## Ohjelmistotekniikan menetelmät, koe 12.12.2012

Vastaa tehtäviin 1, 2 ja 3 **erillisille** konsepteille. Kirjoita jokaiseen palauttamaasi konseptiin kurssin nimi, kokeen päivämäärä, nimi ja opiskelijanumero.

Vastaukset palautetaan tehtäväkohtaisiin pinoihin. **Vaikka jättäisit johonkin tehtävään vastaamatta, tulee vastauspaperi siinäkin tapauksessa palauttaa.**

1. (6p) Kirjoita molemmista kohdista noin **yhden sivun mittainen essee**, jossa selität termien merkityksen ja niiden suhteen toisiinsa. Kirjoita kokonaisia hyvin muotoiltuja lauseita, pelkillä ranskalaisilla viivoilla ei pisteitä ole luvassa.

   (a) ohjelmistotuotantoprojektin vaiheet, vesiputousmalli, ketterät menetelmät

   (b) oliosuunnittelun periaatteet, koodihaju, tekninen velka, refaktorointi

2. (9p) Luokka- ja sekvenssikaavioita.

   (a) (3p) Kurssi koostuu erilaisista oppimiseen tähtäävistä komponenteista. Kurssilla on kontaktiopetusta. Kontaktiopetus jakautuu luentoihin, laskuharjoituksiin ja pajaan. Kurssiin voi kuulua kokeita. Kurssilla on erilaisia oppimistehtäviä. Oppimistehtävät ovat joko teoriatehtäviä tai pajatehtäviä. Kukin teoriatehtävä liittyy tiettyyn laskuharjoitukseen. Mallinna kurssin rakenne luokkakaaviona.

   (b) (2+4p) Tehtäväpaperin lopusta löytyy katkelma Java-koodia. Takaisinmallinna koodi luokka- ja sekvenssikaaviona. Main-luokan main-metodiin on merkitty kohta, josta alkaen sekvenssikaavio piirretään. Luokkakaavioon ei tarvitse merkitä metodeja eikä oliomuuttujia, muista kuitenkin merkitä yhteyksien osallistumisrajoitukset ja suunnat tarkasti.

3. (9p) Tietojenkäsittelytieteen laitos tarvitsee työhuoneiden allokointia tukevan tietojärjestelmän.

   Laitoksen tilavastaava huolehtii, että kaikkien laitoksen käytössä olevien työhuoneiden tiedot ovat ajantasaisena järjestelmässä. Aika ajoin laitos saa käyttöönsä uusia työhuoneita ja välillä myös käytössä olevia työhuoneita vapautetaan muiden laitosten käyttöön. Jokainen työhuone sisältää yhden tai useamman työpisteen, eli yhdelle työntekijälle varatun paikan. Työhuoneen työpisteiden määrä voi nousta tai vähetä jos huoneen kalustusta muutetaan.

   Kukin henkilökunnan jäsen sijoitetaan yhteen työpisteeseen. Kun laitokselle tulee työsuhteeseen uusi työntekijä, siirtää amanuenssi yliopiston henkilönhallintojärjestelmästä uuden työntekijän tiedot työhuonejärjestemään ja sijoittaa hänet johonkin vapaana olevaan työpisteeseen. Kun työntekijä poistuu henkilökunnasta, vapauttaa amanuenssi työpisteen. Amanuenssi voi myös siirtää työntekijän työhuoneesta toiseen. Järjestelmän tulee tarjota amanuenssille ja tilavastaavalle mahdollisuus tarkastella käytössä olevien työhoneiden tilannetta erilaisten hakukriteerien mukaan, esim: näytä kaikki 2. kerroksen huoneet joissa on vapaa työpiste.

Järjestelmän tulee tarjota erilaisia raportteja laitoksen johtajaa varten. Tarjottavia raporttityyppejä ovat esim. huoneisiin sijoitettujen työntekijöiden nimilista, huoneiden käyttöaste sekä inventaario käytössä olevista kalusteista.

(a) (4p) Laadi kuvatusta tietojärjestelmästä karkean tason käyttötapausmalli, eli etsi käyttäjät ja käyttötapaukset. Määrittele kunkin käyttötapauksen kulku lyhyesti (maksimissaan rivi per käyttötapaus) tekstinä. Mainitse myös jokaisen käyttötapauksen yhteydessä siihen liittyvät käyttäjät sekä esiehdot. Piirrä myös käyttötapauskaavio.

(b) (1p) Kuvaa *yksi* käyttötapauksista tarkemmalla tasolla, ns. Cockburnin käyttötapauspohjan tai luentomonisteen tekstuaalisten kuvausten tyyliin.

(c) (4p) Laadi järjestelmän kuvauksen perusteella määrittelyvaiheen (eli kohdealueen) luokkakaavio. Merkitse yhteyksiin kytkentärajoitteet ja nimeä yhteydet ja yhteysroolit tarvittaessa. Ilmeisimmät attribuutit luokille kannattaa merkitä. Muista, että toiminnallisuutta ei kannata määrittelyvaiheen luokkamalliin laittaa!

**Tehtävään 2 (b) liittyvä ohjelmakoodi:**

```
public class Main {
    public static void main(String[] args) {
        Pankki  pankki  = new Pankki();
        Ostoskori ostoskori = new Ostoskori();
        // sekvenssikaavio tästä hetkestä alkaen
        Varasto varasto = new Varasto();
        String tilinumero   = "12345-54321";

        int maara = 3;
        Tuote tuote = varasto.annaTuote("Maito");
        tuote.vahennaSaldoa(maara);
        Ostos ostos = new Ostos(tuote, maara));
        ostoskori.lisaa(ostos);

        maara = 24;
        tuote = varasto.annaTuote("Olut");
        tuote.vahennaSaldoa(maara);
        Ostos ostos = new Ostos(tuote, maara));
        ostoskori.lisaa(ostos);

        int hinta = ostoskori.hinta()
        pankki.veloita(tilinumero, hinta);
    }
}


public class Pankki {
    public boolean veloita(String tilinumero, int summa) {
       // ...
    }
}
```

```
public class Ostos {
    private int maara;
    private Tuote tuote;

    public Ostos(Tuote tuote, int maara) {
        this.tuote = tuote;
        this.maara = maara;
    }

    public int hinta() {
        return tuote.hinta() * maara;
    }
}

public class Ostoskori {
    private ArrayList<Ostos> ostokset = new ArrayList<Ostos>();

    public int hinta() {
        int hinta = 0;
        for (Ostos ostos : ostokset) {
            hinta = hinta + ostos.hinta();
        }
        return hinta;
    }

    public void lisaa(Ostos ostos) {
        this.ostokset.add(ostos);
    }
}

public class Tuote {
    private String nimi;
    private int saldo;
    private int hinta;

    public Tuote(String nimi, int hinta) {
        this.nimi = nimi;
        this.hinta = hinta;
    }

    public String getNimi(){
        return nimi;
    }

    public int hinta() {
        return hinta;
    }

    public void vahennaSaldoa(int vahennys){
        saldo -= vahennys;
```

```
    }
}


public class Varasto {
    private Tuote olut;
    private Tuote maito;

    public Varasto() {
        maito = new Tuote("Maito", 3);
        maito.asetaSaldo(10);

        olut = new Tuote("Olut", 4);
        olut.asetaSaldo(100);
    }

    public Tuote annaTuote(String nimi) {
       if ( nimi.equals("Olut") ) {
            return olut;
       }

       if ( nimi.equals("Maito") ) {
            return maito;
       }

       return null;
    }
}
```