

# Skúšobné zadanie na druhé cvičenie

20. mája 2016

## 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 vracat 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
```

## 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'
```

## 3 Prémia (5 bodov)

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