Python akademie - lekce 2 - DD.MM.YYYY

02_03: Metody

Zajímavé odkazy z této lekce:

- Soupis všech metod pro datový typ string
- Soupis všech metod pro datový typ list
- Soupis všech metod pro datový typ tuple

Kromě **zabudovaných funkcí**, které jsou k dispozici v Pythonu, máme také tzv. *metody* datových typů.

Metody jsou v podstatě **nástroje**, které rozšiřují použití jednotlivých datových typů.

Jaký je tedy rozdíl mezi metodou a funkcí?

1. **Funkce** jsou obecnější, protože umí pracovat s různými datovými typy (`print`),
2. **Metody** jsou úzce zaměřené na konkrétní datový typ (`str`).

Ukázka **funkce**:

```
print(1)
```

```
print(1.00)
```

```
print('Radim')
```

Ukázka **metody**:

```
'radim'.upper()
```

```
'radim'.title()
```

```
'ahoj'.isnumeric()
```

```
seznam = [1,2,3,4,5]
print(seznam)
```

```
seznam.append(5)
print(seznam)
```

```
seznam.insert(0, "nula")
print(seznam)
```

Použití metody se odvíjí od konkrétní proměnné (hodnoty), na kterou s **TEČKOU** napojím příslušnou metodu.

Za metodou je nutné zapsat závorku (často prázdnou, ale není to pravidlo).

Metody pro `string`

Níže najdete tabulku s metodami, které se nám budou hodit (ne všechny):

Metoda	Použití
`upper`	převedene zadaný string na velká písmena
`lower`	převedene zadaný string na malá písmena
`isalpha`	vrátí `True` pokud jsou všechny znaky písmena
`isnumeric`	vrátí `True` pokud jsou všechny znaky číselné znaky
`istitle`	vrátí `True` pokud je první znak velké písmeno
`split`	rozdělí string pomocí zadaného znaku, jinak podle mezer a newlinů (vlevo)
`rsplit`	rozdělí string pomocí zadaného znaku, jinak podle mezer a newlinů (vpravo)
`strip`	ořízne zadaný string o specifikovaný symbol (na začátku a na konci) (vlevo)

Metoda	Použití
`rstrip`	ořízne zadaný string o specifikovaný symbol (na začátku a na konci) (vpravo)
`replace`	v zadaném stringu nahradí konkretní symboly, jinými symboly

```
'Radim Jedlicka'.split(' ')
```

```
'Radim Jedlicka'.strip('Radki')
```

```
'Radim Jedlicka'.replace('i', ' ')
```

Metody pro `list`

Níže najdete tabulku s metodami, které se nám budou hodit (ne všechny):

Metoda	Použití
`append`	přidá hodnotu na poslední index listu
`insert`	vloží hodnotu na daný index listu. Zbytek hodnot posune o index +1
`copy`	vytvoří tzv. *shallow copy* původního listu
`count`	vrací číslo, kolikrát se zadaný údaj nachazí v listu
`sort`	vrací seřazený list
`index`	vrací index prvního výskytu zadané hodnoty

```
list1 = [1, 2, 3]
list2 = [4, 5]
list3 = [6, 7, 8]
```

```
list1 + list2 + list3
```

```
list1.append(list2)
print(list1)
```

```
list1.insert(0, list3)
print(list1)
```

Metody pro `tuple`

Níže najdete tabulku s metodami, které se nám budou hodit (všechny):

Metoda	Použití
`count`	vrací číslo, kolikrát se zadaný údaj nachazí v listu
`index`	vrací index prvního výskytu zadané hodnoty

```
muj_tuple = (6, 7, 8, 6, 7, 8, 1, 2, 3, 4, 4, 5)
```

```
muj_tuple.count(4)
```

```
muj_tuple.index(4)
```