Ohjelmoinnin peruskurssi Y1

CS-A1111

Tiedostojen käsittely

Oppimistavoitteet: tämän videon jälkeen

- Tiedät, miten ohjelman käsitelemää tietoa voi säilyttää ohjelman suorituskerrasta toiseen käyttämällä hyväksi tiedostoja.
- Osaat kirjoittaa ohjelman, joka lukee rivejä tekstitiedostosta.

Tiedostot

- Tiedostojen käsittelyä tarvitaan esimerkiksi seuraavissa tilanteissa:
  - Ohjelman käsittelemiä tietoa halutaan säilyttää ohjelman suorituskerrasta toiseen (esim. puhelinluettelo, opiskelijarekisteri).
  - Halutaan, että käyttäjän ei tarvitse syöttää ohjelman lähtötietoja jokaisella suorituskerralla (esim. mittaussarjan parametrit).
  - Ohjelman on käsiteltävä jonkun muun ohjelman tuottamaa dataa.
- Ohjelma lukee tarvittavat lähtötiedot tiedostosta.
- Jos ohjelma tekee tietoihin muutoksia ja muuttuneita tietoja halutaan käyttää seuraavalla suorituskerralla, ohjelma kirjoittaa muuttuneet tiedot tiedostoon.

Tiedoston avaaminen

- Kerrottaan, mitä “fyysistä” tiedostoa ohjelman muuttuja vastaa.
- Samalla tietokoneen käyttöjärjestelmä varautuu käsittelemään ko. tiedostoa.
- Tätä kutsutaan *tiedoston avaamiseksi*, esimerkki

```
tiedostomuuttuja = open("teksti.txt","r")
```

- Ensimmäinen parametri on käsiteltävän tiedoston nimi käyttöjärjestelmässä. Tarvittaessa sen pitää sisältää polku tiedoston hakemistoon.
- Toinen parametri kertoo tiedoston käsittelytavan:
  - `r` lukeminen
  - `w` kirjoittaminen, vanha sisältö häviää
  - `a` kirjoittaminen, kirjoitetaan vanhan sisällön perään.

Poikkeusten käsittely tiedostojen kanssa

- Tiedostoja avatessa voi tapahtua poikkeuksia, joten poikkeusten käsittely try–except-rakenteella on syytä tehdä aina, kun luetaan tiedostosta tai kirjoitetaan tiedostoon.
- Tiedoston avaaminen lukemista varten voi aiheuttaa poikkeuksen, jos tiedostoa ei ole tai sitä ei pystytä jostain muusta syystä lukemaan.
- Tiedoston käsittelyyn liittyvien poikkeusten "ylätyyppinä" on `OSError`-tyyppinen poikkeus.

Tiedoston sulkeminen

- Kun tiedoston lukeminen päättyy, tiedosto pitää sulkea `close`-käskyllä:

  ```
  tiedostomuuttuja.close()
  ```

Rivin lukeminen tekstiä sisältävästä tiedostosta

- Jos muuttuja `tiedostomuuttuja` viittaa lukemista varten avattuun tiedostoon, niin siitä voi lukea rivin kerrallaan metodin `readline` avulla seuraavasti:

  ```
  luettu_rivi = tiedostomuuttuja.readline()
  ```

  Luettu rivi sisältää myös sen lopussa olevan rivinvaihtomerkin.
- Seuraava `readline`-käsky lukee tiedoston seuraavan rivin jne.
- Jos tiedosto on jo luettu loppuun ja kutsutaan `readline`-metodia, se palauttaa arvona tyhjän merkkijonon `""`

Esimerkkiohjelma tiedoston lukemisesta

```
def main():
    try:
        lahtotiedosto = open("tekstia.txt", "r")
        rivi = lahtotiedosto.readline()
        while rivi != "":
            print(rivi)
            rivi = lahtotiedosto.readline()
        lahtotiedosto.close()
    except OSError:
        print("Virhe tiedoston lukemisessa. Ohjelma paattyy.")


main()
```

Rivinvaihtomerkistä

- Edellisen videon ohjelma tulostaa ylimääräisen tyhjän rivin jokaisen rivin jälkeen.
- Tämä johtuu siitä, että tiedostoista luettujen rivien lopussa on rivinvaihtomerkki.
- Jos ylimääräiset rivinvaihdot voidaan välttää poistamalla rivinvaihtomerkki rivin lopusta ennen tulostamista.
- Yksi tapa poistaa rivinvaihtomerkki on käyttää metodia `rstrip`. Se poistaa kuitenkin myös muut "tyhjät merkit" rivin lopusta.

Parannettu versio tiedostonlukuohjelmasta

```
def main():
    nimi = input("Anna luettavan tiedoston nimi: ")
    try:
        lahtotiedosto = open(nimi, "r")
        rivi = lahtotiedosto.readline()
        while rivi != "":
            rivi = rivi.rstrip()
            print(rivi)
            rivi = lahtotiedosto.readline()
        lahtotiedosto.close()
    except OSError:
        print("Virhe tiedoston", nimi,
              "lukemisessa. Ohjelma paattyy.")

main()
```