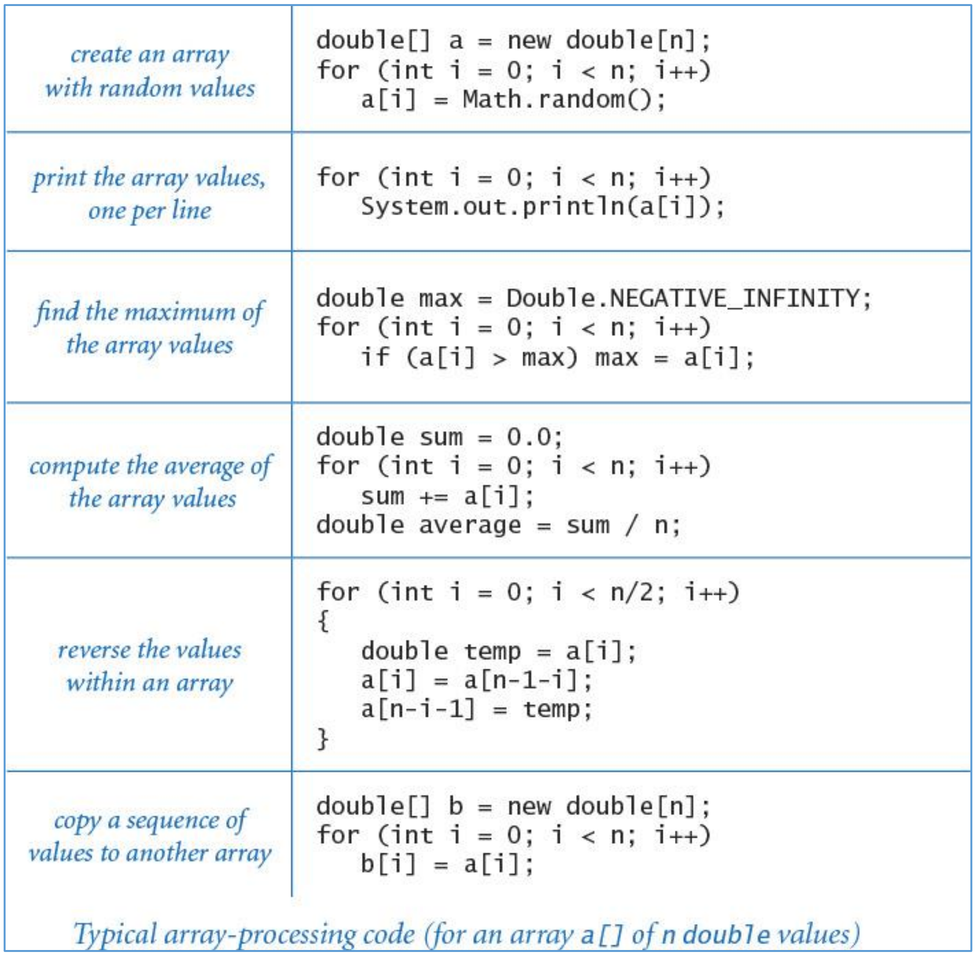
Summary on arrays:

https://ebookreading.net/view/book/EB9780134076461_11.html

- A *data structure* is a way to organize data in a computer.
- A *one-dimensional array* is a data structure that stores a *sequence* of values, all of the same type.
- We refer to the components of an array as its elements.
- We use *indexing* to refer to the array elements: If we have n elements in an array, we think of the elements as being numbered from 0 to n-1

```
int size = 5;

double[] a;                           // declare the array
a = new double[size];                  // create the array
for (int i = 0; i < size; i++)         // initialize the array
   a[i] = 0.0;
```

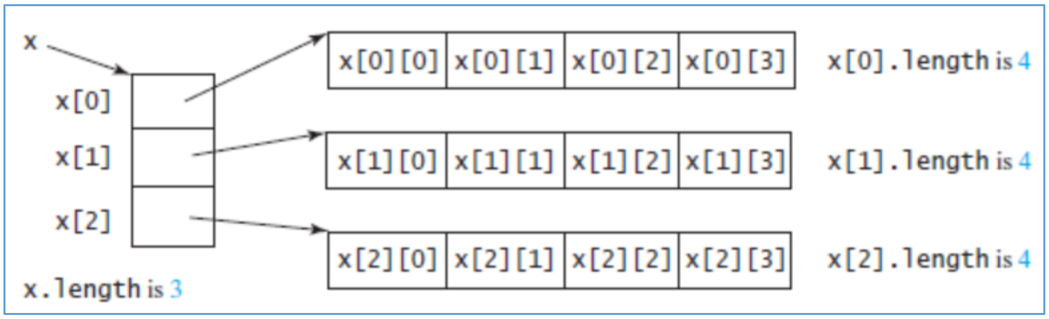


| | |
|---|---|
| *create an array with random values* | `double[] a = new double[n];`<br>`for (int i = 0; i < n; i++)`<br>`   a[i] = Math.random();` |
| *print the array values, one per line* | `for (int i = 0; i < n; i++)`<br>`   System.out.println(a[i]);` |
| *find the maximum of the array values* | `double max = Double.NEGATIVE_INFINITY;`<br>`for (int i = 0; i < n; i++)`<br>`   if (a[i] > max) max = a[i];` |
| *compute the average of the array values* | `double sum = 0.0;`<br>`for (int i = 0; i < n; i++)`<br>`   sum += a[i];`<br>`double average = sum / n;` |
| *reverse the values within an array* | `for (int i = 0; i < n/2; i++)`<br>`{`<br>`   double temp = a[i];`<br>`   a[i] = a[n-1-i];`<br>`   a[n-i-1] = temp;`<br>`}` |
| *copy a sequence of values to another array* | `double[] b = new double[n];`<br>`for (int i = 0; i < n; i++)`<br>`   b[i] = a[i];` |

*Typical array-processing code (for an array* `a[]` *of* `n` `double` *values)*

- A *two-dimensional array* is an array of one-dimensional arrays. Whereas the elements of a one-dimensional array are indexed by a single integer, the elements of a two-dimensional array are indexed by a pair of integers: the first index specifies the row, and the second index specifies the column.

Suppose int[][] x = new int[3][4], x[0], x[1], and x[2] are one-dimensional arrays and each contains four elements, as shown in the figure x.length is 3, and x[0].length, x[1].length, and x[2].length are 4


x
x[0]
x[1]
x[2]
x.length is 3
x[0][0] x[0][1] x[0][2] x[0][3]
x[1][0] x[1][1] x[1][2] x[1][3]
x[2][0] x[2][1] x[2][2] x[2][3]
x[0].length is 4
x[1].length is 4
x[2].length is 4

```
int[][] table = { {2, 4, 7, 15},  {3, 1, 8, 10}, {6, 0, 9, 12} };
```

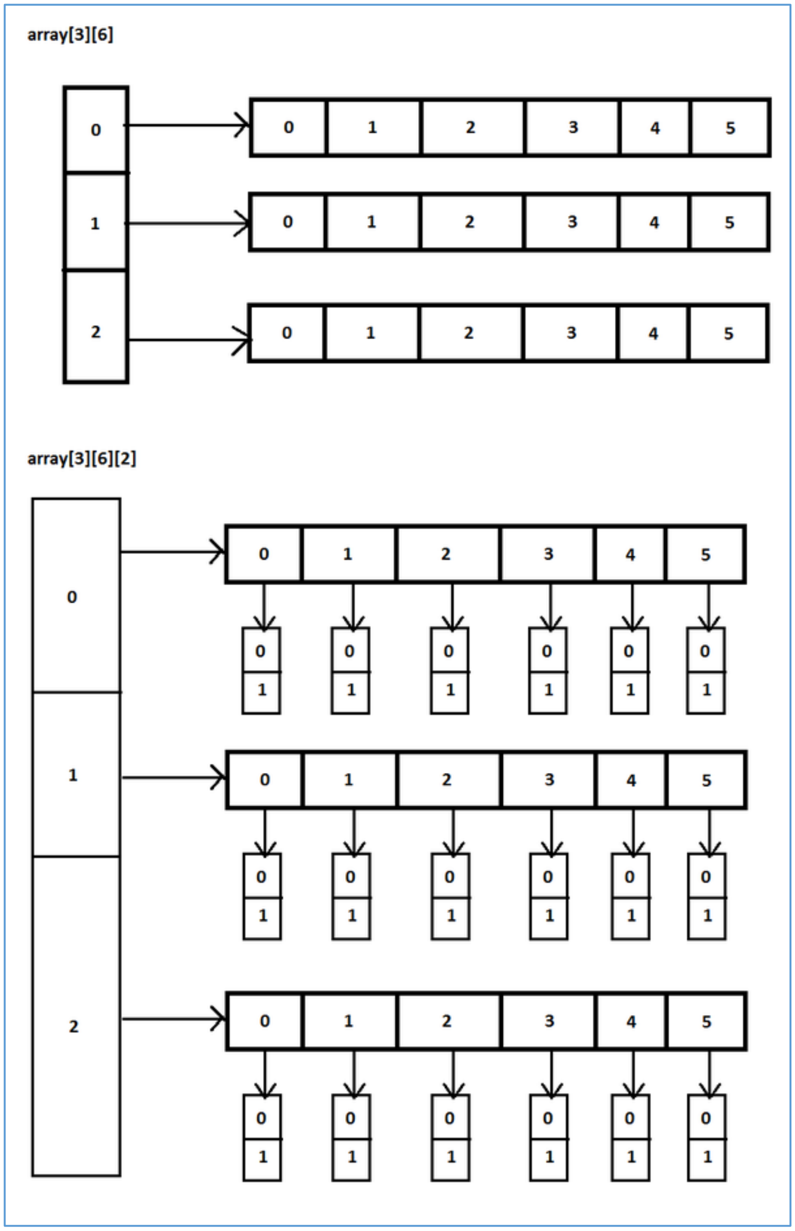
is the same as

```
table[0][0] = 2; table[0][1] = 4; table[0][2] = 7; table[0][3] = 15;
table[1][0] = 3; table[1][1] = 1; table[1][2] = 8; table[1][3] = 10;
table[2][0] = 6; table[2][1] = 0; table[2][2] = 9; table[2][3] = 12;
```

Below is the code to use nested for loop to print the elements of a 2D array.

```
for(int i=0;i<table.length;i++){
    for(int j=0;j<table[i].length;j++){
        System.out.print(table[i][j]+" ");
    }
    System.out.println();
}
```

- Multi-dimensional array


array[3][6]
0
1
2
0 1 2 3 4 5
0 1 2 3 4 5
0 1 2 3 4 5
array[3][6][2]
0
1
2
0 1 2 3 4 5
0 1 0 1 0 1 0 1 0 1 0 1
0 1 2 3 4 5
0 1 0 1 0 1 0 1 0 1 0 1
0 1 2 3 4 5
0 1 0 1 0 1 0 1 0 1 0 1