

L10 Tehtävät

- Tietorakenne sanakirja, dictionary, ja sen käyttö
- Tietorakenne numpy-matriisi ja sen käyttö
- Lajittelu: tuple, sanakirja, oliolista

Tällä viikolla keskitytään uusiin tietorakenteisiin. Ohjelmointitehtävissä on oltava otsikkotiedot ja poikkeusten käsittely.

Poikkeusten käsittely	1
L10T1: Tekstiedoston automerkkien analysointi sanakirjan avulla	1
L10T2: Vuosittain rekisteröityjen autojen lukumäärät lajiteltuna sanakirjalla.....	3
L10T3: Tietorakenteet ja numpy-matriisin käyttö.....	4

Poikkeusten käsittely

Laita ohjelmien tiedoston käsittelyyn tyyliohjeen mukainen poikkeusten käsittely eli

"Tiedoston 'x.txt' käsittelyssä virhe, lopetetaan."

L10T1: Tekstiedoston automerkkien analysointi sanakirjan avulla

Tehtävässä L09T3 luokiteltiin merkkijonoista koostuva tiedosto siten, että saimme selvitettyä tiedostossa nimetyt eri automerkit tunnistamalla erilaiset merkkijonot. Tässä tehtävässä mennään pykälä pidemmälle eli nyt tulee selvittää tiedostossa olevat eri automerkit sekä niiden esiintymismäärät käyttäen hyväksi sanakirjaa.

Tee ohjelma, joka kysyy käyttäjältä luettavan ja kirjoitettavan tiedoston nimet ja selvittää luettavassa tiedostossa olevien eri automerkkien määrät, kun yhdellä rivillä on aina yhden auton merkki. Tulostiedostoon kirjoitetaan kukin automerkki yhden kerran aakkosjärjestyksessä ja automerkin perään tule sen esiintymiskertojen lukumäärä.

Käytä tehtävässä sanakirjaa eli dictionary –tietorakennetta ja sorted-funktiota. Funktio on esitelty ohjelmointioppaassa ja vaikka siellä ei ole valmista ratkaisua, kannattaa kokeilla toimiiko sorted-funktio sanakirjan kanssa vastaavasti kuin tuple:n kanssa. Voit testata ohjelmaa edellisen viikon tiedostoilla L09T3D1.txt, L09T3D2.txt ja L09T3D3.txt. Edelleen tiedosto L09T3D3.txt on tyhjä, ts. siellä ei ole mitään ja näin ollen sitä ei ole saatavilla Moodlella, koska Moodle ei hyväksy tyhjiä tiedostoja.

Toteuta ohjelma kolmella aliohjelmalla eli tiedoston luku, analyysi ja tiedoston kirjoitus pääohjelman lisäksi. Alla näkyy esimerkki ohjelman lukemasta tiedostosta ja tiedostoon kirjoitetaan täsmälleen samat asiat mitä aliohjelma tulostaa näytölle, ts. otsikkorivi ja data-rivit. Kysy tiedostonimet pääohjelmassa ja välitä ne aliohjelmiin parametreina. Tee varsinainen analyysi analyysi-aliohjelmassa, mutta tulosten lajittelun voi tehdä tulostamisen yhteydessä. Esimerkkiajossa näkyy ohjelman tulosteet ml. automerkit ja lukumäärätiedot. Huomaa, että tulosteessa auto-sanan muoto riippuu niiden lukumäärästä. Mikäli luettava tiedosto on tyhjä, älä turhaan kutsu analyysi- ja kirjoitus-aliohjelmiä vaan tulosta ilmoitus "Tiedosto oli tyhjä, yhtään automerkkiä ei tunnistettu."

Ohjelman lukeman tiedoston alku, L09T3D1.txt:

Kia
Kia
Mazda
Mazda
Mazda
Mercedes-Benz

Ohjelman esimerkkiajo:

Anna luettavan tiedoston nimi: L09T3D1.txt
Anna kirjoitettavan tiedoston nimi: L10T1T1.txt
Tunnistettiin 8 automerkkiä ja 15 autoa:
Kia: 2 autoa
Mazda: 3 autoa
Mercedes-Benz: 1 auto
Opel: 1 auto
Renault: 2 autoa
Seat: 1 auto
Toyota: 1 auto
Volkswagen: 4 autoa
Kiitos ohjelman käytöstä.

L10T2: Vuosittain rekisteröityjen autojen lukumäärät lajiteltuna sanakirjalla

Tee Python-ohjelma, joka laskee eri vuosina rekisteröityjen henkilöautojen lukumäärät annetusta tiedostosta L10T2D1.txt. Käytetyssä tiedostossa vuosiluku löytyy toisen kentän neljästä ensimmäisestä merkistä. Käytä laskennassa hyväksi sanakirjaa ja tulosta tiedot lajiteltuna vuosiluvun mukaan laskevaan järjestykseen esimerkkiajon mukaisesti.

Toteuta ohjelma kahdella aliohjelmalla pääohjelman lisäksi, lue ja tulosta. Kysy tiedostonimi pääohjelmassa ja välitä se luku-aliohjelmaan sanakirjan kanssa. Lajittele sanakirja tulosta-aliohjelmassa, jotta saat tulosteet haluttuun järjestykseen. Voit muotoilla tulosteen haluamallasi tavalla, mutta format-käskyssä toimii seuraava muotoilu: "{0:d}: {1:d}".

Huomaa, että tiedostossa on nyt enemmän dataa, kuten alla olevasta tiedoston alusta näkyy. Jos avaat tiedoston, älä muokkaa äläkä ainakaan talleta sitä, sillä muutoin sen datan saattaa muuttua ja ohjelma ei toimi. Mikäli ohjelmasi toimii oikein omalla koneellasi, mutta CodeGradessa tulee joku tiedostomuotoon liittyvä virhe, ota Moodlesta alkuperäisessä muodossa oleva tiedosto uudestaan ja varmistu, että ohjelmasi toimii oikein sen kanssa. Tiedostoa avatessasi käytä UTF-8 -koodausta. Tällaisia tietoja käsitellään usein taulukkolaskentaohjelmilla ja ne muuttavat helpolla merkkijonoja tietotyypeiksi, jolloin erityisesti päivämäärät ja desimaaliluvut voivat muuttaa esitysmuotoa.

Ohjelman lukeman tiedoston alku

```
ajoneuvoluokka;ensirekisterointipvm;ajoneuvoryhma;ajoneuvonkaytto;variantti;versio;
kayttoonottopvm;vari;ovienLukumaara;korityyppi;ohjaamotyyppi;istumapaikkojenLkm;oma
massa;teknSuurSallKokmassa;tieliikSuurSallKokmassa;ajonKokPituus;ajonLeveys;ajonKor
keus;kayttovoima;iskutilavuus;suurinNettoteho;syntereidenLkm;ahdin;sahkohybridi;m
erkkiSelvakielinen;mallimerkinta;vaihteisto;vaihteidenLkm;kaupallinenNimi;voimanval
JaTehostamistapa;tyyppihyaksyntanro;yksittaisKayttovoima;kunta;Co2;matkamittariluk
ema;alue;valmistenumero2;jarnro
```

```
M1;2010-01-04;;01;AABZBX0;FM6FM62S002ST0GG;20100104;0;4;AA;;4;1505;1920;;4886;1855;
1417;01;1798;118;4;true;;Volkswagen;PASSAT CC Sedan (AA) 4ov 1798cm3;1;6;PASSAT
CC;05;e1*01/116*0468*02;01;049;178;54345;021;WVWZZZ3CZ9;3315144
```

```
M1;2010-01-29;;01;AFBLSX01;SGEFM5A40447GG;20090923;8;4;AF;1;5;1325;1820;1820;4315;
1768;1459;02;1896;77;4;true;false;Seat;LEON Monikäyttöajoneuvo (AF) 4ov
1896cm3;1;5;LEON;05;e9*01/116*0052*19;02;179;119;226341;418;VSSZZZ1PZA;3400059
```

```
M1;2010-01-05;;01;14Z;2EU;20100105;8;4;AB;;5;1255;1770;;4490;1755;1470;01;1598;77;
4;false;;Mazda;3 Viistoperä (AB) 4ov
1598cm3;1;5;3;05;e11*01/116*0262*02;01;405;149;108559;539;JMZBL14Z20;3430029
```

```
M1;2010-02-19;;01;ZRT271(E);ZRT271L-AEFEPW(1C);20091109;Y;4;AA;;5;1450;2000;;4715;
1810;1480;01;1798;108;4;false;;Toyota;AVENSIS Sedan (AA) 4ov 1798cm3;1;6;AVENSIS;
05;e11*01/116*0331*02;01;272;154;156283;676;SB1BG76L00;3460463
```

Ohjelman esimerkkiajo:

Anna luettavan tiedoston nimi: L10T2D1.txt

Autot lajiteltuna vuosiluvun mukaan laskevaan järjestykseen.

Vuosi: Autoja

2016: 66

2015: 64

2014: 56

2013: 57

2012: 68

2011: 65

2010: 63

Yhteensä 439 autoa.

Kiitos ohjelman käytöstä.

L10T3: Tietorakenteet ja numpy-matriisin käyttö

Viikolla 8 asensimme numpy-kirjaston Pythonin laajennoksena ja tässä tehtävässä tutustutaan numpy-matriisin perustoimintaan.

Luo 4x4-kokonaislukumatriisi ja alusta se nolliksi numpyn zeros-jäsenfunktiolla. Sen jälkeen käy läpi matriisin kaikki alkiot kahdella silmukalla, ensin rivit ja sitten sarakkeet, ja sijoita matriisin jokaisen alkion arvoksi sen sarake- ja rivi-indeksin tulo siten, että sekä sarake- että rivi-indekseihin lisätään yksi, ts. $(\text{RiviIndeksi}+1) * (\text{SarakeIndeksi}+1)$, ks. esimerkkituloste alla. Indekseihin lisätään 1, jottei taulukko jää suurelta osin nolliille.

Tulosta valmis matriisi print-käskyllä, jolloin se noudattaa numpyn tarjoamaa muotoilua.

Tulosta sitten matriisi uudestaan riveittäin sarakkeet puolipisteillä eroteltuina alla olevan esimerkkiajon mukaisesti, sillä tämä muoto on helppo siirtää esim.

taulukkolaskentaohjelmaan visualisointia varten. Lopuksi muista vapauttaa numpyn varaama muisti delete-käskyllä.

Ohjelman esimerkkiajo 1:

Tämä ohjelma testaa numpy-matriisin käyttöä.

Matriisi tulostettuna numpy-muotoilulla:

```
[[ 1  2  3  4]
 [ 2  4  6  8]
 [ 3  6  9 12]
 [ 4  8 12 16]]
```

```
Matriisi tulostettuna alkiot puolipisteillä eroteltuna:1;2;3;4;
2;4;6;8;
3;6;9;12;
4;8;12;16;
```

Kiitos ohjelman käytöstä.