Arrays

An array is a way of creating any number of variables of a single type all at once.

```
int[] nums = new int[5];  // Creates 5 int variables
```

This allows indexing, which is often useful.

```
int i = 0;
while (in.hasNextInt()) {
    nums[i] = in.nextInt();
    i = i + 1;
}
```

Reference type

```
int[] nums = new int[5];  // Creates 5 int variables
foo(nums);
```

`foo` gets a reference (ie, memory location) for array.

```
public static void foo(int[] array) {
    for (int i=0; i<array.length; i++) {
        array[i] = 1;
    }
}
```

Sets all of the elements of num to 1.

Example: Scaling a sound wave

Given an array representing sound write a method `scale` that takes an array of intgers and a (double) scaling factor and adjusts the values by that scaling factor. If the factor is negative or greater than 100, throw an `IllegalArgumentException`.

```
public static void scale(int[] sound, double factor) {
    if (factor < 0.0  ||  factor > 100.0) {
        throw new IllegalArgumentException();
    }
    for (int i=0; i<sound.length; i++) {
        sound[i] = (int)(sound[i] * factor);
    }
}
```

2D arrays

Some data is naturally a two-dimensional grid: picture, matrix, game board, etc.

```
char[][] board = new char[3][3]; // Tic-Tac-Toe board
for (row=0; row<3; row++) {
    for (col=0; col<3; col++) {
        board[row][col] = ' ';  // Set each pos to space
    }
}
```

An array of arrays

A 2D array is actually an array or arrays.

```
int[][] twoD = new int[r][c];
```

`twoD[0]` refers to an array, the frist row.
`twoD[1]` refers to an array, the second row.
etc.

```
int rows = twoD.length;      // basename.length is number of rows
int[] firstRow = twoD[0];    // basename[0] is the first row
int cols = firstRow.length; // basename[0].length first row length
```

Example: B&W photo

A B&W photo can be stored as a 2D array of bytes (a byte is a type buit in to java that can be any integer -128..127), 0 for white, 127 for black.

Write a method `invert` that takes a 2D array of bytes and returns a negative image (ie, inverts it).

Pseudocode:

```
Make new array of same size as original
for each byte in original
    copy (127 - byte value) to same location in negative
return new array
```

Example: B&W photo

```
public static byte[][] invert(byte[][] original) {
    int rows = original.length;
    int cols = original[0].length;
    byte[][] negative = new byte[rows][cols];
    for (int r=0; r<rows; r++) {
        for (int c=0; c<cols; c++) {
            negative[r][c] = 127 - original[r][c];
        }
    }
    return negative;
}
```

Example: Warhol photo

Write a method that takes a 2D B&W photo and returns a photo twice as wide and twice as tall with the original photo duplicated to each quadrant of new photo.

Pseudocode:

```
Make new array of double size as original
for each byte in original
    copy byte value to the four locations in negative
return new array
```

Example: Warhol photo

```
public static byte[][] warhol(byte[][] original) {
    int rows = original.length;
    int cols = original[0].length;
    byte[][] result = new byte[2*rows][2*cols];
    for (int r=0; r<rows; r++) {
        for (int c=0; c<cols; c++) {
            result[r][c] = original[r][c];
            result[r+rows][c] = original[r][c];
            result[r][c+cols] = original[r][c];
            result[r+rows][c+cols] = original[r][c];
        }
    }
    return result;
}
```