

Python, základní kameny až skály II

Tomáš Svoboda

B4B33RPH, 2018-10-23

funkce pravé a modifikátory

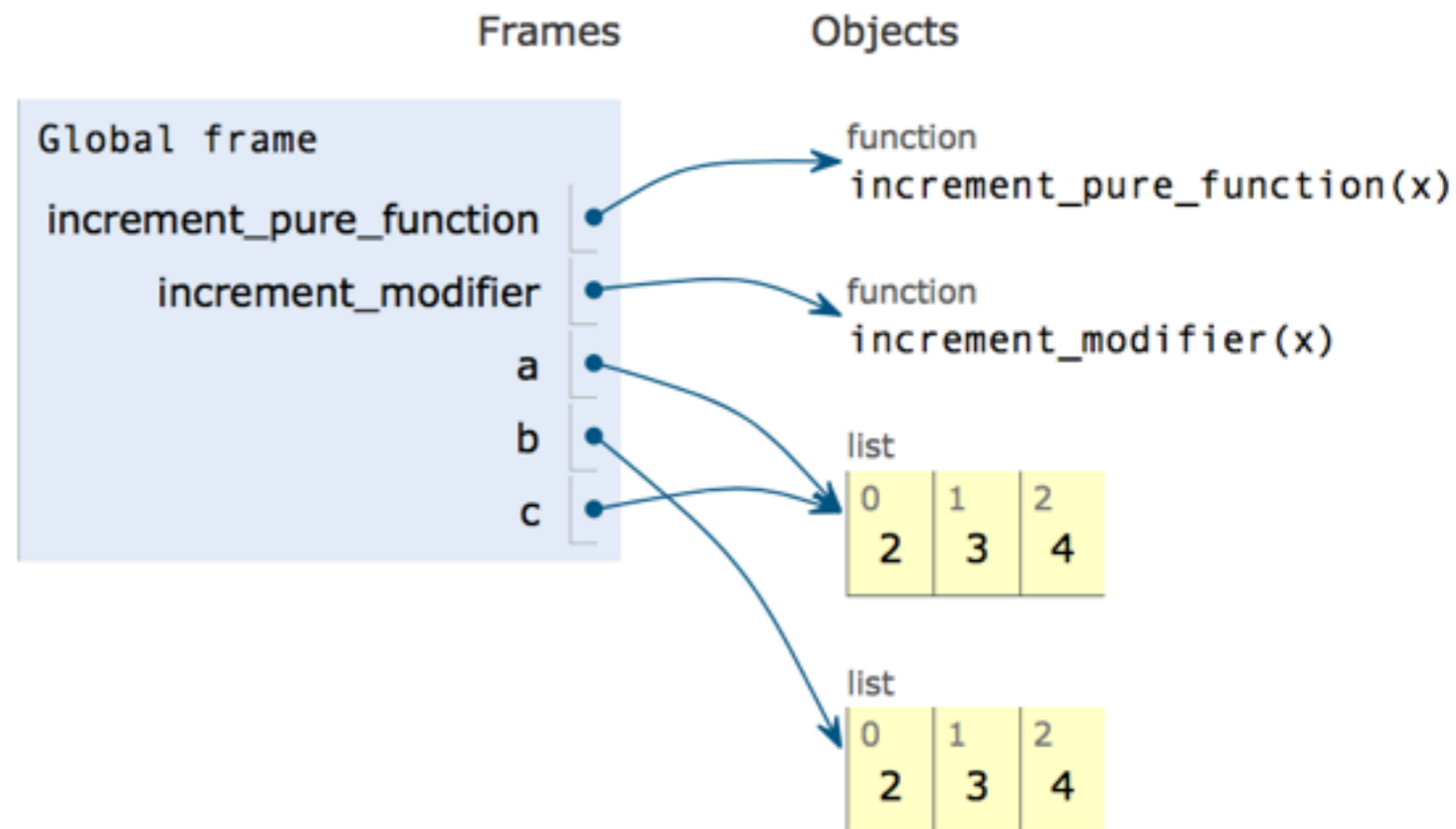
Write code in Python 3.3

(drag lower right corner to resize code editor)

```
1 def increment_pure_function(x):
2     v = []
3     for item in x:
4         v.append(item+1)
5     return(v)
6
7 def increment_modifier(x):
8     for i in range(len(x)):
9         x[i] = x[i]+1
10    return(x)
11
12 a = [1,2,3]
13 b = increment_pure_function(a)
14 print(a,',',b)
15 c = increment_modifier(a)
16 print(a,',',b,',',c)
```

Print output (drag lower right corner to resize)

```
[1, 2, 3] , [2, 3, 4]
[2, 3, 4] , [2, 3, 4] , [2, 3, 4]
```



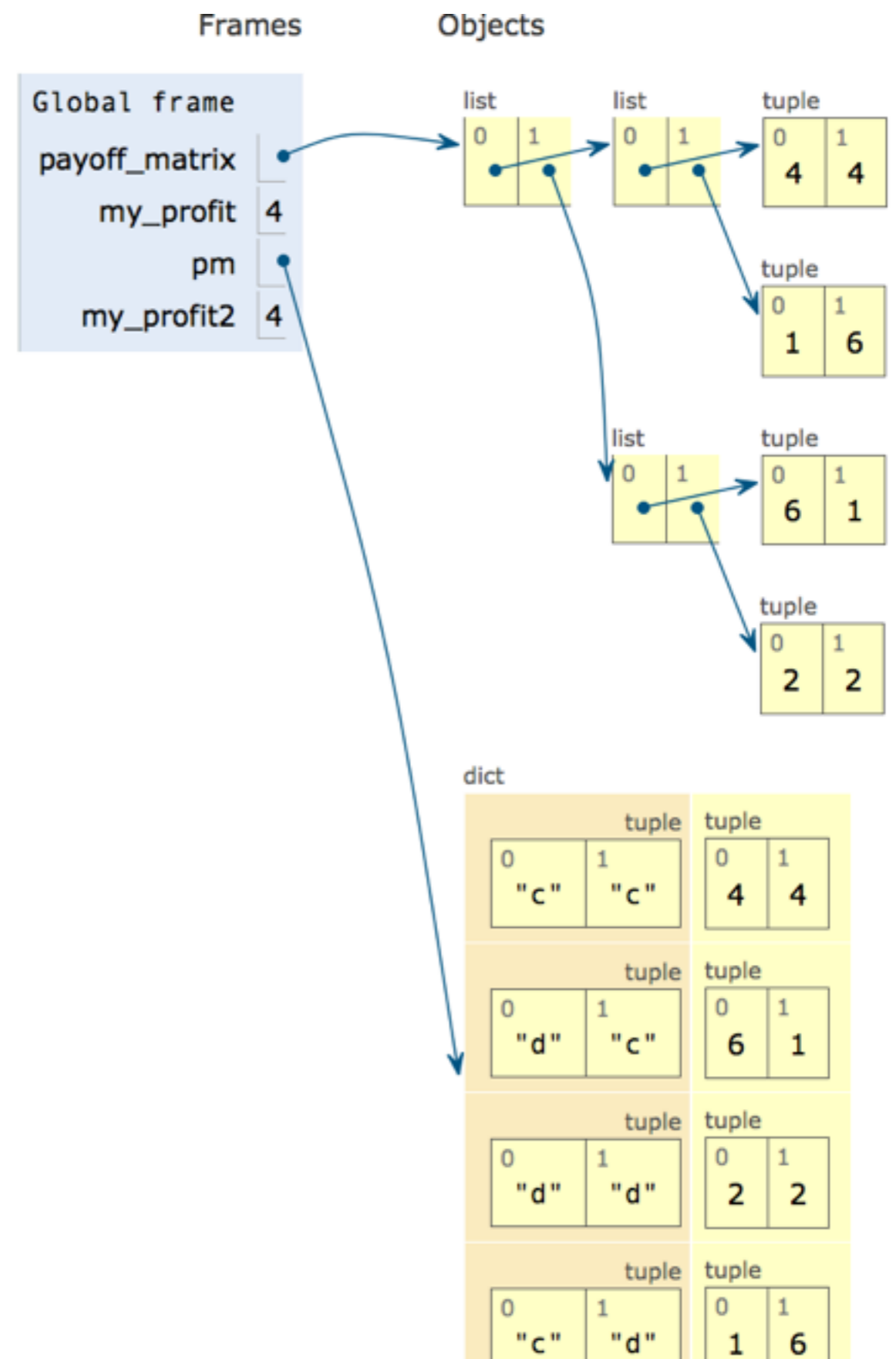
dict - tuples as keys

Write code in Python 3.3 (drag lower right corner to resize code editor)

```
1 payoff_matrix = [ [(4,4),(1,6)] , [(6,1),(2,2)] ]
2 # cooperate, cooperate, mine
3 my_profit = payoff_matrix[0][0][0]
4
5 pm = {}
6 pm['c', 'c'] = (4,4)
7 pm['d', 'd'] = (2,2)
8 pm['c', 'd'] = (1,6)
9 pm['d', 'c'] = (6,1)
10 my_profit2 = pm['c', 'c'][0]
11 |
```

→ line that has just executed
→ next line to execute

<< First < Back Done running (8 steps) Forward > Last >>



dictionary loops ...

```
1 pm = {}
2 pm['c', 'c'] = (4, 4)
3 pm['d', 'd'] = (2, 2)
4 pm['c', 'd'] = (1, 6)
5 pm['d', 'c'] = (6, 1)
6
7 for key in pm:
8     print(key, pm[key])
9
10 for key, value in pm.items():
11     print(key, value)
```

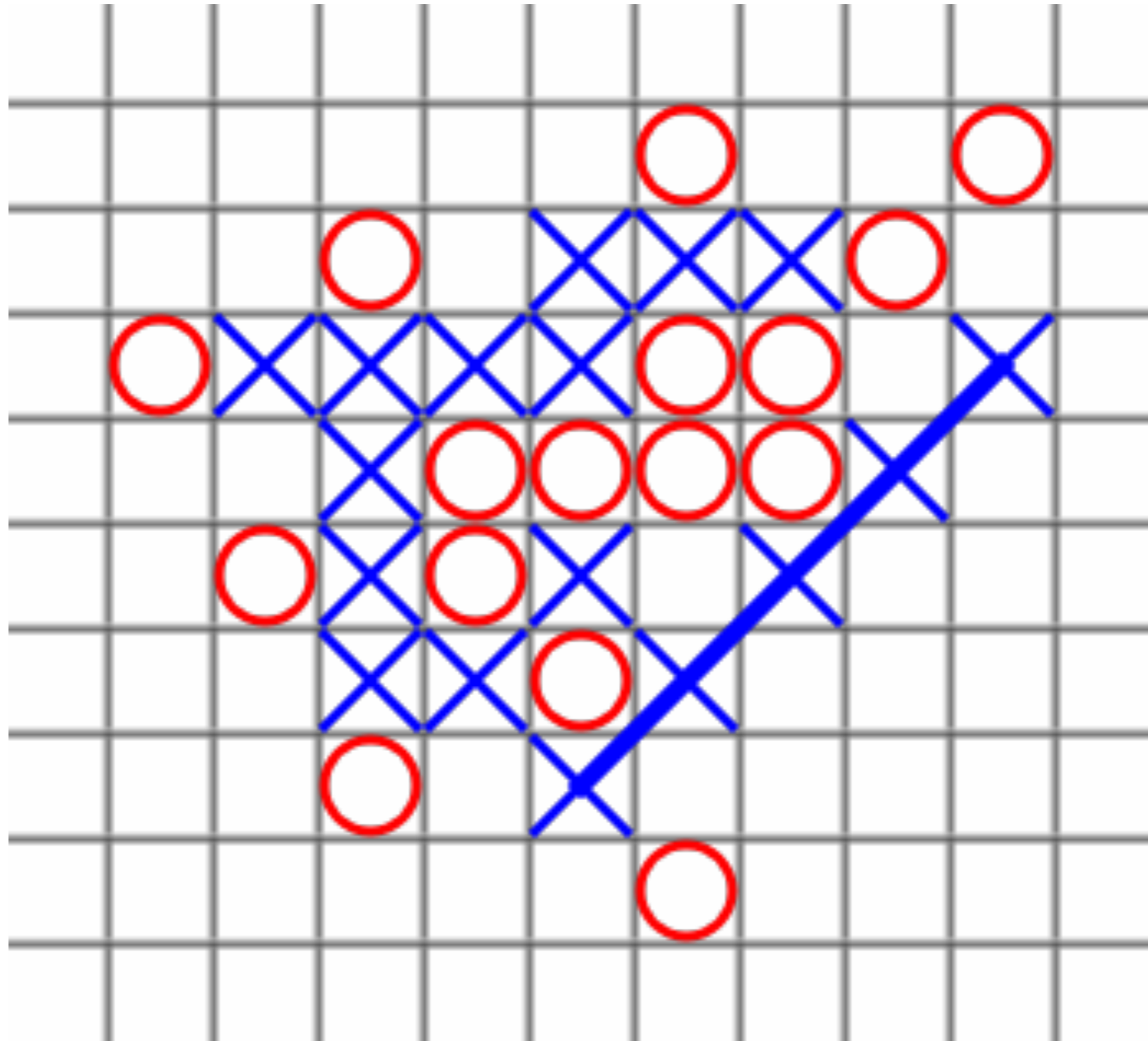
skládání objektů, dědění

- vylepšíme trochu hráče R-P-S
- ukážeme si na příkladu hráče piškvorek (tic-tac-toe)
- live-coding-session

vylepšení hráče R-P-S

- Společné do základní třídy BasePlayer
- Rozdílné strategie jako různí hráči
- Oddělení (zapoždření) technikalit správy hráčovy paměti
- A také vylepšíme hru na typ 2 za 3
- a ukážeme možnost jak řídit ukecanost běhu hry

Piškvorky, tic-tac-toe, m,n,k-game



skládání

```
13
14 class MyPlayer:
15     def __init__(self, mine_sym, opponent_sym, empty_sym):
16         """
17         :param mine_sym: string my symbol
18         :param opponent_sym: string opponent symbol
19         :param empty_sym: string empty symbol
20         :param win_length: int number of own symbols in a row
21         :return:
22         """
23         self.m = mine_sym
24         self.o = opponent_sym
25         self.empty = empty_sym
26         self.pf = playfield.PlayField(empty_sym=self.empty)
27
28     def play(self, field):...
36
37     def find_best_move(self, moves):...
40
```


dědění

```
14 class MyPlayer:
15     def __init__(self, mine_sym, opponent_sym, empty_sym):...
27
28     def play(self, field):...
36
37     def find_best_move(self, moves):...
40
41
42 class RandomPlayerSimple(MyPlayer):
43     """A simple implementation but not the fastest I'm afraid"""
44     def find_best_move(self, moves):
45         return random.choice(moves)
46
47
```

