# Johdanto

## Dokumentin tarkoitus

Tämä dokumentti on määrittelydokumentti kurssia TKT20010 Tekoäly ja algoritmit varten. Kyseessä syksyn 2024 2. implementaatio (28.10.-20.12.2024). Dokumentissa kuvataan toteutuksen tärkeimmät ominaisuudet, toiminnallisuudet ja toteutusvälineet kurssin vaatimusten mukaisessa laajuudessa.

## Projektin yleiskuvaus

Projektin aihe on ”Verkot ja polunetsintä”. Projektissa toteutetaan algoritmit

1. Dijkstra
2. Fringe search

Algoritmeja sovelletaan painotettuihin verkkoihin, joissa kaarilla on ei-negatiiviset, rajoittamattomat painot. Mikäli mahdollista, testaamisessa pyritään fokusoimaan reaalimaailman kartta-aineistojen tyyppiseen materiaaliin. Toteutuksessa tuotetaan ensin käyttöliittymän perustoiminnallisuudet karttojen ja algoritmien suorittamisen hallintaan, sen jälkeen Dijkstra-algoritmi riittävälle tasolle testausta varten ja lopuksi Fringe Search.

## Opinto-ohjelma

Suoritan kurssia erillisten opintojen suoritusoikeudella. En kuulu mihinkään opinto-ohjelmaan.

# Ohjelmien määrittely

## Toteutettavat algoritmit

### Dijkstra

Dijkstra-algoritmista toteutetaan yksisuuntainen, prioriteettijonolla varustettu versio.

Alkuperäisen Dijkstran (1959) tieteellisen artikkelin esittelemän version aikavaativuus oli O(V^2). Toteutettavan prioriteettijonollisen version aikavaativuus on O((E+V) log(V)). Tästä ole ei löydettävissä selkeää alkuperäisviitettä, mutta periaate on esitetty lukuisissa artikkeleissa ja Wikipediassa. Algoritmin tilavaativuus riippuu toteutustavasta, mutta se on lineaarinen O(V) tai O(V+E). Tilavaatimus lienee tässä tapauksessa vähemmän oleellinen kuin aikavaativuus eikä siitä ei ole löydettävissä selkeää tieteellistä viitettä.

Toteutuksen lähtökohtana on Wikipedian viitteen luvusta *Pseudocode* löytyvä pseudokoodi.

Ajan mahdollistaessa on mahdollista lisäksi tutkia optimointeja kuten kaksisuuntaisuus ja kaksisuuntaisen jonon laskennan rinnakkaistaminen eri prosesseihin. Lähtöoletus kuitenkin on, että näitä ei toteuteta.

### Fringe search

Fringe search (*Björnsson et al., 2005*) on selkeästi rajattu algoritmi, jossa käytännön muuntelua voi tapahtua lähinnä heuristiikkafunktion valinnassa. Alustava oletus on, että heuristisena funktiona käytetään euklidista etäisyyttä pisteiden välillä.

Fringe searchin aikavaativuus on O((V+E) log(V)) ja tilavaativuus O(V), joten se on tehokkuusluokaltaan tasoa Dijkstran kanssa.

Toteutuksen lähtökohtana toimii Bjönrsson et al. (2005) artikkelin luvusta 2.3 löytyvä pseudokoodi.

Tässä vaiheessa ei ollut mahdollista ajankäytön vuoksi perehtyä tarkemmin fringe searchin yksityiskohtiin, joten toteutettavan kokonaisuuden mahdollinen tarkentaminen jätetään tehtäväksi myöhemmillä harjoitusviikoilla.

## Kartta

Kartan kuvaamiseen on ainakin kaksi peruslähestymistapaa: kaksiulotteinen ruudukko, jossa ruutu on joko vapaa tai liikkumisen estävä, tai solmujen ja kaarten kuvaaminen niin, että kaareen liittyvä paino liitetään kaaren yhteyteen. Alustavasti tässä työssä valitaan jälkimmäinen tapa, jolloin kaarten painojen arvojoukko ei ole rajoitettu eikä sidottu sijaintiin karttamatriisissa. Solmut ja kaaret kuvataan jollain tehokkaalla tavalla taulukoiksi tai matriiseiksi.

Tällaisen vapaamuotoisen kartan voi esittää ainakin kuvaamalla verkon kaaret taulukoiksi tai kuvaamalla kartan VxV-kokoisena matriisina, jossa solu (x, y) sisältää solmujen x ja y välisen etäisyyden tai nollan, jos yhteyttä ei ole. Jälkimmäinen tapa luonnollisesti nostaa algoritmin tilavaatimuksen luokkaan O(v^2). Tässä työssä valitaan lähtökohtaisesti ensin mainittu eli kuvataan verkon kaaret sopivalla koodauksella. Tämä on tilavaativuudelta universaalisti toimivampi tapa mutta tehnee verkon visualisoinnista haastavamaa. Tarkka koodaustapa suunnitellaan myöhemmin projektin edetessä.

Kartan esittämistapa pyritään validoimaan ohjaajan kanssa 2. kurssiviikolla sen varmistamiseksi, että työ lähtee toteutettavuuden ja tehokkuuden kannalta järkevään suuntaan.

## Toiminnallisuudet

Alustavat toiminnallisuudet on listattu alla Epiceinä ja Featureina. Toiminnallisuudet on määritelty harjoitustyön toteuttamisen näkökulmasta sisältäen kaikki tuotettavat lopputuotokset, ei normaalista ohjelmiston toiminnallisen määrittelyn näkökulmasta.

* **Epic 1. Validointi ja ohjeet**
  + *Feature 1.1 Testimateriaalien hankkiminen*
    - Tuotos: testeissä käytettävät kartat
  + *Feature 1.2 Testisuunnitelman tuottaminen* 
    - Tuotos: testausdokumentin suunnitelmaosuus
  + *Feature 1.3 Validoinnin toteutus ja dokumentointi*
    - Tuotos: testausdokumentin testitulosten osuus
  + *Feature 1.4 Käyttöohjeen tuottaminen*
    - Tuotos: käyttöohje
* **Epic 2. Käyttöliittymä**
  + *Feature 2.1 Lataa kartta* 
    - Toiminnot: lataa kartan käyttöliittymästä, muuntaa sen sisäiseen esitysmuotoon ja visualisoi sen käyttäjälle
    - Tuotos: käyttöliittymä, joka pystyy lataamaan ja visualisoimaan kartan
  + *Feature 2.2 Laske reitti*
    - Toiminnot: laskee reitin valitusta alkupisteestä valittuun loppupisteeseen valitulla kartalla ja algoritmilla
    - Tuotos: käyttöliittymä, joka pystyy suorittamaan reitin laskemisen halutuilla parametreilla
  + *Feature 2.3 Visualisoi reitti*
    - Toiminnot: visualisoi sisäisessä esitysmuodossa olevan, algoritmin laskeman reitin kartalle käyttöliittymässä
    - Tuotos: käyttöliittymä, joka pystyy visualisoimaan reitin
  + *Feature* 2.3 Tallenna tulokset
    - Tuotos: käyttöliittymä, joka pystyy tallentamaan laskennan tulokset niin, että ne ovat jatkohyödynnettävissä
* **Epic 3. Dijkstra**
  + *Feature 3.1 Dijkstran algoritmin perustoteutus*
    - Toiminnot: algoritmi saa parametrina kartan sekä alku- ja loppupisteen ja laskee jollain tavoin reitin niiden välillä
    - Tuotos: toimiva mutta ei lopullinen versio algoritmista
  + *Feature 3.2 Prioriteettijono ja optimointi*
    - Toiminnot: algoritmi saa parametrina kartan sekä alku- ja loppupisteen ja laskee reitin niiden välillä
    - Tuotos: lopullinen versio algoritmista
* **Epic 4. Fringe Search**
  + *Feature* 4.1 Fringe search perustoteutus
    - Tuotos: ajettavissa oleva versio algoritmista
  + *Feature* 4.2 Fringe search viimeistely
    - Tuotos: lopullinen versio algoritmista

# Toteutusvälineet

Toteuskieli on Java

* Java, versio 23
* JavaFX käyttöliittymän toteutukseen
* JUnit testaamiseen

Toteutusympäristö

* Windows 11 desktop PC
* OpenJDK 23.0 + JavaFX 23.0
* VSCode 1.94 IDE + VSCode Extension Pack for Java (Microsoft)

Muita kieliä mahdollista harjoitustyön vertaisarviointia varten

* Python: opintojen myötä kohtalaisesti kokemusta mutta ei ammatillista kokemusta
* JavaScript: jonkin verran kokemusta ammatillisesta historiasta ja opinnoista
* Fortran77: runsaasti ammatillista kokemusta mutta vuosikymmenten takaan

# Viitteet

Dijkstra, E.W (1959). A note on two problems in connexion with graphs. *Numer. Math.* **1**, 269–271 (1959). <https://doi.org/10.1007/BF01386390>

Wikipedian sisällöntuottajat (n.d.), Dijkstra's algorithm. *Wikipedia, avoin tietosanakirja.* Haettu 2.11.2024 osoitteesta<https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm>

Björnsson, Y., Enzenberger, R., Holte, R., Shaeffer, J. (2005). Fringe Search: Beating A\* at Pathfinding on Game Maps. *Epäformaali tieteellinen artikkeli.* Noudettu osoitteesta <https://webdocs.cs.ualberta.ca/~holte/Publications/fringe.pdf>