Arrays (Static/Dynamic)

@pitakill
Leopoldo Caballero

About

Algorithms

Left rotate

Reverse (Iterative/Recursive)

Bubble Sort

Left rotate - O(n)

Algorithm:

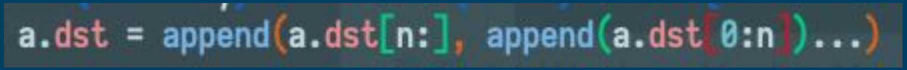
- Store k first elements
- Shift the rest of elements to the left
- Store back the k elements from the first step

Left rotate - O(n)

Ej: [0, 1, 2, 3, 4, 5] rotate left (2 positions)

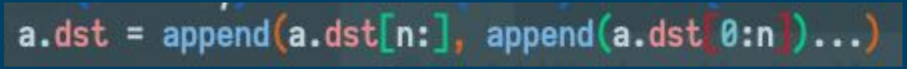
- Store the two first left elements: [0, 1]
- Shift the rest of elements: [2, 3, 4, 5, 4, 5]
- Store back the elements from first step: [2, 3, 4, 5, 0, 1]

Left rotate



```
a.dst = append(a.dst[n:], append(a.dst[0:n])...)
```

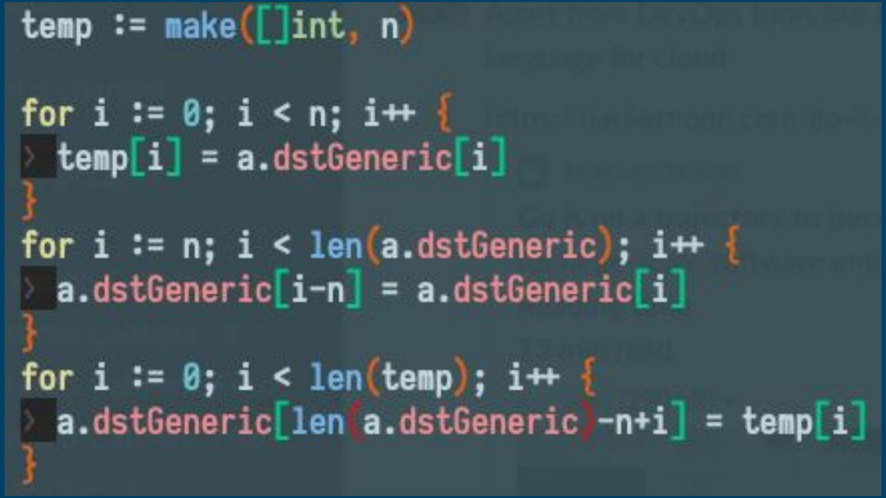
Left rotate



```
a.dst = append(a.dst[n:], append(a.dst[0:n])...)
```

What the *uck?

Left rotate



```
temp := make([]int, n)

for i := 0; i < n; i++ {
	temp[i] = a.dstGeneric[i]
}
for i := n; i < len(a.dstGeneric); i++ {
	a.dstGeneric[i-n] = a.dstGeneric[i]
}
for i := 0; i < len(temp); i++ {
	a.dstGeneric[len(a.dstGeneric)-n+i] = temp[i]
}
```

Reverse Iterative - O(n)

Algorithm:

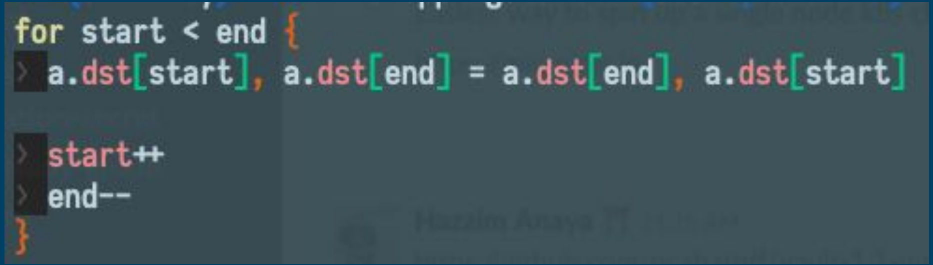
- Track first and last indexes
- In a loop interchange them. The condition of the loop is (start = start + 1 and end = end - 1) when end is greater than start

Reverse Iterative - O(n)

Ex: [10, 20, 30, 40, 50, 60]

- Track first and last indexes: start = 0, end = 5
- Interchange them [60, 20, 30, 40, 50, 10], and so on with the condition: start = start + 1 and end = end - 1 until the start is greater than end
- Resulting: [60, 50, 30, 40, 20, 10]
- Next step: start = 2, end = 3, so run the loop again
- Resulting: [60, 50, 40, 30, 20, 10]
- Next step: start = 3, end = 2, so the loop ends

Reverse Iterative



```
for start < end {
	a.dst[start], a.dst[end] = a.dst[end], a.dst[start]

	start++
	end--
}
```

Reverse Declarative - O(n)

Algorithm:

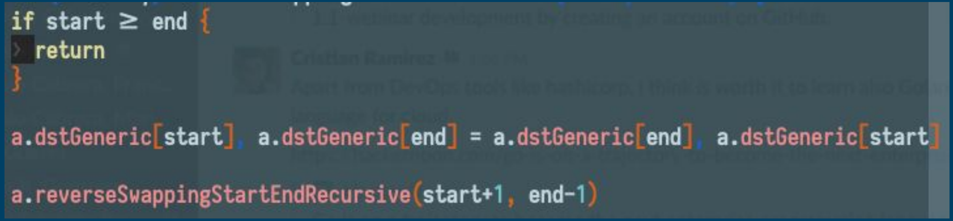
- Track first and last indexes
- Swap the values array[start] with array[end] and vice versa
- Recursively call it again with start + 1 and end - 1

Reverse Declarative - O(n)

Ex: [10, 20, 30, 40, 50, 60]

- Track first and last indexes: start = 0, end = 5
- Interchange them [60, 20, 30, 40, 50, 10]
- Recursively call it again with start + 1 and end - 1
- [60, 50, 30, 40, 20, 10], and so on
- [60, 50, 40, 30, 20, 10]

Reverse Declarative



```
if start ≥ end {
	return
}

a.dstGeneric[start], a.dstGeneric[end] = a.dstGeneric[end], a.dstGeneric[start]

a.reverseSwappingStartEndRecursive(start+1, end-1)
```

Bubble Sort - $O(n^2)$

Algorithm:

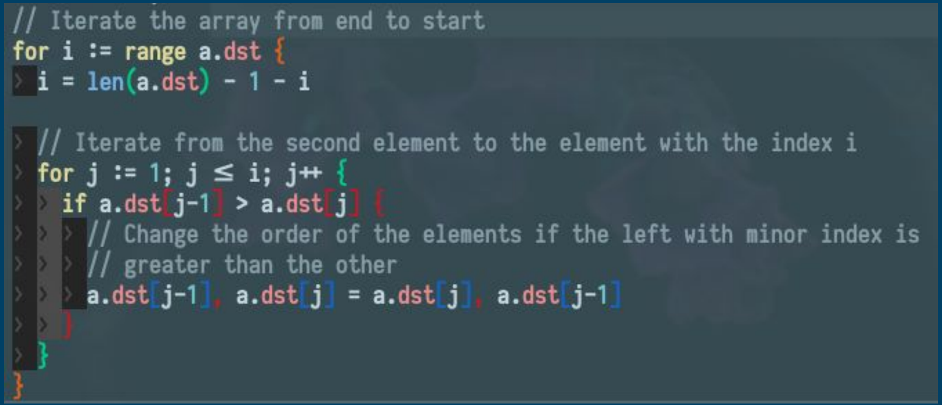
- Iterate the array from last index to first
- Inside the other iteration, iterate the array from the second position to the index of the first iteration
- Compare item in the second iteration with its ancestor and swap them make them the greater to the right position

Bubble Sort - O(n^2)

Ex: [3,1,6,0,2,7]

- Compare 1 with 3. So, 3 is greater than 1, swap them:
- [1,3,6,0,2,7]
- The next iteration compares: 6 with 3. So, 6 is greater than 3, let them in its positions.
- The next iterations compares: 0 with 6. So, 6 is greater than 3, swap them:
- [1,3,0,6,2,7]. So on, until finish the second iteration resulting:
- [1,3,0,2,6,7]. So the first iteration runs over the array again, swapping accordingly:
- [1,0,2,3,6,7]. So the first iteration runs over the array again, swapping accordingly:
- [0,1,2,3,6,7]. Resulting in a sorted array.

Bubble Sort



```
// Iterate the array from end to start
for i := range a.dst {
    i = len(a.dst) - 1 - i

    // Iterate from the second element to the element with the index i
    for j := 1; j ≤ i; j++ {
        if a.dst[j-1] > a.dst[j] {
            // Change the order of the elements if the left with minor index is
            // greater than the other
            a.dst[j-1], a.dst[j] = a.dst[j], a.dst[j-1]
        }
    }
}
```

Thank you