Алгоритми върху масиви

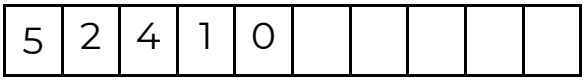
Минимален елемент.

Взимаме първия елемент от масива като минимален, след което за всеки един елемент от масива проверяваме дали е по-малък от нашия, ако е - записваме него вместо нашия елемент.

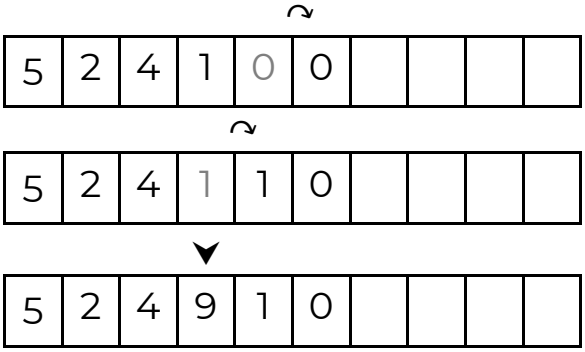
Добавяне на елемент на позиция.

За да добавим елемент на позиция **m**, първо трябва да изместим всички елементи от **m**-тия до последния с една позиция “надясно”, започвайки от последния. След което можем да добавим елемента на позиция **m**, която е вече свободна. “Увеличаваме” размера на масива с 1.

Начален масив (arr_size = 5)



1. Проверяваме дали максималният размер на масива позволява добавяне - ако позволява => arr_size = 6
2. Добавяне на елемент 9 на индекс 3:

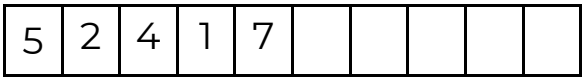


Премахване на елемент на позиция.

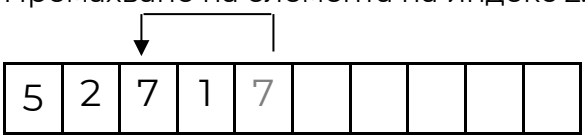
- *когато наредбата на елементите НЕ е от значение*

Разменяме стойностите на елементите на позиция **m** и последния елемент. “Намаляваме” размера на масива с 1.

Начален масив (arr_size = 5)



Премахване на елемента на индекс 2:



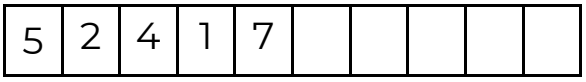
arr_size = 4

Ако е нужно може да занулим последния елемент (седмицата)

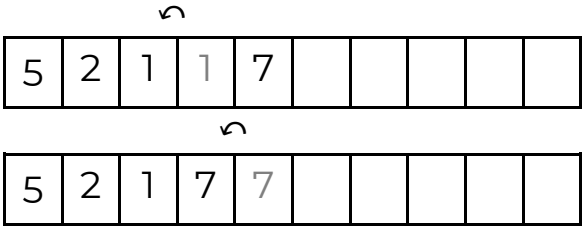
Премахване на елемент на позиция.

- *когато наредбата на елементите е от значение*

Начален масив (arr_size = 5)



Премахване на елемента на индекс 2:



arr_size = 4

Ако е нужно може да занулим последния елемент (седмицата)

Сортировки. Стабилност. Поточност. Частичност.

- Един алгоритъм за сортиране е **стабилен**, ако след сортирането се запазва реда на елементите с еднаква стойност.
- Един сортиращ алгоритъм е **поточен**, ако можем да му подаваме елементи един по един и за всеки да създава сортирана поредица.
- Един сортиращ алгоритъм е **частичен**, ако след известен брой стъпки първите k на брой елемента са най-малките от редицата и са в сортиран вид.
  Можем да си мислим, че:

Частичност ⇔ !Поточност както и *Поточност ⇔ !Частичност*

Сортиране чрез пряка селекция. (Selection sort)

Частичен и нестабилен алгоритъм за сортиране.

Стъпки:

1. Намира най-малкия елемент от несортираните.
2. Премества го най-отпред на несортираните елементи.
3. Ако има несортирани елементи се връща на стъпка 1.

Сортиране чрез вмъкване. (Insertion sort)

Поточен и стабилен алгоритъм за сортиране.
Реденето на карти в ръката е подобно на този алгоритъм.
Взимаме първия елемент от несортираната част и го слагаме подреден в сортираната част на масива.

Метод на мехурчето. (Bubble sort) ♡

Стабилен и частичен алгоритъм за сортиране.
Изплуват/потъват “леките/тежките” елементи. =)
Сравнява всеки два съседни елемента и ги разменя, ако не са наредени. В края на всяка k-та итерация гарантира, че в края стоят k най-големи и подредени елемента от масива.

Сортиране чрез броене.

Може да бъде стабилен и евентуално частичен алгоритъм за сортиране в **частни случаи**.

Последователно търсене.

Обхожда елементите на масива един по един последователно, при достигане на желания елемент връща индекса му.

Двоично търсене.

Работи САМО върху сортиран масив. Трябва да имате *произволен достъп* до елементите от редицата.
По същия начин, когато търсите страница в книга. Отваряте в средата и гледате дали сте отворили на нея. Ако не е тя, то търсената страница е или преди или след текущата.

Ако трябва да се познае число в даден интервал и се казва дали е избрано по-голямо или по-малко число. Избира се средата на интервала и така гарантирано се премахват най-много ненужни числа (половината). Това се повтаря докато числото не се познае.