CS 61B Discussion 3: [A, r, r, a, y, s] Fall 2016

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 Min/Max

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`.

```
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

public static int[] minMax(int[] A) {
    int maxVal = Integer.MIN_VALUE; // smallest int in Java
    int minVal = Integer.MAX_VALUE; // largest int in Java

    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;

}
```

3 Reverse

Given an array `A`, reverse its elements in place (i.e. do not create any new arrays; this should be a destructive method).

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

    for (int i = 0; i < A.length / 2; i++) {
        int temp = A[A.length - i - 1];
        A[A.length - i - 1] = A[i];
        A[i] = temp;
    }


}
```

4 Beast Mode: Matrix Multiplication

Given two matrices `A` and `B`, return the matrix `AB`. For instance if `A = [[1, 2]]` and `B = [[-1], [2]]`, then `AB = [[3]]`. You may assume that `A` and `B` are not ragged and that the number of columns of `A` equals the number of rows of `B` (i.e. we can actually multiply `A` and `B`).

```
public static int[][] multiply(int[][] A, int[][] B) {

    int ARows = A.length;
    int ACols = A[0].length;
    int BCols = B[0].length;
    int[][] AB = new int[ARows][BCols];
    for (int i = 0; i < ARows; i++) {
        for (int j = 0; j < BCols; j++) {
            for (int k = 0; k < ACols; k++) {
                AB[i][j] += A[i][k] * B[k][j];
            }
        }
    }
    return AB;


}
```