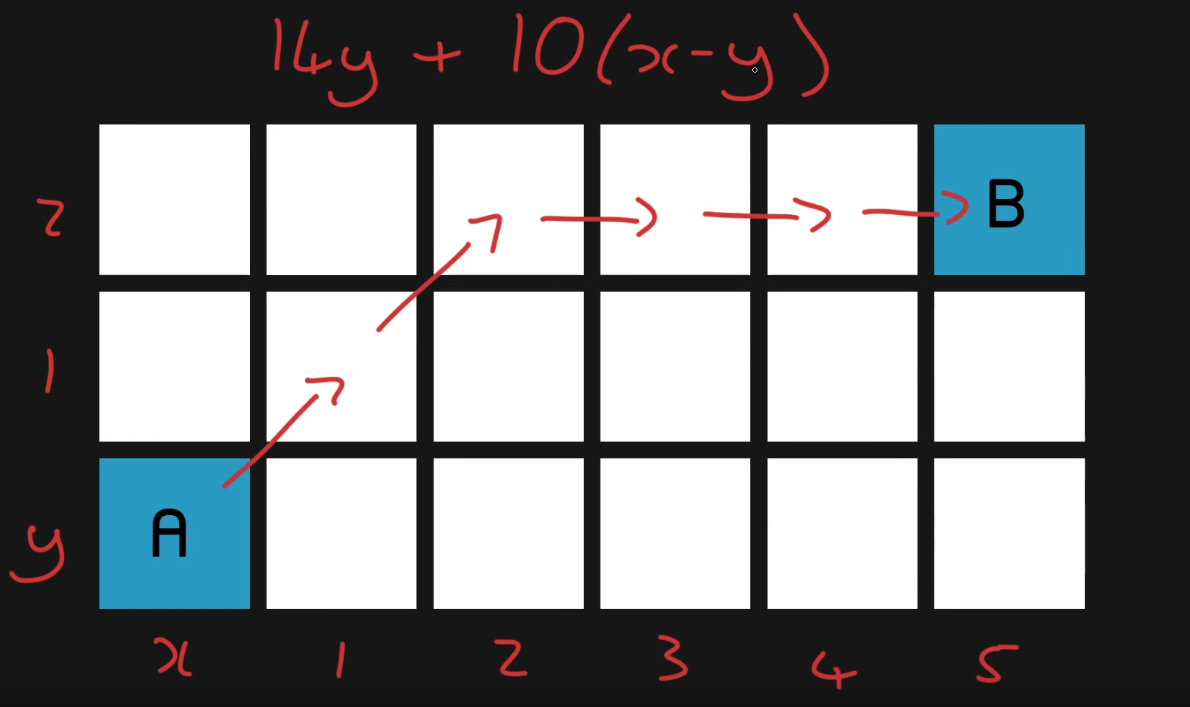
# A\*

De gekozen afstandsfunctie bevindt zich in Map.cpp en heet findPath(). De functie is geïmplementeerd met behulp van [deze](https://youtu.be/-L-WgKMFuhE) en [deze](https://youtu.be/mZfyt03LDH4) video’s.

1.

De heuristiek wordt op de volgende manier berekend:



Als y groter dan x was geweest was de formule 14x +10(y - x) geweest

* Het laagste nummer (In dit geval y) geeft altijd aan hoeveel diagonale stappen nodig zijn om in lijn te komen met de target node.
* De overige horizontale of verticale stappen worden berekend door het laagste nummer van het hoogste nummer af te trekken. (In dit geval x - y)

2.

Bij het berekenen van de afstand naar een buur wordt rekening gehouden met het gewicht van de buur door deze te vermenigvuldigen met de afstand van de huidige buur naar die buur toe.

3.

De manager vind zijn pad naar artiesten toe door de findPath() methode uit te voeren met als starting tile zijn tile en als target tile de tile waarop de artiest zich bevindt.

# Eindigetoestandsautomaten

**Klasse**: Artist

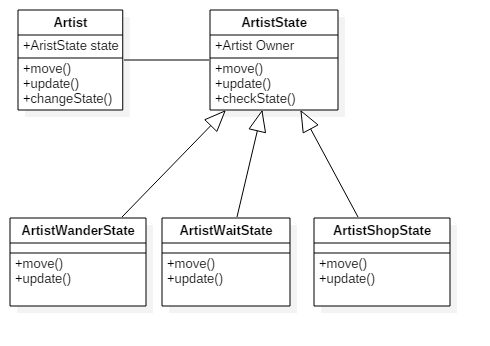
**Toestanden**:

* ArtistWanderState
  + In deze toestand loopt de artist willekeurig rond door de stad.
  + Bij elke stap verliest de artist 20 euro. Als de artist minder dan 100 euro op zak heeft en te horen krijgt van de manager dat deze naar hem toe komt transioneert zijn state naar ArtistWaitState
* ArtistWaitState
  + In deze toestand staat de artist stil en wacht hij op een bezoek van de manager
  + Als de manager hem heeft bereikt transioneert zijn state naar ArtistWanderState
* ArtistShopState
  + In deze toestand loopt de Artiest naar een van de 4 winkels.
  + Als de Artist de winkel heeft bereikt wacht hij 3 periodes en transioneert dan zijn state naar ArtistWanderState

**Klasse**: Manager

**Toestanden**:

* ManagerWanderState
  + In deze toestand loopt de manager willekeurig rond door de stad.
  + Elke stap checkt wordt er gecheckt of een van de artiesten de manager nodig heeft. Als dit zo is transioneert de manager naar ManagerVisitArtistState.
* ManagerVisitArtistState
  + In deze toestand loopt de manager naar de artist die hulp nodig heeft.
  + Als de manager bij de artiest is aangekomen transioneert de state van de artiest naar ArtistShopState en de state van de manager naar ArtistWanderState.



# Force-driven entities

**Klasse:** Fan

**Krachten:**

* CohesionVector
  + CohesionVector wordt berekend door voor iedere nabije fan een vector naar die fan toe te berekenen en deze bij elkaar op te tellen tot 1 vector.
* SeparationVector
  + SeparationVector wordt berekend door voor iedere nabije fan een vector van die fan af te berekenen en deze bij elkaar op te tellen tot 1 vector.
* AlignmentVector
  + AlignmentVector wordt berekend door voor de richtingsvectoren van nabije fans bij elkaar op te tellen tot 1 vector.
* AttractedToArtistsVector
  + AttractedToArtistsVector wordt berekend door voor iedere artiest een vector naar die artiest toe te berekenen en deze bij elkaar op te tellen tot 1 vector. (Als een fan niet aangetrokken is tot een bepaalde artiest kunnen deze vectoren ook vectoren van de artiesten af worden.)
* SteeringVector
  + Steering wordt berekend door een vierkant te tekenen in de richting waar de fan naartoe wil en daarop een willekeurig punt te pakken als vector.

Al deze vectoren worden bij elkaar opgeteld om een vector te creeëren die de richting van de Fan bepaald.

# Genetisch Algoritme

Aan het begin van de eerste simulatie genereerd het programma 100 fans met ieder een random chromosoom.

Voor de fitness van de fans wordt de tijd die ze in de buurt van artiesten hebben doorgebracht gebruikt.

Aan het einde van iedere simulatie wordt een nieuwe populatie fans gegenereerd middels Fitness Proportionate Selection. Dit algoritme is geïmplementeerd met behulp van [deze](https://en.wikipedia.org/wiki/Fitness_proportionate_selection) webpagina.

Ouders worden bepaald met behulp van roulette selection. Ieder fan heeft een kans om gekozen te worden als ouder. Fans met een hogere fitness hebben een hogere kans. Fans die zijn overleden tijdens de simulatie hebben een fitness van 1. Dit is gedaan omdat dit aansluit bij de overlevingsdrang van de fans.

Nakomelingen worden gegenereerd door de chromosomen van 2 ouders op een willekeurig punt te snijden en aan elkaar te koppelen tot een nieuw chromosoom. Daarnaast bestaat er ook nog een kans dat er een mutatie voorkomt in een chromosoom

De grote van de nieuwe populatie is altijd 2/3e van de vorige populatie totdat het minimum van 18 is bereikt.