Objektno programiranje
⟨ Uvod, interpreteri, tipovi podataka ⟩

Tihomir Žilić, Mario Essert, Vladimir Milić

Sveučilište u Zagrebu
Fakultet strojarstva i brodogradnje

Zagreb, 2021./2022.

Objektno orijentirani programski jezik

Stil programiranja temeljen na objektima i njihovoj međusobnoj komunikaciji.
Objekt je skup funkcija i podataka koji zajedno čine jednu funkcionalnu cjelinu.

Postoje i drugi stilovi programiranja: proceduralno (strukturalno), funkcionalno, logičko, itd.

Područja primjene objektno orijentiranog jezika

1. Algorithm and Data Structure
2. Biology, Finance
3. Computer Security
4. Engineering, Mechatronics, Robotics, Automation
5. Geographic Information System
6. GUI Development
7. Graphics and Computer Vision
8. High Performance Computing
9. Machine Learning and Artificial Intelligence
10. Network programming
11. Natural Language Processing
12. Numerical Programming and Data Mining
13. Raspberry Pi, Microcontrollers
14. Web Development
15. ....

O Pythonu

Osmislio Guido van Rossum $\sim$ 1989. godine kao nasljednika programskog jezika ABC. Prva javno objavljena verzija je Python 0.9.0 iz 1991. godine. Ime je dobio po Monty Python's Flying Circus (TV show).

- Interpreterski jezik (Python virtual machine - čita bytecode), a može biti i compiler (bytecode), .py/.pyc,
- objektno-orijentiran, više razine (tipovi podataka),
- podržava proceduralno i funkcionalno programiranje,
- čitljiv i jednostavan (fokus na problem, a ne sintaksu jezika),
- besplatan i otvorenog koda,
- portabilan i neovisan o operativnom sustavu (PC, mobiteli, tableti),
- proširiv i na druge jezike C, C++, Java, itd.,
- posjeduju standardnu biblioteku prema motu *batteries included*,

O Pythonu

- koriste ga Google, Yahoo, NASA, MIT, itd.,
- koristi se također za rapid prototyping, WEB, kompjutorske igrice, bioinformatiku, izradu aplikacija, programiranje mikrokontrolera i mikroračunala. Više pogledati na https://www.python.org/about/success/.
- Verzije Pythona: Python 1.x od 1994., Python 2.x (2.7) od 2000., Python 3.x (3.8) od 2008.

Kompatibilnost između verzija 2.x i 3.x

Python 3.x uvodi:

- bolju Unicode podršku,
- mijenja neke naredbe u funkcije, npr. exec $\rightarrow$ exec(), print$\rightarrow$print(),
- mijenja neke funkcije u nove xrange()$\rightarrow$range(), raw_input()$\rightarrow$ input(),
- mijenja operatore dijeljenja, npr. $5/2 \rightarrow 2$ u verziji 2.x, $5/2 \rightarrow 2.5$ u verziji 3.x.

Kôd translatori 2to3 i 3to2.
Biblioteke za kompatibilnost (Py2, Py3 compatibility library):
>>> import six

Pythonove inačice

Interpreteri:

- CPython (C implementacija),
- Jython (Java implementacija, pokreće na Java Virtual Machine),
- IronPython (.NET Runtime implementacija),
- PyPy (alternativna Python implementacija s JIT compilerom, brža od Python ili CPython),
- MicroPython (Python pokretan na mikrokontroleru).

Compiler:

- Cython (compiler za Python i CPython)

[0]Kompajliranjem datoteke *myscript.py* dobije se bytecode datoteka *myscript.pyc*.
Iz interpretera, `>>> import py_compile; py_compile.compile('myscript.py')`
Iz terminala, `python -m py_compile myscript.py`

Pythonovi editori

- Idle, dolazi s instalacijom Python-a,
- PyScripter,
- Spyder,
- PyCharm (Educators, Learners),
- Thonny,
- IPython (unutar internetskog preglednika),
- PythonTutor (web vizualizacija),
- Jupyter,
- Eclipse (kao plug-in),
- Pythonista (iPad, iPhone),

Portabilne verzije: portable Python, eGenix PyRun, winpython, itd.

Instalacija i pokretanje

Skine se sa stranice `https://www.python.org` i instalira,
u Terminalu se može također raditi, upiše se **python** (ili **python3**) i pritisne Enter,
prikazat će se >>>,
upiše se npr. **2+2** i Enter,
prikazat će se >>>**4**,
za pomoć se upiše npr. **help('quit')** i Enter,
prikazat će se >>> i puno teksta vezanog uz tu riječ, a za izlaz iz tog teksta pritisne se **q**,
za izlaz upiše se **exit()**, **quit()** ili se pritisne Ctrl+d (linux) tj. Ctrl+z (windows).

Istodobni rad s verzijama 2.x i 3.x

Izoliranje programerskih projekata da se izbjegne konflikt između različitih verzija Pythona.

Rješenje: **Virtualne okoline (enviroments)**

Program **Anaconda** (`https://www.anaconda.com/`) za stvaranje i upravljanje virtualnim okolinama.

Tipovi podataka

Svi tipovi podataka su objekti.
Svojstva objekta su:

1. identitet (adresa),
2. tip,
3. vrijednost,
4. metoda.

Tipovi objekta su:

- Skalarni: cijeli broj (int), realni broj (float), logička vrijednost (boolean), ništa (None).
- Složeni: string, kompleksni broj, lista, n-terac, skup, rječnik.

Tipovi podataka-primjeri objekata

Skalarni:

```
>>> type(37)
<class 'int'>
>>> type(56.1)
<class 'float'>
>>> type(True)
<class 'bool'>
>>> type(None)
<class 'NoneType'>
```

Složeni:

```
>>> type('sunce')
<class 'str'>
>>> type([1,"dva",7.7, 3+2j])
<class 'list'>
>>> type({"prvo":7,"danas":"vjetar"})
<class 'dict'>
>>> type((11,"dva dana",5+5, 7+1j))
<class 'tuple'>
```

Promjenjivi i nepromjenjvi objekti - spremnici adresa

- Neki objekti (koje nazivamo spremnici) sadrže adrese (identitete) drugih objekte, a ne vrijednosti.
- To su spremnici poput **liste**, **rječnika** i **n-terca**.
- Pod pojmom vrijednost spremnika misli se na skup svih vrijednosti objekata u njemu.
- Pod pojmovima promjenjivi i nepromjenjivi spremnik misli se na adrese (identitete) objekata, a ne na njihove vrijednosti.
- Promjenjivi spremnik (lista, rječnik) jednom stvoren ima promjenjive adrese i njihovu količinu.
- Nepromjenjivi spremnik (n-terac) jednom stvoren sadrži nepromjenjive adrese i njihovu količinu.
- Tako će nepromjenjivi spremnik koji sadrži adrese na promjenjive objekte promijeniti svoju vrijednost ako se vrijednost nepromjenjivog objekta izmjeni.

Operatori

Relacijski (vraća True ili False)	Aritmetički (unarni i binarni)
$>$	$+$
$<$	$-$
$>=$	$/$
$<=$	$*$
$==$	$\%$
$!=$	$**$
$\vdots$	$\vdots$

Literali

Literali su nazivi za neke ugrađene tipove objekata s konstantnim vrijednostima.

- Skalari, npr. `2.71` ,63, `False`, `None`,
- stringovi npr. `'Literal-neimenovana varijabla'`
- kompleksni broj npr. `14.9-9j`.

Identiteti objekata:

- decimalni, npr. 3068864816: `>>> id("a > 12")`,
- heksadecimalni, npr. '0xb6eb29f8': `>>> hex(id("a > 12"))`.

Funkcija `id()` vraća jedinstveni broj objekta (u CPythonu vraća memorijsku adresu objekta).

Varijable - adresa na objekt

Povezivanje (=):

```
>>> x = "abcd"
>>> id("abcd")
3063371968
>>> id(x)
3063371968
```

Repovezivanje (`x=x+"e"`)

```
>>> x=x+"e"
>>> x
'abcde'
>>> id(x)
3063976000
```

Sakupljanje nepovezanih objekata - *garbage collector*, nakon brisanja varijabli (`del`)

```
>>> del x
>>> x
NameError: name 'x' is not defined
```

Identifikatori - imena

- Imena kojima se referencira na objekte.
- Prvo slovo mora biti slovo abecede ili *underscore* (_), ostatak mogu biti slova i brojke i *underscore* (_).
- Imena varijabli npr. **`bilo_sta1`**, funkcija npr. `def` **`funkc`**`():`, klasa npr. `class` **`Klasa`**`():`, modula npr. `import` **`sys`**.
- Osjetljivi na velika i mala slova *case-sensitiv*, **mojaVar** i **mojavar** su različiti identifikatori.
- Ne smiju biti isti kao ključne riječi.
  Ključne riječi (ver. 3.6.):
  'False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield'.
- Primjeri pravilnih identifikatora: **i**, **i2**, **_k2**, **kuca_1_2**, ...
  i nepravilnih: **moja−var**, **>abc**, **2ime**, **ovo je var**, **pass**, ...

Izrazi

Cilj = Izraz

Izraz - kombinacija varijabli, literala i operatora, npr. `x+y**3-2`,
Cilj - varijable, indeksirani član niza (ili kriška) npr. `x,y,y[2],x[2:4]`

Pridružbe:

- obične, npr. `x = 1.7`, `z = z + 13`,
- proširene, npr. `z += 13`, `z *= 13.3`,
- raspakiravajuća, npr. `a,b = 4,5`,
- istodobna, npr. `a=b=c=0`.

Primjer programiranja - glavnica

Posudiš 1000 kn, vraćaš 1 godinu, s kamatom 10%.
Koliko ti je još ostalo za vratiti nakon 5 mjeseci.

```
posudio = 1000  # kn
kamata = 10  # 10 %
ukupno_za_vratiti = posudio*(1+kamata/100)
godine = 1
trenutni_mjesec = 5
glavnica_mjesec = posudio/(12*godine)  # mjesecni iznos glavnice
kamate_mjesec = glavnica_mjesec*(kamata/100)  # mjesecni iznos kamata
rata_mjesec = glavnica_mjesec+kamate_mjesec  #mjesecni iznos rate
koliko_jos_za_vratiti = ukupno_za_vratiti-rata_mjesec*trenutni_mjesec
print(koliko_jos_za_vratiti)

>>> 641.6666666666667
```