Комбинации ветвлений и циклов. Сортировка Шпаргалка

Алгоритм пузырьковой сортировки

Алгоритм пузырьковой сортировки - простой алгоритм, который проходит по массиву несколько раз, каждый раз перемещая большие элементы в конец массива. На каждом проходе i элемент находится на своем месте, поэтому в следующем проходе можно начинать с i+1. Алгоритм останавливается, когда на очередном проходе не произошло ни одной перестановки.

```
func bubbleSort(arr []int) {
    n := len(arr)
    for i := 0; i < n-1; i++ {
        for j := 0; j < n-i-1; j++ {
            if arr[j] > arr[j+1] {
                arr[j], arr[j+1] = arr[j+1], arr[j]
            }
        }
    }
}
```

Алгоритм сортировки вставками

Алгоритм сортировки вставками - проходится по массиву, вставляя каждый элемент на правильное место в уже отсортированной части массива. На i-м проходе i-й элемент сравнивается с j-м элементом, где j<i, и вставляется на место j.

```
func insertionSort(arr []int) {
    n := len(arr)
    for i := 1; i < n; i++ {
        key := arr[i]
        j := i - 1
        for j >= 0 && arr[j] > key {
            arr[j+1] = arr[j]
            j = j - 1
        }
        arr[j+1] = key
    }
}
```

Алгоритм быстрой сортировки

Алгоритм быстрой сортировки - использует стратегию "разделяй и властвуй". Он выбирает опорный элемент (pivot), разбивает массив на две части - элементы меньше опорного и элементы больше опорного - и рекурсивно сортирует каждую часть.

```
func quickSort(arr []int, low, high int) {
    if low < high {
        pi := partition(arr, low, high)
        quickSort(arr, low, pi-1)
        quickSort(arr, pi+1, high)
    }
}

func partition(arr []int, low, high int) int {
    pivot := arr[high]
    i := low - 1
    for j := low; j < high; j++ {
        if arr[j] < pivot {
            i++
            arr[i], arr[j] = arr[j], arr[i]
        }
    }
    arr[i+1], arr[high] = arr[high], arr[i+1]
    return i + 1
}
```

Метод двух указателей

Метод двух указателей - метод решения задач, который использует два указателя, которые движутся в разных направлениях в массиве. Обычно этот метод используется для нахождения пары элементов, которые в сумме дают заданную цель.

```
func twoPointer(arr []int, target int) []int {
    left, right := 0, len(arr)-1
    for left < right {
        sum := arr[left] + arr[right]
        if sum == target {
            return []int{left, right}
        } else if sum < target {
            left++
        } else {
            right--
        }
    }
```

```
    return []int{-1, -1}
}
```