Úvod do jazyka Python (1/2)

Jan Kybic

http://cmp.felk.cvut.cz/~kybic
kybic@fel.cvut.cz

2016–2021

Python

- Programovací jazyk *Python* `http://www.python.org`
  - autor Guido van Rossum, 1989

Obrázek: Guido van Rossum

Proč Python?

- jazyk vysoké úrovně, pro všeobecné použití
- dobře čitelný
- mnoho knihoven
- velmi populární (v mnoha anketách 1. místo)
- *dynamický*
- *interpretovaný (byte-code)*
- *multiparadigmatický*
- *s automatickou alokací paměti*

Budeme používat Python 3.

Toto není kurz jazyka Python. Detaily najdete v dokumentaci.

Jak Python spustíme?

Napíšeme do příkazové řádky `python3`:

```
!python3
```

```
Python 3.4.3 (default, Oct 14 2015, 20:28:29)
[GCC 4.8.4] on linux
Type "help", "copyright", "credits" or "license" for more i
>>>
```

(nebo spustíme IDE prostředí jako `idle`, nebo Jupyter notebook...
Možností je mnoho)

Python jako kalkulačka

```
>python3
Python 3.4.3 (default, Oct 14 2015, 20:28:29)
[GCC 4.8.4] on linux
Type "help", "copyright", "credits" or "license" for more i
>>> 3+8
11
>>> 11*(5+3)
88
>>> 128./16.
8.0
>>> 2**16
65536
```

Python jako kalkulačka (2)

totéž, hezky vysázeno

```
3+8
```

```
11
```

```
11*(5+3)
```

```
88
```

```
128./16.
```

```
8.0
```

```
2**16
```

```
65536
```

Výrazy

- **Výrazy** *(expressions)* obsahují
  - Celá čísla: `3`, `8`, ...
  - Reálná čísla: `128.`, `11.5`,...
  - Operátory: `+`, `-`, `/`, `*`, ...
  - Oddělovače: `(`, `)`

Co se děje v zákulisí *(REPL)*

- Spustili jsme program `python3`, *interpret* Pythonu
  - tisk výzvy *(prompt)* `>>>`
  - přečtení uživatelského vstupu *(read)*
  - vyhodnocení výrazu *(evaluate)*
  - tisk výsledku *(print)*
- Opakované vykonávání (smyčka, *loop*)
- **REPL** *(read-eval-print-loop)*

Program jako transformace (filtr)

Obrázek: Transformace vstupu na výstup

Toky dat *(data flow)*

Proměnné a přiřazení

Hodnotu výrazu lze uložit pro pozdější použití

identifikátor = *výraz*

Příklad:

```
a=3
b=3+a
```

Jaká je hodnota proměnné *b*?

Proměnné a přiřazení

Hodnotu výrazu lze uložit pro pozdější použití

identifikátor = *výraz*

Příklad:

```
a=3
b=3+a
```

Jaká je hodnota proměnné *b*?

```
b
```

```
6
```

Příklad: Proměnné

```
boys=15
girls=17
total=boys+girls
difference=girls-boys
ratio=boys/total
total
```

```
32
```

```
difference
```

```
2
```

```
ratio
```

```
0.46875
```

Učte se anglicky. Počítače mluví anglicky, programy jsou anglicky, informace jsou anglicky.
Z důvodů přenositelnosti je bezpečnější se omezit na znaky anglické abecedy.

Hodnoty proměnných lze měnit

```
a=10
a=a-2
a=a*2
```

Jaká je hodnota *a*?

Hodnoty proměnných lze měnit

```
a=10
a=a-2
a=a*2
```

Jaká je hodnota *a*?

```
a
```

16

Není-li pro to dobrý důvod, hodnoty proměnných neměňte.

Proč používat proměnné

- **DRY** = *Do not repeat yourself.*
- Šetřme si práci, neopakujme se
- Zlepšení
  - **Srozumitelnosti** - smysluplná jména proměnných
  - **Údržby** - jedna změna jen na jednom místě
  - **Efektivity** - využijeme předchozích výpočtů

První program - *Hello world*

```
# Vypíše pozdravení
print("Hello world")
```

- vytvoříme v textovém editoru
- uložíme do souboru `hello_world.py`
- spustíme (z příkazové řádky, opakovaně)

První program - *Hello world*

```
# Vypíše pozdravení
print("Hello world")
```

- vytvoříme v textovém editoru
- uložíme do souboru `hello_world.py`
- spustíme (z příkazové řádky, opakovaně)

```
>python3 hello_world.py

Hello world
```

První program - *Hello world*

```
# Vypíše pozdravení
print("Hello world")
```

- vytvoříme v textovém editoru
- uložíme do souboru `hello_world.py`
- spustíme (z příkazové řádky, opakovaně)

```
>python3 hello_world.py
```

```
Hello world
```

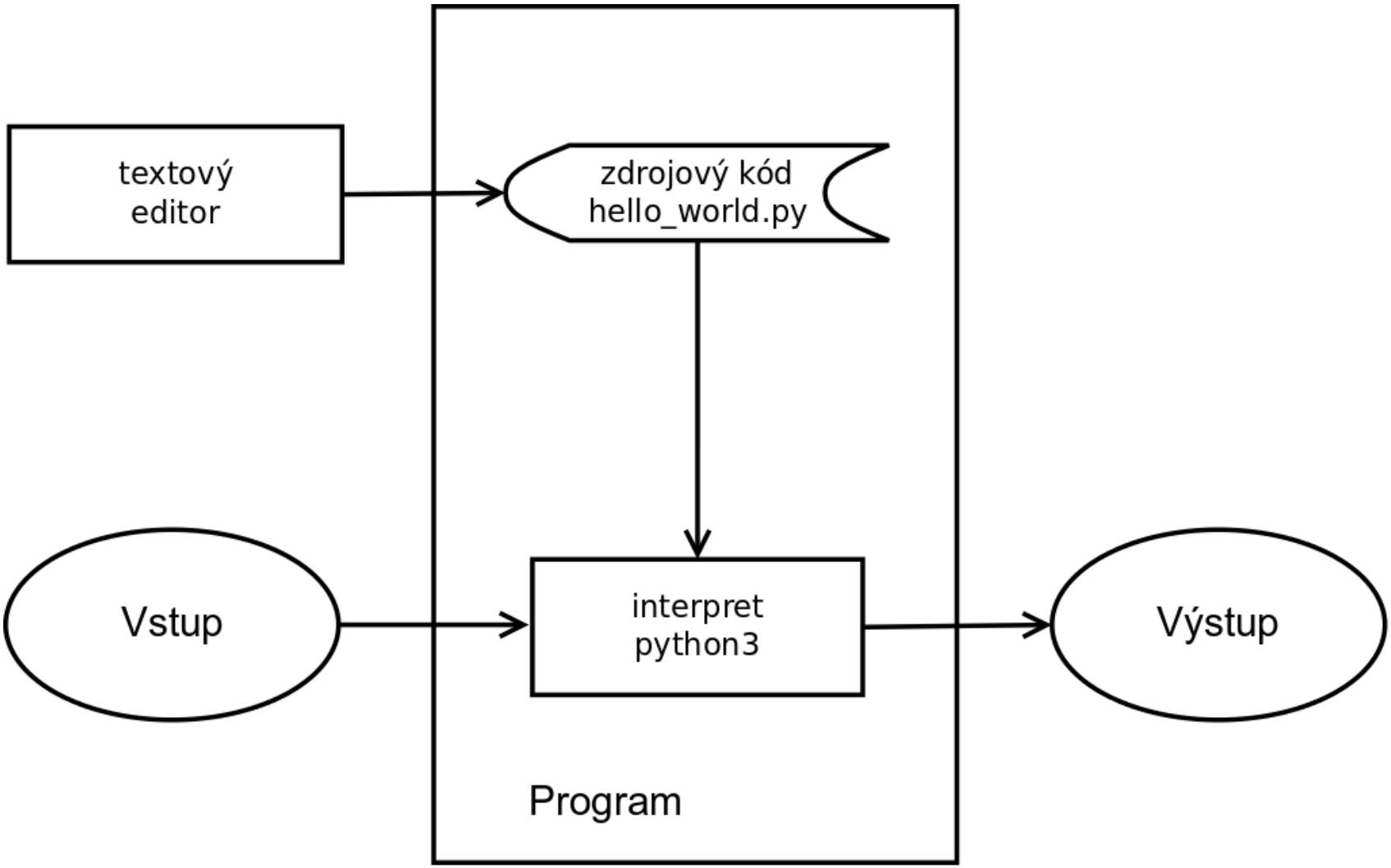
První řádka je komentář - každý program má být komentován.

Editory, integrovaná prostředí (IDE)

- **Editory:** Emacs, gEdit, joe, ...
- spouštění
  - z příkazové řádky
  - z integrovaného prostředí
  - z Jupyter notebooku
- **Integrované prostředí:** IDLE, PyCharm, Eclipse, ...

Na konkrétním editoru, případně IDE, zase tolik nezáleží.

Editace a interpretace programu



Obrázek: Editace a interpretace programu

Příklad: Převod stupňů Fahrenheita na stupně Celsia

Kolik $^\circ C$ je $75^\circ F$?

```
f=75
c=(f-32)*5./9.
print(c)
```

```
23.88888888888889
```

Příklad: Převod stupňů Fahrenheita na stupně Celsia

Kolik $^\circ C$ je $75^\circ F$?

```
f=75
c=(f-32)*5./9.
print(c)
```

```
23.88888888888889
```

Delší výpis:

```
print(f,"stupňů Fahrenheita je",c,"stupňů Celsia.")
```

```
75 stupňů Fahrenheita je 23.88888888888889 stupňů Celsia.
```

- *funkce* `print` vytiskne své argumenty
- argumentem funkce `print` může být číslo nebo *řetězec*

Interpolace řetězců *(pro pokročilé)*

```
print("%f stupňů Fahrenheita je %f stupňů Celsia." % (f,c))
```

```
75.000000 stupňů Fahrenheita je 23.888889 stupňů Celsia.
```

- `%f` do řetězce doplní reálná čísla z dalších argumentů
- Omezíme počet desetinných míst *(pro pokročilé)*:

```
print("%0.1f stupňů Fahrenheita je %0.1f stupňů Celsia." %
      (f,c))
```

```
75.0 stupňů Fahrenheita je 23.9 stupňů Celsia.
```

Všimněte si: uzávorkovaný výraz můžeme rozdělit na víc řádků

Program = automatizace

Co když chceme převést hodnot více? Kolik $^\circ C$ je $30^\circ F$?

- Šetřme si práci, neopakujme se
- **DRY** = *Do not repeat yourself*
- Vytvoříme program, který budeme moci opakovaně spouštět

Program = soubor

Do souboru `convert1.py` uložíme jednotlivé příkazy:

```
# Program pro převod stupňů Fahrenheita na stupně Celsia
f=75
c=(f-32)*5./9.
print("%0.1f stupňů Fahrenheita je %0.1f stupňů Celsia." %
      (f,c))
```

Program = soubor

Do souboru `convert1.py` uložíme jednotlivé příkazy:

```
# Program pro převod stupňů Fahrenheita na stupně Celsia
f=75
c=(f-32)*5./9.
print("%0.1f stupňů Fahrenheita je %0.1f stupňů Celsia." %
      (f,c))
```

a spustíme z příkazové řádky (i opakovaně)

```
>python3 convert1.py
```

```
75.0 stupňů Fahrenheita je 23.9 stupňů Celsia.
```

Čtení parametru z příkazové řádky

Náš program počítá pořád to samé... není flexibilní.

Vylepšená verze (`convert2.py`):

```
# Program convert2.py pro převod stupňů Fahrenheita
# na stupně Celsia
import sys

f=int(sys.argv[1]) # první argument
c=(f-32)*5./9.
print("%0.1f stupňů Fahrenheita je %0.1f stupňů Celsia." %
 (f,c))
```

Čtení parametru z příkazové řádky (2)

Argument (stupně Fahrenheita) zadáme pří spouštění:

```
>python3 convert2.py 60

60.0 stupňů Fahrenheita je 15.6 stupňů Celsia.

>python3 convert2.py 90

90.0 stupňů Fahrenheita je 32.2 stupňů Celsia.

>python3 convert2.py -20

-20.0 stupňů Fahrenheita je -28.9 stupňů Celsia.
```

Gratuluji, tohle je náš první užitečný program!

Čtení parametru z příkazové řádky (2)

Argument (stupně Fahrenheita) zadáme pří spouštění:

```
>python3 convert2.py 60

60.0 stupňů Fahrenheita je 15.6 stupňů Celsia.

>python3 convert2.py 90

90.0 stupňů Fahrenheita je 32.2 stupňů Celsia.

>python3 convert2.py -20

-20.0 stupňů Fahrenheita je -28.9 stupňů Celsia.
```

Gratuluji, tohle je náš první užitečný program!

Poznámky: Příkaz `import sys` zpřístupní knihovnu `sys` - vysvětlíme později. `sys.argv[1]` je první argument, `sys.argv[2]` druhý, atd.

Základní části textu programů

- **Komentáře**:

```
# Program convert2.py pro převod stupňů Fahrenheita
# na stupně Celsia
```

- **Klíčová slova**: `import`

```
import keyword
print(keyword.kwlist)
```

```
['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']
```

Základní části textu programů (2)

- **Identifikátory:** (jména proměnných a funkcí) `f`, `c`, `print`, ...
  Písmena, čísla, podtržítka, nezačíná číslem,není klíčové slovo
- **Operátory:** +,-,*,/,=,...
- **Literály:**
  - Celá čísla: 32,-20,...
  - Reálná čísla: `5.`, `9.`, `32.3`, `1.3e-6` ($1.3 \cdot 10^{-6}$)...
  - Řetězce: `"Hello world"`, `'xyz'`,`"%0.1f stupňů Fahrenheita je %0.1f stupňů Celsia."`...