

```
package genericity;

public class ArrayOps {

    /**
     * @requires A.length >= 1
     * @param A a non-empty array of some type
     * @return the middle element of the array
     */
    static <T> T midpoint (T[] A) {
        assert A.length >= 1;
        return A[A.length/2];
    }

    <T> int nonNullLength (T[] A) {
        int count = 0;
        for (T t : A) {
            if (t != null) {
                count++;
            }
        }
        return count;
    }

    public static void main(String[] args) {
        String result;
        int count;
        ArrayOps arrayWorker = new ArrayOps();

        //correct usage:
        result = ArrayOps.midpoint(args);
        result = ArrayOps.<String>midpoint(args);
        count = arrayWorker.nonNullLength(args);
        count = arrayWorker.<String>nonNullLength(args);

        //correct syntax, poor style
        result = arrayWorker.midpoint(args); //static member via reference
        result = arrayWorker.<String>midpoint(args); //static member via reference

        //compile-time errors
        // result = <String>midpoint(args); //syntax error
        // result = ArrayOps.<Integer>midpoint(args); //type mismatch
        // count = <String>nonNullLength(args); //syntax error
        // count = arrayWorker.<Integer>nonNullLength(args); //type mismatch

        System.out.println(result + count);
    }
}
```