Programming in Python NPRG065

http://d3s.mff.cuni.cz

Department of Distributed and Dependable Systems

Tomas Bures

Petr Hnetynka

{bures,hnetynka}@d3s.mff.cuni.cz

CHARLES UNIVERSITY IN PRAGUE

faculty of mathematics and physics

Course information

- 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

Department of Distributed and Dependable Systems

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*

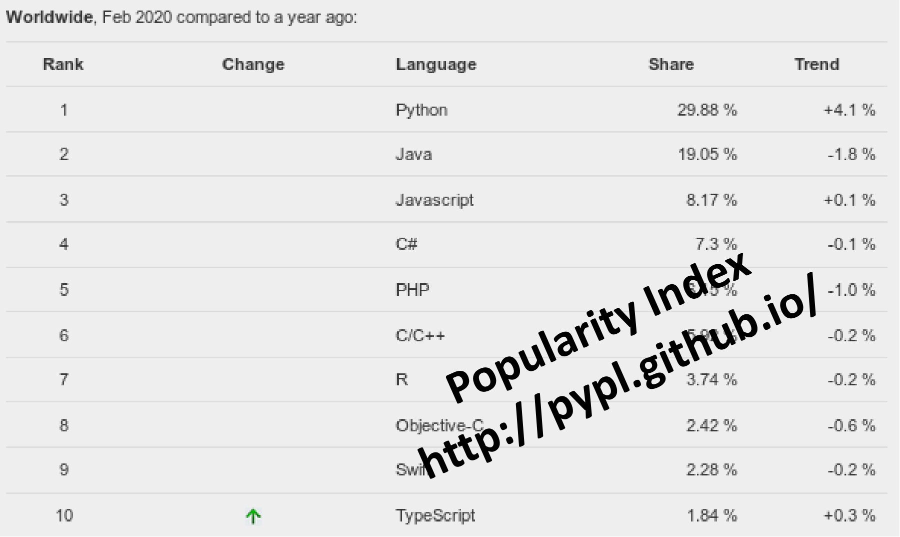
"If it walks like a duck and it quacks like a duck, then it must be a duck."

- 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

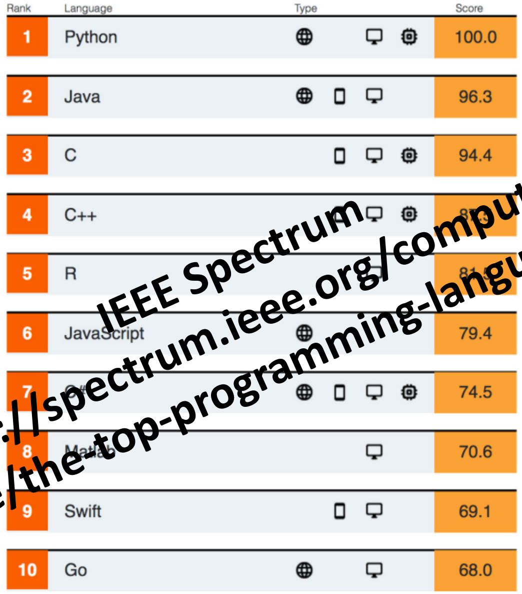
Popularity



Worldwide, Feb 2020 compared to a year ago:

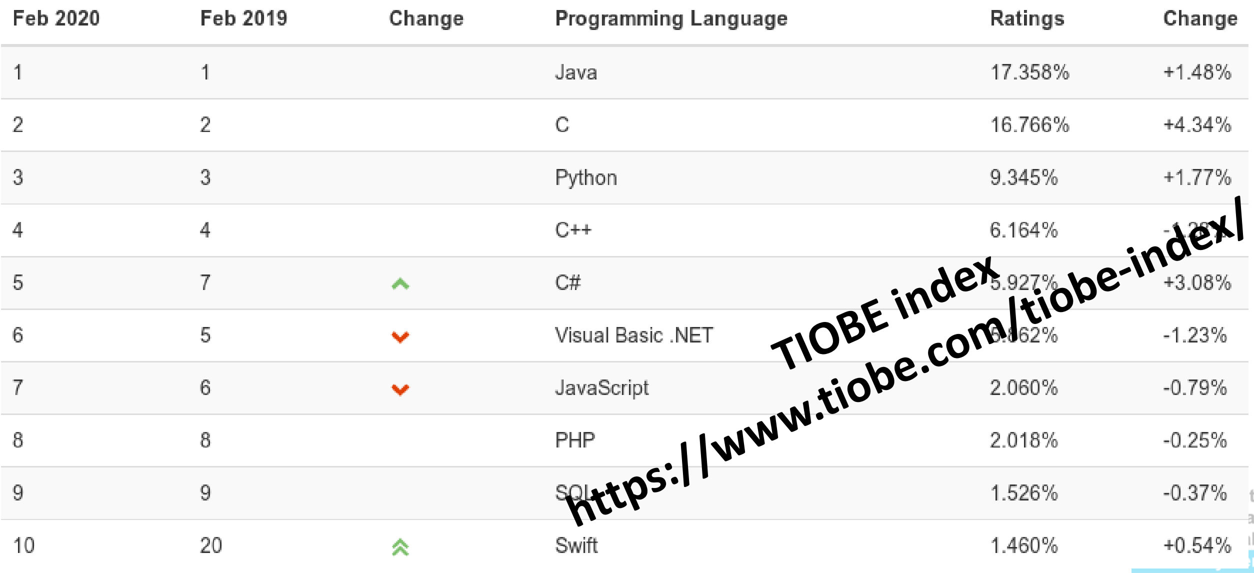
Rank	Change	Language	Share	Trend
1		Python	29.88 %	+4.1 %
2		Java	19.05 %	-1.8 %
3		Javascript	8.17 %	+0.1 %
4		C#	7.3 %	-0.1 %
5		PHP	[illegible].5 %	-1.0 %
6		C/C++	5.9[illegible] %	-0.2 %
7		R	3.74 %	-0.2 %
8		Objective-C	2.42 %	-0.6 %
9		Swift	2.28 %	-0.2 %
10	↑	TypeScript	1.84 %	+0.3 %

Popularity Index
http://pypl.github.io/



Rank	Language	Type	Score
1	Python	🌐 🖥 ⚙	100.0
2	Java	🌐 📱 🖥	96.3
3	C	📱 🖥 ⚙	94.4
4	C++	🖥 ⚙	8[illegible]
5	R	🖥	81.5
6	JavaScript	🌐	79.4
7	C#	🌐 📱 🖥 ⚙	74.5
8	Matlab	🖥	70.6
9	Swift	📱 🖥	69.1
10	Go	🌐 🖥	68.0

IEEE Spectrum
https://spectrum.ieee.org/computing/software/the-top-programming-languages-2019



Feb 2020	Feb 2019	Change	Programming Language	Ratings	Change
1	1		Java	17.358%	+1.48%
2	2		C	16.766%	+4.34%
3	3		Python	9.345%	+1.77%
4	4		C++	6.164%	-1.[illegible]%
5	7	⌃	C#	5.927%	+3.08%
6	5	⌄	Visual Basic .NET	5.862%	-1.23%
7	6	⌄	JavaScript	2.060%	-0.79%
8	8		PHP	2.018%	-0.25%
9	9		SQL	1.526%	-0.37%
10	20	⏫	Swift	1.460%	+0.54%

TIOBE index
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/
  - 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/
  - Python Package Index
  - the repository of python packages

IDE

- 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`**

```
>>> 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

Hello world

No semicolons

```
print('Hello, world.')
```

No begin, no main method,…

Case sensitivity

Two variables

```
a = 1
A = 2
print(a)
print(A)
```

Fibonacci numbers

```
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

Multiplication table

No variable declaration

```
def multi(number):
   print('Multiplication table of ', number)
   for i in range(11):
      print(i * number)
```

No “classical” **for** cycle

Fibonacci numbers v. 2

```
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)
```