

Demo 9 / 9.11

Tehtävät

- V1. Tee [Ville](#)-tehtävät: 5.11, 6.4, 6.5, 7.2
- T1. Katso [Wikistä OhjOpetuksenTyopaja](#) kohdasta ComTest kuinka ComTestiä käytetään. Kirjoita testit ja aja ne tehtävien 4 ja 5 aliohjelmiin. Myös 1-2 voi osittain testata (palauttaako porras oikean pisteen).
- 1-2. **M: 8. Oliotietotyypit:** Ota vanha [Portaat](#) -luokka ja muuta sitä niin, että seuraava pääohjelma toimii:

```
public static void main(String[] args) {
    Window window = new Window();
    window.scale(0, -1, 10, 10);
    RPoint next = new RPoint(0, 0);
    next = porras(window, next);
    next = porras(window, next);
    next = porras(window, next);
    next = porras(window, next);
    next = porras(window, next);
    next = new RPoint(next.getX() - 1, next.getY());
    next = porrasAlas(window, next);
    next = porrasAlas(window, next);
    next = porrasAlas(window, next);
    next = porrasAlas(window, next);
    next = porrasAlas(window, next);
    window.showWindow();
}
```

Vinkki: katso [Graphics](#) -kirjaston [RPoint](#)- ja [Line](#) -luokkien dokumentaatiota. Eli ideana on, että portaalille vietään parametrinä piste-olio kahden reaali-luvun sijaan. Kun porras piirtää itsensä, se palauttaa sen (uuden) pisteen., johon se päättyi. Näin seuraavan portaan piirtäminen on helppoa.

3. **M: 15. Toistorakenteet:** Tee edellisiä käyttäen tarvittavat aliohjelmat, jotta seuraava pääohjelma toimii (jos n=5, tekisi juuri saman kuvan kuin edellinen pääohjelma):

```
public static void main(String[] args) {
    int n = 50;
    Window window = new Window();
    window.scale(0, -1, 2*n, n+10);
    RPoint next = new RPoint(0, 0);
    next = portaat(window, next, n);
    next = new RPoint(next.getX() - 1, next.getY());
    portaatAlas(window, next, n);
    window.showWindow();
}
```

4. **M: 20. Dynaamiset tietorakenteet:** Katso mallia: [LaskeAKirjaimet.java](#). Tee mallia

mukaellen aliohjelma, joka etsii listan pisimmän sanan. Pääohjelma tulostaa pisimmän sanan ja poistaa sitten listasta KAIKKI tämän sanan esiintymät.

5. **M: 20. Dynaamiset tietorakenteet:** Kirjoita edelliseen kopioi (sanat, pituus) -aliohjelma, joka palauttaa uuden listan, johon on kopioitu kaikki ne listan sanat, joiden pituus on pituus.
6. **M: 20. Dynaamiset tietorakenteet , 22. Tiedostot:** Kirjoita edellisen tehtävän aliohjelmaa hyväksi käyttäen ohjelma, joka ottaa pääohjelman argumentista tiedoston nimen ja sitten tulostaa tiedoston niin, että tulostetaan vain ne rivit, jotka ovat yhtä pitkiä kuin pisin rivi. Helppotus: jos tiedostoa et saa muuten auki, niin ohjelman parametrinä saat antaa koko polun tiedostolle.

GURU-tehtävät

- G1-2. Tee ohjelma, joka lukee kaikki komentoriviparametreissa annetut tiedostot, lajittelee tuloksen ja poistaa kaikki samanlaisten rivien esiintymät. Samanlaisiksi tulkitaan rivit joissa on samat merkit isoista ja pienistä kirjaimista välittämättä. Lopuksi ohjelma tulostaa nuo rivit. Käyttötarkoitus voisi olla esimerkiksi sellainen, että on useita postilistoja joista halutaan koota yksi yhteinen, jossa kukin listalainen esiintyy vain yhden kerran. Koosta ohjelma riittävän pienistä paloista ja testaa ComTestillä kukin aliohjelma. Käytä mahdollisimman paljon valmiita listoja ja niihin liittyviä metodeja.
- G3. Kirjoita ComTest -testit tehtävään 6. Vinkkejä: main on kutsuttava funktio siinä missä muutkin funktiot. Sitten tutustu: [VertaaTiedosto](#), [Suuntaaja](#) ja [Suuntaaja.StringOutput](#) -luokkiin.
- G4-5. Tee ohjelma, joka saa jostakin taulukon (tai käyttää vakiotaulukkoa), jossa on kokonaislukuja. Taulukossa sama luku voi esiintyä useasti. Kirjoita aliohjelma, joka saa tuon taulukon parametrinaan ja palauttaa taulukon, jossa on taulukon alkiot kukin vain yhden kerran järjestettynä niiden esiintymiskertojen määrän mukaan nousevaan järjestykseen.

taulukosta {1,2,3,34,34,2,1,34,1,1,1} palautetaan {3, 2, 34, 1}