



Arraylist and Array

used to store multiple variables
with same data types

Imagine that you need to set 100 different grade and you need to keep these 100 values in the memory.

Without an array, you would have to create 10 separate variables with unique names, and then write your code to work with each one explicitly. This would be cumbersome and error prone.

How to Use Them ?

Declaring Variable

Array

```
<base-type> [] <array-variable> = new <base-type> [  
<array-length>];
```

Example

```
String[] bList = new String[1000];  
int[] yay = new int[5];  
double[] p = new double[100];
```

Arraylist

```
ArrayList<object-type> <arrayList-variable> = new  
ArrayList<object-type>();
```

Example

```
ArrayList<String> bList = new ArrayList<String>();  
ArrayList<Integer> yay = new ArrayList<Integer>();  
ArrayList<Double> p = new ArrayList<Double>();
```

Accessing Elements

Array

```
<array-variable> [index of array];
```

Example

```
bList[2];
```

Arraylist

```
<arrayList-variable>.get(index of arrayList);
```

Example

```
bList.get(2);
```

PROS

Array

Easier access to any element using the index
Easy to manipulate and store large data

Arraylist

Resizeable
Can add and remove elements at particular position

CONS

Array

Fixed size. Can not be increased or decrease once declared
Can store a single type of primitives only

Arraylist

Bigger memory use
In the worst case, need more time to remove or add an element

They can do more!

Array

copyOf(originalArray, newLength): This method copies the specified array, truncating or padding with the default value (if necessary) so the copy has the specified length

copyOfRange(originalArray, fromIndex, endIndex): This method copies the specified range of the specified array into a new Arrays

equals(array1, array2): Checks if both the arrays are equal or not

sort(originalArray): Sorts the complete array in ascending order.

toString(originalArray): Returns a String representation of the contents of this Arrays

Arraylist

contains(Object o): Returns true if list contains the specified element

remove(int index): Removes the element at specified position in list

set(int index, E element): Replaces the element at the specified position in this list with the specified element

size(): Returns the number of elements in this list

isEmpty(): Returns true if this list contains no elements