L06 Tehtävät

- Tekstitiedoston käsittely eli lukeminen ja kirjoittaminen
- Merkkijonomuuttujiin liittyvät jäsenfunktiot ja merkkijonojen muokkaus
- Usean tiedoston käsittely rinnakkainen
- Tiedostossa olevan numerotiedon analyysi: summa, minimi ja maksimi
- Laajeneva valikkopohjainen ohjelma T4
- Käytä paaohjelma()-rakennetta kaikissa tehtävissä L05 mukaisesti

Lue oppaan tämän viikon asioita käsittelevä luku 6. Lisäksi tehtävien suorittamiseen tarvitset aiempien lukujen tietoja.

Yleisiä huomioita tiedostonkäsittelyyn liittyen

Tiedostoja käsittelevissä ohjelmointitehtävissä tulee huomata seuraavat asiat:

- Mikäli viikkotehtävissä tarvitaan valmiita tiedostoja, ovat ne kaikki Moodlessa viikon tehtävien jälkeen yhdessä kansiossa.
- Käytä tiedostojen tarkasteluun koodieditoria, esim. IDLEä. Windowsin **Notepad ei näytä rivinvaihtoja oikein** ja tekstinkäsittelyohjelma kuten MS-Word ei sovi myöskään tähän tarkoitukseen.
- IDLE näyttää oletusarvoisesti Python tiedostoja ja esim. tekstitiedostot pitää valita erikseen näkyville, ts. avatessasi tiedostoa (File | Open), valitse tiedostotyypiksi *.txt tai *.*
- **Mikäli tehtävässä luetaan tiedostoa**, tulee luettavan tiedoston olla **samassa hakemistossa Python-koodin kanssa**.
- Jos ohjelma luo tiedostoja, tutki resurssienhallinnan ja editorin avulla, mitä ja minkälaisia tiedostoja ohjelmasi teki ajon aikana. Muuten et voi löytää virheitä koodistasi. Tutki myös luettavien tiedostojen rakenne tarkasti ennen tehtävän aloittamista. Ja tiedostoja tutkimiseen tulee käyttää koodieditoria, esim. IDLEä.
- Käsiteltävien tiedostojen nimet perustuvat aina tehtävän tunnukseen, esim. L06T1, jonka jälkeen luettavissa data-tiedostoissa on D(ata) ja tulos-tiedostoissa on T(ulos). Mikäli tiedostoja on useita, tulee D ja T merkkien jälkeen tiedostojen tunnistenumerot 1 ja 2 jne. Datatiedostoilla on tyypillisesti laajennos .txt.

L06T1: Tekstitiedoston kirjoittaminen ja lukeminen

Tee Python-ohjelma, joka kirjoittaa ja lukee tekstitiedostoja.

1. Ohjelmassa tulee olla pääohjelma ja kaksi aliohjelmaa, tiedostoKirjoita ja tiedostoLue. Kysy pääohjelmassa tallennettavan tiedoston nimi ja välitä se parametrina molempiin aliohjelmiin, kutsu ensin kirjoitus-aliohjelmaa.
2. Kirjoitus-aliohjelma kysyy käyttäjältä nimen ja lisää sen tekstitiedoston loppuun omalle rivilleen. Toteuta kysyminen toistorakenteessa ja lopeta kysyminen, kun käyttäjä antaa syötteeksi enterin (ts. pelkkä rivinvaihtomerkki, enter/return). Tiedosto tulee avata aliohjelman alussa ja mahdollinen aiempi tiedoston sisältö tulee tuhota avauksen yhteydessä. Aliohjelman lopuksi tiedosto suljetaan.
3. Luku-aliohjelmalla varmistetaan, että tiedostossa on haluttu sisältö. Avaa tiedosto, lue sen sisältö ja tulosta kaikki tiedostossa olevat nimet eli rivit näytölle sekä lopuksi sulje tiedosto.

Huomaa, että tässä tehtävässä pitää toteuttaa oikein toistorakenne (L04), pää/aliohjelmat (L05) ja tiedostonkäsittely (L06). Virhe missä tahansa näistä kohdista estää ohjelman oikean toiminnan, joten tarkista em. rakenteet aiemmilta viikoilta tarpeen mukaan. Ohjelman tulosteet ja toiminta näkyy alla olevassa esimerkkiajossa.

Ohjelman esimerkkiajo:

```
Anna tallennettavan tiedoston nimi: L06T1T1.txt
Anna tiedostoon tallennettava nimi (enter lopettaa): Ville
Anna tiedostoon tallennettava nimi (enter lopettaa): Riitta
Anna tiedostoon tallennettava nimi (enter lopettaa): Erkki
Anna tiedostoon tallennettava nimi (enter lopettaa): Seija
Anna tiedostoon tallennettava nimi (enter lopettaa):
Tiedostoon 'L06T1T1.txt' on tallennettu seuraavat nimet:
Ville
Riitta
Erkki
Seija
Kiitos ohjelman käytöstä.
```

L06T2: Tiedoston rivimäärän ja merkkien laskeminen

Tee ohjelma, joka lukee tekstitiedoston ja laskee siinä olevien rivien sekä merkkien määrät. Tiedosto `L06T2D2.txt` sisältää kaikki väestörekisterissä elokuussa 2020 olleet naisten nimet, joita oli vähintään viidellä ihmisellä; tietosuojasyiden vuoksi tätä harvinaisempia nimiä ei julkaista. Laskemalla tiedoston rivien määrän saat selville, kuinka monta nimeä tiedostossa on. Laske lisäksi tiedostossa olevien merkkien määrä sekä nimien keskimääräinen pituus alla olevan esimerkkiajon mukaisesti. Tiedostossa `L06T2D3.txt` on vastaavasti kaikkien miesten nimet.

Ohjelman toiminta ja laskenta kannattaa testata ensin toimivaksi tiedostolla `L06T2D1.txt`, jossa on nimet Matti ja Teppo, jotta voit helposti varmistua laskennan oikeellisuudesta.

Toteuta ohjelma pääohjelmassa, jonka alussa avaa tiedosto UTF-8 koodattuna, jotteivat ääkköset sotke ohjelmaa. Merkkejä laskiessa on hyvä pitää mielessä, mitä merkkejä sisältyy riviin ja toisaalta nimeen.

Huomaa, että nimien pituuden keskiarvossa ei ole desimaaleja. Tämä onnistuu esim. laskemalla keskiarvo ja pyöristämällä se nollan desimaalin tarkkuudelle, jonka jälkeen tulos pitää muuttaa merkkijonoksi ja poistaa sen lopusta merkit ”.0”. Pyydetyn tulosteen voi tehdä myös luennoilla esitellyllä format-lauseella.

L06T2 Ohjelman esimerkkiajo:

```
Anna luettavan tiedoston nimi: L06T2D2.txt
Tiedostossa oli 12356 nimeä ja 101032 merkkiä.
Keskimäärin nimen pituus oli 7 merkkiä.
Kiitos ohjelman käytöstä.
```

L06T3: Tiedoston tietojen testaaminen, palindromit

Tässä tehtävässä käsitellään palindromeja, joita käsiteltiin aiemmin tehtävässä L03T2. Tällä kertaa testattavat merkkijonot luetaan tiedostosta, merkkijonot pitää tarkistaa ja niitä pitää muokata ennen palindromitestausta ja löytyneet palindromit kirjoitetaan toiseen tiedostoon. Toteuta ohjelma pääohjelmana kahden aliohjelman kanssa, joista toinen tarkistaa luetun rivin ja toinen kirjoittaa palindromit tiedostoon.

Kysy pääohjelman alussa luettavan ja kirjoitettavan tiedoston nimet ja varmista, että kirjoitettava tiedosto on tyhjä avaamalla se kirjoitustilassa sekä sulkemalla se. Tämän jälkeen avaa luettava tiedosto, lue sieltä rivi ja tarkista aliohjelmassa, onko rivi palindromi. Jos kyseessä on palindromi, kutsu kirjoitus-aliohjelmaa ja tulosta näytölle tieto siitä, oliko rivillä palindromi vai ei. Katso tulosteen ja tiedoston tarkat esitysmuodot alla olevista esimerkeistä. Toteuta tiedoston lukeminen yksi rivi kerrallaan toistorakenteessa ja toista yllä olevat vaiheet kaikille riveille.

Luettujen rivien tarkistus tulee tehdä omassa aliohjelmassa. Rivillä tulee olla yli 2 merkkiä, jotta kyseessä voisi olla palindromi. Tiedostossa voi olla kommenttirivejä, jotka alkavat #-merkillä ja näitä ei huomioida. Lisäksi numeroita sisältäviä rivejä ei hyväksytä palindromiksi.

Tiedostossa olevat palindromit ovat lauseita, joissa on isoja ja pieniä kirjaimia, välilyöntejä sekä välimerkkejä. Nämä kaikki tulee poistaa tai vaihtaa samaan esitysmuotoon, jotta Pythonin yhtäsuuruustesti toimii ja palindromit voi tunnistaa. Siistimisessä kannattaa käydä kaikki rivillä olevat merkit läpi ja tarkistaa, onko kyseessä numero vai hyväksyttävä merkki. Kaikki hyväksyttävät merkit kannattaa laittaa uuteen merkkijonoon, jolle tehdään lopuksi palindromi-testi ja jos kyseessä on palindromi, kannattaa siistitty merkkijono palauttaa kutsuvaan ohjelmaan jatkotoimenpiteitä varten.

Palindromien kirjoittaminen tiedostoon tehdään omassa aliohjelmassa, joka saa parametrinä kirjoitettavan tiedoston nimen ja rivin sekä alkuperäisessä luetussa muodossa että siistityssä muodossa. Tiedosto tulee avata utf-8 -koodauksella, jotta ääkköset eivät mene sekaisin, uudet rivit kirjoitetaan aina tiedoston loppuun ja tiedosto suljetaan kirjoittamisen jälkeen ennen aliohjelmasta paluuta.

Varsinainen testiaineisto on tiedostossa `L06T3D1.txt`, joka sisältää edellä kuvatun mukaisia tekstirivejä. Ohjelman eri toimintojen testaamisen helpottamiseksi Moodlessa on myös tiedosto `L06T3D1.txt`, jolla voit testata ohjelmasi toimintaan, ks. esimerkki alla.

Merkkien käsittelyssä kannattaa katsoa `isdigit()`, `isalpha()` ja `lower()`-jäsenfunktioita. Kannattaa muistaa, että `readline()`-funktio palauttaa merkkijonossa myös rivinvaihtomerkin, eli "apina" luetaan tiedostosta merkkijonona "apina\n". Rivinvaihtomerkkiä ei tule huomioida numero-tarkastelussa eikä palindromitestauksessa ja vain rivinvaihtomerkin sisältävä rivi lopettaa tiedoston lukemisen.

Ohjelman esimerkkiajo:

```
Anna luettavan tiedoston nimi: L06T3D2.txt
Anna kirjoitettavan tiedoston nimi: L06T3T2.txt
OK: 'innostunutsonni'
OK: 'Aatu osaa soutaa.'
Ei OK: 'diipadaapa'
Kiitos ohjelman käytöstä.
```

L06T3 Esimerkki ohjelman kirjoittamasta tiedostosta:

```
innostunutsonni
----> innostunutsonni
Aatu osaa soutaa.
----> aatuosaasoutaa
```

L06T4: Valikkopohjainen ohjelma, laajeneva tehtävä

Edellisellä viikolla muutimme valikkopohjaisen ohjelman muodostumaan aliohjelmista ja tällä kertaa muokkaamme ohjelman lukemaan ja kirjoittaman tekstitiedostoja. Ohjelmassa tulee olla esimerkkiajon mukaisesti 4 valintaa eli tiedostonimien kysyminen, tietojen analyysi, tulosten kirjoittaminen tiedostoon sekä lopetus. Pääohjelman ja valikko-ohjelman runko on sama kuin edellisellä viikolla, joten ne saa suoraan sieltä.

Tiedostonimiä pitää kysyä 2 eli luettavan ja kirjoitettavan tiedoston nimi. Koska toiminto on sama, mutta sanamuodot vähän erilaiset, tee nimen kysymiseen yksi aliohjelma, johon välitetään sopiva kehote ja aiempi tiedoston nimi. Näin samaa aliohjelmaa voi kutsua kaksi kertaa tiedostojen nimien selvittämiseksi.

Analyysi-vaiheessa tulee selvittää tiedoston pienin ja suurin arvon. Luettavassa tiedostossa on yhden kuukauden jokaisen päivän aikana kertyneiden askelten lukumäärä eli luvut ovat positiivisia kokonaislukuja tai 0 eikä niiden lukumäärästä ole tietoa etukäteen. Pienin ja suurin luku kannattaa etsiä samalla tavalla, mutta molemmille tehtäville kannattaa tehdä oma aliohjelma selkeyden vuoksi. Esimerkiksi etsittäessä pienintä päivittäistä arvoa kannattaa apumuuttuja alustaa arvolla None ja sen jälkeen verrata tiedostosta luettua arvoa aiempaan arvoon ja muuttaa sitä tarpeen mukaan sekä lopuksi palauttaa pienin löytynyt arvo kutsuvaan ohjelmaan.

Tiedoston kirjoittava aliohjelma saa parametrinä tiedoston nimen sekä pienimmän ja suurimman arvon, jotka kirjoitetaan tiedostoon esimerkkiajon mukaisesti.

Ohjelman lukemat tiedot löytyvät tiedostosta `L06T4D1.txt`, jossa on päivittäinen askelmäärä eli yksi kokonaisluku per rivi ja rivi jokaiselle kuukauden päivälle. Moodlessa on myös tiedosto `L06T4D2.txt`, jossa on kaksi lukua, joiden avulla voit helposti varmistua ohjelmasi oikeasti toiminnasta. Tallennettava tiedosto `L06T4T1.txt` löytyy myös Moodlesta, jotta voit tarvittaessa varmistua sen muodosta. Tässä tehtävässä voidaan olettaa, että analysoitavassa tiedostossa on vähintään yksi rivi.

L06T4 Ohjelman esimerkkiajo:

```
Tämä on valikkopohjainen ohjelma, jossa voit valita haluamasi toiminnon.
Valitse haluamasi toiminto:
1) Anna tiedostonimet
2) Analysoi
3) Kirjoita tiedosto
0) Lopeta
Anna valintasi: 1
Anna tiedostonimet
Luettavan tiedoston nimi on ''.
Anna uusi nimi, enter säilyttää nykyisen: L06T4D2.txt
Kirjoitettavan tiedoston nimi on ''.
Anna uusi nimi, enter säilyttää nykyisen: L06T4T2.txt

Valitse haluamasi toiminto:
1) Anna tiedostonimet
2) Analysoi
3) Kirjoita tiedosto
0) Lopeta
Anna valintasi: 2
Suoritetaan analyysi

Valitse haluamasi toiminto:
1) Anna tiedostonimet
2) Analysoi
3) Kirjoita tiedosto
0) Lopeta
Anna valintasi: 3
Kirjoitetaan tulokset tiedostoon

Valitse haluamasi toiminto:
1) Anna tiedostonimet
2) Analysoi
3) Kirjoita tiedosto
0) Lopeta
Anna valintasi: 0
Lopetetaan

Kiitos ohjelman käytöstä.
```

Luettavan tiedoston muoto, `L06T4D2.txt`:

```
10000
12345
```

Kirjoitettavan tiedoston muoto, `L06T4T2.txt`:

```
Analyysin tulokset ovat seuraavat:
Datan pienin arvo on 10000.
Datan suurin arvo on 12345.
```

L06T5: Päivittäisen sähkönkulutuksen laskenta

Tee Python-ohjelma, joka lukee tiedostosta sähkönkulutustietoja, laskee niistä päivittäiset kulutustiedot ja tallettaa tulokset toiseen tiedostoon. Sähkönkulutusdata on tiedostossa tunnin välein siten, että päivä- ja yö-kulutustiedot ovat omissa sarakkeissaan. Tässä tehtävässä tiedoston kentät ovat vakiolevyisiä, jolloin ne voidaan erottaa rivistä merkkijonon leikkauksena. Tiedostoa kirjoitettaessa kenttien leveyksien tulee myös olla vakioita, jotta dataa voi tarvittaessa käsitellä myös jatkossa samalla tavalla.

Ohjelman pääohjelmassa tulee kysyä sekä luettavan että kirjoitettavan tiedoston nimi. Kirjoitettava tiedosto tulee tyhjentää ja sinne tulee kirjoittaa otsikkorivi siten, että päivämäärä-kentän leveys on 10 merkkiä ja se on tasattu oikealle, ja kulutus-kentän leveys on 5 merkkiä ja se on tasattu oikealla.

Pääohjelmassa kulutustiedot luetaan riveittäin, niistä lasketaan päiväkulutus ja kun se on selvillä, kutsutaan tallenna-aliohjelmaa, joka kirjoittaa tiedot tiedostoon. Päiväkulutusta laskiessa kannattaa huomata, että luettavat tiedot ovat aikajärjestyksessä ja kaikki saman päivän kulutustiedot tulee laskea yhteen. Kun yhden päivän tiedot ovat valmiina, kirjoitetaan ne tiedostoon ja lasketaan seuraavan päivän tiedot jne. Laskenta tulee siis toteuttaa toistorakenteessa ja sekä ensimmäistä että viimeistä arvoa laskiessa tulee olla tarkkana, että ne menevät oikein.

Tietojen tallennus tiedostoon tehdään aliohjelmassa, joka saa parametrinä kirjoitettavan tiedoston nimen, päivämäärän ja ko. päivän kulutustiedon. Kulutustiedot kirjoitetaan yllä olevan mukaisessa muodossa tiedoston viimeiseksi riviksi.

Tässä tehtävässä tiedoston rivit on muotoiltu tarkasti, joten siihen liittyvä ohjeet kannattaa tarvittaessa kerrata yltä. Tiedostoa kirjoitettaessa helpoin tapa saada haluttu muoto tulosteeseen on käyttää `format`-lausetta luentojen mukaisesti. Päivämäärää käsitellään tässä tehtävässä merkkijonona, josta voidaan ottaa leikkauksia ja niitä voidaan vertailla tarpeen mukaan jne.

Moodlessa on tiedostot `L06T5D1.txt` ja `L06T5D2.txt`, joista ensimmäisessä on yhden päivän kulutustiedot sekä seuraavan päivän ensimmäinen tieto, kun taas toisessa on koko vuoden data.

Luettavan tiedoston muoto, `L06T5D1.txt`:

```
Aika;Yö (kWh);Päivä (kWh)
01.01.2021 00;3.49;0.00
01.01.2021 01;2.03;0.00
```

Kirjoitettavan tiedoston muoto, `L06T5T1.txt`:

```
        Pvm:  kWh
01.01.2021:   70
02.01.2021:    4
```

Ohjelman esimerkkiajo:

```
Anna luettavan tiedoston nimi: L06T5D1.txt
Anna tallennettavan tiedoston nimi: L06T5T1.txt
Kiitos ohjelman käytöstä.
```