Arrays

Definition. An array is....

(1) Several examples of how to **declare an array**.
- important: You must tell it the size of the array when you declare it.

```
datatype[] arrayName = new datatype[ length ] ;
```

```
String[] names = new String[100];
```

```
int[] numbers = new int[23];
```

q. Declare an array to hold the masses of 100 particles.

(2) How to **assign values** to an array.
- important: *Arrays start indexing at 0*

```
numbers[0] = 23;
numbers[1] = 10;
numbers[23] = 10;       // this is an ArrayIndexOutOfBounds exception!

names[2] = "phil";
names[0] = "bob";
```

(3) How to **read values from an array** (e.g. put them in an if-statement or a print-statement).

```
if (names[2].equals("bill")) {
   System.out.println("The third name on the list is bill!");
}

System.out.println( numbers[0] + " " + numbers[1] );
```

(4) **Length of the array.**

array_name.length gives the length of the array. NOT the last index in the array.

```
System.out.println( numbers.length );              // this will display 23

      // this is an array out of bounds error.
System.out.println( numbers[ numbers.length ] );
```

(5) How to **loop through all elements of an array** to assign them random values.

```
Random generator = new Random();
for (int i = 0; i < numbers.length; i++) {
    a = generator.nextInt(100);
    numbers[ i ] = a;
}
```

Q: What would *this* do?

```
Random generator = new Random();
for (int i = 0; i < numbers.length; i++) {
    a = generator.nextInt(23);
    numbers[ a ] = i;
}
```

(6) Displaying an array.

```
System.out.print( names );          // what does this do?

// this is proper.
for (int i = 0; i < names.length; i++) {
        System.out.print( names[i] + " ");
}
```

(7) How to loop through all elements of an array to check each one for something (e.g. is it even).

Assume *numbers* is full of random numbers.

```
int counter = 0;

for (int i = 0; i < numbers.length; i++) {
    if (numbers[i] % 2 == 0) {
        counter++;
    }
}

System.out.println("There were " + counter + " even numbers in the list!");
```

(8) Passing arrays as parameters and return-values

```
public static int sumOfValues( int[] myArray ) {
        // code here
}
```

Note: Arrays pass by **reference**, not by value!

```
// this method doesn't actually change anything.
public static void m1(int a) {
      a = 10;
}

// this method sets the first element of a to be 10.
public static void m2(int[] a) {
      a[0] = 10;
)

public static void main(String[] args) {

      int b = 3;
      int[] c = {1, 2, 3}

      m1(b);
      m1(c);

}


public static int[] getUserInputs(int n) {
        int[] a = new int[n];
        for (int i = 0; i < n; i++) {
                // get user input and assign to a[i]
        }

        return a
}
```

(7) Split a String into an array of letters, do something to it, and then put them back together into a String. What does this code do?

```
String word = "bird";
String[] letters = word.split("");

String t;
for (int i = 0; i < letters.length; i += 2) {
    t = letters[i];
    letters[i] = letters[i+1];
    letters[i+1] = t
}

word = Array.toString( letters );
```

Array Short exercises

(1)
(a) Write a program that declares an array of 100 integers and fills it with random values (range up to you).

(b) Write a method called displayArray which takes an int array as a parameter and loops through it to display every # in it on a single line, separated by commas.

(c) Write a method called sumArray which takes an int array as a parameter and returns the sum of all the values in it.

(2)
(a) Declare an array of 5 Strings.
(b) Create a loop that will loop 5 times, and ask the user to type 5 words. Store each word in the array in order.
(c) Loop through the array *backwards* to display the words in reverse order.

(3) Implement a method called letterShuffle(String word) that takes a String as input, shuffles all the letters and returns a new String as output. Note: return a STRING as output, not a String array of the letters.

(4) Implement the following methods:

(a) a method called arrayCopy which takes an array as an argument and returns a new array whose values are an exact copy of the argument array.
(b) maxValue which takes an int array as an argument and returns the max value in it.
(c) minValue which takes an int array and returns the min value in it.
(d) meanValue which takes an int array and returns the mean value in it.
(e) a method called isSorted which takes an int array as a parameter and returns true if the array's elements are in sorted order from least to greatest.
*(f) a method called sort which takes an int array as an argument and sorts the elements in it.

(5) Implement a method called rotate(String plaintext, int n) that takes a string and an int as parameters. It should return a String that is the argument string rotated to the left by n positions. For example,

```
String n = rotate("hello there", 3);      // n is "lo there hel"
String m = rotate("hello there", 6);      // m is "there hello"
```

hint: don't be afraid to write a helper method that does a simpler version of this same problem.