Array Recursion Example

DIS has devised an inefficient, but interesting way to reverse the elements in an array. As shown below and on the next page, method `flip` reverses a general 1-D array of integers via a recursive method called `flip2` that has the signature

\[
\text{flip2}(\text{int}[] \ x, \ \text{int} \ a, \ \text{int} \ b).
\]

Indices \(a\) and \(b\) represent the first and last indices of the input array \(x\), respectively.

You need to complete the implementation of method `flip2`. Do not use extra memory by creating arrays in `flip2`!

To recursively reverse the array, you must follow this pattern:

- If the array length is zero or one, stop recursing.
- If the array length is greater than one, swap the left and right halves of that array between indices \(a\) and \(b\) (inclusive) without creating a new array. For example, for \(a=0\) and \(b=3\), flipping \(\{1, 2, 3, 4\}\) once rearranges the array into \(\{3, 4, 1, 2\}\) by swapping elements \(3\) and \(4\) with \(1\) and \(2\). If the array length is odd, swap the elements around the middle element. Continue by flipping both of those halves recursively.

For example, reversing the array \(\{2, 3, 1, 4\}\) would have this pattern:

\[
\{2, 3, 1, 4\} \rightarrow \{1, 4, 2, 3\} \rightarrow \{4, 1, 3, 2\}
\]

An example of an odd-length array \(\{1, 2, 3, 4, 5\}\) has this pattern:

\[
\{1, 2, 3, 4, 5\} \rightarrow \{4, 5, 3, 1, 2\} \rightarrow \{5, 4, 3, 2, 1\}
\]

[code appears on next page; use the following space to refine your algorithm; Hint: Work out formulas for figuring out the index for each half of the array between \(a\) and \(b\).]
public class ReverseArray {

    public static void main(String[] args) {
        int[] x1 = {2,3,1,4};
        int[] x2 = {1,2,3,4,5};
        print(flip(x1)); // outputs {4,1,3,2}
        print(flip(x2)); // outputs {5,4,3,2,1}
    }

    // Reverse the elements in x and return that array:
    public static int[] flip(int[] x) {
        flip2(x, 0, x.length-1);
        return x;
    }

    // Reverse the elements in x in place, so do not create new arrays:
    private static void flip2( int[]x, int a, int b ) {

    }

    public static void print(int[] x) { /* code not shown */ }
}
} // Class ReverseArray