Programování 1: Úvod do Pythonu

Petr Čermák
cermak@mag.mff.cuni.cz

Katedra Fyziky Kondenzovaných Látek
MFF UK Praha

2020
forked from https://gitlab.kam.mff.cuni.cz/mj/prm1

Úvodní informace

- Web cvičení: cermak.science/teaching/p1m/
- Podmínky zápočtu: 70/120 bodů v ReCodExu

Úvodní informace

- Web cvičení: cermak.science/teaching/p1m/
- Podmínky zápočtu: 70/120 bodů v ReCodExu
- Konzultace:
  - slack (nejrychlejší)
  - mail (klasika)
  - zoom (možno si domluvit konzultaci),
  - osobní setkání (max 2 naráz, kancelář mám v Troji)

Programování = psaní algoritmů

- *Co je Algoritmus?* **Návod na dosažení výsledku.**
- Příklad: Euklidův algoritmus (=nalezení **N**ějvětšího **S**polečného **D**ělitele)
  - Vychází z tvrzení $NSD(u, v) = NSD(v, u–qv)$

Programování = psaní algoritmů

- *Co je Algoritmus?* **Návod na dosažení výsledku.**
- Příklad: Euklidův algoritmus (=nalezení **N**ějvětšího **S**polečného **D**ělitele)
  - Vychází z tvrzení $NSD(u, v) = NSD(v, u–qv)$
  - **Návod pro matfyzáka**: *Největší společný dělitel dvou přirozených čísel nalezneme tak, že dokud jsou obě různá, odečítáme menší od většího.*

Programování = psaní algoritmů

- *Co je Algoritmus?* **Návod na dosažení výsledku.**
- Příklad: Euklidův algoritmus (=nalezení **N**ějvětšího **S**polečného **D**ělitele)
  - Vychází z tvrzení $NSD(u, v) = NSD(v, u–qv)$
  - **Návod pro matfyzáka**: *Největší společný dělitel dvou přirozených čísel nalezneme tak, že dokud jsou obě různá, odečítáme menší od většího.*
  - **Návod pro opice**: *Vidíš 2 hromady kamenů. Vezmi z té větší tolik, kolik je v té menší a zahoď je. Opakuj dokud není v obou stejně.*

Programování = psaní algoritmů

- *Co je Algoritmus?* **Návod na dosažení výsledku.**
- Příklad: Euklidův algoritmus (=nalezení **N**ějvětšího **S**polečného **D**ělitele)
  - Vychází z tvrzení $NSD(u, v) = NSD(v, u–qv)$
  - **Návod pro matfyzáka**: *Největší společný dělitel dvou přirozených čísel nalezneme tak, že dokud jsou obě různá, odečítáme menší od většího.*
  - **Návod pro opice**: *Vidíš 2 hromady kamenů. Vezmi z té větší tolik, kolik je v té menší a zahoď je. Opakuj dokud není v obou stejně.*
  - **Návod pro procesor Z80**: *B8C83803 9018F94F 78914779 18F2*

Programování = psaní algoritmů

- *Co je Algoritmus?* **Návod na dosažení výsledku.**
- Příklad: Euklidův algoritmus (=nalezení **N**ějvětšího **S**polečného **D**ělitele)
  - Vychází z tvrzení $NSD(u, v) = NSD(v, u–qv)$
  - **Návod pro matfyzáka**: *Největší společný dělitel dvou přirozených čísel nalezneme tak, že dokud jsou obě různá, odečítáme menší od většího.*
  - **Návod pro opice**: *Vidíš 2 hromady kamenů. Vezmi z té větší tolik, kolik je v té menší a zahoď je. Opakuj dokud není v obou stejně.*
  - **Návod pro procesor Z80**: *B8C83803 9018F94F 78914779 18F2*

```
0000                    GCD:
0000   B8                       CP   b
0001   C8                       RET  z   ; while a != b
0002   38 03                    JR   c,else   ; if a > b
0004   90                       SUB  b   ; a = a - b
0005   18 F9                    JR   gcd
0007                    ELSE:
0007   4F                       LD   c,a   ; Save a
0008   78                       LD   a,b   ; Swap b into a so we can subtract
0009   91                       SUB  c   ; b = b - a
000A   47                       LD   b,a   ; Put a and b back where they belong
000B   79                       LD   a,c
000C   18 F2                    JR   gcd
```

Programovací jazyky

- Strojový kód je pro člověka nečitelný
- Závisí na procesoru
- Programovací jazyk je čitelný, *překladač* ho zkompiluje do zdrojového kódu
- Dynamické programovací jazyky - překládají se za běhu

Programovací jazyky

- Strojový kód je pro člověka nečitelný
- Závisí na procesoru
- Programovací jazyk je čitelný, *překladač* ho zkompiluje do zdrojového kódu
- Dynamické programovací jazyky - překládají se za běhu

Python

- Snadno čitelný kód
- Populární ve vědě, AI, statistice
- Mnoho dostupných knihoven

Euklidův algoritmus v Pythonu

```
>>> from math import gcd
>>> gcd(15,10)
5
```

Na konci dnešní hodiny to budete umět naprogramovat bez vestavěné funkce.

Python jako kalkulačka

```
>>> 1+1
2

>>> 2+3*4+1
15

>>> 2+3 * 4+1
15

>>> (2+3)*(4+1)
25

>>> 2**10
1024

>>> 2**100
1267650600228229401496703205376
```

Desetinná čísla

Dělení vytváří „desetinná“ čísla s omezenou přesností:

```
>>> 1/3
0.3333333333333333

>>> 1/3 * 3
1.0

>>> 1/6 + 1/6 + 1/6 + 1/6 + 1/6 + 1/6
0.9999999999999999
```

Notace s mantisou a exponentem: $m \cdot 10^e$.

```
>>> 1/(2**100)
7.888609052210118e-31
```

Celočíselné dělení

Celočíselné dělení:

```
>>> 7//3
2
```

Zbytek po dělení (modulo):

```
>>> 7%3
1
```

Dělení záporného čísla zaokrouhluje dolů, nikoliv k nule:

```
>>> -7//3
-3

>>> -(7//3)
-2
```

Vždy platí `(a//b)*b + (a%b) = a`

```
>>> -7%3
2
```

Proměnné

Hodnotu výrazu si můžeme pojmenovat:

```
>>> a=100
>>> b=23
>>> a+b
123
```

Obsah proměnné jde měnit:

```
>>> soucet=0
>>> soucet=soucet+10
>>> soucet=soucet+3
>>> soucet
13
```

Zkrácený zápis:

```
>>> soucet+=1
>>> soucet
14
```

Matematické funkce

Matematická knihovna:

```
>>> import math
>>> math.pi
3.141592653589793

>>> math.sin(math.pi / 3)
0.8660254037844386
```

Voláme o pomoc:

```
>>> help(math.sin)
>>> help(math)
```

Místo `math.sin` můžeme psát prostě `sin`:

```
>>> from math import *
>>> sin(0)
```

Klademe Pythonu otázky

```
>>> 5**7 > 7**5
True

>>> cos(0) < 0
False

>>> 0.8 <= sin(pi/3) <= 0.9
True

>>> pi>3 and pi<4
True

>>> x>0 or not x>0
True

>>> 1 == 1
True

>>> 1 != 2
True
```

Náš první program: počítáme od 1 do 10

```
1 i = 1
2 while i <= 10:
3     print(i)
4     i += 1
```

Odsazování je povinné, udává blokovou strukturu programu. Pak je jasné, které příkazy jsou uvnitř cyklu, a které už za ním.

Náš první program: vypisujeme jen sudá čísla

```
1 i = 1
2 while i <= 10:
3     if i%2 == 0:
4         print(i)
5     i += 1
```

Náš první program: zeptáme se, do kolika počítat

```
1  n = int(input("Do kolika chceš počítat? "))
2
3  i = 1
4
5  while i <= n:
6      if i%2 == 0:
7          print(i)
8      i += 1
```

V uvozovkách se zapisují řetězce (o nich později).
Funkce `input()` načte řetězec, `int()` ho převede na číslo.

Náš první program: ještě trocha komentářů

```
1  #!/usr/bin/env python3
2
3  # Nejprve zjistíme, do kolika počítat
4  n = int(input("Do kolika chceš počítat? "))
5
6  # Aktuální číslo
7  i = 1
8
9  while i <= n:              # Ještě pokračovat?
10     if i%2 == 0:           # Je číslo sudé?
11         print(i)
12     i += 1                 # Další, prosím!
```

Od # do konce řádku se vše ignoruje.
Prvni řádek pod Linuxem říká, čím se má soubor spustit.

Náš druhý program: Euklidův algoritmus

```
1 a = 10
2 b = 15
3 while a != b:
4     if b > a:
5         b = b - a
6     else:
7         a = a - b
8 print(a)
```

Trocha Python magic:

```
1 while b:
2     a, b = b, a % b
3 print(a)
```