1) **Arrays in Java**

To use an array in a program, you need to declare, create and initialize it. For one-dimensional arrays, declaration involves specifying the type of array items and the name of the array; creation involves using the 'new' operator and specifying the array size; and initialization invovles assigning values to each array element.

Example:

```
int N = 100; //array size
double[] a = new double[N]; //declaring and creating array
//initializing array
for(int i=0; i<N; i++)
        a[i] = i;
```

Notes:

a) Arrays must be explicitly created at run time because the Java compiler has no way of knowing how much space to reserve for the array at compile time, as it does for primitive types. If the array contents are known at compile time, instead, then the array can be allocated as follows:

```
        int[] a = {1, 2, 3, 4, 5};
```

b) Java does automatic bounds checking: given an array of size N, any attempt at using an index whose value is less than 0 or greater than N-1 leads to ArrayOutOfBoundsException runtime exception.

c) An array name refers to the whole array: given two arrays, assigning one array name to another causes both array names to refer to the same array:

```
        int[] a = new int[N]; //one array named a
        ...
        a[i] = 1234; //for some defined variable i
        ...
        int[] b = a; //b refers to the array pointed to by a (one array, two names)
        ...
        b[i] = 5678; //this overwrites 1234 with 5678 because a and b are reference same array.
```

This phenomenon is called aliasing. Aliasing can cause bugs. If the intention is to copy a, then a safer approach is to create a new array and make b refer to it. Then use a loop construct to copy the contents of a to b.