CSE-A1111 Ohjelmoinnin peruskurssi Y1 Tentti 17.2.2016

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. Jos sisennyksiä ei ole käytetty tai niistä ei saa selvää, 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). Nämä opiskelijat saavat halutessaan myös sekä suomen- että englannin- tai ruotsinkielisen kysymyspaperin.

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():
    nopeus = 115
    if nopeus > 100:
        print("Rikesakko.")
    if nopeus > 120:
        print("Paivasakkoja.")
    else:
        print("Ei sakkoa.")

main()
```

b) (2 p)

```
def tuplaa(luku):
    numero = 8
    return 2 * luku

def main():
    numero = 10
    print(tuplaa(numero))

main()
```

c) (3 p)

```
def main():
    lista = [10, 30, 15, 40, 5]
    tulos = 0
    i = 0
    while i < len(lista):
        if lista[i] > 25:
            tulos = tulos + lista[i]
        i = i + 1
        print(tulos)

main()
```

d) (5 p)

```
def muuta(lista, arvo):
    lista[0] = lista[1] + lista[2]
    arvo = 29
    return arvo

def main():
    numerot = [6, 4, 8]
    numero = 15
    muuta(numerot, numero)
    print(numero)
    for alkio in numerot:
        print(alkio)

main()
```

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

```
def mysteeri1(lista1, lista2):
    i = 0
    tulos = 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(merkkijono1):
    luku = len(merkkijono1)
    if luku >= 6:
        if merkkijono1[0:6] == "kopio-":
            return merkkijono1
    return "kopio-" + merkkijono1
```

g) Funktiolle annetaan parametrina positiivisia kokonaislukuja sisältävä lista, jossa on vähintään kaksi alkiota. (5 p)

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

2. a) Teemu Teekkari on järjestämässä juhlia ja on pyytänyt tarjouksia ruokatarjoilun järjestämisestä kolmesta eri yrityksestä. Yritysten veloittamat summat riippuvat sekä vieraiden lukumäärästä että juhlien kestosta. Yritys A laskuttaa 18 eur / vieras ja lisäksi 120 eur / tunti. Yritys B laskuttaa 24 eur / vieras ja lisäksi 85 eur / tunti. Yritys C veloittaa 40 eur / vieras, jos vieraita on alle 20, ja 35 eur / vieras, jos vieraita on 20 tai enemmän. Yristys C ei laskuta lainkaan juhliin kuluneesta ajasta, vaan sen hinta riippuu vain vieraiden määrästä. Kirjoita Python-ohjelma, joka kysyy käyttäjältä vieraiden lukumäärän ja juhlien keston. Tämän jälkeen ohjelman pitää tulostaa, minkä yrityksen tarjous on halvin ja kuinka paljon juhlien tarjoilu maksaa tämän yrityksen järjestämänä. Jos useammalla kuin yhdellä yrityksellä on sama hinta, ohjelman tarvitsee tulostaa näistä halvimmista yrityksistä vain yksi (ei ole väliä mikä). (10 p.)

   b) Eräässä yrityksessä maksetaan työntekijöille palkkaa tehtyjen tuntien mukaan seuraavasti: Jos yhden päivän työtuntien määrä on korkeintaan 8, maksetaan tunneista normaali tuntipalkka. Jos päivän työtuntien määrä on yli 8, mutta korkeintaan 10, maksetaan normaali tuntipalkka 8 tunnista ja sen yli menevistä tunneista 1,5-kertainen tuntipalkka. Jos päivän työtuntien määrä on yli 10, maksetaan normaali tuntipalkka 8 tunnista, 1,5-kertainen tuntipalkka 2 tunnista ja kaksinkertainen tuntipalkka yli 10:n menevistä tunneista. Kirjoita Python-funktio`laske_kokonaispalkka(tuntilista, tuntipalkka)`. Funktio saa ensimmäisenä parametrina listan, joka sisältää erään työntekijän eri päivinä tekemät työtunnit (desimaalilukuja) ja toisena parametrina tämän työntekijän normaalin tuntipalkan (desimaaliluku). Funktio laskee ja palauttaa työntekijälle listassa olevilta työtunneilta kuuluvan kokonaispalkan. Huomaa, että jokaisen päivän palkka lasketaan erikseen. Vaikka jonain työpäivänä työtuntien määrä olisi ollut alle 8, niin seuraavan päivän palkka lasketaan joka tapauksessa annettujen sääntöjen mukaan. Kirjoita vain tehtävässä pyydetty funktio, ei muita palkanlaskentaohjelman osia. Voit olettaa, että funktiolle parametrina annetut arvot ovat järkeviä. (20 p)

3. Verkkokauppa on tallentanut tiedot asiakkaiden tekemistä tilauksista tekstitiedostoon seuraavasti: Yhdellä rivillä on yhden tuotteen yhden tilauskerran tiedot. Rivillä on tilauksen päivämäärä, tilauksen tekijän asiakasnumero, tilatun tuotteen tuotenumero, kuinka monta kappaletta tuotetta on tilattu (kokonaisluku) ja tuotteen kappalehinta tässä järjestyksessä. Tiedot on erotettu toisistaan kaksoispisteellä.Tiedoston rivit voisivat näyttää esim. seuraavilta:

```
5.12.2015:44556:2322-1:3:10.00
9.12.2015:77221:1177-3:1:15.80
30.12.2015:83432:2322-1:2:8.00
```

   Kirjoita Python-ohjelma, joka pyytää käyttäjältä tilaustiedot sisältävän tiedoston nimen ja tiedon siitä, minkä tuotteen tilauksia etsitään (tuotteen tuotenumeron). Ohjelma lukee tiedoston ja tulostaa, kuinka monta kappaletta annettua tuotetta on tilattu ja mikä on tämän tuotteen tilausten yhteisarvo. Jos esimerkiksi yllä olevasta esimerkkitiedostosta etsittäisiin tuotteen 2322-1 tietoja, ohjelma tulostaisi, että tuotetta on tilattu yhteensä 5 kappaletta ja tilausten yhteisarvo on 46.0 euroa (huomaa, että tuotteen kappalehinta voi olla eri tilauksissa eri suuri).

   Ohjelman on käsiteltävä seuraavat virhetilanteet:
   - Annetun nimistä tiedostoa ei ole olemassa tai tiedoston lukeminen ei onnistu jostain muusta syystä.
   - Tiedoston jollain rivillä kappalemäärän paikalla ei ole kokonaisluku tai kappalehinnan paikalla desimaaliluku.

   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 viisi toisistaan kaksoispisteellä erotettua osaa. Ohjelman ei tarvitse osata käsitellä esimerkiksi sellaisia virhetilanteita, joissa rivi on tyhjä tai ei sisällä päivämäärän lisäksi muuta tekstiä. (20 p)

VIIMEINEN TEHTÄVÄ SEURAAVALLA SIVULLA

4. Ystäväsi aikoo perustaa pikavippiyrityksen ja tarvitsee tietokoneohjelman asiakasrekisteriä varten. Kirjoita ohjelmaa varten Python-luokka `Vippaaja` yhden asiakkaan tietojen kuvaamista varten.

`Vippaaja`-oliolla on oltava seuraavat kentät:

- `__nimi` asiakkaan nimi
- `__luottosaldo` asiakkaan velka yritykselle tällä hetkellä (positiivinen, jos asiakas on velkaa)
- `__pisteet` kokonaisluku (voi olla myös negatiivinen), joka kuvaa sitä, kuinka hyvä asiakas on kysymyksessä
- `__luottoraja` asiakkaalle voidaan yleensä antaa lainaa korkeintaan luottorajaan asti.

Määrittele luokkaan seuraavat metodit. (Jos metodin kuvauksessa ei ole kerrottu mitään metodin palauttamasta arvosta, metodin ei tarvitse palauttaa mitään.)

- `__init__(self, nimi1, alkusaldo, raja)` luo uuden `Vippaaja`-olion. Luotavan asiakkaan nimi, alkuperäinen luottosaldo ja luottoraja annetaan parametreina. Uudella asiakkaalla on 5 pistettä. Metodin ei tarvitse tarkistaa annettujen parametrien järkevyyttä.
- `kerro_luottosaldo(self)` palauttaa asiakkaan luottosaldon.
- `kerro_pisteet(self)` palauttaa asiakkaan pisteet.
- `muuta_luottorajaa(self, uusi_raja)` muuttaa asiakkaan luottorajan parametrin mukaiseksi, jos parametri on vähintään 0. Metodi ei tee muita tarkistuksia, joten esim. luottosaldo voi olla metodin suorituksen jälkeen suurempi kuin luottoraja.
- `tee_lyhennys(self, summa)` pienentää asiakkaan luottosaldoa. Asiakkaan maksama summa annetaan parametrina. Summaa ei kuitenkaan käytetä kokonaan lyhennykseen, vaan siitä vähennetään ensin 5 euroa luoton hoitokuluja sekä edellisen lyhennyksen jälkeen kertynyt korko, joka on 10 % luottosaldosta. Tämän jälkeen jäljelle jäänyt summa käytetään varsinaiseen lyhennykseen eli luottosaldoa pienennetään sen verran. Jos varsinainen lyhennys on vähintään 30 euroa, asiakkaan pisteitä kasvatetaan yhdellä. Jos varsinainen lyhennys jää vähennysten jälkeen negatiiviseksi, lisätään sen itseisarvo luottosaldoon (myös siinä tapauksessa, että luottosaldo ylittää tämän jälkeen luottorajan), ja pisteitä vähennetään yhdellä. Luottosaldo saa mennä myös negatiiviseksi.
- `ota_lisaa_luottoa(self, lisays)` kasvattaa asiakkaan luottosaldoa parametrina annetulla määrällä. Kasvatus onnistuu kuitenkin vain siinä tapauksessa, että asiakkaalla on vähintään viisi pistettä ja että luottosaldo on lisäyksen jälkeen korkeintaan luottoraja. Jos se onnistuu, asiakkaan pisteitä vähennetään viidellä ja metodi palauttaa arvon `True`. Jos luoton kasvatus ei onnistu, luottosaldoa ja pisteitä ei muuteta ja metodi palauttaa arvon `False`.
- `__str__(self)` palauttaa merkkijonon, joka sisältää asiakkaan nimen, luottosaldon, luottorajan ja pisteet.

Kirjoita lisäksi joko samaan tai toiseen moduuliin pääohjelma, joka luo kaksi `Vippaaja`-oliota ja sen jälkeen kutsuu toiselle niistä `kerro_luottosaldo`-metodia ja `tee_lyhennys`-metodia. Tämän jälkeen ohjelman pitää yrittää kasvattaa asiakkaan luottoa ja tulostaa joko "lisaluoton ottaminen onnistui" tai "lisaluoton ottaminen ei onnistunut" sen mukaan, onnistuiko luoton kasvattaminen. Lopuksi ohjelman on tulostettava molempien asiakkaiden tiedot (nimi, luottosaldo, luottoraja ja pisteet). Voit päättää asiakkaiden ja luottojen tiedot itse. Niitä ei tarvitse kysyä käyttäjältä. (25 p)