```java
// SortTools.java
/*
 * EE422C Project 1 submission by
 * Replace <...> with your actual data.
 * <Tao Zhu>
 * <tz3694>
 * <15455>
 * Spring 2017
 * Slip days used:
 */

package assignment1;
public class SortTools {
    /**
      * This method tests to see if the given array is sorted.
      * @param x is the array
      * @param n is the size of the input to be checked
      * @return true if array is sorted
      */
    public static boolean isSorted(int[] x, int n)
    {
        if(n == 0 || x.length == 0)
        {
            return false;
        }
        for(int i = 0; i < n-1; i++)
        {
            if(x[i] > x[i+1])
            {
                return false;
            }
        }
        return true;
    }

    /**
      * This method find whether a value exists in an array.
      * @param nums is the array
      * @param n is the size of the input to be checked
      * @param v is the value to be checked
      * @return the position of the value. Return -1 if v is not found.
      */
    public static int find(int[] nums, int n, int v)
    {
        int low = 0, high = n - 1, mid=0;
        while(low <= high)
        {
            mid = (low + high) / 2;
            if(nums[mid] == v)
            {
                return mid;
            }
            else if(nums[mid] < v)
            {
                low = mid + 1;
            }
            else
            {
                high = mid - 1;
            }
        }
        return -1;
    }

    /**
      * This method insert a value to a sorted array.
      * @param nums is the array
      * @param n is the size of the input to be checked
      * @param v is the value to be inserted
      * @return the new array after insertion
      */
    public static int[] insertGeneral(int[] nums, int n, int v)
    {
        int pos = find(nums, n, v);
        int[] new_nums;
        if(pos != -1)
        {
            new_nums = new int[n];
            for(int i = 0; i < n; i++)
            {
                new_nums[i] = nums[i];
            }
        }
        else
        {
            new_nums = new int[n+1];
            int insert_pos = 0;
            for(insert_pos = 0; insert_pos < n + 1; insert_pos++)
            {
                if(insert_pos < n && nums[insert_pos] < v)  //edge case: insert_pos is at the end of the array
                {
                    new_nums[insert_pos] = nums[insert_pos];
                }
                else
                {
                    new_nums[insert_pos] = v;
                    break;
                }
            }

            for(int i = insert_pos + 1; i < n + 1; i++)
            {
                new_nums[i] = nums[i-1];
            }

        }
        return new_nums;
    }

    /**
      * This method insert a value in a sorted array in place.
      * @param num is the array
      * @param n is the size of the input to be checked
      * #param v is the value to be inserted
      * @return the length of the new array
      */
    public static int insertInPlace(int[] nums, int n, int v)
    {
        int pos = find(nums, n, v);
        if(pos != -1)
        {
            return n;
        }
        else
        {
            int insert_pos = 0;
            for(insert_pos = 0; insert_pos < n + 1; insert_pos++)
            {
                if(insert_pos < n && nums[insert_pos] < v)  //edge case: insert_pos is at the end of the array
                {
```

```java
                    //Do nothing
                }
                else
                {
                    break;
                }
            }

            for(int i = n; i > insert_pos; i--) //precondition: n < nums.length
            {
                nums[i] = nums[i-1];
            }
            nums[insert_pos] = v;
            return (n + 1);
        }
    }

    /**
     * This method sorts an array.
     * @param nums is the array
     * @param n is the size of the input to be sorted
     */
    public static void insertSort(int[] nums, int n)
    {
        for(int i = 1; i < n; i++)
        {
            int key = nums[i];
            for(int j = i - 1; j >= 0; j--)
            {
                if(nums[j] > key)
                {
                    nums[j + 1] = nums[j];
                }
                else
                {
                    nums[j + 1] = key;
                    break;
                }
            }
        }
    }
}
```