Kevät 2009 29.5.2008

Tentissä saa käyttää:

- API-dokumentaatiota
- JCreatoria
- Tekstieditoria kuten Notepad, Wordpad, Word tms.
- Komentokehoitetta
- Moodlen OHII-sivustoa
- X-asemalle tentin ajaksi luomaanne OHIILT-kansiota
- Kynää ja paperia
- Laskinta

Tentissä ei saa käyttää:

- Moodlea ja yllä mainittua X-asemalla olevaa kansiota lukuunottamatta mitään verkkojuttuja
- JCreatoria lukuun ottamatta muita Java-editoreja tms. (esim. NetBeans, JBuilder, BlueJ)
- Mitään muita kyseiseltä työasemalta löytyviä tiedostoja.

Jos kuitenkin haluaisit käyttää jotain välinettä, jota ei ole tässä erikseen mainittu, varmista asia ensin tentin valvojilta.

Sanalliset vastaukset saavat olla .txt, .rtf tai .doc(x) -muodossa. Ohjelmointitehtävien vastaukset kirjoitetaan niin ikään tekstimuodossa asianmukaisesti nimettyyn .java-tiedostoon. Käännösyksikössä voi olla useampia luokkia. **Yhden tehtävän vastaus tallennetaan aina yhteen tiedostoon** (kts. palautuslinkit Moodlesta). Jos sijoitat ohjelman tekovaiheessa luokkia eri tiedostoihin, kopioi ne kaikki lopuksi yhteen palautettavaan tiedostoon. Tiedostojen säilytyspaikkana on tentin ajan yllä mainittu X-asemalle sijoitettu oma hakemisto.

Jokaisesta palautettavasta tiedostosta täytyy löytyä palauttajan nimi!

Ohjelmointitehtävissä olisi tarkoituksena tuottaa Java-kielinen tiedosto, joka suorittaa tehtävänannossa määrätyt tehtävät oikein. Jos et saa tehtävää täydellisesti tehtyä, yritä kuitenkin niin pitkälle kuin osaat. Jos et osaa jotain kohtaa, tee ne jotka osaat. Jos se, jota et osannut, vaikuttaa muihin osiin, sovella sopivalla tavalla. =) Vaikka tiedosto ei olisi edes kääntyvässä muodossa, voi siitä saada pisteitä.

Tiedostot palautetaan kunkin tehtävän numeron mukaisen linkin kautta Moodleen.

Ohjelmointitehtäviä täytyy kommentoida! Kerro kommenteilla erityisesti olio-ohjelmointiin liittyvistä seikoista ja tekemistäsi ratkaisuista (määreet, periytyminen jne.). Selittävät kommentit vaikuttavat arvosteluun siinä missä itse koodikin. Hyvällä kommentoinnilla voi myös pelastaa pienet ohjelmointivirheet.

Kun olet valmis, nosta käsi ylös tai herätä valvojien huomio jollain muulla hiljaisella tavalla. Tentistä "ulos kirjautuminen" hoidetaan yhdessä valvojan kanssa.

Onnea tenttiin!

Tehtävä 1 (15 p)

Esittele mahdollisimman kattavasti seuraavat Java-kieliset määreet. Selvitä niiden merkitys, käyttötavat ja -tilanteet sekä niistä aiheutuvat vaikutukset

a) private
b) abstract
c) final

Tehtävä 2 (6 p)

Alla olevan pääohjelman suoritus tulostaa seuraavaa:

```
Ritva palvelee seuraavaksi numeroa 1 palvelupisteessä 1
Tiina palvelee seuraavaksi numeroa 100 palvelupisteessä 2
Jaana palvelee seuraavaksi numeroa 2 palvelupisteessä 3
Ritva palvelee seuraavaksi numeroa 101 palvelupisteessä 1
Jaana palvelee seuraavaksi numeroa 3 palvelupisteessä 3
```

Laadi luokka Asiakaspalvelija. Luokan täytyy olla sellainen, että se toimisi vaikka asiakaspalvelijoiden nimiä ja kutsujen järjestystä muutettaisi. Metodi seuraavaYksityisasiakas siis "kuuluttaa" aina seuraavaa numeroa väliltä 1-99 ja seuraavaYritysasiakas seuraavaa numeroa väliltä 100-199.

```
public static void main(String[] args) {
   Asiakaspalvelija a = new Asiakaspalvelija("Ritva",1);
   Asiakaspalvelija b = new Asiakaspalvelija("Tiina",2);
   Asiakaspalvelija c = new Asiakaspalvelija("Jaana",3);
   System.out.println(a.seuraavaYksityisasiakas());
   System.out.println(b.seuraavaYritysasiakas());
   System.out.println(c.seuraavaYksityisasiakas());
   System.out.println(a.seuraavaYritysasiakas());
   System.out.println(c.seuraavaYksityisasiakas());
}
```

Kommentoi luokka huolella ja sisällytä luokan alkuun lyhyt selkokielinen kuvaus siitä, mikä luokan merkitys ja käyttötarkoitus oikeastaan on.

Tehtävä 3 (9 p)

Alla oleva *lueLuvutTiedostosta*-metodi lukee tekstitiedossa olevia lukuja ja sijoittaa ne kokonaislukutaulukkoon jonka se lopuksi palauttaa. Mitä poikkeuksia tämä metodi voi nostaa ja mistä syystä ne nousevat? Vastaa kysymykseen kommenttimerkkien sisällä palauttamasi kooditiedoston alussa.
Muokkaa funktio sellaiseksi, että kaikki nousevat poikkeukset käsitellään mielestäsi mielekkäällä tavalla. Kerro kunkin poikkeuskäsittelyn kohdalla miksi päädyit juuri siihen ratkaisuun kyseisen poikkeuksen noustessa. Pyri pitämään metodin suoritus mahdollisimman järkevänä ja sellaisena, että se palauttaisi aina tilanteeseen mahdollisimman hyvin sopivan taulukon.
Et saa tehdä metodiin mitään muita tarkistuksia kuin poikkeusten käsittelyt. Catch-lohkoissa saat kuitenkin käyttää apuna ohjausrakenteita, apumuuttujia jne.

```
public static int[] lueLuvutTiedostosta(String tnimi) {
    String rivi;
    int[] luvut = new int[10];
    int i=0;
    BufferedReader luku;
    luku = new BufferedReader(new FileReader(tnimi));
    while((rivi = luku.readLine())!=null) {
        luvut[i] = Integer.parseInt(rivi);
        i++;
    }
    return luvut;
}
```

Tehtävä 4 (15 p)

Laadi graafinen ikkuna, jossa on kaksi painiketta ja kaksi tekstikenttää, joihin voidaan kirjoittaa tekstiä. Toinen tekstikenttä on yksirivinen ja siinä tulee voida näyttää vähintään 15 merkin mittainen merkkijono. Toinen tekstikenttä puolestaan on sellainen, että siinä voidaan näyttää vähintään 10 riviä tekstiä. Ikkunan otsikkona on "Nimiä".

Ohjelman idea on seuraava: Isompaan tekstikenttään kerätään ihmisten nimiä, yksi rivilleen. Eli isompi tekstikenttä on eräänlainen nimilista. Nimiä ei kuitenkaan voi kirjoittaa suoraan tähän listaan, vaan ne kirjoitetaan pienempään tekstikenttään, josta ne siirtyvät listaan ensimmäisen painikkeen painalluksella. Samalla pienempi tekstikenttä tyhjenee.

Ohjelman rakenteen saat päättää itse.

Toisen painikkeen painallus tyhjentää nimilistan.

Tee luokkaan myös pääohjelmametodi, joka luo ja näyttää kyseisen ikkunan.