1 D, 2 D Arrays and Array Lists

Array: Collection of related values, sequence of values of the same type.

- The values in an array called elements.
- Declaring arrays
  1- 1D array: dataType [] array = new dataType[x]; x : the number of the array elements
  Or dataType [] array = {1,2,3,4};
  2- 2D array: dataType [][] array = new dataType[rows][cols];
  Or dataType [][] array = {{1,2,3,4},{2,3,5}};
  # the [] defines the number of dimensions of an array.
- All elements are initialized to zeros in the case of int's or double's.
- All elements are initialized to false in the case of Boolean.
- All elements are initialized to null in the case of objects.
- Arrays are passed by reference.
- We can get the number of the array elements by calling array.length. Notice: length not length().
- The index of the last element in the array is array.length-1 because the first one is 0.
- To call an element in the array use array[index of the element]
- Calling array[array.length] will cause an array out of bounds error.
- Arrays length is fixed. Once it's initialized it can't be changed.
- Double[] arr; this is only a reference. To construct an actual array use the new operator.
- Use an array of objects instead of processing parallel arrays.

Array Lists

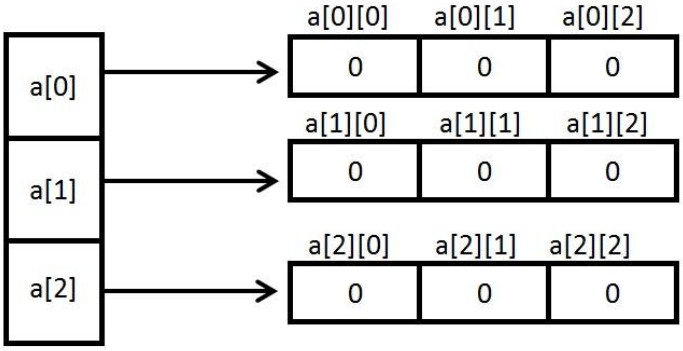
- Advantages:
  1- It can grow and shrink.
  2- ArrayList class supply methods for most of the common tasks such as (add(), remove(),...).
- To declare an array list : ArrayList<DataType> arrlist = new ArrayList< DataType>();
- Array list only accepts objects as a data type.
- Once it is constructed the size of an array list is 0;
- Array lists uses the get(index) operator to get an element not the [index];
- The last index in an array list is size()-1 because the first index is 0.
- To add an element to the end of the array list use arrlist.add(object);
- To add an element in a specified index use arrlist.add(index, object);
- To change the value of an element use arrlist.set(index,object);
- To remove an element use remove(index); to reset an array list use arrlist.clear();
- In order to use int's or double's in an array list use the wrapper classes.
  e.g. use Integer not int, Double not double and so on.
- Wrapper classes contain a value of their primitive type.
- Wrapper classes uses (auto boxing or auto wrapping) which is simply declaring them like
  Double a = 22.1 instead of using Double a = new Double(22.1);
- Array lists uses the partially filled arrays which is simply using arrays of larger number than expected and if it was filled it will automatically copy them into an even bigger one.

Enhanced for loop (for each loop)

- For (dataType element : array)
  { System.out.print(element);} simply it goes through the array elements one by one
- Disadvantage : the for each loop doesn't allow you to modify the elements.

Example of a diagram that shows how multi-dimensional arrays work:-

int [][] a = new int [3][3];



We can notice that the every row is actually pointing to an array. How could you use this to write nested enhanced for loops for 2D arrays?

Java 7 enhancements

- ArrayList<DataType> arrlist = new ArrayList<>(); no need to rewrite the DataType
- It is possible to use [elements] instead of get(elements) with the array lists.
- We can construct an array list out of an array of the same type
  ArrayList<String> a = new ArrayList<>(["C","S","S","E"]);