CT10A0013 Ohjelmointi Pythonilla

L04: Toistorakenteet

Uolevi Nikula

Päivän asiat

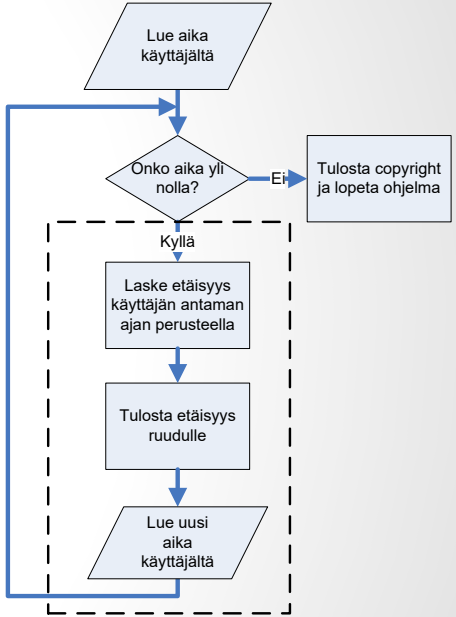
- Käytäntö
  - Alkuehtoinen toisto – while
  - Askeltava toisto – for
  - range-funktio
  - Loppuehtoinen toisto while-käskyllä
  - Muuttujien roolit: tuoreimman säilyttäjä, askeltaja, kokooja
  - Algoritmi ja vuokaavio
- Lopuksi

Käytäntö

Toistolauseet
1. Alkuehtoinen toisto – while
2. Askeltava toisto – for
3. Loppuehtoinen toistorakenne while-käskyllä

Alkuehtoinen toisto

- Toistettavan osan alussa oleva ehto määrää toiston jatkumisen (*alkuehtoinen* toistolause)
  - **Katkoviivalla merkittyä osaa ei välttämättä suoriteta kertaakaan!**
  - Lopetusehto: “aika ei ole > 0” eli “aika on <= 0”
- Muuttuja `Aika` säilyttää käyttäjän viimeksi antamaa syötettä – *tuoreimman säilyttäjä*
- Toistojen määrää ei ole rajoitettu



while –rakenne

```
Aika = int(input("Anna kulunut aika (s): "))
while (Aika > 0):
    Etaisyys = Aika * AanenNopeus / 1000
    print("Salama löi", Etaisyys, "km:n päässä.")
    Aika = int(input("Anna kulunut aika (s): "))
print("Kiitos ohjelman käytöstä.")
```

Vertailulauseke

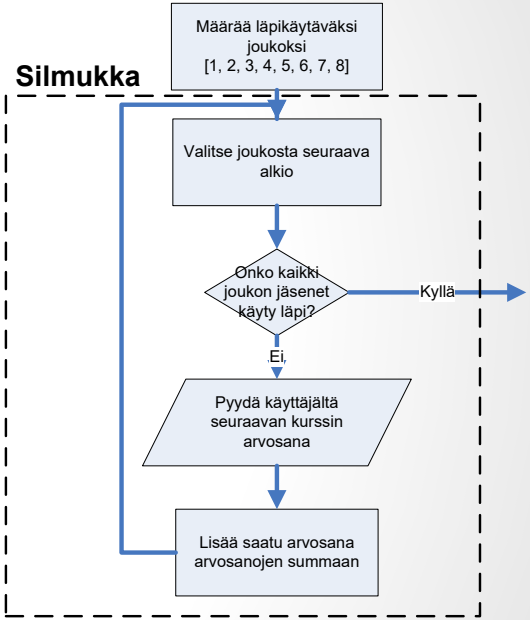
Toistettava osuus

- Lue: ”Mikäli muuttujan `Aika` arvo suurempi kuin 0, suorita sisennetty osa. Muutoin jatka seuraavasta sisentämättömästä kohdasta.”
- Joka kerta kun sisennetty osa on suoritettu, tutki muuttujaa `Aika` uudelleen ja toista sisennettyä osaa niin kauan kuin `Aika` > 0
  - `Aika`-muuttujan rooli on *tuoreimman säilyttyjä*

Askeltava toisto

- Silmukkaan liittyy muuttuja, joka saa jokaisella silmukan kierroksella uuden arvon – sen rooli on **askeltaja**
- Askeltaja käy systemaattisesti läpi ennalta määrätyt arvot eli sekvenssin
  - Oheisessa tapauksessa

    1, 2, 3, 4, 5, 6, 7, 8



Askeltava toistorakenne: for

- Joissain tapauksissa läpikäytävät arvot ovat etukäteen tiedossa
- Tällöin luonnollinen tapa toteuttaa toisto on käyttää for-rakennetta

```
Summa = 0
for i in [1,2,3,4]:
#Varattu sana for, Muuttuja i, in, Sekvenssi, Kaksoispiste
    Syote = input("Anna "+str(i)+". luku: ")
    Luku = int(Syote)
    Summa = Summa + Luku
print("Antamiesi lukujen summa on", Summa)
```

- `Summa`-muuttujan rooli on *kokooja*
  - Kokooja-muuttuja `Summa`: alustus, arvon kasvattaminen ja käyttö lopuksi

Joukon automaattinen luominen - range

- Python käsky `range` luo järjestetyn joukon eli sekvenssin
- range-funktion yleinen muoto on `range(alku, loppu, askel)`
  - *alku* määrittää joukon ensimmäisen luvun
  - *loppu* määrittää joukon viimeisen luvun – **jota ei oteta mukaan!**
  - *askel* määrittää lukujen välin, oletusarvo +1
  - Vertaa merkkijonon leikkaukset, esim. Sana[::]
- Käsky range(1, 4) muodostaa sekvenssin 1,2,3
  - Käsky range(4) muodostaa sekvenssin 0,1,2,3
- Teknisesti lopputulos näyttää listalta ja sitä voidaan ajatella listana, mutta tarkkaan ottaen toteutus ei ole lista vaan *listatyyppinen ratkaisu*
  - Älä yritä muuttaa sekvenssiä läpikäynnin aikana!

Toistorakenteiden eroista 1

- Pythonissa on kaksi erilaista toistorakennetta
  - Alkuehtoinen toisto while
  - Askeltava toisto for
- Molempien tehtävä on sama eli toistorakenteen toteutus
- Rakenteilla on periaatteellisia eroja
  - while-rakenteessa ohjelmoija on vastuussa kaikista rakenteen osista eli alustuksesta, lopetusehdon tarkastuksesta sekä mahdollisesta askeltajan päivityksestä; lopetusehtoja voi olla useita (vrt. vertailulauseke L03 ja Boolen operaattorit)
  - for-rakenteessa ohjelmoija on vastuussa läpikäytävän sekvenssin määrittelystä, mutta sekvenssin sisäisestä toteutuksesta ei ole tietoa eikä sitä voi muuttaa; lopetusehtona on, että sekvenssi on käyty läpi

Toistorakenteiden eroista 2

- Käytännössä while-rakenteen voi yleensä korvata for-rakenteella ja päinvastoin
  - Joissain tapauksissa periaatteelliset ongelmat voivat johtaa huonosti ylläpidettävään, virhealttiiseen ja/tai suorituskyvyltään huonoon lopputulokseen
  - for-lauseen toteutukset ovat erilaiset Python versiossa 2 ja 3
- for-rakenteen rajoitteita
  - Käy läpi vain yhtä sekvenssiä
  - Perustuu aina sekvenssin läpikäynti, ei voi käyttää useita lopetusehtoja
  - Kierrosten lukumäärä eli sekvenssi tiedettävä etukäteen, ei voi muuttaa läpikäynnin aikana

Loppuehtoinen toisto

- Alkuehtoinen toisto aloittaa ehdon tarkistuksella ja lopettaa siihen
  - Toistorakenteeseen kuuluvia asioita ei aina tehdä kertaakaan
- Loppuehtoinen toisto
  - Suorittaa toistoon kuuluvat asiat
  - Toistorakenteen lopussa tarkistetaan lopetusehto ja lopetetaan tarvittaessa
  - Toistorakenteen asiat tehdään aina kerran
- Pythonissa ei ole omaa käskyä loppuehtoiselle toistolle
  - Toistorakenne voidaan toteuttaa while-rakenteella

Toistorakenteiden laajennuksia

- Toistorakenteissa esiintyy usein myös seuraavia laajennoksia
  - **break** – lopetus, toiston tekeminen lopetetaan ja *hypätään pois toistorakenteesta*
  - **continue** – keskeytys, toiston tekeminen keskeytetään ja *siirrytään seuraavalle kierrokselle*
- Joskus hyödynnetään myös seuraavia laajennoksia
  - **else** – haarautuminen, jos/kun toistoa ei enää tehdä
  - **pass** – ohitus, ei suoriteta (erityisesti haarautumisessa)
- Toistorakenteiden yhteydessä näistä yleisimpiä ovat break ja continue –käskyt, ks. tarkemmin ohjelmointioppaasta

Toistorakenteen perustapaukset

```
# while, alkuehtoinen toistorakenne
i = 0
while (i < 10):
    print(i)
    i = i + 1

# for, askeltava toistorakenne
for i in range(10):
    print(i)
```

Alkuehtoinen toistorakenne, variaatioita

```
# while, alkuehtoinen toistorakenne – input-lause 2 kertaa, 4 riviä
i = int(input("Anna luku (-1 lopettaa): "))
while (i != -1):
    print(i)
    i = int(input("Anna luku (-1 lopettaa): "))

# while ikisilmukka ja poistuminen break:lla – vain 1 input, 5 riviä
while (True):
    i = int(input("Anna luku (-1 lopettaa): "))
    if (i == -1):
        break
    print(i)
```

Loppuehtoinen toistorakenne

```
# Loppuehtoinen toistorakenne while-rakenteella
while (True):
    i = int(input("Anna luku (-1 lopettaa): "))
    print(i)
    if (i == -1):
        break # Toistorakenteesta poistutaan sen lopuksi
```

Valikkopohjainen ohjelma toistolla

1. L03: valinta
   - Valikko
   - Valintarakenne
   - Koodilohko
2. L04: toisto
   - Alustus
   - Lopetusehto-Tarkistus
   - Toistorakenne
   - Lopetusehto-Muutos

```
# Muuttujien alustus
Valinta = 1

# Ohjelman toiminnallinen osuus
while (Valinta != 0):
    # Valikko
    print("Valitse haluamasi toiminto:")
    print("1) Anna merkkijono")
    print("2) Tulosta merkkijono etuperin")
    print("3) Tulosta merkkijono takaperin")
    print("0) Lopeta")
    Syote = input("Anna valintasi: ")
    Valinta = int(Syote)

    # Valintarakenne
    if (Valinta == 1):
        Merkkijono = input("Anna merkkijono: ")
    elif (Valinta == 2):
        print(Merkkijono)
    elif (Valinta == 3):
        print(Merkkijono[::-1])
    elif (Valinta == 0):
        print("Lopetetaan")
    else:
        print("Tuntematon valinta, yritä uudestaan.")
    print()

# Ohjelman lopetus
print("Kiitos ohjelman käytöstä.")
#########################################################
# eof
```

Lopuksi

Osaamistavoitteet

Ohjelmointivideot

Osaamistavoitteet

- Alkuehtoinen toisto – while
- Askeltava toisto – for
- Loppuehtoinen toisto toteutettuna while-rakenteella
- Sekvenssin automaattinen luonti – range
- Muuttujien roolit: tuoreimman säilyttäjä, askeltaja, kokooja
- Algoritmi ja vuokaavio
- Valikkopohjainen ohjelma toistolla

Ohjelmointivideot

1. video
   - Toistorakenteet ja niiden periaatteet, perusrakenteet ja erikoistapauksia
   - for, while, if, break, continue, while ja monta ehtoa
2. video
   - Tyypillisiä tapoja toteuttaa toistorakenteita ohjelmissa: parillisten lukujen etsintä, salaman etäisyys, keskiarvon laskenta 4 eri tavalla
3. video
   - Valikkopohjainen ohjelma toistolla

Täydennyksiä oppaan lukuun 4

Tyyliohjeita pienille Python-ohjelmille

Oppaan esimerkit ja käsitellyt asiat

Pienen Python-ohjelman tyyliohjeet 1

- **Toistorakenne for**. for-lausetta käytetään, kun tiedetään läpikäytävä sekvenssi, esim. luvut x:stä y:hyn, ikävuodet x-y jne., sillä näin saadaan käytyä kaikki alkiot läpi kompaktissa ja selkeässä muodossa. Sekvenssi luodaan tyypillisesti range-funktiolla, jos se ei tule valmiina jostain muualta kuten esimerkiksi tiedostosta.
- **Toistorakenne while**. while-lausetta käytetään, kun ei tiedetä etukäteen läpikäytäviä tietoja, vaan ne kysytään käyttäjältä, luetaan tiedostosta tai muuta vastaavaa. while on sopiva toistorakenne myös silloin, kun lopetuspäätökseen vaikuttaa monta tekijää, esim. ikä, paino ja sukupuoli tai liikennevalojen väri ja nopeus tms.
- **Toistorakenteen lopetus**. for-lause loppuu, kun etukäteen määritelty sekvenssi on käyty läpi. while-lause loppuu, kun alkuehtoisen toistorakenteen aloitusehto on epätosi. Molemmat toistorakenteet voi lopettaa kesken suorituksen break-käskyllä, joka siirtää ohjelman suorituksen jatkumaan toistorakennetta seuraavaan käskyyn.
- **Toistorakenteen loppuosan ohitus continue**. continue-käsky mahdollistaa toistorakenteen lopun ohittamisen eli suorittamatta jättämisen ja toistorakenteen suorituksen siirtämisen sen alkuun. Tällöin for-lauseessa jatketaan sekvenssin seuraavalla arvolla ja while-lauseessa suoritetaan lopetusehdon arviointi.
- **Askeltaja-muuttuja**. Toistorakenteissa käytetään usein askeltaja-muuttujaa, joka on perinteisesti kirjan i – tai i ja j, jos askeltajia on kaksi. Askeltaja-rooli on tyypillisesti niin selkeä, että siinä käy poikkeuksellisesti yhden merkin muuttujanimi. Luonnollisesti myös kuvaavia muuttujanimiä voi käyttää kuten Ika, Lukumaara tms.

Käsitellyt asiat oppaan luvussa 4

- Toistorakenne while: Esimerkki 4.1, 4.2, 4.5, 4.7
- Toistorakenne for: Esimerkki 4.3, 4.4, 4.6 (2 sisäkkäistä for:ia), 4.7
- Funktio range
- Laajennokset break, continue, else: Esimerkki 4.4, 4.5, 4.6