Testausdokumentti

Labyrintingeneroija ja –ratkoja

Tietorakenteet ja algoritmit harjoitustyö

Juri Kuronen

Kesä 2014

Sisältö

1. Johdanto

Labyrintingenerointi ja –ratkoja –ohjelman toimivuutta on pääosin testattu JUnit-yksikkötesteillä, mutta ohjelmaa on voinut testata myös graafisen käyttöliittymän avulla. Kattavuus on pyritty saamaan mahdollisimman korkeaksi siten, että kaikkia luokkia ja niiden metodeja on testattu. Jokaiselle mahdollisesti pieleen menevälle asialle, joka on tullut mieleen, on yleensä kirjoitettu jokin yksikkötesti. Graafista käyttöliittymää on kuitenkin testattu pääosin kokeilemalla.

Tässä dokumentissa selostan tarkemmin millä tavalla ohjelman mitäkin osia on testattu, ja raportoin testien tulosta.

2. Testaus

2.1. Esipuhe

Ohjelmoinnin aikana testit ovat toimineet suurena apuna, sillä niiden avulla on saanut virheiden sattuessa täsmällistä tietoa siitä, mikä ei ole toiminut. Graafisen käyttöliittymän avulla on pystynyt havainnoida toimivatko eri generointi- ja ratkonta-algoritmit oikealla tavalla.

2.2. JUnit-yksikkötestaus

Labyrintti-luokka.

- Labyrintin kokotiedot ja 2-ulotteinen byte-array asettuvat oikein konstruktorin ja updateLabyrinth()-metodin kautta.
- Labyrintin kahden solun välisen reitin lisäys toimii oikein. Reitin voi lisätä vain labyrintin sisällä olevien, vierekkäisten solujen välille.
- Solun vierailtujen/vierailemattomien naapureiden haku, kaarella yhdistettynä tai ilman, hakee oikeat naapurit.
- Labyrintin ulkopuolisia koordinaatteja ei voi käyttää.
- Labyrintista voi hakea kaaret oikein.

Labyrintin generoijat.

- Labyrintin generoija tyhjentää labyrintin ennen kuin aloittaa työn.
- Generoija käsittelee ja täyttää kaikki solut.
- Kaikki solut ovat vierailtavissa mistä tahansa solusta.
- Ison labyrintin (100x100) luominen onnistuu eikä vie kamalasti aikaa (yli 1sec).

Labyrintin ratkojat.

Labyrintin ratkojien testit ajetaan kaikkien labyrintin generoijien generoimilla labyrinteilla.

- Algoritmi löytää reitin maaliin.
- Algoritmin löytämä reitti on validi, eli kulkee kaaria pitkin ja käy maalissa vasta reitin lopuksi.
- Vierailtujen solujen määrä haetaan oikein.

Gui.

Graafista käyttöliittymää testataan yksikkötestein vain sen verran, että graafisen käyttöliittymän koko pysyy annetuissa rajoissa.

Tietorakenteet.

Lista.

- Lisääminen toimii oikein, myös silloin kun täytyy päivittää listan aputaulukon koko.
- Poistaminen toimii indeksin ja arvon avulla.
- Poistaminen palauttaa alkion.
- Listojen yhdistäminen toimii.
- Listan järjestyksen vaihtaminen toimii.
- Lista toimii abstrakteilla tietotyypeillä.

Pino.

- Lisääminen ja poistaminen toimivat.
- Abstraktit tietotyypit toimivat.

Jono.

- Jonoon lisääminen ja poistaminen toimivat: siirtää *head* ja *tail* kohtia oikein, päivittää listan aputaulukon koon oikein.

Prioriteettikeko.

- Minimikekoehto säilyy.
- Lisäys ja poisto toimivat, aputaulukko päivittyy oikein.

TreeNode (Labyrintin ratkojien aputietorakenne).

- Solmut linkittyvät oikein.

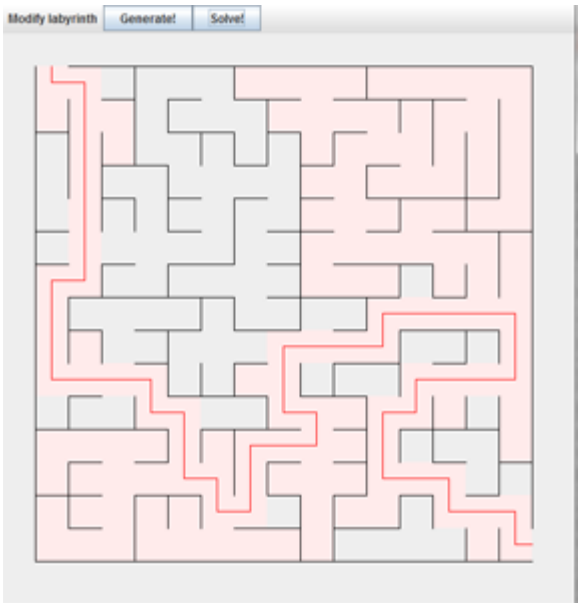
Joukkoalkio (Kruskalin algoritmin aputietorakenne).

- Joukot yhdistyvät oikein.
- Joukkojen siivous toimii oikein.

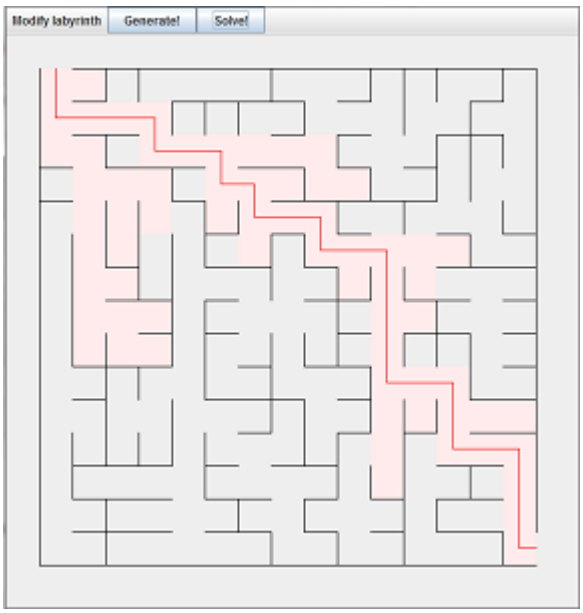
2.3. Testaus graafisen käyttöliittymän avulla

Graafisella käyttöliittymällä on kokeiltu toimivatko algoritmit (todella) oikein. Tässä muutama testitapaus. Kuvia on skaalattu 50% kokoon.

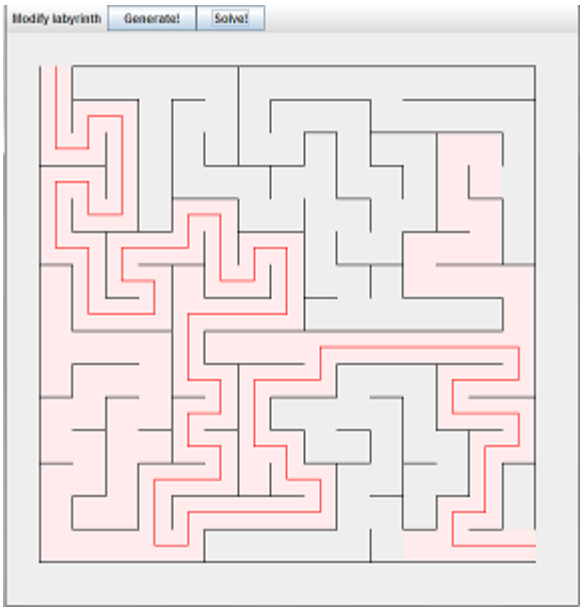
Testitapaus 1. Generointi Kruskalin algoritmilla, ratkaisu DFS:llä.



Testitapaus 2. Generointi Primin algoritmilla, ratkaisu A*-hakualgoritmilla.



Testitapaus 3. Generointi Rekursiivisella peruuttavalla haulla, ratkaisu BFS:llä.



Testitapaus 4. Generointi Kruskalin algoritmilla, ratkaisu Oikean käden sääntö –algoritmilla.