

Algorithms for Beginners — Bubble Sort, Insertion Sort, Merge Sort



Erica Millado [Follow](#)

Oct 28, 2016 · 4 min read



As a new iOS developer, I am quickly learning the importance of honing my white-boarding skills (especially this week as we prepare for our visit to [ustwo](#)). White-boarding is the act of solving algorithmic problem on an actual “white-board” (think smelly dry-erase markers). This practice allows employers to assess your ability to think critically through algorithms. *Wait, what’s an algorithm?* An algorithm is a list of steps (a procedure) that you want you use to execute to accomplish a goal (solve a problem). As developers, we write algorithms everyday!

In computer science, there are many data structures and algorithms to familiarize oneself with. let’s focus on three *classic* algorithms for now: **bubble sort**, **insertion sort**, and **merge sort**.

Here are some images that briefly illustrate how each algorithm works:

Bubble Sort

6 5 3 1 8 7 2 4

Bubble Sort: compare two elements at time and swap if the 2nd element is larger than the first.

Bubble sort is considered the simplest sorting algorithm. It goes through an entire array and compares each neighboring number. It then swaps the numbers and keeps doing this until the list is in ascending order.

```

11 class ViewController: UIViewController {
12
13     var sampleArray = [9, 81, 0, 0, 5, 7, 27, 12, 46]
14
15     override func viewDidLoad() {
16         super.viewDidLoad()
17         print("This is the array before bubbleSort \(sampleArray).")
18         bubbleSort(array: &sampleArray)
19         print("This is the array AFTER bubbleSort \(sampleArray).")
20     }
21
22     func bubbleSort(array: inout [Int]) {
23         let arrayCount = array.count
24
25         for index in 0...arrayCount {
26             for value in 1...arrayCount-2 {
27                 if array[value-1] > array[value] {
28                     let largerValue = array[value-1]
29                     //here, we swap the larger value for the smaller value
30                     array[value-1] = array[value]
31                     array[value] = largerValue
32                 }
33             }
34         }
35     }
36 }

```

#1) Here's our sample array.

#2) Here, we iterate right over the array and in pairs, swap the greater and smaller values until the minimum value is in the correct position.

#3) Here, we call our bubbleSort function, which sorts our input array.

```

37
38     }
39
40
41 }
42

```

Bubble Sort in Swift 3

Above, the **bubbleSort()** function iterates over the array in two for loops and compares the values of two items in the array at a time. It swaps the larger value the place of the smaller value and continues to do so in the loop until the “left” value is no longer greater than the “right” value (the array is fully sorted).

This is the array before bubbleSort [9, 81, 0, 0, 5, 7, 27, 12, 46].
This is the array AFTER bubbleSort [0, 0, 5, 7, 9, 12, 27, 81, 46].

The return of Bubble Sort

Insertion Sort





Insertion Sort: Take a number from an array, put it into a new array in a sorted way.

Insertion sort involves going through a pile, taking one item, comparing it to the first, swapping places if one item is larger than another and continuing this process until the minimum item is in the correct location.

```

6  import UIKit
7
8  class ViewController: UIViewController {
9
10     var sampleArray = [9, 81, 0, 0, 5, 7, 27, 12, 46]
11
12     override func viewDidLoad() {
13         super.viewDidLoad()
14         print("This is the array before insertion sort \(sampleArray).")
15         insertionSort(array: &sampleArray)
16         print("This is the array AFTER insertion sort \(sampleArray).")
17     }
18
19     func insertionSort(array: inout [Int]) -> [Int]{
20         //loops through the array items
21
22         for firstNumToCompare in 1...array.count-1{
23             #4) In this method, we let currentValue = array[firstNumToCompare]
24             compare two var previousPosition = firstNumToCompare-1
25             numbers and
26             even the lesser
27
28         }
29     }
30 }

```

#1) Here's our sample array.

#2) inout!

#3) Here's our call of our insertionSort function.

#4) In this method, we let currentValue = array[firstNumToCompare] compare two numbers and var previousPosition = firstNumToCompare-1 even the lesser

```

28 swap the lesser
29 one to precede
30 the greater one.
31 We iterate left to
32 get the minimum
33 of the array.
34 }
35     return array
36 }
37

```

Insertion Sort in Swift 3

Above, the **insertionSort()** function iterates over the array and compares two items at a time. It swaps the items if one is larger than the other and continues to iterate *left*, comparing and swapping until the minimum is at the front of the array. Notice how I used **inout**! Thanks, [Joanna Huang](#) for the tip!

This is the array before insertion sort [9, 81, 0, 0, 5, 7, 27, 12, 46].
This is the array AFTER insertion sort [0, 0, 5, 7, 9, 12, 27, 46, 81].

The return from Insertion Sort. Personally, this was my least favorite.

Merge Sort

6 5 3 1 8 7 2 4

Merge Sort: Break an array into a smaller arrays (arrays of 1 element), then merge the arrays together while sorting them.

Imagine having to take a deck of cards, split it in two halves and continue splitting those piles in halves, and halves again until all you have is 52 piles of 1 card. UGH. Then, you regroup the piles in pairs again but this time, sort them in ascending order.

```

1 // ViewController.swift
2 // Sort_Blog
3 // Created by Erica Millado on 10/27/16.
4 // Copyright © 2016 Erica Millado. All rights reserved.
5 import UIKit
6
7 class ViewController: UIViewController {
8
9     var sampleArray = [9, 81, 0, 0, 5, 7, 27, 12, 46]
10
11     override func viewDidLoad() {
12         super.viewDidLoad()
13         print("This is the \(sampleArray) sampleArray ")
14         mergeAndSort(array: sampleArray)
15         print("This is the \(mergeAndSort(array: sampleArray)) merged and sorted array.")
16     }
17
18     func merge(leftArray:[Int], rightArray:[Int])->[Int] {
19         var leftIndex = 0
20         var rightIndex = 0
21         var orderedArray:[Int] = []
22         while leftIndex<leftArray.count && rightIndex<rightArray.count {
23             if leftArray[leftIndex] <= rightArray[rightIndex] {
24                 orderedArray = orderedArray + [leftArray[leftIndex]]
25                 leftIndex += 1
26             } else {
27                 orderedArray = orderedArray + [rightArray[rightIndex]]
28                 rightIndex += 1
29             }
30         }
31         orderedArray = orderedArray + Array(leftArray[leftIndex..

#1) Here's our sample array.



#5) Here, we call on our "mergeAndSort" function.



#3) This function takes the "left" and "right" arrays and merges them by comparing their values and appending the lesser values into a new array.



#2) This function takes an array and splits it into 2 arrays: "left" and "right".



#4) After the "merge" function puts them in order, the "mergeAndSort" function returns the finished array.


```

```

40     let leftArray = mergeAndSort(array: Array(array[0..

```

Merge Sort in Swift 3.

Above, two functions are used to merge sort. The function **mergeAndSort()** takes an array, splits it into two arrays (a left array and a right array) and then uses these two arrays as inputs into the **merge()** function, which in turn takes the two arrays and compares the values of the first indices of each array, adding each *lesser* value into a new array. In the end, an ordered array is returned when combining both now-sorted arrays.

This is the [9, 81, 0, 0, 5, 7, 27, 12, 46] sampleArray.
This is the [0, 0, 5, 7, 9, 12, 27, 46, 81] merged and sorted array.

This merge sort took a lot longer than insertion sort and bubble sort.

I hope this post gives you a basic overview of some of the most widely known sort algorithms and that you will start exploring more on your own.

Resources:

Sorting Algorithms Animated

Big-O Algorithm Cheat Sheet

[Programming](#)[Algorithms](#)[Sorting Algorithms](#)[Yayitserica](#)[Swift](#)

Discover Medium

Welcome to a place where words matter. On Medium, smart voices and original ideas take center stage - with no ads in sight. Watch

Make Medium yours

Follow all the topics you care about, and we'll deliver the best stories for you to your homepage and inbox. Explore

Become a member

Get unlimited access to the best stories on Medium — and support writers while you're at it. Just \$5/month. Upgrade

[About](#)[Help](#)[Legal](#)