

C2184 Úvod do programování v Pythonu

Povinné domácí úkoly

Úkoly v této sadě řešte do připravených souborů `is_number.py`, `mean.py`, `patients.py`, `triangles.py`, které pak odevzdáte do odevzdáárny. Doctesty a typové anotace v těchto souborech považujte za součást zadání (proto v tomto zadání nejsou vzorové vstupy a výstupy ani testovací buňky).

Doctesty spustíte z příkazové řádky:

```
python -m doctest ***.py          # default mode
python -m doctest ***.py -v       # verbose mode
```

Úkoly 8.4, 8.5, 8.6 na sebe navazují, tj. v každém dalším úkolu se očekává využití funkcí implementovaných v předchozích úkolech.

DÚ 8.1: Je to číslo?

V souboru `is_number.py` doplňte funkci `is_number`, která bere jako parameter řetězec a vrací hodnotu `True`, pokud tento řetězec je validním zápisem reálného čísla, `False` v opačném případě.

DÚ 8.2: Bezpečný průměr

V souboru `mean.py` doplňte funkci `mean`, která vezme seznam čísel a vrátí jejich průměr.

Zamyslete se nad tím, jaké problémy by mohly během výpočtu nastat, a napište funkci tak, aby neskončila chybou. Pokud průměr nelze spočítat, vraťte hodnotu `math.nan` (*not a number*).

DÚ 8.3: Debuggování

V souboru `patients.py` je připravený program, který počítá a vypisuje BMI pacientů Boba, Alice a Cyrila. U pacientů s BMI mimo normální rozsah (18.5–25) navíc vypíše vykřičník. Nakonec vypíše průměr a medián BMI.

V tomto programu je však několik chyb. Zkuste tyto chyby najít, program opravit a odevzdat. Pomozte si debuggovacími postupy z přednášky:

- Debuggovací režim ve VSCode (nebo jiném vývojovém prostředí)

- Testování (doctest)
- Příkaz assert
- Statická typová kontrola (mypy)

Ujistěte se, že program vypisuje přesně to, co má (napište si doctesty) – jinak vám úkol nebude uznán.

Očekávaný výstup:

```
Bob's BMI is: 27.7 !
Alice's BMI is: 21.8
Cyril's BMI is: 22.8
-----
Average BMI: 24.1
Median BMI: 22.8
```

DÚ 8.4: Obsah trojúhelníku

V libovolném trojúhelníku musí délky stran a , b , c splňovat trojúhelníkové nerovnosti:

$$a + b > c$$

$$a + c > b$$

$$b + c > a$$

Když a , b , c nesplňují byť jednu z těchto nerovností, pak z nich nelze vůbec trojúhelník zkonstruovat.

Pokud známe délky všech tří stran trojúhelníku, umíme spočítat jeho obsah S pomocí Heronova vzorce:

$$S = \sqrt{s(s-a)(s-b)(s-c)}$$

$$s = \frac{a+b+c}{2}$$

(Zkuste si promyslet, co by se stalo, kdybychom do tohoto vzorce chtěli dosadit délky nesplňující trojúhelníkovou nerovnost.)

Úkol:

V souboru `triangles.py` doplňte funkci `triangle_area`, která bere délky stran trojúhelníku a vrací obsah trojúhelníku.

Pokud zadané délky nemohou být délkami stran trojúhelníku (nesplňují trojúhelníkové nerovnosti), vyhodte výjimku `ValueError` s příslušnou hláškou.

DÚ 8.5: Největší trojúhelník

V souboru `triangles.py` doplňte funkci `largest_triangle`, která bere seznam trojic, kde každá trojice reprezentuje délky stran v jednom trojúhelníku.

Funkce vybere z těchto trojúhelníků ten s největším obsahem a vrátí ho jako návratovou hodnotu.

Trojice, které nesplňují trojúhelníkové nerovnosti, bude funkce ignorovat.

Když ani jeden z trojúhelníků nebude splňovat trojúhelníkové nerovnosti, funkce vyhodí výjimku `ValueError` s příslušnou hláškou.

DÚ 8.6: Největší trojúhelník - interaktivně

Naším úkolem v této úloze je vytvořit interaktivní program `triangles.py`, který od uživatele z příkazové řádky načte délky strany několika trojúhelníků, vybere trojúhelník s největším obsahem a ten vypíše na výstup.

Velkou část práce už máme za sebou - výběr největšího trojúhelníku zabezpečí funkce `largest_triangle`.

Samotné načtení vstupu a vypsání výstupu zabezpečí funkce `main`, která je už připravena v šabloně a není třeba ji upravovat.

Doplňte funkci `largest_triangles_io` (IO je zkratka od input-output), která vezme řetězec - načtený vstup od uživatele - a vrátí řetězec, který se má uživateli vypsát na terminál. Načtení a výpis (`main`) je od samotného zpracování (`largest_triangles_io`) odděleno kvůli tomu, aby bylo možné napsat doctesty (v doctestu nelze ovlivnit, co bude výsledkem funkce `input`).

Na vstupu uživatel odděluje jednotlivé trojúhelníky středníkem (;) a v rámci každého trojúhelníku jsou strany odděleny spojovníkem (-). Na výstupu jsou čísla vypsána jako float se dvěma desetinnými místy.

Vzorový vstup:

```
3.0-4.0-5.0; 3-4-8; 6-7-6
```

Vzorový výstup:

```
Largest triangle is: 6.00-7.00-6.00
```

(Další vzorové vstupy/výstupy viz doctesty)