# Objective-C

### NSString

* Allocate and initialize String -
  + NSString \*immutable = [[NSString alloc] init];
* Initialize String -
  + NSString \*immutable = @”Hello”;
* Initialize string with format +
  + NSString \*immutable = [NSString stringWithFormat:@"Hello%@%@",@"eezy",@10];
* Creating and initializing a string with URL -
  + NSString \*immutable = [NSString stringWithContentsOfURL:[NSURL URLWithString:@http://facebook.com/”];
* Length of a string -
  + int x = [immutable length];
* Character at index -
  + NSString \*immutable1 = [immutable characterATIndex:0];
* Append two strings -
  + NSString \*immutable1 = [immutable stringByAppendingString:@”ghgh”];
* Substring from index -
  + NSString \*immutable1 = [immutable substringFromIndex:4];
* Substring from a range -
  + NSString \*immutable1 = [immutable substringWithRange:NSMakeRange(4,12)];
* Replace a substring with substring -
  + NSString \*immutable1 = [immutable stringByReplacingOccurrencesOfString:@"e" withString:@"a"];
* Compare two strings -
  + NSString \*immutable1 = [immutable compare:@"ios eezy tutorials"];
* Check if string has prefix -
  + NSString \*immutable1 = [immutable hasPrefix:@"abc"];
* Check if string has suffix -
  + NSString \*immutable1 = [immutable hasSuffix:@"abc"];
* Change to lowercase -
  + NSString \*immutable1 = [immutable lowercaseString];
* Change to uppercase -
  + NSString \*immutable1 = [immutable uppercaseString];
* Change to double int or bool -
  + NSNumber \*immutable1 = [immutable doubleValue];
  + NSNumber \*immutable1 = [immutable intValue];
  + BOOL \*immutable1 = [immutable boolValue];

### NSMutableString

* Allocate some heap memory +
  + NSMutableString \*mutable = [[NSMutableString alloc]init];
* Creates a string initially containing size characters +
  + NSMutableString \*mutable = [[NSMutableString stringWithString:@"Hello"];
* Initializes a string with an initial capacity of size characters +
  + NSMutableString \*mutable = [[NSMutableString stringWithCapacity:1000];
* Assign the pointer to 0x0 -
  + NSMutableString \*mutable = nil;
* Set mutable string from immutable string -
  + NSString \*a = @”Hello”
  + [mutable setString:a];
* Append mutable string with immutable string -
  + NSString \*a = @”Hello”
  + [mutable appendString:a];
* Delete a range of characters -
  + [mutable deleteCharactersInRange:NSMakeRange(0, 5)];
* Insert a string at an index -
  + [mutable insertString:@”Hello” atIndex:5];
* Replace a range with string -
  + [mutable replaceCharactersInRange: NSMakeRange(0, 5) withString:@”Hello”];

### NSArray

* Initialize an array –
  + NSArray \*immutable = @[@”Hello”, @”World”];
  + NSArray \*immutable = [NSArray arrayWithObjects:@”Hello”, @World”, nil];
  + NSString \*values[3];
    - values[0] = @”Hello”;
    - values[1] = @”World”;
* Count of an array –
  + Int count = [immutable count];
* Return the first object –
  + id \*immutable1 = [immutable firstObject];
* Return the last object –
  + id \*immutable1 = [immutable lastObject];
* Return object at an index –
  + id \*immutable1 = [immutable objectAtIndex:2];
* Return index of an object –
  + int x = [immutable indexOfObject:@"Hello"];
  + if (NSNotFound == x) // This is used for finding if something exists in array
  + NSNotFound is integer so make sure x is also an integer.
* See if two arrays are same –
  + If ([immutable1 isEqualToArray:immutable2]);

### NSMutableArray

* Allocate some heap memory +
  + NSMutableArray \*array = [[NSMutableArray alloc]init];
* Initializes an array with an initial capacity of size characters +
  + NSMutableArray \*mutable = [NSMutableArray arrayWithCapacity:1];
* Initializes an array with some objects -
  + NSMutableArray \*mutable = @[@”A”, @”B”, @”C”];
* Add object to array -
  + [mutable addObject:@"Hello"];
* Add from an array -
  + [mutable addObjectsFromArray:@[@"Hello",@"World"]];
* Add object at an index -
  + [mutable insertObject:@"Hello" atIndex:1];
* Remove all objects -
  + [mutable removeAllObjects];
* Remove last object -
  + [mutable removeLastObject];
* Remove certain object -
  + [mutable removeObject:@”Hello”];
* Remove object in a range -
  + [mutable removeObjectsInRange:NSMakeRange(1, 5)];
* Remove object at index -
  + [mutable removeObjectAtIndex:2];
* Replace at object at index -
  + [mutable replaceObjectAtIndex:0 withObject:@"Hello"];
* Sort an array -
  + NSSortDescriptor \*highestToLowest = [NSSortDescriptor sortDescriptorWithKey:@"self" ascending:NO];
  + [mutableArrayOfNumbers sortUsingDescriptors:[NSArray arrayWithObject:highestToLowest]];

### NSSet

* Initialize set with elements +
  + NSSet \*immutable = [NSSet setWithObjects:@"Hello",@"World",nil];
* Initialize set from an array –
  + NSSet \*immutable = [NSSet setWithArray:@[@"Hello",@"World"]];
* Add a new element to set –
  + NSSet \*immutable1 = [immutable setByAddingObjects:@"Hello"];
* Count in a set –
  + int d = [immutable count];
* All objects as an array –
  + NSArray \*immutable1 = [immutable allObjects];
* Return any object –
  + Id \*hello = [immutable anyObject];
* Check if contains an object –
  + if ([immutable containsObject:@”Hello”]);
* Is subset of set –
  + if ([immutable1 isSubsetOfSet:immutable2]);
* intersects set –
  + if ([immutable1 intersetctsSet:immutable2]);
* Is equal to set –
  + if ([immutable1 isEqualToSet:immutable2]);

### NSMutableSet

* Initialize and allocate set -
  + NSMutableSet \*mutable = [[NSMutableSet alloc ]init];
* Initializes a set with some objects -
  + NSMutableSet \*mutable = @[@”A”, @”B”, @”C”];
* Initializes an array with an initial capacity of size characters +
  + NSMutableSet \*mutable = [NSMutableSet setWithCapacity:1];
* Add object to set -
  + [mutable addObject:@2];
* Remove object -
  + [mutable removeObject:@2];
* Remove all objects -
  + [mutable removeAllObjects];
* Union of two set -
  + [mutable unionSet:mutable2];
* Minus of two set -
  + [mutable minusSet:mutable2];
* Intersect of two set -
  + [mutable intersectSet:mutable2];

### NSDictionary

* Initialize dictionary -
  + NSDictionary \*dict = @{@”key1”:@2.3, @”key2”:@2};
* Initialize dictionary with another dictionary +
  + NSDictionary \*dict = [NSDictionary dictionaryWithDictionary:dict1];
* Initialize dictionary with objects and values +
  + NSDictionary \*dict = [NSDictionary dictionaryWithObjects:@[@"Hello",@"World"]
  + forKeys:@[@"key1",@"key2"]];
* Number of entries in dictionary –
  + Int d = [dict count];
* Check if two dictionaries –
  + If ([dict1 isEqualToDictionary:dict2]);
* Get an array for all keys –
  + NSArray \*allKeys = [dict allKeys];
* Get all keys for certain that contains an object –
  + NSArray \*allKeys = [dict allKeysForObject:@”Hello”];
* Get an array for all values –
  + NSArray \*allValues = [dict allValues];
* Get value for a key -
  + id obj = [dict valueForKey:@"key1"];

### NSMutableDictionary

* Initialize and allocate dictionary -
  + NSMutableDictionary \*mutable = [[NSMutableDictionary alloc] init];
* Make a dictionary with objects and keys +
  + NSMutableDictionary \*mutable = [NSDictionary dictionaryWithObjectsAndKeys: @”roti”, @”Vish”, @”Dal”, @”Bawika”];
  + Key always comes after object here and all of it is in a list
* Set object for a key. You can also add like this -
  + [mutable setObject:@"roti" forKey:@"Vish"];
  + You can add in the dictionary like this as well.
* Remove object for key -
  + [mutable removeObjectForKey:@"key1"];
* Remove all entries –
  + [mutable removeAllObjects];
* Removes from the dictionary entries specified by elements in a given array.
  + [mutable removeObjectsForKeys:@[@"Vish", @”Bawika”]];

# All of the above are in foundations.h