Demo 6 / 18.2

Tehtävät

Katso seuraavissa (tehtävät 1-3) Demo 5:n vastauksia tehtävään 2 (Pvm.java). Java tehtäviä saa merkitä täysille pisteille vain, mikäli ne tai suurin osa metodeista on automaattisesti testattu.

1. Kirjoita luokkaan `Pvm` saantimetodit päivälle, kuukausille ja vuosille.

2. Kirjoita tavallinen funktio (siis staattinen metodi) `compareTo(pv1,pv2)`, joka palauttaa `-1`, mikäli päivämäärä `pv1` on ennen päivämäärää `pv2`, `0` jos päivämäärät ovat samoja ja `1` muuten (`pv1` ja `pv2` ovat tyyppiä `Pvm`, tarvitseeko luokka `Pvm` muutoksia jos staattinen metodi `compareTo` siirrettäisiin toisessa tiedostossa olevaan luokkaan?). Luonnollisesti TDD testi ensin.

3*. Tee edellisestä funktiota vastaava metodi, eli lisää päivämäärä-luokkaan `Pvm` metodi `compareTo(pv2)`, joka palauttaa kuten edellä (`-1`,`0` tai `1`) kun verrataan oliota itseään ja päivämäärää `pv2`. Lisää nyt vielä `equals`-metodi. (Huom! jos et osannut 1&2. tehtävää, voi tämän tehdä silti). Tässäkin TDD-testi ensin.

```
// Käyttöesimerkki 2&3:een kun compareTo-metodit ovat Pvm-luokassa
Pvm pv1 = new Pvm(1,2),pv2 = new Pvm(3,3);
if ( compareTo(pv1,pv2) < 0 ) System.out.println(pv1 + " < " + pv2);
if ( pv1.compareTo(pv2) != 0 ) System.out.println(pv1 + " != " + pv2);
```

4. Kirjoita vertailun vuoksi luennolla jaetut olioalk/Elain.java, olioalk/KissaP.java ja olioalk/KoiraP.java käyttämällä perinnän sijasta rajapintaa, eli siten että pääohjelma näyttää samalta kuin esimerkissäkin olioalk/KoiraP.java, mutta `Elain`-luokka puuttuu (taulukossa alkioiden tyypiksi voidaan vaihtaa `ElainRajapinta`). Mitä voittaa? Mitä häviää? Lisää kummassakin "kirjoittamistavassa" vielä `Kotka`.

5* Kirjoita luokka `Kulkuneuvo`, jossa on ainakin nopeus ja matkustajien lukumäärä. Peri tästä luokat `Laiva` ja `Lentokone` joissa kummassakin on jokin oma erikoisominaisuus yleiseen kulkuneuvoon verrattuna. Kirjoita myös pieni testipääohjelma.

Seuraavia asioita ei ole vielä käsitelty luennolla, mutta ne on käsitelty Ohjelmointi 1-kurssilla ja myös monisteen luvussa 13.

6*. Tee aliohjelma `pienimman_paikka`, joka palauttaa kokonaislukutaulukon pienimmän alkion paikan (indeksin). Esim. taulukosta:

```
                        /*ta,he,ma,hu,to,ke,he,el,sy,lo,ma,jo*/
int k_pituudet[] = {31,28,31,30,31,30,31,31,30,31,30,31};

i = pienimman_paikka(k_pituudet); /* => i=1 */
```

Tee edellistä aliohjelmaa **käyttäen** aliohjelma `pienin`, joka palauttaa kokonaislukutaulukon pienimmän alkion arvon

```
n = pienin(k_pituudet);          // => n = 28
```

Tehtävissä 7&8 katso monisteen luku 11 ja 12:

7*. Piirrä kuva tietorakenteesta, jossa olisi levyjä ja kullakin levyllä voisi olla useita kappaleita. Jos omassa harjoitustyössäsi on vastaava tietorakenne, niin tämä käy myös.

8. Suunnittele oliot edelliseen "ohjelmaan" CRC-kortteja käyttäen.

B1-2 Muuta Astia2.java olio-ohjelmaksi, joka toimisi suurin piirtein seuraavasti:

```
...
public static void main(String args[]) {
  AstiaPeli peli = new AstiaPeli();
  peli.lisaa_astia("8",8);
  peli.lisaa_astia("5",5);
  peli.tulosta_ohje();
  peli.pelaa();
}
```

`Astia2`-luokkaan ei tarvitse tehdä mitään muutosta ja voit käyttää sitä sellaisenaan (Astia2-luokkaa ei tarvitse muuttaa, mutta osan siellä olevista staattisista metodeista voit kopioida tavallisiksi metodeiksi AstiaPeli-luokkaan). Pelin "graafinen versio" löytyy osoitteesta: http://www.mit.jyu.fi/vesal/kurssit/winohj/moniste/tentit/v00/

G1-2 Lisää astiapeliin seuraavanlainen tulostus ennen kuin käyttäjä valitsee astiat:

```
Olet ratkaissut tilavuudet: 3 5 8.
Ratkaisematta on 1 2 4 6 7 9 10 11 12 13.
```

Ja peliin automaattinen lopetus kun kaikki tilavuudet on ratkaistu.