hyväksymispäivä arvosana

arvostelija

COBOL ja Python

Erkki Heino

Tero Huomo

Eeva Terkki

Helsinki 30.1.2013

HELSINGIN YLIOPISTO

Tietojenkäsittelytieteen laitos

Sisältö

1 Tausta ja tyyli

Tässä kappaleessa selostetaan lyhyesti COBOLin ja Pythonin taustat: kielten syntymisen syyt, kielten toteutustavat sekä niiden tyypilliset käyttökohteet. Lisäksi molemmista kielistä esitetään lyhyt esimerkkiohjelma, joka kuvaa tyypillisen ohjelmointitavan sekä kielen ulkoasun.

1.1 COBOL

COBOL on vuonna 1959 kehitetty proseduraalinen ohjelmointikieli, joka kuuluu varhaisimpien ohjelmointikielten joukkoon. COBOL-kielen kehitti Yhdysvaltain puolustusministeriön sponsoroimana muun muassa viranomaisista ja tietokonevalmistajien edustajista koostunut komitea, jonka pyrkimys oli luoda yleinen ohjelmointikieli liike-elämän tarpeisiin [Sam81, s. 200]. Nimi COBOL tulee sanoista "Common Business-Oriented Language". COBOLista pyrittiin tekemään ohjelmointikieli, joka muistuttaisi läheisesti tavallista englannin kieltä [Cou02]. Siten ohjelmakoodi olisi itseään kommentoivaa, ja sitä voisivat lukea ohjelmoijien lisäksi myös esimerkiksi yrityksen johtokunta, työn valvojat sekä ohjelman käyttäjät.

COBOL on edelleen laajalti käytössä kaupallisissa, hallinnollisissa sekä finanssialan järjestelmissä. Vuonna 1997 Gartner Group raportoi jopa 80% yritysmaailman ohjelmakoodista olevan kirjoitettu COBOLilla [KCH00]. COBOLia on laajennettu muutamia kertoja. Viimeisin COBOLin versio on vuodelta 2002, ja kyseinen päivitys lisäsi COBOLiin mahdollisuuden olio-ohjelmoinnille. COBOL-kääntäjiä on olemassa useita, ja COBOLilla voi ohjelmoida lähes kaikille käyttöjärjestelmille [Cou02].

Seuraavan esimerkin COBOL-ohjelma tulostaa syötelukua seuraavan Fibonaccin luvun:

```
IDENTIFICATION DIVISION.
PROGRAM-ID. FIBONACCI.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
77 FIRST-NUMBER         PIC 9(5) COMP VALUE 0.
77 SECOND-NUMBER        PIC 9(5) COMP VALUE 1.
77 RESULT               PIC 9(5) COMP.
77 USER-INPUT           PIC 9(3).

PROCEDURE DIVISION.
    DISPLAY "ANNA LUKU (-999 - 999)"
    ACCEPT USER-INPUT FROM CONSOLE
    IF USER-INPUT LESS THAN 0 THEN
        MOVE 0 TO RESULT
    ELSE
        PERFORM UNTIL RESULT GREATER THAN USER-INPUT
            ADD FIRST-NUMBER TO SECOND-NUMBER GIVING RESULT
            MOVE SECOND-NUMBER TO FIRST-NUMBER
            MOVE RESULT TO SECOND-NUMBER
        END-PERFORM
    END-IF
    DISPLAY "PIENIN ANTAMAASI LUKUA SUUREMPI"
    DISPLAY "FIBONACCIN LUKU ON ", RESULT.
```

Esimerkissä huomataan ensimmäisenä ohjelman rakenne. Ohjelma jaetaan neljään

osioon (division): tunnisteosioon, ympäristöosioon, tieto-osioon sekä proseduuriosioon. Tieto-osiossa määritellään ohjelman käyttämät muuttujat. Proseduuriosio määrittelee ohjelman toiminnan.

Ohjelma pyytää käyttäjältä luvun väliltä 0-999. Koska muuttuja `USER-INPUT`, johon käyttäjän syöte tallennetaan, on määritelty kolmen merkin pituiseksi etumerkittömäksi numeroksi, on käyttäjän annettava luku kolmen merkin pituisena; muuten ohjelma kaatuu. Esimerkiksi luvun 10 tapauksessa on kirjoitettava "010".

Jos käyttäjän syöte on nollaa pienempi, tallennetaan tulokseksi nolla — muussa tapauksessa lasketaan Fibonaccin lukuja, kunnes löydetään syötettä suurempi luku. Muuttujat `FIRST-NUMBER` ja `SECOND-NUMBER` ovat aina kaksi peräkkäistä Fibonaccin lukua ja ne on alustettu tieto-osiossa ensimmäisiksi Fibonaccin luvuiksi 0 ja 1. `PERFORM`-käskyllä alkavassa silmukassa lasketaan Fibonaccin lukuja, kunnes luku on suurempi kuin käyttäjän syöte. Lopuksi tulos näytetään käyttäjälle.

1.2 Python

Python on yleiskäyttöinen, moniparadigmainen ohjelmointikieli, joka tukee paitsi imperatiivista ja olio-ohjelmointia myös funktionaalista ohjelmointia. Guido van Rossum kehitti Pythonin alun perin Amoeba-nimistä hajautettua käyttöjärjestelmää varten, mutta halusi kielen olevan yleisesti laajennettavissa [Pyt13d]. Pythonin ensimmäinen versio julkaistiin vuonna 1991, ja nykyään kieli on jakautunut versioihin 2 ja 3.

Python on tulkattava kieli, ja sille on tuki yleisimmillä käyttöjärjestelmillä. Python on kaikille ilmainen avoimen lähdekoodin ohjelmointikieli. Pythonin standardikirjastot tukevat valmiiksi esimerkiksi merkkijonokäsittelyä, Internet-protokollia sekä käyttöjärjestelmärajapintoja. Laajennettavuutensa ansiosta Pythonille on tarjolla runsaasti myös kolmansien osapuolten laajennoksia. Pythonin kehitys on pitkälti

yhteisöpohjaista [Wik13b].

Python on TIOBE Softwaren [Tio13] mukaan vuoden 2013 alussa maailman kahdeksanneksi suosituin ohjelmointikieli. Suurista yrityksistä Pythonia käyttävät esimerkiksi Google, Cern ja NASA. Tyypillisiä käyttökohteita kielelle ovat matemaattinen ja tieteellinen ohjelmointi, web-ohjelmointi, tietokantaohjelmointi sekä työpöytäsovellukset [Pyt13b]. Pythonia käytetään myös opetuskielenä.

Seuraavassa esimerkissä on Pythonilla toteutettu syötelukua seuraavan Fibonaccin luvun tulostava ohjelma:

```
def main():
   i = int(raw_input("Anna luku: "))
   fibonacci(i)

def fibonacci(n):
   if n < 0:
       b = 0
   else:
       a, b = 0, 1
       while (b <= n):
           a, b = b, a + b
   print "Pienin syöttämääsi lukua suurempi Fibonaccin luku: ", b

main()
```

Esimerkkiohjelma on toteutettu määrittelemällä kaksi funktiota, `main` ja `fibonacci`. Funktiossa `main` kysytään käyttäjältä syöte, joka muunnetaan kokonaisluvuksi ja tallennetaan muuttujaan `i`. Tämän jälkeen kutsutaan funktiota `fibonacci` antaen parametriksi muuttujan `i` arvo. Funktio alustaa ensin muuttujien `a` ja `b` arvoiksi

0 ja 1. Tämän jälkeen kielelle tyypillinen silmukkarakenne vaihtaa `a`:n arvoksi `b`:n ja laskee muuttujaan `b` muuttujien `a` ja `b` alkuperäisten arvojen summan. Silmukan päättyessä käyttäjälle näytetään vastaus. Funktioiden määrittelyn jälkeen ohjelma suoritetaan kutsumalla funktiota `main`.

2 Alkiorakenne ja syntaksi

COBOL- ja Python-koodi eroavat ulkoasultaan paljon toisistaan. Tässä kappaleessa tutustutaan kummankin kielen syntaksiin ja alkiorakenteeseen.

2.1 COBOL

COBOLin formaalin syntaksin ja semantiikan määrittelee riippumaton ANSI COBOL -komitea. COBOLin nykyinen standardi ei kuitenkaan ole ilmaisjakelussa.

COBOL-ohjelmakoodi on jaettu neljään osioon (division), joiden tulee olla alla olevassa järjestyksessä [Cle08, s. 29].

1. `IDENTIFICATION DIVISION`
2. `ENVIRONMENT DIVISION`
3. `DATA DIVISION`
4. `PROCEDURE DIVISION`

Osiot alkavat osion tunnisteella. Osiot voidaan jakaa jaksoihin (section), joiden nimi identifioi jakson. Jaksot koostuvat kappaleista ja kappaleet puolestaan lauseista (sentence). Lauseet erotetaan toisistaan pisteellä ja rivinvaihdolla. Lauseita on kolmea tyyppiä: kääntäjää ohjaavat lauseet, ehdolliset lauseet ja imperatiiviset lauseet.

Lauseet koostuvat pienemmistä lauseista (statement), jotka alkavat COBOLin verbeillä ja sisältävät sanoja. Pienemmät lauseet loppuvat seuraavan verbin tai pisteen kohdalla. Sanat ovat COBOLille kelpaavia merkkijonoja. COBOL74 tunnistaa yhteensä 52 merkkiä.

Koodi koostuu merkkijonoista (character string) ja näiden erottimista (separator). Merkkijono on joko sana, literaali, picture-merkkijono (picture character string) tai kommentti [Cle08, s.41-43]. Sanat muodostavat lauseita (statement), jotka muodostavat virkkeitä (sentence). Virkkeet voivat kuulua kappaleisiin (paragraph) ja kappaleet osastoihin (section) [Cle08, s. 201]. Varattuja sanoja on COBOL-kielessä yli 400 [Cle08, s. 719-725].

Muuttujat määritellään `PICTURE`-lausekkeen avulla. Fibonacci-esimerkissä määriteltiin muuttuja `FIRST-NUMBER`:

```
77 FIRST-NUMBER          PIC 9(5) COMP VALUE 0.
```

Luku ennen muuttujan nimeä on tasonumero (level number). Tietueita (record) voidaan muodostaa käyttämällä eritasoisia muuttujia. Tässä on käytetty tasonumeroa 77, joka tarkoittaa, että mitään hierarkiaa ei ole käytössä [Cle08, s. 112].

`PIC` on synonyymi sanalle `PICTURE`, jota seuraa merkkijono, joka kuvaa muuttujan tyypin ja koon. Merkintä `9(5)` tarkoittaa viiden merkin pituista numeerista merkkijonoa. Sama voitaisiin ilmaista myös näin: `99999`. 9 tarkoittaa numeerista merkkiä; X tarkoittaa mitä tahansa merkkiä [Cle08, s. 162], joten `XXXXX` tarkoittaa mitä tahansa viiden sallitun merkin pituista merkkijonoa. Tyypin määrittelyä seuraava `COMP` ilmaisee muuttujan käyttötarkoitusta: tässä tapauksessa käyttötarkoitus on laskennallinen (computational) [Cle08, s. 184]. `VALUE 0` alustaa muuttujan arvoksi nollan.

COBOL tukee 13 eri tietotyyppiä, joiden lisäksi eri kaupalliset kääntäjät tukevat muutamia muita tietotyyppejä [Wik13a]. Kielen viimeisimmässä versiossa on lisätty

tuet myös muun muassa olio-ohjelmoinnille, liukuluvuille, totuusarvoille ja osoittimille (pointers).

COBOLissa on huomattavan paljon erilaisia säädöksiä muuttujien nimeämisessä ja syntaksissa. Ohjelmoijan nimeämät muuttujat eivät esimerkiksi saa olla yli 30 merkin pituisia, ja ne saavat sisältää vain merkkejä A-Z, 0-9 ja -.

2.2 Python

Python Language Reference [Pyt13c] kuvailee kielen syntaksin ja semantiikan. Pythonin versiossa 2.7 on 31 varattua sanaa, joita ei voi käyttää vakioiden tai nimettyjen arvojen niminä. Python sisältää hyvin pitkälti samoja varattuja sanoja kuin Java, esimerkiksi tyypillisimmät toistorakenteet, ehdollisuuteen liittyvät avainsanat sekä funktioiden ja niiden paluuarvojen määrittelyyn liittyvät sanat. Lisäksi Pythonissa on 35 operaattoria, joita kuvaa merkki tai merkkiyhdistelmä.

Pythonissa tunnisteet ovat tarkoitettu esimerkiksi muuttujan, funktion tai luokan nimeämiseen [Gui13]. Tunnisteen täytyy alkaa aakkosellisella kirjaimella tai alaviivalla, jota seuraa nolla tai useampi kirjain, alaviiva tai numero. Tunnisteissa ei voi käyttää välimerkkejä. Lisäksi Python on merkkikokoriippuvainen. Isoilla tai pienillä kirjaimilla kirjoitetut sanat ovat siis kaksi eri tunnistetta.

Pythonissa loogisia lauseita erottaa toisistaan rivinvaihto, paitsi jos rivinvaihdot ovat sulkujen sisällä, kuten listaa tai hajautustaulua alustettaessa. Loogisen lauseen voi jakaa useammalle riville käyttämällä kenoviivaa rivin lopussa.

Jos lausetta seuraa uusi koodilohko, lause päättyy kaksoispisteeseen ja itse lohko merkitään sisentämällä. Sisentämiseen käytetään joko välilyöntejä tai tabulaattoria. Pythonin tyyliopas opastaa neljän välilyönnin käyttöön [Pyt13a]. Kahden sisentämistyylin sekoittaminen voi johtaa virhetilanteisiin.

Seuraavassa esimerkissä silmukan sisällä kasvatetaan muuttujan `i` arvoa ja tuloste-

taan se. Kun silmukka päättyy, tulostetaan käyttäjälle merkkijono `"loppu"`.

```
while (i < 5):
   i += 1
   print i
print "loppu"
```

Tyypillinen silmukkarakenne alkaa käytettävän silmukkatyypin avainsanalla ja silmukan lopetusehdolla, jota seuraa kaksoispiste. Silmukan sisällä suoritettava ohjelmakoodi sisennetään. Silmukka päättyy, kun koodin sisennys on samalla tasolla kuin silmukan aloittava koodi.

Python tukee muun muassa seuraavia tietotyyppejä:

- Numeeriset tietotyypit
  - Kiinteän pituinen kokonaisluku (`int` versiossa 2, puuttuu versiosta 3)
  - Rajattoman pituinen kokonaisluku (`long` versiossa 2, `int` versiossa 3)
  - Liukuluku (`float`)
  - Kompleksiluku (`complex`)
- Merkkijono (`str`, versiossa 2 myös `unicode`)
- Lista (`list`)
- Read-only-lista (`tuple`)
- Hajautustaulu (`dictionary`)

3 Etuja ja ongelmia

COBOL-kielen monisanainen ilmaisu jakaa mielipiteitä. Toisaalta COBOL-koodi onnistuu itsensä kommentoimisessa, ja myös muut kuin ohjelmoijat voivat ymmärtää,

mitä ohjelma tekee. Toisaalta ohjelmat kasvavat tarpeettoman pitkiksi. COBOLilla kirjoitettuihin ohjelmiin verrattuna esimerkiksi Python-ohjelmat ovat paljon kompaktimpia. COBOL-kieltä on kritisoinut muun muassa E.W. Dijkstra: "The use of COBOL cripples the mind; its teaching should, therefore, be regarded as a criminal offence"[Dij13].

Python pyrkii selkeyteen ja luettavuuteen. Koska koodilohkot määritellään käyttäen sisennystä, ohjelman koodilohkot jäsentyvät väistämättä selkeästi. Siinä missä esimerkiksi Javalla ohjelmoitaessa koodilohkot voidaan aaltosulkeiden vuoksi kirjoittaa vaikkapa samalle riville, ei koodin sisentämiseltä Pythonilla ohjelmoitaessa voida välttyä. Ongelmatilanteita sisentämisestä voi aiheutua esimerkiksi silloin, jos yhdistetään kahta eri koodia, joista toinen on sisennetty välilyönneillä ja toinen tabulaattoria käyttäen.

Lähteet

Cle08 ClearPath Enterprise Servers, Application Development Solutions, *COBOL ANSI-74 Programming Reference Manual, Volume 1: Basic Implementation*, 2008.

Cou02 Coughlan, M., Introduction to cobol, `http://www.csis.ul.ie/cobol/course/COBOLIntro.htm#part1`. [29.1.2013].

Dij13 How do we tell truths that might hurt?, `http://www.cs.utexas.edu/users/EWD/transcriptions/EWD04xx/EWD498.html`. E.W. Dijkstra Archive. [30.1.2013].

Pyt13a Pep 8 – style guide for python code, `http://www.python.org/dev/peps/pep-0008/#indentation`. [30.1.2013].

KCH00 Kizior, R., Carr, D. ja Halpern, P., Does cobol have a future? *The Proceedings of the Information Systems Education Conference 2000*, 2000.

Pyt13b Applications for python, `http://www.python.org/about/apps/`. [26.1.2013].

Pyt13c The python language reference, `http://docs.python.org/2.7/reference/index.html`. [30.1.2013].

Pyt13d Why was python created in the first place, `http://docs.python.org/2/faq/general#why-was-python-created-in-the-first-place`. [20.1.2013].

Gui13 Python quick guide, `http://www.tutorialspoint.com/python/python_quick_guide.htm`. [27.1.2013].

Sam81 Sammet, J. E., History of programming languages i. ACM, 1981, luku The early history of COBOL, sivut 199–243, URL http://doi.acm.org/10.1145/800025.1198367.

Tio13 Tiobe programming community index for january 2013, http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html. [26.1.2013].

Wik13a Cobol: Data types, http://en.wikipedia.org/wiki/COBOL#Data_types. [30.1.2013].

Wik13b Wikipedia: Python (programming language), http://en.wikipedia.org/wiki/Python_(programming_language). [30.1.2013].