Сортиране, 12.10

# Термини

## Stable

## In-place

## Out-place

## Time Complexity

* Как нараства времето на изпълнение спрямо размера на входа.

## Space Complexity

## Number of comparisons

## Number of swaps

## Защо гледаме само доминиращите сложности

## Adaptivity

* Когато необходимият брой операции се влияе от какъв вход се подава
* Когато му подадем напълно сортиран масив или обратно сортиран масив, дали нашия алгоритъм се възползва от наредбата на нашите елементи.
* Quick sort e добър при разбъркани елементи и не толкова добър при почти подредени.

## Online свойство

* Ако добавим един елемент, да не run-ваме алгоритъма наново.

# Теоретични въпроси

## Въпрос 1

* Ако имаме редица и имаме само релацията, R(a, b) -> а < b, можем ли да сортираме редицата.
* Не защото множеството не е частична наредба, трябва да имаме релацията <=
* Ако a <= b и b <= a, то а = b
* Ако а <= b и b <= c, то а <= c
* а <= а

## Въпрос 2

* Адаптивен ли е Bubble Sort?
* Адаптивен е. Правим го адаптивен като преценим, че ако нямаме размяна значи всички след елемента са сортирани

## Въпрос 3

* Кога Класическата имплементация на Bubble Sort се представя най-бавно:
* Когато елементите са в обратен ред

## Въпрос 4

* Използвайки bool flag за проверка каква става Worst-Case сложността на Bubble Sort:
* О(n2)

## Въпрос 5

* Stable ли е Bubble Sort?
* Да.

# Метода на мехурчето

## Логика

* Представяме си всеки елемент като мехурче и тежестта му си я представяме като въздух в мехурче.
* Може да си представим мехурчетата по вертикален начин и на първата итерация балончето с най-много въздух ще отиде най-отгоре.

## Time complexity

* O(n2)

## Space complexity

* O(1)

## In-place

* Да, не използва допълнителна памет.

## Number of comparisons

* O(n2)

## Number of swaps

* O(n2)

## Adaptive

* Да

## Stable

* Да

## Online

* Да
* За линейно време като приемем, че имаме сортирана редица, вкарваме нов и я сортираме.

# Selection sort

## Логика

* В някой случаи е по-добър от bubble sort-a.
* Намираме най-малкия елемент и го слагаме на първо място.

## Time complexity

* O(n2)

## Space complexity

* O(1)

## In-place

* Да, не използва допълнителна памет.

## Number of comparisons

* O(n2)

## Number of swaps

* O(n)

## Adaptive

* Не

## Stable

* Да

## Online

* Не