Demo 8 / 9.3

Tästä alkaa uusi demojakso ja nyt laskennallinen maksimi joka kerta on 8 tehtävää. Enää ei siis ole mahdollista kerätä yli 100% (eikä tarvitse :-). Ja ComTest tai JUnit testit joka tehtävään, joihin se on mielekkäästi mahdollista (merkitty (T)).

1*2. Täydennä edellisen demon tehtävän 7.7-8 vastaus MakiHyppy.java kunnon olio-ohjelmaksi niin, että seuraava testiohjelma toimii (huom! kaikkien "ominaisuuksien" ei tarvitse olla attribuutteina) (ks. Java-kuva)

```
public void kisa() {
  Kilpailija toni = new Kilpailija("Toni",3);
  Kilpailija matti = new Kilpailija("Matti",7);
  toni.tulosta();
  matti.tulosta();

  toni.setPituus(1,107);
  toni.setPituus(2,100);
  toni.setTuomari(2,1,19.0);
  toni.setTuomari(2,2,18.0);
  toni.setTuomari(2,3,19.5);
  toni.setTuomari(2,4,18.0);
  toni.setTuomari(2,5,20.0);

  matti.setPituus(1,125);
  matti.setTuomari(1,1,20.0);
  matti.setTuomari(1,2,20.0);
  matti.setTuomari(1,3,20.0);
  matti.setTuomari(1,4,20.0);
  matti.setPituus(2,109);
  matti.setTuomari(2,1,20.0);
  matti.setTuomari(2,2,20.0);
  matti.setTuomari(2,3,20.0);
  matti.setTuomari(2,4,20.0);

  toni.tulosta();
  matti.tulosta();

}

public static void main(String[] args)  {
  Makihyppy kisa = new Makihyppy();
  kisa.kisa();
}
```

3*. Muuta demotehtävän 7.4 vastaus Kirje.java sellaiseksi, että kahden taulukon sijasta onkin vain yksi taulukko, jonka alkioina on `Hinta` -luokka (ominaisuudet `hinta` ja `paino`). Kirjoita myös lisäksi funktio `double postimaksu(int paino)` . (T)

Toteuta vielä siten, että on yksi 2-ulotteinen taulukko (n x 2), jonka 1. sarakkeessa (sarake 0) on paino reaalilukuna ja toisessa sarakkeessa (sarake 1) hinta reaalilukuna. (T)

4*5. Luennolla tehtiin ohjelma Aanestys3.java johon kuului Vaihtoehdot.java ja Valinta.java.

Yksi vika on siinä, että `Vaihtoehdot` -luokassa on sekä tietorakenteen käsittely että käyttöliittymä samassa. Erota luokasta kaksi luokaa: `Vaihtoehdot` ja `AanestysLiittyma`, jossa edellinen hoitaa kaiken tietorakenteeseen liittyvät tehtävät ja jälkimmäinen kaiken käyttöliittymään liittyvät tehtävät. (T Vaihtoehdot)

6*. Kirjoita aliohjelma `matriisinSuurin`, joka palauttaa 2-ulotteisen matriisin suurimman alkion. (T)

7. Edellisen kerran tehtävän 7.8 vastauksen Rajat.java funktiota `summa` käyttäen kirjoita aliohjelma `matriisinSumma`, joka laskee 2-ulotteisen reaalilukumatriisin summan (älä kopioi, vaan kutsu funktiota, muista että matriisi on taulukko riveistä!). (T)

8. Ota selvää miten päivämäärä saadaan selville Javan kirjastokutsuilla. Kirjoita metodi `paivays`, jolla saadaan nykyinen päiväys selville Pvm2.java:n `Pvm2` tyyppiseen olioon. Muuta tarvittaessa myös luokan muodostaja ja `alusta` -metodi sellaiseksi, että jos alustuksessa ei anneta kaikkia arvoja, arvona käytetään nykypäiväystä puuttuville arvoille (pois saa jättää oikealta): (T, pitää muuttaa `Pvm2Test` -luokkaa hieman) Esimerkiksi:

```
Pvm2 tammi2005 = new Pvm2(1,1,2005)=>  1.1.2005
Pvm2 tammi2008 = new Pvm2(1,1); // =>  1.1.2008
Pvm2 maalis2008 = new Pvm2(1);  // =>  1.3.2008
Pvm2 tanaan = new Pvm2();       // =>  3.3.2008
                                // (testattu 3.3.2008)
```

B1. Edellisessä demossa tehtiin GraafinenAstia. Huonoa tuossa on vielä montakin asiaa. Yksi on kuitenkin se, että astian koodiin on sotkettu paljon grafiikkaa, jolla voisi olla muutakin käyttöä. Ota grafiikkaan liittyvä koodi ja tee siitä `Pylvas` -luokka, joka voidaan testata seuraavalla pääohjelmalla:

```
public static void main(String[] args) {
    Pylvas p5 = new Pylvas(1,5,0);
    Pylvas p8 = new Pylvas(1,8,2);
// muodostajalle pylvään leveys, korkeus ja väritetyn osan määrä
// väritetyn osan korkeudesta käytetään nimeä väli

    Window window = new Window();
    window.scale(0,0,5,8);
    p8.move(2,0,0);
    p8.setValiColor(Color.RED);

    window.add(p8);
    window.add(p5);
    window.showWindow();

    Syotto.kysy("Lisää 5 pylvään väliä");         p5.setVali(4);
    Syotto.kysy("Pienennä 8 pylvään korkeutta");  p8.setKorkeus(6);

}
```

Sitten tee uusi `GraafinenAstia`, joka toimii vanhalla testiohjelmalla, mutta sisältää vain tuon yhden pylvään. Suhde -vakio kuuluu `GraafinenAstia` -luokkaan.

B2-3. Käytä demon 7 vastauksia Esiintymat.java ja Kombinaatiot.java (luokat `Esiintymat` ja `Kombinaatiot`) toteuttaaksesi astiapelille AstiaPeli.java automaattisen lopun tarkistuksen. (Testi työläs, toteutetaan guru -tehtävässä)

G1-4. Malliohjelman vaiheen 5 luokassa `Jasenet` (Jasenet.java) on metodi `anna` joka palauttaa `Jasen` -luokan viitteen (Jasen.java). Tässä on se vika, että luokassa `Naytto` (Naytto.java) olevassa metodissa `tulosteet` voitaisiin muuttaa `Jasen` -luokan olion arvoa (ks. myös Kerho.java):

```
for (int i=0; i<kerho.getJasenia(); i++) {
  Jasen jasen = kerho.anna_jasen(i);
  jasen.vastaa_aku_ankka();  // esim. näin voitaisiin muuttaa
  tulosta("Jäsen nro: " + i);
  tulosta(System.out,jasen);
  tulosta("");
}
```

Mieti ja toteuta millainen pitäisi olla metodin `anna` paluuarvo, jotta metodia käyttävät eivät voisi mitenkään "pilata" palautetun jäsenen arvoa. Vinkki: muista rajapinnat.

G5-7. Luokka `AstiaPeli` (tai B2-3 kohdassa tehty luokka) on vaikea testata koska käyttöliittymä ja datan käsittely on samassa luokassa. Erota erikseen käyttöliittymäluokka ja datankäsittelyluokka ja tee sitten ComTest tai `JUnit` testit kaikille muille luokille paitsi käyttöliittymä-luokalle (saa sillekin tehdä).