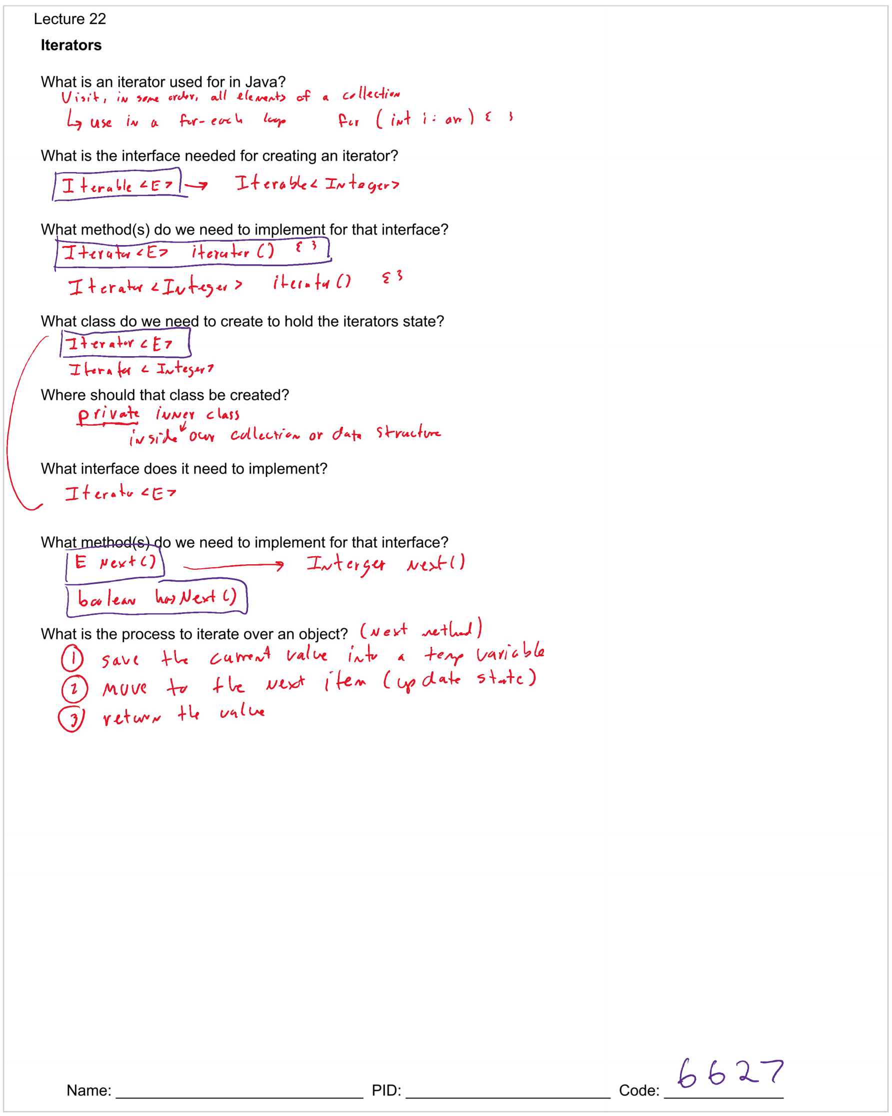
CSE12 - Lecture 22

Monday, November 27, 2023 8:00 AM

PA8 and PA6 Late/Resubmit - due Wednesday @ 8am



Lecture 22

Iterators

What is an iterator used for in Java?
Visit, in some order, all elements of a collection
↳ use in a for-each loop   for (int i : arr) { }

What is the interface needed for creating an iterator?
$\boxed{\text{Iterable<E>}}$ → Iterable<Integer>

What method(s) do we need to implement for that interface?
$\boxed{\text{Iterator<E> iterator() \{ \}}}$
Iterator<Integer> iterator() { }

What class do we need to create to hold the iterators state?
$\boxed{\text{Iterator<E>}}$
Iterator<Integer>

Where should that class be created?
private inner class
inside our collection or data structure

What interface does it need to implement?
Iterator<E>

What method(s) do we need to implement for that interface?
$\boxed{\text{E next()}}$ → Interger next()
$\boxed{\text{boolean hasNext()}}$

What is the process to iterate over an object? (next method)
1. save the current value into a temp variable
2. move to the next item (update state)
3. return the value

Name: ______________ PID: ______________ Code: 6627

```
class MyClass<E> implements Iterable<E> {
    class MyIterator<E> implements Iterator<E> {
        //state → fields
        public MyIterator( ____ ) {
            // save initial state
        }
        public E next() {
            return null;
        }
        public boolean hasNext() {
            return false;
        }
    }
    // fields of MyClass
    public Iterator<E> iterator() {
        return new MyIterator( ____ );
    }
}
```

```
MyClass<Integer> myc = new ________
for (Integer i : myc) {
    S.o.p(i);
}
```

```
Iterator<Integer> itr = myc.iterator();
while ( itr.hasNext() ) {
    Integer i = myc.next();
    S.o.p(i);
}
```

How could we make our linked list work in an enhanced for loop? What changes would we need to make to the LList class?

```
LList<Integer> list = new LList<Integer>();

//code to add data to list

for (Integer i: list) {
  System.out.println(i);
}
```

```
public class LList<E> implements Iterable<E> {
  Node front;
  int size;          boolean changed = false;

  LList() { //... }      changed = true;
  public void prepend(E value) { //... }
  public E get(int index) { //... }
  public int size() { //... }


      class LLIterator<E> implements Iterator<E> {
         //state → fields → Node<E> current;
         public MyIterator( ___ ) {
            current = front.next;
         }    changed = false;

         public E next() {        if ((changed)
                                     // throw an exception
                                  ① E temp = current.value;
                                  ② current = current.next;
                                  ③ return temp;
         }
         public boolean hasNext() {
            return current != null;
         }
      }

      public Iterator<E> iterator() {
         return new LLIterator();
      }

}
```

```
class Node<E> {
  E value;
  Node<E> next;
  public Node(E value, Node<E> next) {
    this.value = value;
    this.next = next;
  }
}
```