

# Programování 1: Standardní knihovna

Martin Mareš

`mj@ucw.cz`

Katedra Aplikované Matematiky  
MFF UK Praha

2021

# Moduly a importování

Programy mohou být členěné na **moduly**.

Provedeme-li:

```
>>> import math
```

načte se soubor `math.py` (hledá se v aktuálním adresáři a pak v knihovnách). Potom píšeme:

```
>>> math.sin(1)
```

Formálně: **math** je objekt, jako jeho atributy vidíme funkce a proměnné definované uvnitř modulu.

Alternativně:

```
>>> from math import sin
```

```
>>> from sys import *
```

navíc zkopíruje do aktuálního modulu definice z importovaného modulu (takže pak **sin** znamená totéž co **math.sin** apod.).

- **math** – matematické funkce a konstanty
- **operator** – funkční podoba operátorů (třeba **add** pro +)
- **collections** – typ **defaultdict**
  - také tu je **deque** (double-ended queue) – seznam s rychlým přidáváním a odebíráním na obou koncích

Typ **bytes** obsahuje neměnnou posloupnost bajtů (8-bitových hodnot).

Literály: `b'Brum'` `b'\x62\x72\x75\x6d'`

```
>>> "číslo".encode('utf-8')
```

```
b'\xc4\x8d\xc3\xadslo'
```

```
>>> b'\xc4\x8d\xc3\xadslo'.decode('utf-8')
```

```
'číslo'
```

```
>>> bytes([1, 2, 3])
```

```
b'\x01\x02\x03'
```

Při práci s binárními soubory čteme/zapisujeme bytes.

## Typ `bytearray`:

- jako `bytes`, ale lze modifikovat
- seznam bajtů (prostorově efektivnější než běžný seznam)
- `bytearray(n)` – seznam  $n$  nul
- `bytearray([1, 2, 3])`
- `bytearray('řetězec', 'utf-8')`

# Homogenní pole

```
>>> import array
>>> a = array.array('i', [1, 2])
>>> a.itemsize
4
```

'i' je kód typu položek, například:

- **i** – integer: aspoň 32-bitové číslo se znaménkem
- **l** – totéž bez znaménka
- **b** – bajt se znaménkem
- **q** – aspoň 64-bitové číslo se znaménkem
- **f** – float: aspoň 32-bitové desetinné číslo
- **d** – double: aspoň 64-bitové desetinné číslo

```
>>> from fractions import Fraction
>>> Fraction(1, 2) + Fraction(1, 3)
Fraction(5, 6)

>>> Fraction(1/3)
Fraction(6004799503160661, 18014398509481984)

>>> Fraction(1/3).limit_denominator(100000)
Fraction(1, 3)

>>> Fraction("1/3")
Fraction(1, 3)

>>> print(Fraction(1, 3))
'1/3'
```

# Pseudonáhodný generátor

```
>>> import random
>>> random.random()
0.28947857702914326    (z intervalu [0, 1])

>>> random.uniform(0, 1000)
50.64122748168531    (z intervalu [a, b])

>>> random.randrange(0, 1000)
524    (celé číslo od a do b - 1)

>>> random.randrange(1000)
451    (stejně jako u range jde dolní mez vynechat)

>>> random.seed(12345)
>>> random.random()
0.41661987254534116    (vyjde vždy stejně)
```



# Pseudonáhodný výběr

```
>>> random.choice(['pátek', 'sobota', 'neděle'])  
'neděle'    (náhodný prvek seznamu)
```

```
>>> random.choices("0123456789", k=10)  
['3', '4', '9', '8', '1', '3', '2', '1', '8', '6']  
(výběr s vrácením zpět)
```

```
>>> random.sample("0123456789", k=10)  
['3', '0', '6', '1', '2', '7', '4', '9', '5', '8']  
(výběr bez vrácení)
```

```
>>> random.sample(range(1000000), k=5)  
[301599, 128323, 695182, 627325, 967424]  
(vzorkuje přímo rozsah, nepřevádí na seznam)
```

## Další zajímavé moduly

- **datetime, calendar** – práce s datem a časem
- **copy** – rekurzivní kopírování
- **cmath** – počítání s komplexními čísly
- **statistics** – základní statistické funkce
- **pickle** – (de)serializace datových struktur
- **re** – analýza textu pomocí regulárních výrazů