




















# Testausdokumentti

## Projektin yksikkötestaus

Projektin yksikkötestauksessa on pyritty saavuttamaan datastructures- ja algorithms -paketeissa 100% koodikattavuus. Yksikkötestejä kirjoitettiin 91 kappaletta ja tavoite saavutettiin. Koodikattavuuden tarkistamiseen käytettiin pääasiassa Eclipse-kehitysympäristön liitännäistä eclEmma. Maven projektitiedosto sisältää lisäksi Cobertura liitännäisen, jolla saatiin sama tulos.

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
tiralabra	 76.7 %	1,901	576	2,477
src/main/java	 76.7 %	1,901	576	2,477
fi.miko.tiralabra	 0.0 %	0	576	576
Benchmark.java	 0.0 %	0	194	194
Main.java	 0.0 %	0	164	164
Solver.java	 0.0 %	0	218	218
fi.miko.tiralabra.algorithms	 100.0 %	1,208	0	1,208
AStar.java	 100.0 %	124	0	124
BellmanFord.java	 100.0 %	126	0	126
Dijkstra.java	 100.0 %	84	0	84
Edge.java	 100.0 %	50	0	50
Graph.java	 100.0 %	220	0	220
GraphUtils.java	 100.0 %	128	0	128
Heuristic.java	 100.0 %	103	0	103
Node.java	 100.0 %	193	0	193
PathFinder.java	 100.0 %	180	0	180
fi.miko.tiralabra.datastructures	 100.0 %	693	0	693
LinkedList.java	 100.0 %	321	0	321
MinimumHeap.java	 100.0 %	372	0	372

Kuva eri tiedostojen koodikattavuuksista.

Ohjelmalogiikkaa sisältäviä luokkia ei yksikkötestattu. Syyt tähän ovat seuraavat:

- Kyseiset luokat sisältävät vain 23.3% ohjelmakoodista ja ovat rakenteeltaan yksinkertaisia.
- Ohjelman oikea toiminta voidaan todeta järjestelmätestauksessa.

## Reitinhakualgoritmien testaus

Lyhimpien polkujen etsimisen oikeellisuus testattiin pääasiassa yksikkötestien avulla. Testit sisältävät 44 käsin valittua labyrinttiä joihin on merkattu lyhin polku valmiiksi. Labyrinttien koot ovat 1x2, 1x3, 2x1, 2x2, 2x3, 3x1, 3x2, 3x3 ja 4x4 ruutua. Suurin 4x4 kokoinen labyrintti sisältää avoimia esteitä jotka perinteisesti tuottavat ongelmia reitinhakualgoritmeille.

Koska algoritmeja toteutettiin useampia, ajetaan testit samanaikaisesti kaikille algoritmeille ja tarkistetaan että jokainen algoritmi tuottaa saman tuloksen. Koska lyhimpiä polkuja voi olla useampia, tarkistetaan, että löydettyjen polkujen pituudet ja painot ovat samat.

Lisäksi testataan satunnaisia labyrinttejä joiden koot ovat välillä 2x2 - 50x50. Seinäruutuja labyrintit sisältävät välillä 20-40%. Näitä labyrinttejä testataan yhteensä  $48 * 48 * 20 = 46080$  kappaletta. Oikeellisuus varmistetaan taas vertailemalla useamman algoritmin tulosta. Bellman-Fordin algoritmia ei testata isommilla syötteillä sen tehottomuuden vuoksi.

Koska toteutetut algoritmit toimivat samalla tavalla riippumatta labyrintin koosta, voidaan tämän testausmenetelmän tulokseen luottaa.

## ***Tietorakenteiden testaus***

Tietorakenteet testattiin yksikkötesteillä. Testauksessa käytettiin lasilaatikkotestausta ja luokkien toimintaa testattiin ekvivalenssiluokkien raja-arvojen avulla.

Minimikeon testausta varten kekon toteutettiin lisäksi metodi toArray() joka palauttaa keon sisällön taulukkomuodossa. Testeissä tarkistetaan ja vaaditaan kekoehdon voimassaolo.

## ***Toiminnallisuuden testaus***

Ohjelman toiminnallisuus testattiin Windows, OS X ja Linux -käyttöjärjestelmillä. Testauksessa jokaista ominaisuutta kokeiltiin useamman kerran ja varmistettiin toiminnan oikeellisuus.

Testissä kokeiltiin myös virheellisiä syötteitä ja todettiin stack tracet riittäviksi virheilmoituksiksi.