Skúšobné zadanie na druhé cvičenie

13. decembra 2022

1 Premenné (7.5 boda)

Rozšírte Vašu funkciu z domáceho zadania tak, aby vedela vyhodnotiť aj výraz, v ktorom sú premenné. Hodnoty premenných sú uvedené v slovníku, ktorý je daný ako druhý argument funkcie.

To znamená, že Vaša funkcia bude mať definíciu

```
def eval_expr(s,d={}):
```

a bude vracať hodnoty takto

```
>>> eval_expr("1 x +",{"x":2})
3
>>> eval_expr("x x +",{"x":2})
4
>>> eval_expr("2 x * x +",{"x":3})
9
>>> eval_expr("x",{"x":3})
3
>>> eval_expr("x y +",{"x":3,"y":2})
5
>>> eval_expr("x y + 2 *",{"x":3,"y":2})
10
>>> eval_expr("x y + y *",{"x":3,"y":2})
10
```

1.1 Pomôcky

Zrejme budete musieť rozoznávať, či je slovo na vstupe číslo alebo premenná. Premenné rozpoznáte podľa toho, že sú to kľúče v danom slovníku (operátor `in`). Ak niečo nie je v slovníku a nie je to aritmetická operácia, treba sa k tomu chovať ako k číslu. Alternatívne môžete testovať, či je niečo číslo pomocou `str.isdigit`.

2 Infixová notácia (7.5 boda)

Napíšte funkciu `to_infix`, ktorá dostane ako argument číselný výraz v postfixovej notácii a vráti reťazec, ktorý bude v normálnej infixovej notácii. Pritom reťazec, ktorý funkcia vracia má obsahovať každý subvýraz dôsledne ozátvorkovaný, aby sme nemuseli špekulovať o prioritách. Čísla, operácie aj zátvorky majú byť oddelené medzerami.

Uvedomte si, že to je veľmi podobné zadanie ako domáce, aj keď to na to možno na prvý pohľad nevyzerá. Namiesto pomocných výsledkov budete na zásobník dávať podvýrazy prevedené do infixovej notácie.

```
>>> to_infix("1 2 *")
'( 1 * 2 )'
>>> to_infix("1 2 * 3 +")
'( ( 1 * 2 ) + 3 )'
>>> to_infix("1 2 3 * +")
'( 1 + ( 2 * 3 ) )'
>>> to_infix("1")
'1'
```

2.1 Pomôcky

Tuto zase vôbec netreba čísla rozoznávať. Buďto slovo je aritmetická operácia alebo nie. Tento prístup má navyše pozitívny efekt v tom, že funkcia bude fungovať aj na výrazy s premennými.

3 Prémia (5 bodov)

Napíšte funkciu `to_postfix`, ktorá je inverzná k `to_infix`.

```
>>> to_postfix("1")
'1'
>>> to_postfix("( 1 + 2 )")
'1 2 +'
>>> to_postfix("( 10 * ( 1 + 2 ) )")
'10 1 2 + *'
>>> to_postfix("( ( 10 + 1 ) * ( 2 / 3 ) )")
'10 1 + 2 3 / *'
```