

## Cvícenie 2

### Instrukcie:

- Vypracujte všetky ulohy. Na cvícení sa pokúste vypracovať čo najviac uloh a ulohy, ktoré nestihnete na cvícení, potom vypracujte doma.
- **POZOR!** Všímajte si, či funkcia, ktorú máte definovať, má niečo vrácať alebo či má niečo vypisovať! V prípade, že má funkcia niečo **vrať**, musí v nej byť použitý príkaz **return!**
- **V prípade, že sa na niektorej ulohy zaseknete, pýtajte sa cvičiaceho alebo odborného praktika.**

### Cast 1: funkcie

1A. Vytvorte skript, ktorý definuje funkciu *tretia\_mocnina*, ktorá pre vstupný argument *a* **vrať** hodnotu  $a*a*a$ .

Napríklad: volanie *tretia\_mocnina(2)* **vrať** hodnotu 8; volanie *tretia\_mocnina(3)* **vrať** hodnotu 27.

1B. Upravte skript z ulohy 1A tak, aby skript pomocou funkcie *tretia\_mocnina* vypísal tretiu mocninu čísla 5.

2A. Vytvorte skript, ktorý definuje funkciu *priemer\_troch*, ktorá pre vstupné argumenty *a, b, c* **vrať** hodnotu  $(a+b+c)/3$ .

Napríklad: volanie *priemer\_troch(1,2,3)* **vrať** hodnotu 2.0; volanie *priemer\_troch(1, 2, 4.5)* **vrať** hodnotu 2.5

2B. Upravte skript z ulohy 2A tak, aby skript pomocou funkcie *priemer\_troch* vypísal priemer čísel 5,6,7.

### Cast 2: funkcie a for cykly

V ulohách 1 až 4 sa budem odvolávať na nasledujúci skript:

```
1 def vypis(n):
2     for i in range(n):
3         print(i)
```

Funkcia *vypis(n)* definovaná v tomto skripte vypíše čísla od 0 po  $n-1$ .

1. Vytvorte skript, ktorý definuje funkciu s parametrom *n*, ktorá **vypíše** čísla od 1 po *n*. Váš funkcia by mala vyzeráť ako funkcia *vypis(n)* na obrázku vyššie a od funkcie *vypis(n)* by sa mala líšiť iba v riadku číslo 3.

Napríklad: funkcia pre hodnotu parametra  $n=10$  **vypíše** čísla 1 2 3 4 5 6 7 8 9 10, každé na nový riadok.

2. Vytvorte skript, ktorý definuje funkciu s parametrom *n*, ktorá **vypíše** prvých *n* čísel väčších ako 100. Váš funkcia by mala vyzeráť ako funkcia *vypis(n)* na obrázku vyššie a od funkcie *vypis(n)* by sa mala líšiť iba v riadku číslo 3.

Napríklad: funkcia pre hodnotu parametra  $n=3$  vypíše čísla 101, 102 a 103, každé na nový riadok.

3. Vytvorte skript, ktorý definuje funkciu s parametrom  $n$ , ktorá **vypise** prvých  $n$  párnych čísel (začínajúc dvojkou). Urobte to bez použitia podmienok (if-ov). Vaša funkcia by mala vyzerat ako funkcia `vypis(n)` na obrázku vyššie a od funkcie `vypis(n)` by sa mala líšiť iba v riadku číslo 3.

Napríklad: funkcia pre hodnotu parametra  $n=1$  **vypise** číslo 2. Funkcia pre hodnotu parametra  $n=5$  **vypise** čísla 2 4 6 8 10, každé na nový riadok. Funkcia pre hodnotu parametra  $n=10$  **vypise** čísla 2 4 6 8 10 12 14 16 18 20, každé na nový riadok.

4. Vytvorte skript, ktorý definuje funkciu s parametrom  $n$ , ktorá **vypise** klesajúcu postupnosť čísel od  $n$  po 1. Vaša funkcia by mala vyzerat ako funkcia `vypis(n)` na obrázku vyššie a od funkcie `vypis(n)` by sa mala líšiť iba v riadku číslo 3.

Napríklad: funkcia pre hodnotu parametra  $n=1$  **vypise** číslo 1. Funkcia pre hodnotu parametra  $n=5$  **vypise** čísla 5 4 3 2 1, každé na nový riadok. Funkcia pre hodnotu parametra  $n=10$  **vypise** čísla 10 9 8 7 6 5 4 3 2 1, každé na nový riadok.

5. Vytvorte skript, ktorý definuje funkciu s parametrom  $n$ , ktorá **vratí** súčet  $1^2 + 2^2 + 3^2 + \dots + n^2$ . (Pre tento súčet síce existuje vzorec, ale od Vás teraz chceme, aby ste to zrážali pomocou for-cykľu.)

Napríklad: funkcia pre hodnotu parametra  $n=1$  **vratí** číslo 1, pre hodnotu parametra  $n=3$  **vratí** číslo 14, pre hodnotu parametra  $n=5$  **vratí** číslo 55.

6. Definujte funkciu, ktorá **vypise** prvých  $N$  členov **aritmetickej postupnosti**  $a_{i+1} = a_i + d$  na základe parametrov  $a_0$ ,  $d$  a  $N$ . Čiže na základe parametrov  $a_0$ ,  $d$  a  $N$  funkcia vypíše postupnosť čísel  $a_0, a_1, a_2, \dots, a_{N-1}$ , v ktorej platí, že  $a_{i+1} = a_i + d$ .

Napríklad: pre hodnoty parametrov  $a_0=0, d=3, N=5$ , funkcia **vypise** 0,3,6,9,12.  
Pre hodnoty parametrov  $a_0=5, d=-3, N=4$ , funkcia **vypise** 5, 2, -1, -4.

7. Teraz vyriešte úlohu 6 tak, že v definícii funkcie nepoužijete operáciu `*` (t.j. operáciu násobenia).

8. Pozrite si skript `meranie_casu.py`, ktorý nájdete na webstránke predmetu. V tomto skripte definujem funkciu `arit`, ktorá rieši úlohu 6 a využíva operáciu `*`. V skripte takisto definujem funkciu `arit2`, ktorá tiež rieši úlohu 6, ale operáciu `*` už nepoužíva. V skripte potom porovnavam rýchlosť týchto funkcií. Spustíte skript `meranie_casu.py`. Budete si môcť všimnúť, že funkcia `arit2` je rýchlejšia ako funkcia `arit`.

9. Definujte funkciu, ktorá **vypise** prvých  $N$  členov **geometrickej postupnosti**  $a_{i+1} = a_i * r$  na základe parametrov  $a_0$ ,  $r$  a  $N$ . Čiže na základe parametrov  $a_0$ ,  $r$  a  $N$  funkcia vypíše postupnosť čísel  $a_0, a_1, a_2, \dots, a_{N-1}$ , v ktorej platí, že  $a_{i+1} = a_i * r$ .  
*Pomôcka:* umocňovanie sa v pythone robí pomocou symbolov `**`. Napríklad `2**3` je 8.

Napríklad: pre hodnoty parametrov  $a_0=1, r=2, N=5$ , funkcia **vypise** 1,2,4,8,16.  
Pre hodnoty parametrov  $a_0=2, r=3, N=4$ , funkcia **vypise** 2, 6, 18, 54.

10. Teraz vyriešte úlohu 9 tak, že v definícii funkcie nepoužijete operáciu umocnenia.

11. Ak sa funkcie, ktoré ste vytvorili v úlohách 9 a 10, od seba líšia, porovnajme ich rýchlosť. Na porovnanie rýchlostí funkcií sa inšpirujte skriptom `meranie_casu.py`, ktorý nájdete na webstránke predmetu. Pri porovnávaní rýchlostí môžete funkcie spustiť napríklad s parametrami  $a_0=1, r=2, N=50\,000$ . Pri porovnávaní rýchlostí zakomentujte volania funkcie `print!` Ak by vám niektorý výpočet trval príliš dlho, stlačte `Ctrl+C`. Výpočet tým prerušite.

12. Definujte funkciu, ktorá **vypíše** prvých  $N$  členov **geometrickeho radu** na základe parametrov  $a_{\{0\}}$ ,  $r$  a  $N$ . (**Geometrický rad** je definovaný tak, že  $i$ -ty člen radu predstavuje súčet prvých  $i$  členov geometrickej postupnosti.) Definujte funkciu tak, aby jej definícia obsahovala iba jeden for cyklus.

Napríklad pre hodnoty parametrov  $a_{\{0\}}=1$ ,  $r=2$ ,  $N=5$ , funkcia **vypíše** 1, 3 (=1+2), 7 (=1+2+4), 15 (=1+2+4+8), 31 (=1+2+4+8+16).

Pre hodnoty parametrov  $a_{\{0\}}=2$ ,  $r=3$ ,  $N=4$  funkcia **vypíše** 2, 8, 26, 80.

13. Ak  $r$  je v absolútnej hodnote menšie ako 1, tak s rastúcim  $N$  sa členy geometrickeho radu blížia k hodnote

$$a_{\{0\}}/(1-r).$$

Tomuto hovoríme, že pre  $r$  v absolútnej hodnote menšej ako 1 geometrický rad konverguje k hodnote  $a_{\{0\}}/(1-r)$ . Napríklad geometrický rad s parametrami  $a_{\{0\}}=1$ ,  $r=0.5$  konverguje k hodnote 2. Otestujte, či sa čísla, ktoré vypíše vaša funkcia z úlohy 12 pre parametre  $a_{\{0\}}=1$ ,  $r=0.5$ ,  $N=100$ , blížia k hodnote 2. Ak nie, potom máte niekde chybu.

Napríklad: pre hodnoty parametrov  $a_{\{0\}}=2$ ,  $r=0.5$ ,  $N=100$  sa hodnoty blížia k hodnote 4. Pre hodnoty parametrov  $a_{\{0\}}=2$ ,  $r=0.75$ ,  $N=100$  sa hodnoty blížia k hodnote 8.

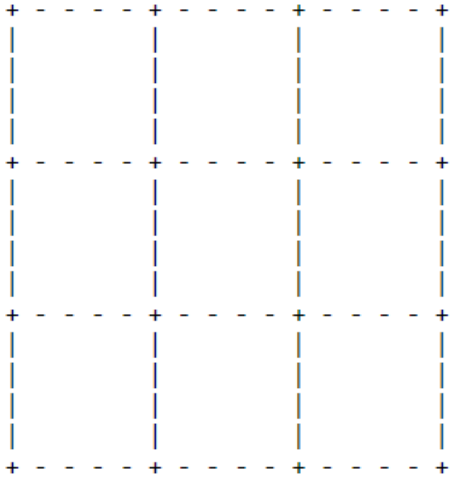
14. Teraz vyriešte úlohu 12 tak, že v definícii funkcie nepoužijete operáciu umocnenia. Definícia by stále mala obsahovať iba jeden for cyklus.

15. Porovnajte rýchlosť vašich funkcií z úloh 12 a 14 a rýchlosť funkcie, ktorú nájdete v skripte *neefektivny\_skript.py* na webstránke predmetu. Funkcia v tomto skripte tiež rieši úlohu 12, ale robí to pomocou dvoch do seba vnorených for cyklov. Na porovnanie rýchlosti funkcií sa opäť inšpirujte skriptom *meranie\_casu.py*, ktorý nájdete na webstránke predmetu. Ak budete porovnávať s funkciou zo skriptu *neefektivny\_skript.py*, môžete napríklad použiť parametre  $a_{\{0\}}=1$ ,  $r=2$ ,  $N=1000$ . Ak budete porovnávať iba funkcie z úloh 12 a 14, potom môžete napríklad použiť parametre  $a_{\{0\}}=1$ ,  $r=2$ ,  $N=50000$ . Pri porovnávaní rýchlosti zakomentujte volania funkcie print! Ak by vám niektorý výpočet trval príliš dlho, stlačte Ctrl+C. Výpočet tým prerušite.

16. Vytvorte skript, ktorý pomocou for-cyklov a funkcie print definuje funkciu *grid* s parametrom  $n$ , ktorá do konzoly vykreslí mriežku s  $n$  riadkami a  $n$  stĺpcami. *Vsimnite si pomocku uvedenú nižšie!* Mriežka s dvoma riadkami a dvoma stĺpcami (teda  $n=2$ ) by mala vyzerat nasledovne:

```
+ - - - - + - - - - +
|         |         |
+ - - - - + - - - - +
|         |         |
+ - - - - + - - - - +
```

Pre  $n=3$  by mriežka zase mala vyzerat nasledovne:



*Pomocka:* funkcia print automaticky vypisuje na novy riadok. Toto nastavenie sa ale da zmenit zmenou hodnoty parametra end. Napriklad prikazy:

```
print('+', end=' ')\nprint('-')
```

vypisu:  
+ -

Dalsi priklad:  
Prikazy:

```
print('+', end=', ')\nprint('-')
```

vypisu:  
+,-