# Arrays

## Overview

In this lab, you will write an application that declares and manipulates arrays. You will also enhance previous lab code to use arrays where appropriate.

## Source folders

| | |
|---|---|
| Student project: | `StudentArrays` |
| Solution project: | `SolutionArrays` |

## Roadmap

There are 5 exercises in this lab, of which the last two are "if time permits". Here is a brief summary of the tasks you will perform in each exercise; more detailed instructions follow later:

1. Using arrays of primitives
2. Using arrays of objects
3. Using array initialization syntax.
4. Using arrays within a class
5. Additional suggestions

### Exercise 1: Using arrays of primitives

Open `ArrayOfPrimitives.java`, and write an application that creates and processes an array of `double` values.

Suggestions and requirements:

- Ask the user how many elements he/she would like in the array (use `Scanner` to get this number). Then create an array of this size.
- Write a for-loop that asks the user to enter values for each array element. Display prompt messages such as "Enter value for element 0:"
- Write a for-each loop to display each value in the array.
- Output the sum of all the positive values in the array, and indicate how many positive values there were.

### Exercise 2: Using arrays of objects

Open `ArrayOfObjects.java`, and write an application that creates and processes an array of `Product` objects (we've already written the `Product` class).

Suggestions and requirements:

- In `main()`, ask the user how many products he/she would like in the array, and then create a `Product-`array of this size
- Write a for-loop that populates the array with `Product` objects (each time round the loop, create a new `Product` object with an appropriate product name from the user).
- Write a for-each loop to display each product.

### Exercise 3: Using array initialization syntax

Open `DateValidation.java`, and take a look at the existing code. This is the solution code from the "Operators and Flow Control" lab.

Refactor the code so that it uses arrays rather than using `switch` statements, where appropriate. For example, you could use an array to hold the month names, and you could also use an array to identify the maximum number of days in a month.

### Exercise 4 (If time permits): Using arrays within a class

Open `Employee.java`, and take a look at the existing code. This is the solution code from the "Defining and Using Classes" lab.

Enhance the `Employee` class so that it holds an array of skills for each employee. For the sake of simplicity, each skill can be a simple `String`. You can also assume a fixed limit to the number of skills an employee can have, e.g. 10 ☺. Write methods to allow client code to add a skill and to display all the skills. Make sure you avoid array overflow!

Write some test code in `UseEmployee.java`.

### Exercise 5 (If time permits): Additional suggestions

Open `MultiDimArrays.java`, and write an application that performs matrix manipulation, using 2-dimensional arrays. Allow the user to specify the number of rows and columns and to enter data, and then output the sum and difference of the matrices.

Experiment with jagged multi-dimensional arrays (i.e. where the number of elements in each row can be different).