

Array Tracing Review Worksheet

Oct 8, 2026 · Name: _____ Period: _____

Part 1: Easy

1. What is printed?

```
int[] a = {3, 1, 4, 1, 5};
int sum = 0;
for (int i = 0; i < a.length; i++)
    sum += a[i];
System.out.println(sum);
```

Output: _____

2. What are the contents of **a** after this code runs?

```
int[] a = {2, 4, 6, 8, 10};
a[1] = a[3];
a[4] = a[0] + a[2];
```

a = _____

3. What are the contents of **a**, and what is the value of **a[2] + a[4]**?

```
int[] a = new int[5];
for (int i = 0; i < a.length; i++)
    a[i] = i * i;
```

a = _____ **a[2] + a[4]** = _____

4. What are the final values of **max** and **updates**?

```
int[] a = {5, 2, 8, 1, 9};
int max = a[0];
int updates = 0;
for (int i = 1; i < a.length; i++) {
    if (a[i] > max) {
        max = a[i];
        updates++;
    }
}
```

max = _____ **updates** = _____

5. What is printed? (Watch the loop update.)

```
String[] w = {"cat", "dog", "bird", "fish"};
for (int i = w.length - 1; i >= 0; i -= 2)
    System.out.print(w[i] + " ");
```

Output: _____

Part 2: Medium

6. What are the contents of **a** after this code runs?

```
int[] a = {1, 2, 3, 4, 5};
int first = a[0];
for (int i = 0; i < a.length - 1; i++)
    a[i] = a[i + 1];
a[a.length - 1] = first;
```

a = _____

7. What are the contents of **a** after this code runs?

```
int[] a = {3, -1, 4, -2, 5};
for (int i = 1; i < a.length; i++)
    a[i] += a[i - 1];
```

a = _____

8. What are the final values of **count** and **total**?

```
int[] a = {4, 7, 2, 9, 6, 3};
int count = 0, total = 0;
for (int x : a) {
    if (x % 2 == 0) {
        count++;
        total += x;
    } else {
        total -= x;
    }
}
```

count = _____ total = _____

9. What are the contents of **a** after this code runs?

```
int[] a = {1, 2, 3, 4, 5};
for (int i = 0; i < a.length - 1; i += 2) {
    int t = a[i];
    a[i] = a[i + 1];
    a[i + 1] = t;
}
```

a = _____

10. Draw a box-and-arrow diagram if it helps. What is printed?

```
int[] a = {1, 2, 3};
int[] b = a;
int[] c = {1, 2, 3};
b[0] = 9;
c[1] = b[2] + 1;
System.out.println(Arrays.toString(a));
System.out.println(Arrays.toString(c));
System.out.println(a == b);
System.out.println(a == c);
```

Line 1: _____ Line 2: _____ Line 3: _____ Line 4: _____

Part 3: Challenging

11. (Nested loops) What are the contents of `r` after this code runs?

```
int[] a = {5, 3, 5, 1, 4};
int[] r = new int[a.length];
for (int i = 0; i < a.length; i++)
    for (int j = 0; j < a.length; j++)
        if (a[j] < a[i])
            r[i]++;
```

`r` = _____

12. (A method that modifies an array) Consider this method:

```
public static void mystery(int[] arr, int n) {
    for (int i = 0; i < arr.length; i++)
        arr[i] += n;
    arr = new int[]{0, 0, 0};
    n = 100;
    arr[0] = n;
}
```

What are the contents of `data` and the value of `k` after this code runs?

```
int[] data = {1, 2, 3};
int k = 5;
mystery(data, k);
```

`data` = _____ `k` = _____

13. (A method that modifies its parameter and returns a new array) Consider this method:

```
public static int[] process(int[] a) {  
    int[] result = new int[a.length];  
    for (int i = 0; i < a.length; i++) {  
        result[i] = a[i] * 2;  
        if (i > 0)  
            a[i - 1] = result[i];  
    }  
    return result;  
}
```

What are the contents of **a** and **b** after this code runs?

```
int[] a = {1, 2, 3, 4};  
int[] b = process(a);
```

a = _____ b = _____

14. (Array of objects) Consider this class:

```
public class Counter {  
    private String name;  
    private int count;  
    public Counter(String name) { this.name = name; this.count = 0; }  
    public void increment() { count++; }  
    public String toString() { return name + ":" + count; }  
}
```

What is printed by the last line of this code?

```
Counter[] cs = new Counter[3];  
for (int i = 0; i < cs.length; i++)  
    cs[i] = new Counter("c" + i);  
Counter x = cs[1];  
cs[2] = x;  
x.increment();  
cs[0].increment();  
cs[2].increment();  
cs[1] = new Counter("new");  
cs[1].increment();  
System.out.println(cs[0] + " " + cs[1] + " " + cs[2] + " " + x);
```

Output: _____

15. (Array of arrays) What are the contents of `g` after this code runs? Write it as three rows.

```
int[] [] g = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
for (int r = 0; r < g.length; r++) {
    for (int c = r + 1; c < g[r].length; c++) {
        int t = g[r][c];
        g[r][c] = g[c][r];
        g[c][r] = t;
    }
}
int[] row = g[0];
g[0] = g[2];
g[2] = row;
for (int r = 0; r < g.length; r++)
    g[r][r] = 0;
```

Row 0: _____ Row 1: _____ Row 2: _____

16. (Array of objects, shared references) Consider this class and these methods:

```
public class Player {
    private String name;
    private int pts;
    public Player(String name, int pts) { this.name = name; this.pts = pts; }
    public void add(int n) { pts += n; }
    public String toString() { return name + "=" + pts; }
}

public static void swap(Player[] ps, int i, int j) {
    Player t = ps[i];
    ps[i] = ps[j];
    ps[j] = t;
}

public static void bonus(Player[] ps) {
    for (Player p : ps)
        p.add(5);
    ps = new Player[] {new Player("X", 0)};
    ps[0].add(1);
}
```

What is printed by the last line of this code?

```
Player[] ps = {new Player("A", 10), new Player("B", 20), new Player("C", 30)};
Player top = ps[2];
swap(ps, 0, 2);
bonus(ps);
top.add(100);
System.out.println(ps[0] + " " + ps[1] + " " + ps[2] + " " + top);
```

Output: _____

17. (Calling a method that modifies the array twice) Consider this method:

```
public static void shiftRight(int[] a) {  
    for (int i = a.length - 1; i > 0; i--)  
        a[i] = a[i - 1];  
}
```

What are the contents of `a` after this code runs?

```
int[] a = {1, 2, 3, 4, 5};  
shiftRight(a);  
shiftRight(a);  
a[0] = a[a.length - 1] + a[1];
```

`a` = _____

18. (Jagged array) What is printed?

```
int[][] t = new int[4][];  
int total = 0;  
for (int i = 0; i < t.length; i++) {  
    t[i] = new int[i + 1];  
    t[i][0] = 1;  
    t[i][i] = 1;  
    for (int j = 1; j < i; j++)  
        t[i][j] = t[i - 1][j - 1] + t[i - 1][j];  
    for (int v : t[i])  
        total += v;  
}  
System.out.println(Arrays.toString(t[3]) + " " + total);
```

Output: _____

19. (Shifting inside a loop) What are the final values of `n` and the contents of `a`?

```
int[] a = {2, 2, 3, 3, 3, 5, 2};  
int n = 1;  
for (int i = 1; i < a.length; i++) {  
    if (a[i] != a[n - 1]) {  
        a[n] = a[i];  
        n++;  
    }  
}
```

`n` = _____ `a` = _____

20. (Array of arrays with aliased rows) What is printed?

```
int[][] g = new int[2][2];  
int[] r = {1, 2};  
g[0] = r;  
g[1] = r;  
g[0][0] = 7;  
r = new int[]{9, 9};  
g[1][1] = g[0][0] + 1;  
System.out.println(Arrays.deepToString(g));  
System.out.println(Arrays.toString(r));
```

Line 1: _____ Line 2: _____

Answer key (teacher copy)

Every answer below was checked by compiling and running the code. Delete this section before copying the worksheet for students.

#	Answer	What it tests
1	14	Accumulator over an array
2	[2, 8, 6, 8, 8]	Index assignment; $a[4] = 2 + 6$
3	[0, 1, 4, 9, 16]; $4 + 16 = 20$	Filling with a loop
4	max = 9, updates = 2	Running max; only 8 and 9 beat the max
5	fish dog	Counting down by 2
6	[2, 3, 4, 5, 1]	Left rotation
7	[3, 2, 6, 4, 9]	Running sums use already-updated values
8	count = 3, total = -7	Enhanced for; evens $4+2+6 = 12$ added, odds $7+9+3 = 19$ subtracted
9	[2, 1, 4, 3, 5]	Pair swaps; the last element has no partner
10	[9, 2, 3]; [1, 4, 3]; true; false	Aliasing: b is a, c is a separate array
11	[3, 1, 3, 0, 2]	Counting smaller elements; equal values are not counted
12	data = [6, 7, 8]; k = 5	Elements change through the reference; reassigning arr and n has no effect outside
13	a = [4, 6, 8, 4]; b = [2, 4, 6, 8]	Method mutates its parameter; result is separate
14	c0:1 new:1 c1:2 c1:2	Two references to one object; cs[1] is reassigned, x still points to the old c1
15	[0, 6, 9], [2, 0, 8], [1, 4, 0]	Transpose, row swap by reference, then zero the diagonal
16	C=135 B=25 A=15 C=135	Array of objects; swap moves references; enhanced for mutates the objects
17	[4, 1, 1, 2, 3]	Repeated calls to a mutating method; a[0] uses the shifted values
18	[1, 3, 3, 1] 15	Jagged array (Pascal's triangle); row sums $1 + 2 + 4 + 8$
19	n = 4; [2, 3, 5, 2, 3, 5, 2]	Compact in place; the tail past n is left over from the original
20	[[7, 8], [7, 8]]; [9, 9]	Both rows are one array; reassigning r afterwards does not affect g

Notes on the trickiest ones

- **12:** arr starts as an alias of data, so the += n loop changes the caller's array. Then arr = new int[] {0, 0, 0} points the local variable elsewhere, so arr[0] = n touches only the new array.

- **13:** At `i = 1`, `a[0]` becomes 4, and so on. The last element of `a` is never overwritten, so it stays 4. Students often forget that `a` itself changed.
- **14:** After `cs[2] = x`, slots 1 and 2 and `x` all name the same object. It is incremented twice (`c1:2`). `cs[1] = new Counter("new")` only repoints slot 1.
- **15:** After the transpose, `g` is $\{\{1,4,7\},\{2,5,8\},\{3,6,9\}\}$. Swapping `g[0]` and `g[2]` swaps whole row references, giving $\{\{3,6,9\},\{2,5,8\},\{1,4,7\}\}$. Zeroing the diagonal gives the final answer.
- **16:** After `swap`, slot 0 holds C and slot 2 holds A. `bonus` adds 5 to every object, so C is 35, B is 25, A is 15. The new array inside `bonus` is thrown away. `top` still refers to C, so `top.add(100)` makes C 135, and it shows in slot 0 as well.
- **17:** One call gives $\{1,1,2,3,4\}$, two give $\{1,1,1,2,3\}$. Then `a[0] = 3 + 1 = 4`.
- **18:** The rows are $\{1\}$, $\{1,1\}$, $\{1,2,1\}$, $\{1,3,3,1\}$, so `total` is 15. Students who think every row has 4 columns get an out-of-bounds error.
- **19:** Only positions 0 to 3 are meaningful after the loop (`n = 4`). Positions 4 to 6 still hold the original 3, 5, 2, which is why the array ends in $\{\dots, 3, 5, 2\}$.
- **20:** After `g[0] = r`; `g[1] = r`; both rows are the same array. Writing 7 and then 8 changes that one array, giving $\{7, 8\}$. `r = new int[]{9, 9}` repoints only the variable `r`.