

1 More Practice with Linked Lists

Recall the definition of `SLList` from lecture:

```
public class SLList {
    private class IntNode {
        public int item;
        public IntNode next;
        public IntNode(int item, IntNode next) {
            this.item = item;
            this.next = next;
        }
    }

    private IntNode first;

    public void addFirst(int x) {
        first = new IntNode(x, first);
    }
}
```

1.1 Insert

Add a method to the `SLList` class that inserts a new element at the given position. If the position is past the end of the list, insert the new node at the end of the list. For example, if the `SLList` is `5 -> 6 -> 2`, `insert(10, 1)` should result in `5 -> 10 -> 6 -> 2`.

```
public void insert(int item, int position) {
```

```
}
```

1.2 Reverse

Add another method to the `SLList` class that reverses the elements. Do this using the existing `IntNodes` (you should not use `new`).

```
public void reverse() {
```

```
}
```

Bonus: If you wrote `reverse()` iteratively, write a second version that uses recursion (you may need a helper method). If you wrote it recursively, write an iterative version.

2 Arrays

2.1 Insert

Write a method that non-destructively inserts `item` into array `x` at the given position. The method should return the resulting array. For example, if `x = [5, 9, 14, 15]`, `item = 6`, and `position = 2`, then the method should return `[5, 9, 6, 14, 15]`. If `position` is past the end of the array, insert `item` at the end of the array.

```
public static int[] insert(int[] x, int item, int position) {
```

```
}
```

Is it possible to write a version of this method that returns `void` and changes `x` in place (i.e., destructively)?

2.2 Bonus: reverse

Write a method that destructively reverses the items in `x`. For example calling `reverse` on an array `[1, 2, 3]` should change the array to be `[3, 2, 1]`.

```
public static void reverse(int[] x) {
```

```
}
```

2.3 Bonus: xify

Write a non-destructive method `xify(int[] x)` that replaces the number at index `i` with `x[i]` copies of itself. For example, `xify([3, 2, 1])` would return `[3, 3, 3, 2, 2, 1]`.

```
public static int[] xify(int[] x) {
```

```
}
```