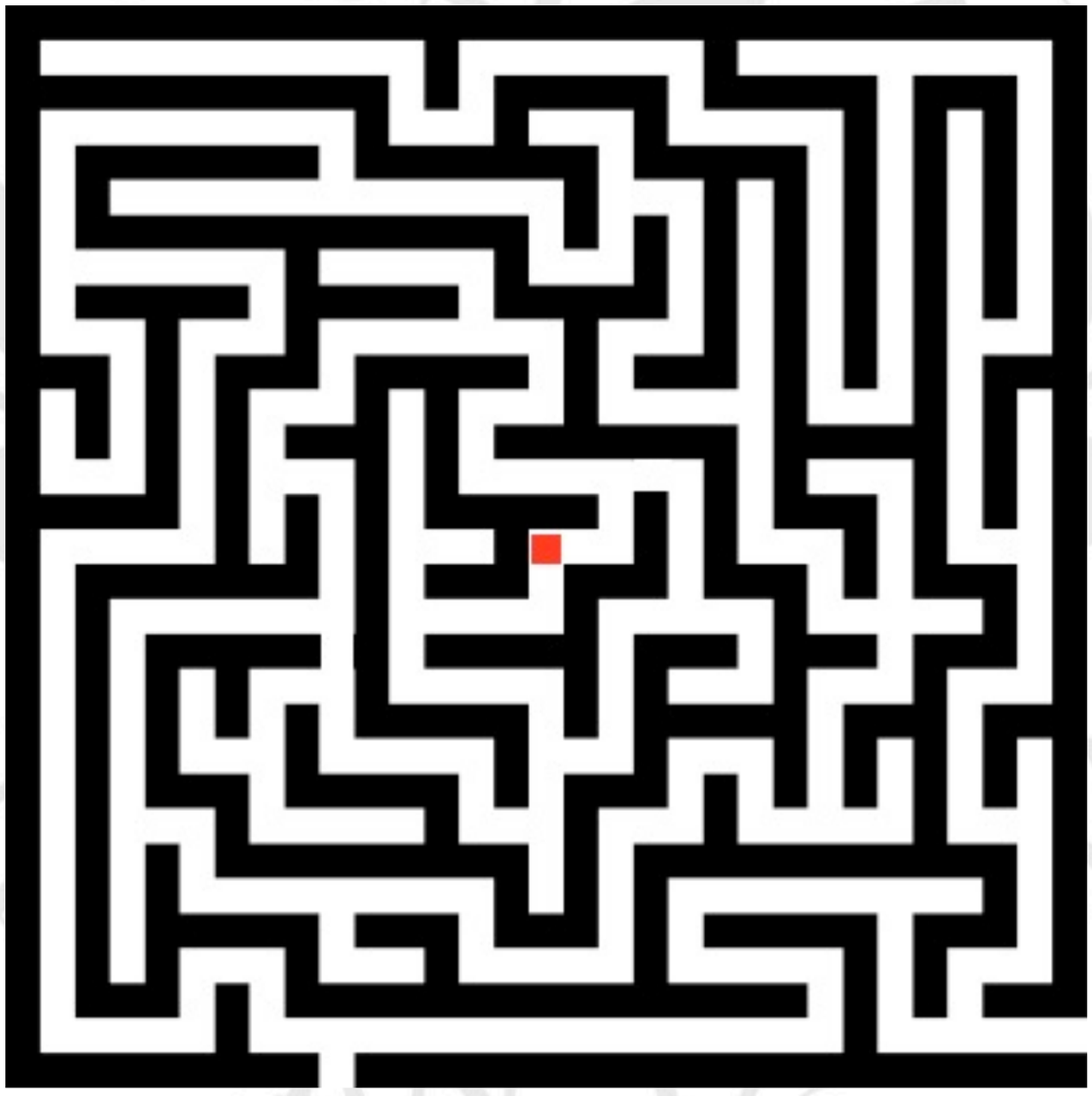
Hvilken strategi ville du bruge for at finde udgangen?



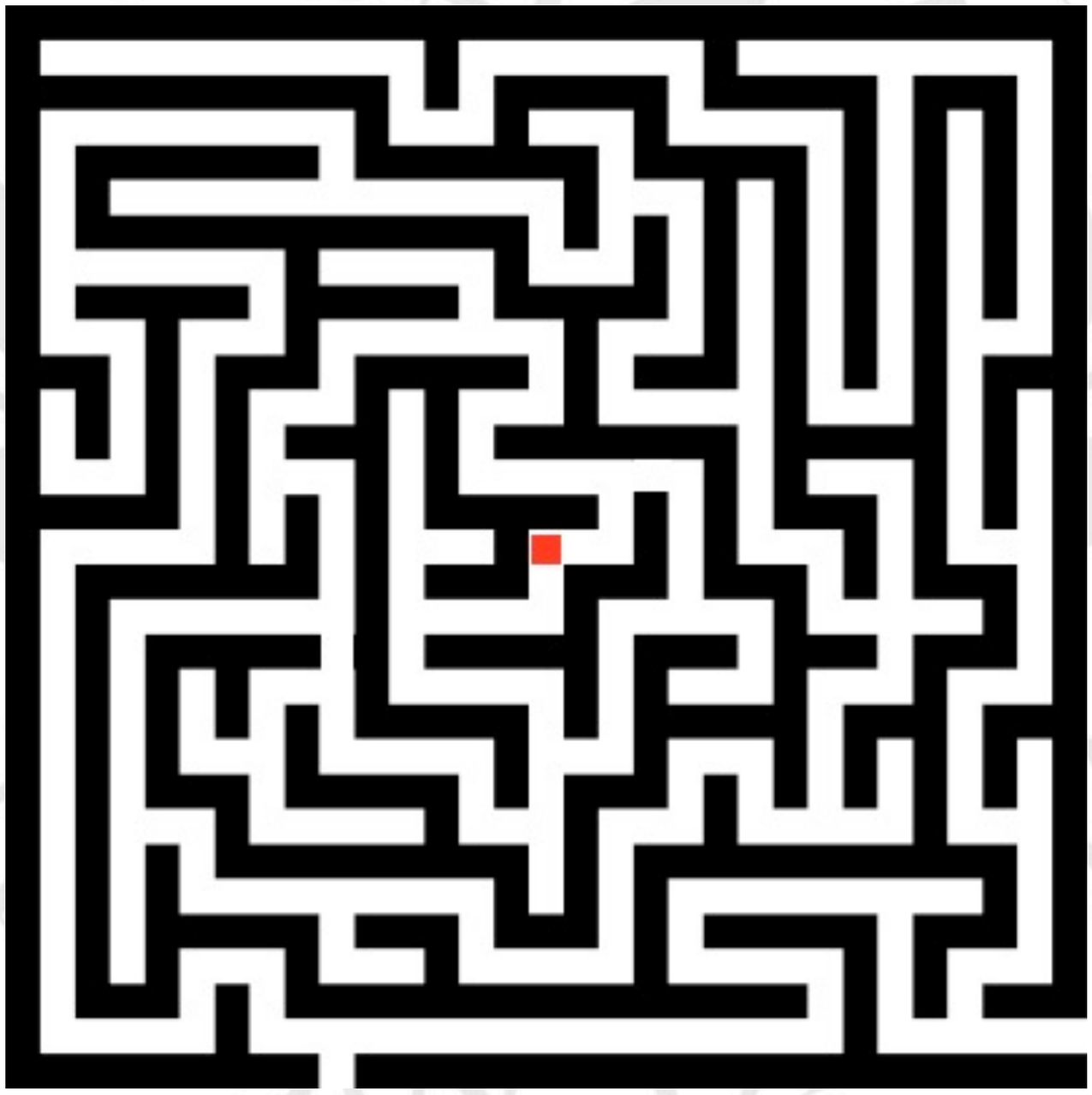
Hvirker din strategi også for denne labyrint?



Hvordan kan vi modellere labyrinten som en graf?



Hvordan kan man anvende dybde-først-søgning?



Brede-først-søgning

- BFS
  - Algoritmen
  - Eksempel
  - Egenskaber
  - Java-kode

Brede-først-søgning

Brede-først-søgning

- Brede-først-søgning (BFS) er en generel teknik til at gennemsøge en graf

Brede-først-søgning

- Brede-først-søgning (BFS) er en generel teknik til at gennemsøge en graf
  - Besøger alle knuder og kanter

Brede-først-søgning

- Brede-først-søgning (BFS) er en generel teknik til at gennemsøge en graf
  - Besøger alle knuder og kanter
  - Afgør om grafen er sammenhægende

Brede-først-søgning

- Brede-først-søgning (BFS) er en generel teknik til at gennemsøge en graf
  - Besøger alle knuder og kanter
  - Afgør om grafen er sammenhægende
  - Beregner sammenhængskomponenterne

Brede-først-søgning

- Brede-først-søgning (BFS) er en generel teknik til at gennemsøge en graf
  - Besøger alle knuder og kanter
  - Afgør om grafen er sammenhægende
  - Beregner sammenhængskomponenterne
  - Beregner en udspændende skov for grafen

Brede-først-søgning

- Brede-først-søgning (BFS) er en generel teknik til at gennemsøge en graf
  - Besøger alle knuder og kanter
  - Afgør om grafen er sammenhægende
  - Beregner sammenhængskomponenterne
  - Beregner en udspændende skov for grafen
- BFS tager tid der er proportionalt med n+m

Brede-først-søgning

- Brede-først-søgning (BFS) er en generel teknik til at gennemsøge en graf
  - Besøger alle knuder og kanter
  - Afgør om grafen er sammenhægende
  - Beregner sammenhængskomponenterne
  - Beregner en udspændende skov for grafen
- BFS tager tid der er proportionalt med n+m
- BFS kan udvides til at

Brede-først-søgning

- Brede-først-søgning (BFS) er en generel teknik til at gennemsøge en graf
  - Besøger alle knuder og kanter
  - Afgør om grafen er sammenhægende
  - Beregner sammenhængskomponenterne
  - Beregner en udspændende skov for grafen
- BFS tager tid der er proportionalt med n+m
- BFS kan udvides til at
  - Beregne den korteste sti mellem to givne knuder (hvis der findes en sti)

Brede-først-søgning

- Brede-først-søgning (BFS) er en generel teknik til at gennemsøge en graf
  - Besøger alle knuder og kanter
  - Afgør om grafen er sammenhægende
  - Beregner sammenhængskomponenterne
  - Beregner en udspændende skov for grafen
- BFS tager tid der er proportionalt med n+m
- BFS kan udvides til at
  - Beregne den korteste sti mellem to givne knuder (hvis der findes en sti)
  - Finde en kreds i grafen (hvis der er en)

BFS-algoritmen

BFS-algoritmen

- Algoritmen sætter "mærker" på knuderne og kanterne

BFS-algoritmen

- Algoritmen sætter "mærker" på knuderne og kanterne
- Sæt mærket "ubesøgt" på alle kanter og knuder

BFS-algoritmen

- Algoritmen sætter "mærker" på knuderne og kanterne
- Sæt mærket "ubesøgt" på alle kanter og knuder
- For alle knuder v i G

BFS-algoritmen

- Algoritmen sætter "mærker" på knuderne og kanterne
- Sæt mærket "ubesøgt" på alle kanter og knuder
- For alle knuder v i G
  - Hvis v er ubesøgt

BFS-algoritmen

- Algoritmen sætter "mærker" på knuderne og kanterne
- Sæt mærket "ubesøgt" på alle kanter og knuder
- For alle knuder v i G
  - Hvis v er ubesøgt
    - BFS(G, v)

BFS-algoritmen

BFS-algoritmen

- BFS(G, v):

BFS-algoritmen

- BFS(G, v):
  - q = ny tom kø (FIFO) af knuder

BFS-algoritmen

- BFS(G, v):
  - q = ny tom kø (FIFO) af knuder
  - Marker v som "set"

BFS-algoritmen

- BFS(G, v):
  - q = ny tom kø (FIFO) af knuder
  - Marker v som "set"
  - Indsæt v i q

BFS-algoritmen

- BFS(G, v):
  - q = ny tom kø (FIFO) af knuder
  - Marker v som "set"
  - Indsæt v i q
  - Sålænge q ikke er tom

BFS-algoritmen

- BFS(G, v):
  - q = ny tom kø (FIFO) af knuder
  - Marker v som "set"
  - Indsæt v i q
  - Sålænge q ikke er tom
    - u = den næste knude i q

BFS-algoritmen

- BFS(G, v):
  - q = ny tom kø (FIFO) af knuder
  - Marker v som "set"
  - Indsæt v i q
  - Sålænge q ikke er tom
    - u = den næste knude i q
    - Besøg u

BFS-algoritmen

- BFS(G, v):
  - q = ny tom kø (FIFO) af knuder
  - Marker v som "set"
  - Indsæt v i q
  - Sålænge q ikke er tom
    - u = den næste knude i q
    - Besøg u
  - For alle ikke-sete naboer w til u

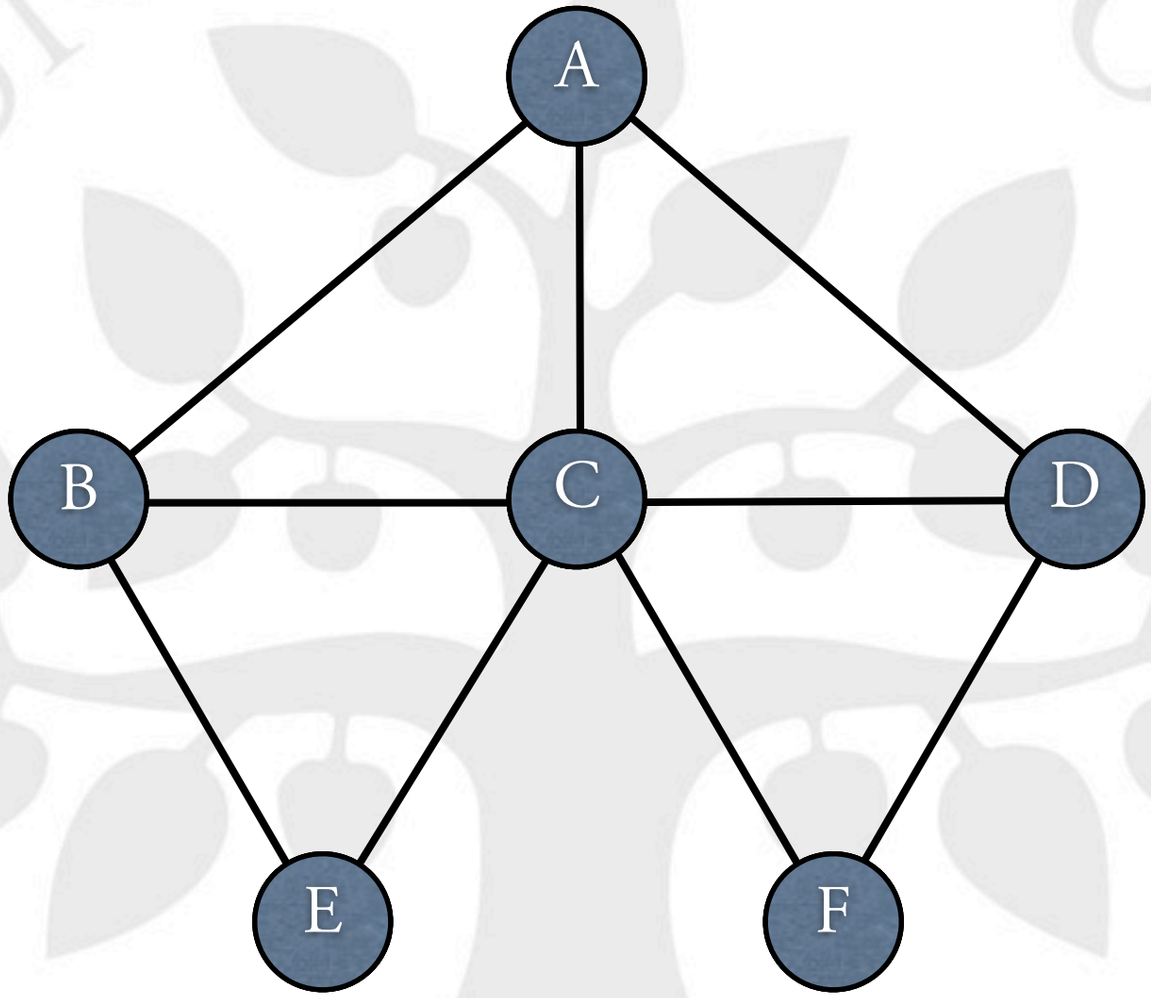
BFS-algoritmen

- BFS(G, v):
  - q = ny tom kø (FIFO) af knuder
  - Marker v som "set"
  - Indsæt v i q
  - Sålænge q ikke er tom
    - u = den næste knude i q
    - Besøg u
  - For alle ikke-sete naboer w til u
    - Marker w som "set"

BFS-algoritmen

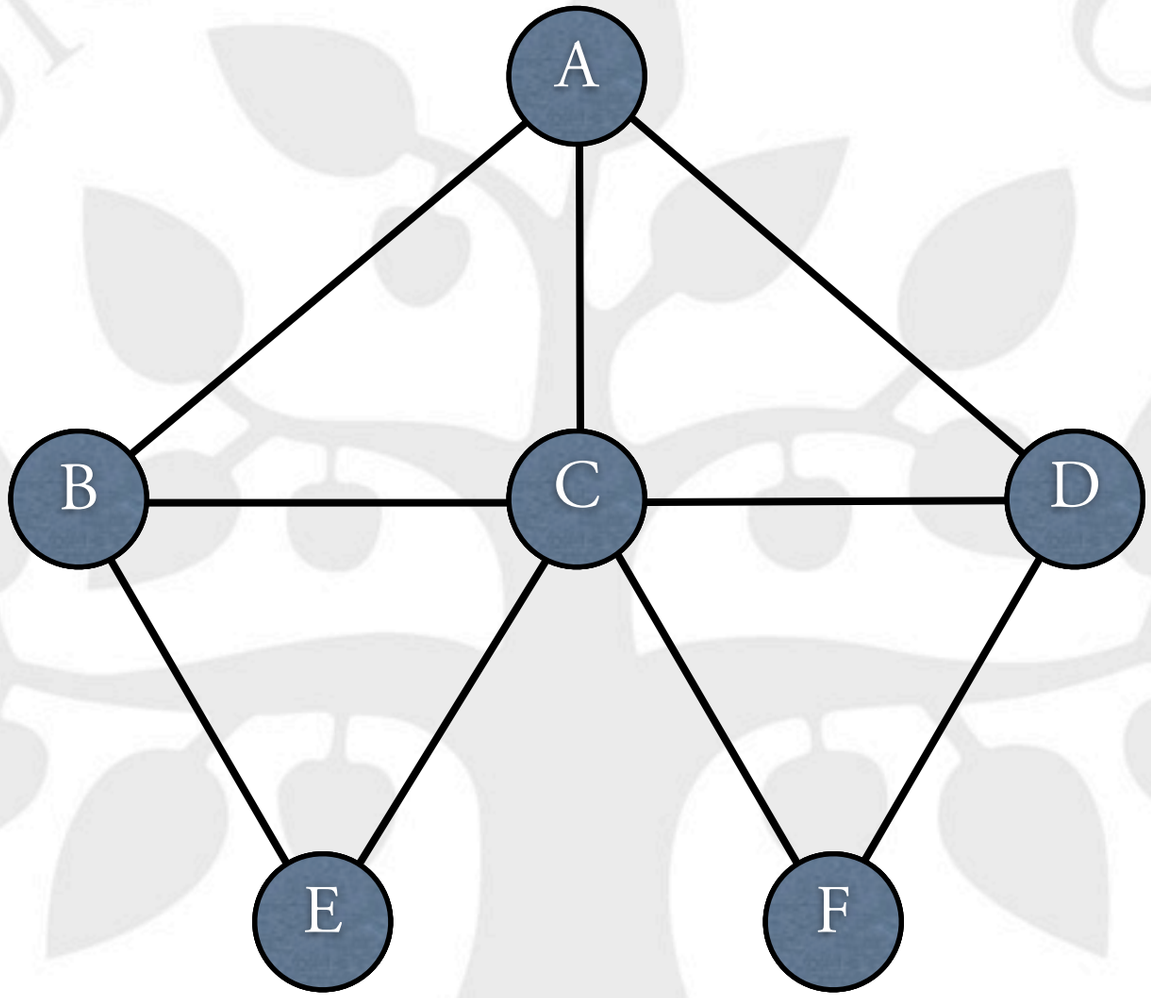
- BFS(G, v):
  - q = ny tom kø (FIFO) af knuder
  - Marker v som "set"
  - Indsæt v i q
  - Sålænge q ikke er tom
    - u = den næste knude i q
    - Besøg u
    - For alle ikke-sete naboer w til u
      - Marker w som "set"
      - Tilføj w til q

Eksempel

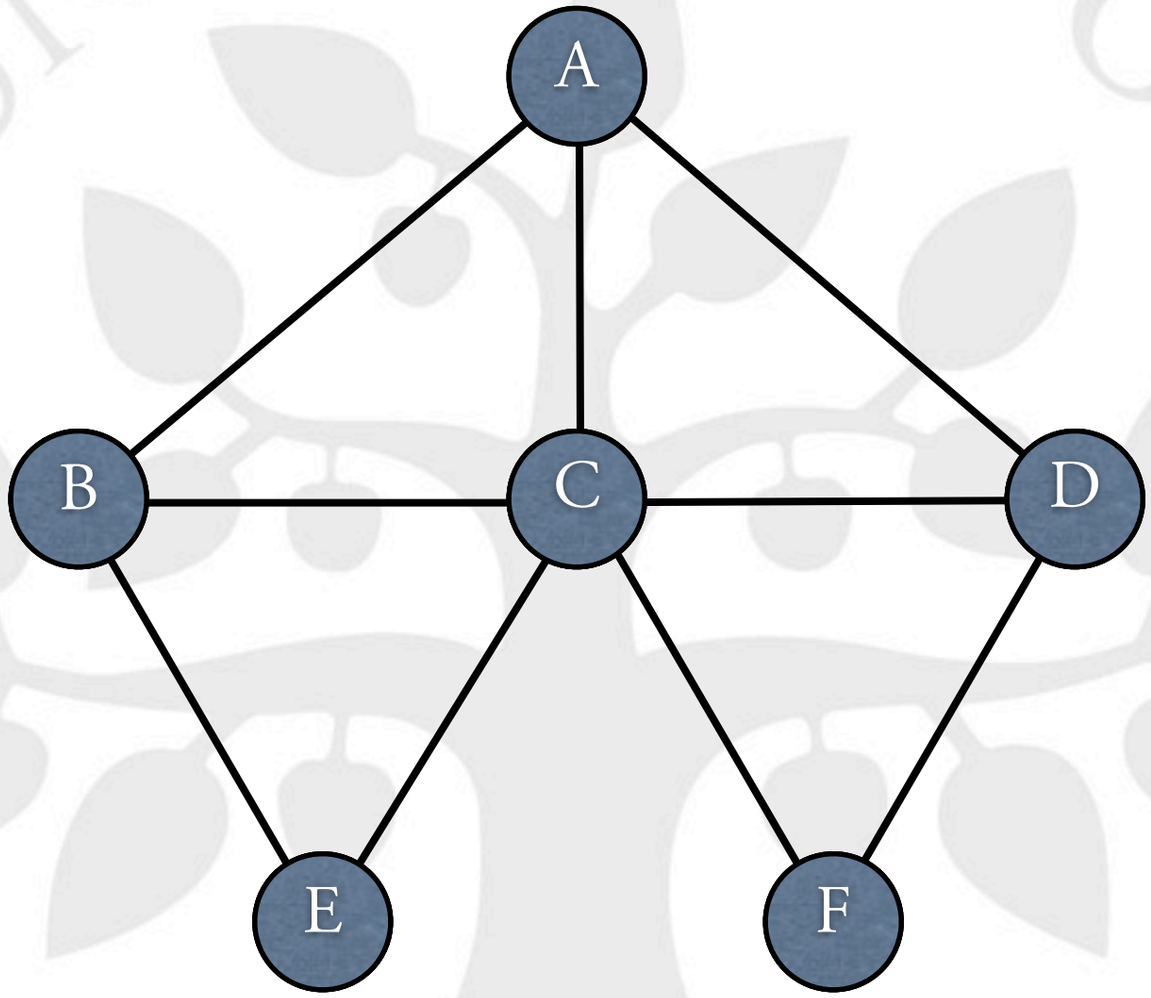


A

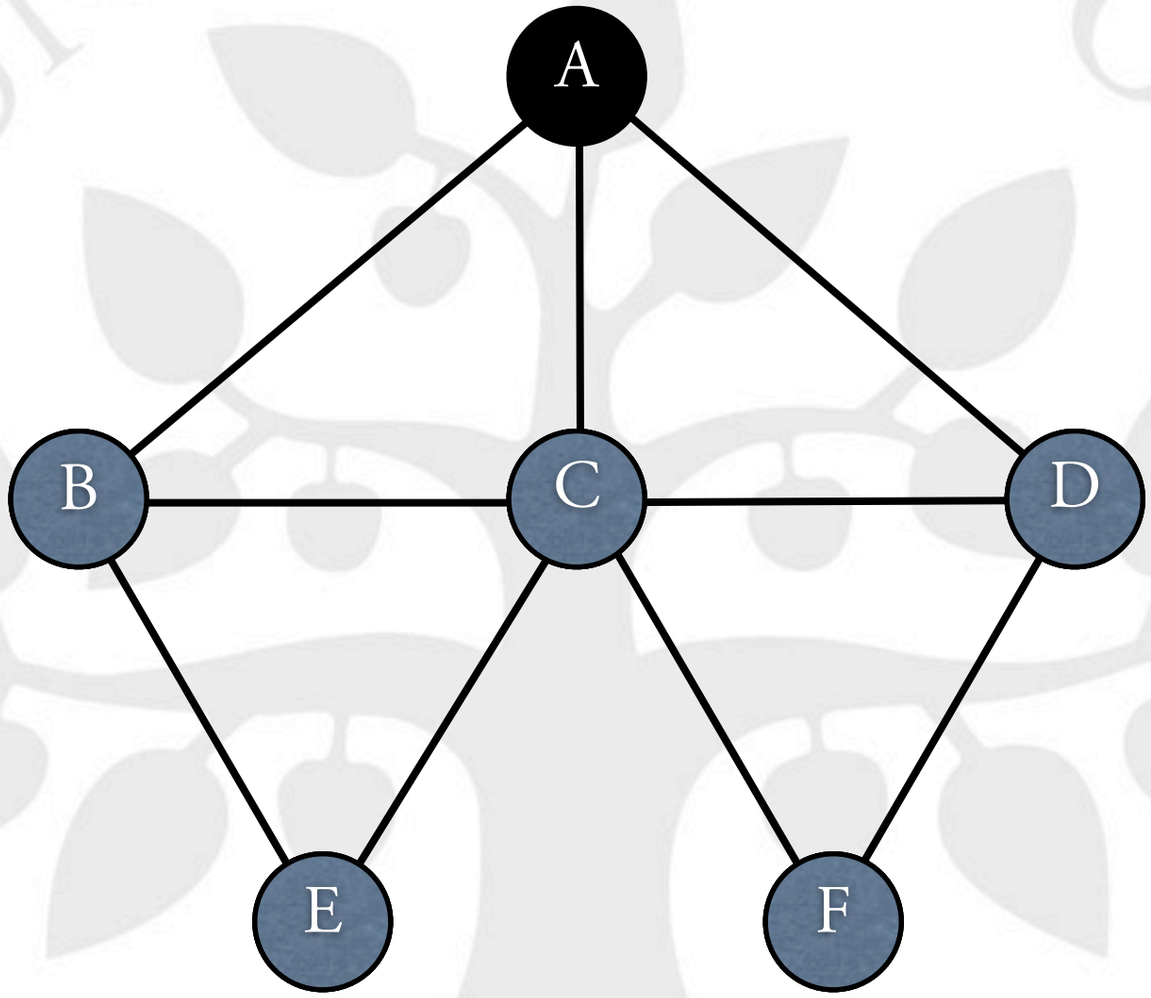
Eksempel



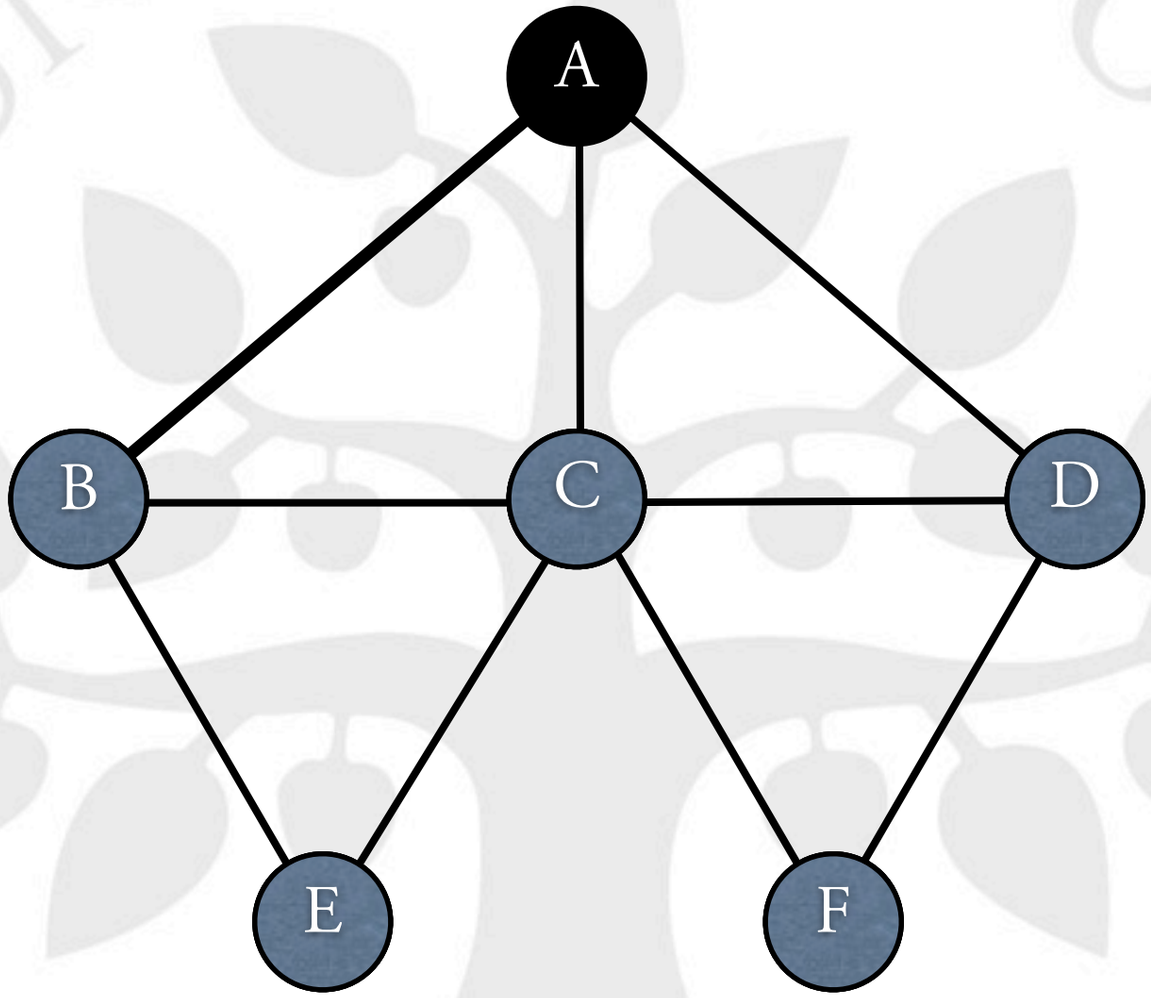
Eksempel



Eksempel

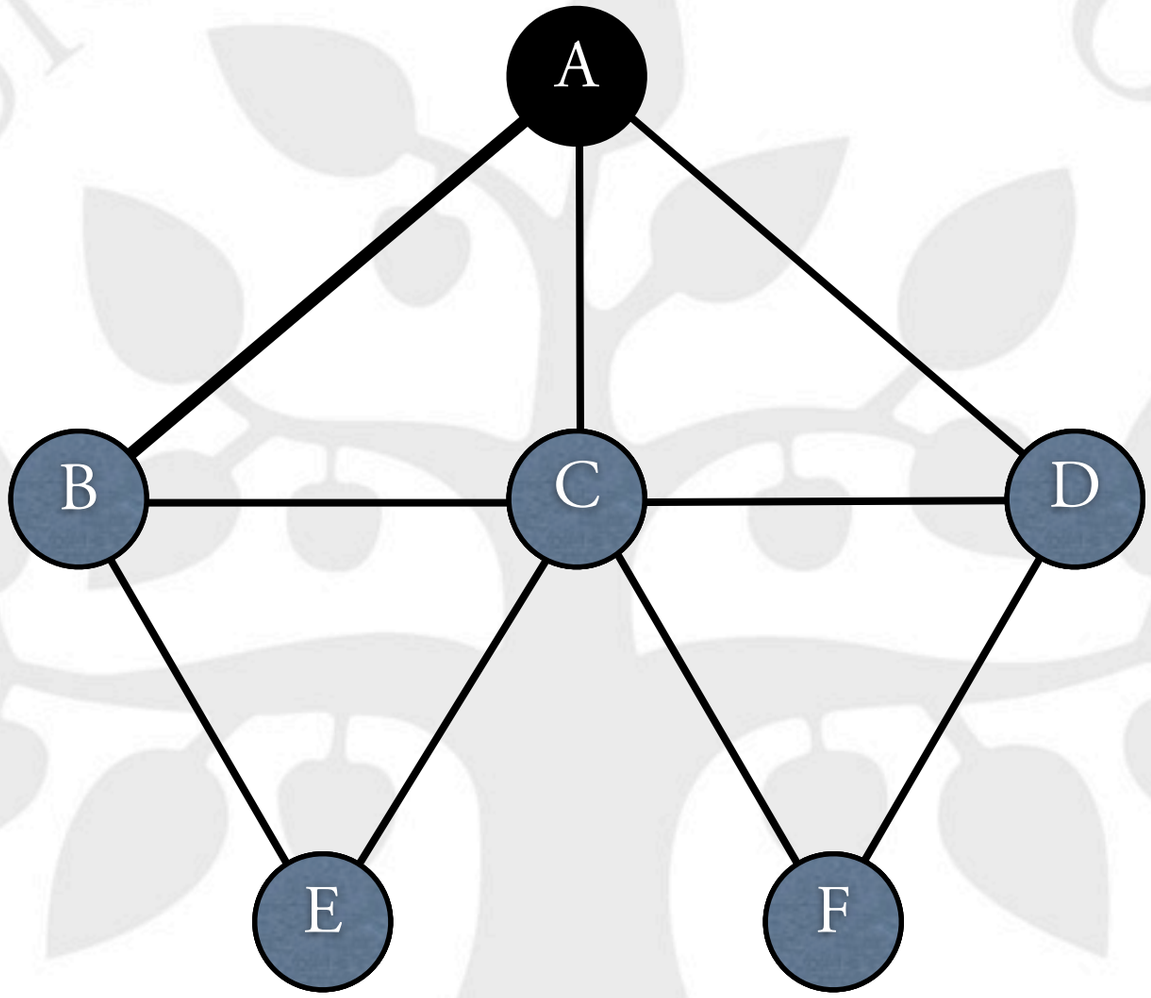


Eksempel

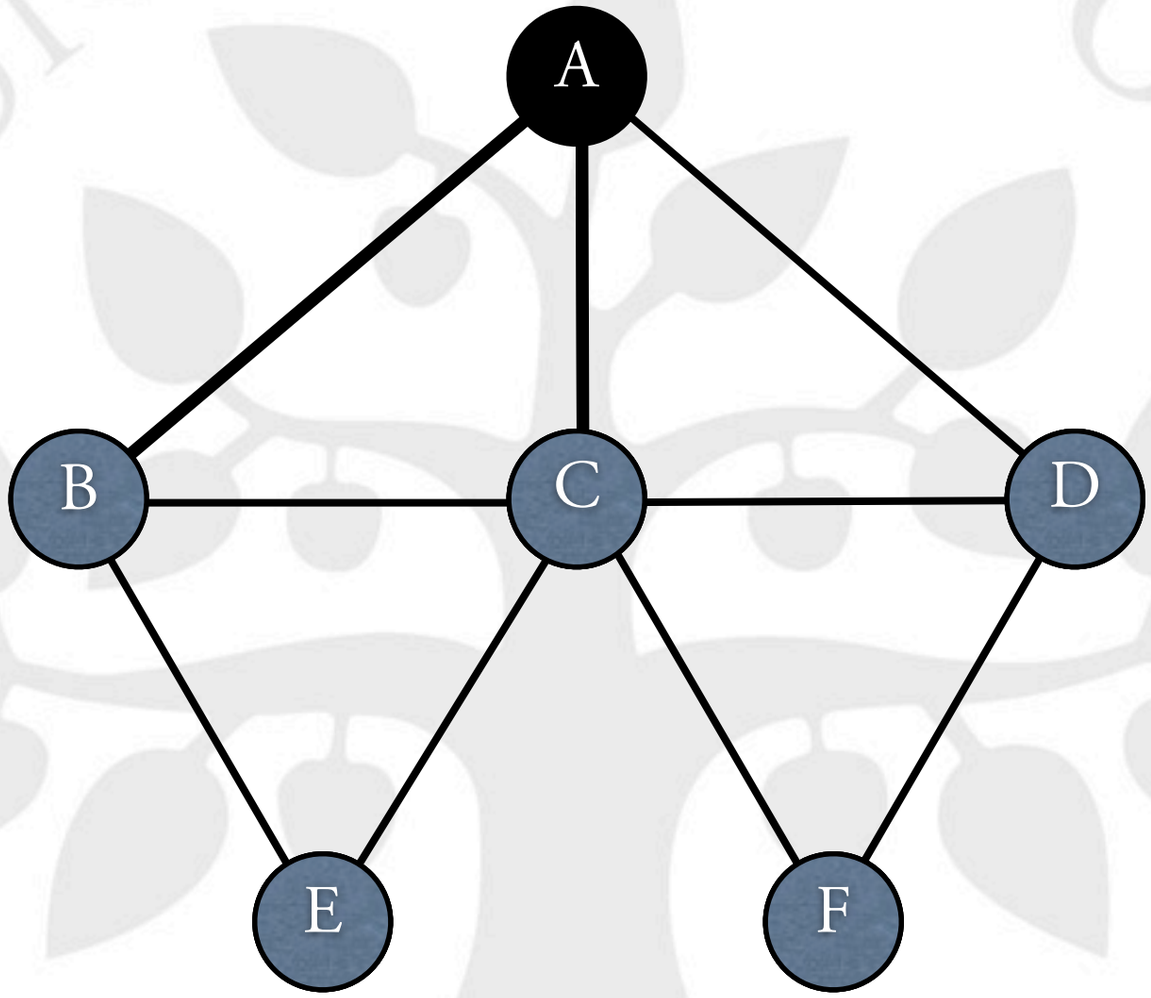


B

Eksempel

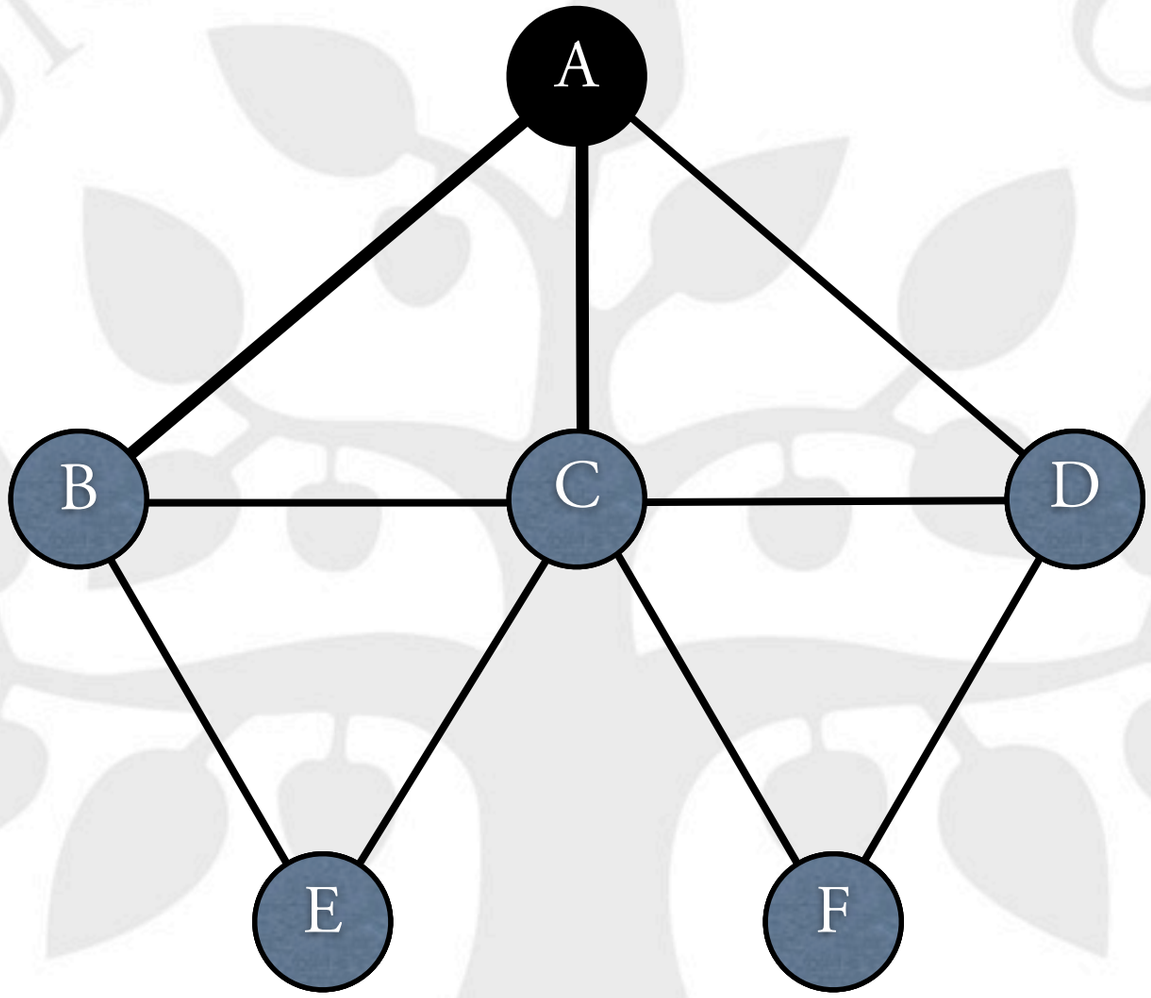


Eksempel



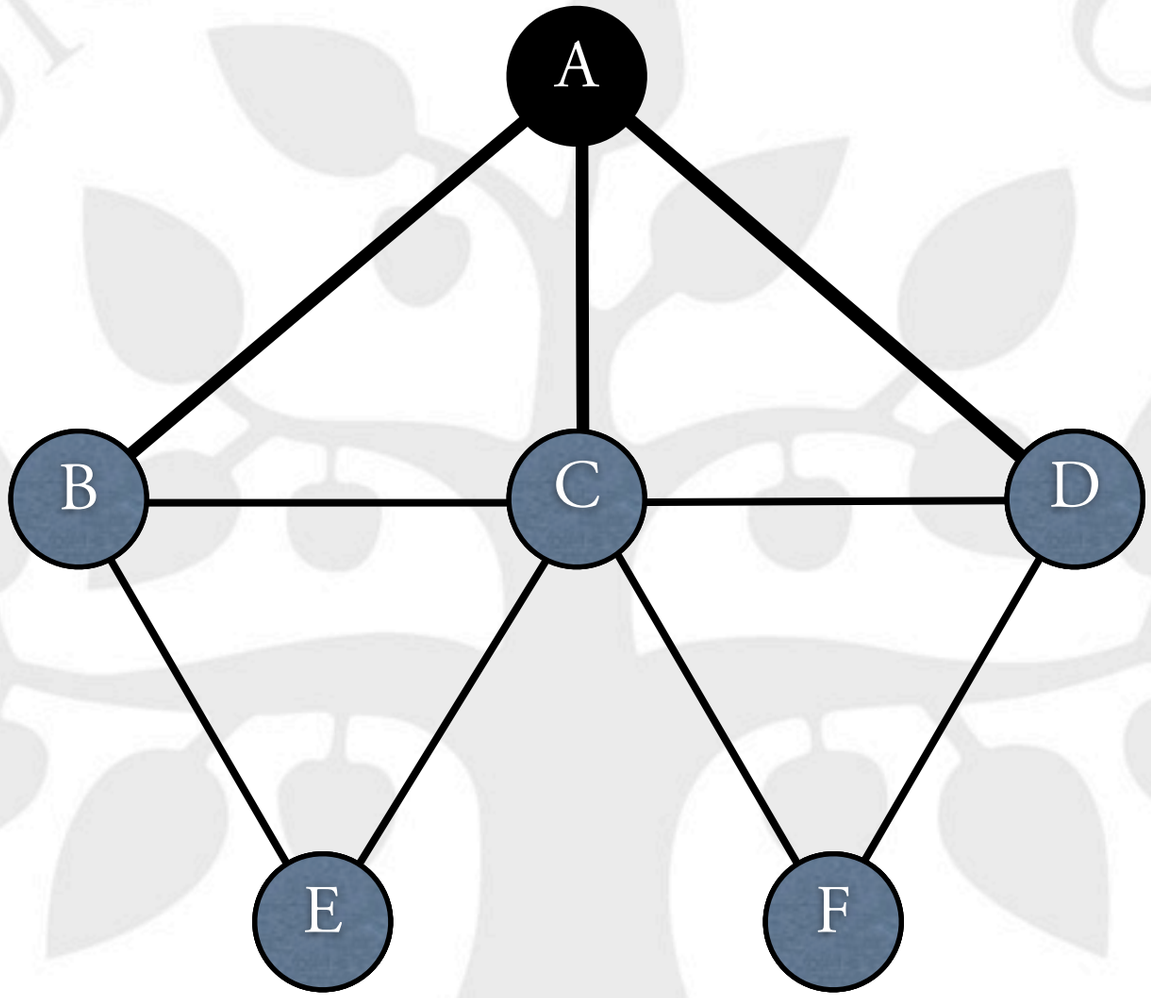
B C

Eksempel



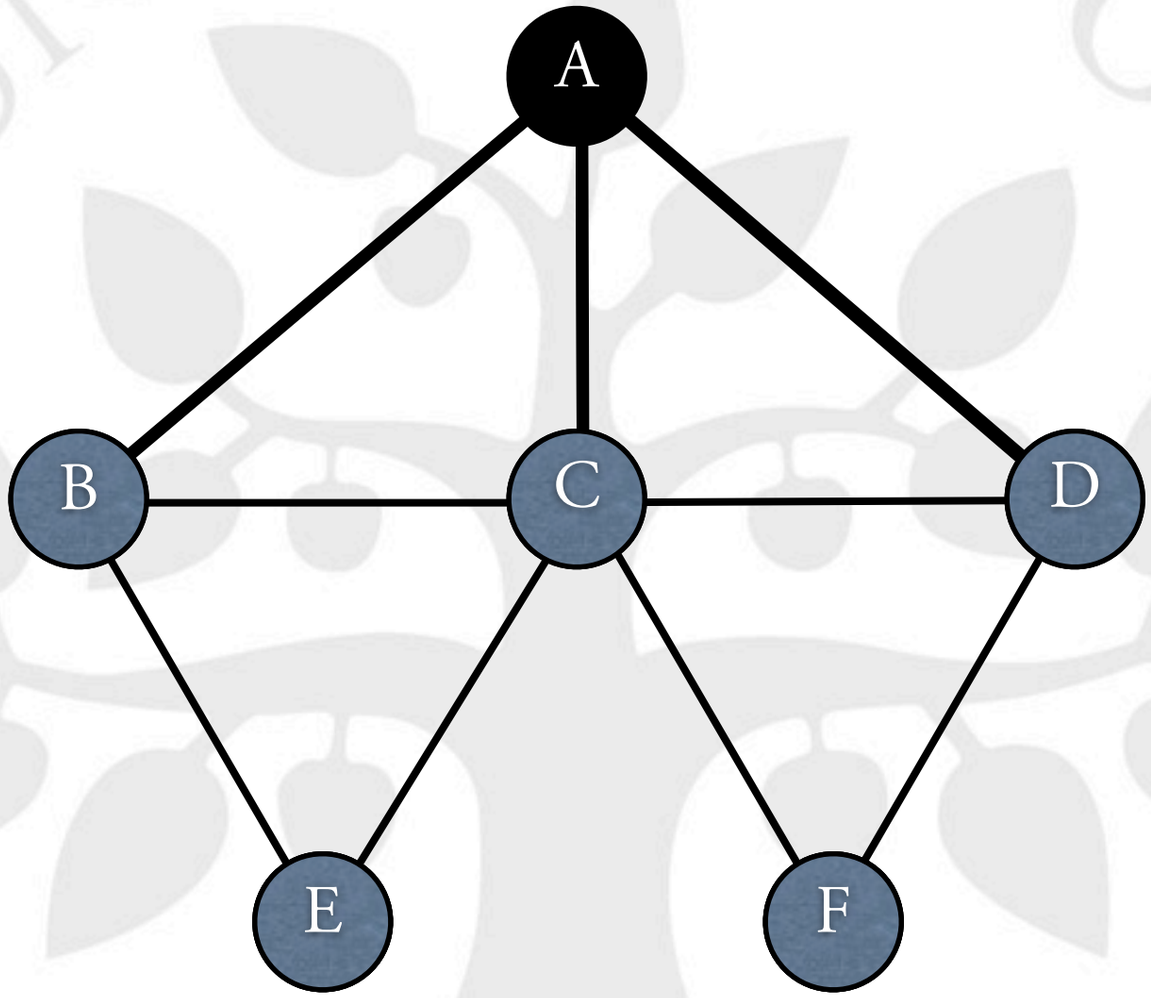
B C

Eksempel



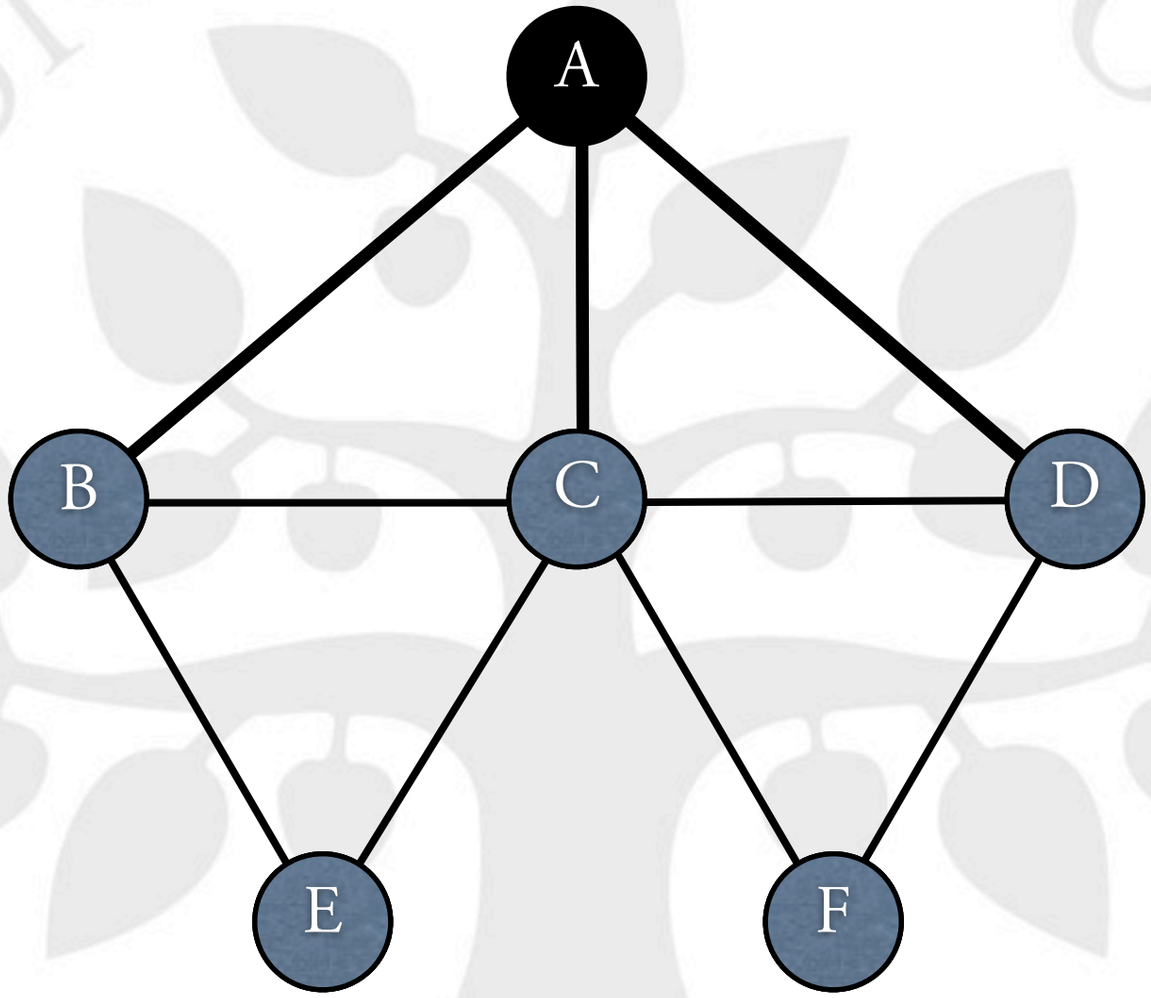
B C D

Eksempel



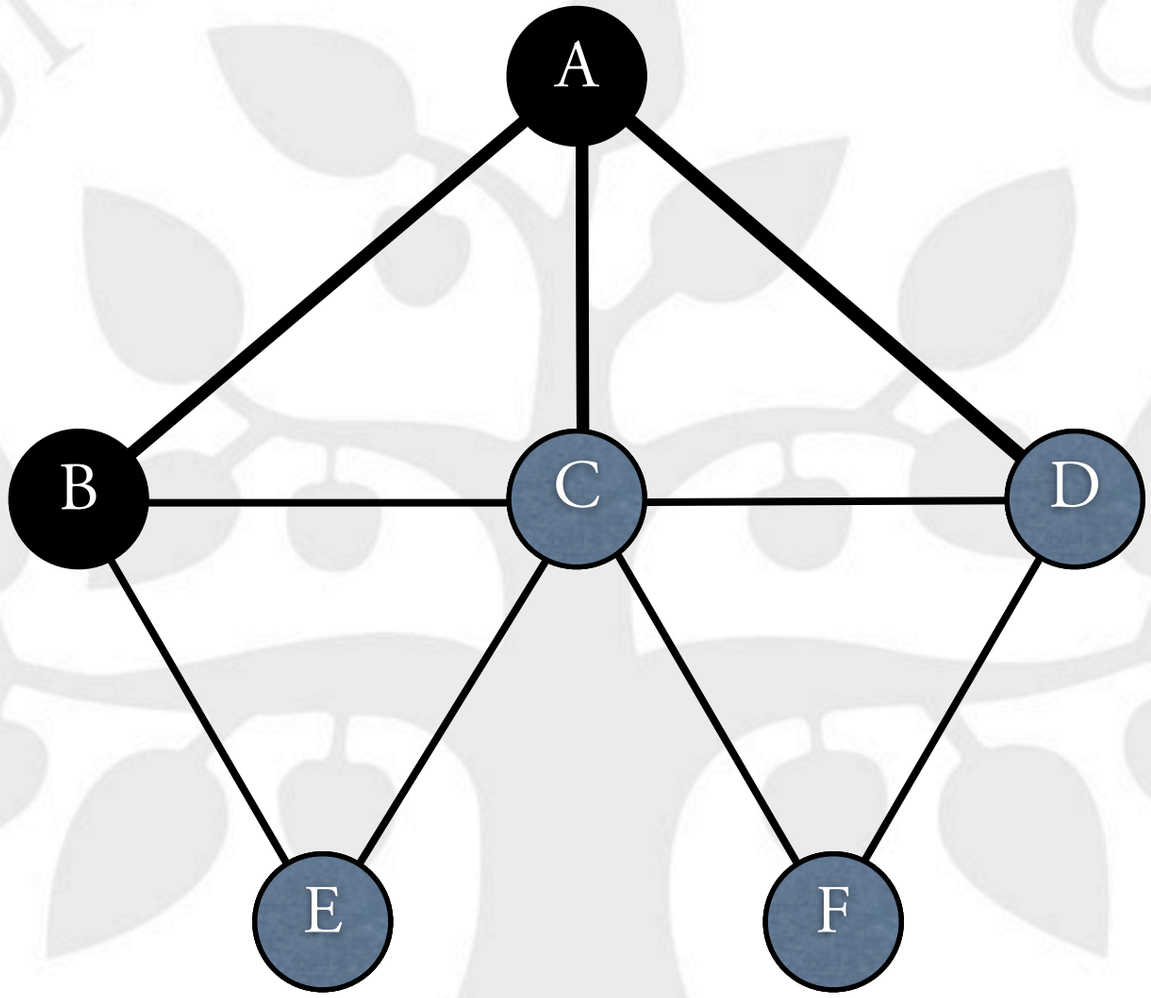
C D

Eksempel



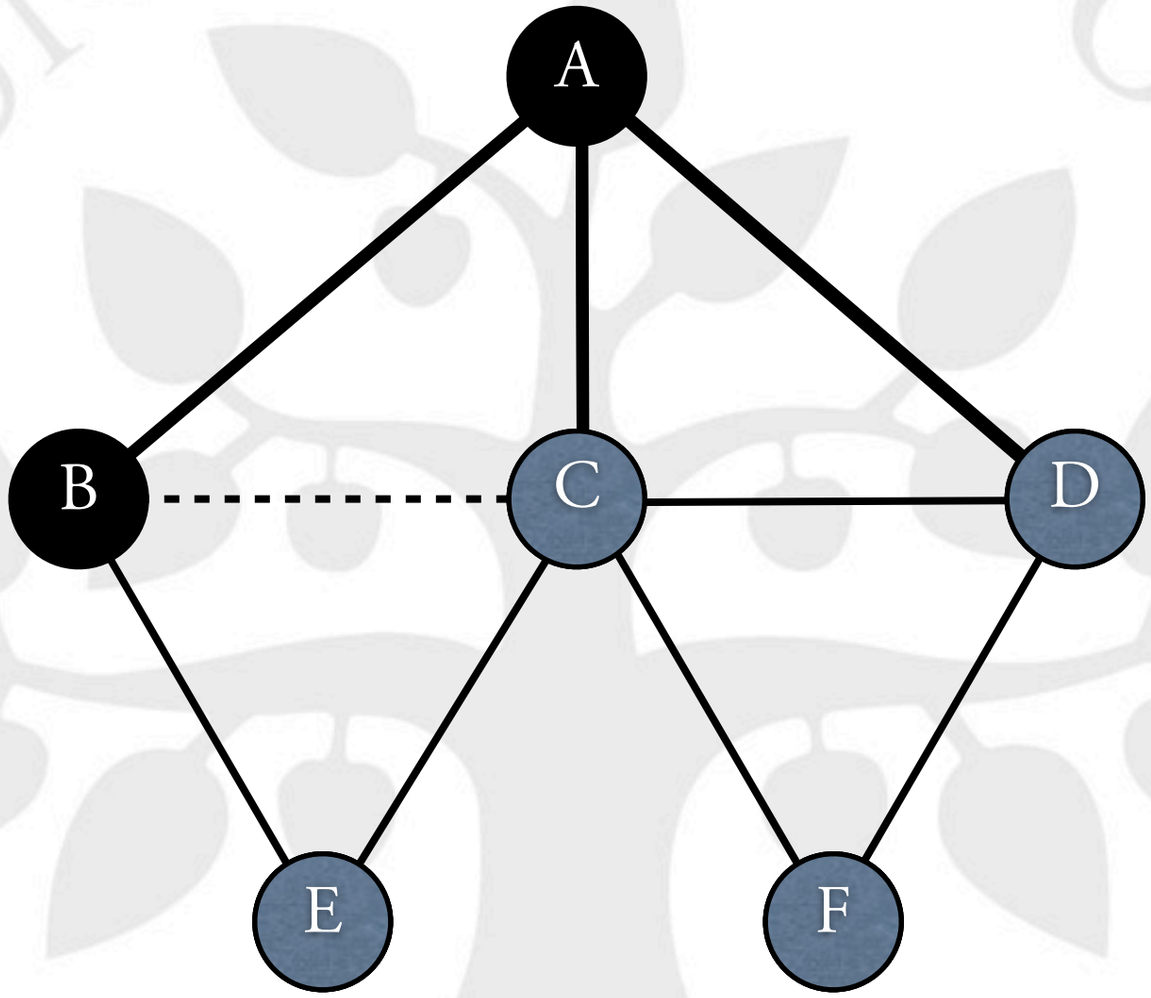
C D

Eksempel



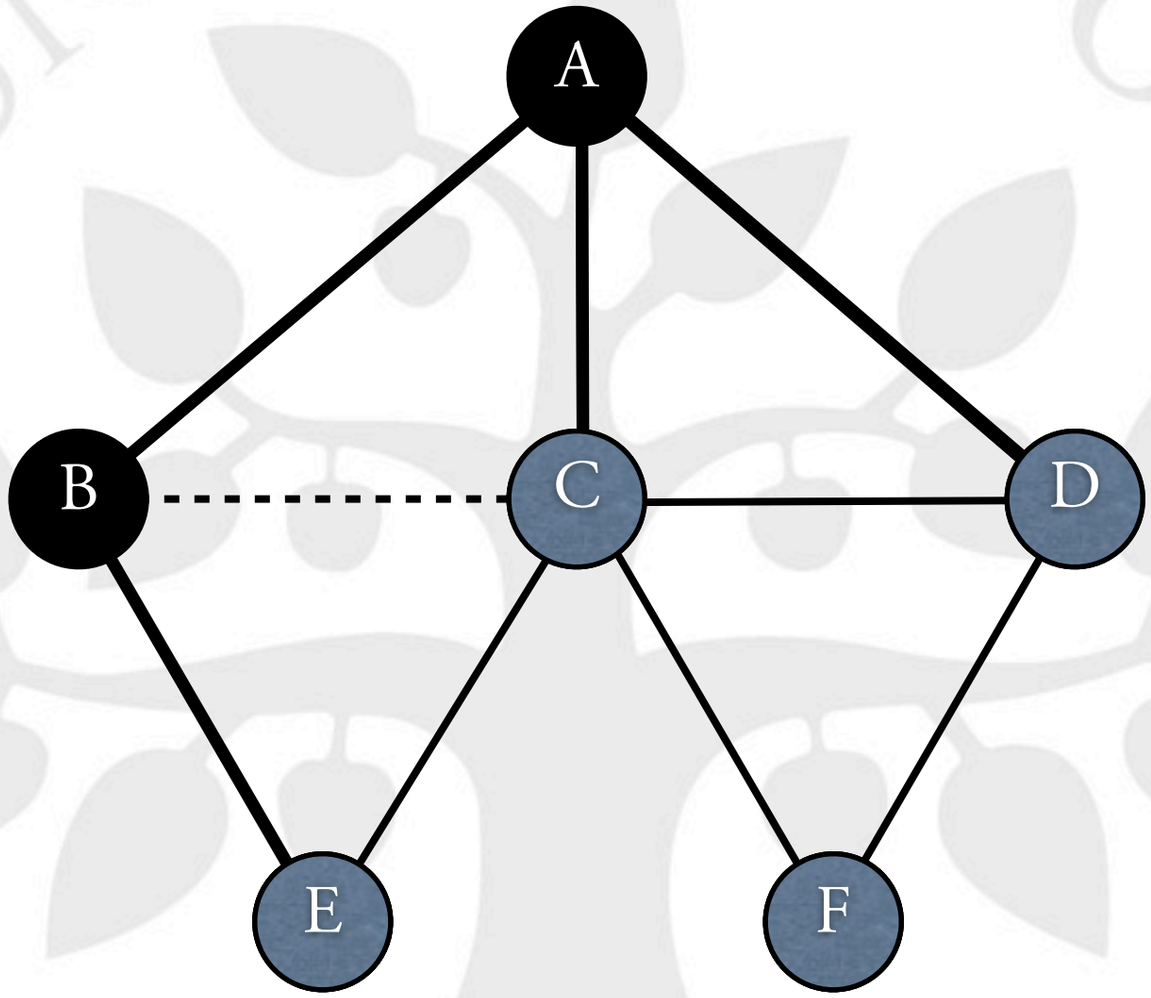
C D

Eksempel



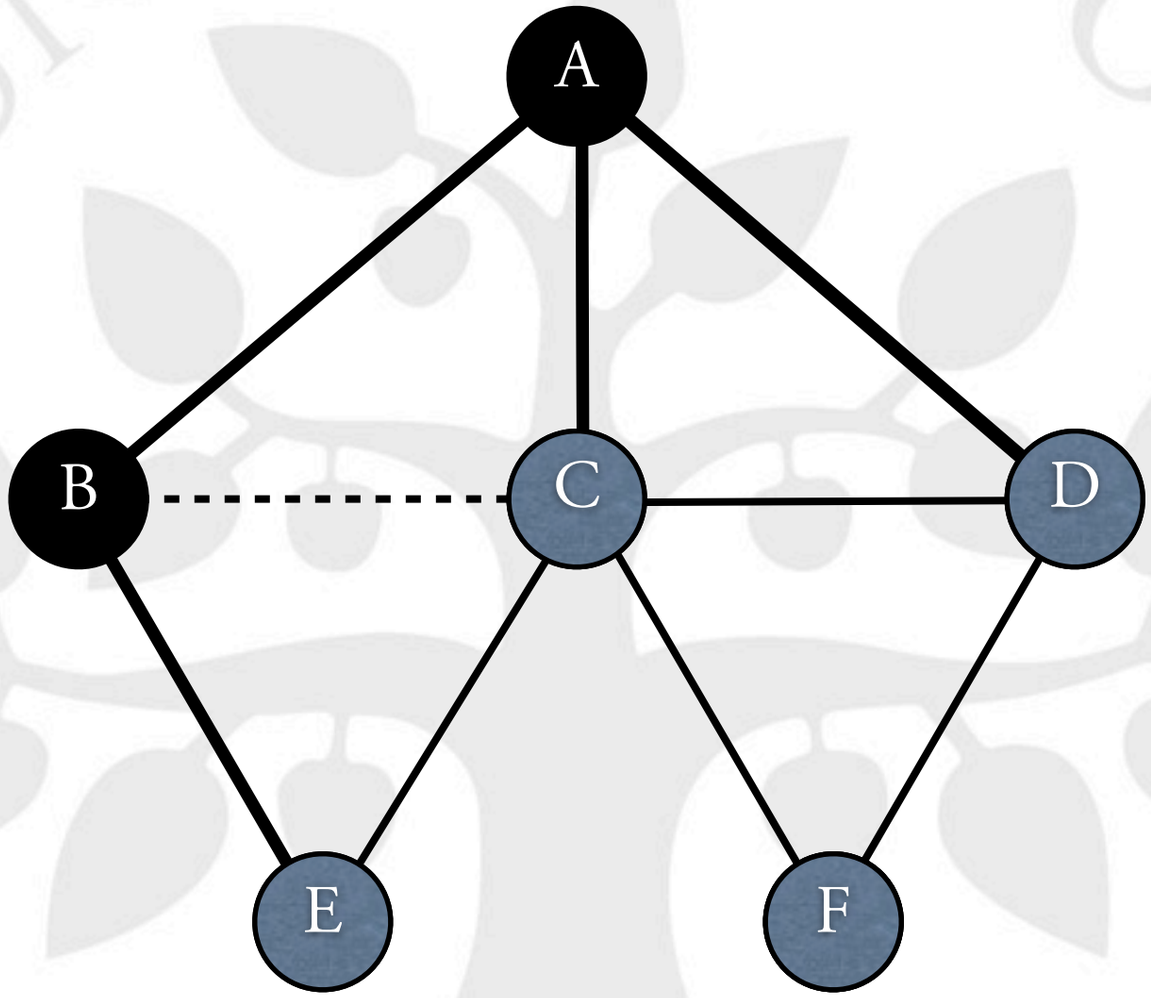
C D

Eksempel



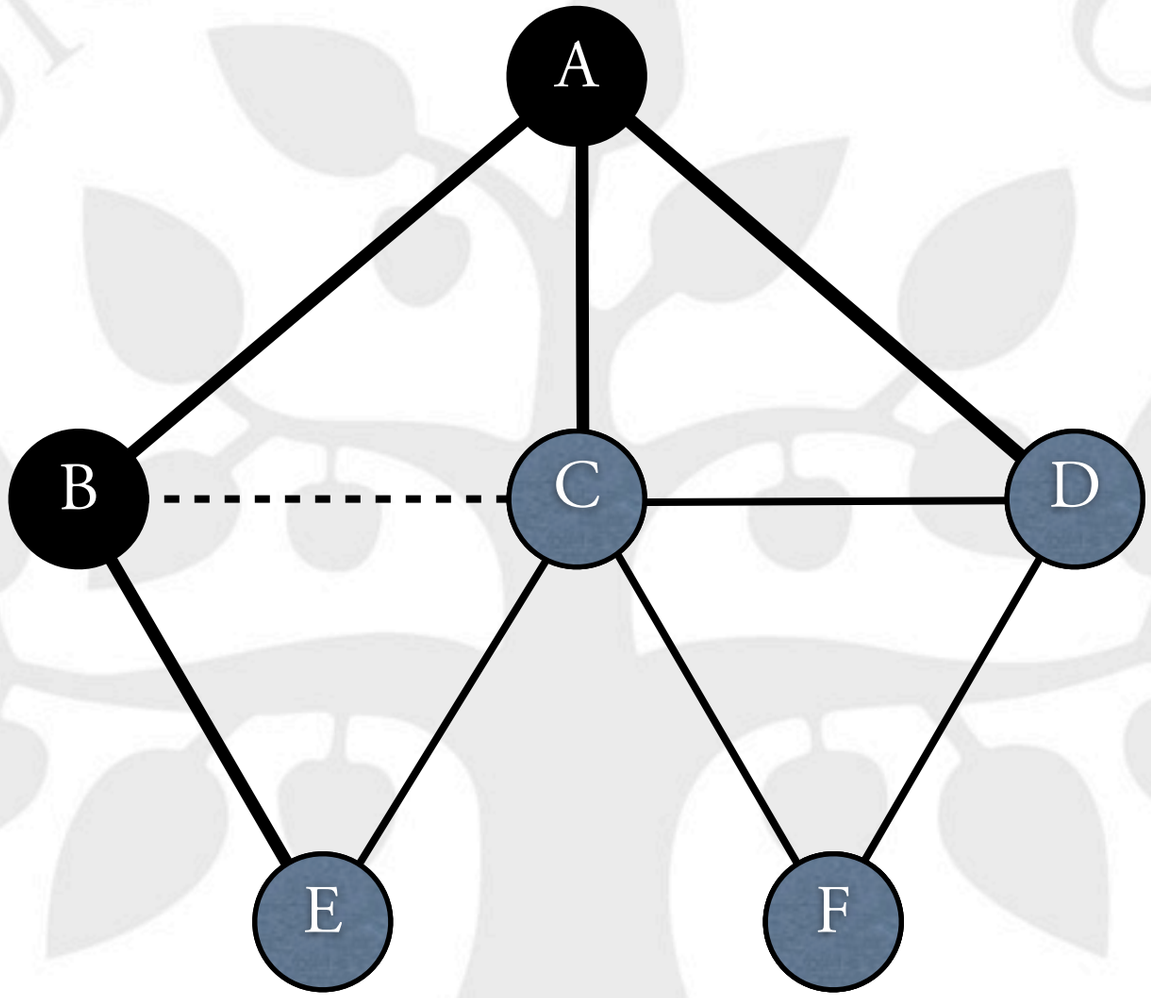
C D E

Eksempel

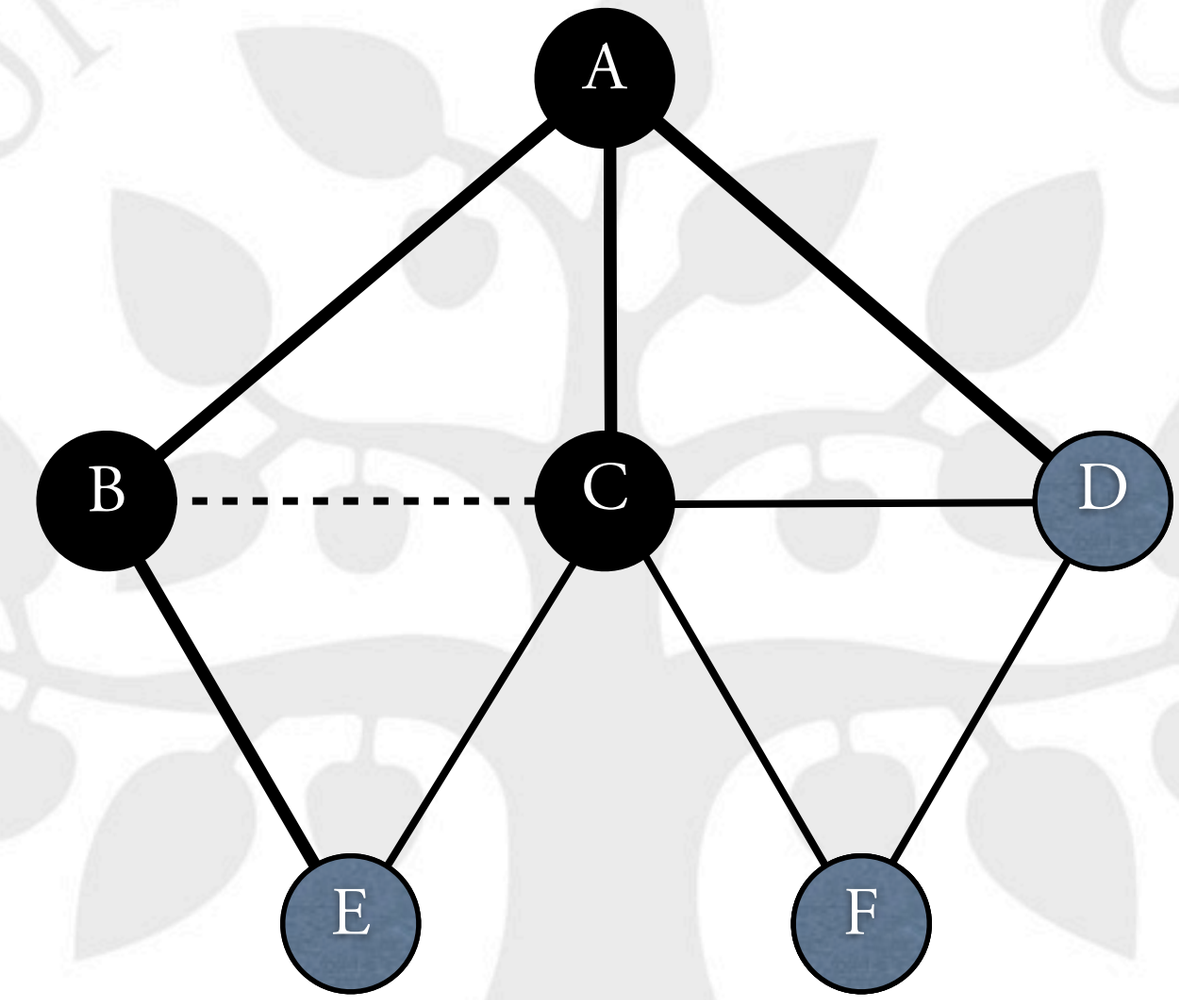


D E

Eksempel

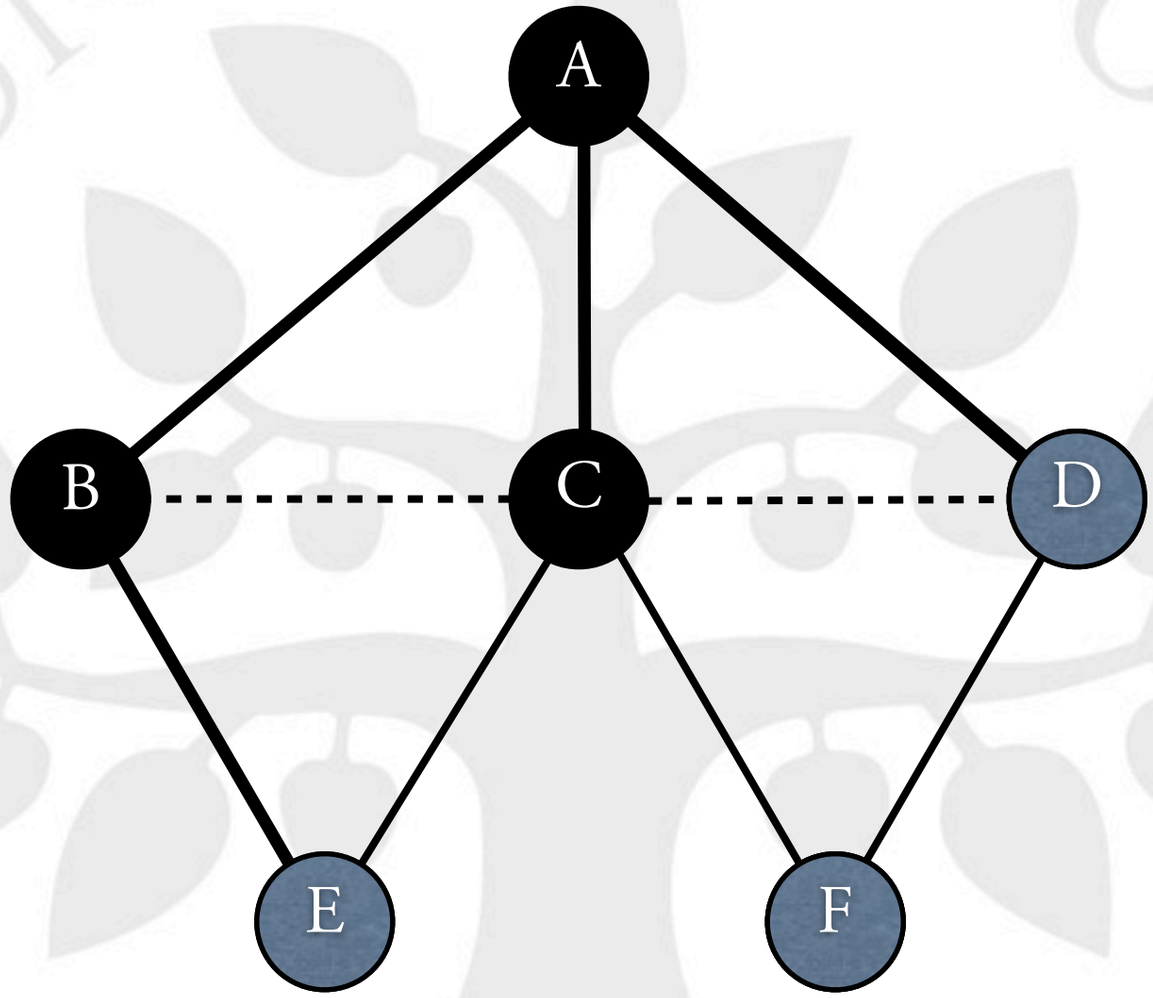


Eksempel



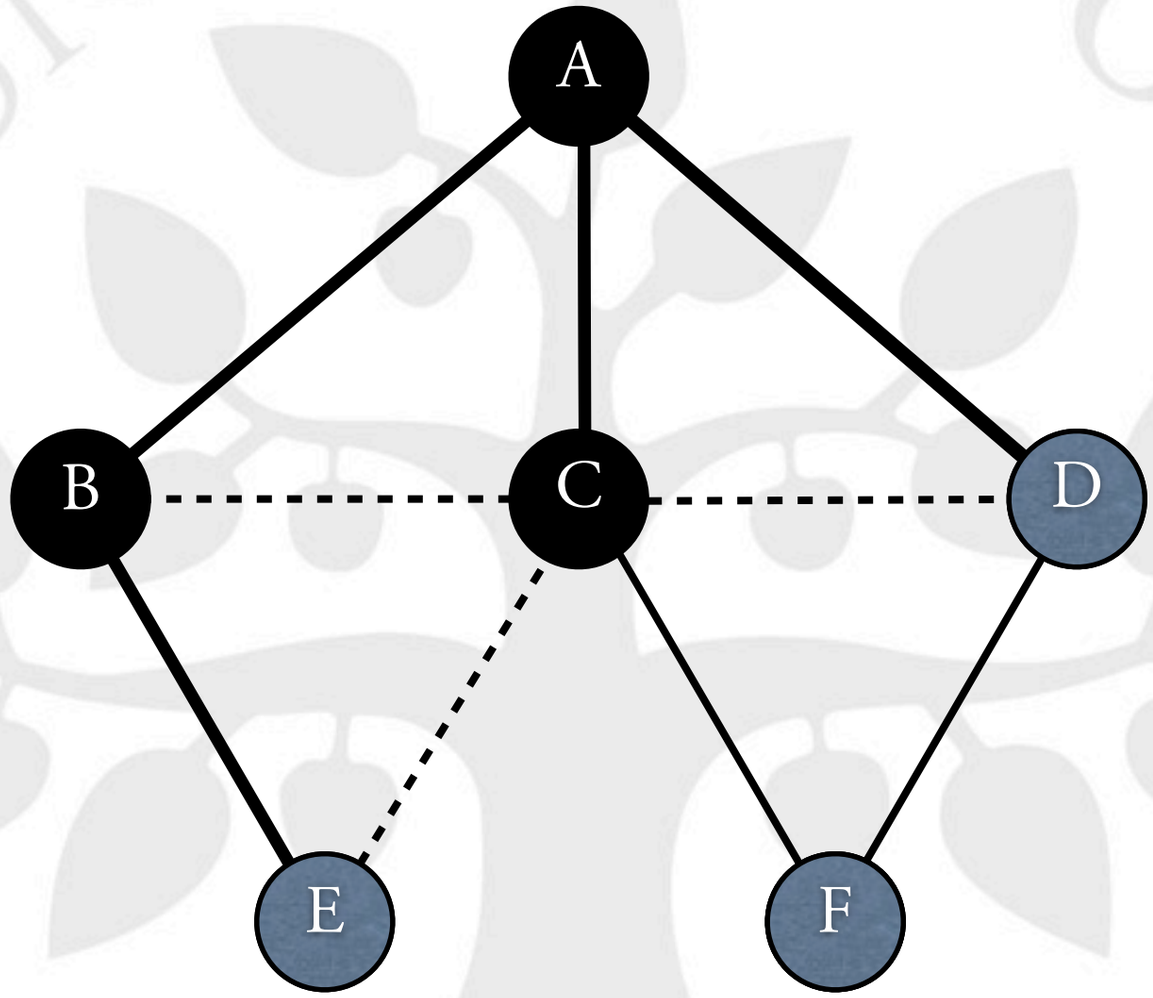
D E

Eksempel

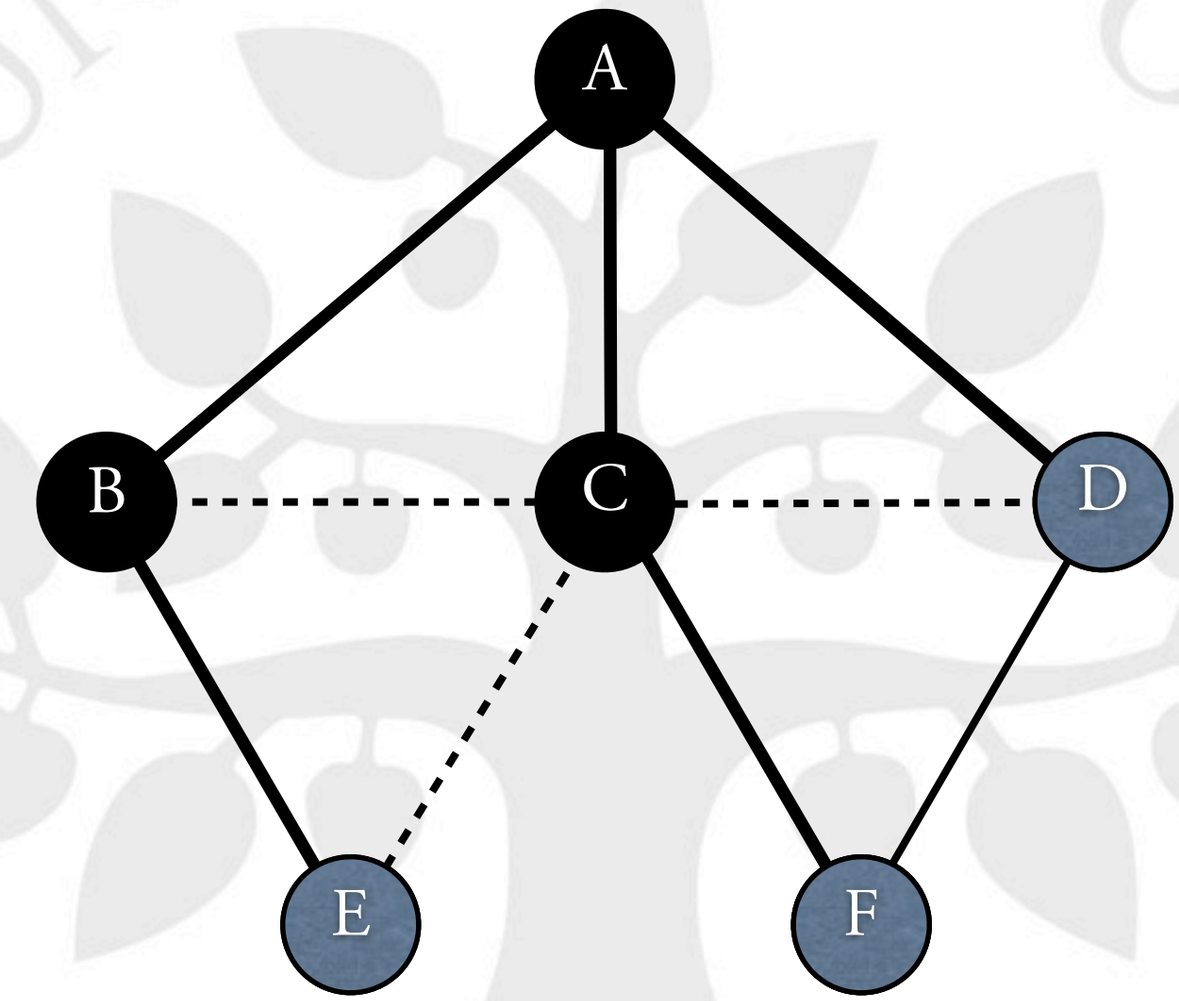


D E

Eksempel

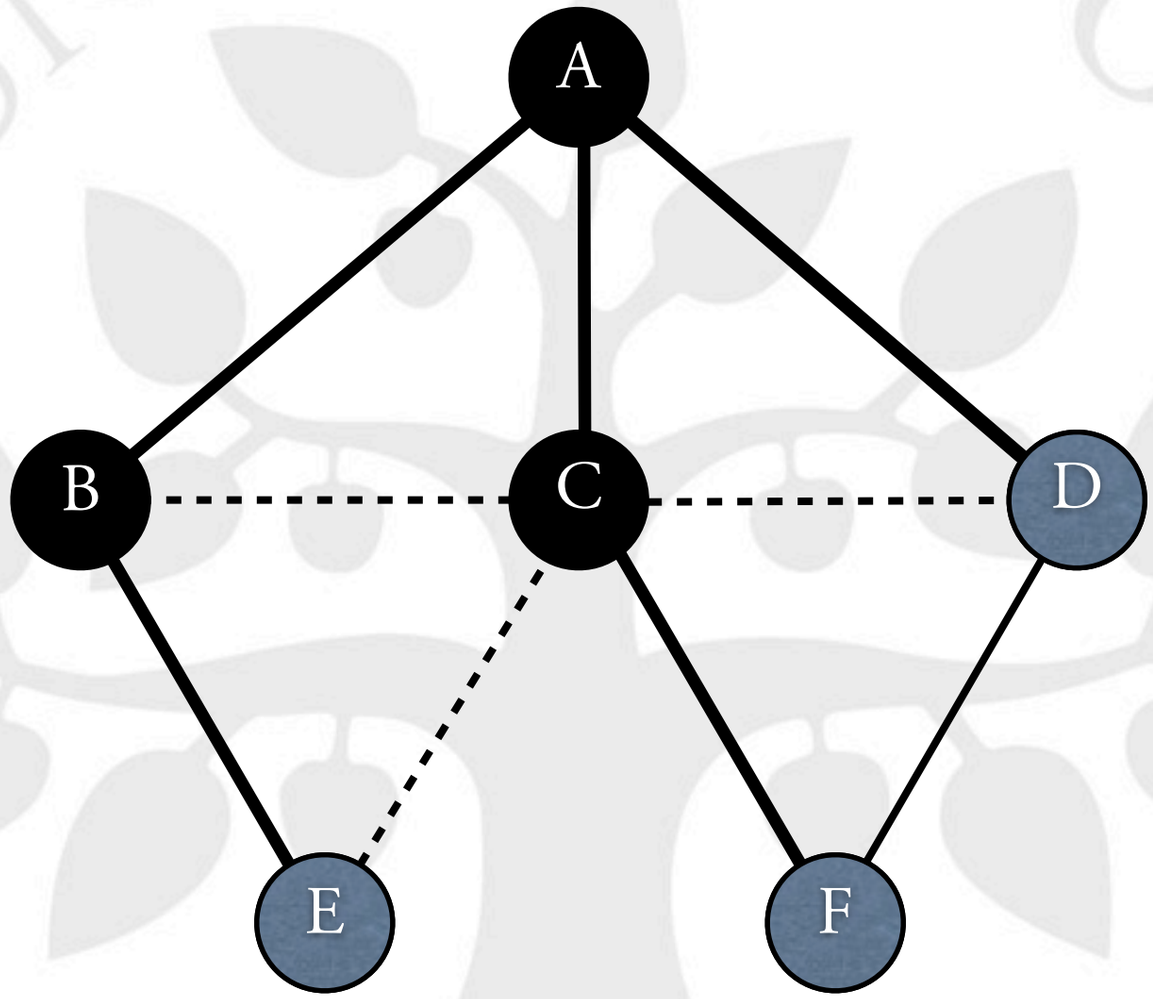


Eksempel

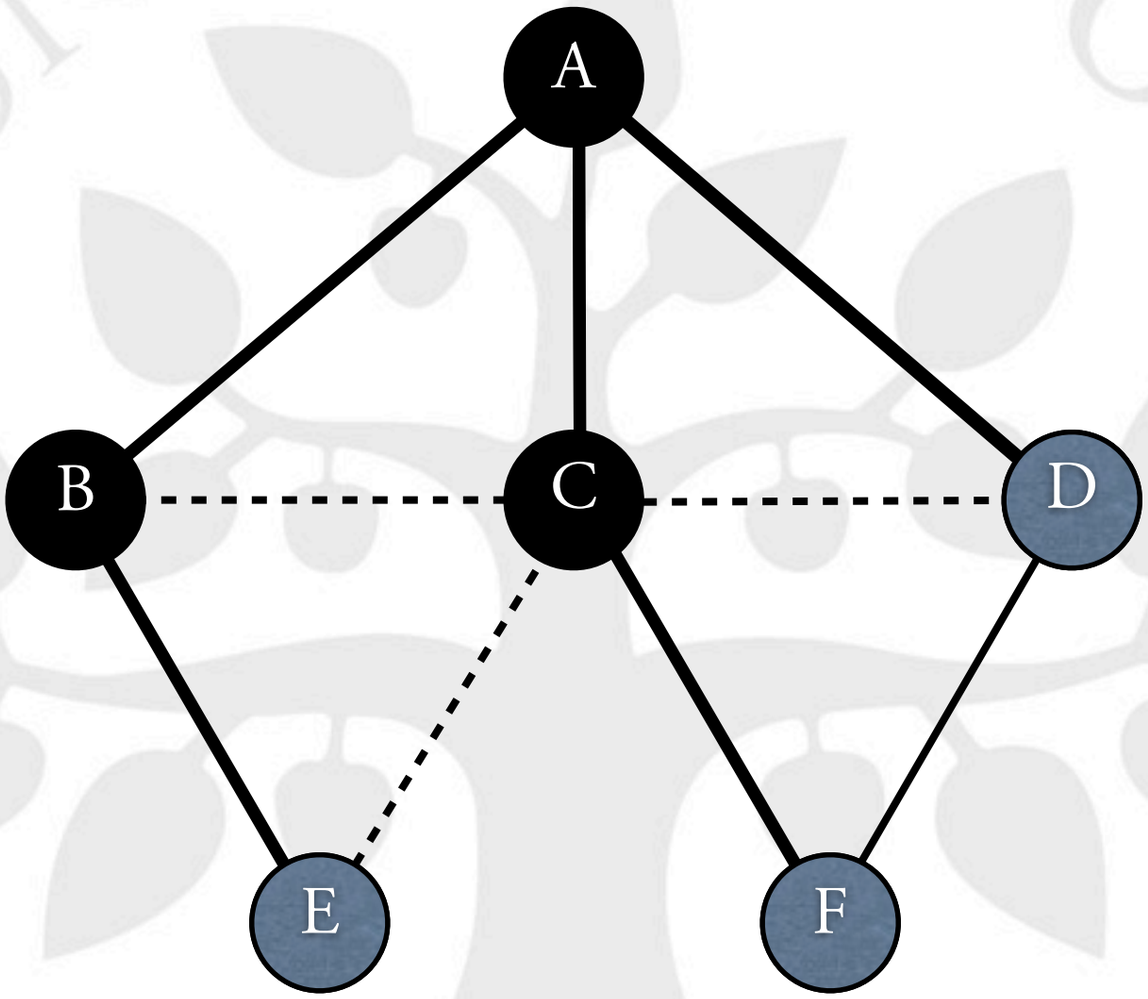


D E F

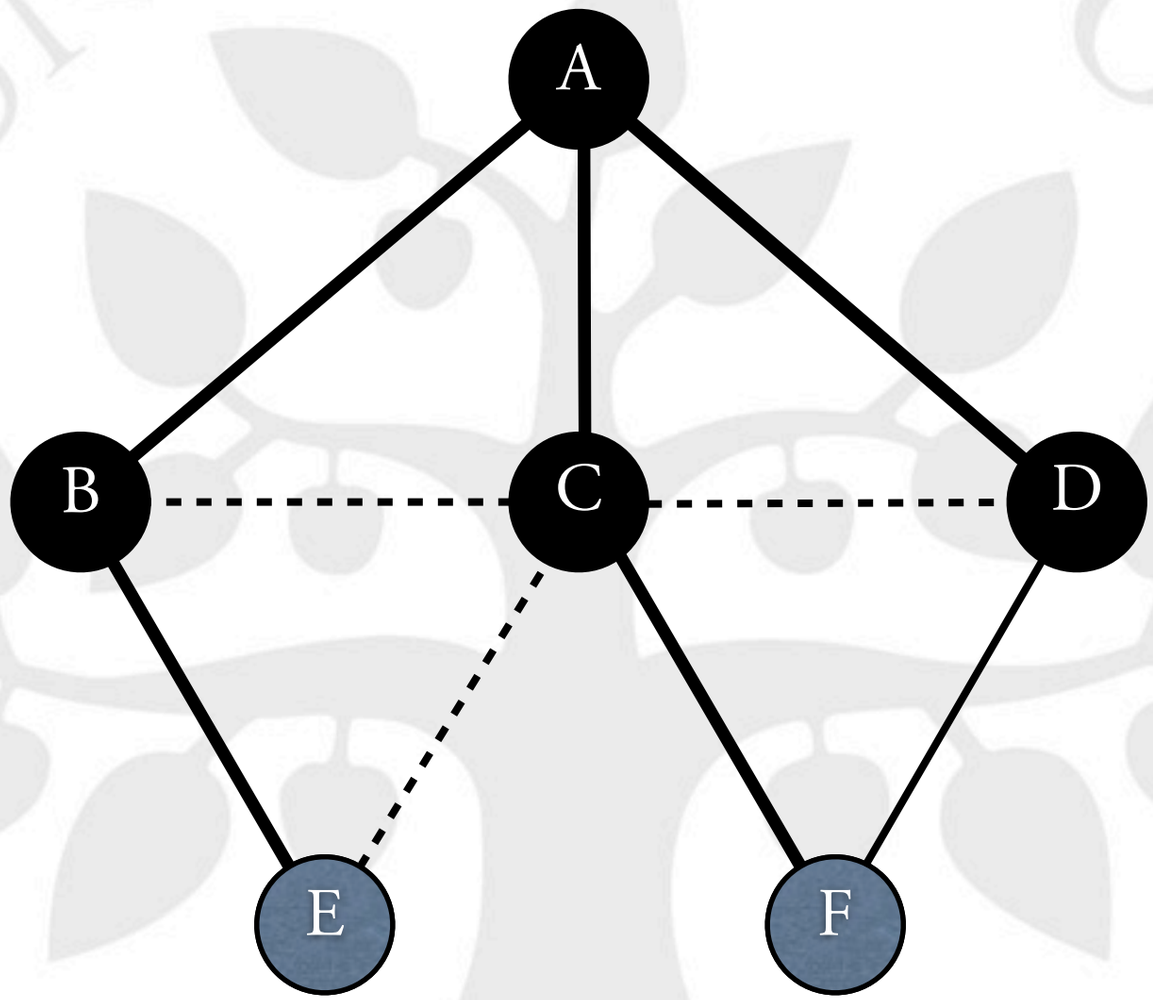
Eksempel



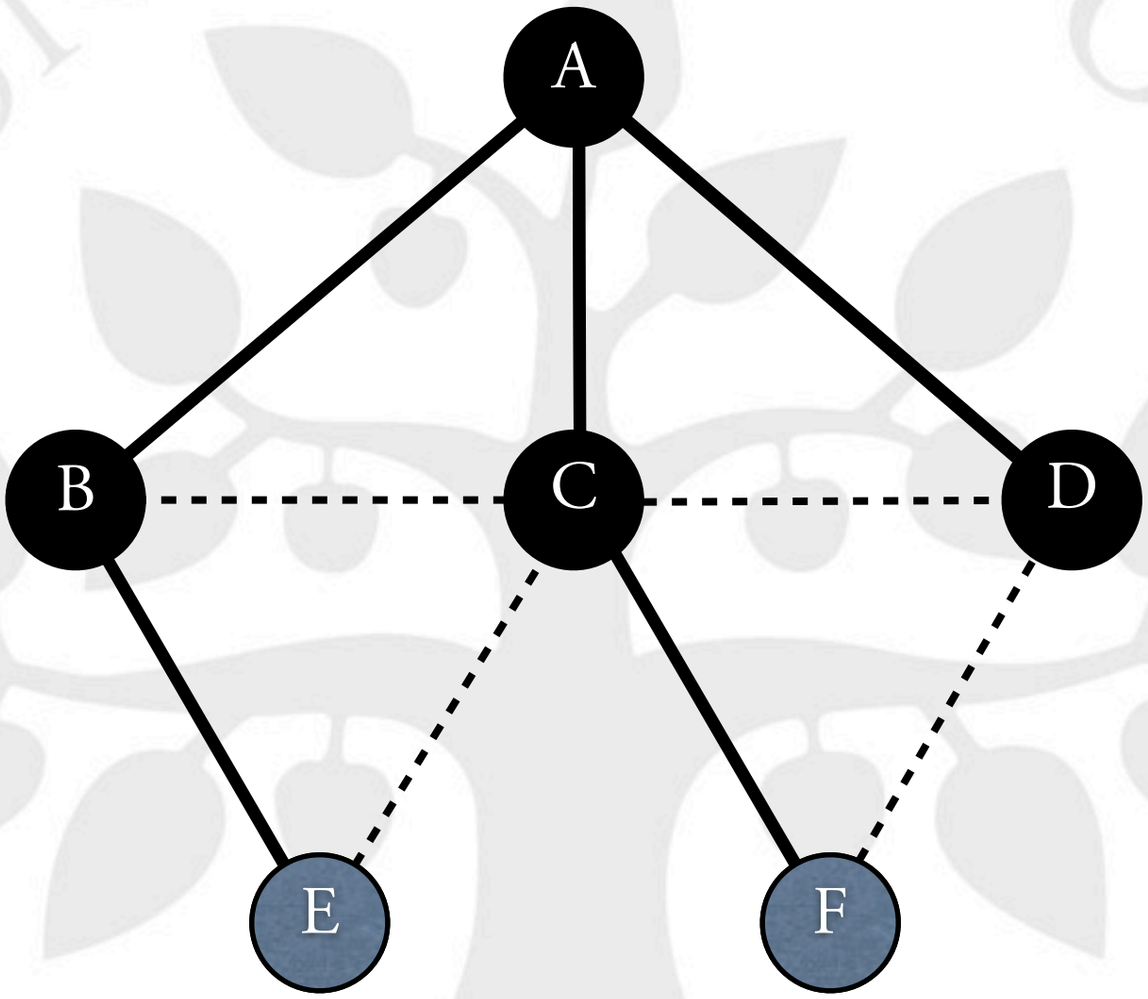
Eksempel



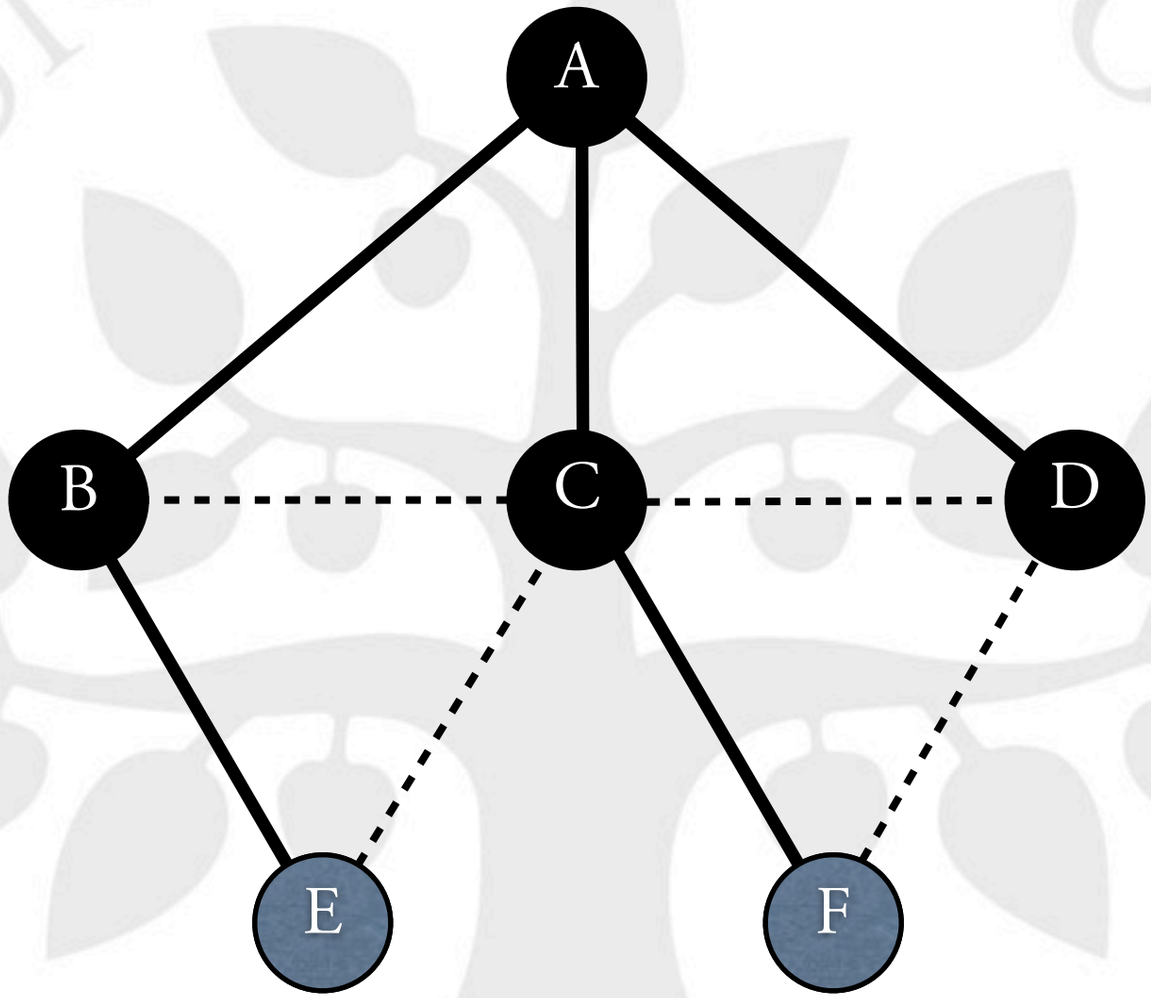
Eksempel



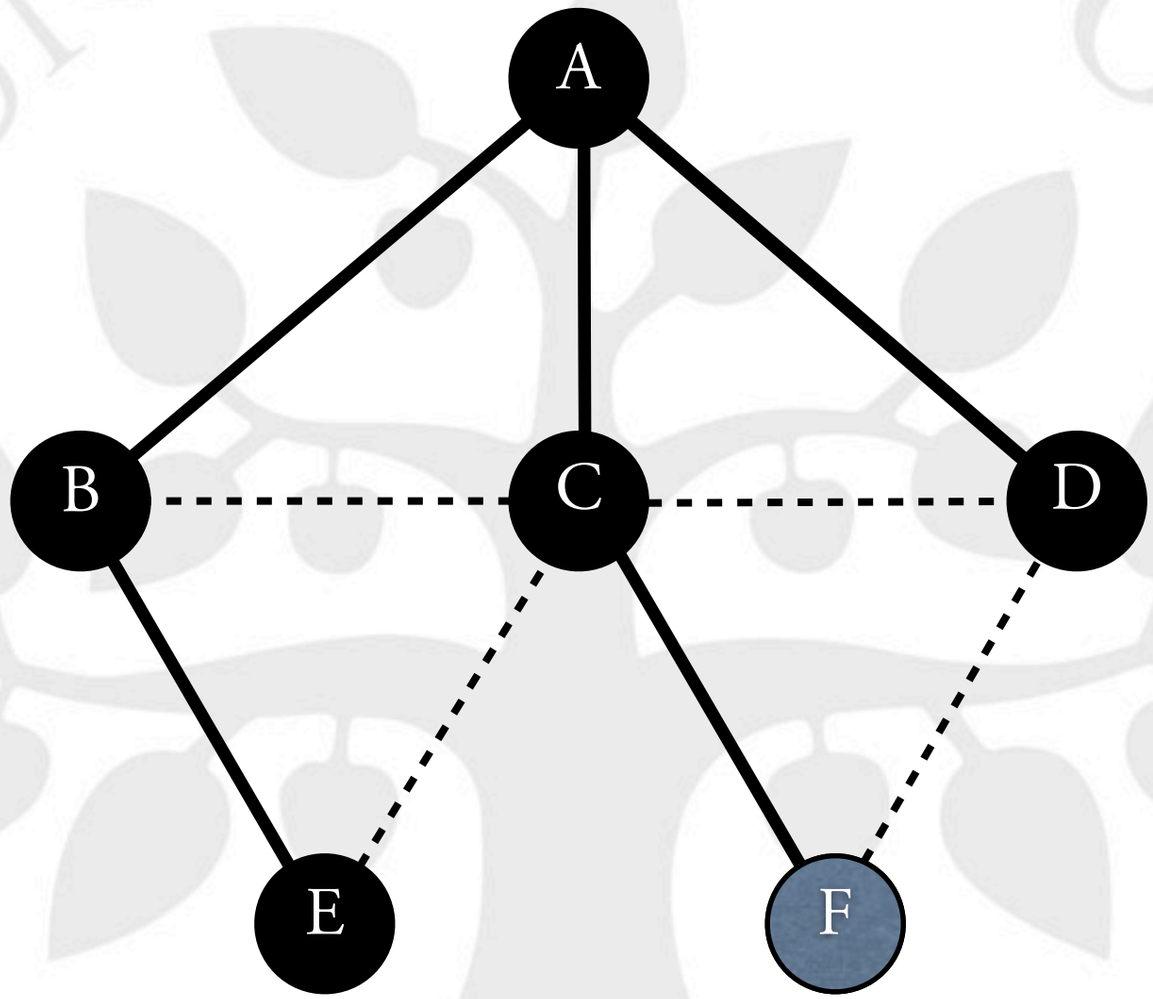
Eksempel



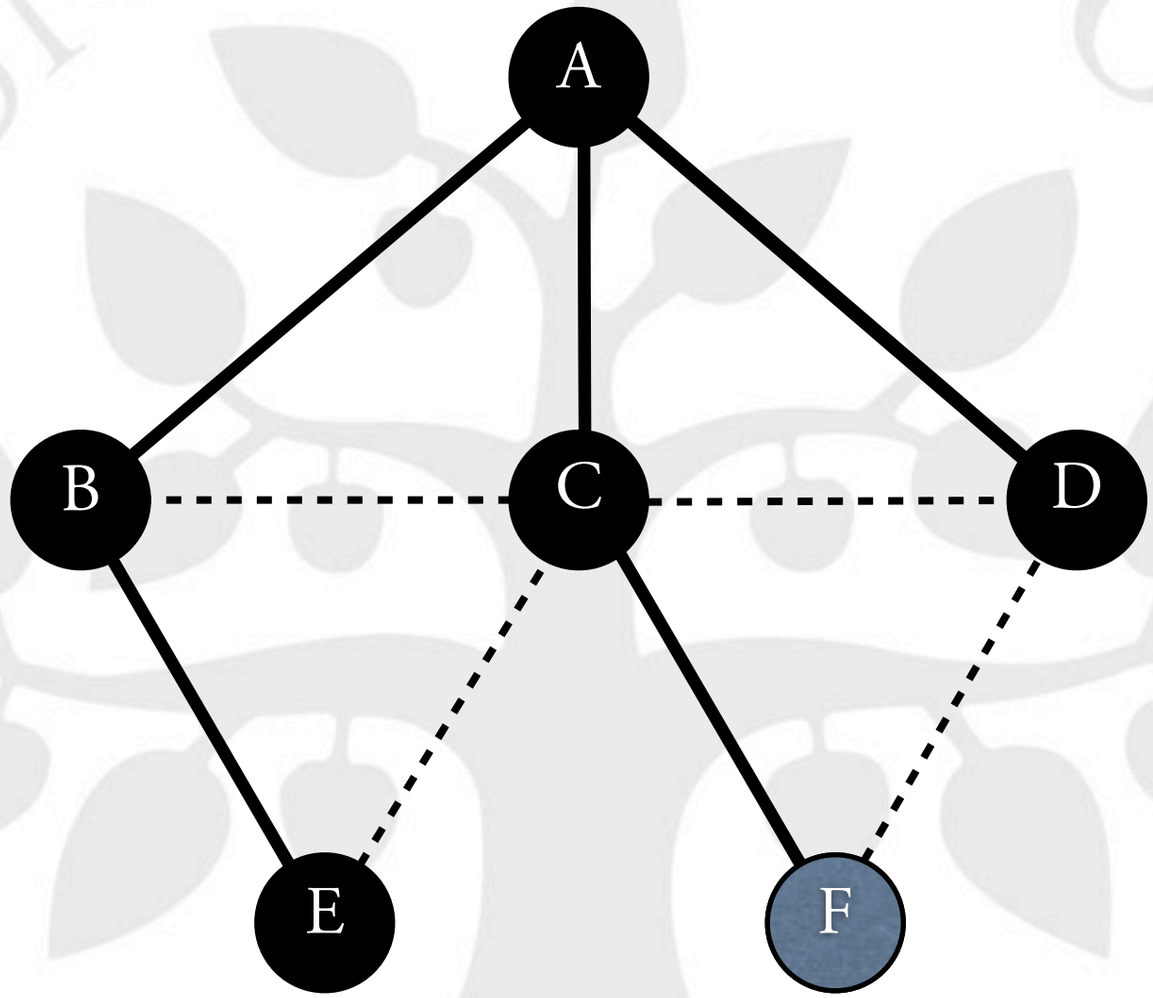
Eksempel



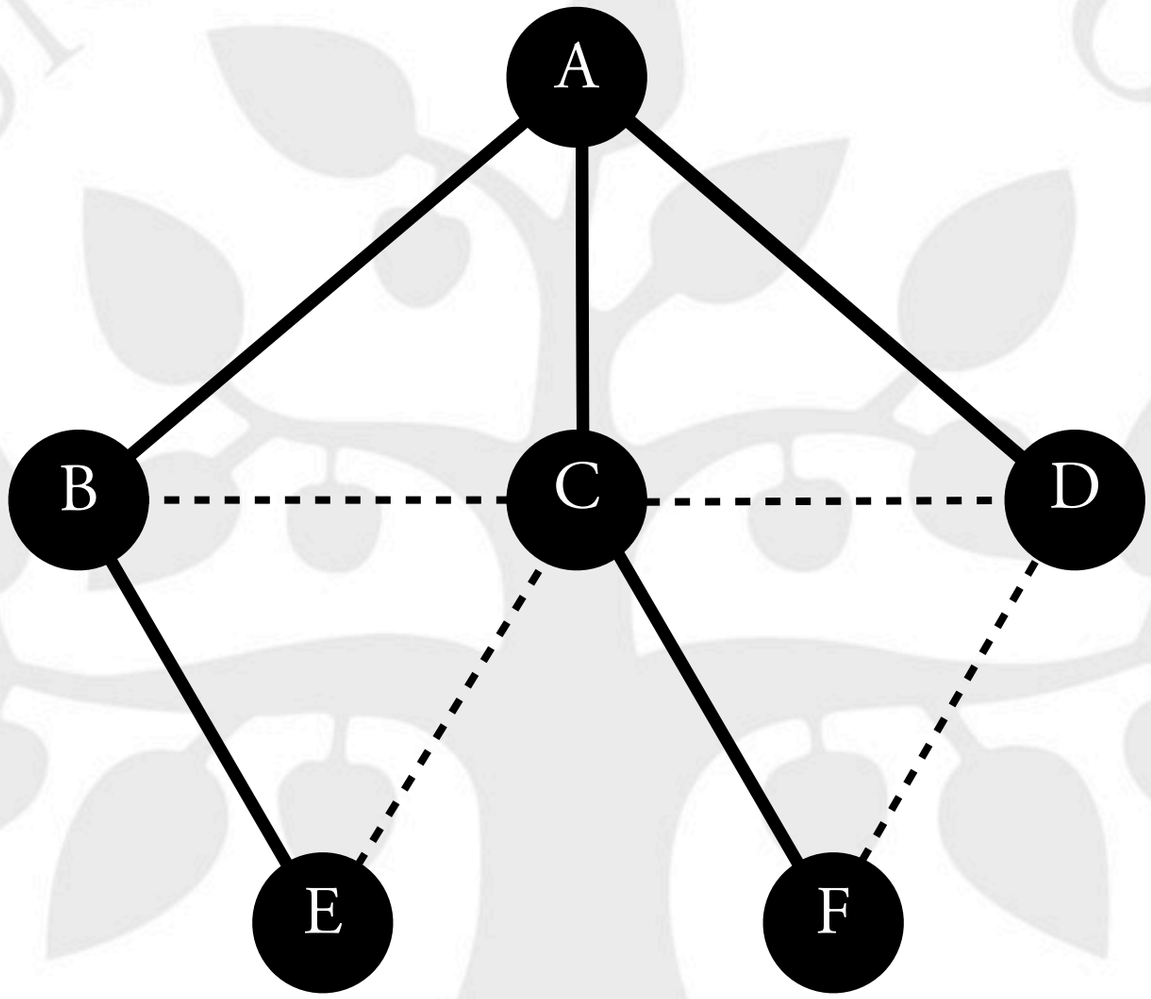
Eksempel



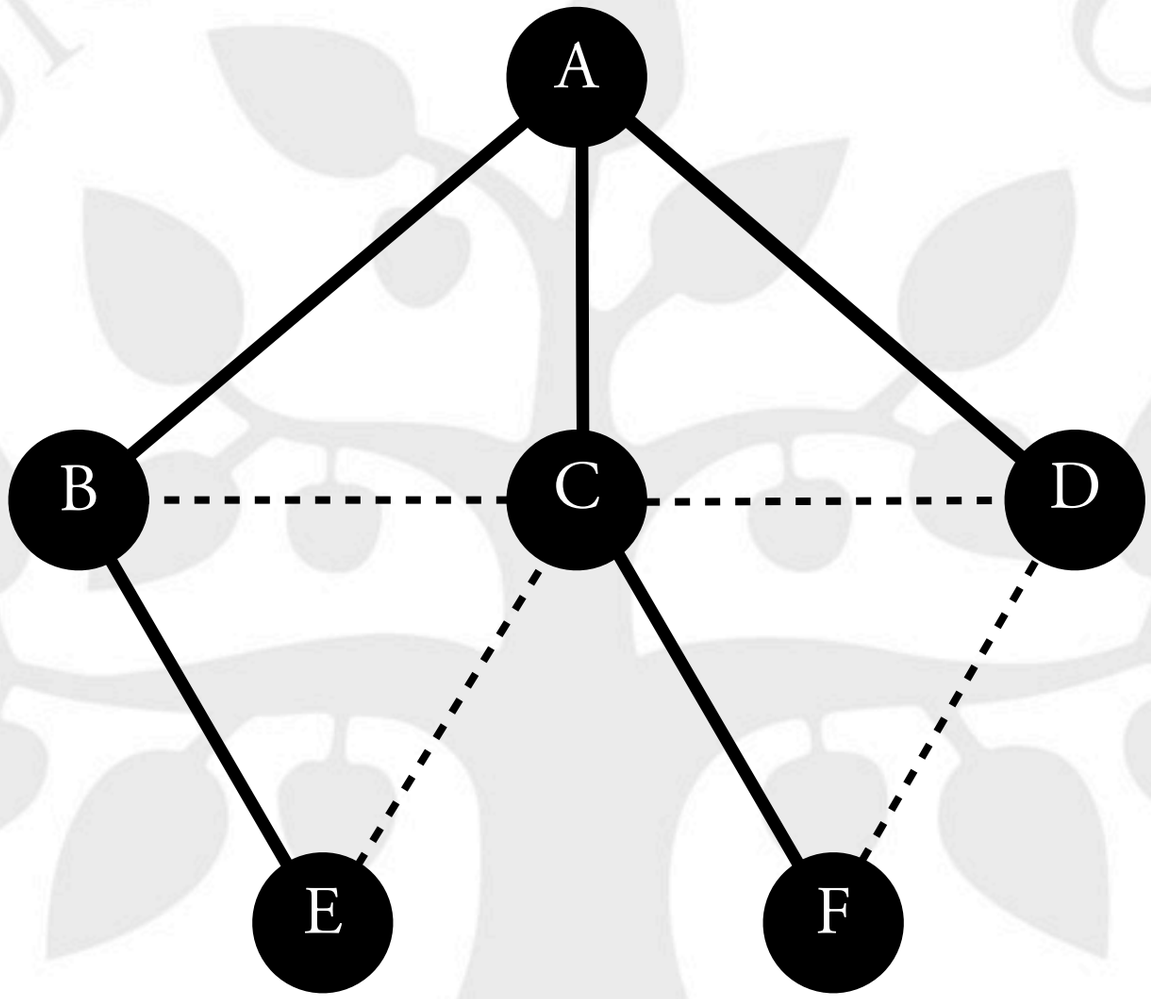
Eksempel



Eksempel

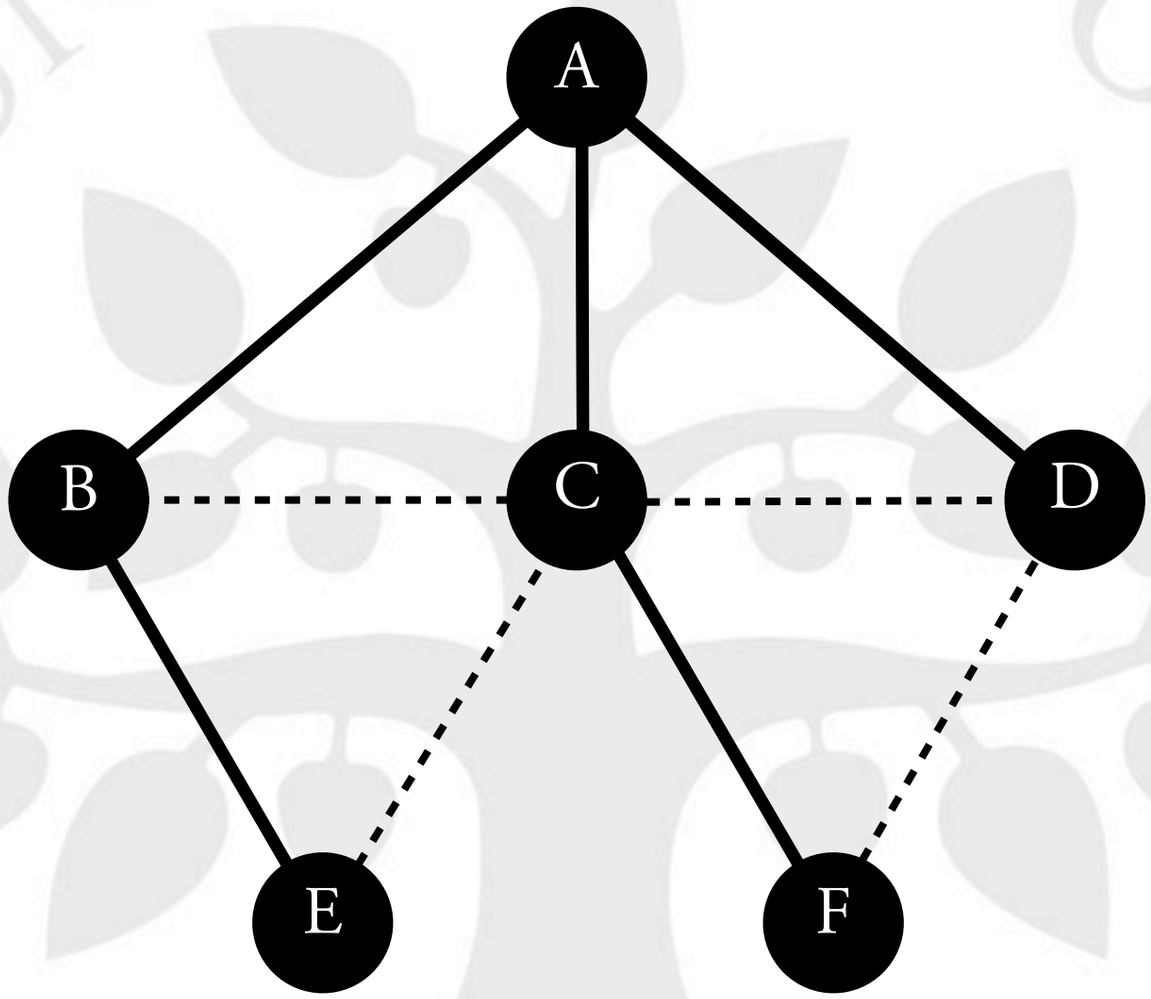


Eksempel



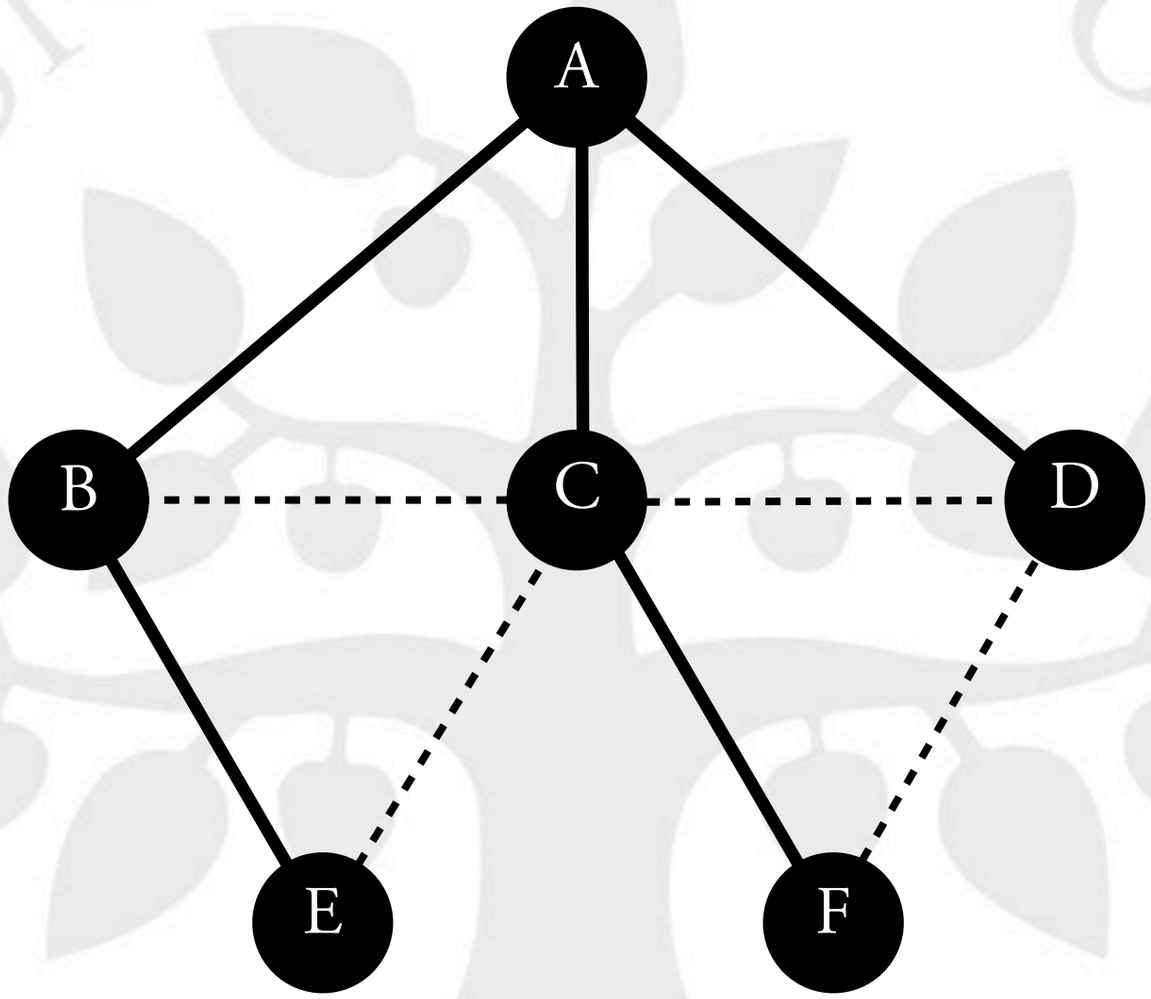
- Hvordan kan BFS udvides til at

Eksempel



- Hvordan kan BFS udvides til at
  - finde en kreds i grafen?

Eksempel



- Hvordan kan BFS udvides til at
  - finde en kreds i grafen?
  - beregne den korteste sti mellem to givne knuder?

BFS's egenskaber

BFS's egenskaber

- Egenskab 1

BFS's egenskaber

- Egenskab 1
  - BFS(G, v) besøger alle knuder og kanter i den sammenhængskomponent v er en del af

BFS's egenskaber

- Egenskab 1
  - BFS(G, v) besøger alle knuder og kanter i den sammenhængskomponent v er en del af
- Egenskab 2

BFS's egenskaber

- Egenskab 1
  - BFS(G, v) besøger alle knuder og kanter i den sammenhængskomponent v er en del af
- Egenskab 2
  - De kanter BFS(G, v) følger danner et udspændende træ T_v af den sammenhængskomponent v er en del af

BFS's egenskaber

- Egenskab 1
  - BFS(G, v) besøger alle knuder og kanter i den sammenhængskomponent v er en del af
- Egenskab 2
  - De kanter BFS(G, v) følger danner et udspændende træ T_v af den sammenhængskomponent v er en del af
- Egenskab 3

BFS's egenskaber

- Egenskab 1
  - BFS(G, v) besøger alle knuder og kanter i den sammenhængskomponent v er en del af
- Egenskab 2
  - De kanter BFS(G, v) følger danner et udspændende træ T_v af den sammenhængskomponent v er en del af
- Egenskab 3
  - For enhver knude w i G gælder at enhver sti fra v til w i G er mindst lige så lang som stien fra v til w i T_v.

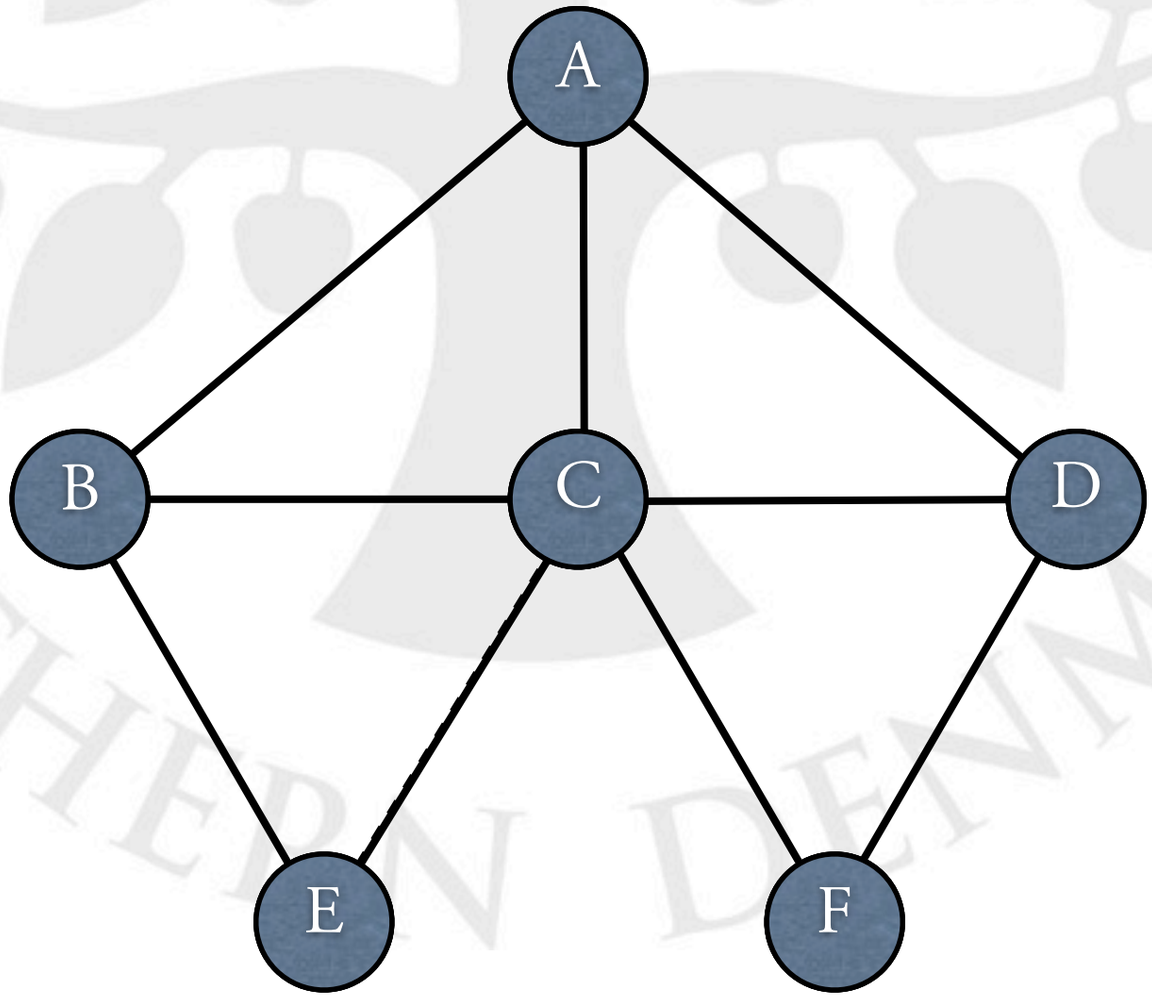
BFS.java

```
import java.util.*;

public class BFS {
    public static void main( String[] args ) {
        Node a = new Node( "A" );
        Node b = new Node( "B" );
        Node c = new Node( "C" );
        Node d = new Node( "D" );
        Node e = new Node( "E" );
        Node f = new Node( "F" );

        a.addNeighbour( b );
        a.addNeighbour( c );
        a.addNeighbour( d );

        b.addNeighbour( a );
        b.addNeighbour( c );
        b.addNeighbour( e );

        c.addNeighbour( a );
        c.addNeighbour( b );
        c.addNeighbour( d );
        c.addNeighbour( e );
        c.addNeighbour( f );

        d.addNeighbour( a );
        d.addNeighbour( c );
        d.addNeighbour( f );
```

BFS.java (cont.)

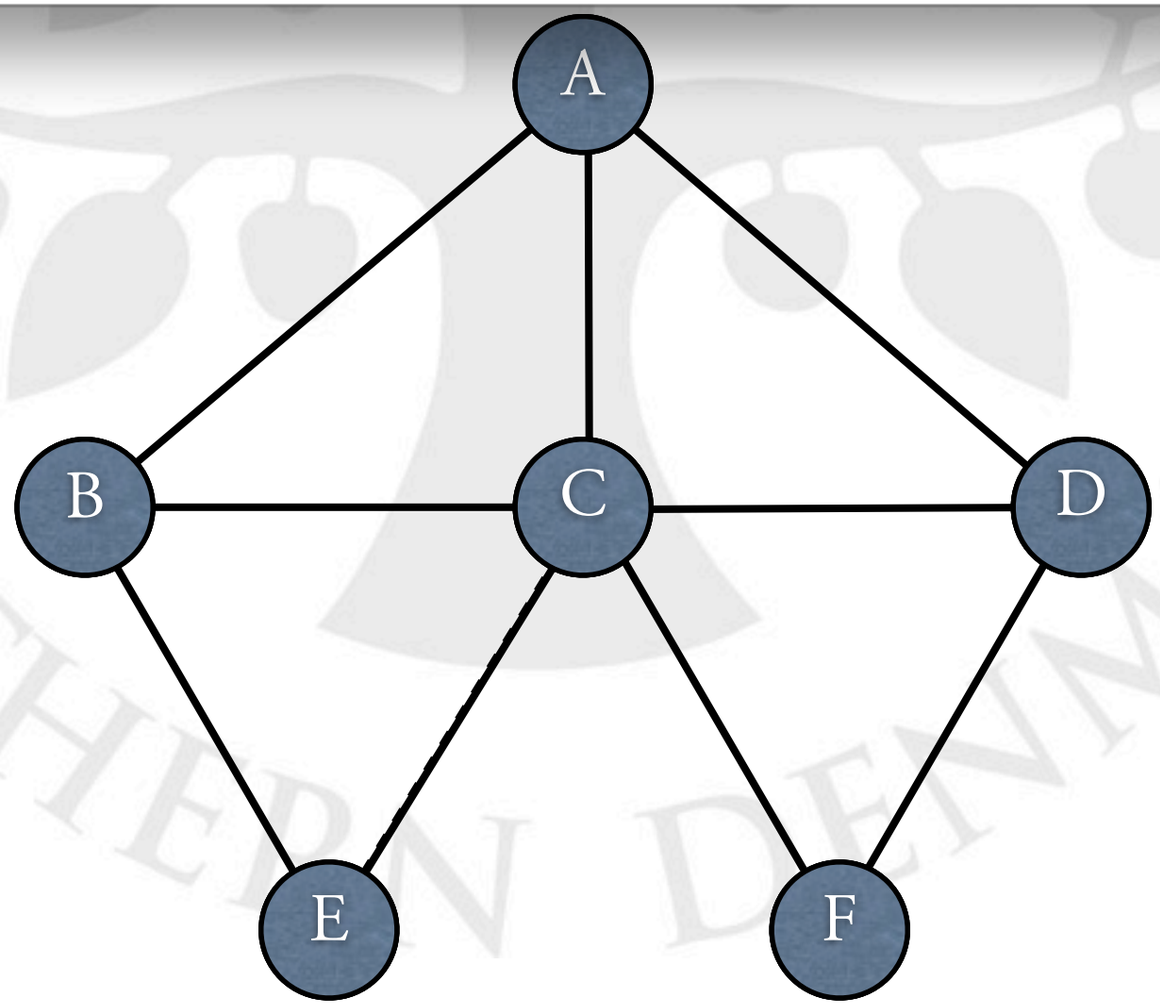
```
        e.addNeighbour( b );
        e.addNeighbour( c );

        f.addNeighbour( c );
        f.addNeighbour( d );

        BFS( a );
    }

    public static void BFS( Node v ) {
        ArrayDeque<Node> q = new ArrayDeque<Node>();
        v.setMark( true );
        q.add( v );
        while( !q.isEmpty() ) {
            Node u = q.remove();
            System.out.println( u + ": Visting" );
            for( Node w : u.getNeighbours() ) {
                if( !w.getMark() ) {
                    System.out.println( u + ": Adding "+w+" to the queue" );
                    w.setMark( true );
                    q.add( w );
                }
            }
        }
    }
}
```

Output



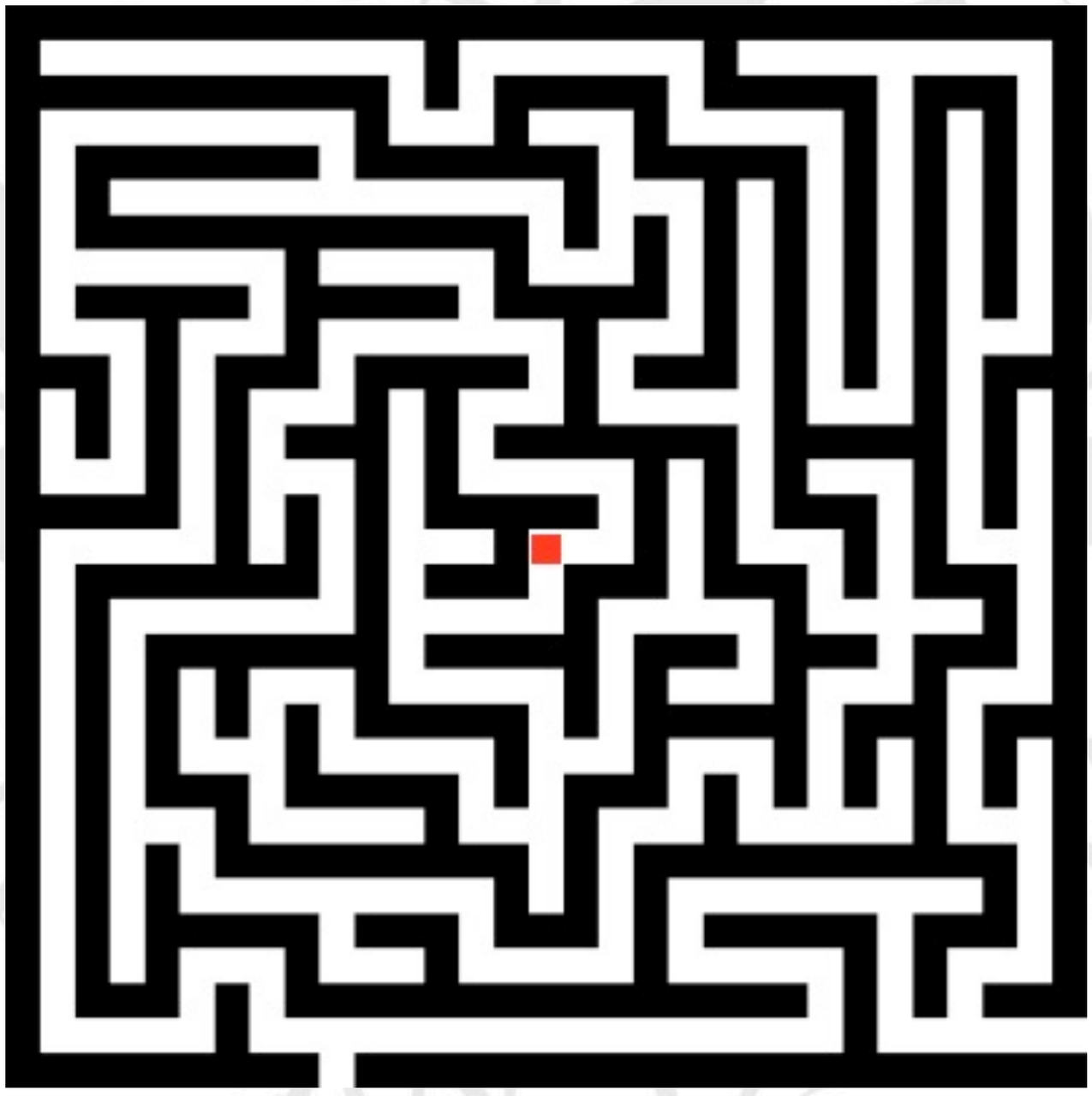
Output

```
Terminal — bash — 66×14
MAC-SDU-00001:8 petersk$ javac BFS.java
MAC-SDU-00001:8 petersk$ java BFS
A: Visting
A: Adding B to the queue
A: Adding C to the queue
A: Adding D to the queue
B: Visting
B: Adding E to the queue
C: Visting
C: Adding F to the queue
D: Visting
E: Visting
F: Visting
MAC-SDU-00001:8 petersk$
```

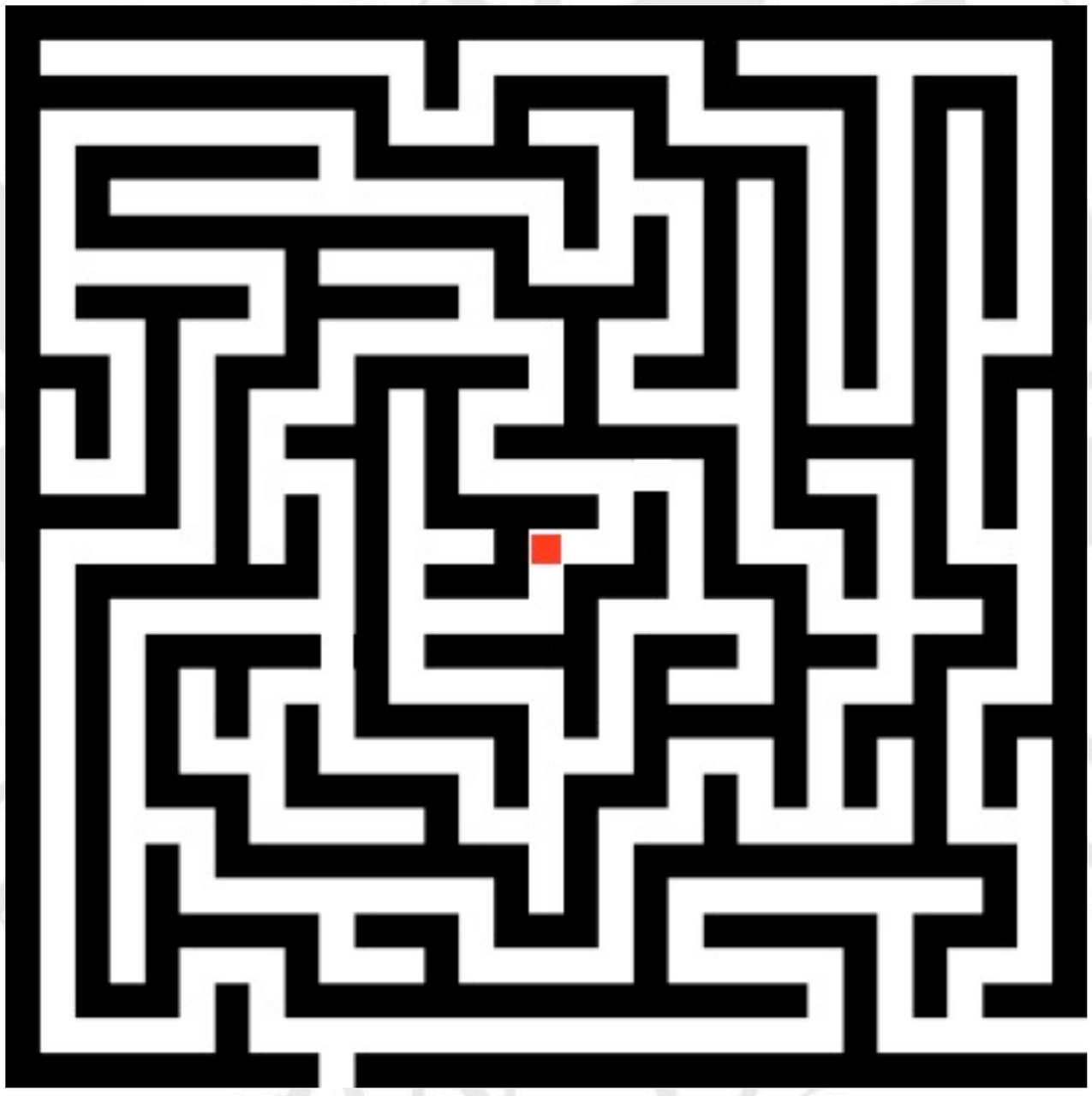


Labyrint

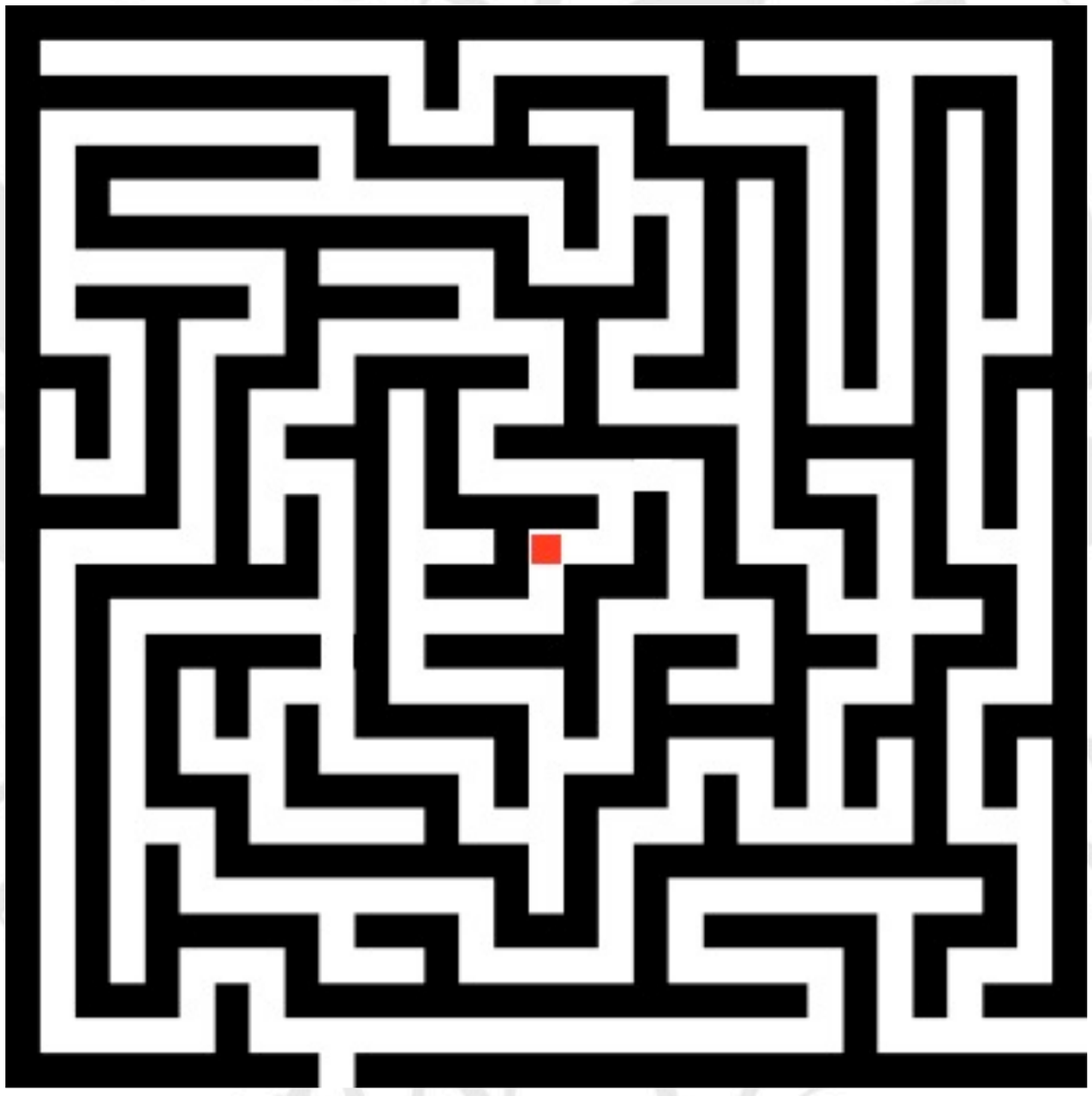
- helten bliver droppet i midten og skal finde en udgang
- helten har et kort over labyrinten og kan planlægge



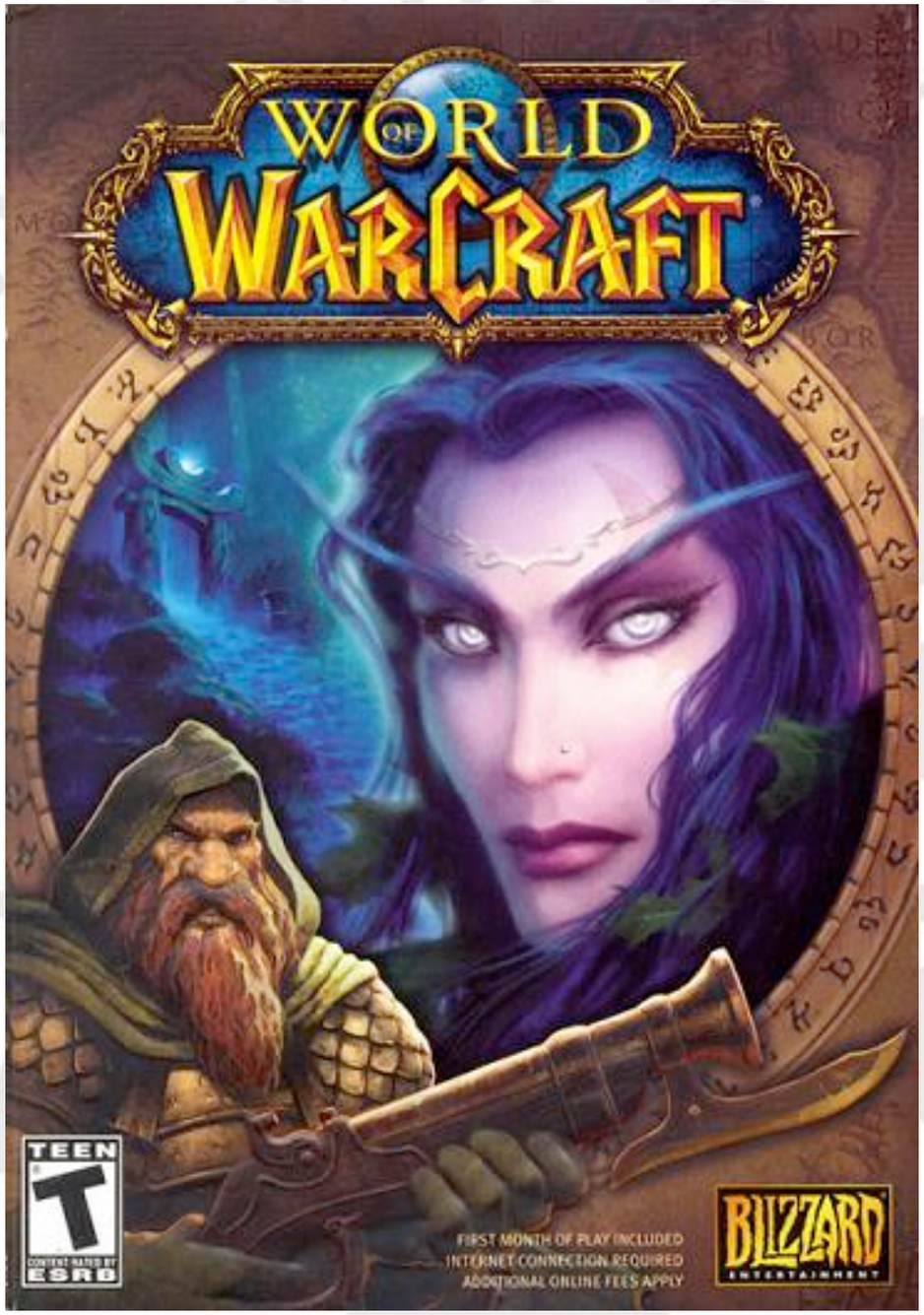
Hvilken strategi ville du bruge for at planlægge vejen?



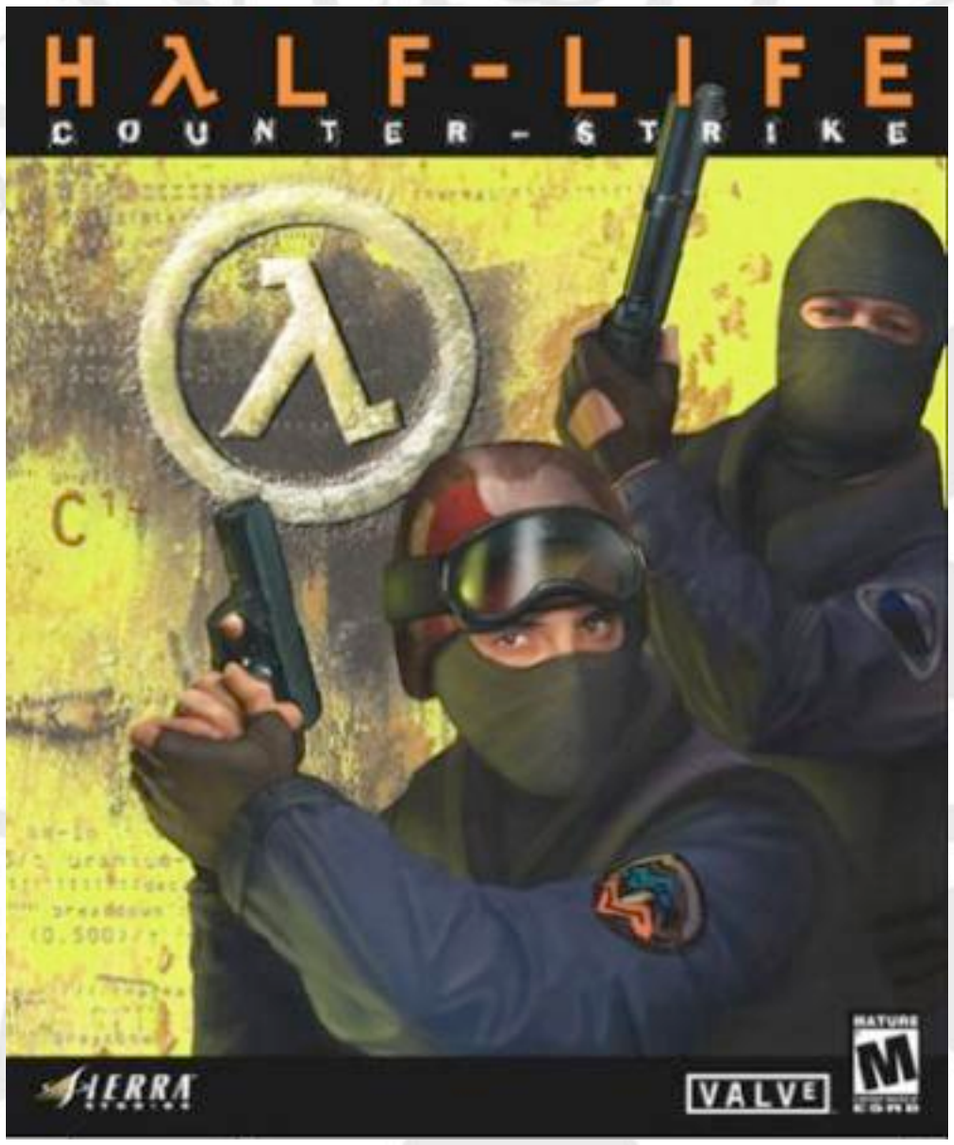
Hvordan kan man anvende brede-først-søgning?



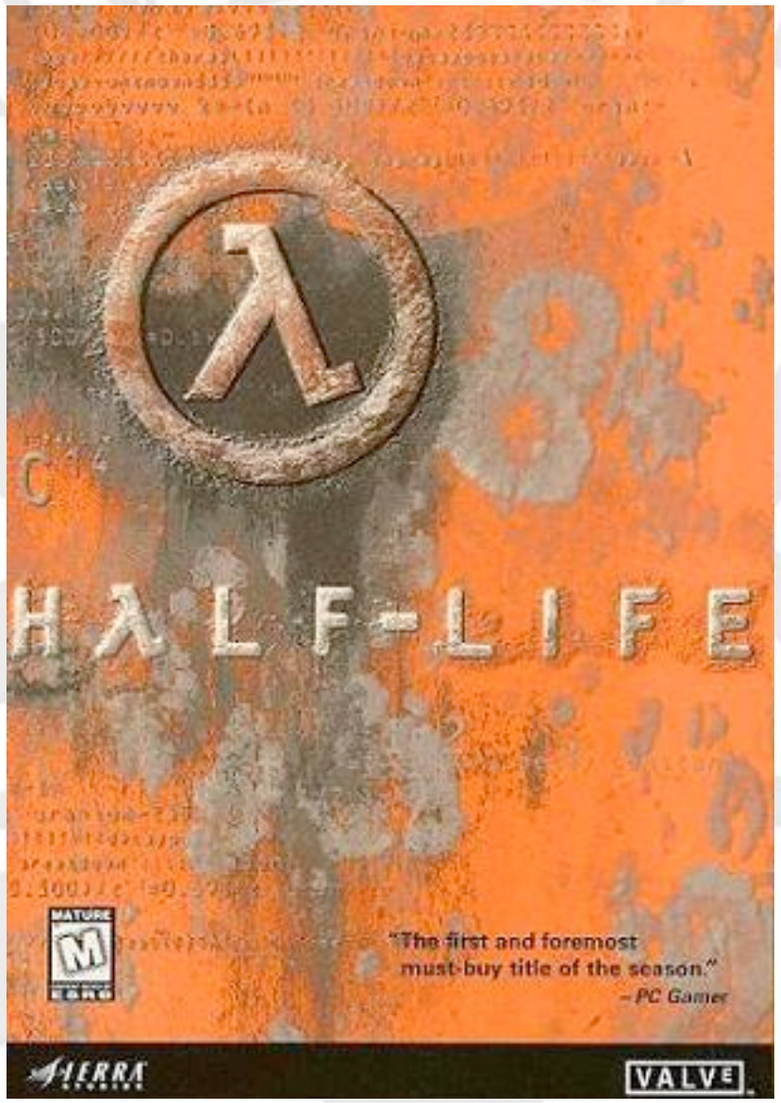
2. Eksamensdelprojekt



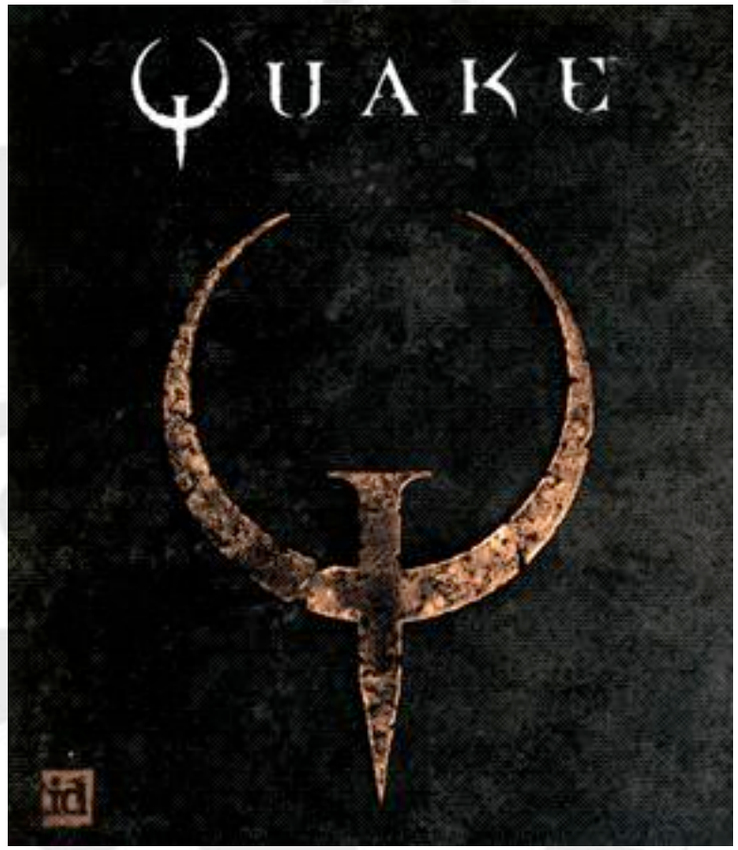
World of Warcraft - 2004



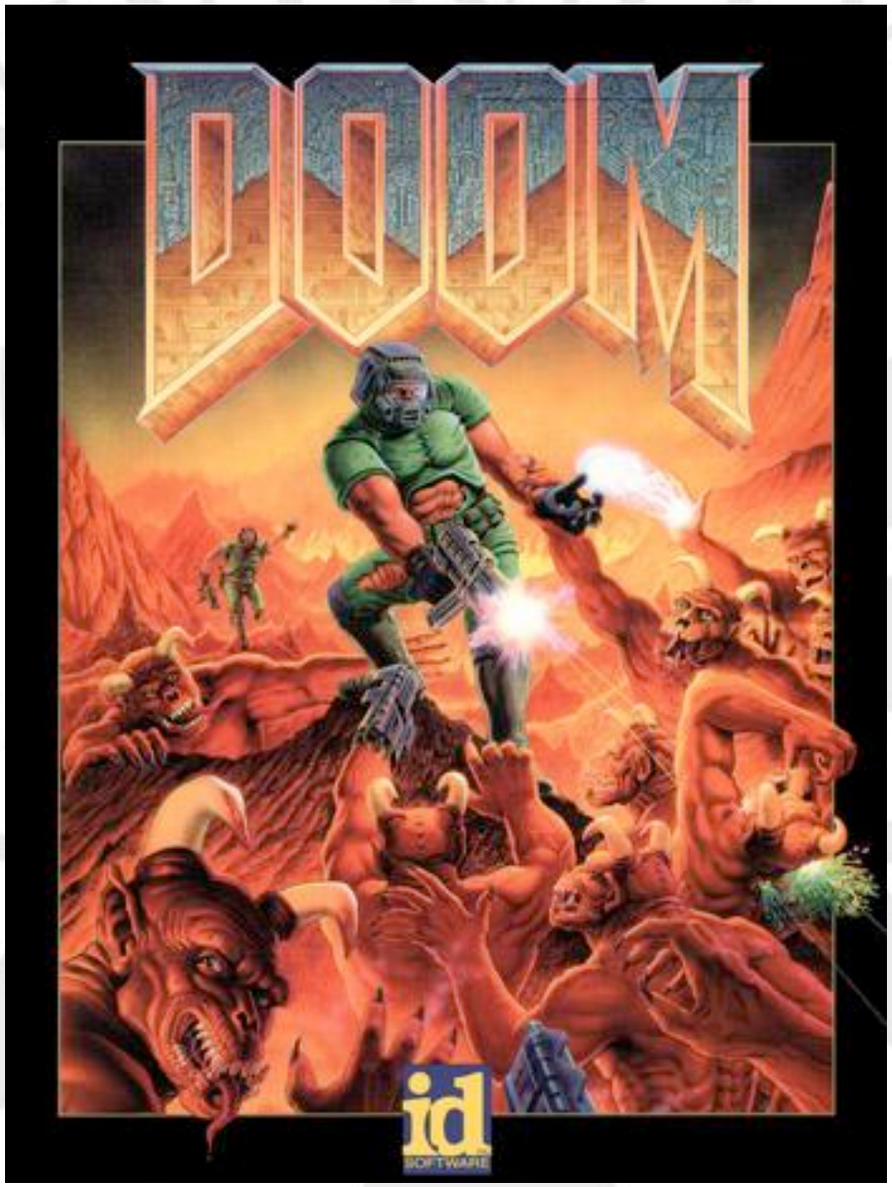
Counter Strike - 2000



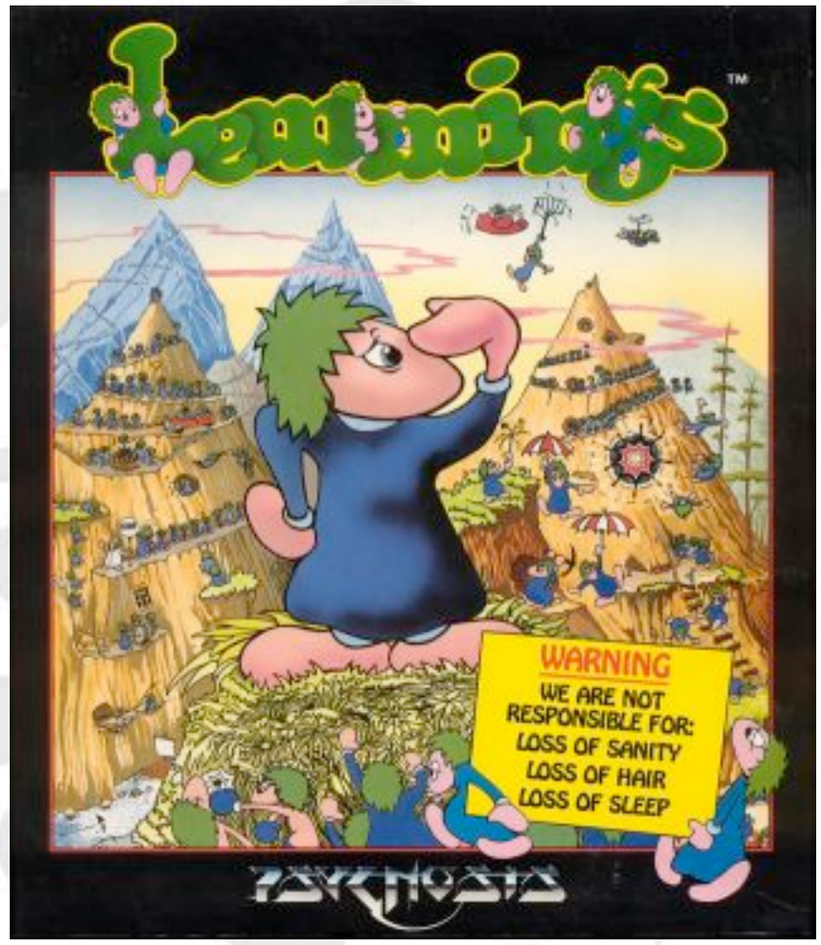
Half-Life - 1998



Diablo & Quake - 1996

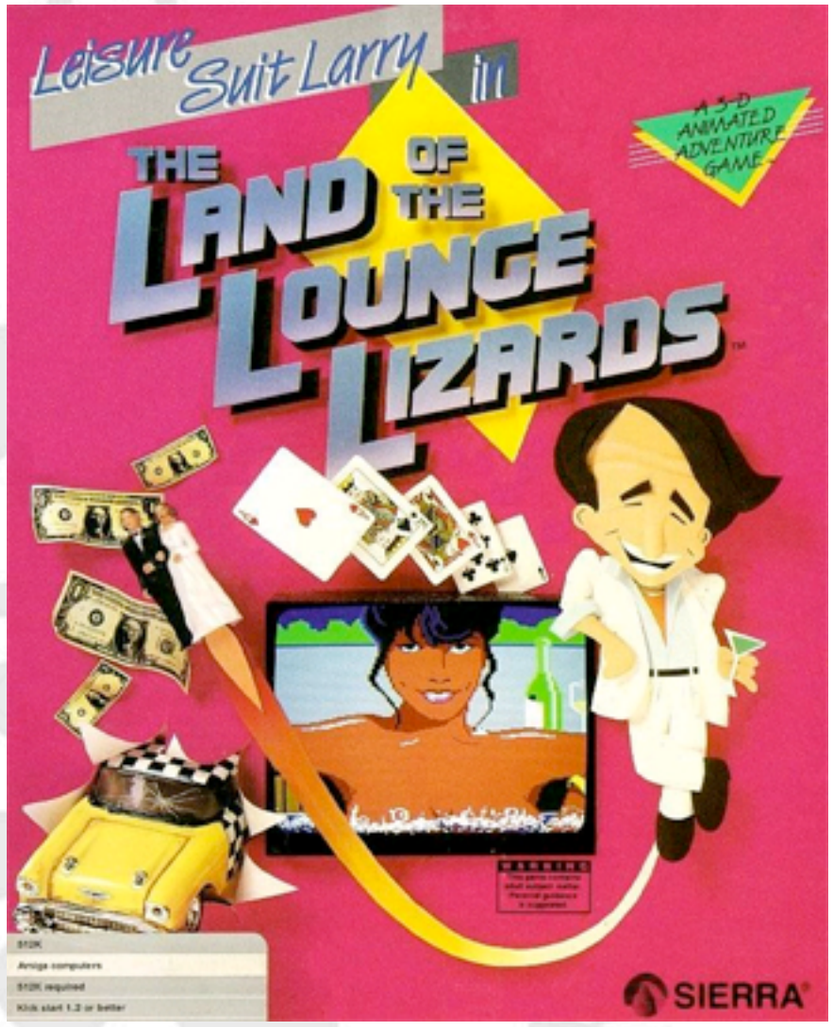


Doom - 1993

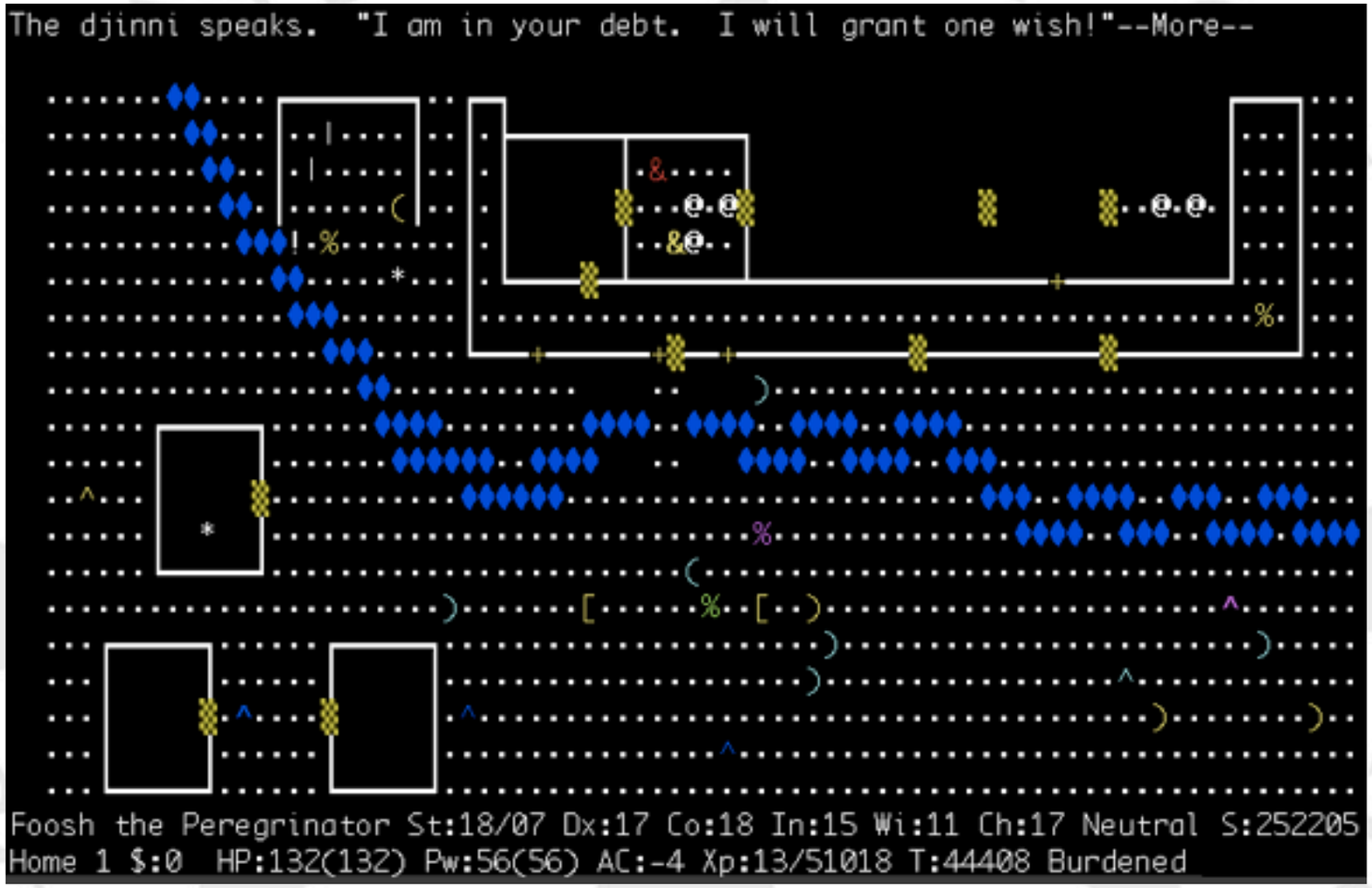


Gorillas & Lemmings - 1991

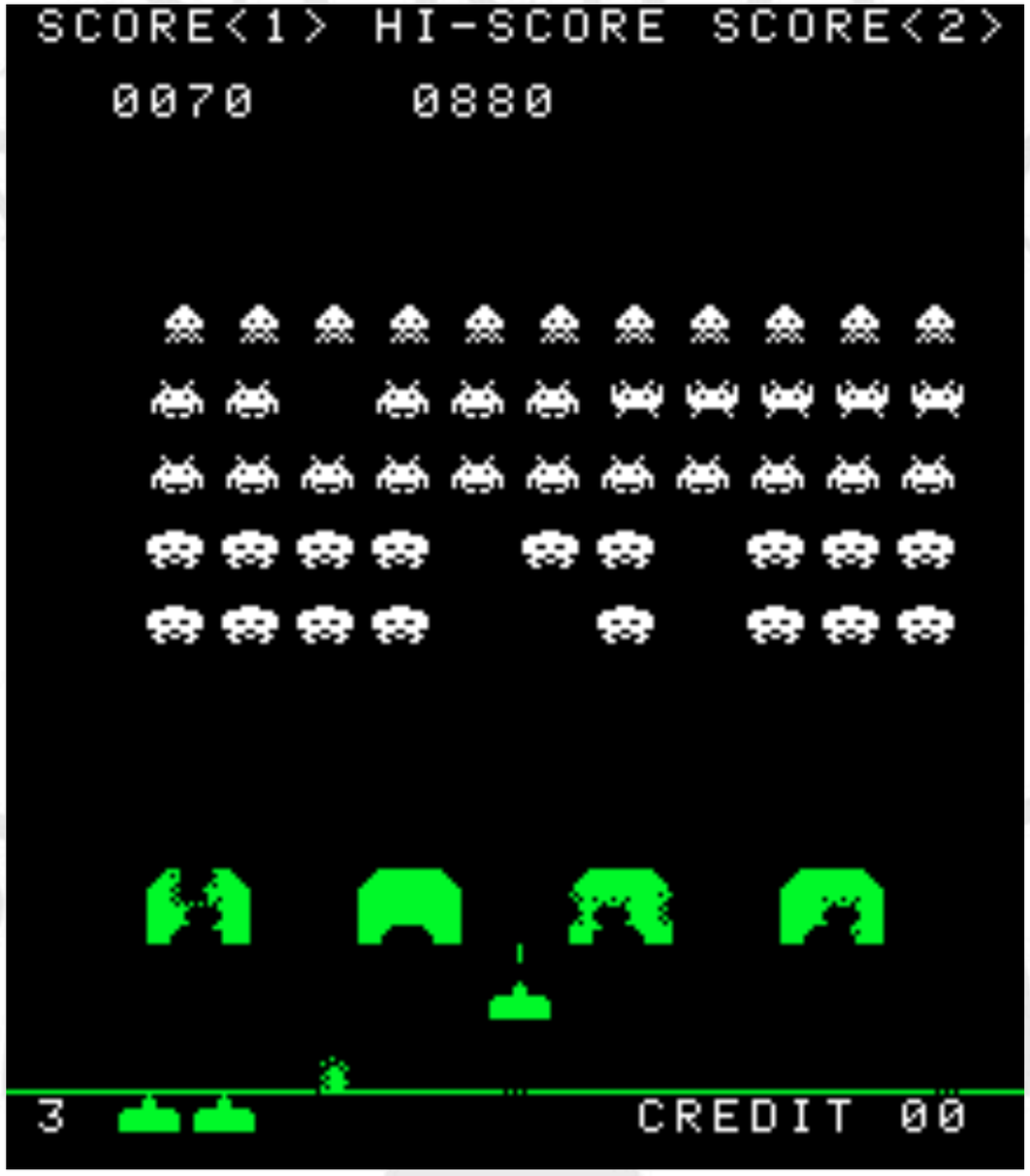
Captain Comic - 1988



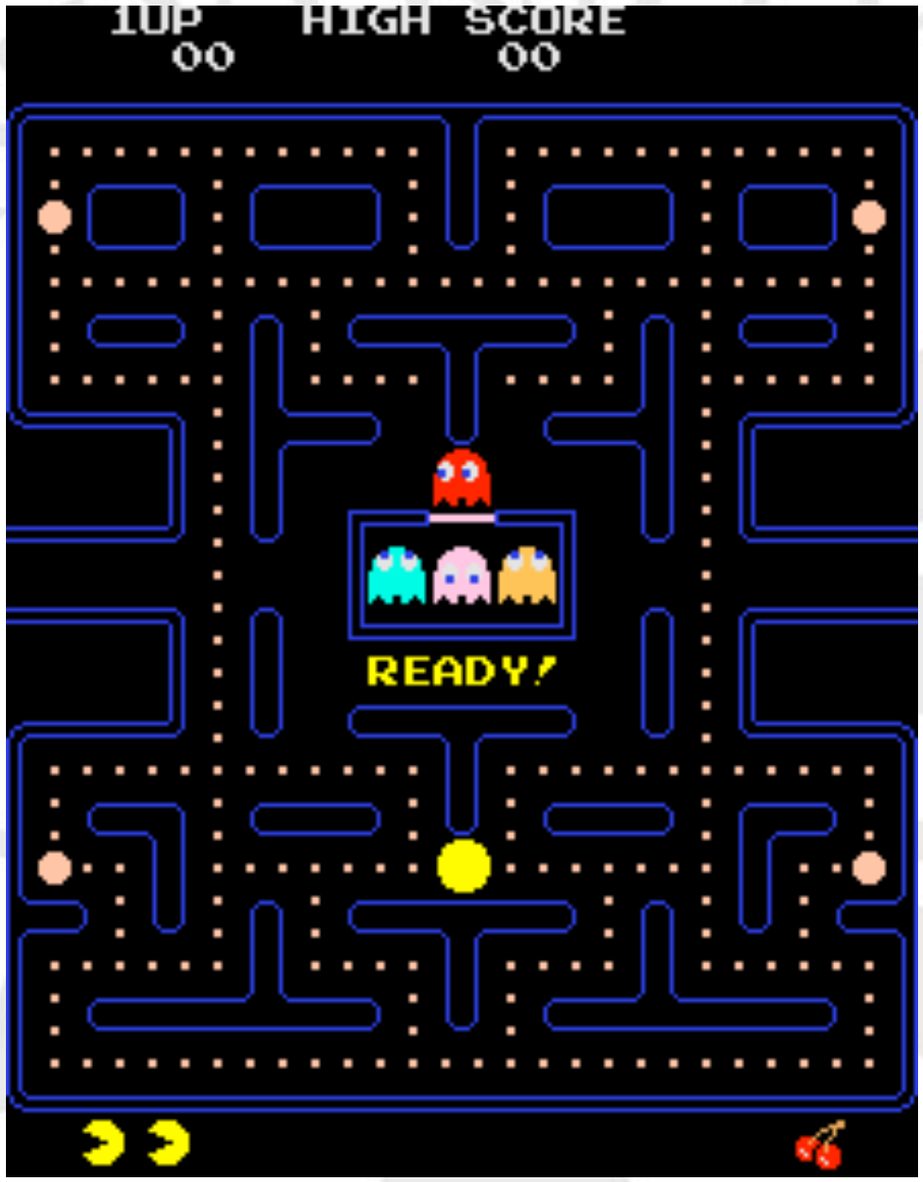
International Karate +
&
Leisure Suit Larry - 1987



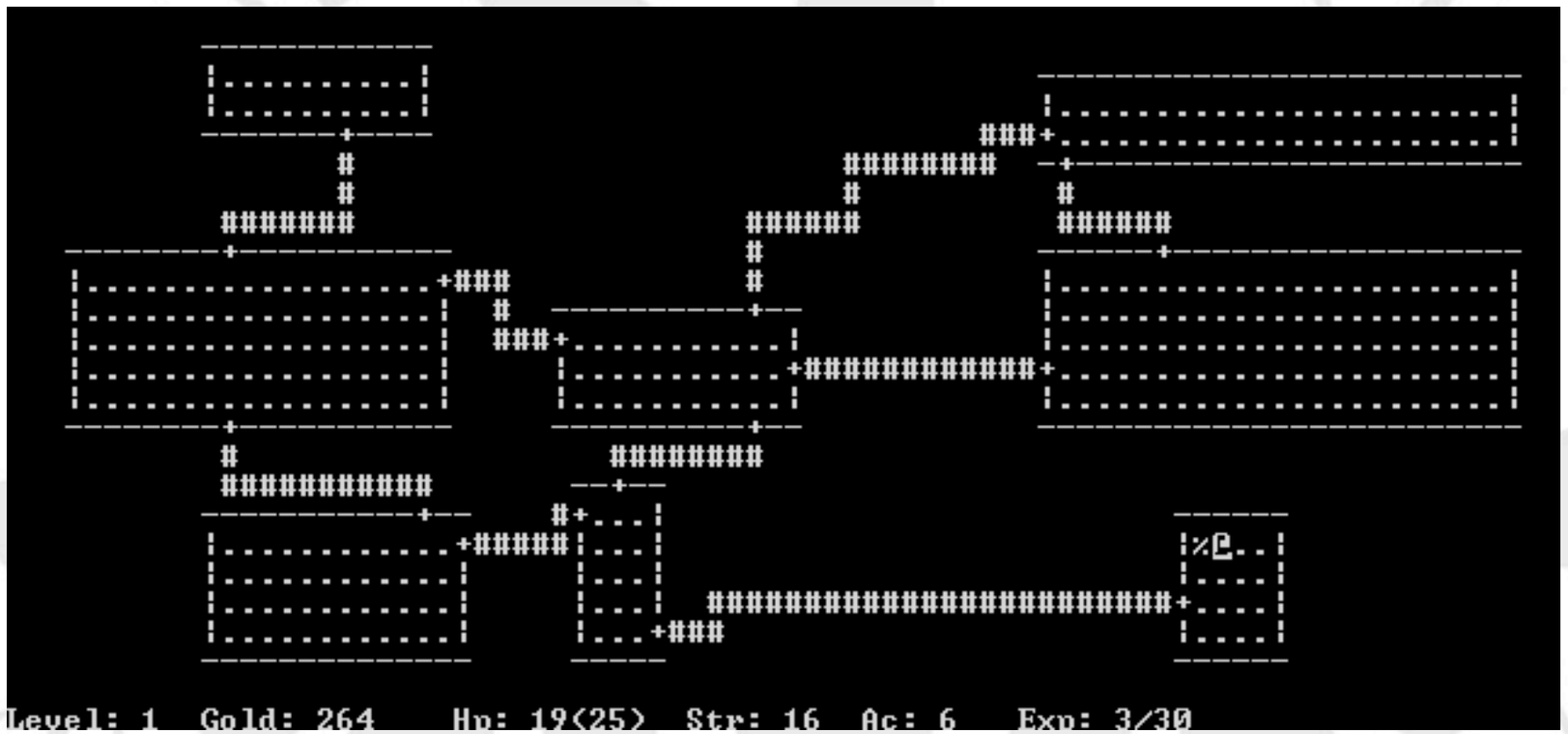
NetHack - 1986



Space Invaders - 1983



Pac-Man - 1980



Rogue - 1980

Rogue

Rogue

- Én helt

Rogue

- Én helt
- Ét monster

Rogue

- Én helt
- Ét monster
- Målet for helten

Rogue

- Én helt
- Ét monster
- Målet for helten
  - Flygt fra monsteret så længe som muligt

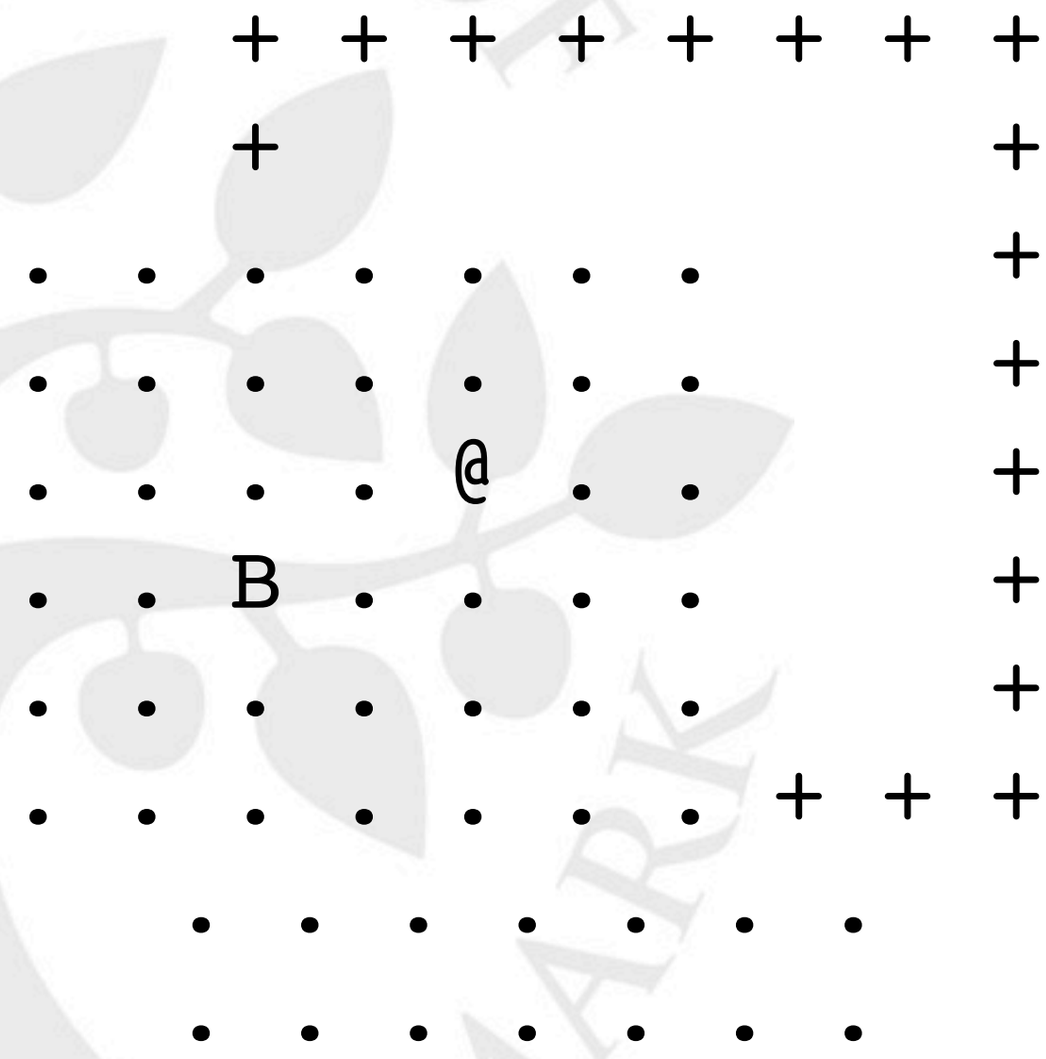
Rogue

- Én helt
- Ét monster
- Målet for helten
  - Flygt fra monsteret så længe som muligt
- Målet for monsteret

Rogue

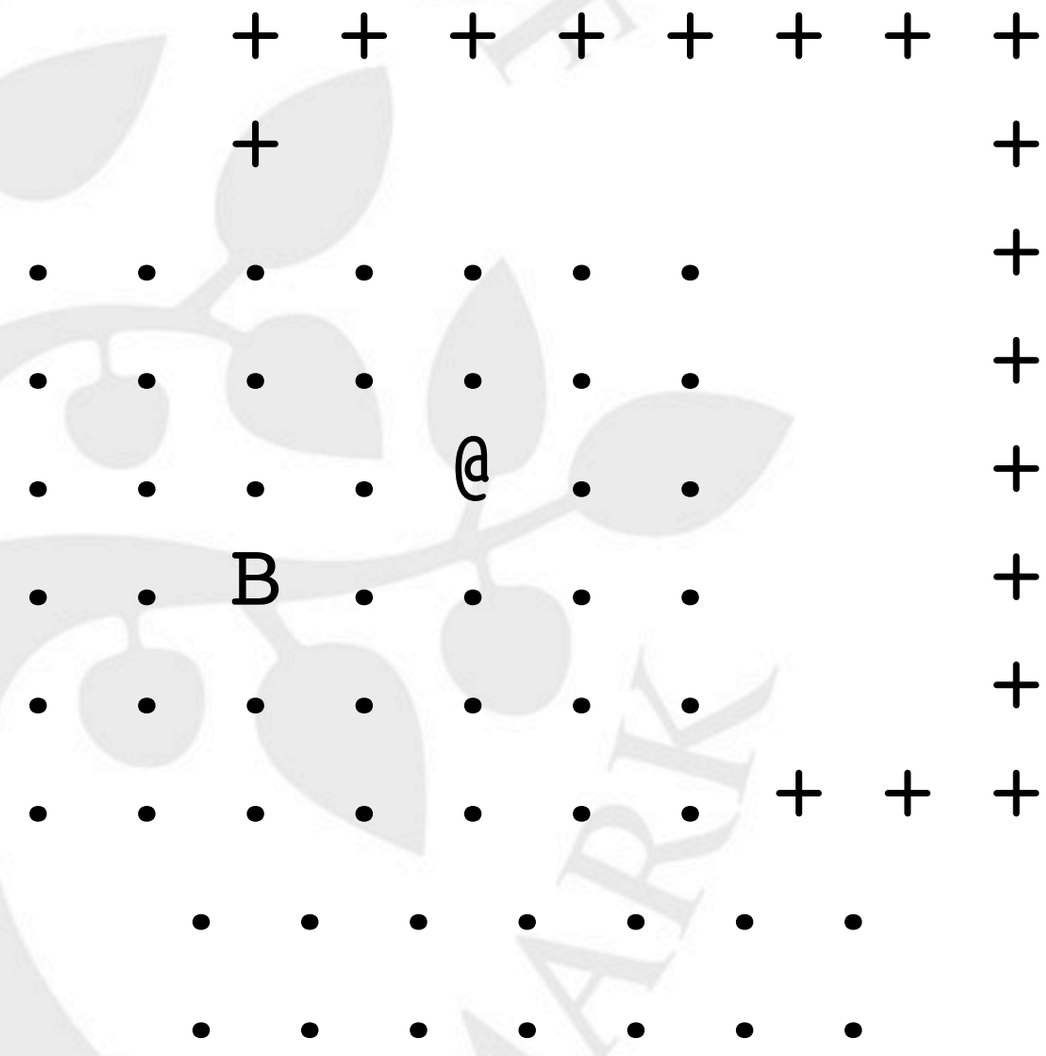
- Én helt
- Ét monster
- Målet for helten
  - Flygt fra monsteret så længe som muligt
- Målet for monsteret
  - Fang og "tilintetgør" helten så hurtigt som muligt

Spillepladen



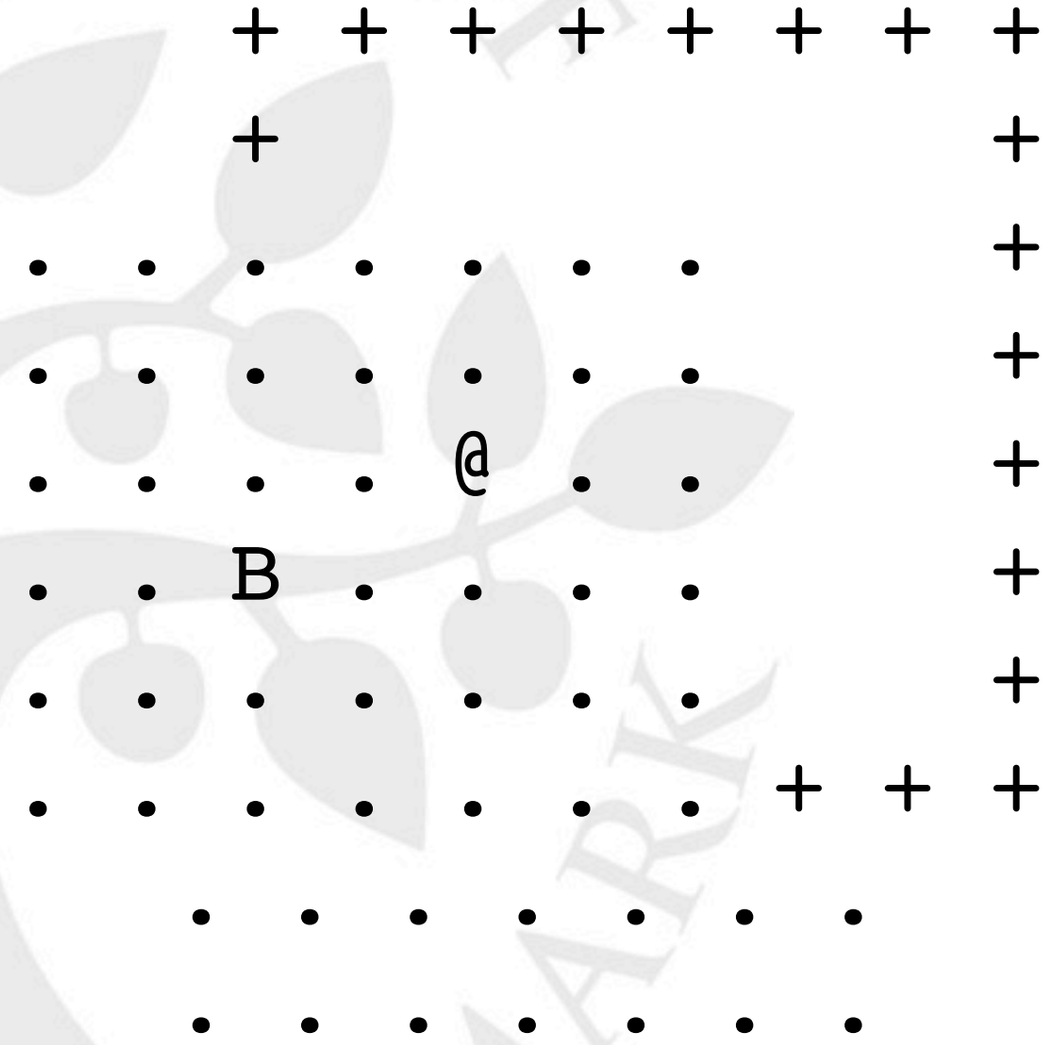
Spillepladen

- N x N gitter



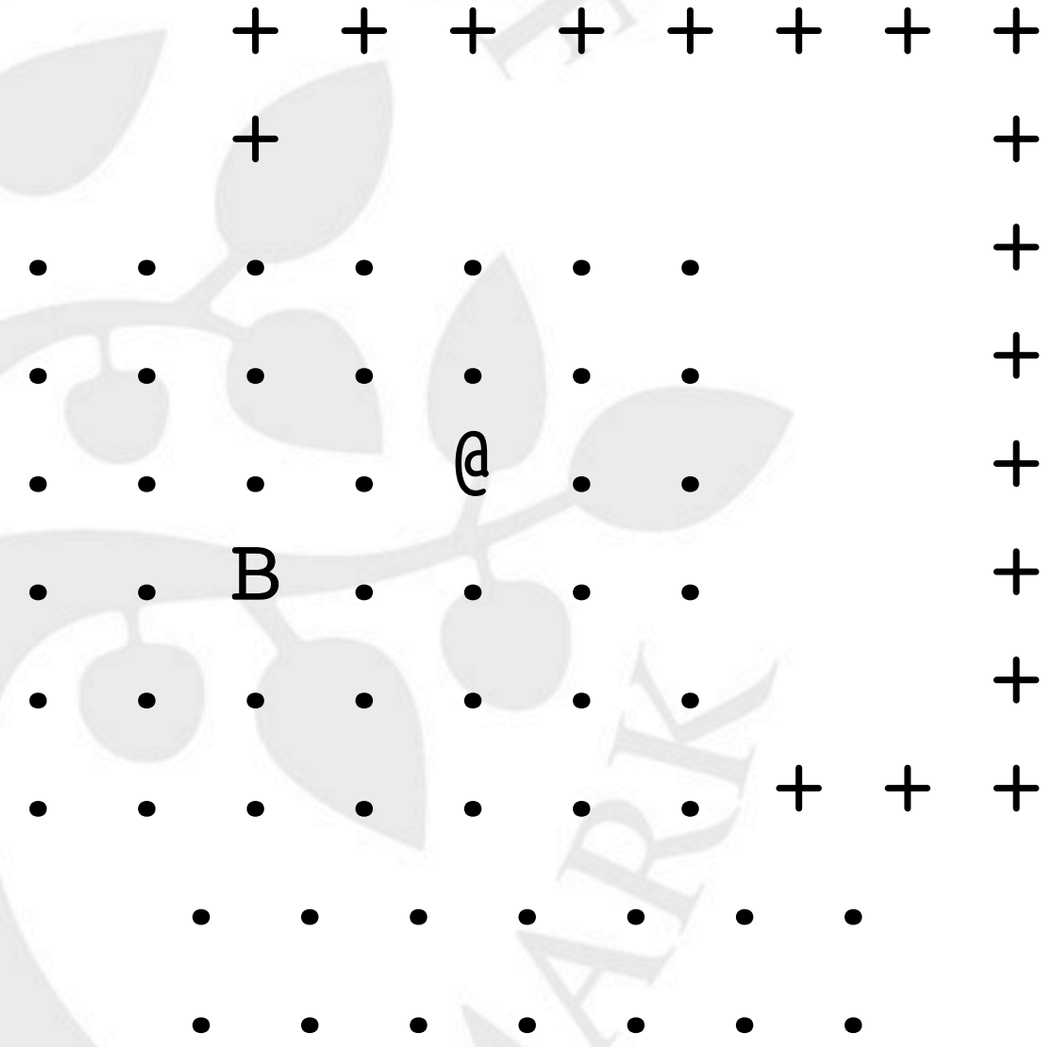
Spillepladen

- N x N gitter
  - N = 10 i dette tilfælde



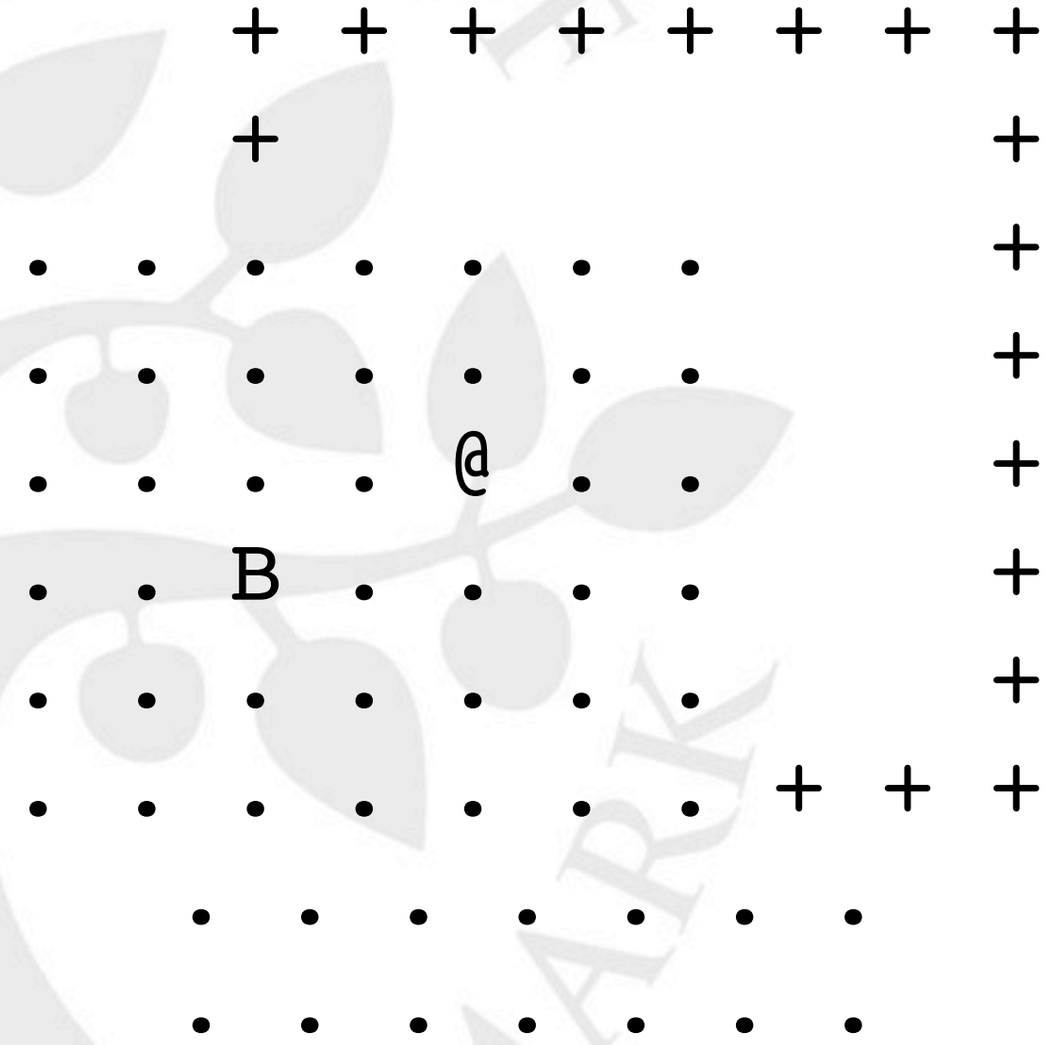
Spillepladen

- N x N gitter
  - N = 10 i dette tilfælde
- .



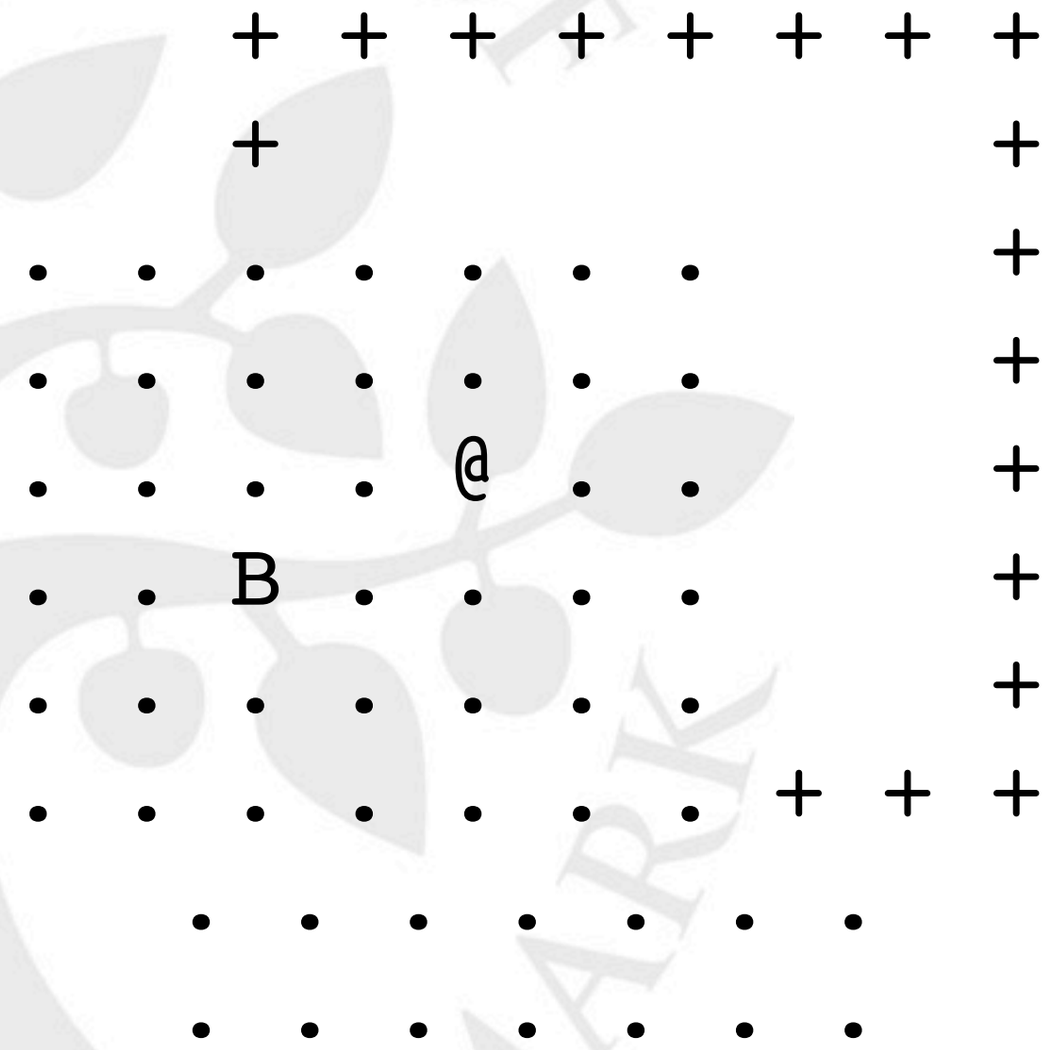
Spillepladen

- N x N gitter
  - N = 10 i dette tilfælde
- .
  - Del af et rum



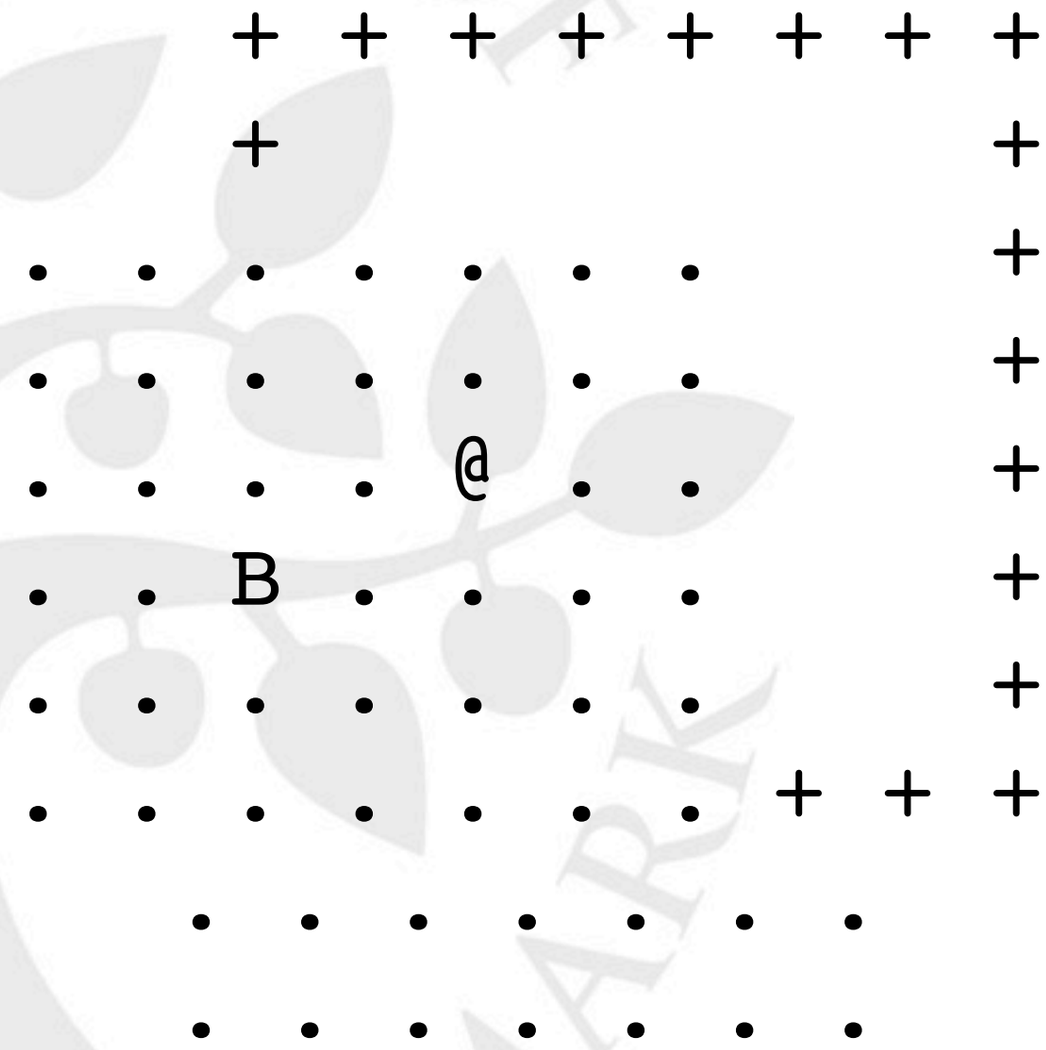
Spillepladen

- N x N gitter
  - N = 10 i dette tilfælde
- .
  - Del af et rum
- +



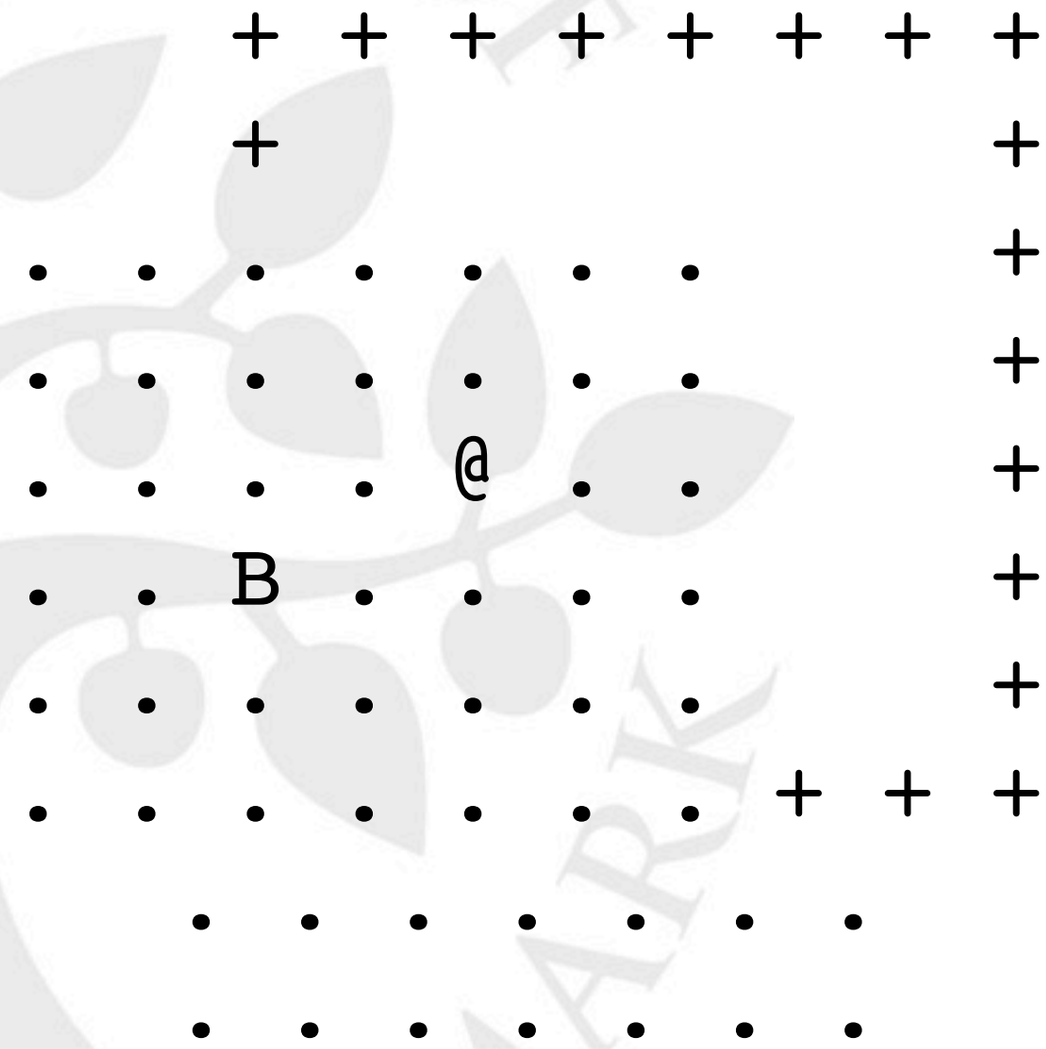
Spillepladen

- N x N gitter
  - N = 10 i dette tilfælde
- .
  - Del af et rum
- +
  - En del af en korridor



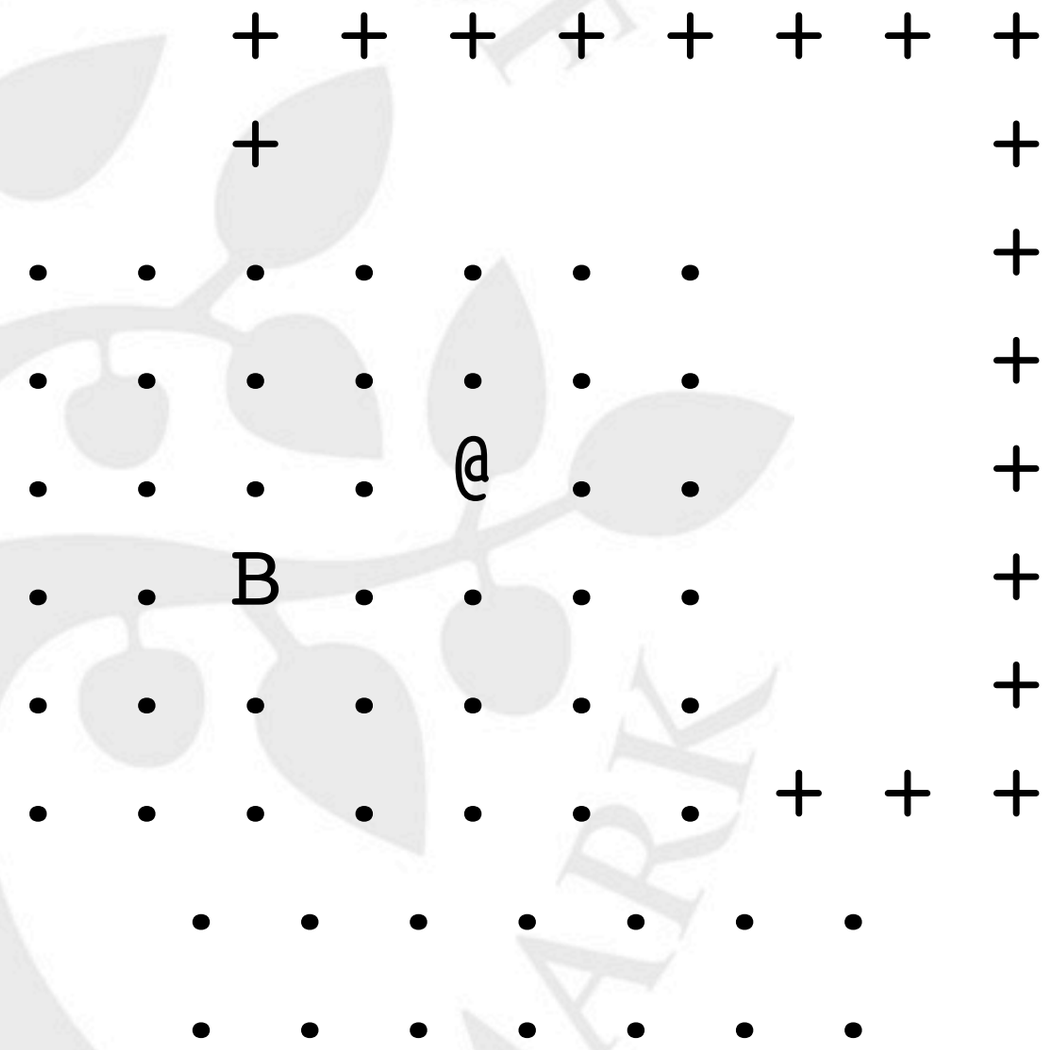
Spillepladen

- N x N gitter
  - N = 10 i dette tilfælde
- .
  - Del af et rum
- +
  - En del af en korridor
- @



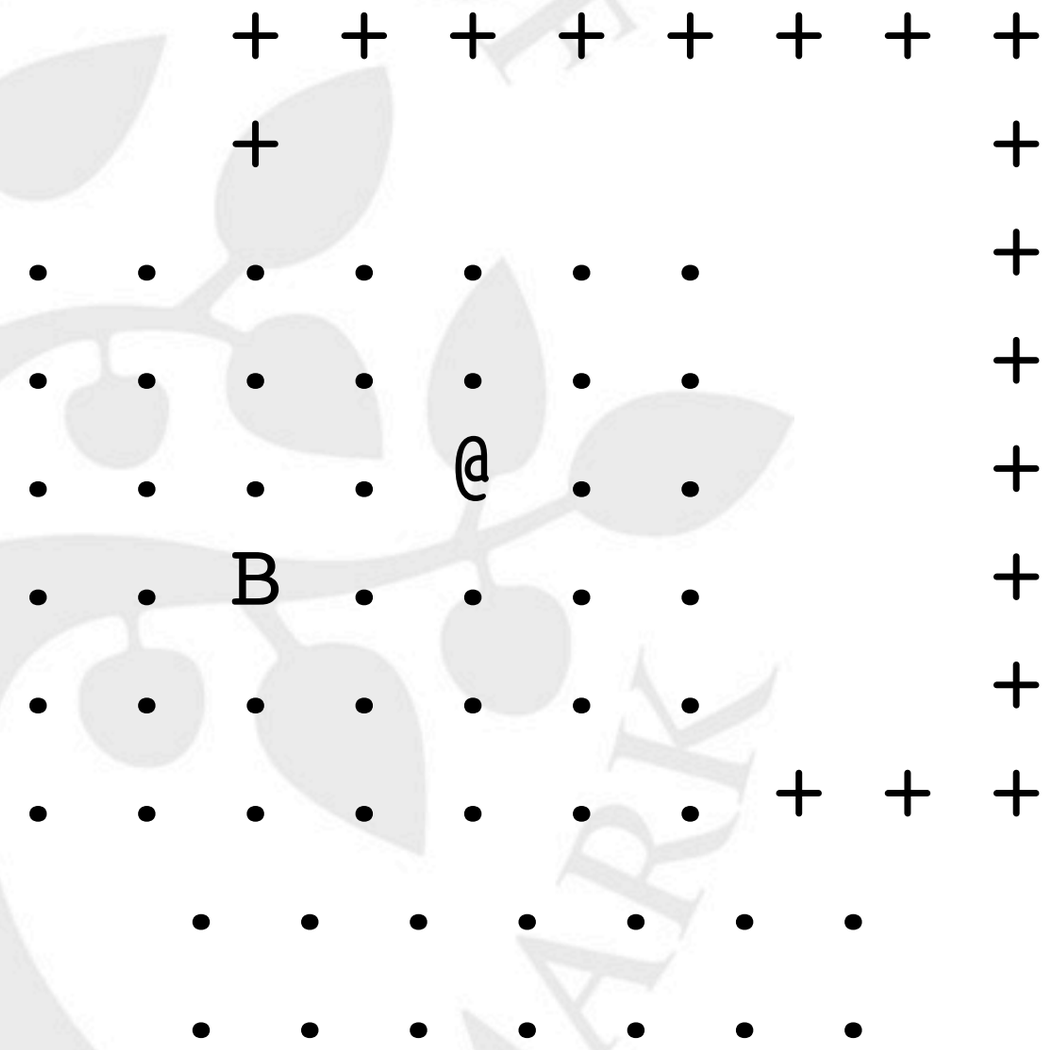
Spillepladen

- N x N gitter
  - N = 10 i dette tilfælde
- .
  - Del af et rum
- +
  - En del af en korridor
- @
  - Helten



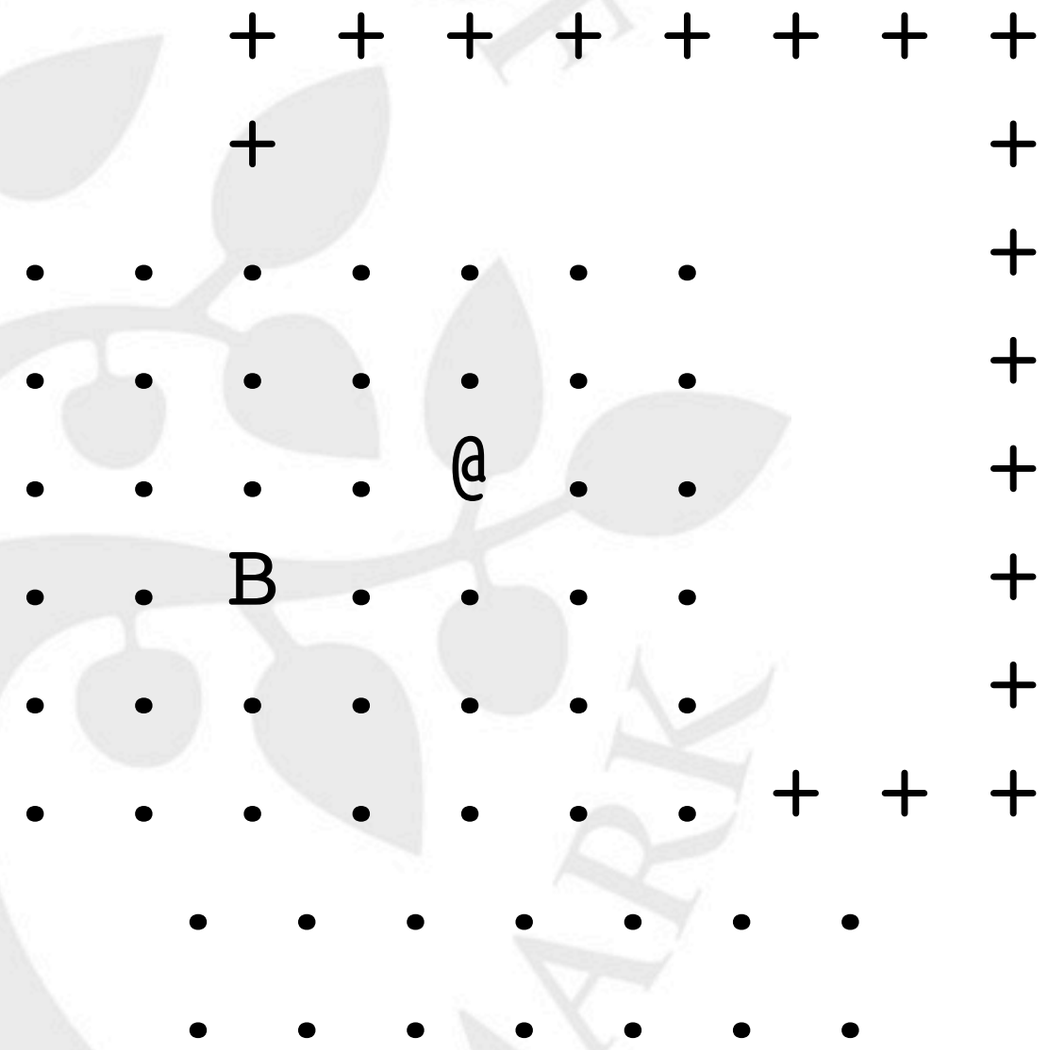
Spillepladen

- N x N gitter
  - N = 10 i dette tilfælde
- .
  - Del af et rum
- +
  - En del af en korridor
- @
  - Helten
- A-Z



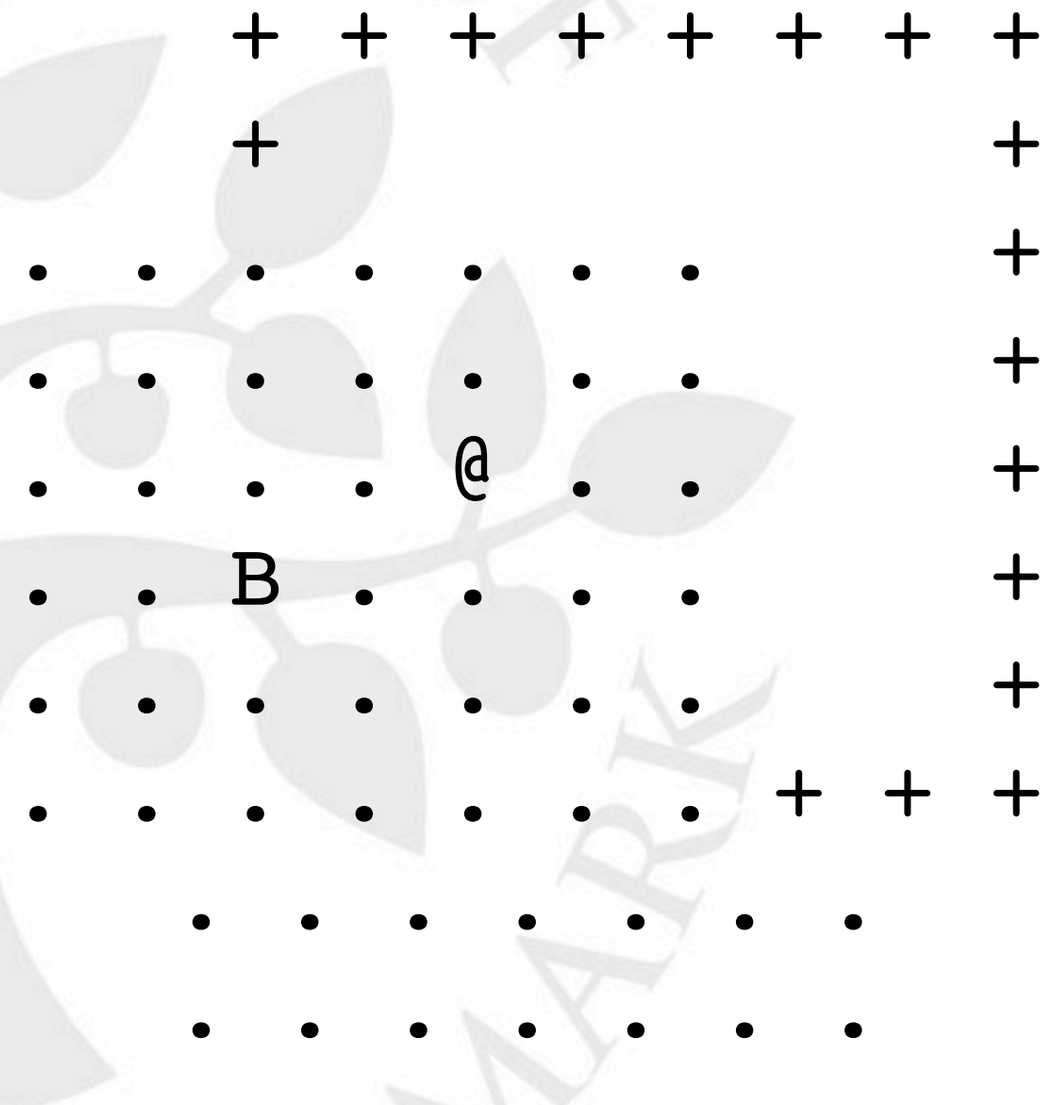
Spillepladen

- N x N gitter
  - N = 10 i dette tilfælde
- .
  - Del af et rum
- +
  - En del af en korridor
- @
  - Helten
- A-Z
  - Monsteret

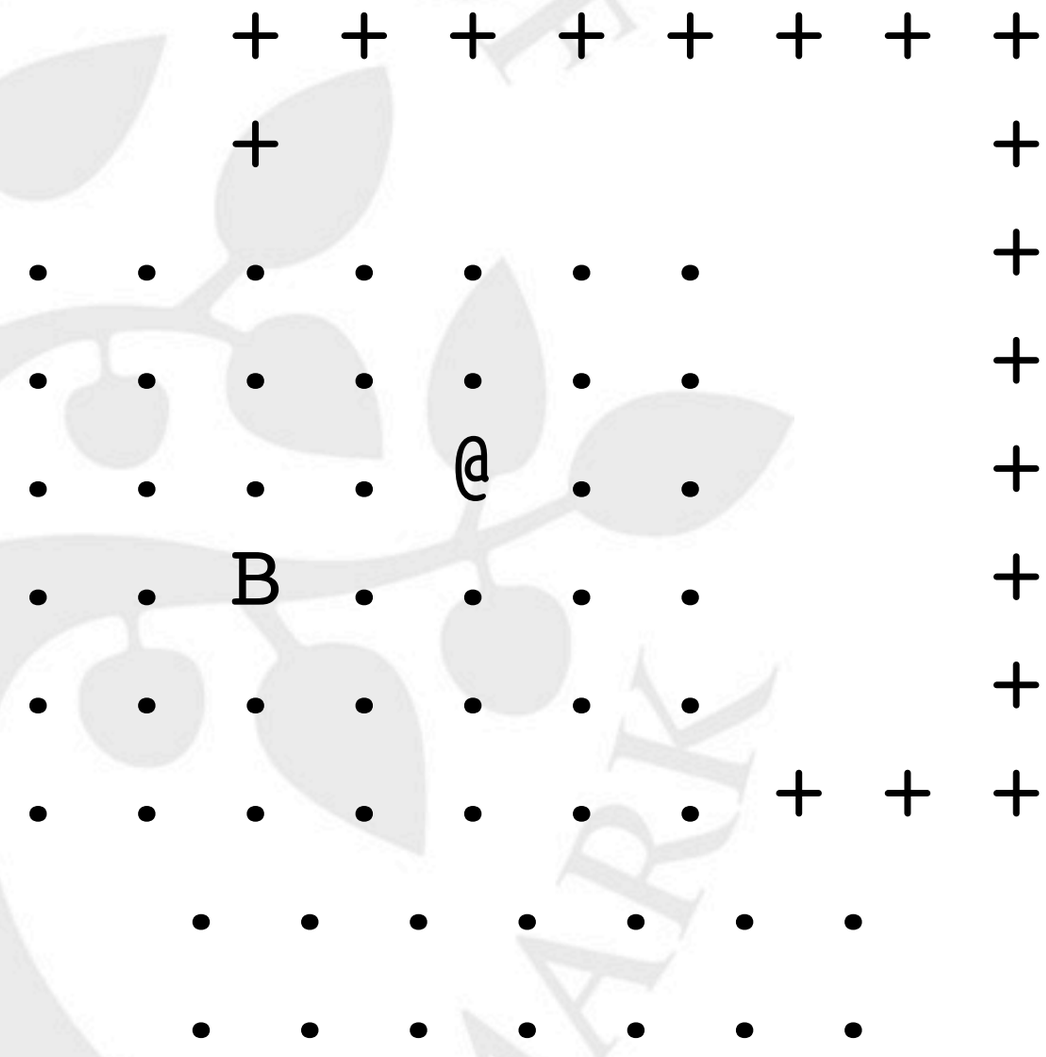


Spillepladen

- N x N gitter
  - N = 10 i dette tilfælde
- .
  - Del af et rum
- +
  - En del af en korridor
- @
  - Helten
- A-Z
  - Monsteret
  - B i dette tilfælde

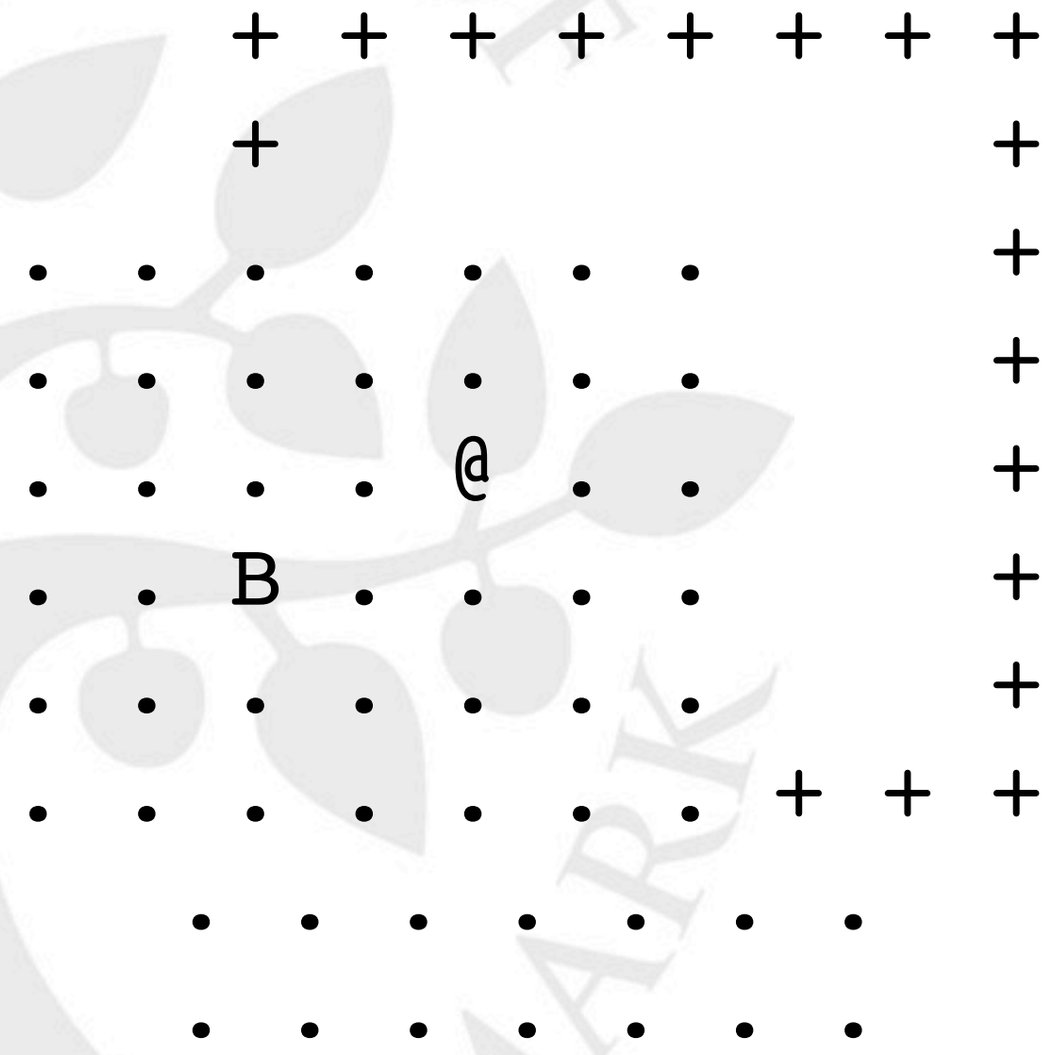


Ryk



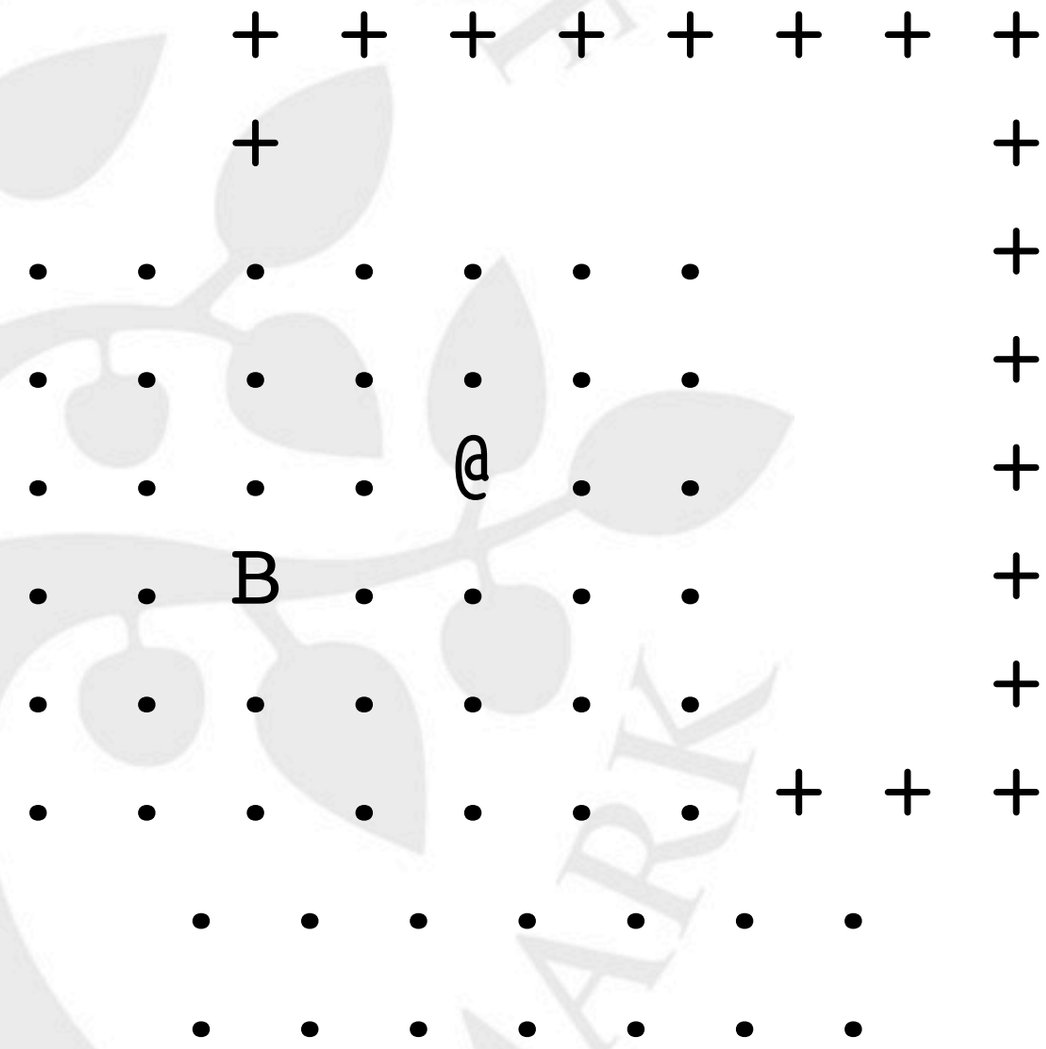
Ryk

- Tur-baseret



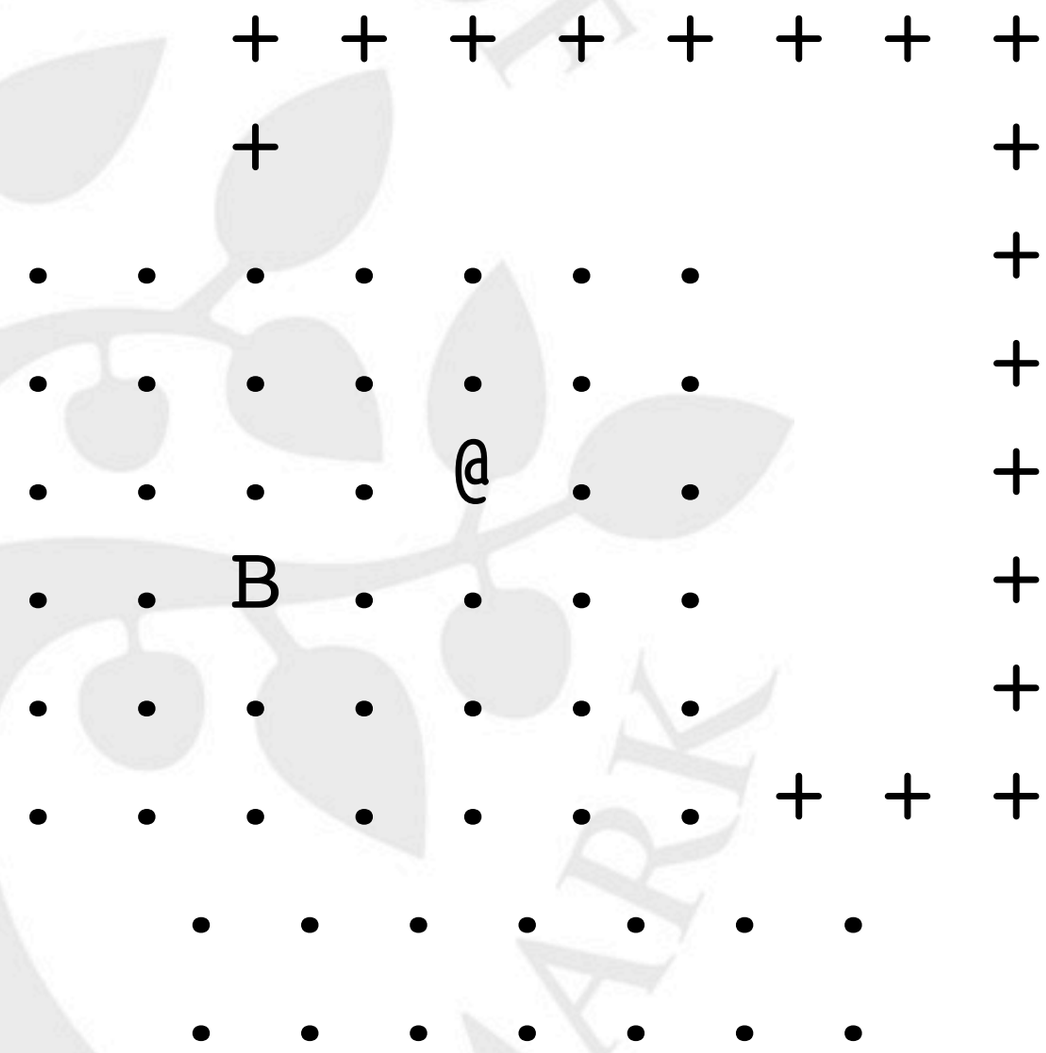
Ryk

- Tur-baseret
  - Et ryk pr. runde



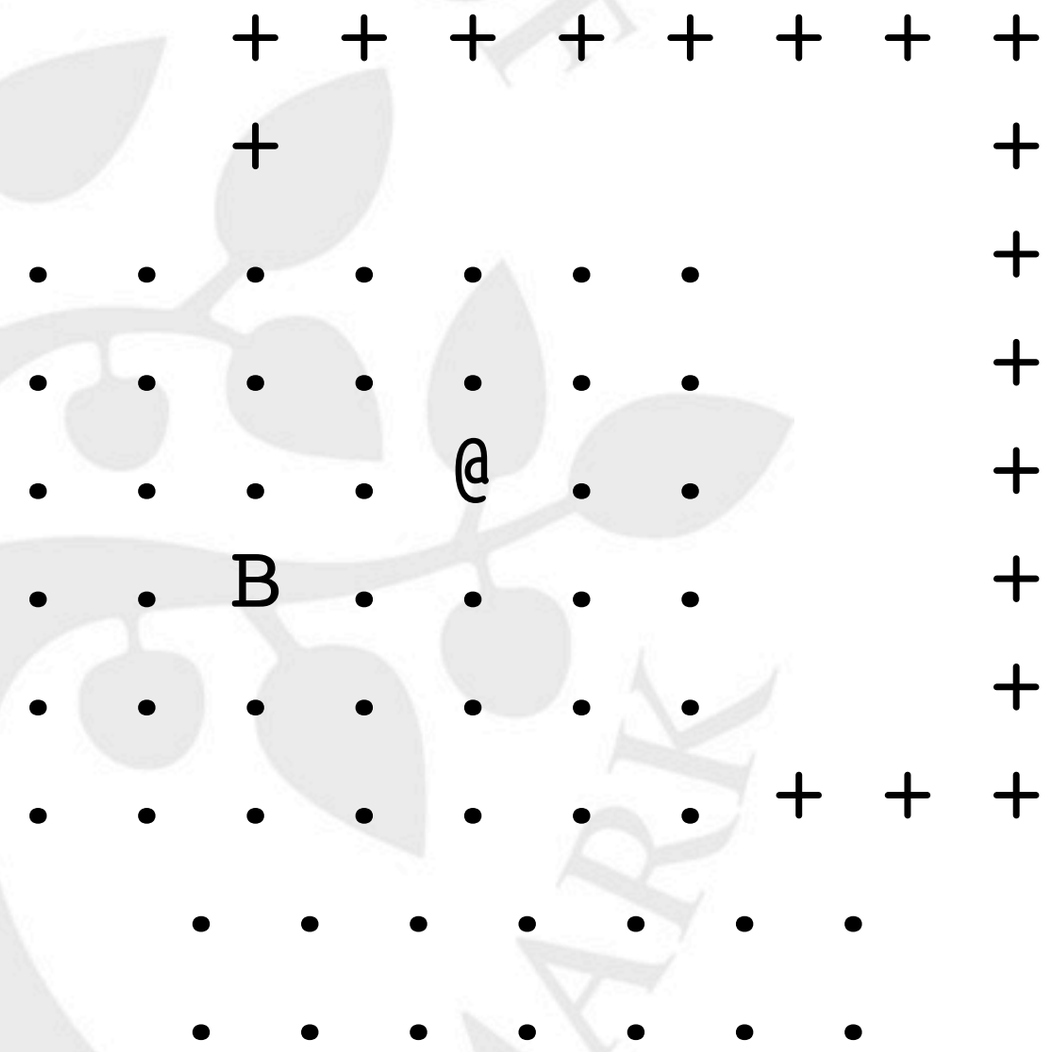
Ryk

- Tur-baseret
  - Et ryk pr. runde
  - Monsteret starter



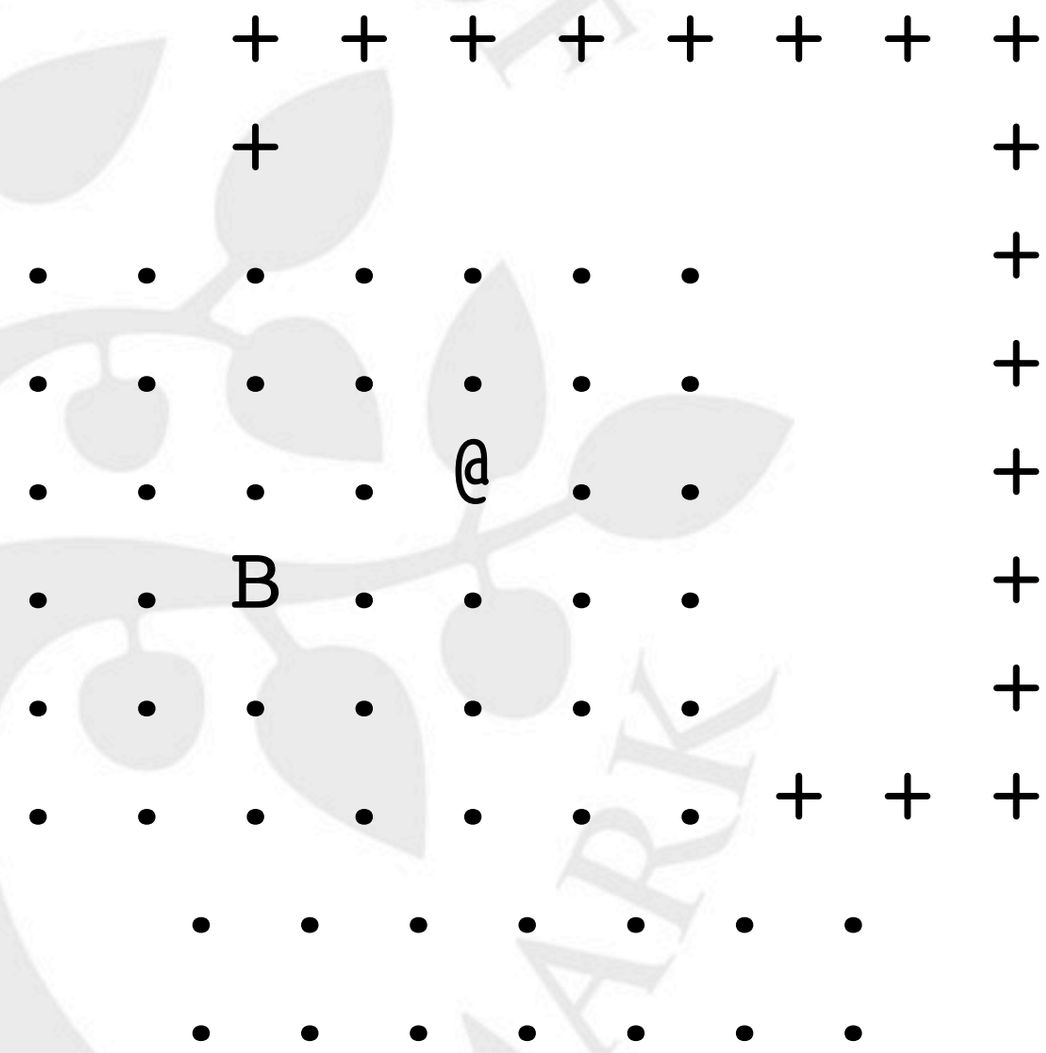
Ryk

- Tur-baseret
  - Et ryk pr. runde
  - Monsteret starter
- På et rum-felt (.)



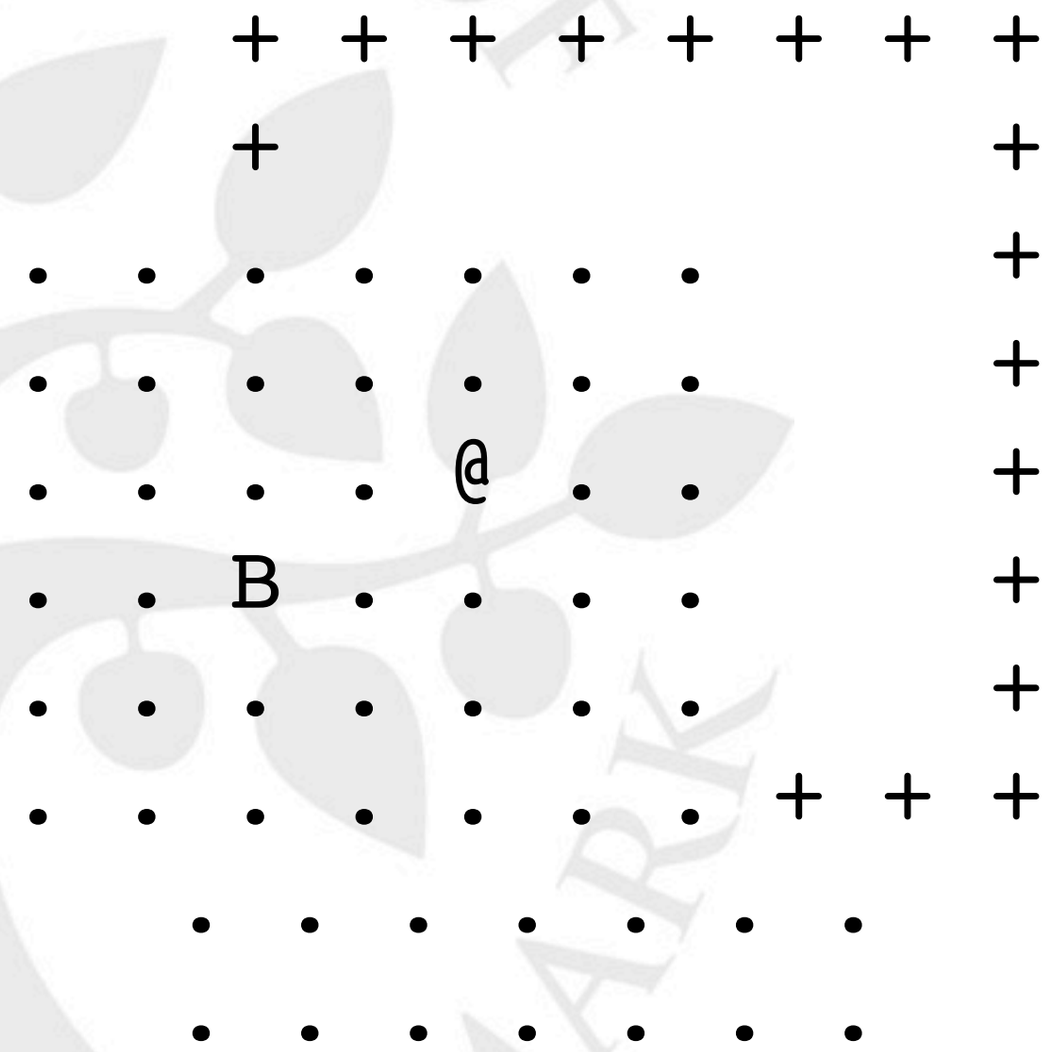
Ryk

- Tur-baseret
  - Et ryk pr. runde
  - Monsteret starter
- På et rum-felt (.)
  - N,S,Ø,V,NV,NØ,SV,SØ



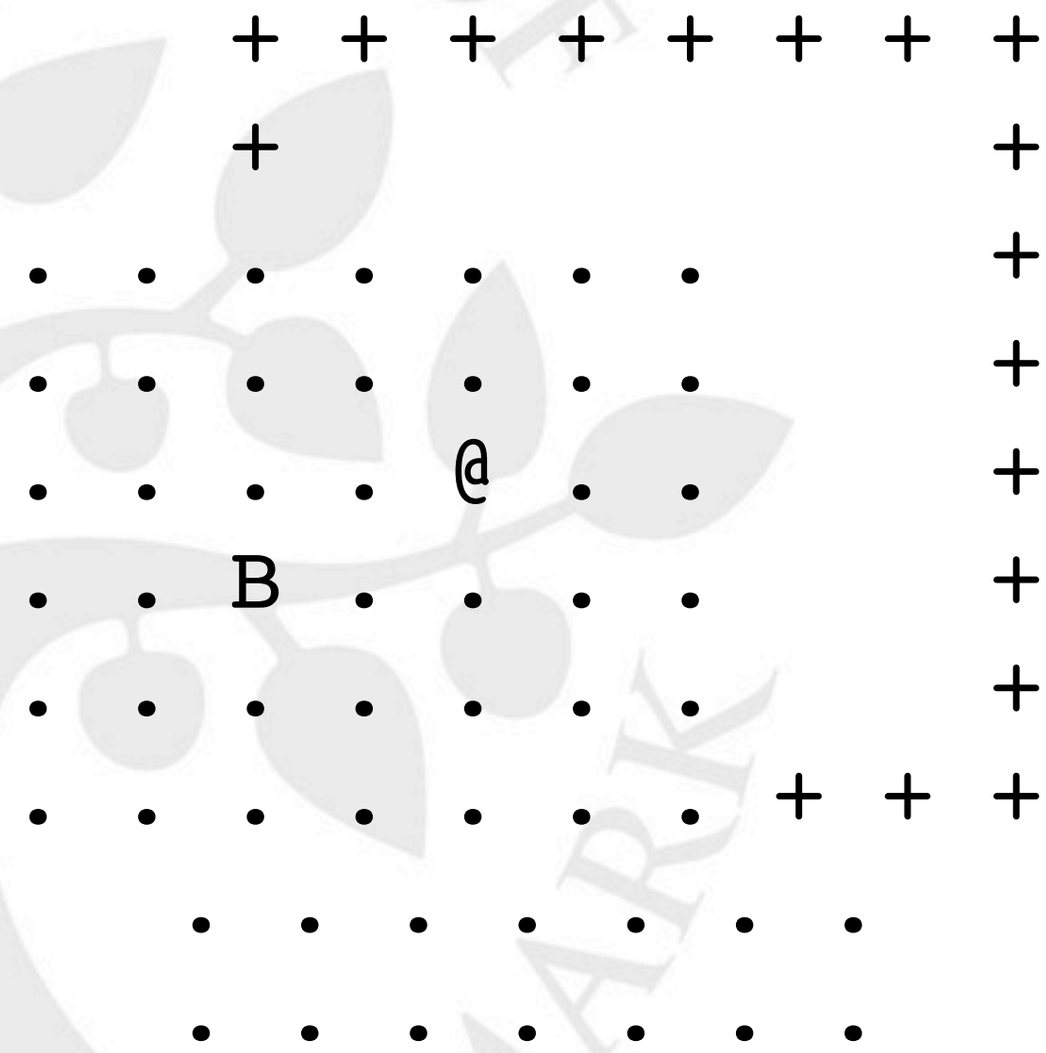
Ryk

- Tur-baseret
  - Et ryk pr. runde
  - Monsteret starter
- På et rum-felt (.)
  - N,S,Ø,V,NV,NØ,SV,SØ
- På et korridor-felt (+)



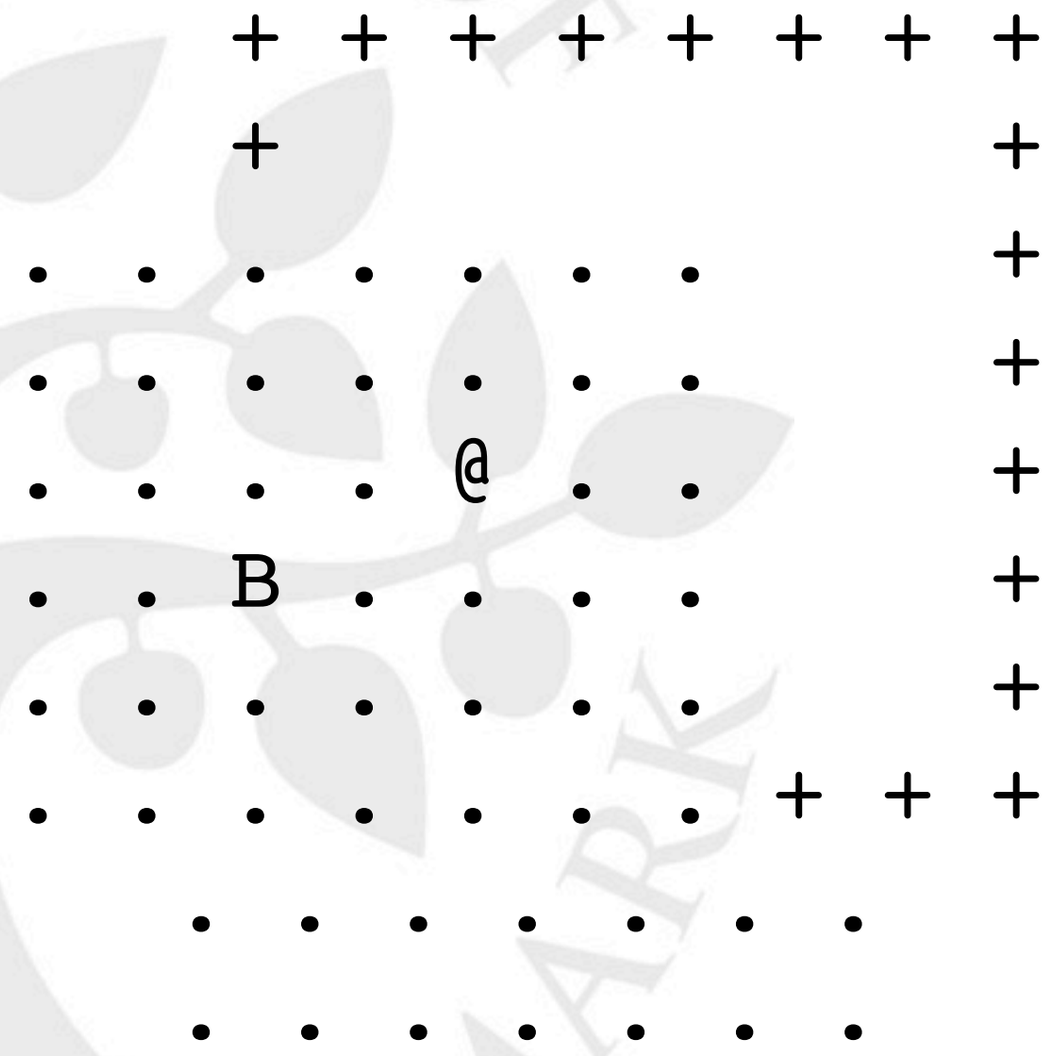
Ryk

- Tur-baseret
  - Et ryk pr. runde
  - Monsteret starter
- På et rum-felt (.)
  - N,S,Ø,V,NV,NØ,SV,SØ
- På et korridor-felt (+)
  - N,S,Ø,V

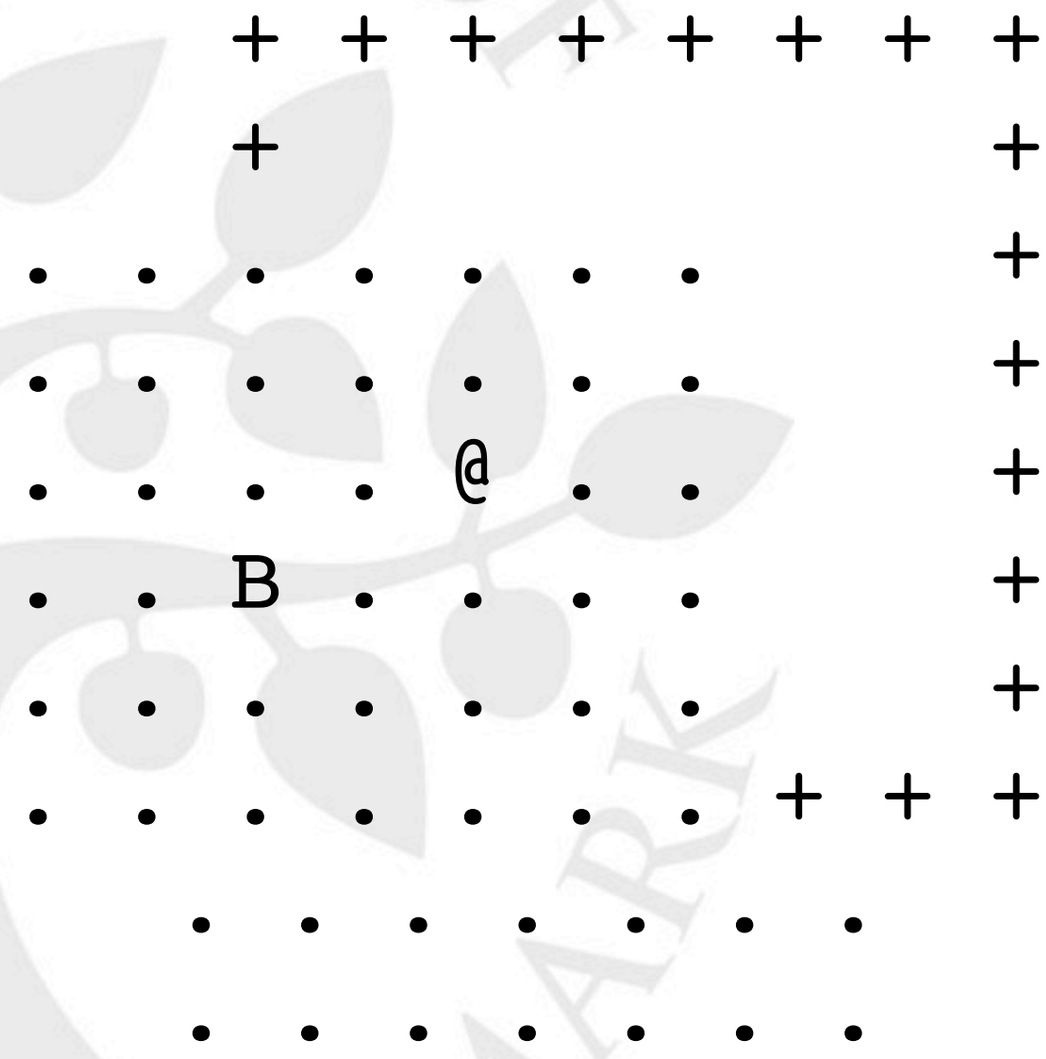


Ryk

- Tur-baseret
  - Et ryk pr. runde
  - Monsteret starter
- På et rum-felt (.)
  - N,S,Ø,V,NV,NØ,SV,SØ
- På et korridor-felt (+)
  - N,S,Ø,V
- Kan ikke gå igennem vægge

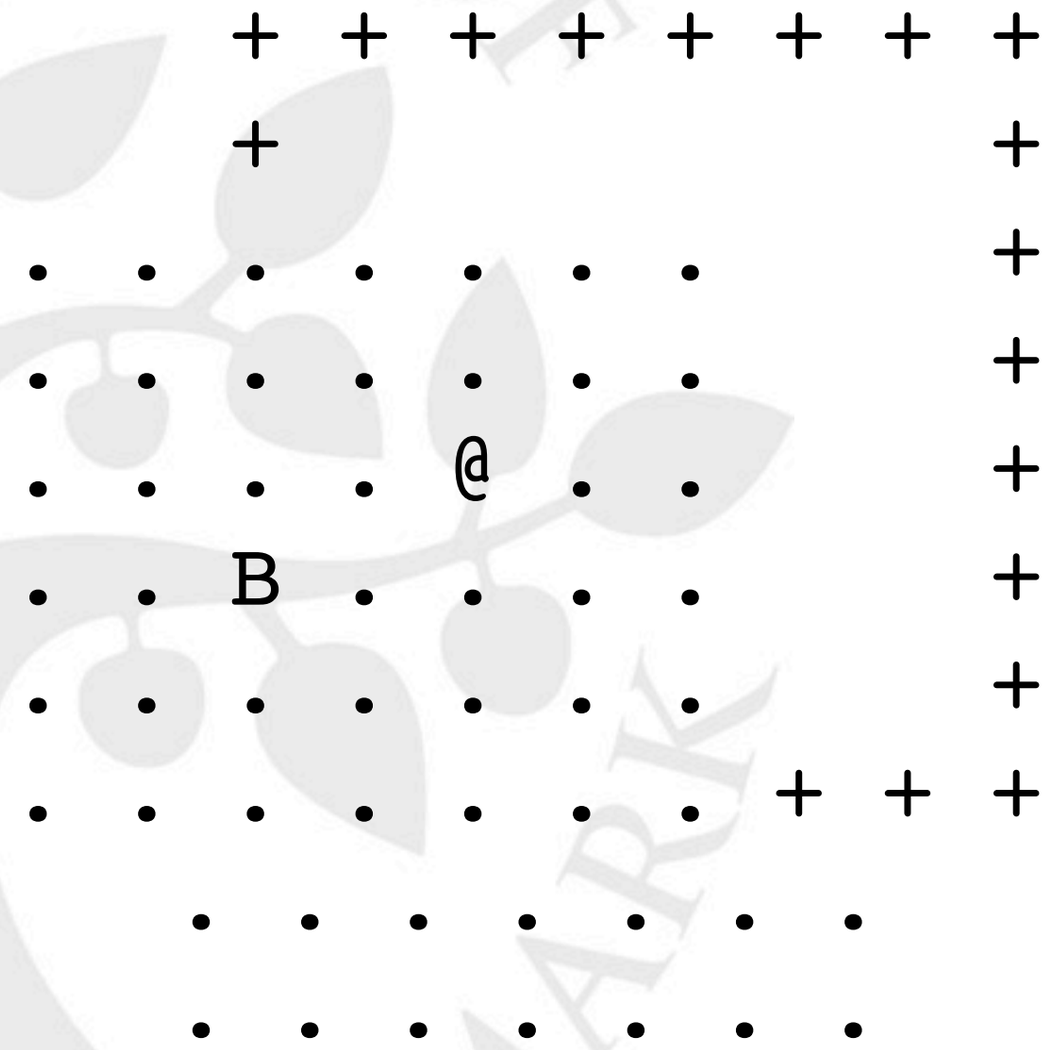


Den underliggende graf



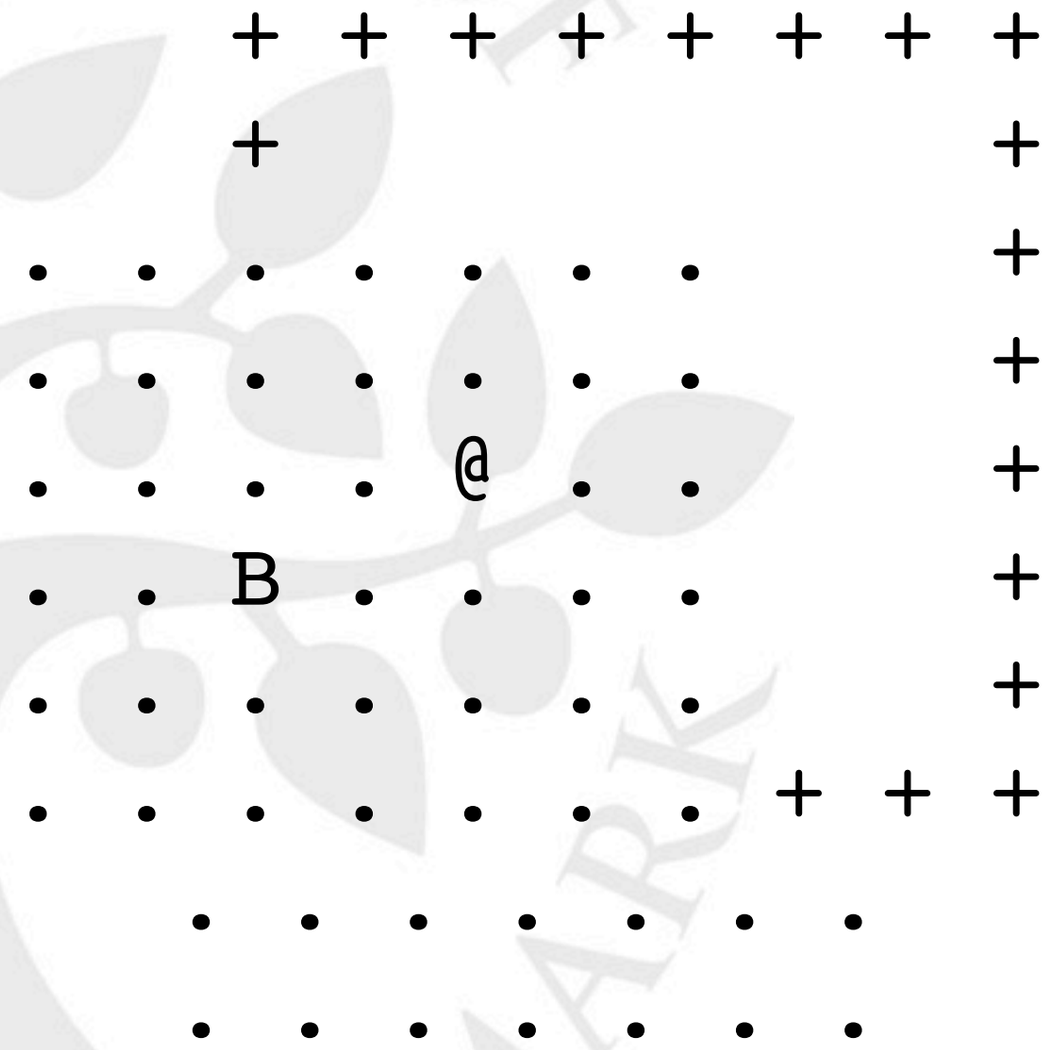
Den underliggende graf

- Et felt på spillepladen



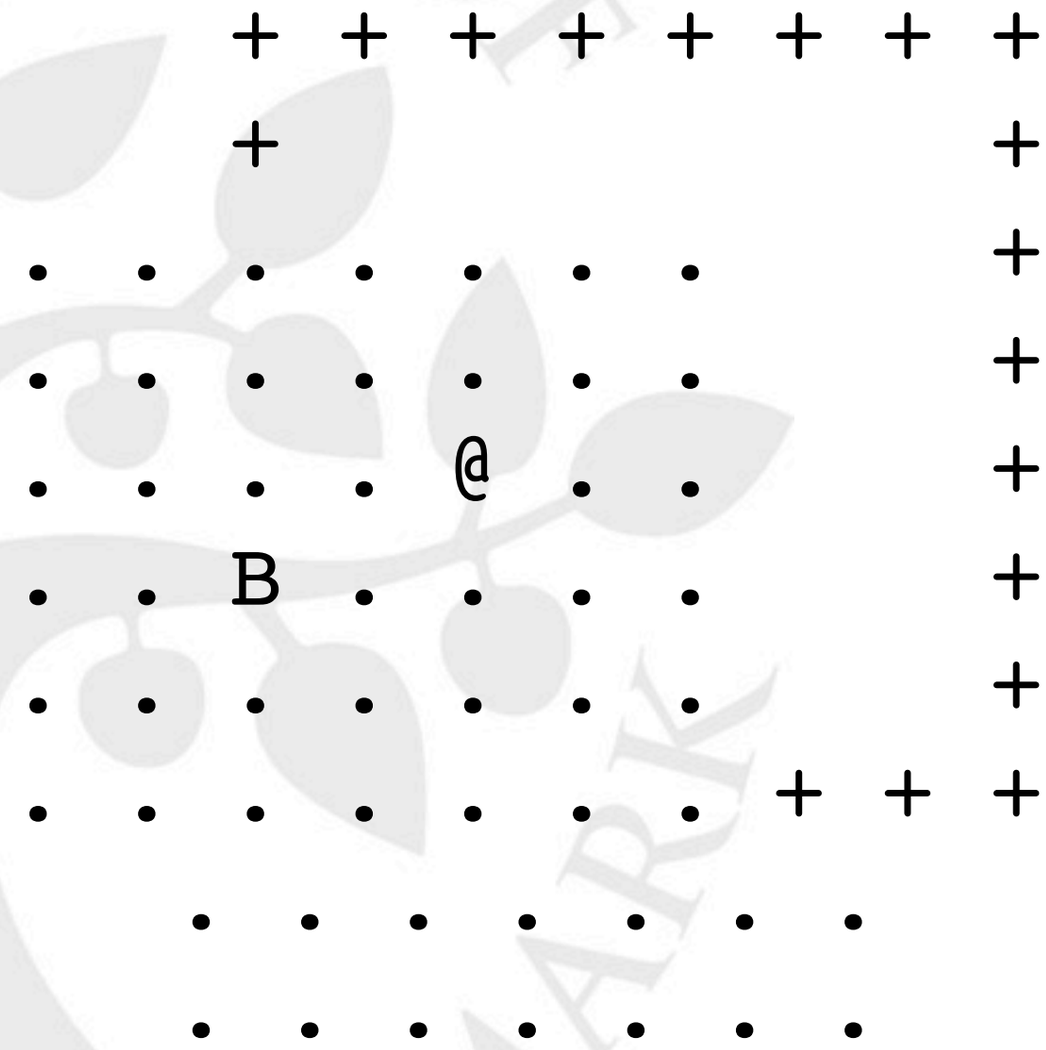
Den underliggende graf

- Et felt på spillepladen
  - En knude i grafen



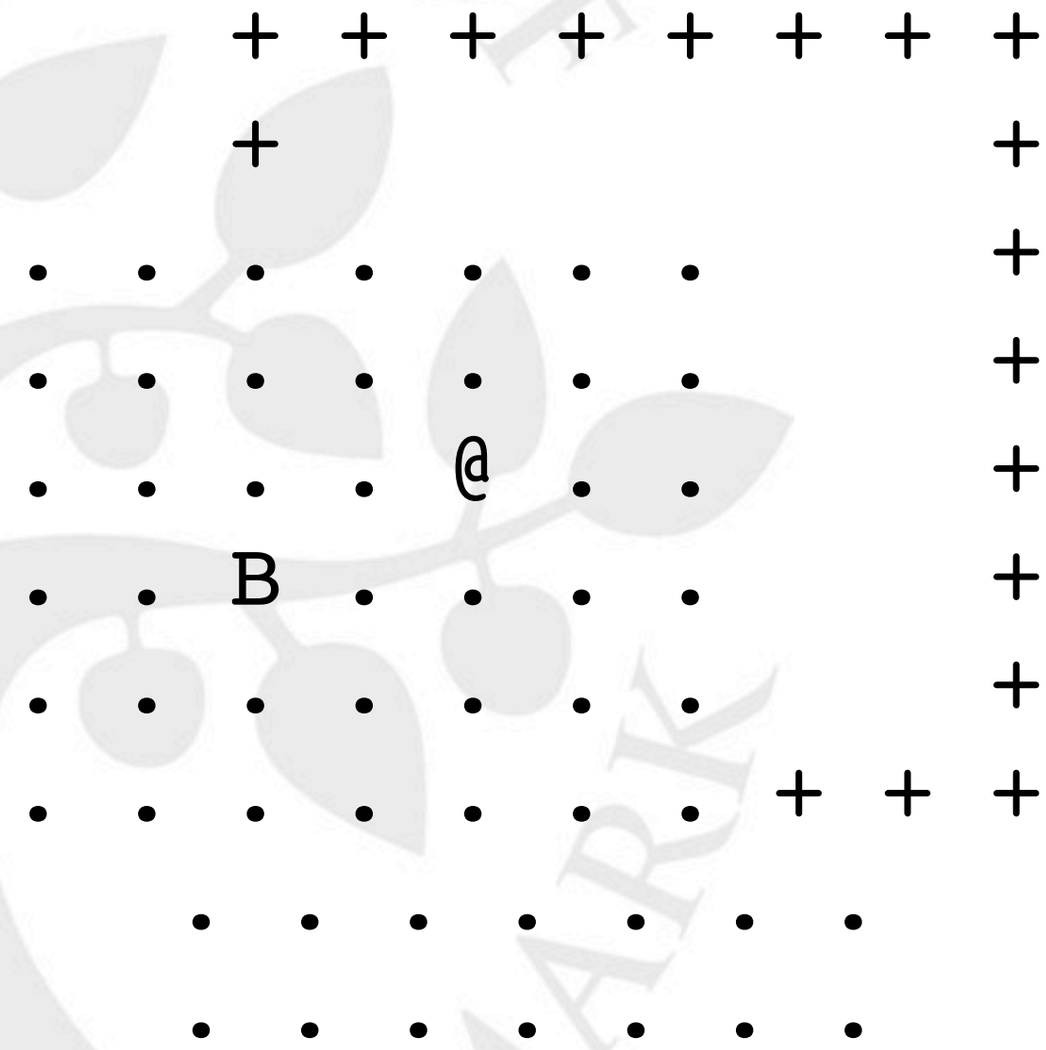
Den underliggende graf

- Et felt på spillepladen
  - En knude i grafen
- En kant mellem v og w



Den underliggende graf

- Et felt på spillepladen
  - En knude i grafen
- En kant mellem v og w
  - Lovligt at rykke fra feltet v repræsenterer til feltet w repræsenterer



Monsterets strategi

Monsterets strategi

- Husk

Monsterets strategi

- Husk
  - Fang helten så hurtigt (i så få ryk) som muligt

Monsterets strategi

- Husk
  - Fang helten så hurtigt (i så få ryk) som muligt
- Kender sin egen position

Monsterets strategi

- Husk
  - Fang helten så hurtigt (i så få ryk) som muligt
- Kender sin egen position
- Kender heltens position

Monsterets strategi

- Husk
  - Fang helten så hurtigt (i så få ryk) som muligt
- Kender sin egen position
- Kender heltens position
- "Gå hen imod helten"

Monsterets strategi

- Husk
  - Fang helten så hurtigt (i så få ryk) som muligt
- Kender sin egen position
- Kender heltens position
- "Gå hen imod helten"
  - Ta et skridt på en korteste sti fra monsteret til helten

Monsterets strategi

- Husk
  - Fang helten så hurtigt (i så få ryk) som muligt
- Kender sin egen position
- Kender heltens position
- "Gå hen imod helten"
  - Ta et skridt på en korteste sti fra monsteret til helten
  - Bredde-først-søgning fra monsteret indtil helten findes

Monsterets strategi

- Husk
  - Fang helten så hurtigt (i så få ryk) som muligt
- Kender sin egen position
- Kender heltens position
- "Gå hen imod helten"
  - Ta et skridt på en korteste sti fra monsteret til helten
  - Bredde-først-søgning fra monsteret indtil helten findes
  - Ta et skridt af den fundne sti

Monsterets strategi

- Husk
  - Fang helten så hurtigt (i så få ryk) som muligt
- Kender sin egen position
- Kender heltens position
- "Gå hen imod helten"
  - Ta et skridt på en korteste sti fra monsteret til helten
    - Bredde-først-søgning fra monsteret indtil helten findes
  - Ta et skridt af den fundne sti
- Din opgave

Monsterets strategi

- Husk
  - Fang helten så hurtigt (i så få ryk) som muligt
- Kender sin egen position
- Kender heltens position
- "Gå hen imod helten"
  - Ta et skridt på en korteste sti fra monsteret til helten
    - Bredde-først-søgning fra monsteret indtil helten findes
    - Ta et skridt af den fundne sti
- Din opgave
  - Implementer ovenstående strategi

University of Southern Denmark • Syddansk Universitet

Heltens strategi

Heltens strategi

- Husk

Heltens strategi

- Husk
  - Flygt fra monsteret så lang tid som muligt

Heltens strategi

- Husk
  - Flygt fra monsteret så lang tid som muligt
- Kender sin egen position

Heltens strategi

- Husk
  - Flygt fra monsteret så lang tid som muligt
- Kender sin egen position
- Kender monsterets position

Heltens strategi

- Husk
  - Flygt fra monsteret så lang tid som muligt
- Kender sin egen position
- Kender monsterets position
- "Flygt væk fra monsteret"

Heltens strategi

- Husk
  - Flygt fra monsteret så lang tid som muligt
- Kender sin egen position
- Kender monsterets position
- "Flygt væk fra monsteret"
  - Kig på omkringliggende felter

Heltens strategi

- Husk
  - Flygt fra monsteret så lang tid som muligt
- Kender sin egen position
- Kender monsterets position
- "Flygt væk fra monsteret"
  - Kig på omkringliggende felter
  - Kig kun på de felter det er lovligt at flytte til

Heltens strategi

- Husk
  - Flygt fra monsteret så lang tid som muligt
- Kender sin egen position
- Kender monsterets position
- "Flygt væk fra monsteret"
  - Kig på omkringliggende felter
  - Kig kun på de felter det er lovligt at flytte til
  - Beregn afstanden fra monsteret hen til hvert felt

Heltens strategi

- Husk
  - Flygt fra monsteret så lang tid som muligt
- Kender sin egen position
- Kender monsterets position
- "Flygt væk fra monsteret"
  - Kig på omkringliggende felter
  - Kig kun på de felter det er lovligt at flytte til
  - Beregn afstanden fra monsteret hen til hvert felt
  - Flyt til det felt der er længst væk fra monsteret

Heltens strategi

- Husk
  - Flygt fra monsteret så lang tid som muligt
- Kender sin egen position
- Kender monsterets position
- "Flygt væk fra monsteret"
  - Kig på omkringliggende felter
  - Kig kun på de felter det er lovligt at flytte til
  - Beregn afstanden fra monsteret hen til hvert felt
  - Flyt til det felt der er længst væk fra monsteret
- Din opgave

Heltens strategi

- Husk
  - Flygt fra monsteret så lang tid som muligt
- Kender sin egen position
- Kender monsterets position
- "Flygt væk fra monsteret"
  - Kig på omkringliggende felter
  - Kig kun på de felter det er lovligt at flytte til
  - Beregn afstanden fra monsteret hen til hvert felt
  - Flyt til det felt der er længst væk fra monsteret
- Din opgave
  - Implementer ovenstående strategi

Ikke optimale

Ikke optimale

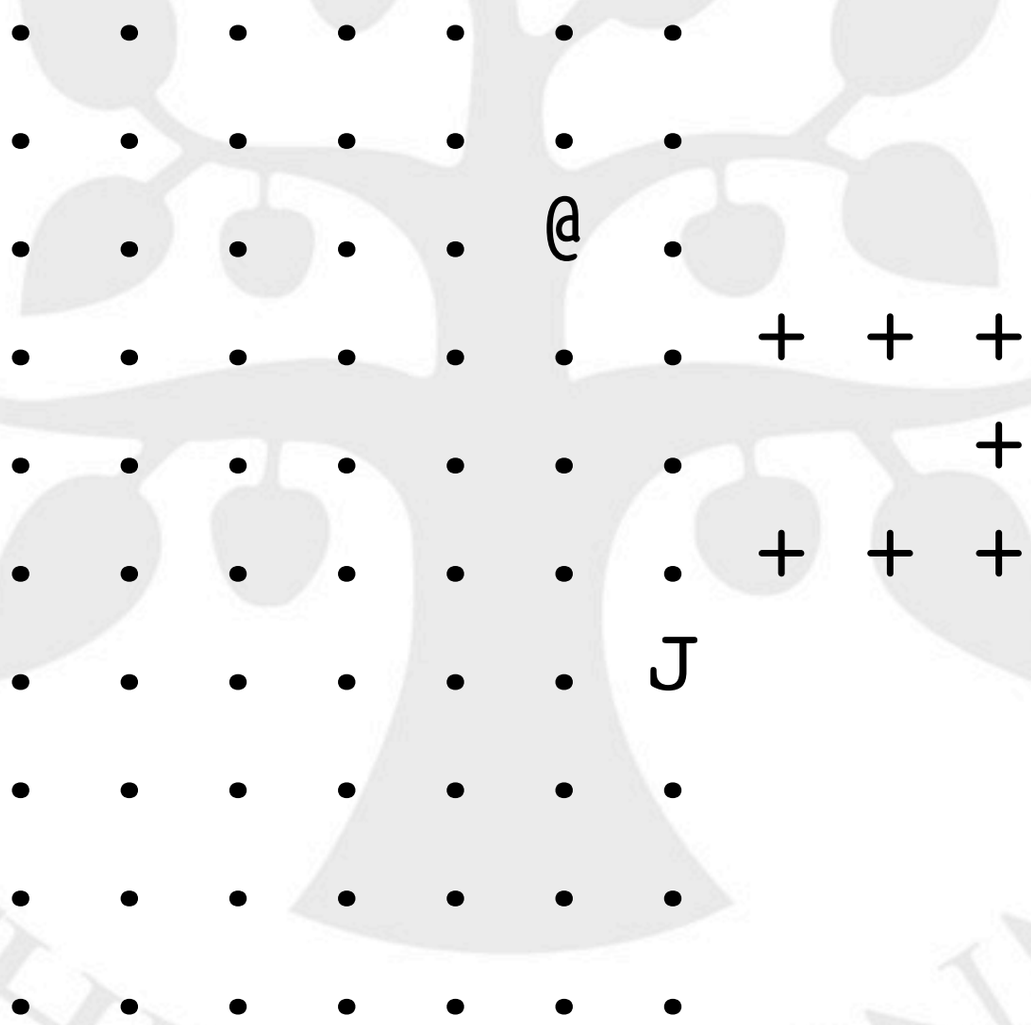
- Begge strategier er ikke optimale

Ikke optimale

- Begge strategier er ikke optimale
- Eksempel for helten

Ikke optimale

- Begge strategier er ikke optimale
- Eksempel for helten



Ikke optimale

Ikke optimale

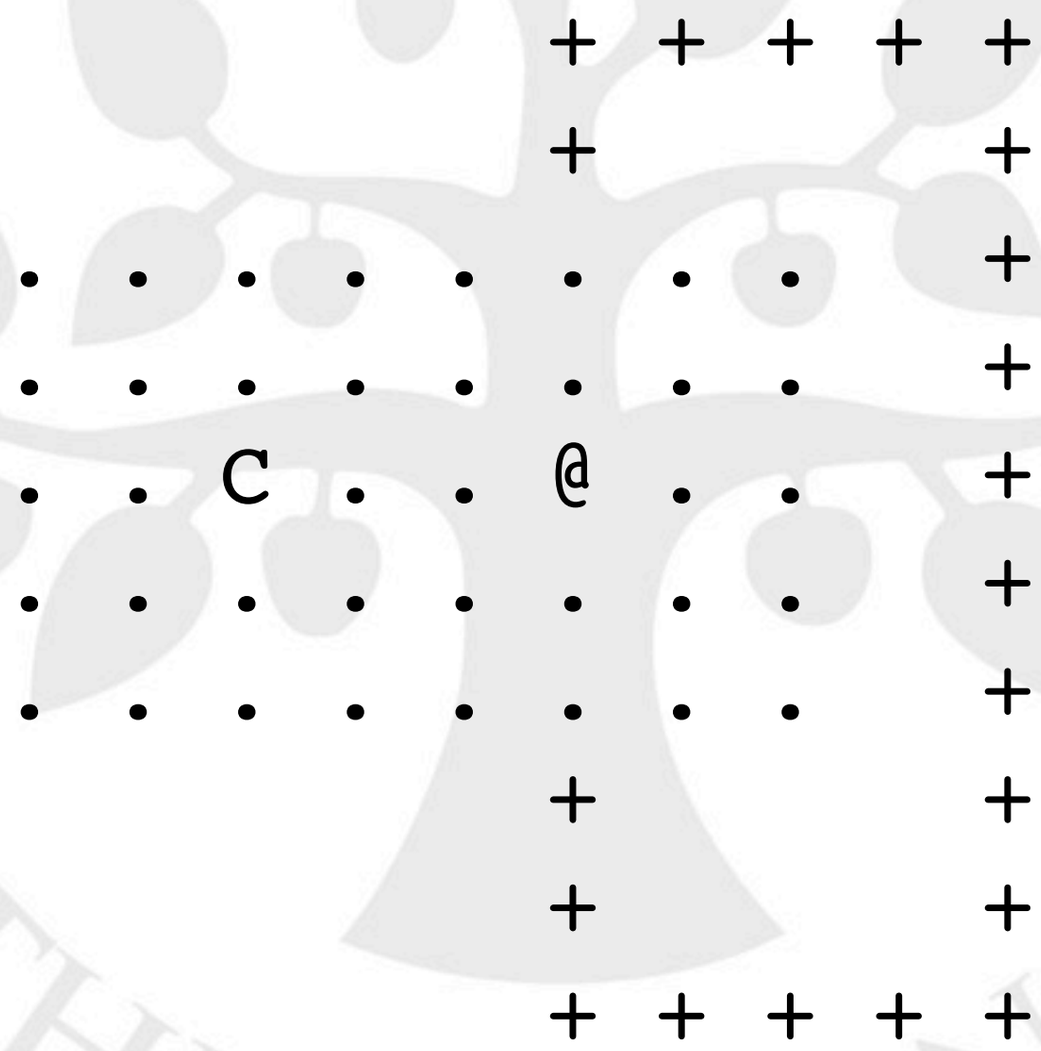
- Begge strategier er ikke optimale

Ikke optimale

- Begge strategier er ikke optimale
- Eksempel for monsteret

Ikke optimale

- Begge strategier er ikke optimale
- Eksempel for monsteret



Løsningen

Løsningen

- I får givet fire filer I skal bruge

Løsningen

- I får givet fire filer I skal bruge
- Klassen Position

Løsningen

- I får givet fire filer I skal bruge
- Klassen Position
  - Repræsenterer en position på spillepladen

Løsningen

- I får givet fire filer I skal bruge
- Klassen Position
  - Repræsenterer en position på spillepladen
- Klassen Dungeon

Løsningen

- I får givet fire filer I skal bruge
- Klassen Position
  - Repræsenterer en position på spillepladen
- Klassen Dungeon
  - Repræsenterer spillepladen

Løsningen

- I får givet fire filer I skal bruge
- Klassen Position
  - Repræsenterer en position på spillepladen
- Klassen Dungeon
  - Repræsenterer spillepladen
  - Kan aflæse heltens og monsterets position

Løsningen

- I får givet fire filer I skal bruge
- Klassen Position
  - Repræsenterer en position på spillepladen
- Klassen Dungeon
  - Repræsenterer spillepladen
  - Kan aflæse heltens og monsterets position
  - Kan aflæse alle felter og afgøre hvad der er lovlige ryk

Løsningen

- I får givet fire filer I skal bruge
- Klassen Position
  - Repræsenterer en position på spillepladen
- Klassen Dungeon
  - Repræsenterer spillepladen
  - Kan aflæse heltens og monsterets position
  - Kan aflæse alle felter og afgøre hvad der er lovlige ryk
- Klassen AbstractCharacter

Løsningen

- I får givet fire filer I skal bruge
- Klassen Position
  - Repræsenterer en position på spillepladen
- Klassen Dungeon
  - Repræsenterer spillepladen
  - Kan aflæse heltens og monsterets position
  - Kan aflæse alle felter og afgøre hvad der er lovlige ryk
- Klassen AbstractCharacter
  - Repræsenterer en spiller (monster/helt)

Løsningen

- I får givet fire filer I skal bruge
- Klassen Position
  - Repræsenterer en position på spillepladen
- Klassen Dungeon
  - Repræsenterer spillepladen
  - Kan aflæse heltens og monsterets position
  - Kan aflæse alle felter og afgøre hvad der er lovlige ryk
- Klassen AbstractCharacter
  - Repræsenterer en spiller (monster/helt)
  - Kender spillepladen og sin egen position

Løsningen

- I får givet fire filer I skal bruge
- Klassen Position
  - Repræsenterer en position på spillepladen
- Klassen Dungeon
  - Repræsenterer spillepladen
  - Kan aflæse heltens og monsterets position
  - Kan aflæse alle felter og afgøre hvad der er lovlige ryk
- Klassen AbstractCharacter
  - Repræsenterer en spiller (monster/helt)
  - Kender spillepladen og sin egen position
  - Har en metode move() som returnerer spilleres nye position når der skal rykkes

Løsningen

Løsningen

- Klassen GameEngine

Løsningen

- Klassen GameEngine
  - Styrer afvikligen af spillet

Løsningen

- Klassen GameEngine
  - Styrer afvikligen af spillet
  - Indeholder main-metoden

Løsningen

- Klassen GameEngine
  - Styrer afvikligen af spillet
  - Indeholder main-metoden
- Spillet startes på følgende måde

Løsningen

- Klassen GameEngine
  - Styrer afvikligen af spillet
  - Indeholder main-metoden
- Spillet startes på følgende måde
  - java GameEngine dungeonA.txt Rogue Monster

Løsningen

- Klassen GameEngine
  - Styrer afvikligen af spillet
  - Indeholder main-metoden
- Spillet startes på følgende måde
  - java GameEngine dungeonA.txt Rogue Monster
  - dungeonA.txt er en fil med spillepladen

Løsningen

- Klassen GameEngine
  - Styrer afvikligen af spillet
  - Indeholder main-metoden
- Spillet startes på følgende måde
  - java GameEngine dungeonA.txt Rogue Monster
  - dungeonA.txt er en fil med spillepladen
  - Rogue er klassen som implementerer helten

Løsningen

- Klassen GameEngine
  - Styrer afvikligen af spillet
  - Indeholder main-metoden
- Spillet startes på følgende måde
  - java GameEngine dungeonA.txt Rogue Monster
  - dungeonA.txt er en fil med spillepladen
  - Rogue er klassen som implementerer helten
  - Monster er klassen som implementerer monsteret

Løsningen

- Klassen GameEngine
  - Styrer afvikligen af spillet
  - Indeholder main-metoden
- Spillet startes på følgende måde
  - java GameEngine dungeonA.txt Rogue Monster
  - dungeonA.txt er en fil med spillepladen
  - Rogue er klassen som implementerer helten
  - Monster er klassen som implementerer monsteret
- I må ikke rette i de fire filer

Løsningen

- Klassen GameEngine
  - Styrer afvikligen af spillet
  - Indeholder main-metoden
- Spillet startes på følgende måde
  - java GameEngine dungeonA.txt Rogue Monster
  - dungeonA.txt er en fil med spillepladen
  - Rogue er klassen som implementerer helten
  - Monster er klassen som implementerer monsteret
- I må ikke rette i de fire filer
  - Skal kunne bruge andres kode

Løsningen

- Klassen GameEngine
  - Styrer afvikligen af spillet
  - Indeholder main-metoden
- Spillet startes på følgende måde
  - java GameEngine dungeonA.txt Rogue Monster
  - dungeonA.txt er en fil med spillepladen
  - Rogue er klassen som implementerer helten
  - Monster er klassen som implementerer monsteret
- I må ikke rette i de fire filer
  - Skal kunne bruge andres kode
- I kan hente de fire filer og nogle spilleplader på kursets hjemmeside

Opgaven

Opgaven

- Skriv to klasser

Opgaven

- Skriv to klasser
  - Rogue

Opgaven

- Skriv to klasser
  - Rogue
    - Nedarver fra AbstractCharacter

Opgaven

- Skriv to klasser
  - Rogue
    - Nedarver fra AbstractCharacter
    - Implementerer heltens strategi

Opgaven

- Skriv to klasser
  - Rogue
    - Nedarver fra AbstractCharacter
    - Implementerer heltens strategi
  - Monster

Opgaven

- Skriv to klasser
  - Rogue
    - Nedarver fra AbstractCharacter
    - Implementerer heltens strategi
  - Monster
    - Nedarver fra AbstractCharacter

Opgaven

- Skriv to klasser
  - Rogue
    - Nedarver fra AbstractCharacter
    - Implementerer heltens strategi
  - Monster
    - Nedarver fra AbstractCharacter
    - Implementerer monsters strategi

Opgaven

- Skriv to klasser
  - Rogue
    - Nedarver fra AbstractCharacter
    - Implementerer heltens strategi
  - Monster
    - Nedarver fra AbstractCharacter
    - Implementerer monsters strategi
- I skal aflevere

Opgaven

- Skriv to klasser
  - Rogue
    - Nedarver fra AbstractCharacter
    - Implementerer heltens strategi
  - Monster
    - Nedarver fra AbstractCharacter
    - Implementerer monsters strategi
- I skal aflevere
  - den sædvanlige rapport med test (engelsk hvis mulig)

Opgaven

- Skriv to klasser
  - Rogue
    - Nedarver fra AbstractCharacter
    - Implementerer heltens strategi
  - Monster
    - Nedarver fra AbstractCharacter
    - Implementerer monsters strategi
- I skal aflevere
  - den sædvanlige rapport med test (engelsk hvis mulig)
  - De to filer Rogue.java og Monster.java

Opgaven

- Skriv to klasser
  - Rogue
    - Nedarver fra AbstractCharacter
    - Implementerer heltens strategi
  - Monster
    - Nedarver fra AbstractCharacter
    - Implementerer monsters strategi
- I skal aflevere
  - den sædvanlige rapport med test (engelsk hvis mulig)
  - De to filer Rogue.java og Monster.java

Opgaven

- Skriv to klasser
  - Rogue
    - Nedarver fra AbstractCharacter
    - Implementerer heltens strategi
  - Monster
    - Nedarver fra AbstractCharacter
    - Implementerer monsters strategi
- I skal aflevere
  - den sædvanlige rapport med test (engelsk hvis mulig)
  - De to filer Rogue.java og Monster.java

- Afleveringsfrist d. 10. januar 2010 kl. 12

Vejledning

- diskussionsforum i Blackboard
- “Discussion Board” -> “Project Part 2”

- faste træffetider med underviseren
  - Mandag 12-14
  - Torsdag 12-14
- indenfor træffetider normal hurtig svar (minutter)
- udenfor træffetider svar efter mulighed (ca. 1-3 dage)

- spørgsmålene og svarene tilgængelig for alle deltager
- i må meget gerne diskutere spørgsmålene med hinanden