CS 61B Discussion 3: [A, r, r, a, y, s] Fall 2015

1 Boxes and Pointers II

Draw a box and pointer diagram for each code block.

(a)
```
int[] x = {1, 2, 3};
int[] y = x;
y[2] = 7;
```

x and y should both point to an array with values [1, 2, 7].

(b)
```
IntList l = IntList.list(1, 2, 3);
IntList l2 = l;
l.tail.tail.head = 7;
```

l and l2 should both point to an IntList with values [1, 2, 7].

(c)
```
IntList[] ll = new IntList[3];
ll[0] = IntList.list(1, 2);
ll[1] = IntList.list(2);
```

ll should point to an array, where the first two elements point to IntLists and the third is null.

2 Debugging is good for your health

The following code is broken. Please identify and fix the errors.

```
/** Returns the sum of squares of numbers in nums. */
public int sumOfSquares(int[] nums) {
    int total = 0;
    for (int i = 0; i < nums.size; i += 1) {
        total += (nums + i) * (nums + i);
    }
    return total;
}
```

In the for loop, it should be nums.length, not nums.size. Also, to access the ith element of an array nums, it should be nums[i], not nums + i.

3 Fun with arrays

Complete the following methods according to their specifications.

```
/** Given an array A (size > 1), return the avg. of all items in A. */
public static double average(double[] A) {
    double total = 0.0;

    for (double num : A) {
        total += num;
```

```
    }

    return total / A.length;
}

import static java.lang.Math.max; // max(a, b) returns max of a, b
import static java.lang.Math.min; // min(a, b) returns min of a, b

/** Given an array A, return a 2 element array B where B[0] is the
  * minimum element of A and B[1] is the maximum element of A. */
public static int[] minMax(int[] A) {
    int maxVal = Integer.MIN_VALUE;
    int minVal = Integer.MAX_VALUE;
    int[] B = new int[2];

    for (int i = 0; i < A.length; i+= 1) {
        maxVal = max(maxVal, A[i]);
        minVal = min(minVal, A[i]);
    }
    B[0] = minVal;
    B[1] = maxVal;
    return B;
}
```

4 Bonus for Bosses: LinkedFaceitteroogle Interview

Welcome to LinkedFaceitteroogle! I hear you're interested in an engineering position here. First, let's see how well you can program.

Given an integer `k` and an array `A` of `n` integers, design an algorithm to move `A[k]` to the left-most index such that all elements up to index `k` are sorted in increasing order. You may assume that prior to moving `A[k]`, all elements up to index `k - 1` are sorted in increasing order and that `k < n`.

(a)
```
public static void moveInt(int[] A, int k) {
    int temp;

    for (int i = k; i > 1; i -= 1) {
        if (a[i - 1] >= a[i]) {
            temp = a[i];
            a[i] = a[i - 1];
            a[i - 1] = temp;
        }
    }
}
```

(b) How can we extend this algorithm to sort an entire array?
We can call `moveInt` multiple times on an array `A` and set `k = 1, 2, ..., A.length - 1`. Each call will place `A[k]` in the correct position and build a sorted array from the beginning of the array up until `k`, for each `k`.