Week 3: Iterators

Friday, September 18, 2020 8:18 AM

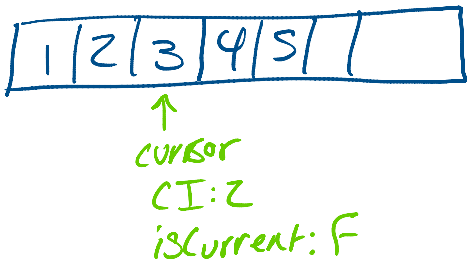
Cursors

→ Data Structure:
- → currentIndex
- → isCurrent

*not an object!

→ part of sequence
- → anytime you use cursor:
  - → advance
  - → start
  - → removeCurrent
    - *changes sequence!

*cannot have more than one cursor!



Pro to cursor:

→ simpler for user
- *can't mess anything up

Cons:

→ restrictive — only one
→ makes coding more challenging & complicated
- → a lot of checks to see if current or not

Iterators → interface *

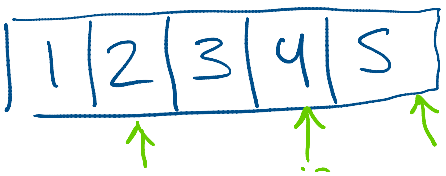
→ Data structure
- → currentIndex
- → isCurrent
- → myVersion **

→ Behaviors:
- → next
- → remove ** → default: throw UnsupportedOpE.
- → hasNext

→ all methods are abstract
- *no implementation
- *must override
- * some can have default & don't have to be overridden

*nested Class → an object
- → not part of ADT

→ collection.add → update collection version
- → i & i2 are stale
- *versions out of sync



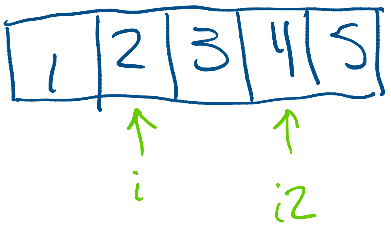
→ not part of ADT

* fail fast:
  → changing collection w/ collection methods: iterators go "stale"

  → change collection w/ iterator method: that iterator does not go stale *all others go stale

  * check versions before every iterator method
    * if different: throw CME

* can have any # of iterators
  * user's responsibility to use them well



→ i.remove()
  → update collection version & i's version
  * i2 goes stale

---

hasNext

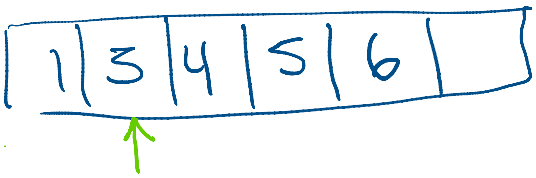
→ checks if there is a next

next

→ first make sure hasNext is true
→ advances
→ returns next element "current"

remove

→ removes current element
  → return element



isCurrent: F

Start at -1
hasNext = T
next() → move first
  → "1" is current
  → return "1"
next() → "2" is current
  → return "2"
remove() → remove "2"
  → return "2"
  → no current!
next() → "3" new current

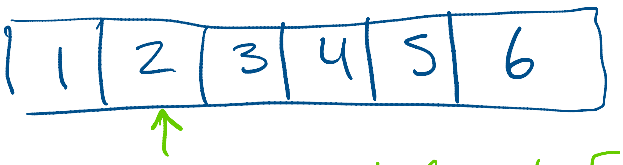
→return element

* no current element!

* update both collection & iterator version

* cannot call twice
→ need new current

next() ⇒ "3" new current



currentI = 0 isCurrent = F

hasNext: I

next: check hasNext
→ save element
→ "1" is current
→ advance
→ return "1"

Versions

→ iterator remove:
update both versions
* iterator that did removal should not go stale
→ all other iterators go stale

→ collection method that changes the collection
→ only update collection version
* all iterators go stale

* if versions do not match:
iterator is stale
* cannot be used

enhanced for loop

```
for (String element : data)
```

String: type of object
element: variable for current element
data: collections

* uses iterator *

** There is a big hint for answers.txt in the comments for

uses iterator

```
Iterator it = new Iterator();
While(it.hasNext())
    element = it.next()
```

answer . . .
the comments for
Count Occurrences!

* allowed to have multiple iterators
* as long as collection is not changing
  -> nest

```
it = new Iterator()
While (it.hasNext)
    it2 = new Iterator()
    While (it2.hasNext())
        print(it2.next())

    print(it.next())
```