

MORE FUNCTIONAL JAVA WITH JAVA SLANG

daniel.hinojosa



ABOUT ME

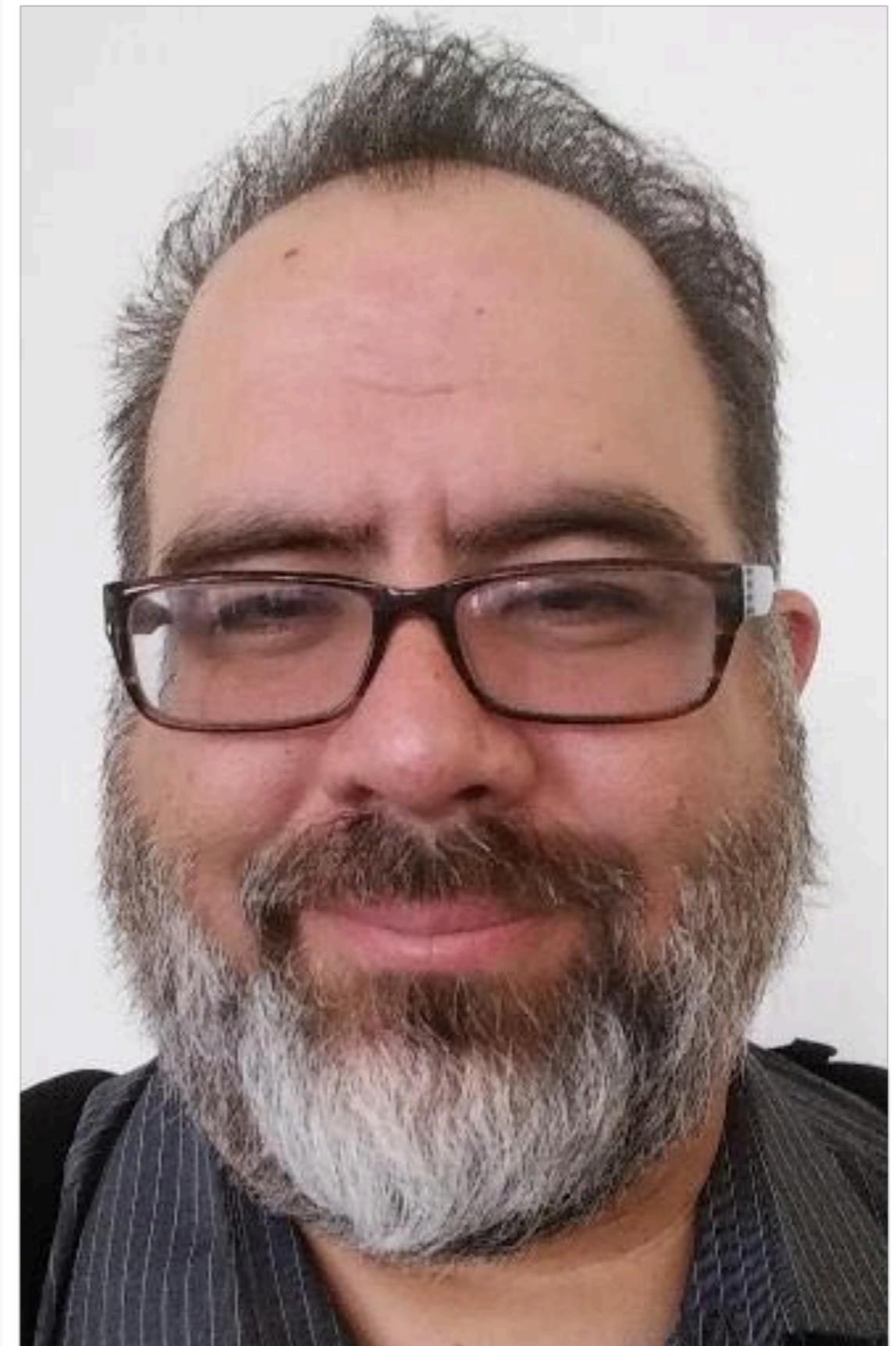
daniel.hinojosa

Independent Consultant,
Programmer. Trainer, Author

(Beard Optional)

 @dhinojosa

 dhinojosa@evolutionnext.com



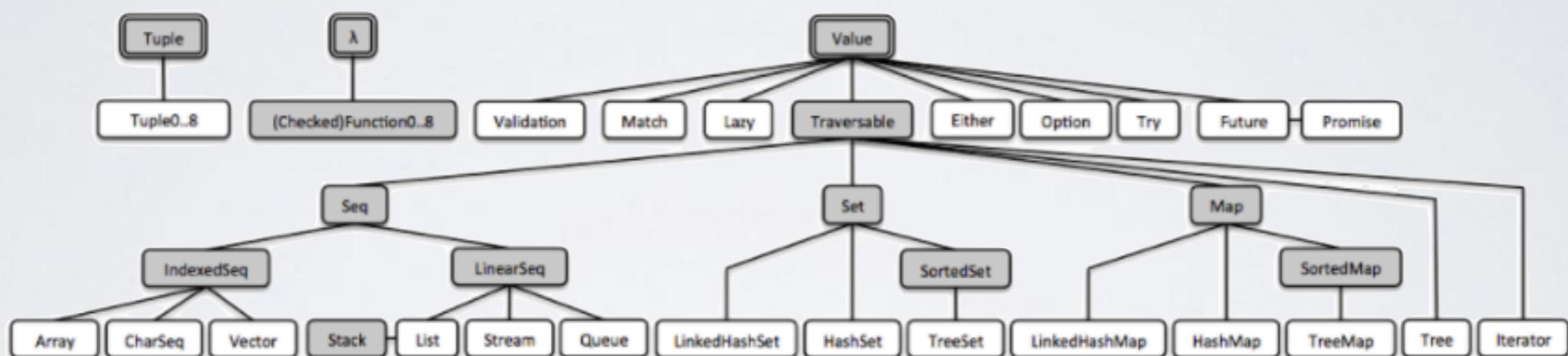
Code and Slides are available on Github



<https://github.com/dhinojosa/vavr-study>

<https://github.com/dhinojosa/javaslang-study>





BASIC COMPONENTS

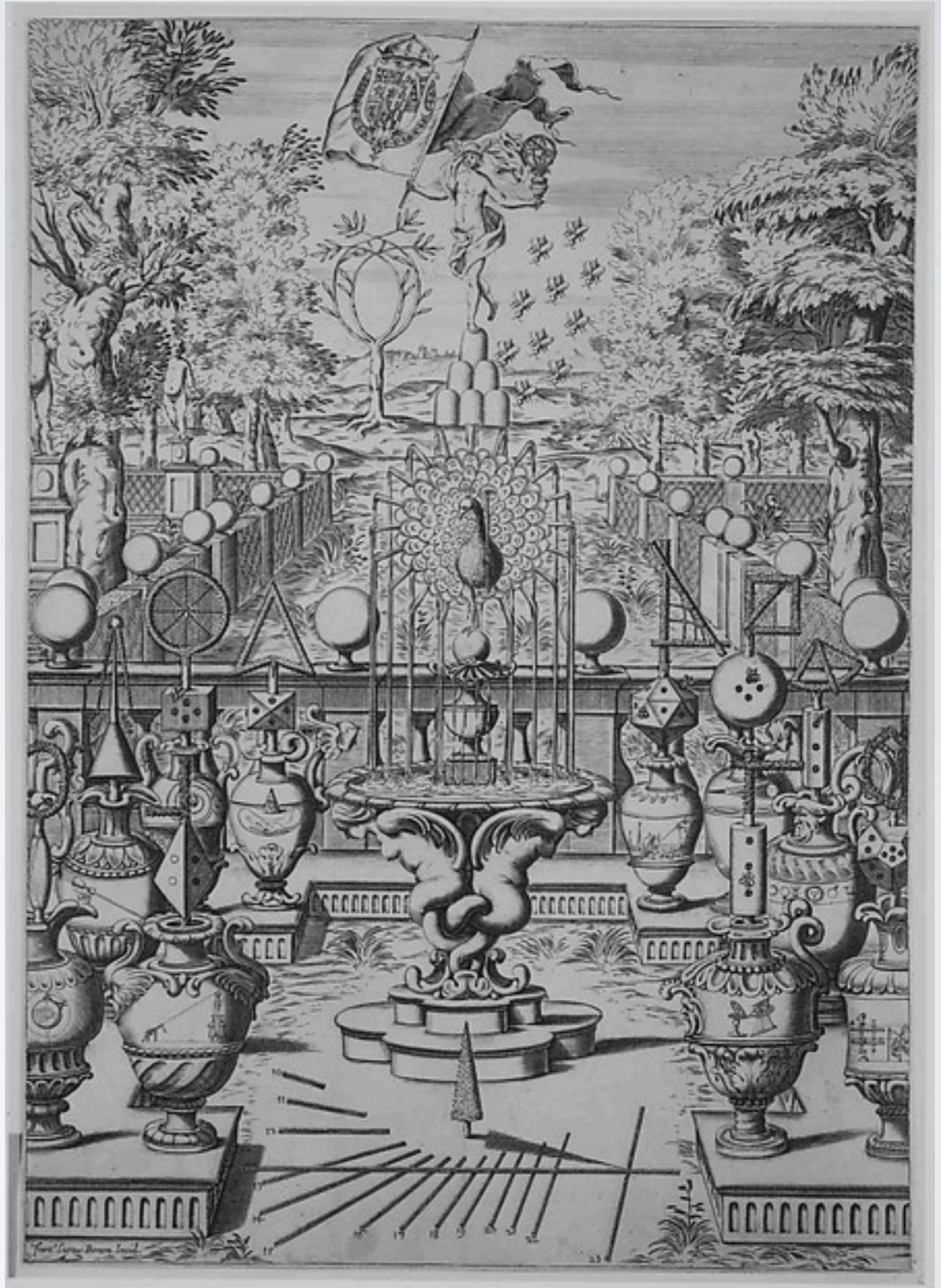




TUPLES



FUNCTIONS



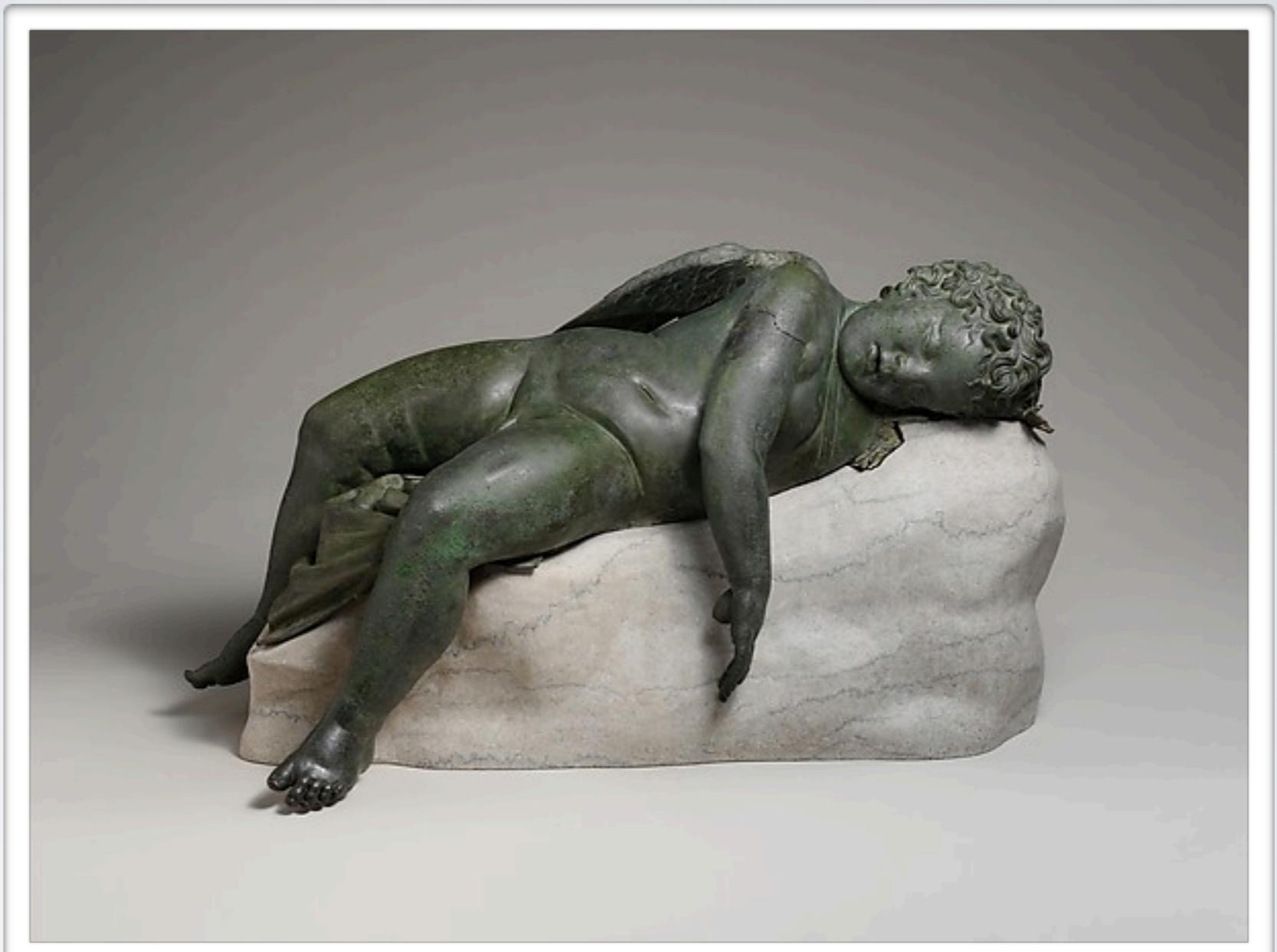
- Extended from λ (say what?)
- Instead of `Supplier<T>`, `Function<T, R>`, or `BiFunction<T1, T2, R>` it's `Function0`, `Function1`, `Function2`, ...
- Up to `Function8`

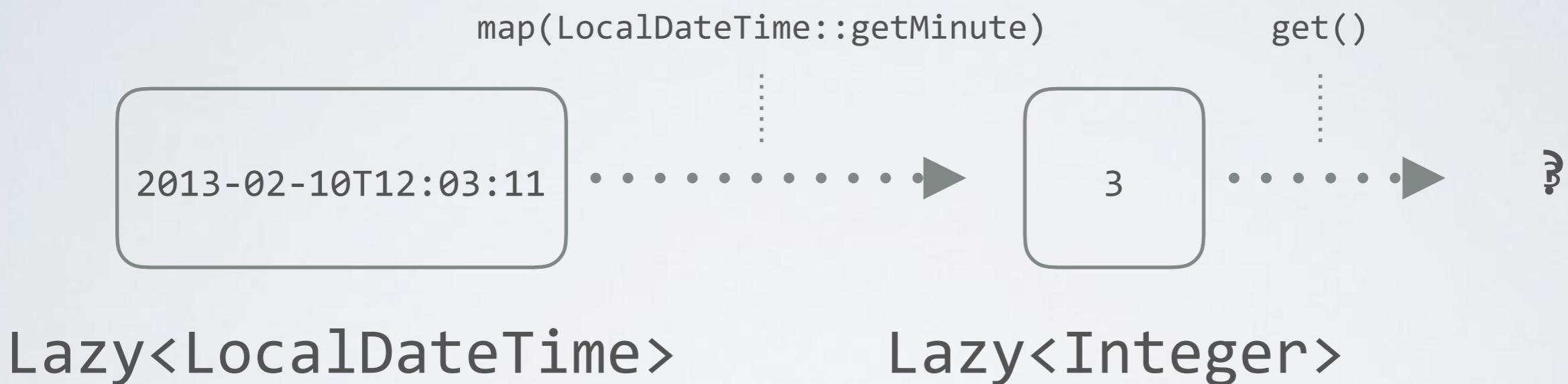
$$f(g(x))$$

TUPLES & FUNCTIONS

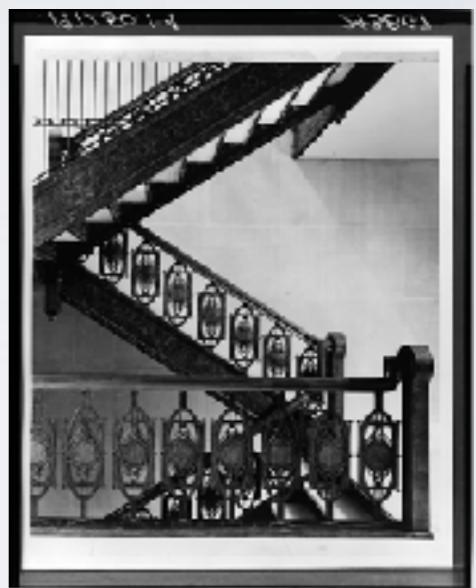


LAZY





PERSISTENT DATA STRUCTURES



IMMUTABILITY

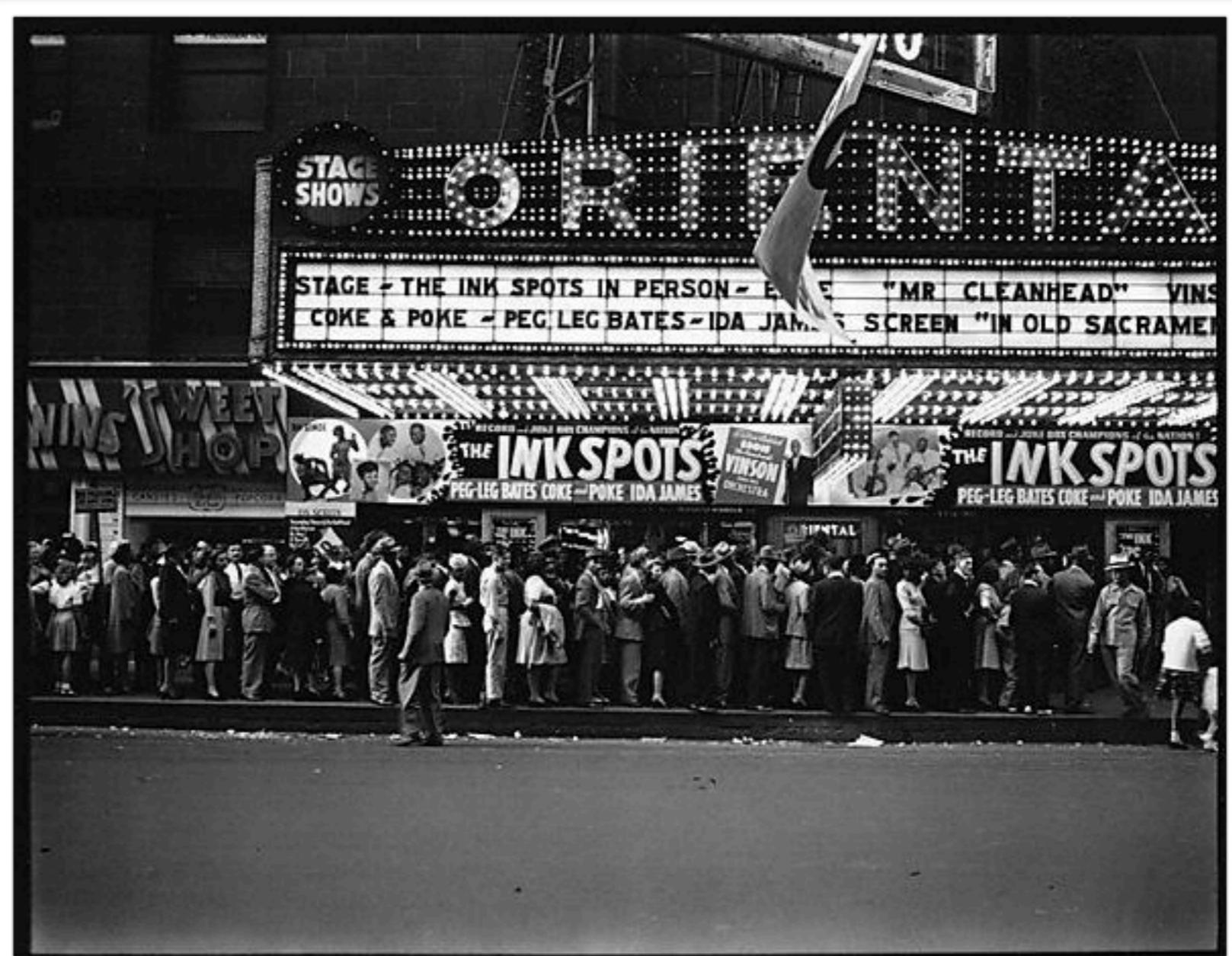
OPERATIONS RETURN COPIES

The integrity of the original stay intact

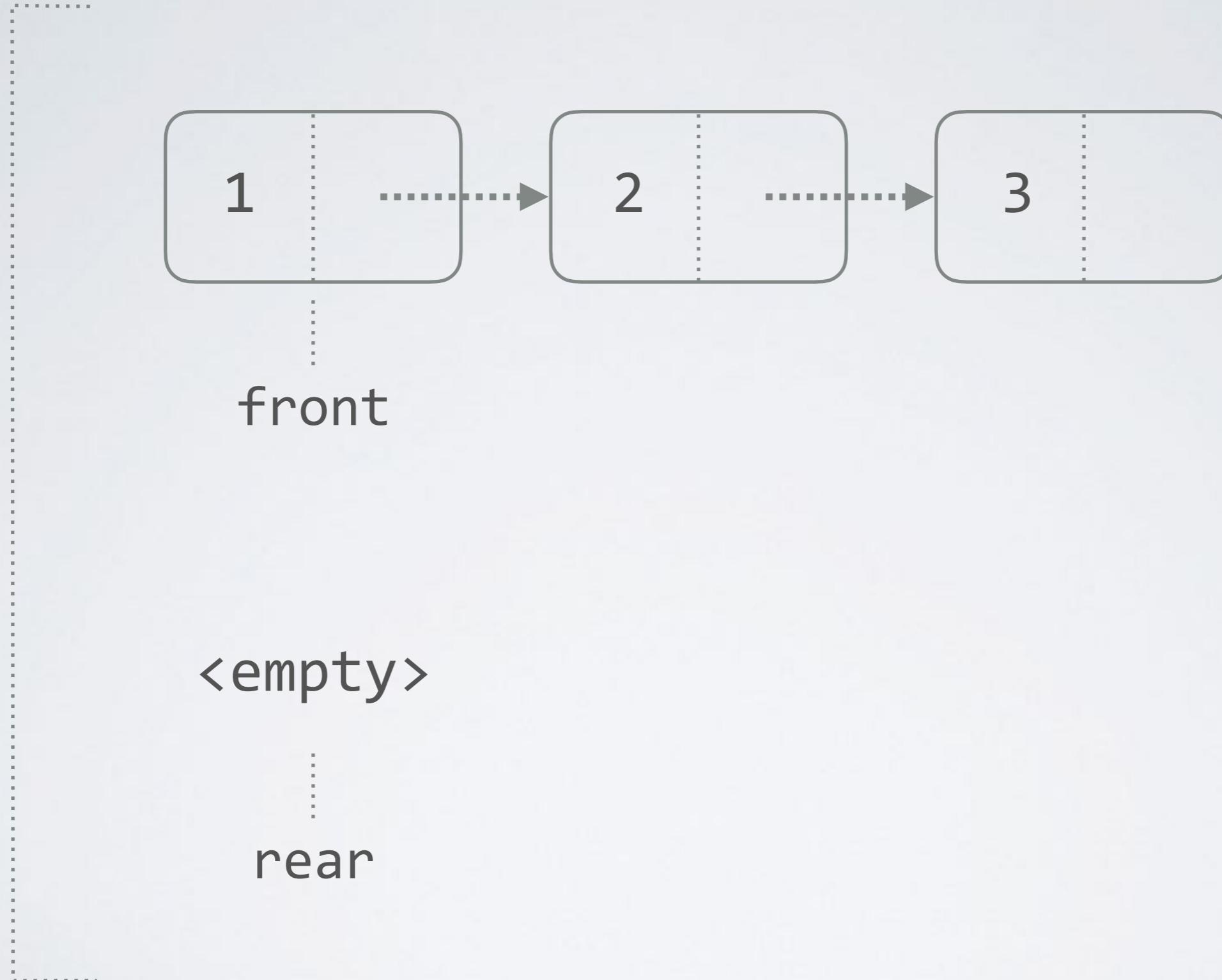
REFERENTIALLY TRANSPARENT

An expression is said to be referentially transparent if it can be replaced with its corresponding value without changing the program's behavior.

QUEUE



Queue.of(1,2,3)



`Queue.of(1,2,3).enqueue(4,5)`



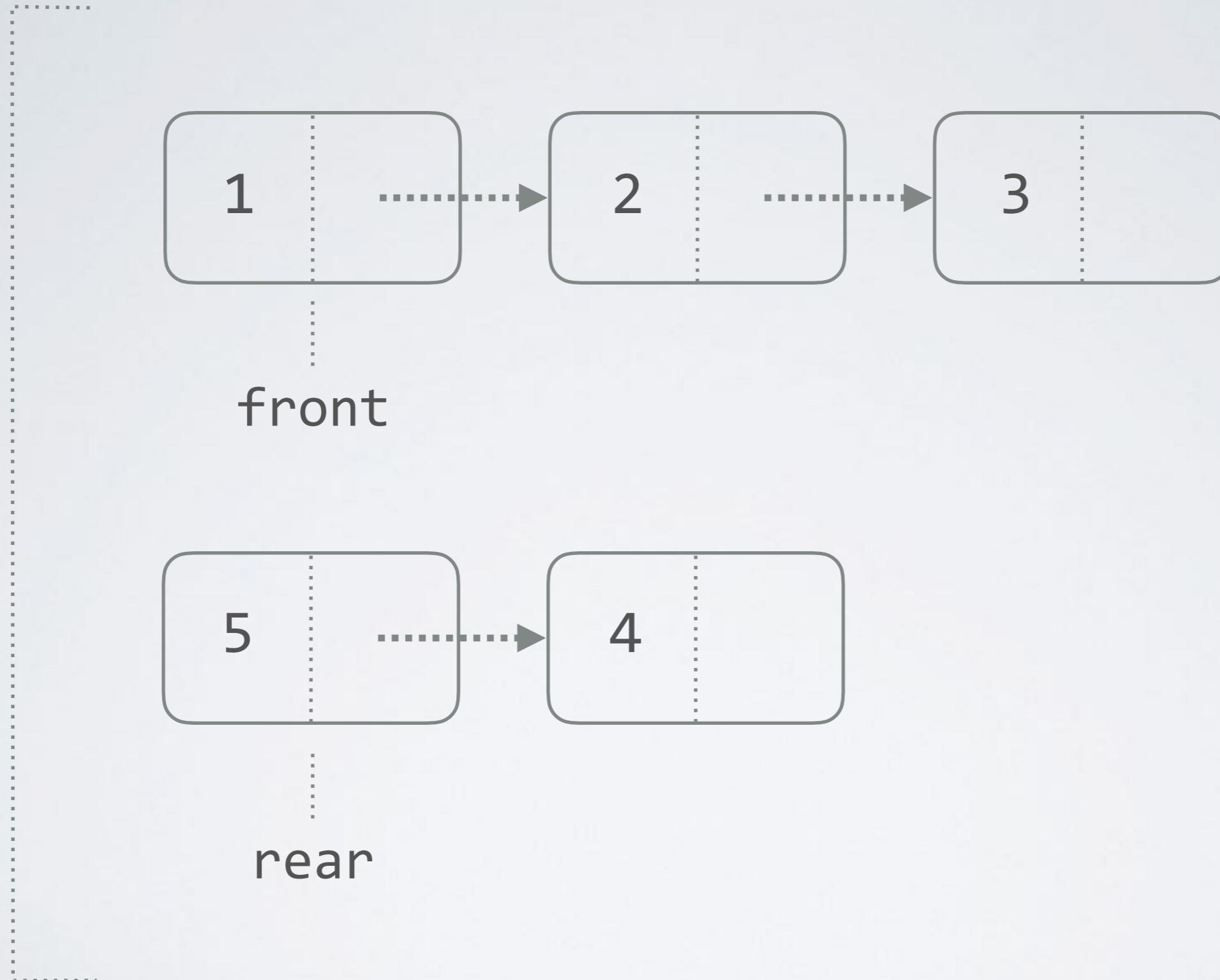
`front`



`temp` `rear`

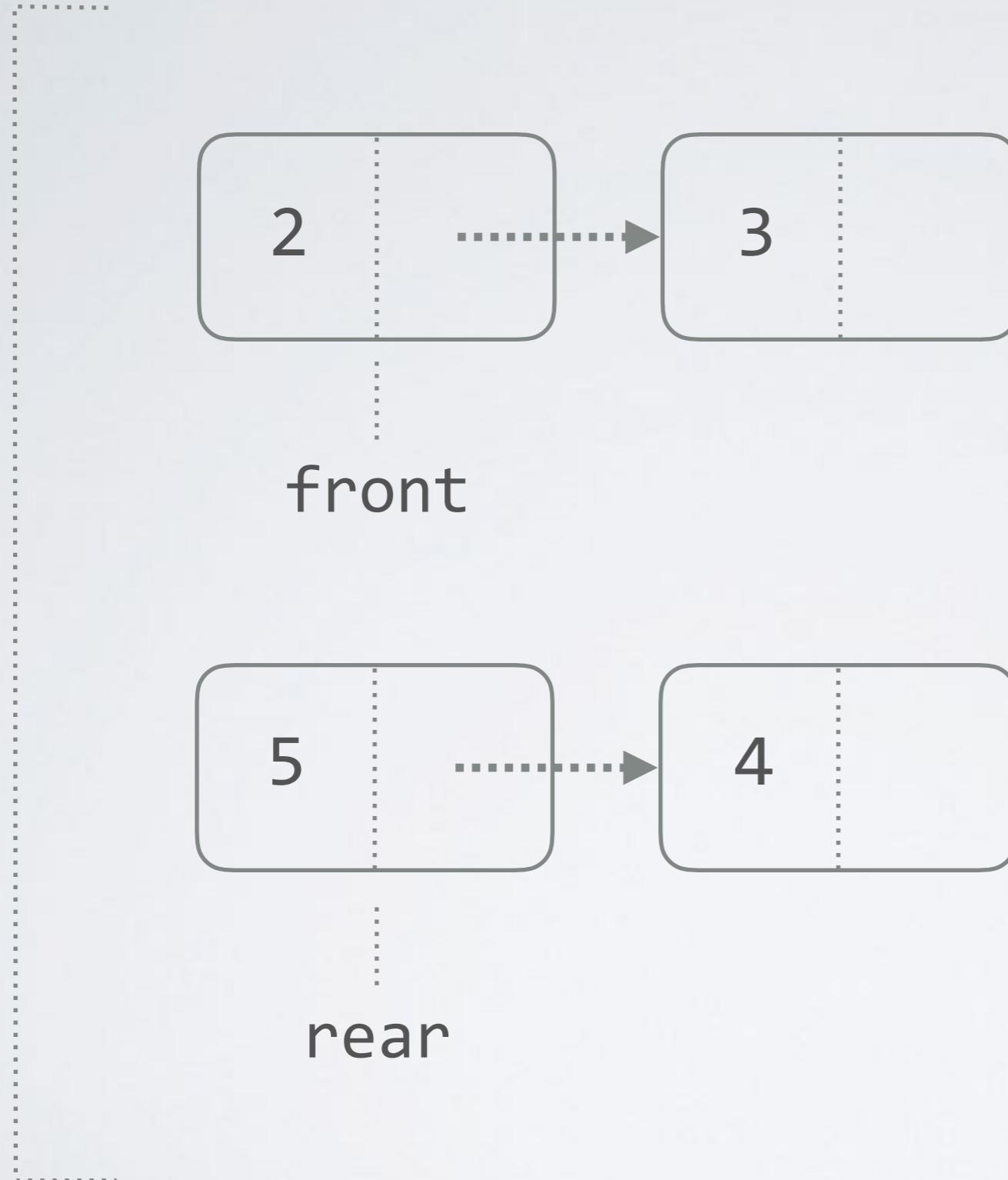
`Queue.of([1,2,3], [5,4])`

Queue.of(1,2,3,4,5).dequeue()



Tuple(1, Queue.of([2,3], [5,4]))

`Queue.of(2,3,4,5).dequeue()`

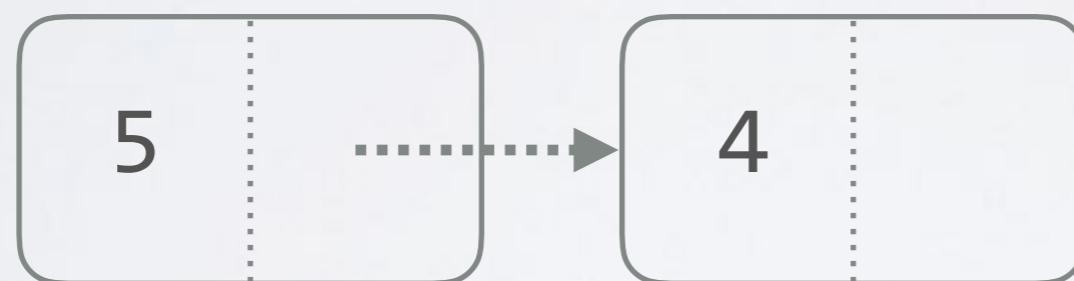


`Tuple(2, Queue.of([3], [5,4]))`

`Queue.of(3,4,5).dequeue()`



`front`



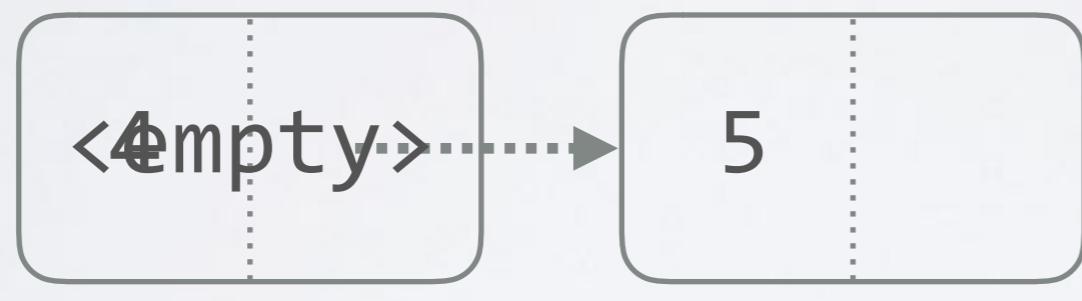
`rear`

`Tuple(3, Queue.of([], [5,4]))`

Queue.of(4,5).dequeue()

<empty>

front



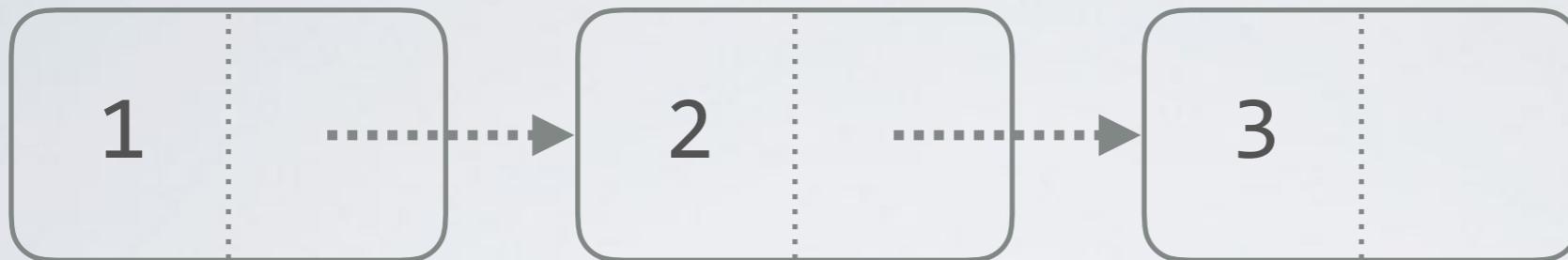
rear

Tuple(3, Queue.of([], [5,4]))

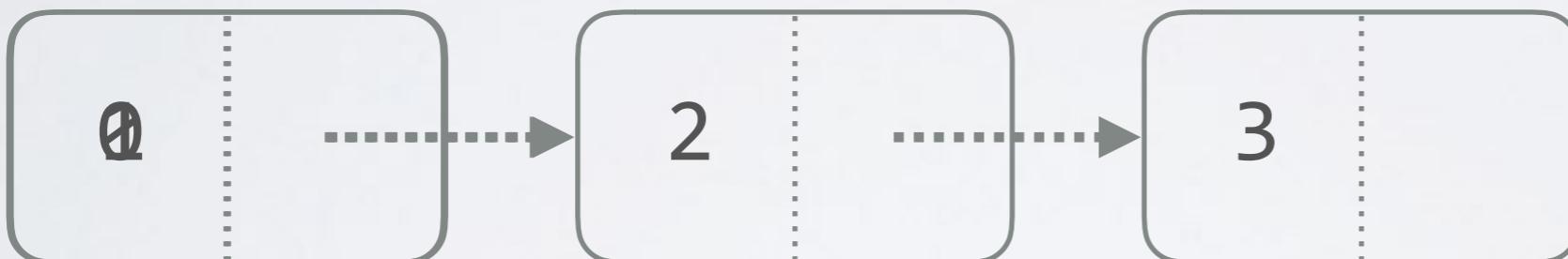
LINKED LISTS



```
list = List.of(1, 2, 3);
```



```
newList = list.prepend(0);
```

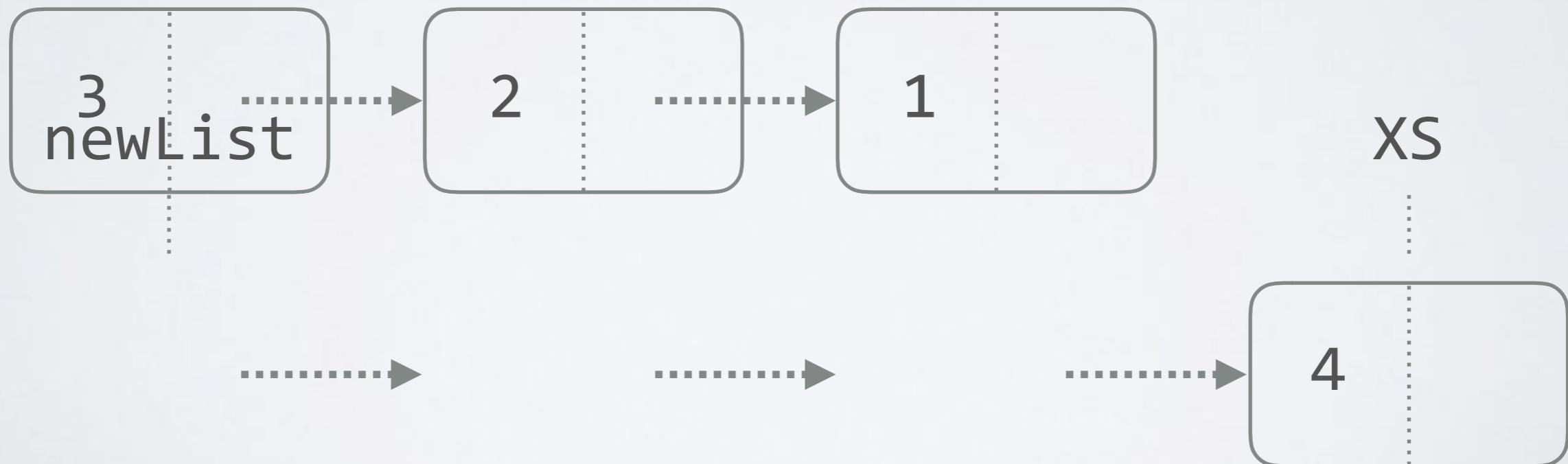


newList

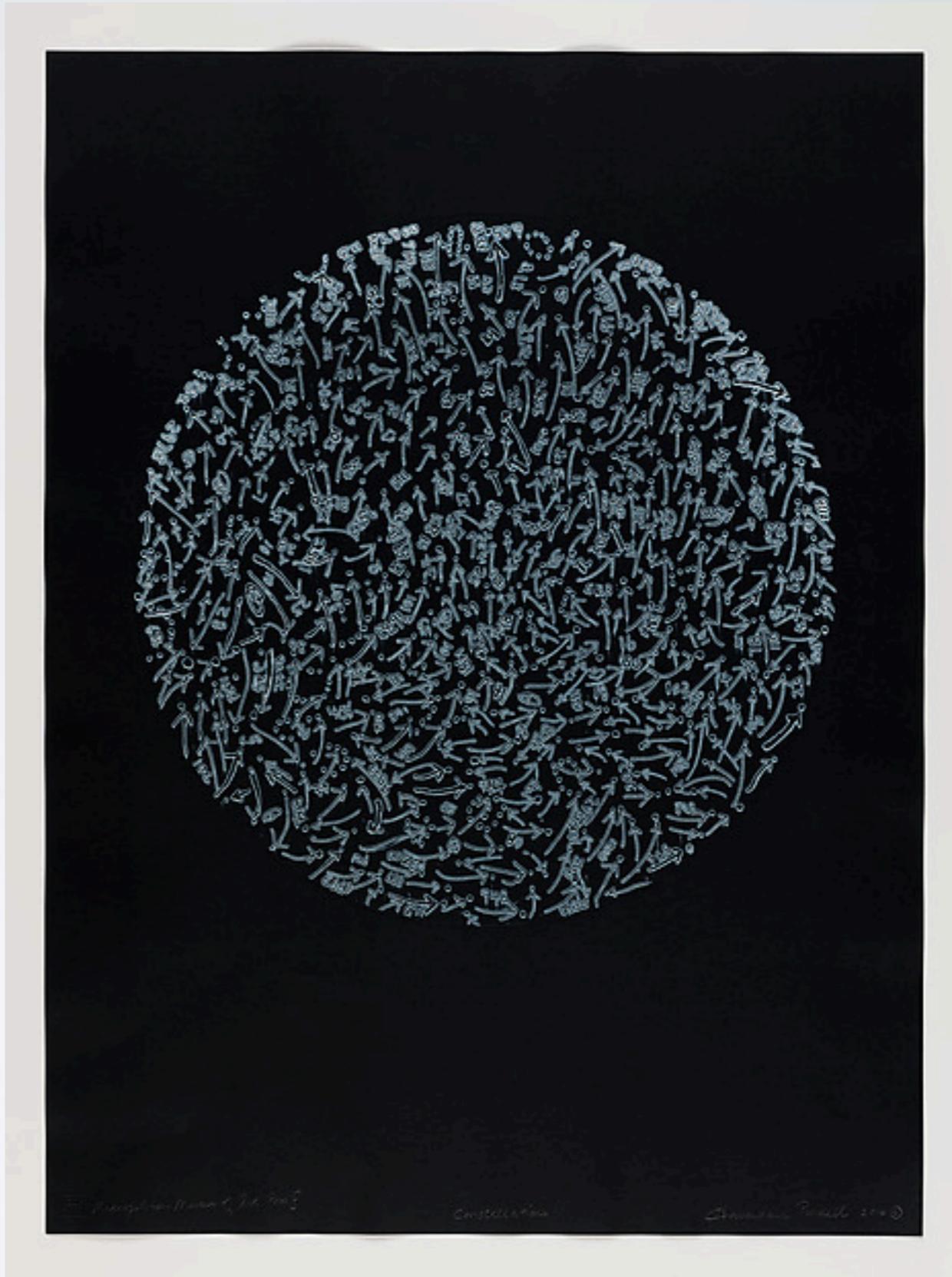
list



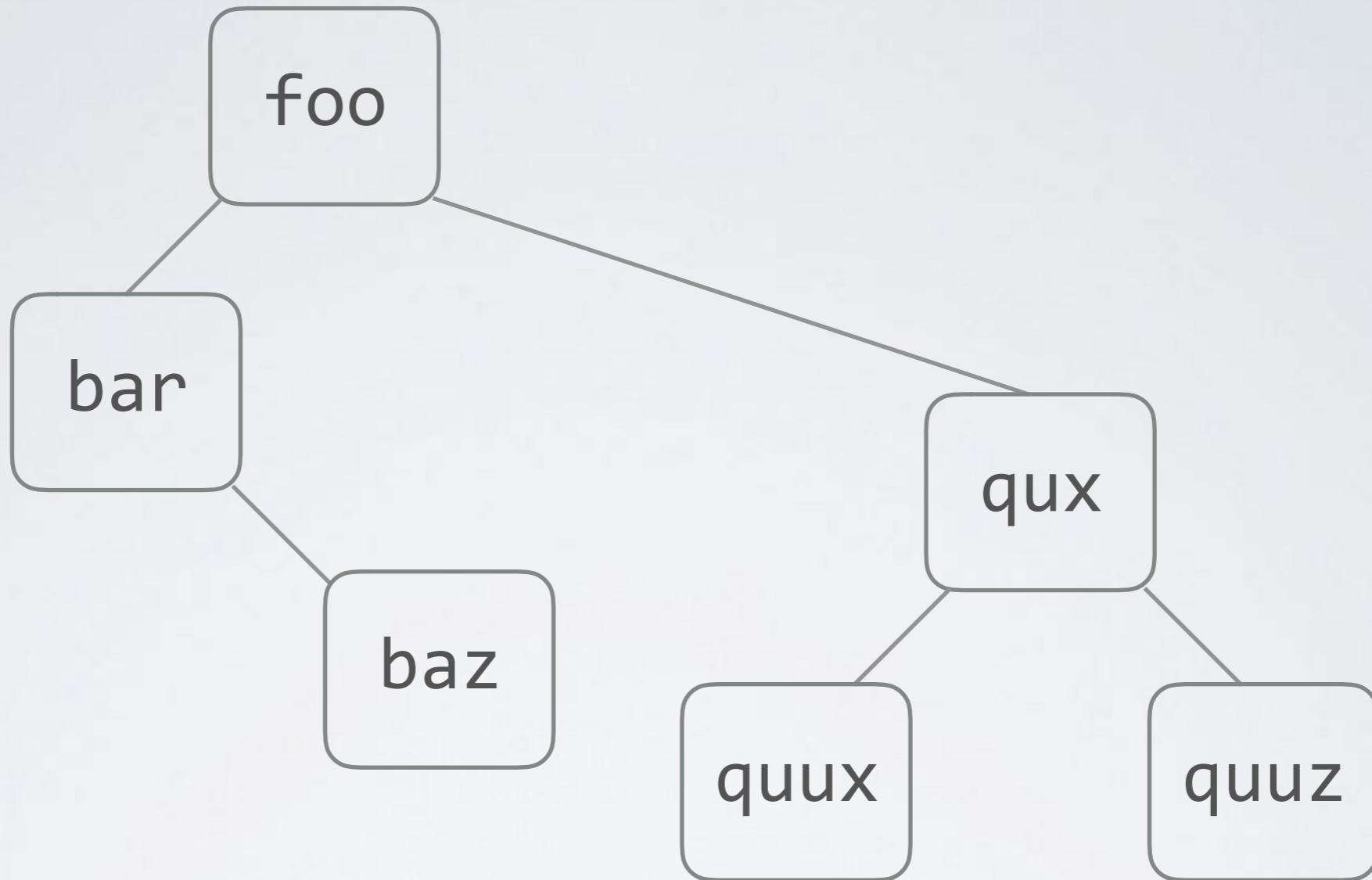
~~newList is the list piped(d(4);4)xs .presup(expnd(x))~~



SORTED
SET

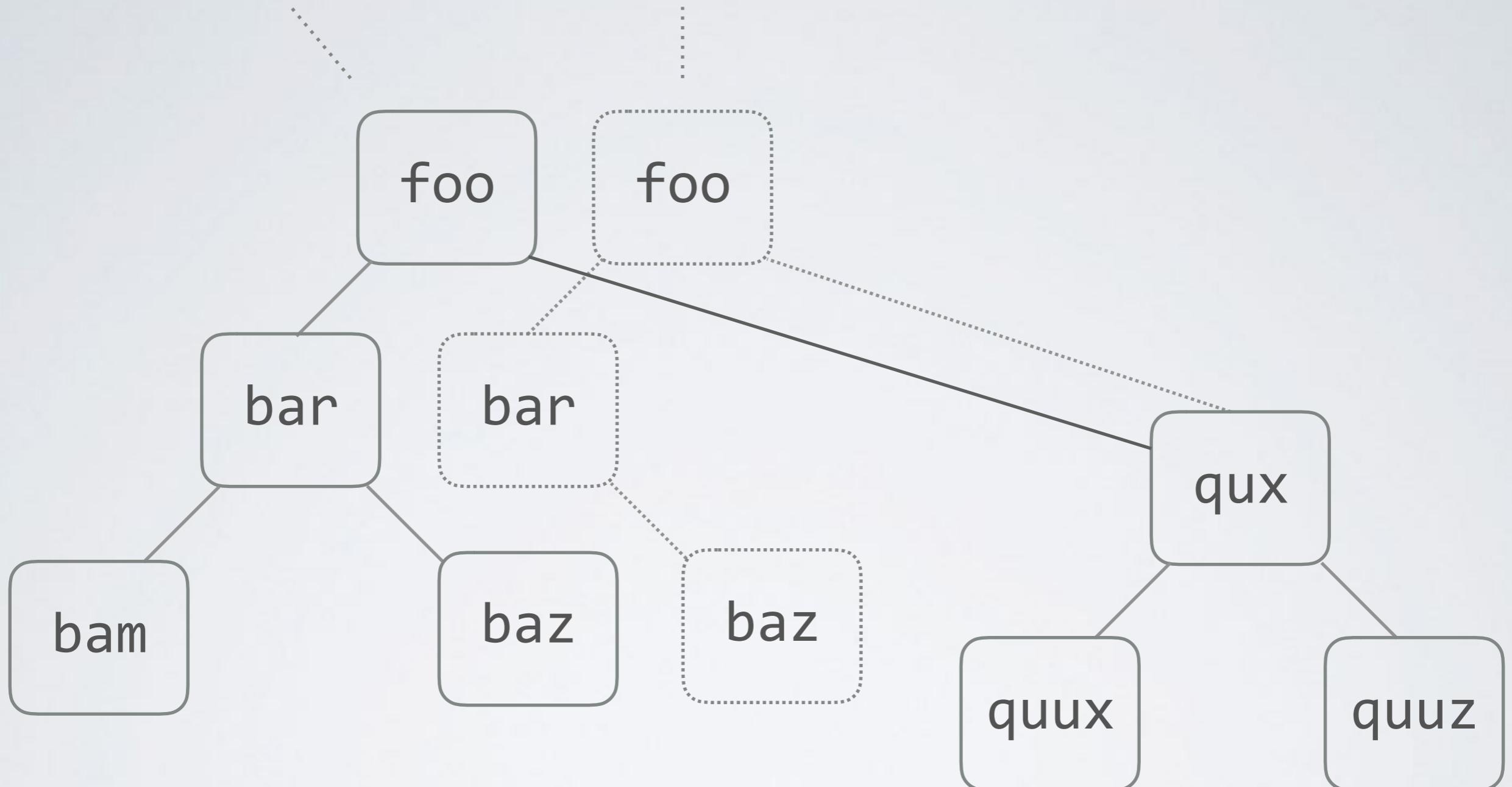


sortedSet



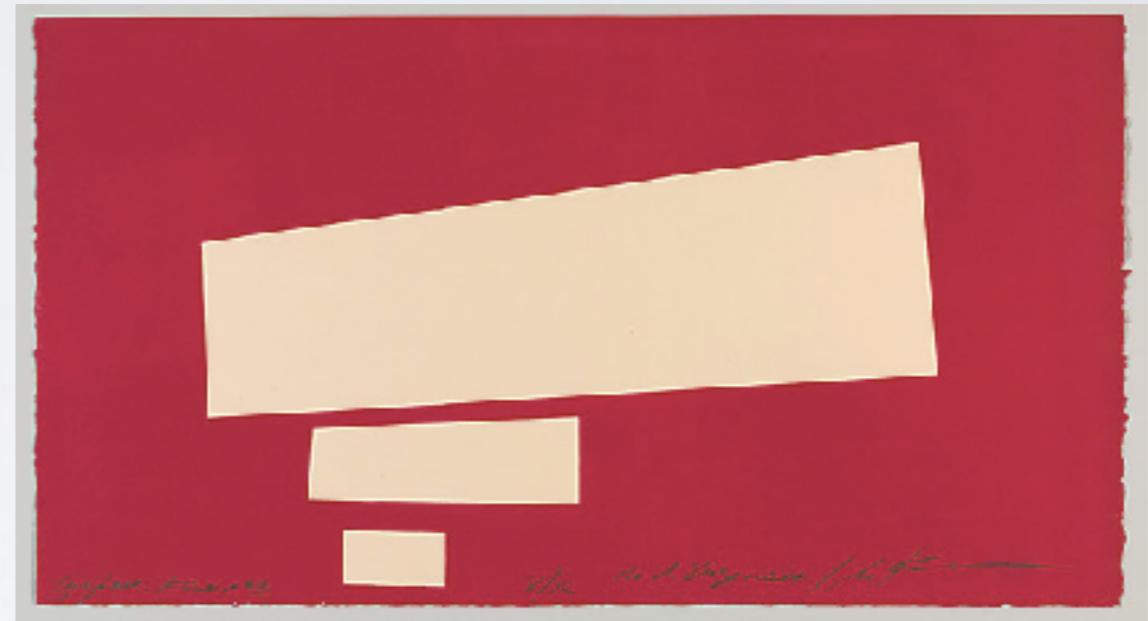
```
SortedSet<String> sortedSet =  
TreeSet.of("foo", "bar", "baz", "qux", "quux", "quuz")
```

`newSortedSet` `sortedSet`



```
SortedSet<String> newSortedSet = sortedSet.add("bam");
```

VALUES



OPTIONS



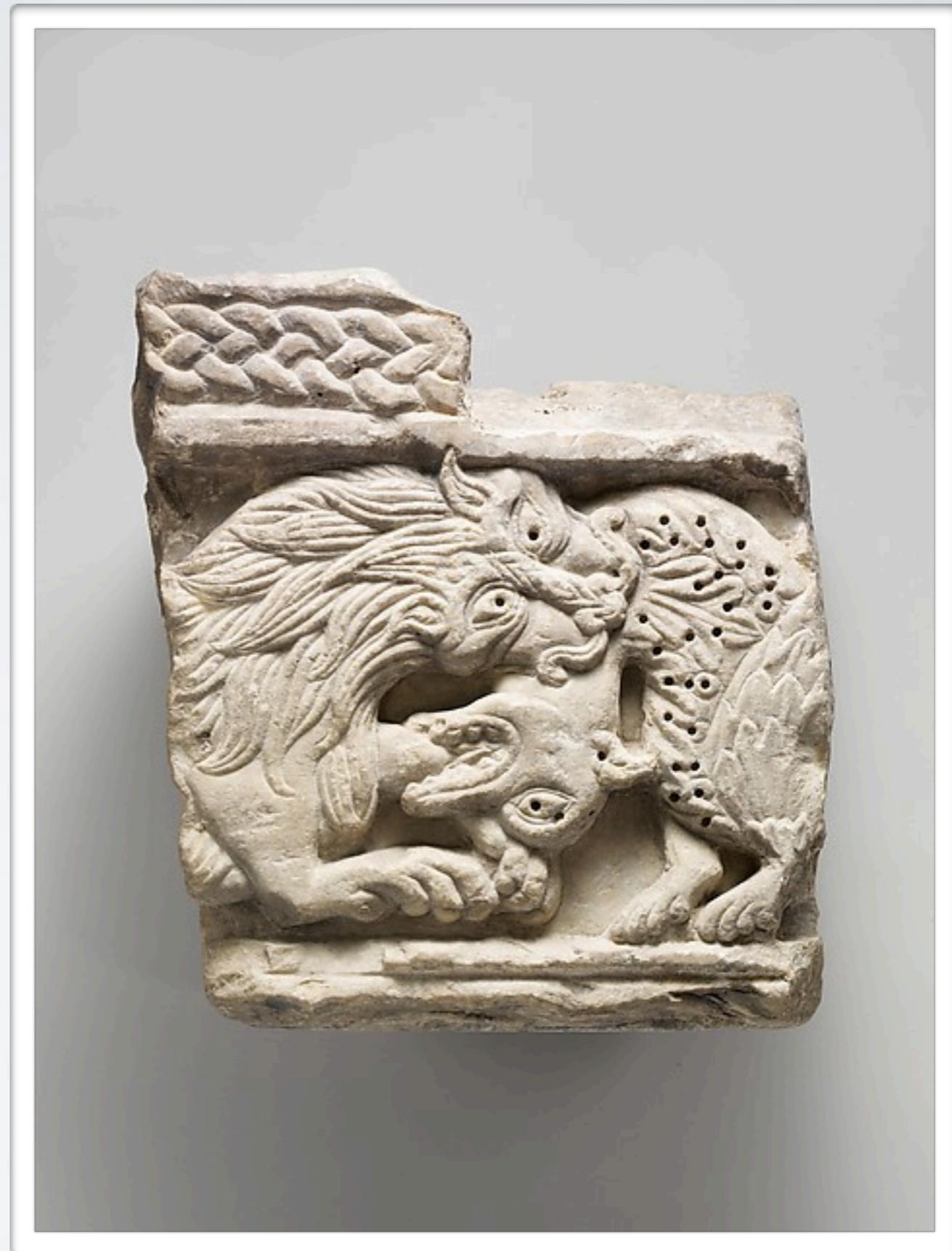
FUTURES



TRY



EITHER



PATTERN MATCHING



```
import static javaslang.API.*;
```

`$()` - wildcard pattern

`$(value)` - equals pattern

`$(predicate)` - conditional pattern



VALIDATE





PROPERTIES

FOR COMPREHENSIONS





All images are from
the creative commons open
access initiative

<http://www.metmuseum.org/>



 @dhinojosa

 dhinojosa@evolutionnext.com