CT10A0013 Ohjelmointi Pythonilla

L03: Valintarakenne

Uolevi Nikula

Päivän asiat

- Teoria
  - Erilaisia näkökulmia ohjelmointiin
- Käytäntö
  - Vertailu ja loogiset operaattorit
  - Ehdollinen suorittaminen ja valintarakenne
  - Koodilohkot ja kommentointi
  - Boolen algebra
  - Valikkopohjainen ohjelma
- Lopuksi
- Täydennyksiä oppaan lukuun 3

Teoria

Erilaisia näkökulmia ohjelmointiin

- Käyttäjä
- Määrittelijä
- Suunnittelija
- Ohjelmoija
- Testaaja

Käyttäjän näkökulma ohjelmaan

- Perinteisissä peleissä on usein paljon sääntöjä
  - Ohjekirjassa mainitut *eksplisiittiset* säännöt
  - Pelaajien perimätietona oppimat *implisiittiset* säännöt, esim. lautapeleissä nappulat liikkuvat vain pelilaudalla (vrt. tammi tai shakki)
- Jotta tietokone voisi noudattaa sääntöjä, on ne kaikki koodattava suoritettavaan ohjelmaan
  - Perinteisellä tietokoneella ei ole ”yleistietoa”
  - Perinteinen tietokone ei opi
  - Perinteinen tietokone noudattaa suoritettavaa ohjelmaa juuri niin kuin se on kirjoitettu
- Tietokoneohjelmaan on siis kirjoitettava kaikki säännöt siten, että ne näkyvät ohjelmakoodissa *eksplisiittisesti*
  - Koodissa asiat näkyvät ohjelmoijalle
  - Käyttäjää kiinnostaa yleensä vain ohjelman näkyvä toiminta
  - Ohjelmalle ja ohjelmoijalle on kerrottava ohjelman toiminta kaikki yksityiskohdat mukaan lukien

Ohjelman määrittelystä

- Ohjelman määrittelyn tehtävänä on selvittää, miten ohjelman tulee toimia erilaisissa tilanteissa
- Esimerkiksi mitä tekstinkäsittelyohjelman tulisi tehdä kirjoitusvirheiden kohdalla?
  - Ei mitään erikoista
  - Pysähtyä ja kieltäytyä uusien merkkien vastaanottamisesta kunnes virhe on korjattu
  - Merkitä väärin kirjoitettu sana korostamalla se esim. alleviivauksella ja jatkaa muutoin normaalisti
  - Lopettaa ohjelman suoritus kokonaan
  - Jotain muuta – mitä?
- Huomaa, että ei ole olemassa yhtä ainutta toimintavaihtoehtoa vaan tilanteet ja käyttäjien toiveet vaihtelevat
  - Käyttäjän pitää kertoa, miten ohjelman tulee toimia erilaisissa tilanteissa käyttäjän näkökulmasta

Määrittely, suunnittelu ja toteutus

- Mikäli ohjelma on määritelty täydellisesti, voivat ohjelman suunnittelija ja toteuttaja keskittyä toteuttamaan tehtyä määrittelyä
  - *Määrittelijän* tehtävä on selvittää, miten ohjelman tulee toimia erilaisissa tilanteissa ja mitä toiminnallisuutta siinä on oltava erilaisista näkökulmista katsottaessa sekä dokumentoitava se ohjelman kehitystä varten
  - *Suunnittelijan* tehtävä on löytää sopivat tietorakenteet (kokonais- tai desimaaliluku, merkkijono, …) ja algoritmit (toimintamallit), ts. ohjelman sisäiset rakenteet, siten että se vastaa määrittelyssä kirjattuja vaatimuksia
  - *Ohjelmoijan* tehtävä on toteuttaa ohjelma siten, että se toimii tehokkaasti ja on helposti ylläpidettävissä eli muutettavissa

Ohjelman testaamisesta

- Valmis ohjelma on testattava, ettei sinne jää virheitä
  - Kaupallisessa ohjelmistokehityksessä testauksen tekevät erikoistuneet henkilöt eli *testaajat*
- Virheitä on erilaisia, esimerkiksi
  - Ohjelma ei toimi käyttäjän haluamalla tavalla – tällöin kyseessä on *määrittelyvirhe*
  - Tietorakenteet tai algoritmit eivät sovi ohjelmaan – *suunnitteluvirhe*
  - Ohjelmassa on kirjoitusvirhe, nollalla jako tms. – *ohjelmointivirhe*
- Virheen aiheuttaja ei ole aina itsestään selvää
  - Ohjelmistotoimittajat puhuvat usein ominaisuuksista tai piirteistä, koska virheitä ei haluta myöntää
  - Myös käyttäjän epätarkka tarpeiden määrittely voi johtaa virheisiin
  - Käyttäjän tarpeiden muuttuminen tekee asiasta entistä vaikeamman

Käytäntö

Vertailu

Valintarakenne

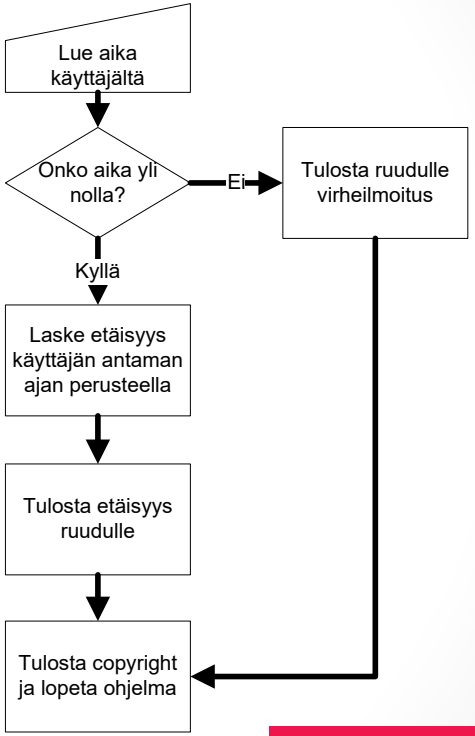
Valintarakenne-esimerkki

Ohjelman tehtävä on laskea etäisyys nopeuden ja ajan perusteella. Ohjelma **valitsee seuraavan toiminnon käyttäjän antamaan lukuun perustuen**.
Ohjelman runko eli algoritmi on seuraava:

1. Pyydä käyttäjältä aika ja sijoita se muuttujaan
2. Tehdään vertailu: Onko aika yli nolla?
3. Mikäli vertailu on tosi, suoritetaan kohta 5; mikäli vertailu on epätosi, suoritetaan kohta 4
4. Tulosta virheilmoitus, siirry kohtaan 7
5. Laske etäisyys
6. Tulosta vastaus
7. Tulosta copyright ja lopeta ohjelma

Valintarakenne-vuokaavio

- Valintarakenteen käyttö ohjelman haarautumisen toteuttamiseen
- Algoritmi esitetty vuokaaviona



Vertailulausekkeet ja valintarakenne

- Vertailulausekkeen idea on selvittää esim. ovatko x ja y yhtäsuuria?
- Vertailu edellyttää, että vertailtavat asiat (muuttujat) ovat samaa tyyppiä
  - Numeroita ja numeroarvoja sisältävät muuttujat ovat keskenään samaa tyyppiä
  - Merkkijonoja voidaan vertailla samoilla operaattoreilla kuin numeroita. Järjestysperusteisiin palataan myöhemmin
- Vertailujen perusteella ohjelma voi valita, mitä kohtia ohjelmasta suoritetaan tai jätetään suorittamatta (*valintarakenne*)

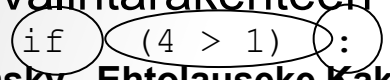
Vertailu: loogiset operaattorit

Matematiikka	Python	Tarkoitus
=	==	Yhtäsuuri
≠	!=	Erisuuri
≥	>=	Suurempi tai yhtäsuuri
≤	<=	Pienempi tai yhtäsuuri
>	>	Suurempi
<	<	Pienempi

- Katso Python-opas Taulukko 3.1

Valintarakenne

- Valintarakenteen osat ovat seuraavat



Käsky Ehtolauseke Kaksoispiste

- Ohjelman kannalta ehtolausekkeen arvo on joko
  - tosi, True eli 1

  tai
  - epätosi, False eli 0
- Ehtolausekkeen arvo lasketaan/evaluoidaan ensin ja sen perusteella päätetään, suoritetaanko seuraavana oleva sisennetty koodiosa
  - Ehtolausekkeen ollessa epätosi (0) koodia ei suoriteta
  - Ehtolausekkeen arvo on totuusarvo tosi tai epätosi
  - Vertailulausekkeen arvo on myös tosi tai epätosi, esim. 2>1 on tosi

Valintarakenteen variaatioita

```
# Ehdollinen koodi - if
Luku = 1
if (Luku == 1):
    print("Luku oli 1")

# Valintarakenne – if-else
Luku = 2
if (Luku == 1):
    print("Luku oli 1")
else:
    print("Luku oli", Luku)
```

Monihaarainen if-elif-else –rakenne

```
Kirjain = 'C'
if (Kirjain == 'A'):
    print("Tulosta A")
elif (Kirjain == 'B'):
    print("Tulosta B")
else: # muussa tapauksessa
    print("Tulosta", Kirjain)
print("Tulostuu aina ehtolauseen jälkeen.")
```

- `if` –haara on pakollinen ehdollisessa rakenteessa
- `elif` -haaroja voi olla 0, 1 tai monta
- `else` -haara on aina vapaaehtoinen

Boolen algebra / logiikka

- Ehtolauseke voi koostua useista osista, joita yhdistellään Boolen operaattoreilla JA, TAI ja EI
  - Pythonissa nämä ovat **and**, **or** ja **not** –operaattoreita
  - Esim. “if ((Vari == VIHREA) and (Nopeus < 60)): …” tulkitaan siis “mikäli väri on vihreä ja nopeus alle 60 niin …”
- Mikäli ehtolausekkeessa on useita osia, katso tarkemmin ohjelmointioppaasta, ohjelmointivideoista tai wikistä https://fi.wikipedia.org/wiki/Boolen_algebra

Ohjelmalohkot

- if –rakenteessa on keskeisenä osana suoritettavien käskyjen eli *koodilohkon*, tai ohjelmalohkon, tunnistaminen
- Esimerkiksi valintarakenne

```
if (ehtolauseke):
    print("ehdollinen käsky 1")
    print("ehdollinen käsky 2")
print("ei enää ehdollinen käsky")
```

- **Sisennys** ja **kaksoispiste** määräävät suoritettavat käskyt eli kyseessä olevan ohjelmalohkon
  - Muissa kielissä lohkoja erotetaan esim. *begin* ja *end* sanoilla tai suluilla, esim. { ja }
  - **Pythonissa oleellista on sisennys**

Ohjelmalohkojen kommentointi

- Ohjelmalohkon ajatus on, että se sisältää loogisesti yhteen kuuluvaa koodia ja toimintoja
- Näin ollen jokaiseen lohkoon kuuluu luonnollisena osana selitys siitä, mitä siinä tapahtuu eli *kommentti*
- Normaalisti tulkki/kääntäjä ei välitä kommenteista vaan ne on suunnattu vain koodia lukevalle ihmiselle helpottamaan sen ymmärtämistä
- Katso kommentoinnista lisää Python-oppaan liitteestä 4

Ohjelman kommentointi

- Ohjelman normaali kommentointi rakentuu seuraavista osista

```
# Ohjelman hallintoa: kuka tehnyt, milloin, tiedostonimi
# Ohjelman tarkoitus
# Muuttujien esittely kommentteina
# * esim. HeTu = henkilötunnus
# Kiintoarvojen ja muuttujien alustukset ts. suoritettava ohjelma alkaa tästä
# Syötteiden kysyminen käyttäjältä
# Tiedonkäsittely eli laskenta tms.
# Tulostus
# Lopetus
```

- Näiden lisäksi eri asioihin keskittyvät lohkot kannattaa kommentoida eli kuvata selväkielisesti

Valikkopohjainen ohjelma

- Merkkipohjaiset ohjelmat perustuvat tyypillisesi valikkoon, jossa on listattuna toiminnot numeron kanssa ja käyttäjä valitsee toiminnon numerolla

  1) Kysy nimi
  2) Tulosta nimi
  0) Lopeta

  Anna valintasi: 1
- Valikkopohjainen ohjelma oli normaali ratkaisu esim. matkapuhelimissa silloin kun niiden käyttöliittymät olivat merkkipohjaisia
- Tällä kurssilla tehtävät ohjelmat perustuvat usein tähän valikko-ajatukseen eli se kannattaa opetella
  - Yllä näkyy valikko, josta käyttäjä valitsee haluamansa toiminnon
  - Ohjelmassa on valintarakenne ja ohjelman suoritus haarautuu annetun valinnan mukaisesti

Lopuksi

Osaamistavoitteet/Teoria

Osaamistavoitteet/Käytäntö

Teoria: Erilaisia näkökulmia ohjelmointiin

- Tämän kurssin tavoitteena on käydä läpi, mitä ohjelmien tekeminen edellyttää käytännössä
- Vaikka kurssilla tehdään pieniä ohjelmia, ovat näkökulmat ohjelmointiin samat kuin isoja ohjelmia tehtäessä:
  - *Käyttäjä* tarvitsee ohjelman johonkin tehtävään
  - *Määrittelijän* tehtävä on selvittää, mitä tarpeita käyttäjällä on ja miten tämä haluaa ohjelman toimivan eri tilanteissa
  - *Suunnittelijan* tehtävä on löytää näihin tarpeisiin sopivat tekniset ratkaisut ja muodostaa niistä toimiva kokonaisuus
  - *Ohjelmoija* toteuttaa ohjelman
  - *Testaaja* varmistaa, että ohjelma toimii käyttäjän haluamalla tavalla ja ettei siinä ole virheitä
- Isoissa projekteissa nämä roolit jaetaan eri henkilöille, mutta pienten ohjelmien yhteydessä usein sama henkilö tekee kaikki nämä tehtävät

Käytäntö

- Ohjelmointia
  - Vertailulauseke, loogiset operaattorit
  - Valintarakenne, ehdollinen suorittaminen ja haarautuminen
  - Koodilohkot ja kommentointi
  - Boolen algebra/logiikka
  - Valikkopohjainen ohjelma

Täydennyksiä oppaan lukuun 3

Tyyliohjeita pienille Python-ohjelmille

Oppaan esimerkit ja käsitellyt asiat

Pienen Python-ohjelman tyyliohjeet 1

- **Sisennykset**. Python tunnistaa koodilohkon sisennyksistä. Sisennysten pitää olla aina systemaattisesti samalla tavoin ja hyvä tyyli on sisentää koodia 4 välilyöntiä. Monissa editoreissa tämä onnistuu sarkainnäppäimellä eli tabulaattorilla
  - Huom. Sarkainmerkki voi olla 4 välilyönnin levyinen, mutta kyseessä on yksi merkki. Molemmat tavat/merkit ovat systemaattisesti käytetty hyväksyttäviä, mutta Python ei hyväksy niitä molempia samassa ohjelmassa
- **Valikkopohjainen ohjelma**. Valikon käyttö edellyttää kahta erillistä ohjelman osaa
  - Valikko. Tulosta valikko sopivassa kohdassa ohjelmaa ja kysy sen jälkeen valinta. Tallenna valinta muuttujaan kokonaislukuna
  - Valintarakenne. Valintarakenteen tulee olla yhteensopiva valikon kanssa eli kaikille valikon eri kohdille tulee olla haarat valintarakenteessa

Käsitellyt asiat oppaan luvussa 3

- Koodin sisentäminen
- Valintarakenne if: Esimerkki 3.1, 3.2, 3.4
- Ehtolausekkeet ja loogiset operaattorit: Taulukko 3.1, 3.2, Esimerkki 3.3
- Operaattorien suoritusjärjestys
- Boolen-arvot: Esimerkki 3.3, 3.4
- Tyypillisiä virheilmoituksia/sisennykset