Lajittelualgoritmien vertailu

Toteutusdokumentti

Projektirakenne

Projektin Main-luokka, joka sisältää ohjelman käyttöliittymän toiminnallisuuden on omassa paketissaan “lajittelualgoritmien_vertailu”. Ohjelmaa varten luodut algoritmiluokat ovat erikseen paketissa “algoritmit”.

Algoritmit

Projektin algoritmit ovat laskemisjärjestäminen, lisäysjärjestäminen, kekojärjestäminen, intosort, quicksort ja timsort. Laskemisjärjestäminen toimii myös negatiivisilla arvoilla. Kekojärjestämien on introsortin vaatimusten vuoksi tavallisesta toteutuksesta hieman poikkeava versio, jossa järjestettäväksi voidaan antaa myös jokin väli taulukosta. Introsortin toiminnasta lukiessani kohtasin monia erilaisia malleja. Omassa toteutuksessani introsort järjestää annettua kynnysarvoa lyhyemmät osuudet lisäysjärjestämisen avulla. Muuten käytetään hyödyksi quicksortin partition-metodia ja kekojärjestämistä. Timsort on tämän harjoitustyön algoritmeista laajin projekti. Algoritmi järjestää taulukon onnistuneesti mutta on valitettavan hidas. Ainakin algoritmin merge-metodia ei ole optimoitu lainkaan. Siitä huolimatta suoritusnopeuden tulisi olla nykyistä parempi.

Käyttöliittymä

Käyttöliittymä on arkaainen ja yksinkertainen ja se sijaitsee Main-luokassa. Käyttäjän antamasta syötteestä taulukon asetuksiksi otetaan huomioon mahdolliset virheet. Sitten luodaan satunnaisia lukuarvoja sisältävä taulukko, joka kopioidaan kaikille algoritmeille erikseen järjestettäväksi. Jokaisen algoritmin suoritusaika mitataan ja tulostetaan erikseen. Tähän olisi varmasti olemassa parempikin toteutustapa. Käyttöliittymä sisältää itsessään myös ohjelman käyttöohjeet.

Tietorakenteet

Tietorakenteita projektissa ovat keko heap sort -algoritmissa, run eli alijono timsortissa ja pino myöskin timsortissa. Pino sisältää myös timsortin vaatiman tasapainoituksen toiminnallisuuden. Run kerää taulukon arvoja taulukkoon. Alijonon minimipituus on ennalta määrätty. Run tunnistaa onko alijono nouseva vai aidosti laskeva. Jos minimipituus ei täyty, kerätään arvoja kunnes niitä on minimipituuden verran ja järjestetään alijono lisäysjärjestämisen avulla.

Suorituskyvyn testaus

Algoritmit toimivat hienosti hidasta timsorttia lukuunottamatta. Introsort on joissakin tapauksissa jopa Javan omaa Arrays.sort-algoritmia nopeampi.

Esimerkkiajo:

```
Sorting with array length of 100000 and value range of 10000000
-Arrays.sort: 13.54 ms
-Counting sort: 130.21 ms
-Heapsort: 20.79 ms
-Insertion sort: 3446.37 ms
-Introsort: 9.05 ms
-Timsort(ish): 1164.32 ms
-Quicksort: 13.10 ms
```