MidtermReview/BinarySearch.java

```
// P4BinarySearch.java

/**
 * P4
 * Recursive Programming Problem
 *
 * Recall the binary search algorithm, which searches through a sorted array for
    a particular value, and returns the index of where the value was found in the
    array; it returns with -1 if the value cannot be located.

    An incomplete recursive solution for this algorithm appears below:

    static int binary (int [] a, int fromIndex,
        int toIndex, int key)
    {
        if (fromIndex > toIndex) return ???1 ;
        else {
            int middle = (fromIndex + toIndex)/2;
            if (key == a[middle]) return ???2 ;
            else if (key > a[middle]) return binary( ???3 ) ;
            else return binary ( ???4 ) ;
        }
    }
 */

/**
 * The first parameter a, is an array of sorted integers that gets searched. The
fourth parameter (key) is the integer value that is being search for inside of a.
The second and third parameters (fromIndex and toIndex) stipulate the array
indices of where to begin and end the search. If we wanted to search
through an entire sorted integer array named foobar for the value -17, we
might call on the above method as follows:

    binary (foobar, 0, foobar.length-1, -17)

*/

public class BinarySearch {
    public static void main(String[] args) {
        int[] foobar = { -20, -15, -10, -5, 0, 5, 10, 15, 20 }; // sorted array
        int key = 5;

        int result = binary(foobar, 0, foobar.length - 1, key);

        if (result == -1) {
            System.out.println(key + " not found in the array.");
        } else {
            System.out.println(key + " found at index " + result);
        }
    }

    static int binary(int[] a, int fromIndex, int toIndex, int key) {
        if (fromIndex > toIndex) {
```

```java
            return -1; // Return -1 to indicate that the key was not found
        } else {
            int middle = (fromIndex + toIndex) / 2;
            if (key == a[middle]) {
                return middle; // Return the index where the key was found
            } else if (key > a[middle]) {
                return binary(a, middle + 1, toIndex, key); // Recursively search the right half
            } else {
                return binary(a, fromIndex, middle - 1, key); // Recursively search the left half
            }
        }
    }
}
```