**Java Arrays**

An *array* is a container object that holds a fixed number of values of a single type. The length of an array is established when the array is created. After creation, its length is fixed.

An **element** is the value stored in a cell of an array.

**Why u****sing Arrays**

**Here is an example to help explain why arrays are utilized to store values:**

*Suppose we want to store the 50 US States in our Java program. We can create 50 string variables and store 1 state name in each variable, as below:*

*String strState1 = “Montana”, strState2 = “Idaho”, … , strState50 = “Wyoming”;*

**or**

*We can make a state array that can store all 50 state names in a single container.*

*String[] statesArray = {“Montana”, “Idaho”, … , “Wyoming};*

QUESTION 1:

If you are writing a program that reads in 100 numbers, why is an array a good choice?

**Declaring an array:**

***ArrayDataType[] ArrayName;***

ArrayDataType = data type of array element (int, double, etc.)

ArrayName = name of the array

**Creating/** **Instantiating** **an array:**

***int arrayName = new int[10];***

QUESTION 2:

int[] numberArray = new int[5];

1. What is the length of the array numberArray?
2. What are the indexes of numberArray?

**Assigning values to arrays:**

***arrayName[0] = 10;*** this make the first element of the array 10

***arrayName[1] = 12;*** this make the second element of the array 5

**Assigning values when you create the array:**

***int[] numberArray = {1, 2, 3, 4, 5};***this array is five elements in length

\*You will use numberArray and the values that were assigned above to answer the rest of the questions.

QUESTION 3:

What value is in numberArray[2] ?

An expression such as numberArray[3] is called a **subscripted variable**. A subscripted variable can be used anywhere an ordinary variable of the same type can be used. Since numberArray[3] contains an int it can be used anywhere an int variable may be used.

An arithmetic expression can contain a mix of literals, variables, and subscripted variables. For example, if x contains a 10, then (x + numberArray[2]) / 4 evaluates to (10+14) / 4 == 6.

Here are some other legal statements:

numberArray[0] = (x + numberArray[2]) / 4 ;

numberArray[2] = numberArray[2] + 1;

// increment the numberArray in cell 3

x = numberArray[3]++ ;

numberArray[4] = numberArray[1] / numberArray[3];

QUESTION 4:

What is the value of the arithmetic expression:

numberArray[2] + numberArray[4]

QUESTION 5:

What will be the result of executing the statement:

numberArray[0] = numberArray[2] + 8;

**Bounds Checking**

The **length** of an array is how many cells it has. An array of length N has cells indexed

0 through (N-1).

Indexes must be of **integer** type.

It is *not legal* to refer to a cell that does not exist.

As a Java program is running, each time an array index is used it is checked to be sure that it is OK. This is called **bounds checking**, and is extremely important for catching errors. If an executing program refers to a cell that does not exist, an ArrayIndexOutOfBoundsException is thrown, and (usually) the program is terminated.

If an array is declared: ***int[] data = new int[10];*** the table shows some legal and illegal subscripted variables of the array:

data[ -1 ] always illegal

data[ 10 ]   illegal (given the above declaration)

data[ 1.5 ] always illegal

data[ 0 ] always OK

data[ 9 ] OK (given the above declaration)

data[ j ] can't tell (depends on the value of j)

QUESTION 6:

Here is a declaration of another array:

double[] scores = new double[25];

Which of the following are legal?

Top of Form

scores[ 0 ] scores[1] scores[ -1 ]

scores[ 10] scores[ 25 ] scores[ 24 ]

scores[ 1.2] scores [2] scores [ 4 ]

class ArrayExample

{

public static void main ( String[] args )

{

int[] stuff = new int[5];

stuff[0] = 23;

stuff[1] = 38;

stuff[2] = 7\*2;

System.out.println("stuff[0] has " + stuff[0] );

System.out.println("stuff[1] has " + stuff[1] );

System.out.println("stuff[2] has " + stuff[2] );

System.out.println("stuff[3] has " + stuff[3] );

System.out.println("stuff[4] has " + stuff[4] );

}

}

QUESTION 7:

Type in the program above – ArrayExample. What does the program output?

**Using a Variable as an Index**

The index of an array is always an integer type. It can be a literal, or an expression that evaluates to an integer. For example, the following are legal:

int values[] = new int[7];

int index;

values[ 6 ] = 42; // literal

index = 0;

values[ index ] = 71; // put 71 into cell 0

index = 5;

values[ index ] = 23; // put 23 into cell 5

index = 3;

values[ 2+2 ] = values[ index-3 ]; // same as values[ 4 ] = values[ 0 ];

QUESTION 8:

Given the above declarations, is the following legal?

index = 4;

values[index+2 ] = values[ index-1 ];

**Several Arrays per Program**

A program can use any number of arrays. Often values are copied back and forth between the various arrays. Here is an example program that uses two arrays:

Top of Form

class ArrayExample2

{

public static void main ( String[] args )

{

int[] valA = { 12, 23, 45, 56 };

int[] valB = new int[4];

 =  ;

 =  ;

 =  ;

 =  ;

}

}

Bottom of Form

QUESTION 9:

Fill in the blanks above so that the values in valA are copied into the corresponding cells of valB.

QUESTION 10:

1. Before typing in the following program, what do you think the output is going to be?
2. After typing and running the program, what is the output?

public class TestArray {

public static void main(String[] args) {

double[] myList = {1.9, 2.9, 3.4, 3.5};

// Print all the array elements

for (int i = 0; i < myList.length; i++) {

System.out.println(myList[i] + " ");

}

// Summing all elements

double total = 0;

for (int i = 0; i < myList.length; i++) {

total += myList[i];

}

System.out.println("Total is " + total);

// Finding the largest element

double max = myList[0];

for (int i = 1; i < myList.length; i++) {

if (myList[i] > max) max = myList[i];

}

System.out.println("Max is " + max);

}

}

QUESTION 11:

Complete the assignment statement so that it computes the sum of all the numbers in the array.

public class Exercise1

{

public static void main ( String[] args )

{

int[] val = {0, 1, 2, 3};

sum =

System.out.println( "Sum of all numbers = " + sum );

}

}

**A Review of Arrays**

1. [An array](http://chortle.ccsu.edu/Java5/Notes/chap46/ch46_2.html#array)is an object that has room for several values, all of the same type.
2. Each value is stored in a cell of the array. The values stored in an array are called the elements of the array.
3. If there are N cells in the array, the cells are indexed from 0 up to (N-1).
4. The index must be an integer value (byte, short, or int).
5. [An array declaration](http://chortle.ccsu.edu/Java5/Notes/chap46/ch46_5.html#array,_declaration)looks like:

int[] intArray;

This declaration declares the array reference intArray. It does not create the actual object.

1. An array can be declared and constructed in a combined statement:

int[] intArray = new int[17];

This declaration declares the array reference intArray, and constructs an array object containing 17 cells that can hold int.

1. When an array object is constructed using the new operator, the cells are [initialized](http://chortle.ccsu.edu/Java5/Notes/chap46/ch46_7.html#array,_default_initialization)to the default value of the type of the cell. Numeric types are initialized to zero.
2. Once an array object has been constructed, the number of cells it has can not be changed. (However, a completely new array object, with a different number of cells, can be constructed to replace the first array object.)
3. [A subscripted variable](http://chortle.ccsu.edu/Java5/Notes/chap46/ch46_4.html#subscripted_variable) such as intArray[12] can be used anywhere an ordinary variable of the same type can be used.
4. The index used with an array can be stored in a variable, for example

int j = 5 ;

intArray[ j ] = 24; // same as: intArray[ 5 ] = 24

1. The index used with an array can be computed in an expression, for example

int j = 5 ;

intArray[ j\*2 + 3 ] = 24; // same as: intArray[ 13 ] = 24

1. [The index](http://chortle.ccsu.edu/Java5/Notes/chap46/ch46_6.html#array,_bounds_checking)used with an array must be within the range 0..(N-1) where N is is number of cells of the array.
2. If an index that is out of bounds is used with an array, an exception is thrown and the program stops running (unless it catches the exception.)
3. An array can be declared, constructed, and initialized using an [initializer list.](http://chortle.ccsu.edu/Java5/Notes/chap46/ch46_11.html#array,_initializer_list)This can only be done when the array is first declared.

double[] gpaArray = {4.0, 3.5, 1.2, 3.23, 2.45};