hyväksymispäivä arvosana

arvostelija

COBOL ja Python: Datan kapselointi

Erkki Heino, Tero Huomo, Eeva Terkki

Helsinki 20.2.2013

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Sisältö

1 Datan kapselointi

1.1 COBOL

COBOLissa tietoa voidaan ryhmitellä muodostamalla tietueita (record) ja ryhmiä (group). `DATA DIVISION` -osiossa käytettävät muuttujien tasonumerot (level number) määrittävät, minkälaisia kokonaisuuksia muuttujat muodostavat. Seuraavassa esimerkissä on kuvattu opiskelijatietue.

```
DATA DIVISION.
WORKING-STORAGE SECTION.
01 OPISKELIJA.
   05 NIMI.
       10 ETUNIMI  PIC A(10).
       10 SUKUNIMI PIC A(10).
   05 NUMERO       PIC 9(10).
   05 SUKUPUOLI    PIC A.
       88 MIES     VALUE "M".
       88 NAINEN   VALUE "N".
```

`OPISKELIJA` on tietue, joka koostuu nimestä, opiskelijanumerosta ja sukupuolesta. Nimi on jaettu kahteen osaan: etunimi ja sukunimi.

Tasonumerot kuvaavat tiedon hierarkian, mutta tietyillä numeroilla on erityismerkitys. Hierarkiaa kuvaavat tasonumerot 02-49. Tasonumero 01 tarkoittaa, että kyseessä on tietue. Merkitsemällä tasonumeroksi 66 voidaan edeltävälle muuttujalle antaa toinen nimi. 77 kuvaa muuttujaa, joka ei ole osa mitään rakennetta. Tason 88 avulla voidaan määrittää muuttujaan liittyviä ehtoja: yllä olevassa esimerkissä opiskelijan sukupuoli on mies, jos muuttujan `SUKUPUOLI` arvo on M, ja nainen, jos arvo on N [Ans74, s. I-84]. Esimerkiksi voitaisiin tarkistaa, onko opiskelija mies:

```
IF MIES OF OPISKELIJA
```

Todellisuudessa tietue ei ole juuri muuta kuin merkkijono, jonka eri osat kuvaavat tietueen kenttiä. Tietueelle voidan antaa arvot kenttä kerrallaan:

```
MOVE "ARTO" TO ETUNIMI OF OPISKELIJA.
```

Toisaalta koko tietueelle voidaan antaa arvo kerralla:

```
MOVE "ARTO      WIKLA     1234567890M" TO OPISKELIJA.
```

Esimerkin tietue sisältää kerrallaan yhden opiskelijan tiedon — tällaisenaan koodissa ei siis voida käsitellä useampaa opiskelijaa samanaikaisesti. Rakenne voidaan määritellä taulukoksi (table), jolloin samaa rakennetta voidaan käyttää useamman samanrakenteisen tiedon käsittelemiseen ilman saman rakenteen uudelleenmäärittelyä.

Muuttuja määritellään taulukoksi `OCCURS`-lauseen avulla. `OPISKELIJA` voitaisiin muuttaa taulukoksi muokkaamalla määrittelyä seuraavasti:

```
01 OPISKELIJA OCCURS 5 TIMES.
```

Esimerkiksi taulukon toiseen opiskelijaan viitattaisiin indeksillä 2: `OPISKELIJA(2)`. Taulukon sisällä voidaan myös määritellä sisäkkäisiä taulukoita.

1.1.1 Olio-orientoitunut COBOL

COBOL ei alunperin ole olio-ohjelmointikielin, vaan tieto kapseloitiin tietueisiin ja ryhmiin. COBOLista on kuitenkin kehittynyt olio-ohjelmoinnin mahdollistava laajennos, Object-oriented COBOL, joka on lisännyt kieleen olio-ohjelmoinnin tärkeimmät piirteet.

Teron muistilistaa epämääräisiin lähteisiin:

- http://supportline.microfocus.com/documentation/books/sx20books/opintr.htm

- http://www.objs.com/x3h7/oocobol.htm

- http://supportline.microfocus.com/documentation/books/nx60/opconc.htm

(IBM:n Enterprise COBOL tukee myös olioita, mutta OO COBOLista eroten estää moniperinnän. - http://pic.dhe.ibm.com/infocenter/pdthelp/v1r1/index.jsp?topic=

1.2 Python

Pythonissa muuttujien arvot ovat viitteitä olioihin, ja kaikki esitetään olioina tai olioiden välisinä suhteina [].

Pythonin tarjoamia sekvenssityyppejä ovat muun muassa lista (list) ja monikko (tuple). Muita kielen tietorakenteita ovat esimerkiksi joukko (set) ja hajautustaulu (dictionary).

Listan alkiot erotellaan pilkuilla ja lista ympäröidään hakasuluilla. Listan alkioiden ei tarvitse olla keskenään samaa tyyppiä [Pyt13b]. Listan sisältö on muokattavissa

sen luomisen jälkeen.

Pythonissa on myös funktionaaliselle ohjelmoinnille tyypillisiä funktioita, jotka ovat hyödyllisiä listojen käsittelyssä: `map`, `filter` ja `reduce`. Listakomprehensioilla (list comprehension) voidaan

Monikkoa ympäröivät kaarisulut, ja alkiot erotellaan toisistaan pilkuilla. Toisin kuin listan tapauksessa, kerran luotua monikkoa ei voi muokata [Pyt13a]. Monikko voi kuitenkin sisältää muuttuvia tietotyyppejä, kuten listoja. Monikon sisällä voi myös olla toisia monikkoja.

Seuraavassa esimerkissä puretaan monikko:

```
>>> monikko = ("a", 123, 3.14)
>>> x, y, z = monikko
>>> x
"a"
>>> y
123
>>> z
3.14
```

Joukko on järjestämätön tietorakenne, ja se voi sisältää saman alkion korkeintaan kerran. Joukkojen käsittelemiseen käytössä ovat matemaattiset joukko-operaatiot, kuten yhdiste, leikkaus ja erotus. Python tukee myös joukkokomprehensioita (set comprehension), jotka toimivat listakomprehensioiden tapaan:

```
>>> joukko = {x for x in "nakkivanukas" if x not in "aeiouyäö"}
>>> joukko
set(['k', 'v', 's', 'n'])
```

Esimerkissä luodaan joukko, joka sisältää merkkijonon `nakkivanukas` ne merkit, jotka eivät ole pieniä vokaaleja. Kukin merkki esiintyy joukossa vain kerran.

2 Etuja ja haittoja

Lähteet

Ans74 *American national standard programming language COBOL*. ANSI, New York, USA, 1974.

Pyt13a The python tutorial – data structures, `http://docs.python.org/2/tutorial/datastructures.html`. [19.2.2013].

Pyt13b The python tutorial – an informal introduction to python – lists, `http://docs.python.org/2/tutorial/introduction.html#lists`. [19.2.2013].