T-106.1250 Informaatioverkostojen ohjelmointikurssi

T-106.2001 Informaatioverkostot: Studio 1

Tentti 19.12.2011

Kirjoita jokaiseen vastauspaperiin vähintään päivämäärä, kurssin nimi, nimesi, opiskelijanumerosi ja allekirjoituksesi. Apuvälineinä ei saa käyttää kuin muistiinpanovälineitä.

**1. Kysymykset (8 pistettä)**

Vastaa seuraaviin kysymyksiin.

a) Mikä on kutsupino (*engl. call stack*)? (1 piste)

b) Kerro kaksi olennaista eroa kiinteästi hinnoitellun ja aika- ja materiaaliperusteisesti hinnoittelun ohjelmointiprojektin välillä. (2 pistettä)

c) Mitä ylikirjoittaminen tarkoittaa ja mitä sillä saavutetaan? (2 pistettä)

d) Miten poikkeuksia käsitellään Javassa? (3 pistettä)

**2. Väittämät (8 pistettä)**

Perustele ovatko seuraavat väittämät totta vai eivät. Pisteet saa perustelusta.

a) Olkoon määritelty luokka `Talo`. Tällöin seuraavat ohjelmat tuottavat ajettaessa saman tulosteen.

```
1   public class Testi {
2
3       public static void main(String[] args) {
4           Talo talo = new Talo("Muumitalo");
5           System.out.println(talo);
6       }
7
8   }
```

```
1   public class Testi {
2
3       public static void main(String[] args) {
4           Talo talo = new Talo("Muumitalo");
5           System.out.println(talo.toString());
6       }
7
8   }
```

b) Ainoa tapa tehdä oma tapahtumien kuuntelijan on toteuttaa `ActionListener` rajapinta.

c) Javan valmiiden pakkauksien lisäksi (kuten `java.util`) on mahdollista luoda myös omia pakkauksia.

d) Enumeraatioluokka (perii `java.lang.Enum`) on poikkeuksellinen Javan säännöille ja se voi periä myös jonkun toisen luokan.

**3. Mitä tämä koodi tulostaa? (8 pistettä)**

Kirjoita näkyviin tuloste, jonka `Tulostaja`-luokan ajaminen aiheuttaa.

```
1   import java.util.ArrayList;
2   import java.util.Iterator;
3
4   public class Tulostaja {
5       private ArrayList<String> lista;
6       private static int JUURI = 2;
7
8       public Tulostaja(int pituus) {
9           this.lista = new ArrayList<String>();
10
11          for (int i = 0; i < pituus; i++) {
12              this.lista.add((new Integer(JUURI+i)).toString());
13          }
14          JUURI += 1;
15      }
16
17      public ArrayList<String> AnnaLista() {
18          return this.lista;
19      }
20
21      public static void Tulostin1(Tulostaja t1, Tulostaja t2) {
22          Iterator<String> it1 = t1.AnnaLista().iterator();
23          Iterator<String> it2 = t2.AnnaLista().iterator();
24
25          while(it1.hasNext() && it2.hasNext()) {
26              System.out.println(it1.next());
27              System.out.println(it2.next());
28          }
29      }
30
31      public void Tulostin2(Tulostaja tulos) {
32          Iterator<String> it = tulos.AnnaLista().iterator();
33
34          while(this.lista.iterator().hasNext() && it.hasNext()) {
35              if (this.lista.iterator().hasNext())
36                  System.out.println(this.lista.iterator().next());
37              if (it.hasNext())
38                  System.out.println(it.next());
39          }
40      }
41
42      public void Tulostin3() {
43          Iterator<String> it = this.lista.iterator();
44          while (it.hasNext()) {
45              System.out.println(it.next());
46              JUURI -= 1;
47          }
48      }
49
50      public static void main(String[] args) {
51          Tulostaja t = new Tulostaja(3);
52          Tulostaja ti = new Tulostaja(6);
53          t.Tulostin3();
54          System.out.println("\n---\n");
55          Tulostin1(t, ti);
56          System.out.println("\n---\n");
57          t.Tulostin2(ti);
58          System.out.println("Ohjelman suoritus loppui!");
59      }
60  }
```

**4. Määrittelyt (8 pistettä)**

Vastaa kumpaankin kohtaan esseemuotoisella lyhyellä (noin 1 sivu, joka toiselle riville) vastauksella. Esimerkkien ja/tai selventävien kuvien käyttö on varsin sallittua.

a) MVC-malli.
Selitä, mitä MVC-mallilla tarkoitetaan. Miten se toimii? Miksi ohjelmia tulisi kirjoittaa mallin mukaisesti? Mitä hyötyjä ja haittoja mallin käytöstä on?

b) Rekursio.
Selitä, mitä rekursio tarkoittaa. Rekursio tahattomana on yleensä huono asia ohjelmakoodissa. Sitä voidaan kuitenkin käyttää tarkoituksenmukaisesti hyödyksi. Millaisia hyötyjä rekursiosta voi olla? Millaisia haittoja ja vaaroja piilee rekursion käytössä?