Selected Array Problems

Practice writing methods that traverse and manipulate arrays. For each problem, write the complete method body.

---

countEvens — Return the number of even values in the array.

```
public static int countEvens(int[] arr)
```

countInRange — Return how many elements fall between `lo` and `hi`, inclusive.

```
public static int countInRange(int[] arr, int lo, int hi)
```

sumArray — Return the sum of all elements.

```
public static int sumArray(int[] arr)
```

averageArray — Return the average of all elements as a `double`.

```
public static double averageArray(int[] arr)
```

findLast — Return the index of the last occurrence of `target`, or `-1` if not found.

```
public static int findLast(int[] arr, int target)
```

isSorted — Return `true` if the array is in non-decreasing order.

```
public static boolean isSorted(int[] arr)
```

allPositive — Return `true` if every element is greater than zero.

```
public static boolean allPositive(int[] arr)
```

anyNegative — Return `true` if at least one element is less than zero.

```
public static boolean anyNegative(int[] arr)
```

findRange — Return the difference between the largest and smallest elements.

```
public static int findRange(int[] arr)
```

longestRun — Return the length of the longest streak of consecutive equal elements.

```
public static int longestRun(int[] arr)
```

countPairs — Return the number of adjacent pairs where both elements are equal.

```
public static int countPairs(int[] arr)
```

isUnique — Return `true` if no two elements in the array are equal.

```
public static boolean isUnique(int[] arr)
```

reverseArray — Return a new array with the elements in reverse order.

```
public static int[] reverseArray(int[] arr)
```

runningSum — Return a new array where element `i` is the sum of elements `0` through `i`.

```
public static int[] runningSum(int[] arr)
```

removeAll — Return a new array with all occurrences of `target` removed.

```
public static int[] removeAll(int[] arr, int target)
```

rotateLeft — Return a new array with every element shifted one position left; the first element wraps to the end.

```
public static int[] rotateLeft(int[] arr)
```

twoSum — Return `true` if two different elements in the array add up to `target`.

```
public static boolean twoSum(int[] arr, int target)
```

intersect — Return the count of values that appear in both arrays (count each value only once).

```
public static int intersect(int[] arr1, int[] arr2)
```

mostFrequent — Return the value that appears most often (first occurrence wins ties).

```
public static int mostFrequent(int[] arr)
```

removeDuplicates — Given a sorted array, return a new array with duplicates removed.

```
public static int[] removeDuplicates(int[] arr)
```