

## Solutions

Each answer shows the steps, then the **result**.

### 2.1

- down each column: **2 1 9 4 3 8 6 5 7**

### 2.2

```
int total = 0;
for (int[] row : g)
    for (int v : row)
        total += v;
```

- outer loop over rows with type `int[]`
- inner loop over each value with type `int`
- adds every value to `total`, which starts at 0

### 2.3

```
for (int r = 0; r < a.length; r++)
    System.out.println(a[r][a[0].length - 1]);
```

- loop over every row
- fixed column `a[0].length - 1`

### 2.4

- `g[0][2]`, `g[1][1]`, `g[2][0]`: **6, 3, 9**

### 3.1 A

- $4 - 1$ ,  $5 - 2$ ,  $6 - 3$ : each 3

### 3.2 B

- columns outside, rows inside, each with its own bound

### 3.3 C

- rows 1 to 3 (3 rows) times columns 0 to 3 (4 columns): 12

### 3.4

(a)

```
public static int studentTotal(int[][] marks, int s)
{
    int total = 0;
    for (int c = 0; c < marks[s].length; c++)
    {
        total += marks[s][c];
    }
    return total;
}
```

- loops over every column of row `s`
- adds `marks[s][c]` to a total that starts at 0
- returns the total

(b)

```
public static double testAverage(int[][] marks, int t)
{
    int total = 0;
    for (int r = 0; r < marks.length; r++)
    {
        total += marks[r][t];
    }
    return (double) total / marks.length;
}
```

- loops over every **row**
- adds `marks[r][t]` with the column fixed
- divides by the number of rows, `marks.length`
- casts to `double` to avoid integer division, and returns it