```
/**
 * Removes x from the prefix a[0..n] (but see below for details)
 *
 * @param a the non-null array to remove from
 * @param n the length of the prefix
 * @param x the value to remove
 * @requires 0 <= n <= a.length
 * @modifies a
 */
public static int removeValue(int[] a, int n, int x)
```

Specifications

spec A
```
// @effects if x does not occur in a[0..n], leaves a unchanged;
//   otherwise, if the first occurrence of x in a[0..n] is at index i,
//   then the new values in a[0..n-1] will be the old values of a[0..i]
//   followed by the old values of a[i+1..n], and the new value of a[n-1]
//   is unspecified.
// @return n - 1 if x was removed, else n
```

spec B
```
// @effects if x does not occur in a[0..n], leaves a unchanged;
//   otherwise, the new values in a[0..n-1] are a possibly reordered
//   version of the old values of a[0..n] with one fewer occurrence
//   of x, and the new value of a[n-1] is unspecified
// @return n - 1 if x was removed, else n
```

spec C
```
// @effects if x does not occur in a[0..n], leaves a unchanged;
//   otherwise, if x occurs k > 0 times in a[0..n], then remove all k
//   occurrences of x, leaving remaining elements in the same relative
//   order as before; the new value of elements in a[n-k..n] is
//   unspecified
// @return n - k
```

spec D
```
// @requires a[0..n] contains no duplicates
// @effects if x does not occur in a[0..n], leaves a unchanged;
//   otherwise, if x occurs in a[0..n] at index i, then the new
//   values in a[0..n-1] will be the old values of a[0..i]
//   followed by the old values of a[i+1..n], and the new value
//   of a[n-1] is unspecified.
// @return n - 1 if x was removed, else n
```

Implementations

impl 1
```
int w = 0;
for (int r = 0; r < n; r++) {
    if (a[r] != x) {
        a[w] = a[r];
        w++;
    }
}
return w;
```

impl 2
```
for (int i = 0; i < n; i++) {
    if (a[i] == x) {
        for (int k = i; k < n - 1; k++) {
            a[k] = a[k + 1];
        }
        return n - 1;
    }
}
return n;
```

impl 3
```
for (int i = n - 1; i >= 0; i--) {
    if (a[i] == x) {
        for (int k = i; k < n - 1; k++) {
            a[k] = a[k + 1];
        }
        return n - 1;
    }
}
return n;
```

impl 4
```
for (int i = 0; i < n; i++) {
    if (a[i] == x) {
        a[i] = a[n - 1];
        return n - 1;
    }
}
return n;
```