Olio-ohjelmoinnin harjoitustyön dokumentti
Jyri Lehtonen (72039)
Taneli Tuovinen (67160)

1 Tehtävän kuvaus ja analysointi

1.1 Tehtävänanto

Tee luokka, jolla mallinnetaan sarjaan kytkettyjä kondensaattoreita. Sarjaan voidaan kytkeä yksi tai useampi yksittäinen kondensaattori, jolloin saadaan yhdistetty kondensaattori. Yksittäisestä kondensaattorista tiedetään sen kapasitanssi, valmistaja ja valmistuspäivämäärä. Toteuta yhdistetylle kondensaattorille havainnointi- ja muutosoperaatioiden lisäksi operaatio, joka palauttaa yhdistetyn kondensaattorin kapasitanssin.
Yhdistetyn kondensaattorin kapasitanssin käänteisarvo on sen yksittäisten kondensaattorien kapasitanssien käänteisarvojen summa. Esimerkiksi kahden sarjaan kytketyn kondensaattorin (C1, C2) kapasitanssi on 1/C = 1/C1 + 1/C2.

1.2 Tehtävänannon analysointi

Tehtävänannossa määrätään luomaan ohjelma, jolla pystytään mallintamaan sarjaan kytkettyjä kondensaattoreita. Tähän sarjaan voidaan kytkeä 1 – N yksittäinen kondensaattori, jonka lopputuloksena saadaan kyseinen yhdistetty kondensaattori. Yhdistetylle kondensaattorille on tehtävä funktio- ja proseduuri metodi. Lisäksi on tehtävä myös metodi, joka laskee yhteen yhdistettyjen kondensaattoreiden kapasitanssin. Rajaa sarjaan kytketyistä kondensaattoreista ei ole annettu tehtävänannossa, joten oletetaan että sarjaan voidaan kytkeä kaikki kondensaattorit joille näin halutaan tehdä.

Kondensaattorin ominaisuuksiksi luetellaan sen kapasitanssi, valmistaja ja valmistuspäivämäärä.

Ohjelman syötteistä ja tulosteista ei anneta tietoa tehtävänannossa. Oletetaan, että syötteeksi riittävät testaustilanteessa ohjelmoijan suoraan kooditasolla antamat syötteet ja tulosteeksi kuvaruudulle tulevat tulosteet.

2 Ratkaisuperiaate

Varsinaista ongelmanratkaisua varten kirjoitetaan kaksi luokkaa: `Kondensaattorit` ja `Yhdistys`. Näiden lisäksi kirjoitetaan kolmas `Testi`-luokka kahden edellisen luokan testausta varten.

`Kondensaattorit`-luokka mallintaa yhtä kondensaattoria. Tehtävänannon mukaiset ominaisuudet saadaan toteutettua valitsemalla instanssimuuttujiksi seuraavat ominaisuudet: sarjanumero (`int`), kapasitanssi (`double`), valmistaja (`String`) sekä valmistuspäivämäärä (`Date`) Alustaja saa kaikki nämä arvot argumenttina.

Jokaista `Kondensaattorit`-luokan attribuuttia varten kirjoitetaan havainnoija. Ylimääräinen havainnoija metodi `annaIkä()` laskee annetun kondensaattori olion iän sen päivämäärän perusteella. Muuttajametodit jätetään toteuttamatta, koska ratkaisun kannalta ne eivät ole olennaisia. Tällainen metodi voisi olla esimerkiksi `asetaSarjanumero(uusiNumero)`, joka muuttaisi kondensaattorin sarjanumeron toiseksi.

`Yhdistys`-luokan keskeisimmäksi tietosisällöksi valitaan kokoelma `Kondesaattorit`-tyyppisiä olioita (`ArrayList<Kondensaattorit>`). Lisäksi on vielä `double` tyyppinen muuttuja, johon tallennetaan yhdistettyjen kondensaattorien kapasitanssi. Muita attribuutteja ei ole.

Yhdistykseen voi lisätä kondensaattoreita omalla metodillaan, ja kondensaattoreita voi poistaa sarjanumeron perusteella. Yhdistyksen tilaa voidaan havainnoida tiedustelemalla yhdistettävien kondensaattorien lukumäärää ja mitä kondensaattori olioita yhdistyksessä on sisällä.

Testauksen hoitavassa luokassa `Testaus` on erillisinä metodeina `Kondensaattorit`-luokan testaus sekä `Yhdistys`-luokan testaus. Syötteet annetaan suoraan ohjelmakoodin sisällä eikä niitä pyydetä käyttäjältä. Tarvittavat testauksen tulosteet tulostetaan ruudulle `system.out.print` käskyjen avulla. Luokkien väliset asiakassuhteet on esitetty kuviossa 1.

Kuvio 1: Systeemin asiakaskaavio

3 Ohjelman ja sen osien kuvaaminen

3.1 Luokka Kondensaattorit

Luokka kuvaa yhtä ratkaisun kondensaattoria.

Ilmentymämuuttujat

```
/**
 * Kondensaattorin sarjanumero
 */
private int sarjanumero;

/**
 * Kondensaattorin kapasitanssi
 */
private double kapasitanssi;

/**
 * Kondensaattorin valmistajan nimi
 */
private String valmistaja;

/**
 * Kondensaattorin valmistuspäivämäärä
 */
private Date päivämäärä = new Date();
```

Alustajat

```
/**
 * KONSTRUKTORI
 * alustaa uuden Kondensaattorit olion, jolla on sarjanumero,
 * kapasitanssi, valmistaja, valmistus päivämäärä.
 *
 * AE: (c != null && d != null)
 */
public Kondensaattorit(int a, double b, String c, Date d) {
 this.sarjanumero = a;
 this.kapasitanssi = b;
 this.valmistaja = c;
 this.päivämäärä = d;

}
```

Havainnoijat

```
/**
 * Palauttaa kondensaattorin sarjanumeron integerinä.
 * AE: true
 */
public int annaSarjanumero() {
           return sarjanumero;
           }
/**
 * Palauttaa kondensaattorin kapasitanssin doublenä
 * AE: true
 */
public double annaKapasitanssi() {
           return kapasitanssi;
           }
/**
 * Palauttaa kondensaattorin valmistajan Stringinä.
 * AE: true
 */
public String annaValmistaja() {
           return valmistaja;
           }
/**
 * Palauttaa kondensaattorin luontipäivän Date olion avulla.
 * AE: true
 */
public Date annaPäivämäärä() {
           return päivämäärä;
           }

/**
 * Palauttaa kondensaattorin iän double lukuna.
 * AE: (nyk != null && päivämäärä != null)
 *
 */
public double annaIkä() {
 Date nyk = new Date();
 double sekunttiaVuodessa = 31557600.0;

 // getTime() antaa päivämäärän millisekunneissa.
 //[annettu päivämäärä] - January 1, 1970, 00:00:00 GMT
 double ikä = (nyk.getTime() - päivämäärä.getTime()) /
               sekunttiaVuodessa / 1000;

// Laskee iän 2 desimaalin tarkkuudella.
//(int) muuttaa luvun integeriksi ja (double) double luvuksi
return ((int)ikä + ((double)(Math.round((ikä - (int)ikä)*100))/100));
}
```

3.2 Luokka Yhdistetty

Ilmentymämuuttujat

```
//Säiliö johon kondensaattoreita kerätään yhdistykseksi
private ArrayList<Kondensaattorit> yhdistyssäiliö = new ArrayList();
//Yhdistettyjen kondensaattorien kapasitanssi.
private double yhdistettyKapasitanssi;
```

Alustajat

```
/*
 * KONSTRUKTORI
 */
public Yhdistys() {   }
```

Havainnoijat

```
  /**
  * Näytä säiliön sisällön lukumäärä
  */       public int näytäLukumäärä() {
            return yhdistyssäiliö.size();
            }

/**
 * Näytä mitkä oliot ovat säiliössä sisällä
 */
public void näytäSisältö() {
           for (int i=0; i<yhdistyssäiliö.size(); i++) {
                      Kondensaattorit kond = yhdistyssäiliö.get(i);

                      System.out.println("Sarjanumero: "
                                +kond.annaSarjanumero());
                      System.out.println("Kapasitanssi: "
                                +kond.annaKapasitanssi());
                      System.out.println("Valmistaja: "
                                +kond.annaValmistaja());
                      System.out.println("Valmistettu: "
                                +kond.annaPäivämäärä());
                      System.out.println("");

           }
}

/**
* Yhdistää eri kondensaattorien kapasitanssit ja
* laskee sarjaan kytketyn kondensaattorin (c1, c2)
* kapasitanssit kaavalla (1/c = 1/c1 + 1/c2).
*
* AE: yhdistyssäiliö != null && kond != null
*/
public void yhdistä() {
  double muuttuja=0.0;
  for (int i=0; i<yhdistyssäiliö.size(); i++) {
            Kondensaattorit kond = yhdistyssäiliö.get(i);
            muuttuja+= 1/(kond.annaKapasitanssi());
   }

  yhdistettyKapasitanssi=muuttuja;
  System.out.println(yhdistettyKapasitanssi);
}
```

```
    /**
     * Etsitään yhdistyssäiliöstä Kondensaattori
     * sen sarjanumeron avulla, ja palautetaan
     * sen indeksi (esim. poistoa varten tai
     * vaan tietoa onko sellainen säiliössä)
     *
     * AE: yhdistyssäiliö != null & kond != null
     */
    public int etsiSäiliöstä(int a) {
              int palautin=-1;
              int etsittävänSN=a;
              int tutkittavanSN=0;

              for (int i=0; i<yhdistyssäiliö.size(); i++) {
              Kondensaattorit kond = yhdistyssäiliö.get(i);
                        tutkittavanSN=kond.annaSarjanumero();

                        if(tutkittavanSN == a) {
                                  palautin=(yhdistyssäiliö.size()
                                  - (yhdistyssäiliö.size() -i));;

                        }
              }
              return palautin;

    }
```

Proseduurit

```
    /**
     * Lisää säiliöön kondensaattorin a
     * AE: true
     */
    public void lisääKondensaattori(Kondensaattorit a) {
              yhdistyssäiliö.add(a);
    }

    /**
     * Poistaa säiliöstä kondensaattorin sen
     * sarjanumeron perusteella.
     *
     * AE: (yhdistyssäiliö != null && h != null)
     */
    public void poistaKondensaattori(int numero) {
Iterator<Kondensaattorit> iter = yhdistyssäiliö.iterator();

while (iter.hasNext()) {
    Kondensaattorit h = iter.next();
    if (numero == h.annaSarjanumero()) {
        iter.remove();
              }
     }
    }

    /**
      * Tyhjentää säiliön
      */
    public void tyhjennäSäiliö() {
              yhdistyssäiliö.clear();
    }
```

3.3 Luokka Testi

Testausluokka, jolla testataan `Kondensaattorit` ja `Yhdistys` toimintaa.
Sisältää vain staattisia metodeja.

Staattiset metodit

```
/**
 * Testaussysteemin pääohjelma.
 */
public static void main(String[] args) {
testaaKondensaattorit();
testaaYhdistys();
}


/**
 * Testataan luokan Kondensaattorit metodeja.
 */
public static void testaaKondensaattorit()



/**
 * Testataan luokan Yhdistys metodeja.
 */
public static void testaaYhdistys()
```

4 Testausjärjestelyt

Testaus suoritetaan kaksivaiheisesti siten, että ensin testataan `Kondensaattorit`-luokan toiminta ja sitten sitä hyödyntävän `Yhdistys`-Luokan toiminta.

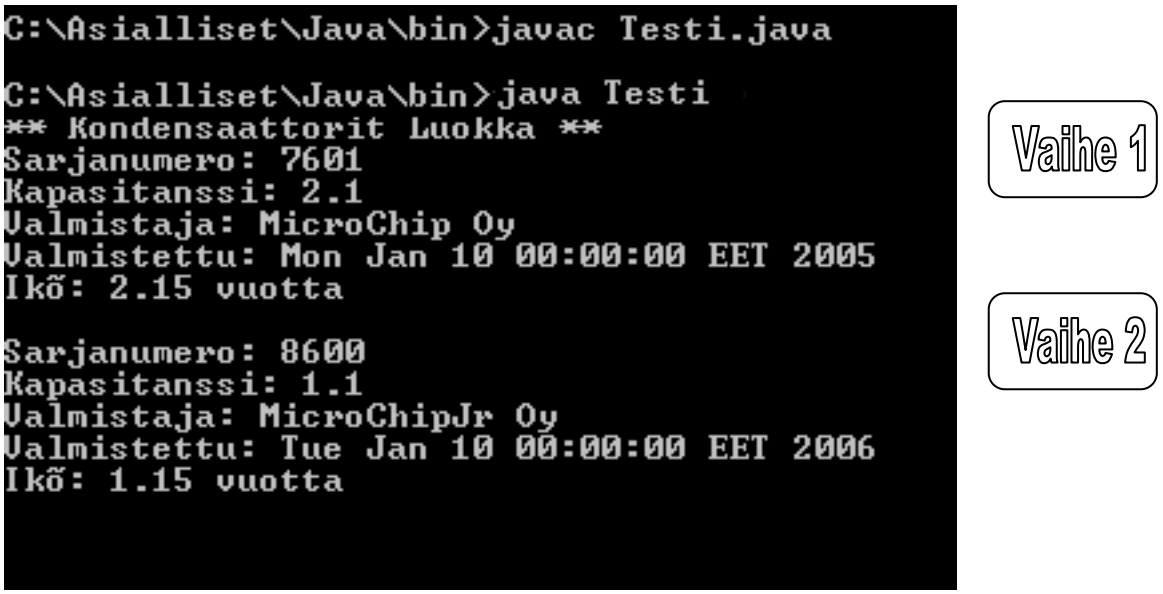
4.1 Kondensaattorit-luokan testaus

Testaussuunnitelma

1. alustuksen testaus *(luodaan kaksi kondensaattorioliota ja alustetaan ne)*
2. havainnoijien testaus *(kutsutaan jokaista havainnoijaa kerran ja tulostetaan niiden palauttamat arvot)*

Havainnot

Testattaessa ruudulle tulostuu seuraavaa (vaiheet on merkitty kuvan viereen):



Havaitaan testauksen etenevän suunnitelman mukaisesti.

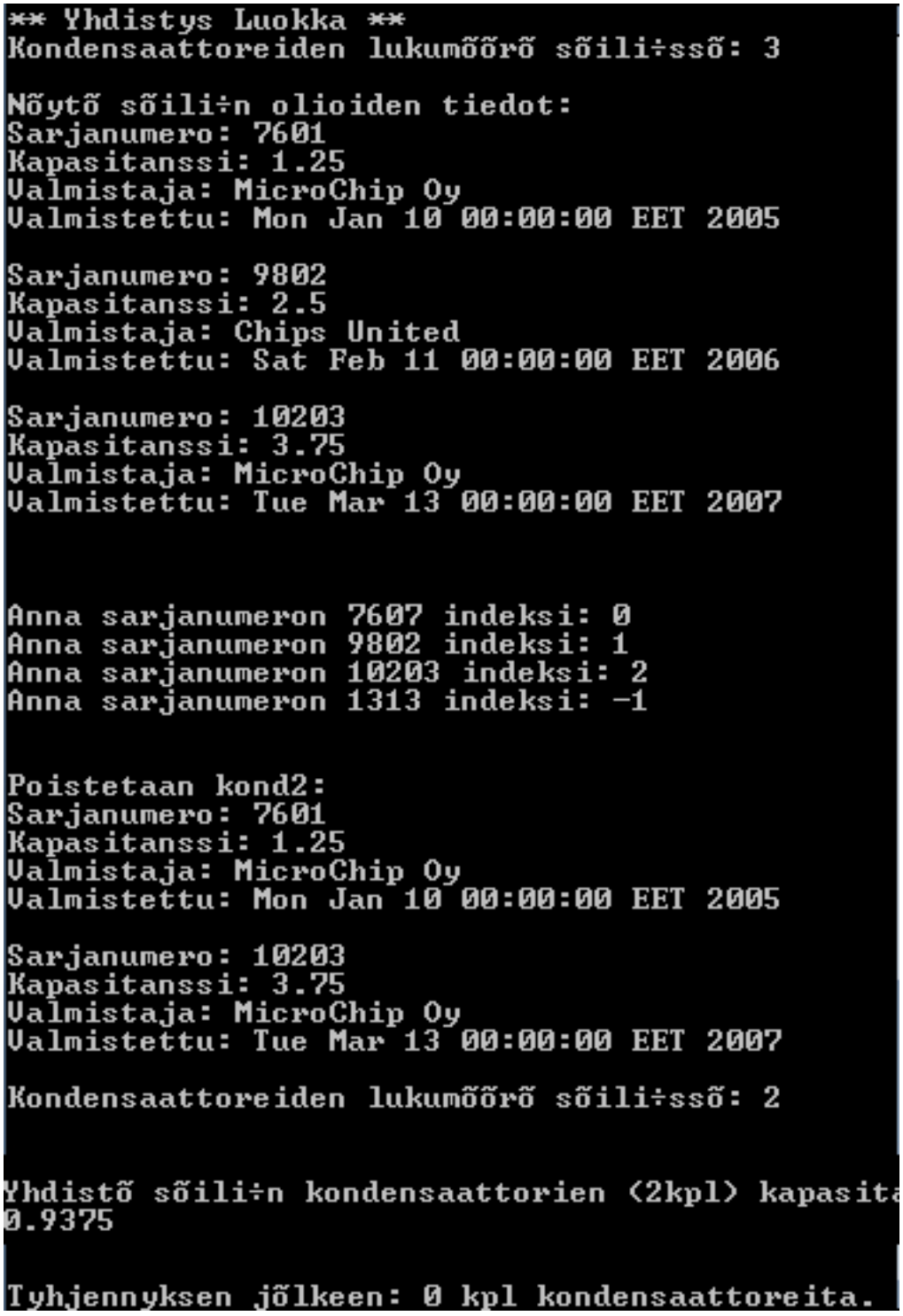
4.2 Kondensaattorit-luokan testaus

Testaussuunnitelma

1. luodaan kondensaattorit olioita avustamaan testausta *(3 kpl)*
2. Katsotaan mitkä oliot ovat säiliössä sisällä.
3. Metodin testaus *(Etsitään säiliöstä kondensaattori sen sarjanumeron perusteella).*
4. Metodin testaus(*Suoritetaan yhden poisto sarjanumeron avulla)*
5. Metodin testaus(*Lasketaan kahden jäljellä olevan yhdistetyn kapasitanssi.)*
6. Metodin testaus*(Tyhjennetään säiliö).*

Havainnot

Testattaessa ruudulle tulostuu seuraavaa (vaiheet on merkitty kehystetyin tekstein eri kirjasimella):



Vaihe 1

Vaihe 2

Vaihe 3

Vaihe 4

Vaihe 5

Vaihe 6

5 Liitteet

Tehtäväpaperi

Tehtävän anto on määritelty kohdassa 1.1,
tämän dokumentin palautus tapahtuu sähköisesti,
jolloin aitoa paperia ei voida liittää mukaan.

Ohjelmalistaukset

```
import java.util.Date;
public class Kondensaattorit{


/*
 * ILMENTYMÄMUUTUTJAT
 */

           /**
            * Kondensaattorin sarjanumero
            */
           private int sarjanumero;

           /**
            * Kondensaattorin kapasitanssi
            */
           private double kapasitanssi;

           /**
            * Kondensaattorin valmistajan nimi
            */
           private String valmistaja;

           /**
            * Kondensaattorin valmistuspäivämäärä
            */
           private Date päivämäärä = new Date();



           /**
            * KONSTRUKTORI
            * alustaa uuden Kondensaattorit olion, jolla on sarjanumero,
            * kapasitanssi, valmistaja, valmistus päivämäärä.
            *
            * AE: (c != null && d != null)
            */
           public Kondensaattorit(int a, double b, String c, Date d) {
                      this.sarjanumero = a;
                      this.kapasitanssi = b;
                      this.valmistaja = c;
                      this.päivämäärä = d;
           }
```

```
        /*
         * HAVAINNOIJAT
         */

        /**
         * Palauttaa kondensaattorin sarjanumeron integerinä.
         * AE: true
         */
        public int annaSarjanumero() {
                  return sarjanumero;
        }

        /**
         * Palauttaa kondensaattorin kapasitanssin doublenä
         * AE: true
         */
        public double annaKapasitanssi() {
                  return kapasitanssi;
        }

        /**
         * Palauttaa kondensaattorin valmistajan Stringinä.
         * AE: true
         */
        public String annaValmistaja() {
                  return valmistaja;
        }
        /**
         * Palauttaa kondensaattorin luontipäivän Date olion avulla.
         * AE: true
         */
        public Date annaPäivämäärä() {
                  return päivämäärä;
        }

        /**
         * Palauttaa kondensaattorin iän double lukuna.
         * AE: (nyk != null && päivämäärä != null)
         *
         */
        public double annaIkä() {
         Date nyk = new Date();
         double sekunttiaVuodessa = 31557600.0;

         // getTime() antaa päivämäärän millisekunneissa.
         //[annettu päivämäärä] - January 1, 1970, 00:00:00 GMT
            double ikä = (nyk.getTime() - päivämäärä.getTime()) /
        sekunttiaVuodessa / 1000;

        // Laskee iän 2 decimaalin tarkkuudella.
        //(int) muuttaa luvun integeriksi ja (double) double luvuksi
        return ((int)ikä + ((double)(Math.round((ikä - (int)ikä)*100))/100));
        }
}
```

```
import java.util.*;
public class Yhdistys {



           //ILMENTYMÄMUUTTUJAT
           /*
            *Säiliö johon kondensaattoreita kerätään yhdistykseksi
            */
           private ArrayList<Kondensaattorit> yhdistyssäiliö = new ArrayList();
           //Yhdistettyjen kondensaattorien kapasitanssi.
           private double yhdistettyKapasitanssi;


           /*
            * KONSTRUKTORI
            */
           public Yhdistys() {   }



           /*
            * PROSEDUURIT
            */

           /**
            * Lisää säiliöön kondensaattorin a
            * AE: true
            */
           public void lisääKondensaattori(Kondensaattorit a) {
                     yhdistyssäiliö.add(a);
           }

           /**
            * Poistaa säiliöstä kondensaattorin sen
            * sarjanumeron perusteella.
            *
            * AE: (yhdistyssäiliö != null && h != null)
            */
           public void poistaKondensaattori(int numero) {
         Iterator<Kondensaattorit> iter = yhdistyssäiliö.iterator();

         while (iter.hasNext()) {
             Kondensaattorit h = iter.next();
             if (numero == h.annaSarjanumero()) {
                 iter.remove();
                       }
              }
           }

            /**
             * Tyhjentää säiliön
             */
           public void tyhjennäSäiliö() {
                     yhdistyssäiliö.clear();
           }
```

```
/**
 * Etsitään yhdistyssäiliöstä Kondensaattori
 * sen sarjanumeron avulla, ja palautetaan
 * sen indeksi (esim. poistoa varten tai
 * vaan tietoa onko sellainen säiliössä)
 *
 * AE: (yhdistyssäiliö != null & kond != null)
 */
public int etsiSäiliöstä(int a) {
           int palautin=-1;
           int etsittävänSN=a;
           int tutkittavanSN=0;

           for (int i=0; i<yhdistyssäiliö.size(); i++) {
           Kondensaattorit kond = yhdistyssäiliö.get(i);
                      tutkittavanSN=kond.annaSarjanumero();

           if(tutkittavanSN == a) {
   palautin=(yhdistyssäiliö.size()  - (yhdistyssäiliö.size() -i));;

                      }
           }
           return palautin;
}



/*
 * HAVAINNOIJAT
 */

 /**
  * Näytä säiliön sisällön lukumäärä
  */
public int näytäLukumäärä() {
           return yhdistyssäiliö.size();
}

/**
 * Näytä mitkä oliot ovat säiliössä sisällä
 * AE: (yhdistyssäiliö != null && kond != null)
 */
public void näytäSisältö() {
for (int i=0; i<yhdistyssäiliö.size(); i++) {
           Kondensaattorit kond = yhdistyssäiliö.get(i);
System.out.println("Sarjanumero: "+kond.annaSarjanumero());
System.out.println("Kapasitanssi: "+kond.annaKapasitanssi());
System.out.println("Valmistaja: "+kond.annaValmistaja());
System.out.println("Valmistettu: "+kond.annaPäivämäärä());
           System.out.println("");

           }
}
```

```
	/**
	 * Yhdistää eri kondensaattorien kapasitanssit ja
	 * laskee sarjaan kytketyn kondensaattorin (c1, c2)
	 * kapasitanssit kaavalla (1/c = 1/c1 + 1/c2).
	 *
	 * AE: yhdistyssäiliö != null && kond != null
	 */
	public void yhdistä() {
		double muuttuja=0.0;

		for (int i=0; i<yhdistyssäiliö.size(); i++) {
			Kondensaattorit kond = yhdistyssäiliö.get(i);
			muuttuja+= 1/(kond.annaKapasitanssi());
		}

		yhdistettyKapasitanssi=muuttuja;
		System.out.println(yhdistettyKapasitanssi);
	}
}
```

```
import java.util.Date;
public class Testi {

     /**
     * Testaussysteemin pääohjelma.
     */
    public static void main(String[] args) {
        testaaKondensaattorit();
        testaaYhdistys();
    }



    /**
     * Testataan luokan Kondensaattorit metodeja.
     */
    public static void testaaKondensaattorit() {

// Luodaan kolme Kondensaattori oliota, ja annetaan niille parametrit.
Kondensaattorit esimerkki =
new Kondensaattorit(7601, 2.1, "MicroChip Oy", new Date(2005 - 1900, 0, 10));

Kondensaattorit esimerkki2 =
new Kondensaattorit(8600, 1.1, "MicroChipJr Oy", new Date(2006 - 1900, 0, 10));

// Testataan kondensaattorio olion havainnoijat.
System.out.println("** Kondensaattorit Luokka **");
           System.out.println("Sarjanumero: " + esimerkki.annaSarjanumero());
           System.out.println("Kapasitanssi: " + esimerkki.annaKapasitanssi());
           System.out.println("Valmistaja: " + esimerkki.annaValmistaja());
           System.out.println("Valmistettu: " + esimerkki.annaPäivämäärä());
           System.out.println("Ikä: " + esimerkki.annaIkä() + " vuotta");
                                System.out.println("");

           System.out.println("Sarjanumero: " + esimerkki2.annaSarjanumero());
           System.out.println("Kapasitanssi: " + esimerkki2.annaKapasitanssi());
           System.out.println("Valmistaja: " + esimerkki2.annaValmistaja());
           System.out.println("Valmistettu: " + esimerkki2.annaPäivämäärä());
                     System.out.println("Ikä: " + esimerkki2.annaIkä() + "
vuotta");
                                System.out.println("");
           }


           /**
            * Testataan luokan Yhdistys metodeja.
            */
           public static void testaaYhdistys() {


           //Luodaan kolme Kondensaattori oliota, ja annetaan niille parametrit.
           Kondensaattorit kondi1 =
new Kondensaattorit(7601, 1.25, "MicroChip Oy", new Date(2005 - 1900, 0, 10));
           Kondensaattorit kondi2 =
new Kondensaattorit(9802, 2.50, "Chips United", new Date(2006 - 1900, 1, 11));
           Kondensaattorit kondi3 =
new Kondensaattorit(10203, 3.75, "MicroChip Oy", new Date(2007 - 1900, 2, 13));
                                System.out.println("");
                                System.out.println("");
                                System.out.println("");
```

```
            System.out.println("** Yhdistys Luokka **");
                        Yhdistys yhdistä = new Yhdistys();
                        yhdistä.lisääKondensaattori(kondi1);
                        yhdistä.lisääKondensaattori(kondi2);
                        yhdistä.lisääKondensaattori(kondi3);

            System.out.println("Kondensaattoreiden lukumäärä säiliössä: " +
                                    yhdistä.näytäLukumäärä());
                        System.out.println("");
                        //Näytä mitkä kondensaattorit ovat säiliössä:
                        System.out.println("Näytä säiliön olioiden tiedot: ");
                        yhdistä.näytäSisältö();
                                    System.out.println("");
                                    System.out.println("");


//Etsi sarjanumeron perusteella, -1 tarkoittaa ei löydy.
                        System.out.println("Anna sarjanumeron 7607 indeksi:
                                                "+yhdistä.etsiSäiliöstä(7601));
                        System.out.println("Anna sarjanumeron 9802 indeksi:
                                               "+yhdistä.etsiSäiliöstä(9802));
                        System.out.println("Anna sarjanumeron 10203 indeksi:
                                               "+yhdistä.etsiSäiliöstä(10203));
                        System.out.println("Anna sarjanumeron 1313 indeksi:
                                               "+yhdistä.etsiSäiliöstä(1313));
                                    System.out.println("");
                                    System.out.println("");

                        yhdistä.poistaKondensaattori(9802);
                        System.out.println("Poistetaan kond2: ");
                        yhdistä.näytäSisältö();
            System.out.println("Kondensaattoreiden lukumäärä säiliössä:
                                     "+yhdistä.näytäLukumäärä());


                                    System.out.println("");
                                    System.out.println("");

                        //Yhdistetään säiliössä olevat kapasitanssi.
System.out.print("Yhdistä säiliön kondensaattorien (2kpl) kapasitanssit: \n");
                        yhdistä.yhdistä();

                                    System.out.println("");
                                    System.out.println("");


                        /*Tyhjennä säiliö*/
                        yhdistä.tyhjennäSäiliö();
                        System.out.println("Tyhjennyksen jälkeen:
                        "+yhdistä.näytäLukumäärä() +" kpl kondensaattoreita.");
            }
}
```

Ohjelman käyttöohje

Ohjelma käännetään komentamalla `javac Testi.java` ja ajetaan komentamalla `java Testi`.