Arrays in Java – Understanding Arrays

Many problems involve the manipulation of large amounts of data – for example, lists and tables of data. For many such problems, an array provides an appropriate structure for storing and organizing the data.

Consider the following table, which contains price information for one litre of gasoline from 1980 to 2000:

Year	1980	1985	1990	1995	2000
Price ($)	0.27	0.51	0.59	0.56	0.72

To keep track of these data in a computer program, we might use variables: `price1980`, `price1985`, `price1990`, `price1995`, and `price2000`. Although valid, this strategy is cumbersome, and would become even more inconvenient if we wanted to add additional prices.

In contrast, Java offers a data structure called an *array*, which is a collection of data items (elements) of the *same type*. In this example, we could store our gasoline prices in an array of 5 *elements*. We might name the identifier for the array `price`, and each element of the array would be referenced using an *index*. In Java, every array index starts at zero (0). In this case, with 5 elements, the index starts at zero (0), and the fifth element is at an index value of 4.

Index →	0	1	2	3	4
`price`	0.27	0.51	0.59	0.56	0.72

The identifiers of each element in the array are `price[0]`, `price[1]`, and so on.

In Java, an array is a type of object. To create an array, we follow the same steps used for creating any other object, and we must consider, and avoid, the same possible sources of errors.

To create an array to hold the gasoline price data, we could declare

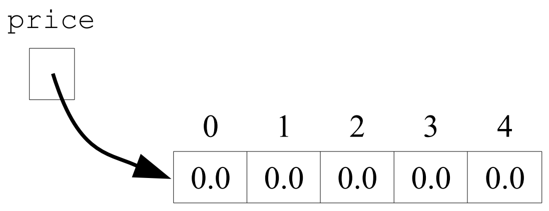
```
    double[] price;
```

This creates the variable `price` and determines that `price` is a reference to an array of `double` values. As with other objects, however, this does not actually create the array. It simply creates the variable that can act as a *reference* to the array once it has been created.

To actually create the array in memory, we use the `new` keyword:

```
double[] price;
price = new double[5];
```

which creates the following (a representation of an array in the computer memory).



Notice the value 0.0 in each element of the array. Numeric arrays are automatically initialized to zero when space is allocated to them. For `char` values, they are initialized to the *null character*. For `boolean` arrays, they are initialized to `false`.

This same declaration, creation, and initialization can be expressed as a single command:

```
double[] price = new double[5];
```

Using the Array

Now that the array has been created in memory, we need to put it to use. We can put data into the array, or read the data that is contained in the array. The simplest way to do this is one element at a time:

```
double[] price = new double[5];
price[0] = 0.27;
price[1] = 0.51;
price[2] = 0.59;
price[3] = 0.56;
price[4] = 0.72;
```

Notice that the **index of the array**, the number in the square brackets, started at 0 and ended at 4, for a total of 5 elements, which matches the length of our array.

Each piece of the array represents a single data item (in this case, a `double` value), just like a variable. We can also use this data for calculations or output, just as we would any other variable.

```
System.out.println("The price of gas in 1985 was " + price[1]);
```

Length of an Array

Once an array has been created in memory, its size is fixed for the duration of its existence. Every array object has a `length` field whose value can be obtained by appending `.length` to the array identifier.

```
int len = price.length;
```

Since all arrays start numbering at zero, the length is always one greater than the highest index.

Explicit Array Declaration and Initialization

Java has another form of array declaration that allows us to initialize the array with any values desired at the time of declaration.

```
int[] primes = {2, 3, 5, 7, 11, 13, 17, 19, 23, 29};
```

Notice that we have not used the keyword `new`, and we have used brace brackets instead of square brackets. The size of the array is not stated, but it is understood to match the number of elements included in the array declaration. In this case, the array has 10 elements, or a length of 10.

Arrays in Java – Understanding Arrays

Exercises

1. An array to store marks for twenty students has been declared as follows:

```
int[] marks = new int[20];
```

   a) What is the array identifier?
   b) What is the identifier of the first element in the array?
   c) What value is stored in each array element, or position, by the declaration?
   d) What is the value of `marks.length`?
   e) What are the indices of the array? (i.e., the lowest and highest positions in the array)

2. Write declarations to create arrays that would be appropriate for storing the indicated data.
   a) the number of votes cast for five candidates in an election
   b) the answers to a twenty-question true/false quiz
   c) the average family size in the years 1900, 1910, ..., 2000, 2010

3. a) Write a statement that creates and initializes an array `terms` of `double` values (i.e., decimal values) to store the terms of the sequence

$$\frac{1}{2}, \frac{2}{3}, \frac{3}{4}, \frac{4}{5}, \frac{5}{6}, \frac{6}{7}$$

   b) What is the value of `terms.length`?

4. The table gives atomic masses of the eight lightest elements listed according to atomic number.

Atomic Number	1	2	3	4	5	6	7	8
Atomic Mass	1.0	4.0	6.9	9.0	10.8	12.0	14.0	16.0

Suppose that the data in this table were to be represented by the array `mass` declared by the statement

```
double[] mass = {1, 4, 6.9, 9, 10.8, 12, 14, 16}
```

   a) What is the value of `mass[2]`?
   b) What is the value of `mass[5]`?
   c) What are the possible values of the indices of the array?
   d) What is the identifier of the element whose value is `6.9`?
   e) Of what type are the elements?
   f) What is the value of `mass.length`?

Solutions

1. An array to store marks for twenty students has been declared as follows:

```
int[] marks = new int[20];
```

a) What is the array identifier?

The identifier is the same as the name of the array: `marks`

b) What is the identifier of the first element in the array?

To identify the first element of the array, we need to include the name of the array and the position of the first element: `marks[0]`

c) What value is stored in each element by the declaration?

Each element holds an integer value.

d) What is the value of `marks.length`?

The length method will return the number of elements in the array, which is 20.

e) What are the indices of the array?

The lowest index is 0, and with 20 elements, the highest must be 19.

2. Write declarations to create arrays that would be appropriate for storing the indicated data.

a) the number of votes cast for five candidates in an election

```
int[] votes = new int[5];
```

b) the answers to a twenty-question true/false quiz

```
boolean[] quiz = new boolean[20];
```

c) the average family size in the years 1900, 1910, ..., 2000, 2010

There are 12 elements in this array:
1900, 1910, 1920, 1930, 1940, 1950, 1960, 1970, 1980, 1990, 2000, 2010

```
int[] familySize = new int[12];
```

3. a) Write a statement that creates and initializes an array `terms` of `double` values to store the terms of the sequence

$$t_1=\frac{1}{2}, t_2=\frac{2}{3,} \ldots, t_6=\frac{6}{7}$$

This array has 6 elements (which you can tell from looking at the numerators, which go from 1 to 6).

```
double[] terms = {1.0/2, 2.0/3, 3.0/4, 4.0/5, 5.0/6, 6.0/7}
```

Need to do division with a float to avoid integer division.

b) What is the value of `terms.length`?

The value is 6 (there are 6 elements).

4. The table gives atomic masses of the eight lightest elements listed according to atomic number.

Atomic Number	1	2	3	4	5	6	7	8
Atomic Mass	1.0	4.0	6.9	9.0	10.8	12.0	14.0	16.0

Suppose that the data in this table were to be represented by the array `mass` declared by the statement

```
double[] mass = {1, 4, 6.9, 9, 10.8, 12, 14, 16}
```

a) What is the value of `mass[2]`?

Since the array numbering starts from 0, `mass[2]` is actually the 3rd element. Therefore, `mass[2]` is 6.9.

b) What is the value of `mass[5]`?

`mass[5]` is the 6th element, which has a value of 12.0.

c) What are the possible values of the indices of the array? **From 0 to 7.**

d) What is the identifier of the element whose value is `6.9`?

The value 6.9 is the 3rd element of the array, which means it has an index of 2. The identifier is `mass[2]`.

e) Of what type are the elements? **These elements are of type `double`.**

f) What is the value of `mass.length`?

There are 8 elements, so `mass.length` will return a value of 8.