T-106.1208 Ohjelmoinnin perusteet Y (Python). Tentti 19.10.2013

Kirjoita jokaisen vastauspaperisi alkuun kurssin nimi, kokeen päivämäärä, nimesi, opiskelijanumerosi, 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. **Tentissä ei saa käyttää laskimia eikä lisämateriaalia. Opiskelijat, joiden äidinkieli ei ole suomi, saavat kuitenkin käyttää sanakirjaa, jos siinä ei ole merkintöjä (tentin valvoja tarkistaa sanakirjan).**

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

a) (2 p)

```
def main():
    tulos = 2900
    if tulos > 2300:
        print("OK")
    elif tulos > 2800:
        print("Loistavaa!")
    else:
        print("Meni kehnosti")

main()
```

b) (2 p)

```
def main():
    paiva = "lauantai"
    kello = 17
    if paiva == "lauantai" and paiva == "sunnuntai":
        print("Lippu maksaa 5 euroa.")
    else:
        if kello >= 8 or kello <= 16:
            print("Lippu maksaa 10 euroa.")
        else:
            print("Lippu maksaa 3 euroa.")

main()
```

c) (3 p)

```
def main():
    lista = [10, 30, 40, 60, 20]
    tulos = 100
    for luku in lista:
        if tulos > 50:
            tulos = tulos - luku
    print(tulos)

main()
```

d) (5 p)

```
def vaihda1(luku):
    luku = 5

def vaihda2(lista):
    lista[2] = 4

def main():
    arvo = 25
    vaihda1(arvo)
    luvut = [10, 20, 30, 40]
    vaihda2(luvut)
    i = 0
    print(arvo)
    while i < 4:
        print(luvut[i])
        i += 1

main()
```

e) Funktiolle annetaan parametreina kaksi kokonaislukuja sisältävää listaa, joilla on sama pituus. Tämä pituus on vähintään yksi. (4 p)

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

f) Funktiolle annetaan parametrina merkkijono. (4 p)

```
def mysteeri2(nimi):
    pit = len(nimi)
    if pit >= 4 and nimi[pit-4:pit] == ".doc":
        return nimi[0:pit-4] + ".txt"
    else:
        return nimi
```

g) Funktiolle annetaan parametrina kokonaislukuja sisältävä lista (jossa on vähintään 2 alkiota) ja kokonaisluku. (5 p)

```
def mysteeri3(luvut, numero):
    i = 1
    while i < len(luvut):
        if luvut[i] != luvut[i - 1] + numero:
            return False
        i += 1
    return True
```

2. a) Kiinteistönvälittäjä perii välittämistään asuntokaupoista palkkiona 3,9 % asunnon myyntihinnasta, kuitenkin vähintään 1900 euroa. Näin lasketun palkkion lisäksi välittäjä perii vielä arvonlisäveron, joka on 24 % lasketusta palkkiosta. Kirjoita Python-ohjelma, joka pyytää käyttäjältä asunnon myyntihinnan ja tulostaa välityspalkkion, johon on laskettu mukaan arvonlisävero. (10 p.)

   b) Erään yrityksen puhelinmyyjillä provisiopalkka lasketaan päivittäin sen mukaan, kuinka monta kappaletta myyjä on päivän aikana myynnyt yrityksen tuotetta. Jos myyjä on myynyt yrityksen tuotetta päivän aikana alle 18 kappaletta, hän saa jokaista myytyä kappaletta kohti määrätyn peruskorvauksen. Jos päivän aikana myyty määrä on 18-36 kappaletta, jokaisesta myydystä kappaleesta saa 1,15 kertaa peruskorvauksen. Jos päivän myynti on yli 36 kappaletta, jokaisesta myydystä kappaleesta saa 1,35 kertaa peruskorvauksen. Myyjällä on kuitenkin päiväkohtainen takuupalkka. Jos päivän myynnin mukaan laskettu provisiopalkka jäisi alle takuupalkan, myyjä saa tuosta päivästä takuupalkan (mutta ei siitä päivästä enää erikseen myyntimäärään perustuvaa korvausta). Takuupalkkaa ei makseta lainkaan niinä päivinä, kun jo päivän provisiopalkka on vähintään yhtä suuri kuin takuupalkka. Yritys haluaa laskea yhdelle myyjälle pidemmältä ajalta kertyvän kokonaispalkan. Kirjoita tätä varten Python-funktio `laske_kokonaispalkka(myyntitiedot, peruskorvaus, takuupalkka)`. Funktion ensimmäisenä parametrina on lista, joka sisältää eri päivien myyntimäärät (listan yksi alkio on yhden päivän aikana myytyjen kappaleiden määrä). Listan alkiot ovat kokonaislukuja, eikä listan pituutta ole määrätty etukäteen. Funktion toisena parametrina on yhden kappaleen myynnistä maksettava peruskorvaus euroina (desimaaliluku), ja kolmantena parametrina päiväkohtainen takuupalkka euroina. Funktion on laskettava ja palautettava puhelinmyyjälle listassa olevilta päiviltä kuuluva palkka yhteensä. (20 p)

3. Yritys on tallentanut tiedot sen saamista tilauksista tiedostoon. Tiedoston kukin rivi kuvaa yhden tuotteen yhtä tilausta, Rivillä on ensimmäisenä päivämäärä, sitten tilatun tuotteen tuotenumero, sitten tietot siitä, kuinka monta kappaletta tuotetta on tilattu ja lopuksi tilaajan asiakasnumero. Eri tiedot on erotettu toisistaan puolipisteellä. Tiedoston rivit voisivat näyttää esim. seuraavilta:

```
12.03.2013;T-12345;26;A-2345
13.03.2013;T-22556;30;A-2255
14.03.2013;T-12345;5;A-5673
```

   Huomaa, että saman tuotteen tilauksia voi olla usealla eri rivillä. Kirjoita Python-ohjelma, joka pyytää käyttäjältä tilaukset sisältävän tiedoston nimen ja tiedon siitä, minkä tuotteen tilauksia haetaan (käyttäjä antaa tuotteen tuotenumeron). Ohjelma lukee tiedot tästä tiedostosta ja tulostaa, kuinka monta kappaletta tuotetta on tilattu yhteensä. Esimerkiksi jos käyttäjä antaisi tuotenumeron T-12345, tulostaisi ohjelma yllä olevassa tapauksessa arvon 31. Jos tiedostosta ei löydy lainkaan halutun tuotteen tilauksia, ohjelma tulostaa tilattujen kappaleiden määräksi nollan.

   Ohjelman on käsiteltävä seuraavat virhetilanteet:
   - Annetun nimistä tiedostoa ei ole olemassa tai tiedoston lukeminen ei onnistu jostain muusta syystä
   - Tiedoston jollain rivillä kappalemäärän 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 neljä toisistaan puolipisteellä erotettua osaa. Ohjelman ei tarvitse osata käsitellä esimerkiksi sellaisia virhetilanteita, joissa rivi on tyhjä tai ei sisällä päivämäärän lisäksi muuta tekstiä. (20 p)

TENTTI JATKUU SEURAAVALLA SIVULLA

4. Kirjoita Python-kielellä luokka `Liittyma` yhden prepaid-tyyppisen matkapuhelinliittymän tietojen käsittelyyn. (Prepaid-tyyppisissä liittymissä liittymälle ladataan rahaa etukäteen, ja puhelimella voi soittaa ja lähettää tekstiviestejä vain sen verran kuin mihin liittymälle ladatut rahat riittävät.)

   `Liittyma`-oliolla on oltava seuraavat kentät:
   - `__numero` matkapuhelinliittymän puhelinnumero (merkkijono).
   - `__saldo` liittymän saldo tällä hetkellä eli liittymällä olevien rahojen määrä sentteinä (kokonaisluku). Saldo pienenee, kun soitetaan tai lähetetään tekstiviestejä. Sitä voi kasvattaa lataamalla liittymälle lisää rahaa.
   - `__onko_suljettu` kentän arvo on `True`, jos liittymä on suljettu (siitä ei voi soittaa tai lähettää tekstiviestejä) ja muuten `False` (liittymää voi käyttää normaalisti).

   Määrittele luokkaan seuraavat metodit. (Jos metodin kuvauksessa ei ole kerrottu mitään metodin palauttamasta arvosta, metodin ei tarvitse palauttaa mitään.)
   - `__init__(self, puhelinnumero, alkusaldo)` luo uuden `Liittyma`-olion. Luotavan liitymän puhelinnumero ja liittymälle ladattava alkusaldo sentteinä annetaan parametreina. Jos viimeinen parametri on negatiivinen, alkusaldoksi asetetaan 0. Uusi liittymä ei ole suljettu.
   - `kerro_saldo(self)` palauttaa liittymällä tällä hetkellä olevan saldon.
   - `onko_suljettu(self)` palauttaa arvon `True`, jos liittymä on suljettu ja muuten arvon `False`.
   - `avaa_liittyma(self)` avaa liittymän.
   - `sulje_liittyma(self)` sulkee liittymän.
   - `lataa_rahaa(self, lisays)` Lataa liittymälle parametrina annetun määrän rahaa eli kasvattaa liittymän saldoa parametrina annetulla määrällä, jos parametri on positiivinen. Jos parametri ei ole positiivinen, metodi ei tee mitään. Parametri on kokonaisluku ja sisältää lisättävän määrän sentteinä.
   - `soita_puhelu(self, kesto)` "soittaa" liittymästä parametrina annetun keston (sekunneissa, kokonaisluku) mittaisen puhelun, jos liittymä ei ole suljettu ja liittymän saldo riittää puhelun maksamiseen. Jos liittymän saldo ei riitä koko puhelun maksamiseen mutta liittymä ei ole suljettu, metodi "soittaa" kuitenkin niin pitkän puhelun kuin mihin saldo riittää. Puhelun jokainen alkava 10 sekuntia maksaa yhden sentin (esimerkiksi 10 sekunnin pituinen puhelu maksaa 1 sentin, mutta 11 sekuntia kestävä puhelu 2 senttiä). Metodi palauttaa puhelun todellisen keston (joka on siis parametrina annettua kestoa pienempi, jos saldo ei riitä haluttuun kestoon). Käytännössä "soittaminen" näkyy siten, että metodi pienentää liittymän saldoa puhelun hinnalla. Jos liittymä on suljettu, metodi palauttaa arvon 0.
   - `laheta_tekstiviesti(self)` "lähettää" liittymästä yhden tekstiviestin, jos liittymä ei ole suljettu ja liittymän saldo riittää tekstiviestin lähettämiseen. Yhden tekstiviestin hinta on 6 snt. Metodi palauttaa arvon `True`, jos tekstiviestin lähettäminen onnistuu ja muuten arvon `False`. Jos lähettäminen onnistuu, metodi myös vähentää liittymän saldoa tekstiviestin hinnalla.
   - `__str__(self)` palauttaa merkkijonon, joka sisältää liittymän puhelinnumeron, saldon ja joko tekstin "liittyma on suljettu" tai "liittyma on kaytossa" sen mukaan, onko liittymä suljettu vai ei.

   Kirjoita lisäksi pääohjelma, joka luo kaksi `Liittyma`-oliota, lataa niistä toiselle rahaa, soittaa sen jälkeen toisella liittymällä puhelun ja kertoo sen keston. Tämän jälkeen pääohjelman pitää lähettää toisella liittymällä tekstiviesti ja kertoa, onnistuiko viestin lähetys. Lopuksi ohjelman pitää sulkea toinen liittymä ja tulostaa molemmista liittymistä puhelinnumero, saldo ja tieto siitä, onko liittymä suljettu vai käytössä. Voit päättää liittymien alkutiedot sekä latauksessa ja puhelussa tarvittavat tiedot itse. Pääohjelman ei siis tarvitse kysyä mitään käyttäjältä. Voit kirjoittaa pääohjelman valintasi mukaan joko niin, että se on samassa moduulissa luokan kanssa tai sitten niin, että se on eri moduulissa. (25 p)