Tutorial 7
CSC 201

Java Programming Concepts

مبادئ البرمجة باستخدام الجافا

Chapter 7: Arrays

What is an Array?

An array is a set or collection of values, where each value is identified by an index. You can make an array of ints, floats, or any other data type. All the values of an array have to be the same type.

Declaring and Initializing Arrays

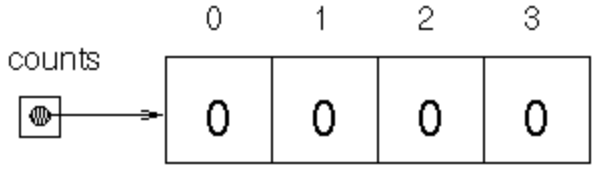
For example int[ ] is the type "array of integers" and
double[ ] is the type "array of doubles"

You can declare variables with these types in the usual ways:

```
int [ ] counts;
double [ ] values;
```

Until you initialize these variables, they are set to null. To actually create the array, we have to use the new command:

```
counts = new int [4];
values = new double[size];
```



When you first allocate a new array, the elements are initialized with a default value, depending on the data type of the array.

Arrays:

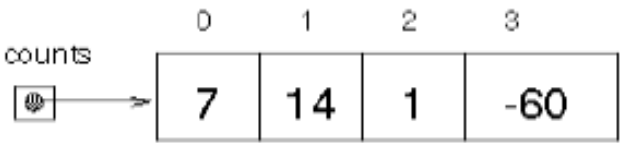
Accessing Elements of an Array

To access a certain element in an array, we use the [ ] operator to indicate which element in the array we want to access depending on its position

Take a look at the following statements, performed on the array counts:

```
counts[0] = 7;
counts[1] = counts[0] * 2;
counts[2]++;
counts[3] -= 60;
```

All of these are legal assignment statements. Here is the effect of this code fragment:



By now you should have noticed that the four elements of this array are numbered from 0 to 3,

You can use any expression as an index, as long as it has type int. One of the most common ways to index an array is with a loop variable.

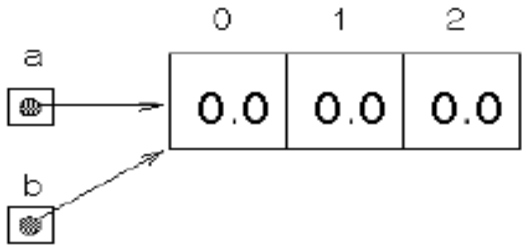
For example:

```
int i= 0;
while (i< 4)
{
  System.out.println(counts[i]);
  i++;
 }
```

This is a standard while loop that counts from 0 up to 4, and when the loop variable i is 4, the condition fails and the loop terminates. Thus, the body of the loop is only executed when i is 0,1, 2 and 3. Each time through the loop we use i as an index into the array, printing the ith element. This type of array traversal is very common.

Copying Arrays

When you copy an array variable, remember that you are copying a reference to the array.



For example:

```
double[ ] a = new double[3];
double[ ] b = a;
```

This code creates one array of three doubles, and sets two different variables to refer to it.

you can make a copy of the array, by allocating a new array and copying each element from one to the other.

```
double[ ] b = new double [3];
int i= 0;
while (i< 3)
{
    b[i] = a[i];
    i++;
}
```

Using while loop

```
double[ ] b = new double [3];
for (int i=0; i<3; i++)
    b[i] = a[i];
```

Using for loop

Array Length

The array object has one instance variable: length. This variable is of type int and it contains the length of the array (number of elements).
It is a good idea to use this value as the upper bound of a loop, rather than a constant value.

```
for (int i = 0; i < a.length; i++)
{
  b[i] = a[i];
 }
```

The last time the body of the loop gets executed, i is a.length - 1, which is the index of the last element.

Basic Ideas of Array Processing

In many cases, processing an array means applying the same operation to each item in the array.

A loop for processing all the items in an array A has the form:

```
// do any necessary initialization
for (int i = 0; i < A.length; i++)
{
  . . . // process A[i]
}
```

for example that A is an array of type double[]. Suppose that the goal is to add up all numbers in the array. An informal algorithm for doing this would be:

```
Start with 0;
Add A[0];   (process the first item in A)
Add A[1];   (process the second item in A)

Add A[ A.length - 1 ];   (process the last item in A)
```

```
double  sum = 0;  // The sum of the numbers in A.
for (int i = 0; i < A.length; i++)
  sum += A[i];  // add A[i] to the sum, for  i = 0, 1, ..., A.length - 1
```

Example1:

The following loop that counts the number of items in the array A, which are less than zero:

```
int count; // For counting the items
count = 0; // Start with 0 items counted
for (int i = 0; i < A.length; i++)
{
   if (A[i] < 0.0) // if this item is less than zero...
       count++;  // ...then count it
}
//At this point, the value of count is the //number of of items that have passed the test of being < 0
```

Example2:

Another typical problem is to find the largest number in A.

```
double max = A[0];
        for (int i = 1; i < A.length; i++)
        {
          if (A[i] > max)
            max = A[i];
        }           // at this point, max is the largest item in A
```

1) Searching an Array

This method of searching an array by looking at each item in turn is called linear search.

If the integer is found, the method returns the index of the location in the array where it is found. If the integer is not in the array, the method returns the value -1 as a signal that the integer could not be found:

```
static int find(int[ ] A, int x)
{
   for (int index = 0; index < A.length; index++)
     {
        if ( A[index] == N )
           return index;        // N has been found at this index!
     }
    // If we get this far, then N has not been found anywhere in the array.
   return -1;      // Return a value of -1.
}
```

2) Sorting an Array

A typical sorting method uses the idea of finding the biggest item in the list and moving it to the end.

```
static void selectionSort(int[] A)
{
// Sort A into increasing order, using selection sort
 for (int lastA= A.length-1; lastA> 0; lastA--)
 {
   int maxLoc= 0; // Location of largest item seen so far.
   for (int j = 1; j<=lastA; j++)
   {  if (A[j] > A[maxLoc])
            maxLoc= j;
   }
   int temp = A[maxLoc]; // Swap largest item with item in lastPlace
   A[maxLoc] = A[lastA];
   A[lastA] = temp;
 } // end of for loop
}
```