Programming in Python
NPRG065

http://d3s.mff.cuni.cz

Tomas Bures
Petr Hnetynka
{bures,hnetynka}@d3s.mff.cuni.cz
Course information

- [https://d3s.mff.cuni.cz/teaching/nprg065/](https://d3s.mff.cuni.cz/teaching/nprg065/)

- 2/2 Exam + “Zápočet“
- Exam
  - practical in lab
    - implement a simple assignment
- “Zápočet“
  - homework
    - via ReCodEx
    - [https://recodex.ms.mff.cuni.cz](https://recodex.ms.mff.cuni.cz)
Courses to consider

- Prg. languages
  - Python for practice (NPRG067) – winter semester
    - Continuation of this course
    - GUI apps, machine learning, big data,…
  - Concepts of Modern Programming Languages (NPRG014) – winter semester
  - Java (NPR013) – winter semester
  - Advanced Java (NPR021) – Mo 12:20, Mo 14:00
  - Practical Dynamic Compilation (NSWI176) – Tue 9:00
Courses to consider

- Development

  - Middleware (NSWI080) – Fri 10:40
  - Performance Evaluation of Computer Systems (NSWI131) – Wed 14:00
  - Advanced Tools for Software Development and Monitoring (NSWI126) – Thu 9:00
  - Software Development Tools (NSWI154) – winter semester
  - Software Engineering for Dependable Systems (NSWI054) – Mon 12:20
  - Code Optimization in Production Compilers (NSWI134)
Courses to consider

- Formal analysis
  - Program Analysis & Code Verification (NSWI132) – Thu 14:00
  - Selected Chapters on Combinatorics (NDMI056)
  - Complex network analysis (NDMI096)
    - Suitable time will be agreed upon at a meeting on Tue 18.2. at 9:00, in S9.
Approx. time-line of the course

- Introduction
- Core types
- Control structures
- Data structures
- Classes and objects
- Core parts of the std. library
About Python

- Dynamically-typed
  - *duck typing*
- Object-oriented language
  - there are classes but it is not a strictly class-based language
- Interpreted
  - no explicit compilation
  - “JIT” compilation to Python bytecode
- Started around 1990 by Guido Van Rossum
- Now in version 3.8
  - 2.7 – the last version of Python 2 still used
    - but unsupported since January 1, 2020
- One of the most popular languages today
  - mainly for data analysis and machine learning

"If it walks like a duck and it quacks like a duck, then it must be a duck."
### Popularity

**Worldwide, Feb 2020 compared to a year ago:**

<table>
<thead>
<tr>
<th>Rank</th>
<th>Change</th>
<th>Language</th>
<th>Share</th>
<th>Trend</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>Python</td>
<td>29.88%</td>
<td>+4.1%</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>Java</td>
<td>19.05%</td>
<td>-1.8%</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>Javascript</td>
<td>8.17%</td>
<td>+0.1%</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>C#</td>
<td>7.3%</td>
<td>-0.1%</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>PHP</td>
<td>4.97%</td>
<td>-1.0%</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>C/C++</td>
<td>3.94%</td>
<td>-0.2%</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>R</td>
<td>3.74%</td>
<td>-0.2%</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>Objective C</td>
<td>2.42%</td>
<td>-0.8%</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>Swift</td>
<td>2.29%</td>
<td>-0.2%</td>
</tr>
<tr>
<td>10</td>
<td>↑</td>
<td>TypeScript</td>
<td>1.84%</td>
<td>+0.3%</td>
</tr>
</tbody>
</table>

---

**Feb 2020 vs Feb 2019 Change:**

<table>
<thead>
<tr>
<th>Rank</th>
<th>Change</th>
<th>Programming Language</th>
<th>Ratings</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>Java</td>
<td>17.358%</td>
<td>+1.48%</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>C</td>
<td>16.766%</td>
<td>+4.34%</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>Python</td>
<td>9.345%</td>
<td>+1.77%</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>C++</td>
<td>6.164%</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>C#</td>
<td>5.927%</td>
<td>+3.08%</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>Visual Basic.NET</td>
<td>5.682%</td>
<td>-1.23%</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>JavaScript</td>
<td>2.060%</td>
<td>-0.79%</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>PHP</td>
<td>2.018%</td>
<td>-0.25%</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>SQL</td>
<td>1.526%</td>
<td>-0.37%</td>
</tr>
<tr>
<td>10</td>
<td>20</td>
<td>Swift</td>
<td>1.460%</td>
<td>+0.54%</td>
</tr>
</tbody>
</table>

---


*Popularity Index: [Link](http://pypl.github.io/)*

*TIODE Index: [Link](https://www.tiobe.com/tiobe-index/)*
About Python

• Name – why Python
  ▪ Monty Python’s Flying Circus ;)  
• Portable
  ▪ Windows, Linux, *BSD,..., anywhere
• Installation [https://www.python.org/downloads/](https://www.python.org/downloads/)
  ▪ on Windows – download installer
  ▪ on Linux – use a package manager
• License
  ▪ Python Software Foundation license
  ▪ BSD style license, can be used for anything
• PyPI – [https://pypi.python.org/](https://pypi.python.org/)
  ▪ Python Package Index
  ▪ the repository of python packages
IDE

- PyCharm
  - [https://www.jetbrains.com/pycharm/](https://www.jetbrains.com/pycharm/)
  - Community edition – free
  - Professional edition – free for students/teachers
    - register via your university email

- Other IDEs
Sources

• Scripts
  - my_script.py
  - no explicit main – just start code
  - executable programs
    • `python my_script.py`
      or
    • `my_script.py`
      - on unix systems
      - shebang line: `#!/usr/bin/env python3`
Shell

- Interactive shell
  - immediate evaluation
  - history (like in bash)
  - ...
  - run just `python`

```python
>>> 1 + 2
3
```
Multiple Python implementations

- **CPython**
  - “the” Python
- **MicroPython**
  - a variant of CPython
  - runs on microcontrollers (pyboard, ESP32,...)
- **PyPy**
  - implementation in Python
  - JIT
- **Jython**
  - in Java, Python2 only
  - can be embedded in Java
- **IronPython**
  - in .NET
- ...

Python introduction...

• ...via examples
print('Hello, world.')
Case sensitivity

Two variables

```python
a = 1
A = 2
print(a)
print(A)
```
Fibonacci numbers

```python
def fib(a):
    if a <= 1:
        return 1
    else:
        return fib(a - 1) + fib(a - 2)

print(fib(10))
```

- No return type
- No difference between functions/procedures
- No begin/end, no `{ }`
- Blocks by indentation
def multi(number):
    print('Multiplication table of ', number)
    for i in range(11):
        print(i * number)
def Fib(k):
    prev = 1
    prevprev = 1
    while k > 0:
        tmp = prev + prevprev
        prevprev = prev
        prev = tmp
        k -= 1
    return prev
Command line arguments

import sys

print('Num. of args', len(sys.argv))
for arg in sys.argv:
    print(arg)

We will use elements from the sys module

A list with command line arguments
Max value in “array”

arr = [0, 9, 1, 8, 2, 7, 3, 6, 4, 5]
max = 0
i = 0
while i < len(arr):
    if arr[i] > max:
        max = arr[i]
    i += 1
print(max)
The slides are licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.