Lekce 3: Úvod do Pythonu I – První program

Python pro GIS - Základy programování

Vojtěch Barták, FŽP ČZU Praha

2025-10-20

Table of contents

1 Cíle lekce **3**

2 Co je Python? **3**
2.1 Úrovně programovacích jazyků . . . 3
2.1.1 Nízkoúrovňové jazyky (blízko hardwaru) . . . 3
2.1.2 Vysokoúrovňové jazyky (blízko lidskému myšlení) . . . 4
2.1.3 Kde je Python? . . . 4
2.2 Interpretovaný vs. kompilovaný jazyk . . . 4
2.3 Proč Python pro GIS? . . . 5

3 První program: Hello World **5**
3.1 Způsob 1: Interaktivní režim . . . 6
3.2 Způsob 2: Python skript . . . 6

4 Proměnné a datové typy **7**
4.1 Co je proměnná? . . . 7
4.2 Základní datové typy . . . 8
4.2.1 **Celá čísla (`int`)** . . . 8
4.2.2 **Desetinná čísla (`float`)** . . . 8
4.2.3 **Text (`string` nebo `str`)** . . . 8
4.2.4 **Logické hodnoty (`bool`)** . . . 8
4.3 Pojmenování proměnných . . . 9

5 Práce s čísly **9**
5.1 Základní operace . . . 9
5.2 Příklady s proměnnými . . . 10

6 Práce s textem (stringy) 10
6.1 Spojování stringů . . . 10
6.2 f-stringy (doporučený způsob!) . . . 11
6.3 Převod mezi typy . . . 11

7 Komunikace s uživatelem 12
7.1 Funkce `input()` . . . 12
7.2 Praktický příklad . . . 12

8 Seznamy (Lists) 13
8.1 Co je seznam? . . . 13
8.2 Vytvoření seznamu . . . 13
8.3 Indexování . . . 13
8.4 Délka seznamu . . . 14
8.5 Přidávání prvků . . . 14
8.6 Praktický příklad . . . 14

9 Praktická cvičení 15
9.1 Cvičení 1: Kalkulačka vzdálenosti . . . 15
9.2 Cvičení 2: Průměrná teplota . . . 15
9.3 Cvičení 3: Seznam měst . . . 15

10 Shrnutí 16
10.1 Co jsme se naučili . . . 16
10.2 Co bude příště? . . . 16

11 Cheatsheet 17

12 Poznámky pro vyučujícího 18
12.1 Běžné chyby studentů . . . 18
12.2 Časový plán (90 min) . . . 18
12.3 Klíčové momenty . . . 18

1 Cíle lekce

Po absolvování této lekce budete umět:

- Vysvětlit, co je Python a proč se používá v GIS
- Spustit první Python program
- Pracovat se základními datovými typy (čísla, text)
- Používat proměnné pro ukládání dat
- Komunikovat s uživatelem pomocí `print()` a `input()`
- Pracovat se seznamy (vytvoření, indexování, přidávání prvků)

Časová dotace: 90 minut

---

2 Co je Python?

Jazyk Python vyvinul nizozemský vývojář **Guido van Rossum** koncem 80. let, první verze vyšla v roce **1991**. Název odkazuje na britskou komediální skupinu **Monty Python**.

2.1 Úrovně programovacích jazyků

Počítač rozumí pouze **binárním instrukcím** (0 a 1). Programovací jazyky se liší tím, jak blízko jsou lidskému myšlení nebo hardwaru.

2.1.1 Nízkoúrovňové jazyky (blízko hardwaru)

Strojový kód: - Binární instrukce (01001011 10110100…) - Přímo vykonává procesor - Téměř nečitelné pro člověka

Assembly: - Nejnižší "lidsky čitelný" jazyk - Každá instrukce = jedna operace procesoru - Zkratky: `MOV`, `ADD`, `JMP`

```
MOV AX, 5       ; Ulož 5 do paměti
ADD AX, 3       ; Přičti 3
```

Kdy se používá: Ovladače hardwaru, jádro operačního systému, kritické části vyžadující maximální rychlost.

2.1.2 Vysokoúrovňové jazyky (blízko lidskému myšlení)

Vlastnosti: - Čitelný kód (`if`, `for`, `print()`) - Abstrakce - nemusíte řešit detaily hardwaru - Přenositelnost - stejný kód na různých platformách - Rychlejší vývoj, méně chyb

Příklady: Python, Java, C++, R, JavaScript

Srovnání Assembly vs. Python:

```
; Assembly
MOV AX, 5
ADD AX, 3
```

```
# Python
vysledek = 5 + 3
```

2.1.3 Kde je Python?

Python patří mezi **vysokoúrovňové jazyky** - hodně abstraktní a čitelný. Snadno se učí, ale je pomalejší než C++.

Hierarchie:

```
Strojový kód (0101...)
       ↓
Assembly (MOV, ADD...)
       ↓
C, C++
       ↓
Python, Java
```

2.2 Interpretovaný vs. kompilovaný jazyk

Python = interpretovaný jazyk - kód se překládá a vykonává řádek po řádku:

```
script.py → Python interpret → Spuštění
```

Kompilované jazyky (C, C++) - celý program se nejprve přeloží:

```
program.c → Kompilátor → strojový kód (.exe) → Spuštění
```

Výhody interpretace: - Napíšete kód a okamžitě ho spustíte - Snadné ladění - Přenositelnost

Nevýhody: - Pomalejší než kompilované jazyky - Vyžaduje nainstalovaný Python interpret

> Pro GIS je rychlost dostatečná
>
> Pro většinu GIS úloh je rychlost Pythonu více než dostatečná. Co ušetříte na času psaní kódu mnohonásobně převýší ztrátu rychlosti běhu.

2.3 Proč Python pro GIS?

1. **Jednoduchý** - zvládne ho úplný začátečník bez zkušeností s programováním
2. **Výkonný** - objektově orientovaný jazyk pro krátké skripty i rozsáhlé programy
3. **Multiplatformní** - funguje stejně ve Windows, Linuxu i macOS
4. **Populární** - rozsáhlá dokumentace, výukové materiály, fóra, obrovské množství knihoven
5. **Open source a zdarma**
6. **Bohaté GIS knihovny** - vektorové a rastrové analýzy, tvorba map, publikace na webu
7. **Výlučný skriptovací jazyk v ArcGIS Pro** pro automatizaci GIS operací

---

3 První program: Hello World

Každý programátor začíná programem, který vypíše "Hello World". Existuje několik způsobů, jak Python kód spustit - od jednoduchého příkazového řádku až po pokročilá vývojová prostředí. Začneme těmi nejjednodušším způsoby.

> Kde psát Python kód?
>
> Python kód můžete psát v:
>
> - **Příkazovém řádku** (interaktivní režim) - pro rychlé testování
> - **Textovém editoru** (Notepad, VS Code) + spuštění z příkazové řádky
> - **IDE** (PyCharm, Spyder, VS Code) - komplexní vývojová prostředí
> - **Jupyter Notebook** - kombinuje kód, výsledky a text (ideální pro analýzy)
>
> Pro tento kurz budeme používat hlavně **textový editor + příkazová řádka** a později **Jupyter Notebook v ArcGIS Pro**.

3.1 Způsob 1: Interaktivní režim

1. Otevřete **Command Prompt** (Windows) nebo **Terminal** (Mac/Linux)
2. Napište `python` a stiskněte Enter
3. Uvidíte něco jako:

```
Python 3.9.11 (...)
>>>
```

4. Napište:

```
>>> print("Hello World")
```

5. Stiskněte Enter

Výsledek:

```
Hello World
```

> Interaktivní režim
>
> Tento režim je skvělý pro **rychlé testování** kódu. Každý příkaz se provede okamžitě po stisku Enter. Pro ukončení napište `exit()` nebo stiskněte Ctrl+Z (Windows) / Ctrl+D (Mac/Linux).

3.2 Způsob 2: Python skript

1. Otevřete textový editor (Notepad, VS Code, Notepad++)
2. Napište:

```
print("Hello World")
```

3. Uložte jako `hello.py` (důležité je přípona `.py`)
4. V příkazové řádce (v adresáři se souborem) spusťte:

```
python hello.py
```

Výsledek:

```
Hello World
```

Python skripty

Soubory s příponou `.py` jsou **Python skripty**. Výhoda oproti interaktivnímu režimu:

- Kód můžete uložit a znovu použít
- Můžete psát delší programy
- Snadno se sdílí s kolegy

Jupyter Notebook v ArcGIS Pro

Později v kurzu budeme používat **Jupyter Notebook**, který je integrovaný v ArcGIS Pro. Otevřete ho přes: **Insert → New Notebook**
Výhoda: Kombinuje kód, výsledky a text v jednom dokumentu. Ideální pro analýzy a experimenty s ArcPy.

Virtuální prostředí (pro pokročilé)

ArcGIS Pro má vlastní **conda prostředí** s nainstalovaným Pythonem a ArcPy. Zatím se o to nemusíte starat - vše je připravené. Pokud budete později pracovat s Pythonem mimo ArcGIS, vrátíme se k tématu virtuálních prostředí.

Funkce `print()`

`print()` je základní funkce pro **výpis textu** na obrazovku. Používá se pro:

- Zobrazení výsledků
- Ladění programu (debugging)
- Komunikaci s uživatelem

---

4 Proměnné a datové typy

4.1 Co je proměnná?

Proměnná je **pojmenované místo v paměti**, kde ukládáme data.

```
jmeno = "Jan"
vek = 25
```

- `jmeno` je proměnná, která obsahuje text `"Jan"`
- `vek` je proměnná, která obsahuje číslo 25

4.2 Základní datové typy

4.2.1 Celá čísla (`int`)

```
vek = 25
pocet_bodu = 150
```

4.2.2 Desetinná čísla (`float`)

```
prumerna_teplota = 15.7
nadmorska_vyska = 234.5
```

4.2.3 Text (`string` nebo `str`)

```
jmeno = "Jan Novák"
mesto = 'Praha'  # Fungují jednoduché i dvojité uvozovky
```

4.2.4 Logické hodnoty (`bool`)

```
je_student = True
je_zamestnanec = False
```

> ⚠ Pozor na velká písmena!
>
> V Pythonu záleží na velikosti písmen:
>
> - `True` a `False` - správně (s velkým písmenem)
> - `true` a `false` - CHYBA!

Pozor na velká písmena!

V Pythonu lze přiřadit více hodnot více proměnným jedním příkazem:

```
a, b = 3, 5
```

Toho lze využít např. při prohození hodnot mezi dvěma proměnnými:

```
# Klasický přístup přes pomocnou proměnnou
c = a   # zachytí se původní hodnota proměnné a
a = b   # nyní je možné hodnotu proměnné a změnit
b = c   # původní hodnota proměnné a je uchována v c
del c   # volitelně lze pomocnou proměnnou c smazat
```

4.3 Pojmenování proměnných

Pravidla:

- Začíná písmenem nebo podtržítkem: `jmeno`, `_temp`
- Může obsahovat písmena, čísla, podtržítka: `vek_2`, `pocet_bodu`
- **Nesmí** obsahovat mezery: `moje jmeno`
- **Nesmí** začínat číslem: `2vek`
- Je case-sensitive: `Jmeno` ≠ `jmeno`

Konvence (doporučení):

- Používejte malá písmena s podtržítky: `pocet_obyvatel`
- Výstižné názvy: `teplota` je lepší než `t`

---

5 Práce s čísly

5.1 Základní operace

```
# Sčítání
5 + 3  # 8

# Odčítání
```

```
10 - 4  # 6

# Násobení
3 * 4  # 12

# Dělení (výsledek je vždy float!)
10 / 3  # 3.3333...

# Celočíselné dělení
10 // 3  # 3

# Zbytek po dělení (modulo)
10 % 3  # 1

# Mocnina
2 ** 3  # 8 (2 na třetí)
```

5.2 Příklady s proměnnými

```
vyska = 180  # cm
hmotnost = 75  # kg

# BMI = hmotnost / (vyska v metrech)^2
vyska_m = vyska / 100
bmi = hmotnost / (vyska_m ** 2)

print(bmi)  # 23.148...
```

---

6 Práce s textem (stringy)

6.1 Spojování stringů

```
jmeno = "Jan"
prijmeni = "Novák"
```

```python
# Spojení (concatenation)
cele_jmeno = jmeno + " " + prijmeni
print(cele_jmeno)  # Jan Novák
```

6.2 f-stringy (doporučený způsob!)

f-stringy jsou nejmodernější a nejčitelnější způsob práce s textem:

```python
jmeno = "Jan"
vek = 25

# Starý způsob (concatenation)
zprava = "Jmenuji se " + jmeno + " a je mi " + str(vek) + " let."

# f-string (moderní způsob)
zprava = f"Jmenuji se {jmeno} a je mi {vek} let."

print(zprava)
# Výsledek: Jmenuji se Jan a je mi 25 let.
```

Proč f-stringy?

- Čitelnější kód
- Automatická konverze typů
- Můžete vkládat výrazy: `f"BMI: {hmotnost / (vyska ** 2)}"`

6.3 Převod mezi typy

```python
vek = 25
vek_text = str(vek)  # "25"

cislo_text = "100"
cislo = int(cislo_text)  # 100

desetinne_cislo_text = "3.14"
desetinne_cislo = float(desetinne_cislo_text)  # 3.14
```

---

7 Komunikace s uživatelem

7.1 Funkce `input()`

`input()` slouží k **načtení vstupu od uživatele**:

```
jmeno = input("Jak se jmenujete? ")
print(f"Zdravím vás, {jmeno}!")
```

Průběh:

```
Jak se jmenujete? Jan
Zdravím vás, Jan!
```

> ⚠ Pozor! `input()` vrací vždy STRING!
>
> I když uživatel zadá číslo, `input()` vrací text:
>
> ```
> vek = input("Kolik je vám let? ")  # vek je STRING!
> # Musíme převést:
> vek = int(input("Kolik je vám let? "))  # Teď je to INT
> ```

7.2 Praktický příklad

```
# Kalkulačka BMI
jmeno = input("Vaše jméno: ")
vyska = float(input("Výška v cm: "))
hmotnost = float(input("Hmotnost v kg: "))

vyska_m = vyska / 100
bmi = hmotnost / (vyska_m ** 2)

print(f"{jmeno}, vaše BMI je: {bmi:.2f}")
```

Příklad běhu:

```
Vaše jméno: Jan
Výška v cm: 180
Hmotnost v kg: 75
Jan, vaše BMI je: 23.15
```

---

8 Seznamy (Lists)

8.1 Co je seznam?

Seznam je **uspořádaná kolekce prvků**. Může obsahovat čísla, text, nebo cokoliv jiného:

```
teploty = [15, 18, 22, 19, 16]
mesta = ["Praha", "Brno", "Ostrava"]
smiseny = [1, "text", 3.14, True]  # Může být i mix!
```

8.2 Vytvoření seznamu

```
prazdny_seznam = []
cisla = [1, 2, 3, 4, 5]
jmena = ["Anna", "Petr", "Jana"]
```

8.3 Indexování

DŮLEŽITÉ: Python indexuje od 0!

```
mesta = ["Praha", "Brno", "Ostrava", "Plzeň"]

print(mesta[0])   # Praha (první prvek!)
print(mesta[1])   # Brno (druhý prvek)
print(mesta[3])   # Plzeň (čtvrtý prvek)
print(mesta[-1])  # Plzeň (poslední prvek)
print(mesta[-2])  # Ostrava (předposlední)
```

⚠ Pozor na indexování od 0!

Toto je častá chyba začátečníků:

```
cisla = [10, 20, 30, 40, 50]
print(cisla[1])  # 20, NE 10!
print(cisla[5])  # CHYBA! Index mimo rozsah
```

8.4 Délka seznamu

```
mesta = ["Praha", "Brno", "Ostrava"]
pocet = len(mesta)  # 3
```

8.5 Přidávání prvků

```
cisla = [1, 2, 3]
cisla.append(4)  # Přidá 4 na konec
print(cisla)  # [1, 2, 3, 4]

cisla.append(5)
print(cisla)  # [1, 2, 3, 4, 5]
```

8.6 Praktický příklad

```
# Program pro sběr teplot
teploty = []

teploty.append(float(input("Teplota v pondělí: ")))
teploty.append(float(input("Teplota v úterý: ")))
teploty.append(float(input("Teplota ve středu: ")))

print(f"Naměřené teploty: {teploty}")
print(f"První den: {teploty[0]}°C")
print(f"Poslední den: {teploty[-1]}°C")
```

---

9 Praktická cvičení

9.1 Cvičení 1: Kalkulačka vzdálenosti

Napište program, který:

1. Zeptá se uživatele na souřadnice dvou bodů (X1, Y1, X2, Y2)
2. Vypočítá vzdálenost mezi nimi pomocí Pythagorovy věty: $d = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2}$
3. Vypíše výsledek

Nápověda:

```
# Odmocninu získáte pomocí mocniny 0.5:
odmocnina = cislo ** 0.5
```

9.2 Cvičení 2: Průměrná teplota

Napište program, který:

1. Vytvoří prázdný seznam
2. Zeptá se uživatele na teploty pro 5 dní (použijte `append()`)
3. Vypočítá průměrnou teplotu: `prumer = sum(seznam) / len(seznam)`
4. Vypíše výsledek ve formátu: "Průměrná teplota byla: XX.X°C"

9.3 Cvičení 3: Seznam měst

Napište program, který:

1. Vytvoří seznam 3 českých měst
2. Vypíše první město
3. Vypíše poslední město
4. Přidá 2 další města pomocí `append()`
5. Vypíše celý seznam
6. Vypíše počet měst v seznamu

---

10 Shrnutí

10.1 Co jsme se naučili

Python je interpretovaný jazyk vhodný pro GIS
`print()` pro výpis, `input()` pro načtení vstupu
Základní datové typy: `int`, `float`, `str`, `bool`
Proměnné ukládají data
f-stringy pro práci s textem
Seznamy (`list`) - indexování od 0!
`append()` pro přidávání prvků
`len()` pro zjištění délky

10.2 Co bude příště?

V další lekci se naučíme:

- **Podmínky** (`if`, `elif`, `else`)
- **Cykly** (`for`, `while`)
- **Algoritmizace** (vyřešit úlohu pomocí kódu)

---

11 Cheatsheet

```
# === ZÁKLADY ===
print("text")              # Výpis
input("otázka? ")          # Načtení textu
int(input("číslo? "))      # Načtení čísla

# === DATOVÉ TYPY ===
int      # celá čísla: 5, -10, 100
float    # desetinná: 3.14, -0.5
str      # text: "ahoj", 'svět'
bool     # logická hodnota: True, False

# === ARITMETIKA ===
+, -, *, /     # základní operace
//             # celočíselné dělení
%              # zbytek po dělení
**             # mocnina

# === STRINGY ===
"ahoj" + "svět"            # spojení
f"Věk: {vek}"              # f-string
str(5)                     # převod na string

# === SEZNAMY ===
list_a = [1, 4, 3]          # vytvoření
list_a[0]                   # první prvek (index 0!)
list_a[-1]                  # poslední prvek
list_a[1] = 2               # přepsání (druhého) prvku
len(list_a)                 # délka
list_a.append(4)            # přidání prvku
sum(list_a)                 # součet prvků
```

12 Poznámky pro vyučujícího

12.1 Běžné chyby studentů

```
# 1. Zapomínají uvozovky
jmeno = Jan         #   CHYBA
jmeno = "Jan"       #   SPRÁVNĚ

# 2. Sčítání čísla + string
"Věk: " + 25        #   CHYBA
f"Věk: {25}"        #   SPRÁVNĚ

# 3. Zapomínají int() u inputu
vek = input()       # vrací string!
vek = int(input()) # vrací int

# 4. Indexování od 1
lista[1]            # to je DRUHÝ prvek!
lista[0]            # to je PRVNÍ prvek!
```

12.2 Časový plán (90 min)

Čas	Obsah
0-10 min	Úvod, motivace, proč Python
10-25 min	Hello World, první program, prostředí
25-40 min	Proměnné, datové typy, čísla
40-55 min	Stringy, f-stringy, input/output
55-70 min	Seznamy, indexování, append
70-85 min	Praktická cvičení
85-90 min	Shrnutí, Q&A

12.3 Klíčové momenty

- **Indexování od 0** - zdůraznit vícekrát!
- **Input vrací string** - ukázat chybu a řešení
- **f-stringy** - naučit hned, je to nejlepší způsob
- **Seznamy** - základ pro cykly (příští lekce)