1 Filtered List

We want to make a `FilteredList` class that selects only certain elements of a `List` during iteration. To do so, we're going to use the `Predicate` interface defined below. Note that it has a method, `test` that takes in an argument and returns `True` if we want to keep this argument or `False` otherwise.

```
1  public interface Predicate<T> {
2          boolean test (T x);
3  }
```

For example, if `L` is any kind of object that implements `List<String>` (that is, the standard `java.util.List`), then writing

`FilteredList<String> FL = new FilteredList<String> (L, filter);`

gives an iterable containing all items, x, in L for which `filter.test(x)` is `True`. Here, `filter` is of type `Predicate`. Fill in the `FilteredList` class below.

```
1  import java.util.*;
2  public class FilteredList<T> _____ {
3
4
5          public FilteredList (List<T> L, Predicate<T> filter) {
6
7          }
8
9          @Override
10         public Iterator<T> iterator () {
11
12         }
13
14
15
16
17
18
19
20
21
22
23
24
25  }
```

2   Iterator of Iterators

Implement an `IteratorOfIterators` which will accept as an argument a `List` of `Iterator` objects containing `Integers`. The first call to `next()` should return the first item from the first iterator in the list. The second call to `next()` should return the first item from the second iterator in the list. If the list contained n iterators, the n+1th time that we call `next()`, we would return the second item of the first iterator in the list.

For example, if we had 3 `Iterators` A, B, and C such that A contained the values [1, 2, 3], B contained the values [4, 5, 6], and C contained the values [7, 8, 9], calls to `next()` for our `IteratorOfIterators` would return [1, 4, 7, 2, 5, 8, 3, 6, 9]

Feel free to modify the input `a` as needed.

```java
1  import java.util.*;
2  public class IteratorOfIterators  _____ {
3
4
5      public IteratorOfIterators(List<Iterator<Integer>> a) {
6
7
8
9
10
11
12
13      }
14
15      @Override
16      public boolean hasNext() {
17
18
19
20
21      }
22
23
24
25      @Override
26      public Integer next() {
27
28
29
30
31      }
32  }
```

3   Every κth Element (Fall 2014 MT1 Q5)

Fill in the `next()` method in the following class. Do not modify anything outside
of `next`.

```
1   import java.util.Iterator;
2   import java.util.NoSuchElementException;
3   /** Iterates over every Kth element of the IntList given to the constructor.
4    *   For example, if L is an IntList containing elements
5    *   [0, 1, 2, 3, 4, 5, 6, 7] with K = 2, then
6    *       for (Iterator<Integer> p = new KthIntList(L, 2); p.hasNext(); ) {
7    *           System.out.println(p.next());
8    *       }
9    *   would print get 0, 2, 4, 6. */
10  public class KthIntList implements Iterator <Integer> {
11      public int k;
12      private IntList curList;
13      private boolean hasNext;
14
15      public KthIntList(IntList I, int k) {
16          this.k = k;
17          this.curList = I;
18          this.hasNext = true;
19      }
20
21      /** Returns true iff there is a next Kth element. Do not modify. */
22      public boolean hasNext() {
23          return this.hasNext;
24      }
25
26      /** Returns the next Kth element of the IntList given in the constructor.
27       *   Returns the 0th element first. Throws a NoSuchElementException if
28       *   there are no Integers available to return. */
29      public Integer next() {
30          _____
31          _____
32          _____
33          _____
34          _____
35          _____
36          _____
37          _____
38          _____
39          _____
40      }
41  }
```