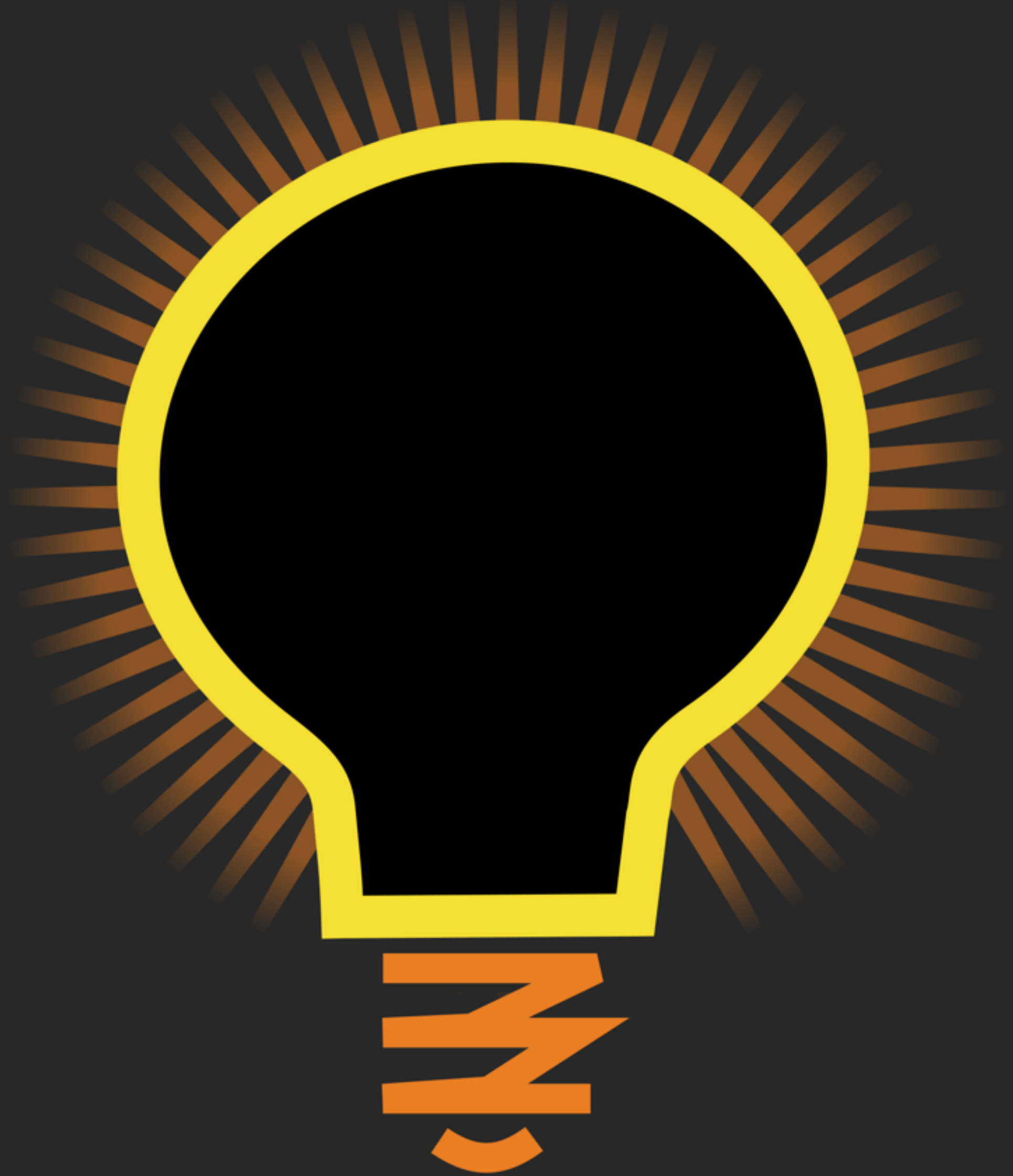


Python, základní kameny až skály I

Tomáš Svoboda

B4B33RPH, 2017-10-17



platnost proměnných

```
1 w = 'to se je zde, je viditelne vsude'
```

```
2 a = 'definovano v hlavnim programu'
```

```
3
```

```
4 def my_function(x):
```

```
5     a = 'definovano uvnitr funkce'
```

```
6     print('W je znamo: ',W)
```

```
7     return x + ' ' + a
```

```
8
```

```
9 # toto se vykona vzdy (import nebo behu programu)
```

```
10 print(a)
```

```
11
```

```
12 if __name__ == "__main__":
```

```
13     # toto se pri importu nevykona
```

```
14     a = 'neco jineho'
```

```
15     b = my_function(a)
```

visualizace

program structure - basic blocks

```
1 import math
```

imports

```
3 class MyClass:
```

```
4     '''class for '''
```

```
5     def __init__(self):
```

```
6         '''MyClass constructor'''
```

```
7         pass # nothing at the moment
```

```
8
```

```
9 def my_function(a,b):
```

```
10     '''compute sum a+b'''
```

```
11     pass # nothing at the moment
```

```
12
```

```
13 if __name__ == "__main__":
```

```
14     # actual program starts here
```

```
15     c = MyClass() # don't forget the parantheses! I will show!
```

```
16
```

Definitions:
classes
functions

main program

V krátkých ukázkách budeme někdy ukazovat jen kód bez hlaviček a spol.

funkce vs. metoda

```
1 import math
2
3 class MyClass:
4     '''class for '''
5     def __init__(self):
6         '''MyClass constructor'''
7         pass # nothing at the moment
8     def my_class_method(self):
9         print('nothing to report')
10
11
12 def my_function(a,b):
13     '''compute sum a+b'''
14     pass # nothing at the moment
15
16 if __name__ == "__main__":
17     # actual program starts here
18     c = MyClass() # don't forget the parantheses! I will show!
19     c.my_class_method()
20
```

Two red arrows are present. The first arrow points from the left edge of the slide to the definition of the class method on line 8: `def my_class_method(self):`. The second arrow points from the left edge to the definition of the function on line 12: `def my_function(a,b):`.

není číslo jako číslo

```
1 a = 0.1
2 b = 0.3
3 c = 3*a
4 if (b==c):
5     print(b, 'and', c, 'are equal')
6 else:
7     print(b, 'and', c, 'are NOT equal')
```

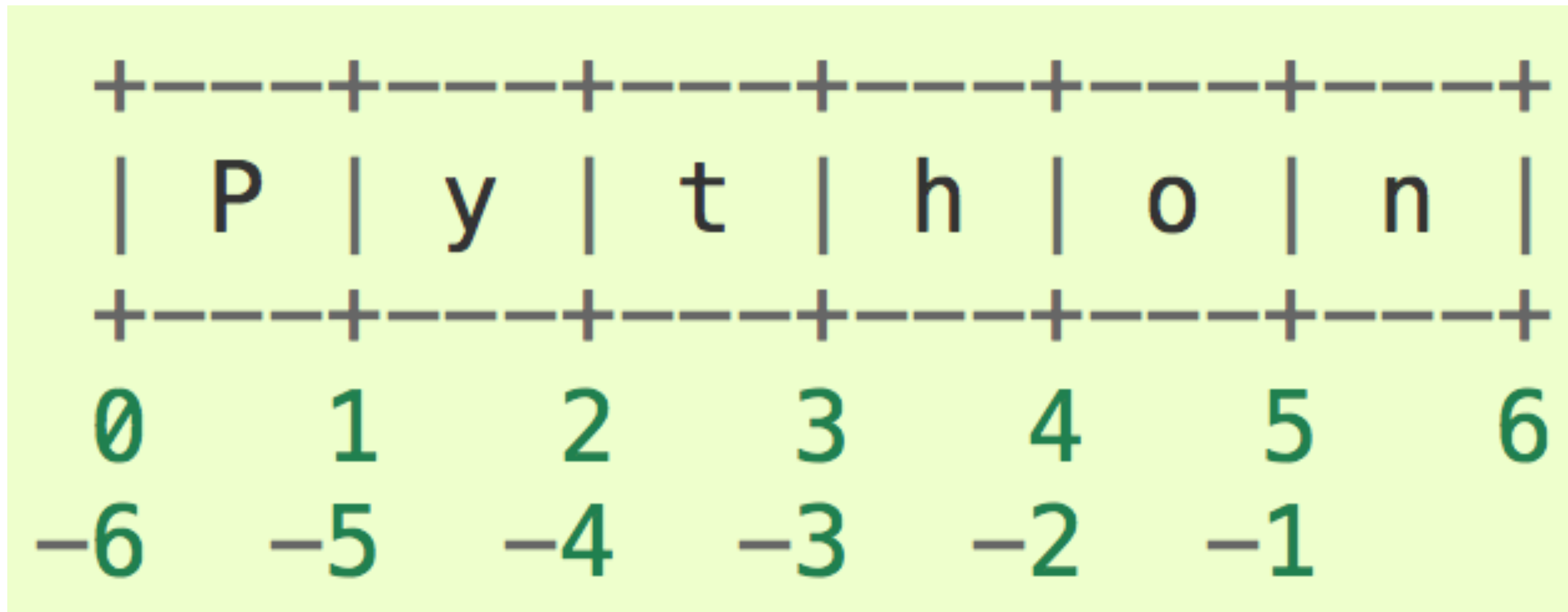
- vizualizace
- <https://docs.python.org/3/tutorial/floatingpoint.html>
- <http://floating-point-gui.de/formats/binary/>
- opatrnost při testování rovnosti (float) čísel
- pokud oprvadu potřeba: `abs(a-b) < threshold`

řetězce neboli stringy

```
1 a = 'ahoj'
2 b = 'svete'
3 c = a+b
4 for i,item in enumerate(c):
5     print(i, '-', item)
6 banner = ['ahoj', 'svete']
7 for i,item in enumerate(banner):
8     print(i, '-', item)
9 for i,item in enumerate(banner):
10     for j,elem in enumerate(item):
11         print(i, '-', item, '***', j, ':', elem)
```

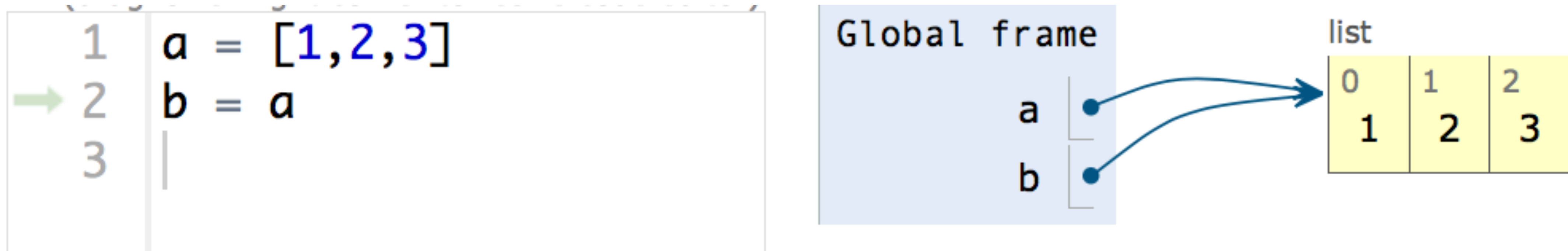
visualizace

python indexing, slicing, ...



dokončíme teď R-P-S
třída Game

pointers



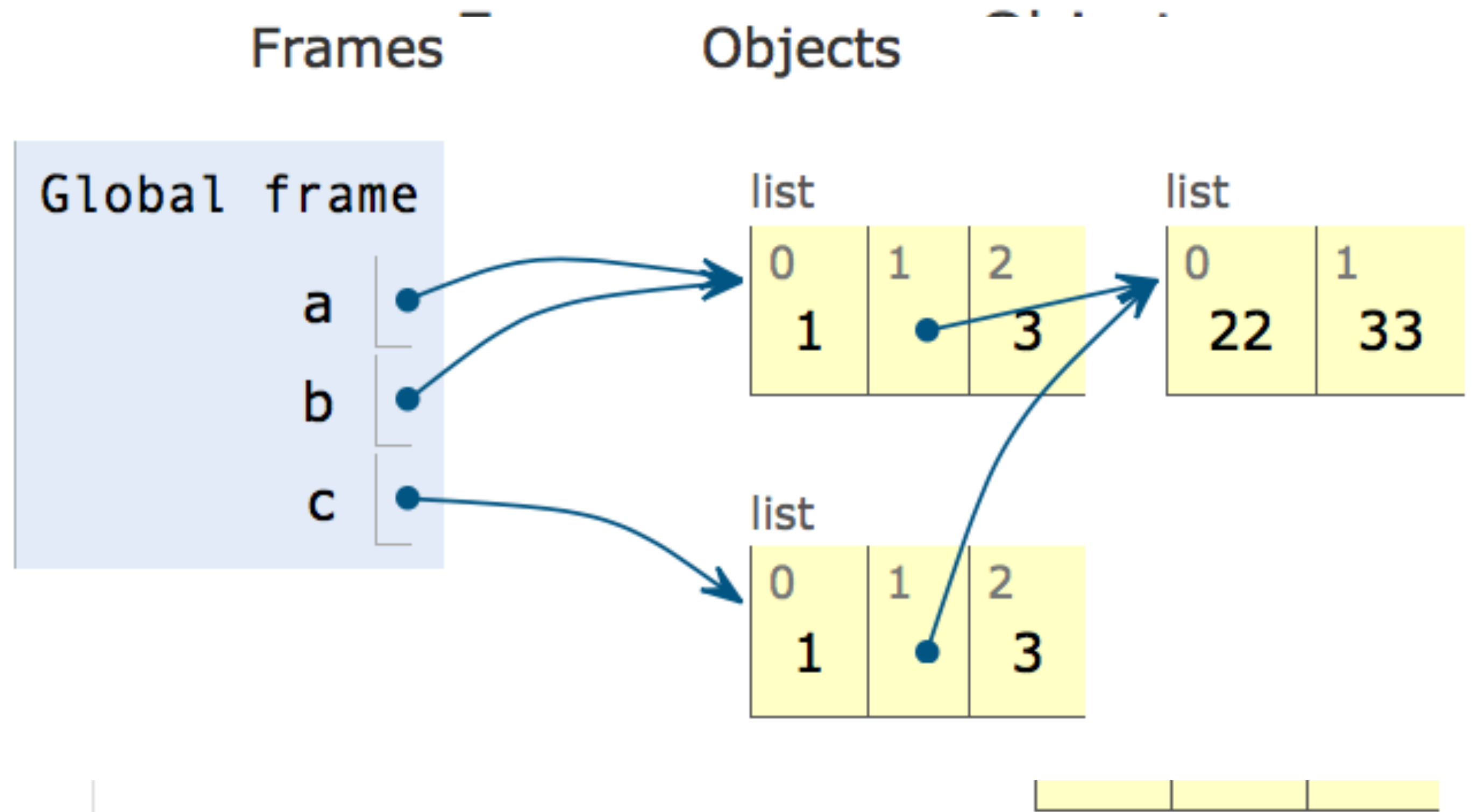
visualisation

is vs ==

making copy, a[:], rychle, ale ...

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

```
1 a = [1, [22, 33], 3]
2 b = a
3 c = a[:]
4
5
```



import copy and go deep

<http://docs.python.org/3.4/library/copy.html>

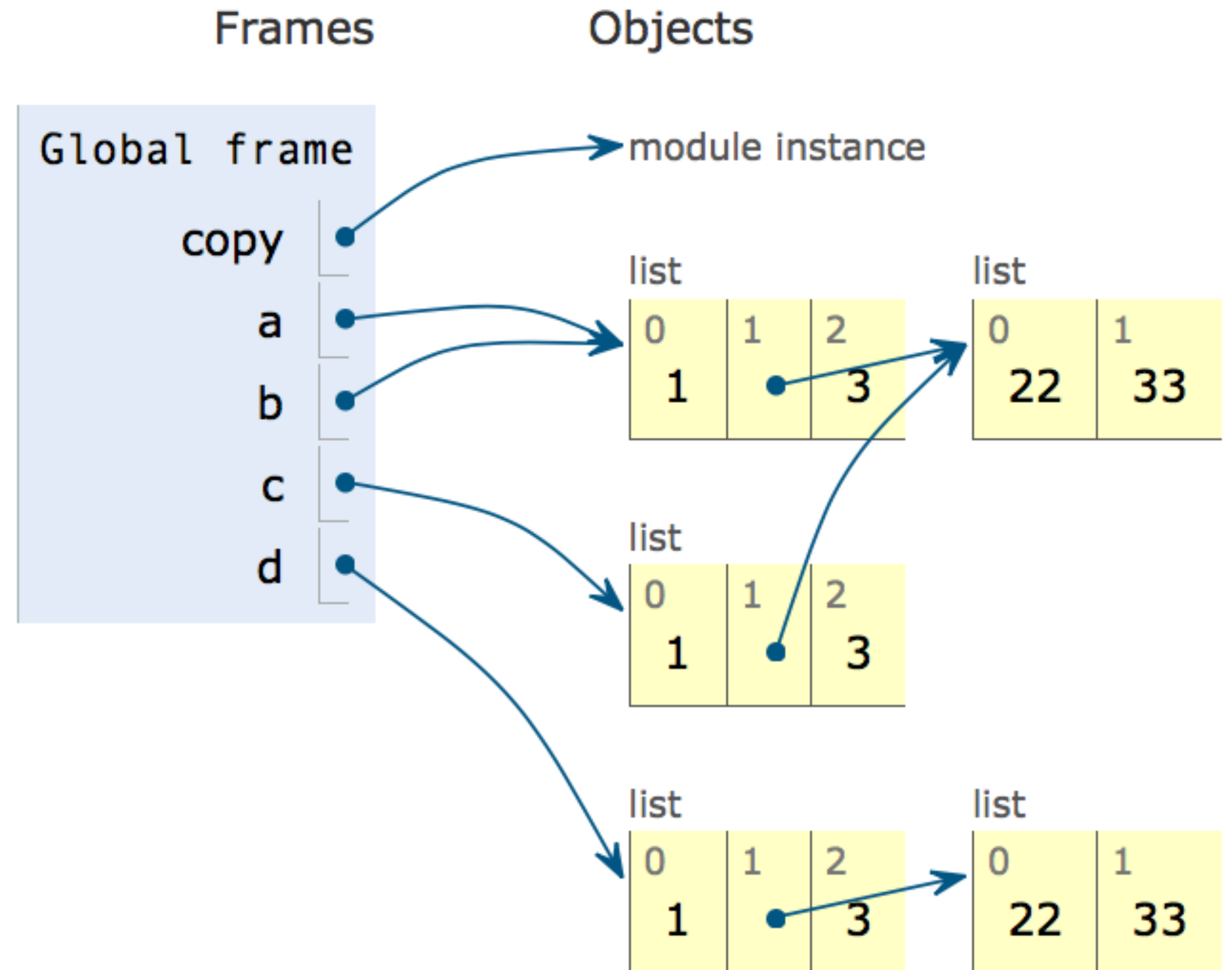
Write code in Python 3.3

(drag lower right corner to resize code editor)

```
1 import copy
2 a = [1, [22, 33], 3]
3 b = a
4 c = a[:]
5 d = copy.deepcopy(a)
6
7
```

→ line that has just executed

→ next line to execute

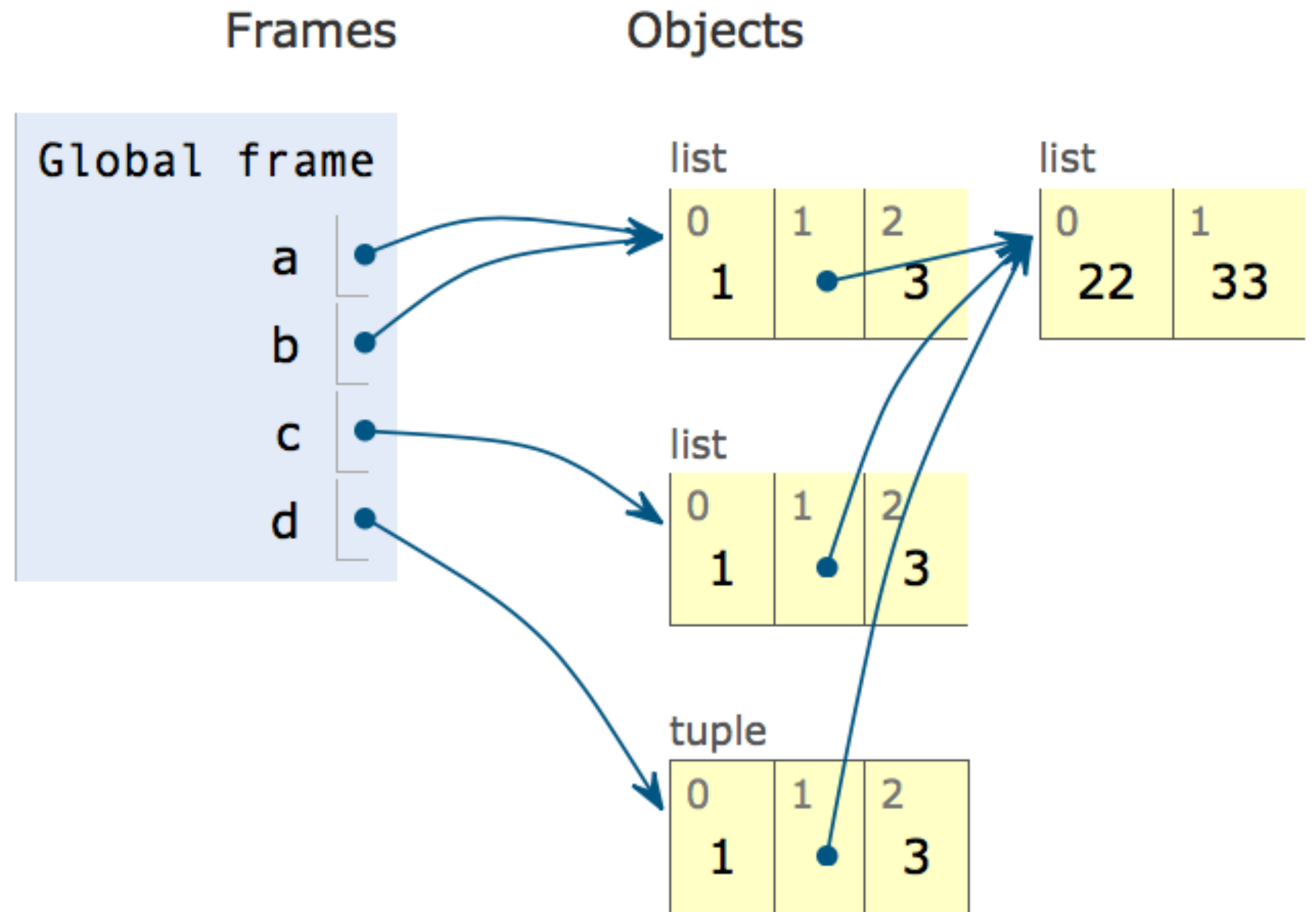


pozor na mělkost kopií

Write code in Python 3.3

(drag lower right corner to resize code editor)

```
1 a = [1, [22, 33], 3]
2 b = a
3 c = list(b)
4 d = tuple(a)
5
```



funkce pravé a modifikátory

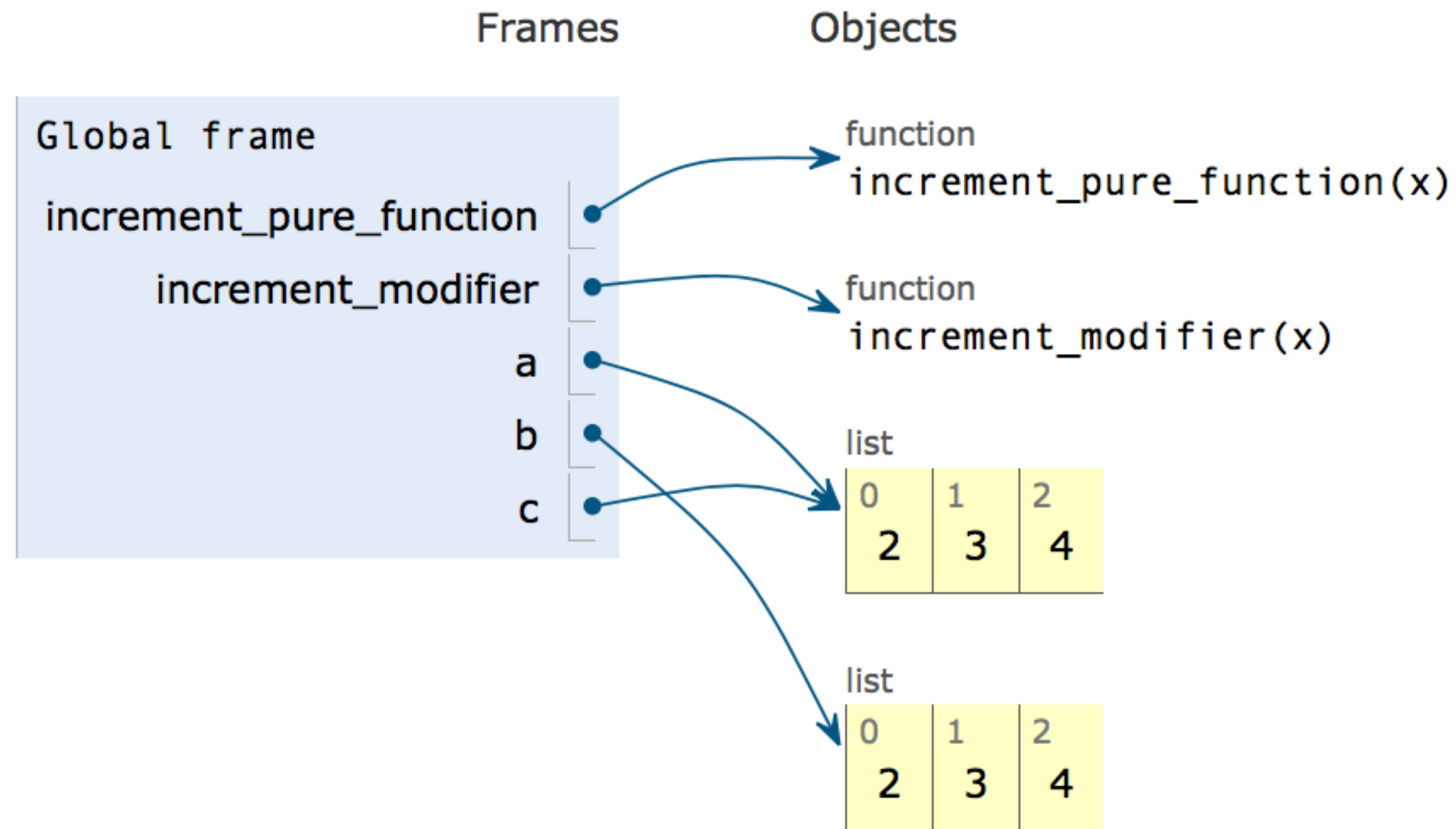
Write code in Python 3.3

(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]
```

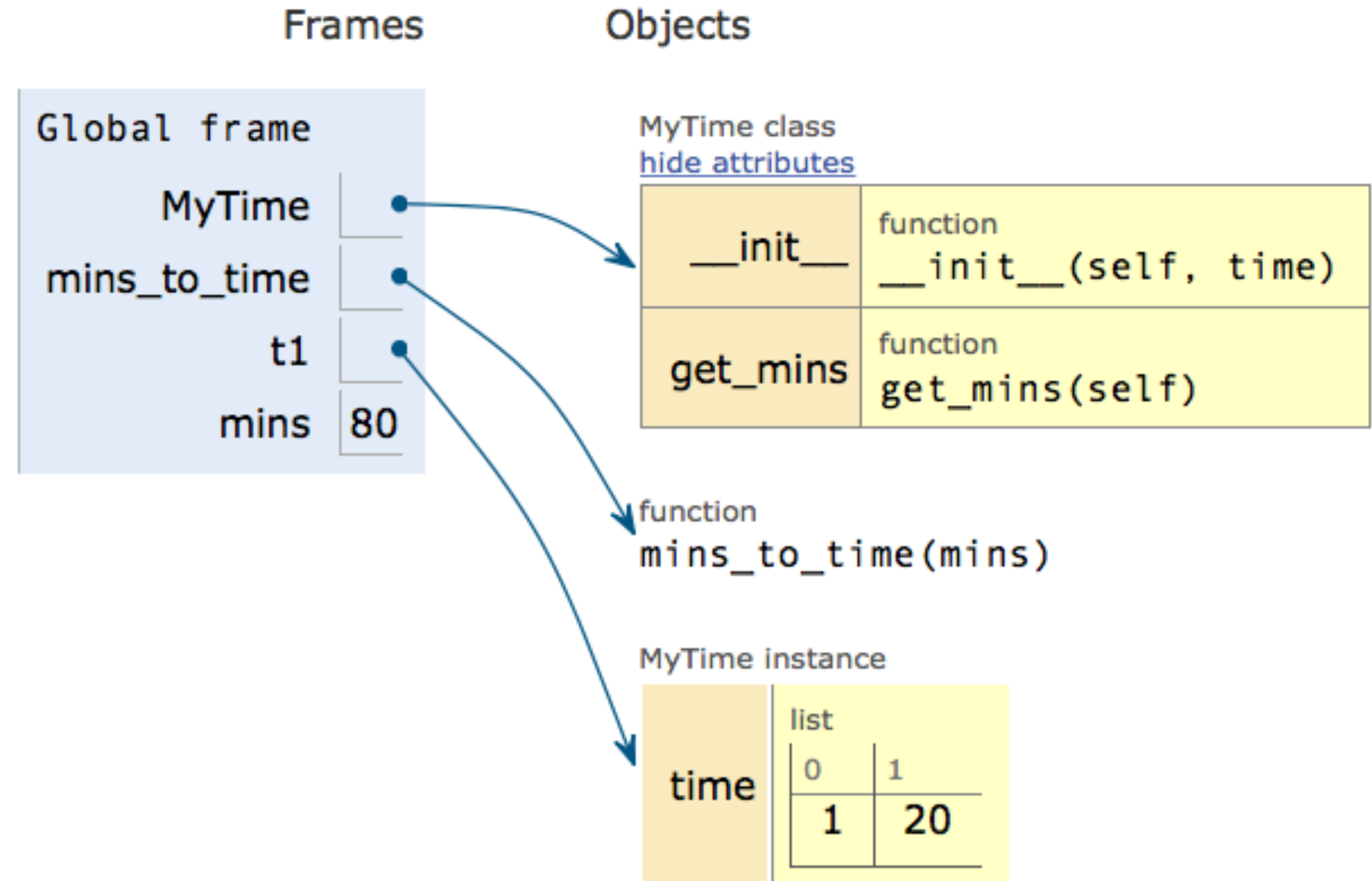


objekty, třídy a tak

Write code in Python 3.3

(drag lower right corner to resize code editor)

```
1 class MyTime:
2     def __init__(self, time=None):
3         self.time = time
4
5     def get_mins(self):
6         return(self.time[0]*60+self.time[1])
7
8 def mins_to_time(mins):
9     return([mins//60, mins%60])
10
11 t1 = MyTime([1,20])
12 mins = t1.get_mins()
13 time_vec = mins_to_time(mins)
```



visualisation

ale pozor ...

Write code in Python 3.3

(drag lower right corner to resize code editor)

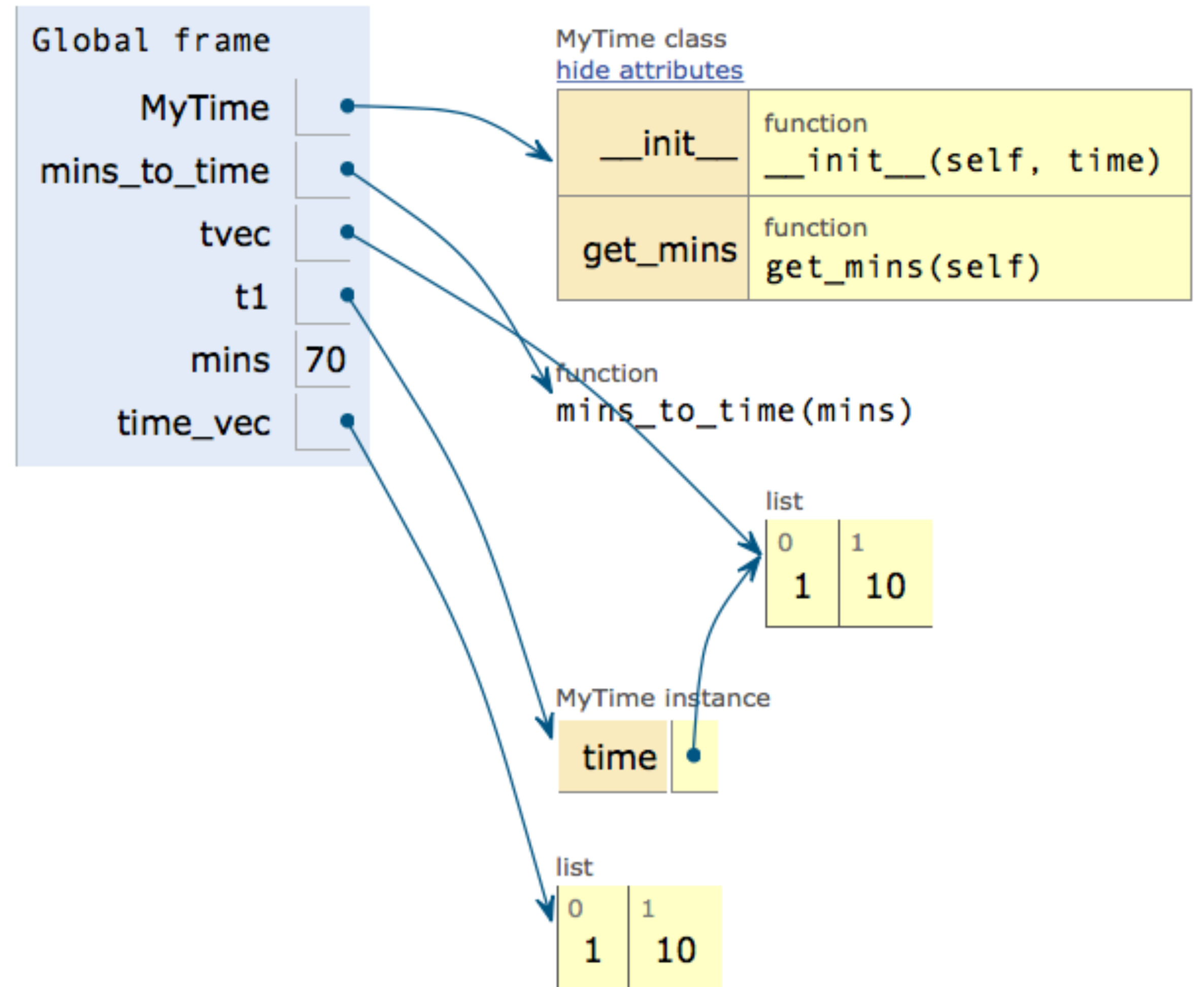
```
1 class MyTime:
2     def __init__(self, time=None):
3         self.time = time
4
5     def get_mins(self):
6         return(self.time[0]*60+self.time[1])
7
8 def mins_to_time(mins):
9     return([mins//60, mins%60])
10
11 tvec = [1,20]
12 t1 = MyTime(tvec)
13 tvec[1] = 10
14 mins = t1.get_mins()
```

→ line that has just executed

→ next line to execute

Frames

Objects



<< First

< Back

Done running (16 steps)

Forward >

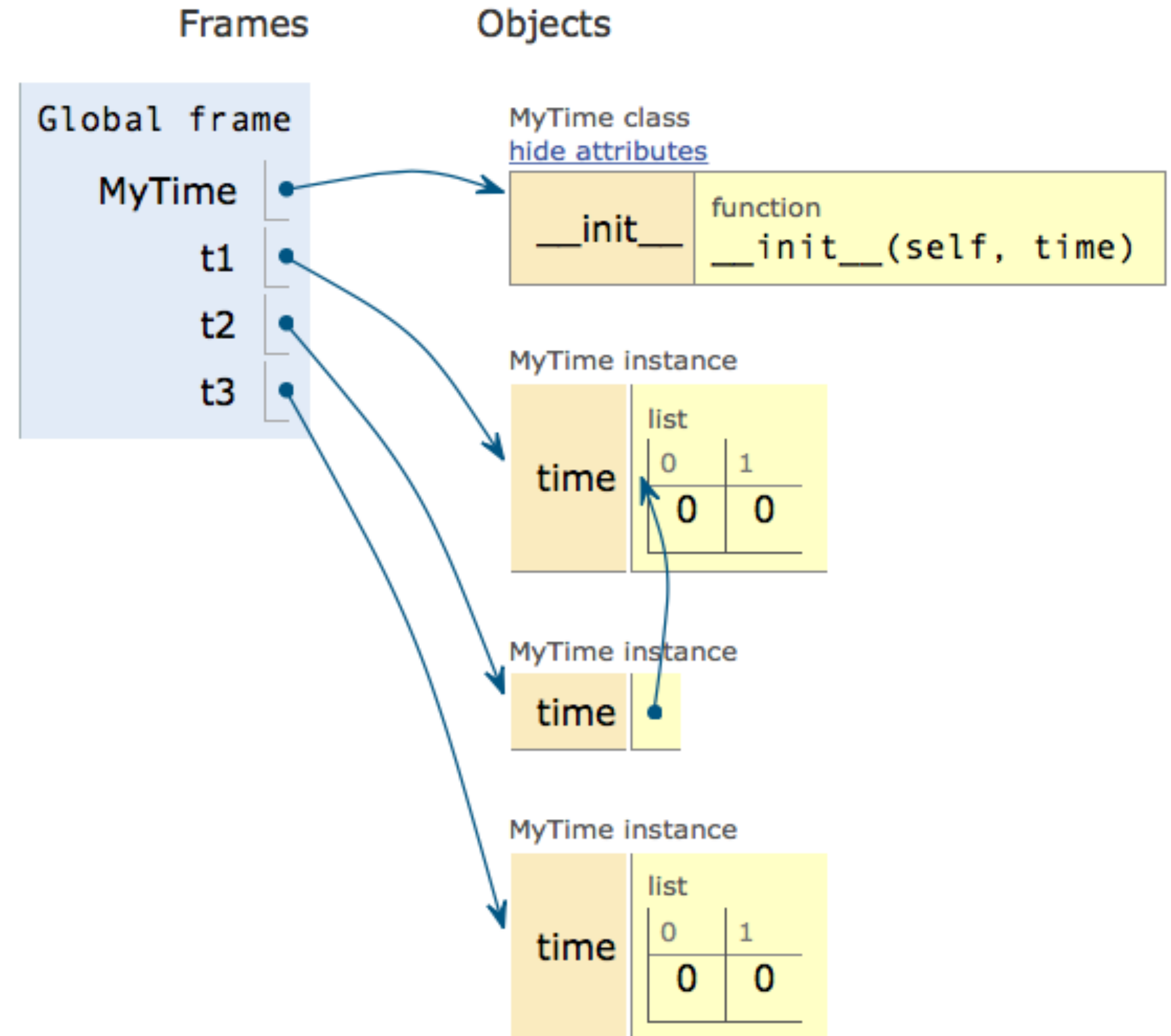
Last >>

a ještě větší pozor na implicitní parametry

Write code in Python 3.3

(drag lower right corner to resize code editor)

```
1 class MyTime:
2     def __init__(self, time=[0,0]):
3         self.time = time
4
5 t1 = MyTime()
6 t2 = MyTime()
7
8 t3 = MyTime([0,0])
9
```



implicitní parametry detailněji

Write code in Python 3.3

(drag lower right corner to resize code editor)

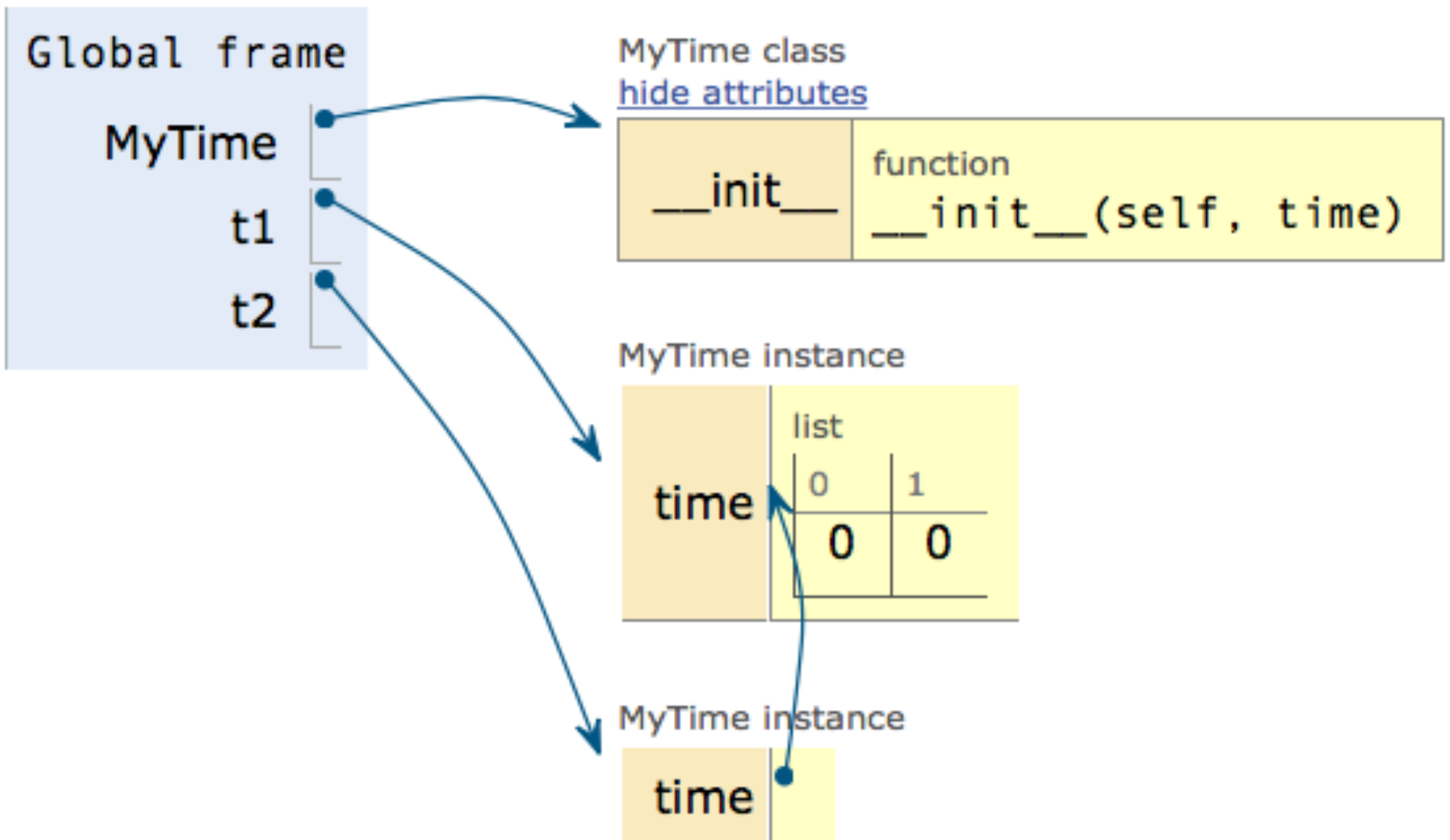
```
1 class MyTime:
2     def __init__(self, time=[0,0]):
3         self.time = time
4
5 t1 = MyTime()
6 t2 = MyTime()
7 print(id(t1.time))
8 print(id(t2.time))
9 print(t1.time is t2.time)
10
```

Print output (drag lower right corner to resize)

```
140145092713432
140145092713432
True
```

Frames

Objects



běžte a programujte!



- <http://pythontutor.com/visualize.html#mode=edit>
- <http://openbookproject.net/thinkcs/python/english3e/index.html>