Python: ajalugu ja lühiiseloomustus

Python on interpreteeriv üldotstarbeline programmeerimiskeel, mida algselt arendati skriptikeeleks. Python võimaldab programmeerida nii objektorienteeritult kui ka protseduurselt. Siiski osa tööd jääb tegemata, kui püüda rangelt objektideta hakkama saada.

Sel kursusel läheneme programmeerimisele ja Pythonile peamiselt klassideta ja objektideta, kuid ei ole pääsu ka objektide lühikesest ülevaatest. Muidu jääks osa töid tegemata (nt andmevahetus failidega või stringide töötlemine).

Pythoni loojaks on Guido van Rossum Hollandist *Stichting Mathematisch Centrumist* (Hollandi Rahvuslik Matemaatika ja Arvutiteaduse Instituut). Keel on loodud 1991. aastal. Autor kutsub keelt skriptikeeleks, kuid praeguseks on tegemist siiski võimsama programmeerimiskeelega. Guido van Rossum alustas Pythoni projektiga 1989 aasta lõpus ja keele eellaseks võib pidada (tema enda sõnade kohaselt) keelt ABC, mis oli 1980-tel loodud keel õpetamiseks. Python oli esialgu suunatud Unix-i/C kasutajatele ja mitmed keele omadused on laenatud C keelest.

Python on oma nime saanud briti naljameeste telesarja "Monty Pythoni lendav tsirkus" järgi.

Python (täpsemalt tema interpretaator) on vaba tarkvara (enamus väljalaskeid vastab GPL-litsentsile). See tähendab muuhulgas, et Pythoni interpretaatori eest ei pea raha maksma ja seda võib kasutada ka kommertstoodete tegemiseks. Pythoni interpretaatorit võib leida erinevatele platvormidele: Unixid, Linux, Macintosh, MS-DOS, OS/2 ja Windows. Pikemat aega kehtisid erinevad Pythoni 2.x versioonid. 2008 aasta lõpust on väljas uus versioon Python 3.0, mida kutsutakse ka “Python 3000” või “Py3K”. Tasub tähele panna, et võrreldes eelnevate 2.x versioonidega tähendab see keeles mitmeid muudatusi ning 2.x vaimus kirjutatud programmid ei pruugi Py3K-ga korrektselt töötada. Ka Guido van Rossum rõhutab, et *“Py3K is the first ever intentionally backwards incompatible Python release“*. Järeldus: kõik internetist ja mujalt kättesattuvad Python-programmid ei pruugi käima minna või töötada oodatud viisil.

Pythoni interpretaatoriga on kaasas väike graafiline töökeskkond IDLE ning lisaks saab Pythonis kirjutatud programme käivitada käsurealt. Windows'i ja teiste operatsioonisüsteemide jaoks on loodud lisaks erinevaid töökeskkondi: PythonWin, PyScripter jms. Need vajavad eraldi installeerimist.

Keele tuuma *(core Python)*, tema süntaksi ja semantikat peetakse minimalistlikuks, kuid **lisateegid** *(library)* ja laiendused avardavad tunduvalt keele võimalusi. Suur osa vajaminevatest võimalusest (funktsioonidest, meetoditest) on saadaval **standardteegis** *(standard library)*.

Kohustuslik lausete treppimine ja selle kasutamine lausete grupeerimiseks pärineb keelest **ABC**. Treppimine tähendab seda, et osa programmi lauseid kirjutatakse taandega, mille abil rõhutatakse programmi struktuuri ja nn juhtlausete (tingimus, tsükkel) kasutamist. Treppimine tõstab tunduvalt programmide loetavust inimese jaoks ning seda on soovitatud ja rakendatud juba mitukümmend aasatat alates struktuurprogrammeerimise põhimõtete kirjapanekust. Inimloetavuse tõstmiseks on ka lähtutud põhimõttest, et liigne võimaluste rohkus samade asjade kirjapanekuks ei ole hea. Loetavus on oluline inimeste jaoks, kes sama koodiga (või ka oma vana koodiga) töötama peavad. Eelneva ideega on seotud kirjavahemärkide-tehtemärkide kasutamine samal viisil, nagu seda ka nt koolimatemaatikas tehakse. Samuti võib see olla põhjuseks, miks 3.0 versioon varasematega ühilduvaks tehtud ei ole.

Väidetavalt on Pythonit kerge õppida oma selge struktuuri tõttu. Ja ehkki ta on oma olemuselt OO (objektorienteeritud) keel, saab esimesed sammud teha ka objektideta.

Siin kohal ei tohi aga unustada, et programmeerimiskeel on vaid abivahend käskude edastamiseks arvutile ning põhiraskus läheb siiski mõtlemise-algoritmimise peale ehk milliseid käske ja mil viisil

arvutile andma peaks, et ta soovitud viisil käituks.

Python on küll interpreteeritav keel (ja seetõttu võrreldes kompileeritavate keeltega aegalsem, st antud keeles kirjutatud programm töötab aeglasemalt võrreldes mõnes kompileeritavas keeles kirjutatud programmiga), kuid interpreteerimine ei toimu mitte lähtekoodi järgi, vaid eelnevalt on inimese kirjutatud lähtekood "tõlgitud" interpretaatori poolt nn baitkood ja see on juba lähedasem masinkoodile. Selles osas sarnaneb Python Javaga.