Seznamy

Umíme informatiku >> Témata >> Programovací jazyk Python >> Datové typy a jejich využití >> Seznamy v Pythonu

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

Vytvořte posloupnost s diferencí 3 a spojte s listem *seznam*

```
Začněte programovat nebo generovat kód s AI.
```

Co bude výstupem

```
seznam_pismen=list("tramvaj")
seznam_pismen[:]+seznam
```

Vysvětlete

```
seznam[0], seznam[1] = seznam[1], seznam[0]
print(seznam)
```

Vysvětlete

```
print(max([4, 1][1], [4, 2][1]))
[[6, 9], [3, 1]][0][1]
```

Zjednodušte pomocí enumerate

```
seznam = [1, 2, 3, 4, 5]
for i in range(len(seznam)):
    print(i, seznam[i])
```

Počet čísel menších než k

💎 💎 Napište program, který vrátí počet čísel menších než k v seznamu cisla.

Použijte List Comprehension `[novy for prvek in seznam if podminka]`

vytvořte seznam *kontrola* takový, že `kontrola[i] = 1` když je `cisla[i] < k`.

```
cisla = [1, 2, 3, 4, 5,0,-1]
k = 3
```

naucse.python.cz - Iterátory n-tic

List comprehension - https://www.python-programator.cz/clanky/list-comprehension

Součin nenulových prvků

💎 💎 Napište program, která pro zadaný seznam čísel vrátí součin všech nenulových čísel v seznamu.

```
cisla=[0,1,2,5,0,1,0,]
print(cisla)
```

```
import math
math.prod(cisla)
```

Úprava seznamu, zkopírování seznamu

Popište rozdíl mezi `seznam[2]="něco"` a `seznam.insert(2,"něco")` a `seznam.append("něco")`

```
Začněte programovat nebo generovat kód s AI.
```

Jaký je rozdíl mezi `mylist.remove(3)`, `mylist.pop(3)`, `del mylist[3]`

```
mylist = [3, 33, 66, 67, 69, 99]
```

Úkol - odstraňte ze seznamu všechny nuly užitím while a metody `remove`.

```
seznam = [0, 1, 0, 2, 3, 0]
```

Pozor, **kopie** můžeme vytvářet jen metodou copy nebo řezem

```
novylist = mylist # Obě proměnné ukazují na stejný seznam v paměti
novylist[0]="nic"
print(mylist)
```

```
#novylist = mylist.copy()
novylist = mylist[:]
novylist[0]="nic"
print(mylist)
```

```
[4, 2, 1] > [4, 5] # porovnání lexikograficky, jako ve slovníku
```

Třídění

Jaký je rozdíl mezi `sorted(fib)` a `fib.sort()`

```
fib=[21,0,1,13,1,2,3,5,8]
```

Co bude výstupem

```
Jupiter=["Io", "Europa", "Ganymed", "Callisto"]
Jupiter.insert(1, "Titan")
Jupiter.sort()
print(Jupiter)
```

```
Jupiter=["Io", "Europa", "Ganymed", "Callisto"]
```

```
def delky(e):
  return len(e)

Jupiter.sort(key=delky)
print(Jupiter)
```

Výraz lambda x: vytváří anonymní funkci, písmeno λ z teorie funkcí. Je to jen kratší zápis — hodí se, když potřebuješ funkci jen na jednom místě, například jako argument pro sorted(), map(), filter() atd.

`lambda argumenty: výraz`

```
Jupiter.sort(key=lambda e: len(e),reverse = True)
print(Jupiter)
```

Sudoku

💎 💎 💎 Napište program který zkontroluje, zda v zadaném řádku čísel je každé číslo od 1 do 9 právě jedenkrát (a nic jiného).

```
row = [1, 2, 3, 4, 5, 6, 6, 8, 9]
```

Procvičování

Umíme informatiku - rozhodovačka - seznamy v Pythonu - střední (štít 4), těžké (štít 3)

Průměr

💎 💎 Napište program pro průměr z čísel v seznamu numbers zaokrouhlený na 1 desetinné místo. Zaukrouhlování děláme pomocí funkce round.

```
from statistics import mean

cisla=list(range(10))
round(mean(cisla),1)
```

Každý druhý

💎 💎 Napište program, který ze zadaného seznamu čísel vypíše každé druhé číslo (počínaje prvním).

Použije list comprehension

`[novy for i, cislo in enumerate(cisla) if podminka]`

```
Začněte programovat nebo generovat kód s AI.
```

Monotonní posloupnost

💎 Napište funkci monotonic(numbers), která vrátí True, pokud je seznam čísel numbers rostoucí nebo klesající.

```
def monotonic(numbers):
    return numbers == sorted(numbers) or numbers == sorted(numbers, reverse=True)

print(monotonic([1, 12, 30, 35]))
```

```
def monotonic(numbers):
    increasing = all(numbers[i] <= numbers[i + 1] for i in range(len(numbers) - 1))
    decreasing = all(numbers[i] >= numbers[i + 1] for i in range(len(numbers) - 1))
    return increasing or decreasing

print(monotonic([1, 12, 30, 35]))
```