Pole

E 3011

Jan Böhm

RECETOX

April 5, 2023

Co nás dnes čeká

1. Pole
2. Vektory

Pole - definice

Pole

Pole (anglicky list) je datová struktura v Pythonu. Je to posloupnost hodnot. Vytváří se pomocí hranatých závorek.

```
# list init
listNumeric = [3, 0, 1, 9]
listString = ["apple", "banana", "cherry"]
listBool = [True, True, False, True]
emptyList = []
```

Pole - vlastnosti

Pole - indexování

Python indexuje prvky od 0. Tzn. že první hodnota má index 0, druhá má index 1, atd. Pokud chceme brát hodnoty od konce, poslední je -1, předposlední -2, ...

```
1  listString = ["apple", "banana", "cherry"]
2  print(listString[1])
3  print(listString[0])
4  print(listString[3])
5
6  print(listString[-2])
7
8  # change value at given index
9  listString[1] = "beer"
10 print(listString[1])
```

Pole - vlastnosti

Pole - přidávání hodnot

Existuje více způsobů, jak přidávat hodnoty do pole. Ukážeme si dva z nich:

- `list.append(value)` – přidá value na konec pole
- `list.insert(index, value)` – přidá value na pozici před index

```
1 listString = ["apple", "banana", "cherry"]
2
3 listString.append("dragonfruit")
4 listString.insert(1,"avocado")
5
6 print(listString)
```

Pole - vlastnosti

Pole - odstraňování hodnot

Existuje více způsobů, jak odstraňovat hodnoty z pole. Ukážeme si tři z nich:

- `del list[index]` – odstraní hodnotu na pozici index
- `list.remove(value)` – odstraní první výskyt value z pole
- `list.pop(index = -1)` – odstraní hodnotu na pozici index a vrátí ji

```
1 listString = ["apple", "banana", "cherry",
2                "banana", "dragonfruit"]
3 del listString[2]
4 listString.remove("banana")
5 fruit = listString.pop()
6
7 print(listString)
8 print(fruit)
```

Pole - vlastnosti

Pole - několik dalších užitečností.

- `list1 + list2` – spojení (concat) dvou listů
- `len(value)` – počet prvků v poli (délka pole)
- `list.sort(reverse = False)` – seřadí pole (s `reverse = True` pozpátku

```
1 fruits = ["apple", "banana", "cherry"]
2 veggies = ["pepper", "potato", "carrot", "onion"]
3
4 food = fruits + veggies
5 print(len(food))
6 food.sort()
7 print(food)
8 food.sort(reverse = True)
9 print(food)
```

Posloupnost

Funkce `range(start, stop, step)` vytváří aritmetickou posloupnost od `start` (inclusive) po `stop` (exclusive) z krokem `step`. Výsledkem je objekt typu `range`, který ale můžeme pomocí `list()` převést na pole:

```
1 print(list(range(5)))
2 print(list(range(1,5)))
3 print(list(range(10,0,-1)))
```

Aplikování funkce na každý prvek v poli

```
1 L = [1,2,3]
2 L = L + 2
```

Aplikování funkce na každý prvek v poli

```
1 L = [1,2,3]
2 L = L + 2
```

Tento kus kódu ohlásí chybu, protože dělení je operace která potřebuje dvě čísla, ne list a číslo. Pokud chceme nějakou operaci provést s každým prvkem v poli, existuje na to typicky "pythonovská" konstrukce:

Aplikování funkce na každý prvek v poli

```
1 L = [1,2,3]
2 L = L + 2
```

Tento kus kódu ohlásí chybu, protože dělení je operace která potřebuje dvě čísla, ne list a číslo. Pokud chceme nějakou operaci provést s každým prvkem v poli, existuje na to typicky "pythonovská" konstrukce:

```
1 L = [1,2,3]
2 L = [l + 2 for l in L]
3 print(L)
```

Nebo pokud například si chceme připravit posloupnost od 0 do 1 s krokem 0.1, tak můžeme takto:

```
1 seq = [x/10 for x in list(range(11))]
```

Statistiky

Statistiky

Nad polem čísel `L` můžeme chtít spočítat statistiky. Např.

- minimum `min(L)` a maximum `max(L)`
- součet `sum(L)`
- počet prvků `len(L)`
- ...

Co nás dnes čeká

1. Pole
2. Vektory

Vektory

Vektory v Pythonu

Pro práci s vektory existují specializované moduly do Pythonu, např. `numpy`. My si však vystačíme se základním Pythonem a potřebné funkce si naimplementujeme.

Co je vektor?

Vektorem budeme v Pythonu rozumět pole čísel `u = [1, -2, 1/3]`.

Co umíme dělat s vektory I

- Umím vytvořit nulový vektor $\mathbf{0} = (0, 0, \ldots, 0)$ libovolné délky n.
- Pro libovolný vektor $\boldsymbol{u}$ mohu spočítat jeho (euklidovskou) velikost $\|\boldsymbol{u}\| = \sqrt{u_1^2 + u_2^2 + \cdots + u_n^2}$
- Pro libovolný vektor $\boldsymbol{u} \in \mathbb{R}^n$ a skalár $c \in \mathbb{R}$ mohu spočítat $c \cdot \boldsymbol{u} = (cu_1, cu_2, \ldots, cu_n)$.
- Dva vektory $\boldsymbol{u}$ a $\boldsymbol{v}$ stejné délky můžeme sečíst (nebo odečíst): $\boldsymbol{u} \pm \boldsymbol{v} = (u_1 \pm v_1, u_2 \pm v_2, \ldots, u_n \pm v_n)$.
- Pro libovolnou dvojici stejně dlouhých vektorů umím spočítat jejich euklidovskou vzdálenost $\|\boldsymbol{u} - \boldsymbol{v}\|$
- Lineární kombinace $c\boldsymbol{u} + d\boldsymbol{v} + \ldots$

Co umíme dělat s vektory II: násobení

Je více způsobů, jak násobit vektory stejné délky.

- Skalární součin $\boldsymbol{u} \cdot \boldsymbol{v} = u_1 v_1 + u_2 v_2 + \ldots u_n v_n$.
- Odchylka vektorů $\angle \boldsymbol{uv} = \theta$, kde $\cos\theta = \frac{\boldsymbol{u}\cdot\boldsymbol{v}}{\|\boldsymbol{u}\|\|\boldsymbol{v}\|}$
- Vektorový součin (v $\mathbb{R}^3$) $\boldsymbol{u} \times \boldsymbol{v} = \boldsymbol{n}\|\boldsymbol{u}\|\|\boldsymbol{v}\| \sin\theta$. kde θ je odchylka $\angle \boldsymbol{uv}$ a $\boldsymbol{n}$ je vektor kolmý na $\boldsymbol{u}$ a $\boldsymbol{v}$.
- Smíšený součin $\boldsymbol{u} \cdot (\boldsymbol{v} \times \boldsymbol{w})$.

Vektorová algebra

Skalární součin

Skalární součin $\boldsymbol{u} \cdot \boldsymbol{v}$ nám umožňuje poznat, zda jsou 2 vektory na sebe kolmé (když je 0).

Vektorový součin

Velikost vektorového součinu v $\mathbb{R}^3$ $\boldsymbol{u} \times \boldsymbol{v}$ je obsah rovnoběžníku tvořeného vektory $\boldsymbol{u}$ a $\boldsymbol{v}$

Smíšený součin

Smíšený součin v $\mathbb{R}^3$ $\boldsymbol{u} \cdot (\boldsymbol{v} \times \boldsymbol{w})$ je objem rovnoběžnostěnu tvořeného vektory $\boldsymbol{u}$, $\boldsymbol{v}a\boldsymbol{w}$.