**NSArray:**

NSObject

NSArray

NSArray is Objective-C’s general-purpose array type. It represents an ordered collection of objects.  NSArray creates static arrays. Immutable arrays can be defined as literals using the @[] syntax.

NSArray and its subclass [NSMutableArray](https://developer.apple.com/library/mac/documentation/Cocoa/Reference/Foundation/Classes/NSMutableArray_Class/index.html" \l "//apple_ref/occ/cl/NSMutableArray) manage ordered collections of objects called **arrays**. NSArraycreates static arrays, and NSMutableArray creates dynamic arrays. You can use arrays when you need an ordered collection of objects.

NSArray is “toll-free bridged” with its Core Foundation counterpart, [CFArrayRef](https://developer.apple.com/library/mac/documentation/CoreFoundation/Reference/CFArrayRef/index.html" \l "//apple_ref/c/tdef/CFArrayRef). See [Toll-Free Bridging](https://developer.apple.com/library/mac/documentation/General/Conceptual/CocoaEncyclopedia/Toll-FreeBridgin/Toll-FreeBridgin.html#//apple_ref/doc/uid/TP40010810-CH2) for more information on toll-free bridging.

Eg:

NSArray \*fruits = @ [@”mango”,@”apple”,@”orange”];

NSArray \*colors = @[NSArray arraywithobjects:@”yellow”,@”blue”@”red”];

NSLog(@”The fruits are %@”,fruits[0]);

NSLog(@”The colors are:%@”,[colors objectAtIndex:0]);

* **Comparing arrays:**

The arrays can be compared to check the equality of two different arrays. The array object use isEqualToArray method to compare the two arrays. If the two arrays elements are equal then it returns YES.

Eg:

NSArray \*fruits = @ [@”mango”,@”apple”,@”orange”];

NSArray \*colors = @[NSArray arraywithobjects:@”yellow”,@”blue”@”red”];

If([fruits isEqualToArray:colors])

{

NSLog(@”the given arrays are equal “);

}

**4>NSMutableArray:**

NSObject

NSArray

NSMutableArray

The easiest way to create mutable arrays is arrayWithObjects:method we can create the mutable empty array using ArrayWithCapacity: class method.

The NSMutableArray class declares the programmatic interface to objects that manage a modifiable array of objects. This class adds insertion and deletion operations to the basic array-handling behavior inherited from [NSArray](https://developer.apple.com/library/mac/documentation/Cocoa/Reference/Foundation/Classes/NSArray_Class/index.html" \l "//apple_ref/occ/cl/NSArray).

There is typically little reason to subclass NSMutableArray. The class does well what it is designed to do—maintain a mutable, ordered collection of objects. But there are situations where a custom NSArray object might come in handy. Here are a few possibilities:

* Changing how NSMutableArray stores the elements of its collection. You might do this for performance reasons or for better compatibility with legacy code.
* Acquiring more information about what is happening to the collection (for example, statistics gathering).

Eg:

NSMutableArray \*city = [NSMutableArray arrayWithObjects:@”hassn”,

@”banglore”,@”manglore”]);

* addObject and removeLastObject: adds the elements to the end of the array.
* insertObject:atIndex and removeObjectAtIndex: if we know the index of the object we use remove fromObjectatindex .if we don’t know the particular index means we will use the removeObject:method
* replaceObjectAtIndex:withObject: used to replace the content of the objects.