

Challenge: Frauds List

(6 Points). Suppose we have the `IntList` and `FraudsList` classes below (Summer 2021, Final)

```
1 public class IntList {
2     public int first;
3     public IntList rest;
4
5     public IntList(int f, IntList r) {
6         first = f;
7         rest = r;
8     }
9
10    public int size() {
11        IntList p = this;
12        int totalSize = 0;
13        while (p != null) {
14            totalSize += 1;
15            p = p.rest;
16        }
17        return totalSize;
18    }
19 }
20
21 class FraudList extends IntList {
22     public FraudList(int f, IntList r) {
23         super(f, r);
24     }
25     public int size() {
26         return -super.size();
27     }
28 }
```

Implement the method `findFrauds` which accepts an array of `IntLists` in which some of the elements are, or may contain, `FraudLists`! That is, the dynamic type of certain `IntList` instances is `FraudList`. As shown above, a `FraudList` is an `IntList` whose `size` method returns the negative of the correct size. You must report these `FraudLists` by **non-destructively** returning a **new** `FraudList` of all the `FraudList` instances linked together in the order they appear in `arr`.

You may **not** modify the given array `arr` or the `IntLists` inside of `FraudList`. You may **not** use `instanceOf`, `getClass()`, `isInstance()` or any method not explicitly written in the classes above or imported. An instance of the problem is shown below:

```
1 IntList first = new IntList(1000, new IntList(1002, new FraudList(1, new FraudList(2, null))));
2 IntList second = new FraudList(3, null);
3 IntList third = new IntList(3000, null);
```

```
4 IntList fourth = new FraudList(4, new IntList(231, new FraudList(5, null)));
5 IntList[] arr = new IntList[]{first, second, third, fourth};
6 FraudList frauds = findFrauds(arr);
```

After executing the lines above, `frauds` should be equal to the `FraudList` with the elements 1, 2, 3, 4, 5 and `arr`, as well as the contents within `arr`, should be unchanged. Fill in the skeleton below. You may not delete, modify, or add to any of the provided skeleton code.

```
1 import static java.lang.System.arraycopy;
2
3 public static FraudList findFrauds(IntList[] arr) {
4     IntList[] copy = new IntList[arr.length];
5     arraycopy(arr, 0, copy, 0, arr.length);
6     return helper(_____, _____);
7 }
8
9 public static FraudList helper(IntList[] copy, int index) {
10     if (_____) {
11         return null;
12     } else if (_____) {
13         return _____;
14     }
15     _____;
16     _____;
17     if (_____) {
18         return _____;
19     } else {
20         return _____;
21     }
22 }
```