T-106.1208 Ohjelmoinnin perusteet Y (Python). Tentti 16.8.2010

Kirjoita jokaisen vastauspaperisi alkuun kurssin nimi, kokeen päivämäärä, nimesi, opiskelijanumerosi (myös tarkistuskirjain), vastauspaperiesi kokonaismäärä sekä allekirjoituksesi.

Tärkeitä ohjeita vastausten kirjoittamiseen: Kun kirjoitat ohjelmakoodia, käytä kahden ruudun levyisiä sisennyksiä ja merkitse sisennykset selvästi. Jos sisennyksiä ei ole käytetty tai ne on merkitty epäselvästi, vähennetään siitä pisteitä. Kirjoitettavaan ohjelmakoodiin ei tarvitse lisätä kommentteja. Missään tehtävässä tulostusta ei tarvitse muotoilla. Voit myös olettaa, että käyttäjän antama syöte on virheetöntä, ellei tehtävässä erikseen käsketä käsittelemään virhetilanteita.

Vastaa myös kesäkurssin palautekyselyyn, jos et ole vielä vastannut siihen. Kyselyyn vastaamisesta saa 100 harjoitustehtäväpistettä kesäkurssille. Linkki kyselyyn on kesäkurssin www-sivulla.

1. Kohdissa a, b ja c kerro, mitä annettu ohjelma tulostaa. Vastausta ei tarvitse perustella. Kohdissa d, e, f ja g kerro, mitä tehtävässä esitetty funktio tekee. Älä selitä funktion toimintaa käsky käskyltä, vaan selitä parilla lauseella, mikä on funktion tarkoitus. Funktioille annettavien parametrien luonne on selitetty kunkin kohdan yhteydessä. Huomaa, että annetuissa ohjelmissa tai funktioissa voi olla myös virheitä. Selitä siinä tapauksessa, miten annettu virheellinen ohjelma tai funktio toimii - ei siis sitä, miten ohjelman tai funktion pitäisi toimia, jos siinä ei olisi virheitä.

a) (2 p)

```
def main():
    paino = 8.0
    if paino > 3.0:
        print "Hinta on 10.0 euroa."
    elif paino > 6.0:
        print "Hinta on 15.0 euroa."
    else:
        print "Hinta on 5.0 euroa."

main()
```

b) (2 p)

```
def main():
    ika = 10
    kellonaika = 18
    if ika >= 15:
        if kellonaika <= 16:
            print "Lippu maksaa 15 euroa."
        else:
            print "Lippu maksaa 10 euroa."
    else:
        print "Lippu maksaa 5 euroa."

main()
```

c) (3 p)

```
def main():
    luku = 25
    luvut = [10, 5, 8, 2, 10]
    for alkio in luvut:
        luku = luku + alkio
    print luku

main()
```

d) Funktiolle annetaan parametrina kaksi kokonaislukuja sisältävää listaa, joiden pituus on sama. (4 p)

```
def mysteeri1(lista1, lista2):
    i = 0
    summa = 0
    while i < len(lista1):
        if lista1[i] < lista2[i]:
            summa += lista1[i]
        else:
            summa += lista2[i]
        i += 1
    return summa
```

e) Funktiolle annetaan ensimmäisenä parametrina desimaalilukuja sisältävä lista sekä toisena ja kolmantena parametrina desimaaliluku. (4 p)

```
def mysteeri2(lukulista, luku1, luku2):
    summa = 0.0
    for luku in lukulista:
        if luku > luku1:
            summa += luku2 * luku
        else:
            summa += luku
    return summa
```

f) Funktiolle annetaan parametrina kokonaislukuja sisältävä lista. (5 p)

```
def mysteeri3(lista):
    i = 1
    while i < len(lista):
        if lista[i] < lista[i - 1] + 5:
            return False
        i += 1
    return True
```

g) Funktiolle annetaan parametrina merkkijono. (5 p)

```
def mysteeri4(merkkijono):
    luku = len(merkkijono)
    if luku >= 6:
        if merkkijono[0:6] == "kopio-":
            return merkkijono
    return "kopio-" + merkkijono
```

2. a) Kiinteistönvälittäjä perii välittämistään asuntokaupoista palkkiona 4,5 % asunnon myyntihinnasta, kuitenkin vähintään 2500 euroa. Näin lasketun palkkion lisäksi välittäjä perii vielä arvonlisäveron, joka on 23 % lasketusta palkkiosta. Kirjoita Python-ohjelma, joka pyytää käyttäjältä asunnon myyntihinnan ja tulostaa välityspalkkion, johon on laskettu mukaan arvonlisävero. (10 p.)

   b) Eräs kurssi suoritetaan tekemällä harjoitustehtäviä. Harjoitustehtävät pisteytetään, ja kustakin tehtävästä on saatava vähintään määrätty minipistemäärä, jotta tehtävä olisi hyväksytty. Minimipistemäärä on sama kurssin kaikille harjoitustehtäville. Jos opiskelijalla on viisi hyväksyttyä harjoitustehtävää, hän saa kurssin suoritettua arvosanalla 1. Kuudesta hyväksytystä harjoitustehtävästä saa arvosanan 2, seitsemästä arvosanan 3, kahdeksasta arvosanan 4 ja yhdeksästä tai useammasta arvosanan 5. Kirjoita kurssin yksittäisen opiskelijan arvosanan laskemista varten Python-funktio `laske_arvosana(pisteet, minimipistemaara)`, joka saa ensimmaisenä parametrina kokonaislukulistan, joka sisältää opiskelijan pisteet eri harjoitustehtävistä (tehtävien määrä eli listan pituus voi olla eri opiskelijoilla eri) ja toisena parametrina yksittäisen tehtävän hyväksymiseen vaadittavan minimipistemäärän. Funktion tulee palauttaa opiskelijalle kuuluva arvosana. (Tässä tehtävässä ei siis tarvitse kirjoittaa muuta kuin pyydetty funktio, ei esimerkiksi funktiota kutsuvaa ohjelman osaa.) (20 p)

3. Eräässä kesätapahtumassa järjestetään tikanheittokilpailu. Kukin osallistuja saa heittää tikkoja kaksi kierrosta, ja näistä paremman kierroksen pistemäärä otetaan huomioon kilpailun tuloksissa. Kilpailijoiden tulokset tallentetaan teksitiedostoon siten, että kullakin rivillä on aina ensin kilpailijan nimi, sitten ensimmäisen kierroksen tulos ja tämän jälkeen toisen kierroksen tulos. Eri tiedot on erotettu toisistaan kaksoispisteellä. Tiedoston kaksi peräkkäistä riviä voisivat näyttää esim. seuraavalta:

```
Tiina Teekkari:37:28
Risto Raksalainen:22:38
```

   Kirjoita Python-ohjelma, joka pyytää käyttäjältä tulokset sisältävän tiedoston nimen. Ohjelma lukee tästä tiedostosta tuloksia ja tulostaa luettelon, jossa on kunkin kilpailijan nimi ja sen jälkeen tämän kilpailijan parempi tulos. Kilpailijat tulostetaan samassa järjestyksessä kuin he ovat tiedostossa. Heitä ei siis tarvitse järjestää paremmuusjärjestykseen. Esimerkiksi yllä olevassa tapauksessa ohjelma tulostaisi annettujen rivien osalta

```
Tiina Teekkari 37
Risto Raksalainen 38
```

   Ohjelman on käsiteltävä seuraavat virhetilanteet:

   - Annetun nimistä tiedostoa ei ole olemassa tai tiedoston lukeminen ei onnistu jostain muusta syystä
   - Tiedoston jollain rivillä ensimmäisen tai toisen kierroksen tuloksen paikalla olevaa tekstiä ei voi tulkita kokonaisluvuksi.

   Näissä tapauksissa ohjelma ilmoittaa käyttäjälle, millainen virhe on sattunut, ja lopettaa toimintansa. Ohjelman ei siis tarvitse jatkaa rivien lukemista virheellisen rivin jälkeen. Voit myös olettaa, että tiedoston jokaisella rivillä on täsmälleen kolme toisistaan kaksoispisteellä erotettua osaa. Ohjelman ei tarvitse osata käsitellä esimerkiksi sellaisia virhetilanteita, joissa rivi on tyhjä tai ei sisällä nimen lisäksi muuta tekstiä. (20 p)

4. Eräs hotelliketju antaa asiakkailleen bonuspisteitä yöpymisistä. Jokaisesta yöstä standard-luokan huoneessa saa yhden bonuspisteen, ja jokaisesta yöstä superior-luokan huoneessa kaksi bonuspistettä. Kymmenellä bonuspisteellä saa yhden ilmaisen yöpymisen standard-luokan huoneessa ja 20 pisteellä ilmaisen yön superior-luokan huoneessa. Hotelliketju voi myös nimetä osan asiakkaistaan huippuasiakkaiksi. Heidän etunsa ovat muuten samat kuin edellä, mutta he saavat jo 10 bonuspisteellä yön superior-luokan huoneessa. Kirjoita Python-kielellä luokka `Hotelliasiakas` yhden ketjun hotelliasiakkaan kuvaamiseen.

   `Hotelliasiakas`-oliolla on oltava seuraavat kentät:
   - `__nimi` asiakkaan nimi
   - `__bonuspisteet` asiakkaalla tällä hetkellä olevat bonuspisteet
   - `__onko_huippuasiakas` kentän arvo on `True`, jos hotelliketju on nimennyt tämän asiakkaan huippuasiakaaksi ja muuten `False`.

   Määrittele luokkaan seuraavat metodit. (Jos metodin kuvauksessa ei ole kerrottu mitään metodin palauttamasta arvosta, metodin ei tarvitse palauttaa mitään.)
   - `__init__(self, nimi1)` luo uuden `Hotelliasiakas`-olion. Luotavan asiakkaan nimi annetaan parametrina. Uudella asiakkaalla on nolla bonuspistettä, eikä hän ole huippuasiakas.
   - `kerro_nimi(self)` palauttaa asiakkaan nimen.
   - `kerro_bonuspisteet(self)` palauttaa asiakkaan bonuspisteiden määrän.
   - `muuta_huippuasiakkaaksi(self)` nimeää tämän asiakkaan huippuasiakkaaksi.
   - `lisaa_yopymisia(self, lkm, onko_superior)` lisää asiakkaalle bonuspisteitä hotelliyöpymisistä. Lisättävien bonuspisteiden määrä riippuu toisena parametrina annetusta hotelliöiden määrästä ja hotellihuoneen luokasta (standard/superior) tehtävänannon alussa kuvatulla tavalla. Metodin kolmas parametri on `True`, jos yöt on nukuttu superior-luokan huoneessa, ja `False`, jos yöt on nukuttu standard-luokan huoneessa. Jos metodin toinen parametri ei ole positiivinen, metodi ei tee mitään.
   - `kayta_bonuksia(self, onko_superior)` käyttää asiakkaan bonuspisteitä yhden yön maksamiseen, jos asiakkaalla on tarpeeksi bonuspisteitä yöhön halutussa luokassa. Tarvittavien bonuspisteiden määrä on selitetty tehtävänannon alussa. Metodin toinen parametri on `True`, jos bonuspisteillä halutaan maksaa yö superior-luokassa, ja `False`, jos bonuspisteillä halutaan maksaa yö standard-luokassa. Metodi palauttaa arvon `True`, jos asiakkaan bonuspisteet riittävät yhden yön maksamiseen halutussa luokassa (tällöin myös vähennetään asiakkaan bonuspisteitä), ja muussa tapauksessa arvon `False` (silloin asiakkaan bonuspisteitä ei muuteta).
   - `__str__(self)` palauttaa merkkijonon, joka sisältää asiakkaan nimen, bonuspisteiden määrän ja joko tekstin "on huippuasiakas" tai "ei ole huippuasiakas" sen mukaan, onko asiakas huippuasiakas.

   Kirjoita lisäksi pääohjelma, joka luo kaksi `Hotelliasiakas`-oliota ja sen jälkeen ensin kutsuu toiselle niistä `kerro_nimi`-metodia ja sitten lisää samalla asiakkaalle 13 yötä standard-luokassa. Tämän jälkeen ohjelman pitää muuttaa asiakas huippuasiakkaaksi. Sitten ohjelman pitää yrittää käyttää saman asiakkaan bonuspisteitä yhteen yöhön superior-luokassa ja tulostaa, onnistuiko pisteiden käyttö. Lopuksi ohjelman on tulostettava molemmista asiakkaista nimi, bonuspisteet ja tieto siitä, onko asiakas huippuasiakas. Voit kirjoittaa pääohjelman valintasi mukaan joko niin, että se on samassa moduulissa luokan kanssa tai sitten niin, että se on eri moduulissa. (25 p)