# Määrittelydokumentti / Specification Document

**Data Structures Lab Course (start of summer 2023)**

The problem to be solved in this project is to implement an efficient solution to find the shortest distance path between two street addresses. Algorithms and data structures I implement in this work includes IDA* graph data structure, and possibly other data structures enabling street address data management. Why I have chosen those algorithms/data structures is that IDA*(Iterative Deepening A*) enjoys a better space complexity then A* and Fringe Search. And it can combine depth-first and breadth-first search advantages. One major practical drawback of A* is its $O(b^d)$ space complexity. IDA* algorithm uses less memory than the A* algorithm because it simply keeps track of the present node. IDA* algorithm's worst case could be $O(b^d)$. It is a depth-first search (DFS) algorithm, the estimated time complexity in the planned program could be $O(|V| + |E|)$, where $|V|$ is the number of nodes and $|E|$ the number of edges.

In order to implement IDA* for this project, it's critical to know how to convert the problem into a graph with nodes and edges, and a heuristic estimation for the problem for optimal and faster search probably will be needed. OpenStreetMap (https://www.openstreetmap.org) will be used as the map services for this project.

Inputs are starting node's address and destination node's address. They will be converted into a graph with nodes and edges. The algorithm is expected to take the graph, starting node, and destination node as inputs and performs computations to find the shortest path.

Programming language **Python** will be used for this course. Other languages I can understand are Java, PHP, Perl, HTML.

Sources used are listed below:

- What is ASCII art: https://en.wikipedia.org/wiki/ASCII_art
- JPS: https://en.wikipedia.org/wiki/Jump_point_search
- A Visual Explanation of Jump Point Search: https://zerowidth.com/2013/a-visual-explanation-of-jump-point-search.html
- IDA*:
  https://en.wikipedia.org/wiki/Iterative_deepening_A*
  https://www.geeksforgeeks.org/iterative-deepening-a-algorithm-ida-artificial-intelligence/
- Fringe search: https://en.wikipedia.org/wiki/Fringe_search
- A*: https://en.wikipedia.org/wiki/A*_search_algorithm
- Map Service: OpenStreetMap https://www.openstreetmap.org

This course is for my Bachelor's Degree in Computer Science (TKT, tietojenkäsittelytieteen kandidaatti). The language used in this documentation file will be English. In order to ensure language consistence, the text of the project's code, comments and all documents will be also in **English**.