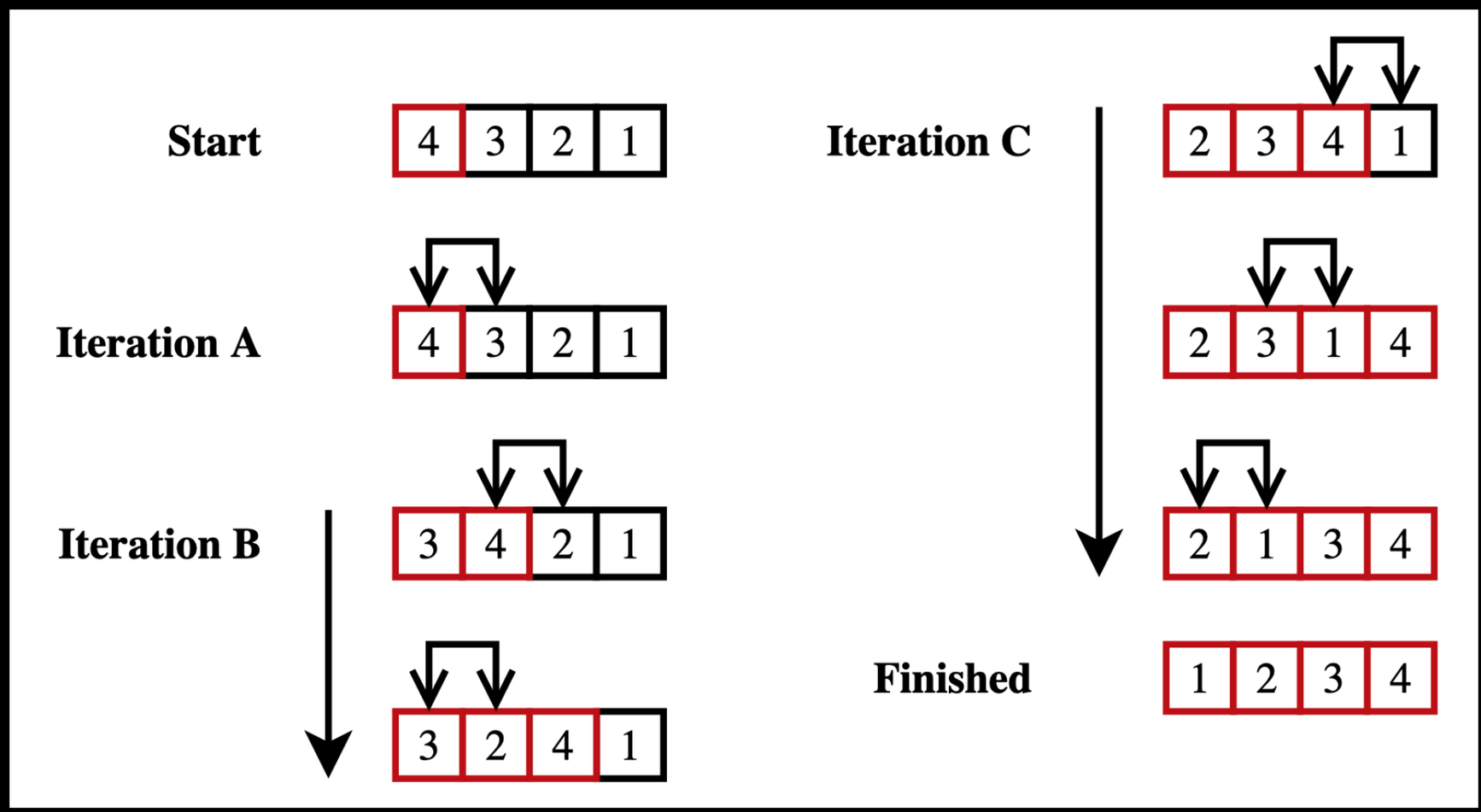
Algorithm
Insertion Sort

Bill Kim(김정훈) | ibillkim@gmail.com

목차

Insertion Sort

Insertion Sort(삽입 정렬)은 정렬 알고리즘으로서 특정 배열에서 처음 요소 왼쪽부터 오른쪽으로 순차적으로 비교해가면서 정렬해 나가는 알고리즘입니다.



Concept

기본적인 알고리즘의 컨셉을 살펴보면 아래와 같습니다.

1. 정렬되지 않는 배열 리스트(더미)가 있습니다.
2. 배열 리스트(더미)에서 숫자 하나를 선택합니다. 편한대로 첫번째 요소를 선택합니다.
3. 새로운 배열을 준비하고 고른 숫자를 삽입(Insert)합니다.
4. 더미에서 또다른 숫자를 선택합니다. 처음처럼 그냥 첫번째 요소를 선택합니다.
5. 새로운 배열에 삽입 후 해당 배열의 모든 요소와 비교하여 정렬된 상태로 변경합니다.
6. 더미에서 숫자가 없을때까지 반복합니다.

Concept

정렬되지 않는 더미 리스트 : [8, 3, 5, 4, 6]

1. 8을 선택 후 새로운 배열에 삽입

정렬된 배열 : [8]
더미 : [3, 5, 4, 6]

2. 3을 선택 후 새로운 배열에 삽입 후 정렬

정렬된 배열 : [3, 8]
더미 : [5, 4, 6]

3. 5를 선택 후 새로운 배열에 삽입 후 정렬

정렬된 배열 : [3, 5, 8]
더미 : [4, 6]

4. 4를 선택 후 새로운 배열에 삽입 후 정렬

정렬된 배열 : [3, 4, 5, 8]
더미 : [6]

5. 6을 선택 후 새로운 배열에 삽입 후 정렬

정렬된 배열 : [3, 4, 5, 6, 8]
더미 : [ ]

Concept

앞서 설명한 기본 알고리즘을 컨셉을 제자리(in-place)에서 삽입 정렬하는 방식으로도 가능합니다.

1. 배열의 요소 제일 왼쪽의 수부터 기준선을 잡고 시작한다.
2. 기준선을 기준으로 왼쪽에 있는 모든 요소들에 대해서 순차적으로 비교해가면서 정렬해 나갑니다.
3. 기준선을 하나 증가시킵니다.
4. 기준선이 배열 요소 끝까지 갈때까지 2번-3번의 과정을 반복한다.

Concept

[8, 3, 5, 4, 6] 인 배열을 정렬한다고 생각해봅니다.

위에서 설명한 방식대로 진행하면 아래와 같은 흐름으로 진행됨을 알 수 있습니다.

1: [| 8, 3, 5, 4, 6]
2: [8 | 3, 5, 4, 6]
3: [3, 8 | 5, 4, 6]
4: [3, 5, 8 | 4, 6]
5: [3, 4, 5, 8 | 6]
6: [3, 4, 5, 6, 8 |]

Features

삽입 정렬은 아래와 같은 특징을 가진 알고리즘입니다.

1. 정렬되지 않은 배열의 요소를 꺼내어 정렬 배열에 삽입하면서 부분적으로 정렬해가는 알고리즘
2. 하나하나 값을 꺼내고 새로운 배열로 정렬을 시키기때문에 모든 요소를 순차적으로 최악의 경우는 O(n^2) 의 속도를 가질 수도 있습니다.
3. 어느정도 정렬이 되어 있는 상태에서는 매우 빠른 속도를 보여줍니다.
4. 작은 크기의 배열에서 사용하면 더욱 효율적이다.

Implementation

Swift를 활용하여 삽입 정렬 알고리즘을 살펴보겠습니다.

```swift
func insertionSort<T: Comparable>(_ array: [T]) -> [T] {
    var sort = array

    for i in 1..<array.count {
        var j = i
        let temp = sort[j]
        while j > 0 && temp < sort[j - 1] {
          sort[j] = sort[j - 1]
          j -= 1
        }

        sort[j] = temp
    }
    return sort
}
```

Implementation

```swift
// isAscending: (T, T) -> Bool :
// 두 개의 T 객체를 받아서, 첫 번째 객체가 두 번째 것보다
// 앞에 위치하면 true를 반환하고, 그 반대라면 false를 반환하는 함수
func insertionSort<T: Comparable>(_ array: [T],
                                  _ isAscending: (T, T) -> Bool) -> [T] {
    var sort = array

    for i in 1..<array.count {
        var j = i
        let temp = sort[j]
        while j > 0 && temp < sort[j - 1] && isAscending(sort[j - 1], temp) {
          sort[j] = sort[j - 1]
          j -= 1
        }

        sort[j] = temp
    }
    return sort
}
```

Implementation

```
let numbers = [ 10, -1, 3, 9, 2, 27, 8, 5, 1, 3, 0, 26 ]

print(insertionSort(numbers)) // [-1, 0, 1, 2, 3, 3, 5, 8, 9, 10, 26, 27]
print(insertionSort(numbers, >)) // [-1, 0, 1, 2, 3, 3, 5, 8, 9, 10, 26, 27]
print(insertionSort(numbers, <)) // [10, -1, 3, 9, 2, 27, 8, 5, 1, 3, 0, 26]
```

References

[1] [Swift Algorithm Club 번역] 삽입 정렬 (Insertion Sort) : https://oaksong.github.io/2018/04/12/swift-algorithm-club-ko-insertion-sort/

[2] [Swift]Insertion Sort : http://minsone.github.io/programming/insertion-sort-in-swift

[3] 삽입 정렬(Insertion Sort) : https://kka7.tistory.com/76

[4] Swift Language 삽입 정렬 : https://riptutorial.com/ko/swift/example/28303/삽입-정렬

[5] [Swift 자료구조 ch09] 삽입정렬 (Insertion sort) : https://kor45cw.tistory.com/245

References

[6] 삽입 정렬 insertion sort with swift : https://hyerios.tistory.com/67

[7] Insertion Sort Swift : https://hryang.tistory.com/13

[8] What is Insertion Sort? : https://www.codingame.com/playgrounds/506/sorting-in-swift/insertion-sort

[9] Playing With Code: Insertion Sort In Swift : https://learnappmaking.com/insertion-sort-swift-how-to/

[10] Insertion Sort in Swift : https://medium.com/@notestomyself/insertion-sort-in-swift-fc593b6a7564

Thank you!