1 Flatten

Write a method `flatten` that takes in a 2-D array `x` and returns a 1-D array that contains all of the arrays in `x` concatenated together.

For example, `flatten({{1, 2, 3}, {}, {7, 8}})` should return `{1, 2, 3, 7, 8}`.

(Summer 2016 MT1)

```java
public static int[] flatten(int[][] x) {
    int totalLength = 0;

    for (int i = 0; i < x.length; i++) {  // Outer loop
        for (int j = 0; j < x[i].length; j++) {  // Inner loop
            totalLength += x[i][j];  // Incorrect
        }
    }

    int[] a = new int[totalLength];
    int aIndex = 0;

    for (int i = 0; i < x.length; i++) {  // Outer loop
        for (int j = 0; j < x[i].length; j++) {  // Inner loop
            a[aIndex] = x[i][j];  // Correct assignment
            aIndex++;  // Increment index
        }
    }

    return a;
}
```
2 Skippify

Suppose we have the following IntList class, as defined in lecture and lab, with an added skippify function.

Suppose that we define two IntLists as follows.

```
1 IntList A = IntList.list(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);
2 IntList B = IntList.list(9, 8, 7, 6, 5, 4, 3, 2, 1);
```

Fill in the method skippify such that the result of calling skippify on A and B are as below:
- After calling A.skippify(), A: (1, 3, 6, 10)
- After calling B.skippify(), B: (9, 7, 4)

(Spring '17, MT1)

```java
1 public class IntList {
2     public int first;
3     public IntList rest;
4
5     @Override
6     public boolean equals(Object o) { ... }
7     public static IntList list(int... args) { ... }
8
9     public void skippify() {
10         IntList p = this;
11         int n = 1;
12         while (p != null) {
13             IntList next = __________________________;
14             for (_______________________________) {
15                 if (_______________________________) {
16                     _______________________________
17                 }
18             }
19             _______________________________
20         }
21     }
22 }
```
3 Remove Duplicates

Fill in the blanks below to correctly implement removeDuplicates.
(Spring '17, MT1)

```java
public class IntList {
    public int first;
    public IntList rest;
    public IntList (int f, IntList r) {
        this.first = f;
        this.rest = r;
    }
}

/**
 * Given a sorted linked list of items - remove duplicates.
 * For example given 1 -> 2 -> 2 -> 2 -> 3,
 * Mutate it to become 1 -> 2 -> 3 (destructively)
 */
public static void removeDuplicates(IntList p) {
    if (p == null) {
        return;
    }

    IntList current = _________________________;
    IntList previous = ________________________;

    while (__________________________________) {
        if (____________________________________) {
            ________________________________
        } else {
            ________________________________
        }
    }
}
```