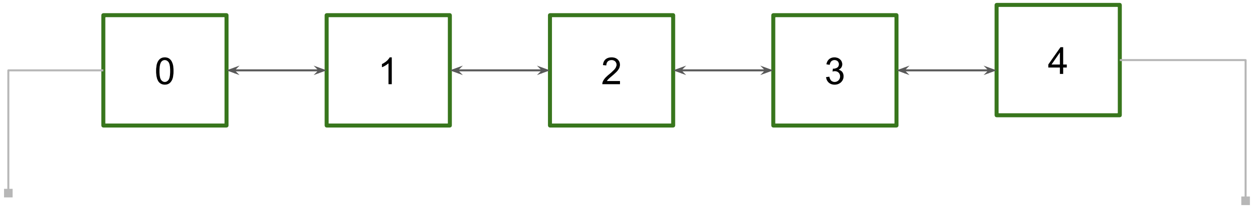
ICS-211 Lab

Assignment 4

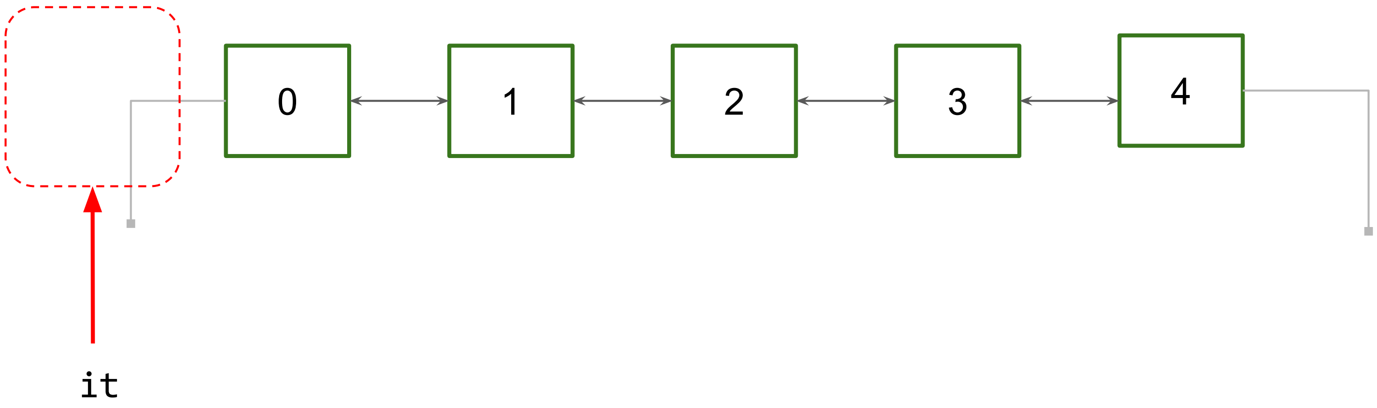
Assignment 4

- Unit tests and `ListIterator` interface in “a4” directory
- Same tests can be used for array list (need to change `MyLinkedList` to `MyArrayList`)
- The iterator tests may expose bugs in your list implementations
- `MyLinkedList` and `MyArrayList` must implement the `Iterable` interface
    - Necessary for use with special `for` loop syntax
    - The `iterator()` method returns an instance of your `ListIterator` class
    - `ListIterator` class ideally a nested class (but could be implemented differently)
-

Iterator Concept Example

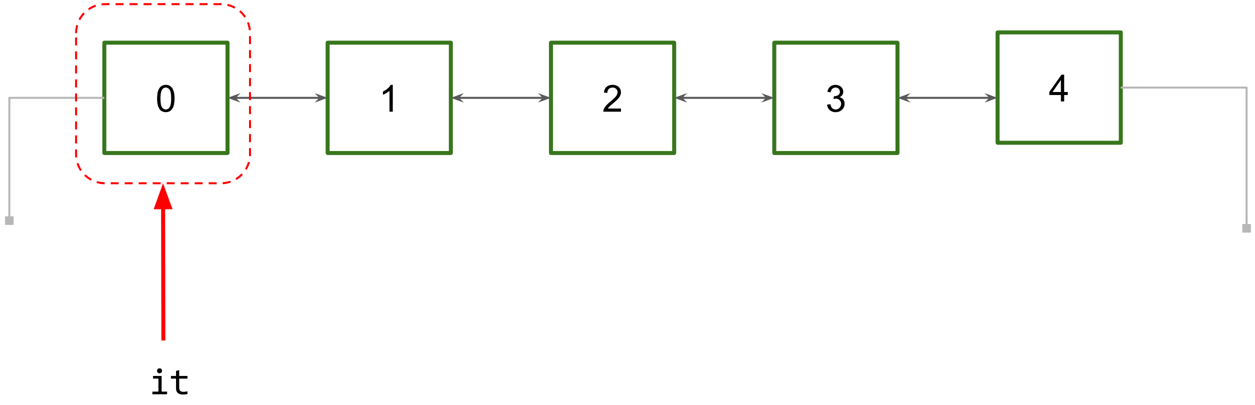


Iterator Concept Example



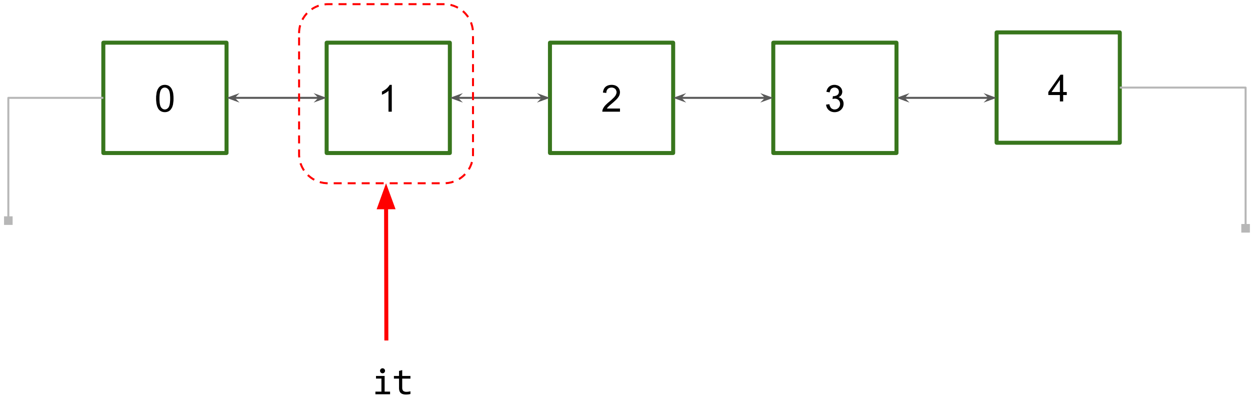
```
1.   it = list.iterator();   // it.hasNext() == true
```

Iterator Concept Example



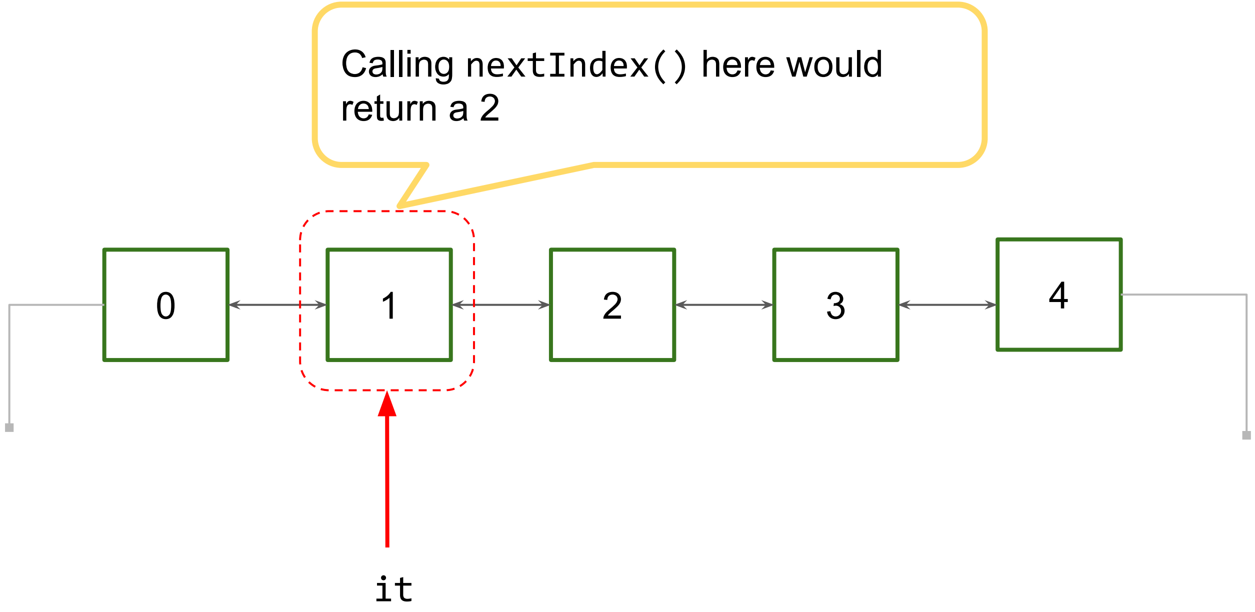
```
1.   it = list.iterator();   // it.hasNext() == true
2.   it.next();              // returns “0”
```

Iterator Concept Example



```
1.   it = list.iterator();   // it.hasNext() == true
2.   it.next();              // returns “0”
3.   it.next();              // returns “1”
```

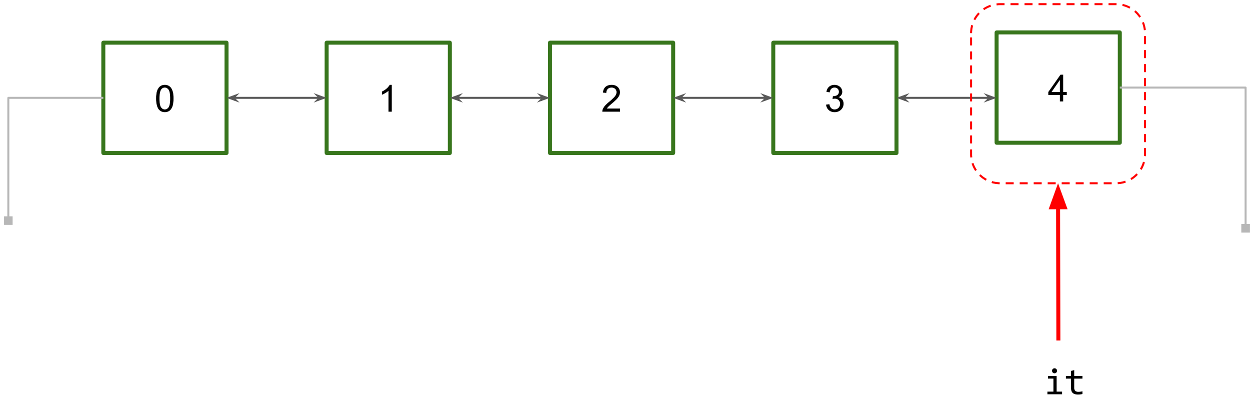
Iterator Concept Example



Calling nextIndex() here would return a 2

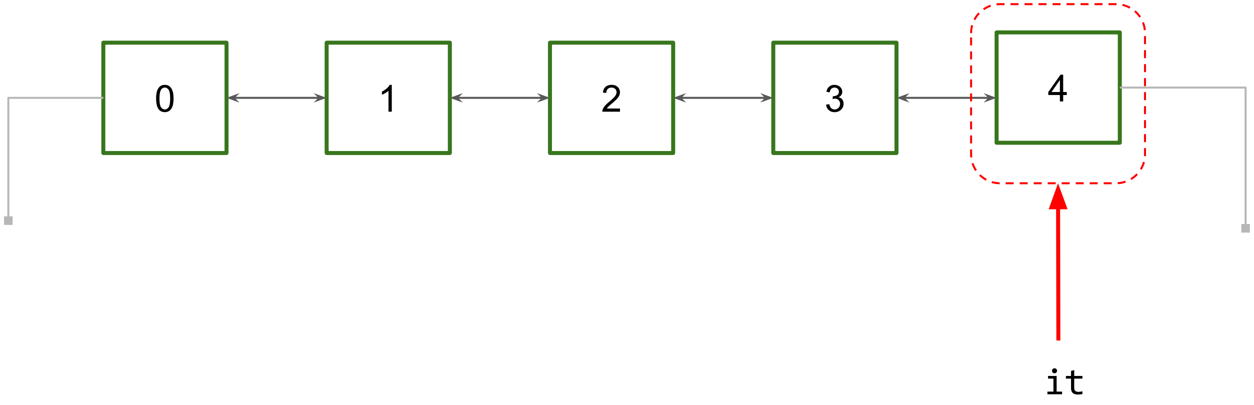
```
it = list.iterator();   // it.hasNext() == true
it.next();              // returns “0”
it.next();              // returns “1”
```

Iterator Concept Example



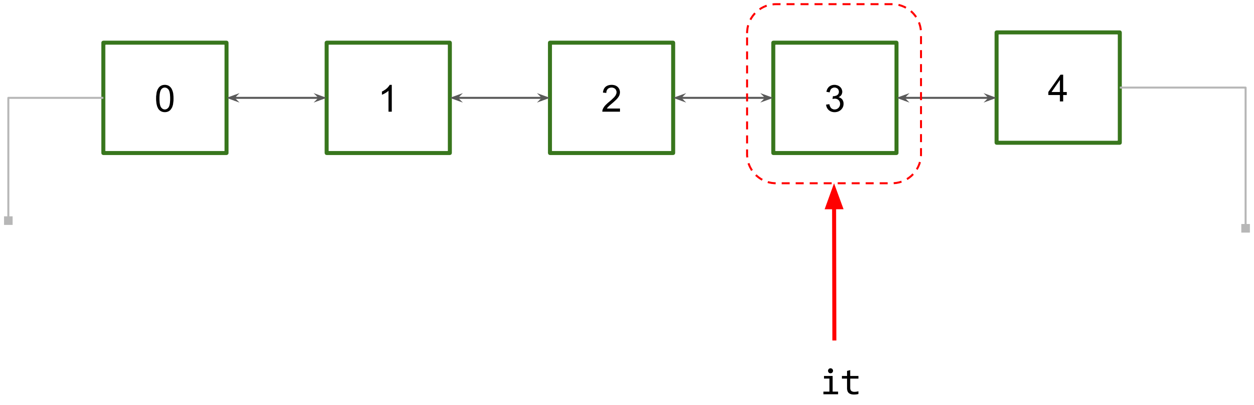
```
1.   it = list.iterator();   // it.hasNext() == true
2.   it.next();              // returns “0”
3.   it.next();              // returns “1”
4.   Keep going until “hasNext()” returns false
```

Iterator Concept Example



```
1.   it = list.iterator();   // it.hasNext() == true
2.   it.next();              // returns “0”
3.   it.next();              // returns “1”
4.   Keep going until “hasNext()” returns false
5.   hasPrevious() should return true
```

Iterator Concept Example



```
1.   it = list.iterator();   // it.hasNext() == true
2.   it.next();              // returns “0”
3.   it.next();              // returns “1”
4.   Keep going until “hasNext()” returns false
5.   hasPrevious() should return true
6.   it.previous()           // returns “3” ← this is important
```