# Практически задачи: задачи върху масиви (обръщане на масив отзад напред, ротация на масив наляво / надясно)

## Обръщане на последователността на елементите на масив

Напишете програма, която въвежда масив от цели числа, Обръща го и извежда елементите. Входните данни са числото **n** (брой на елементите) + **n** цели числа, всяко на отделен ред. Изведете резултата на един ред, за разделител да се ползва интервал

### **Примери**

|  |  |
| --- | --- |
| Вход | Изход |
| 3  10  20  30 | 30 20 10 |
| 4  -1  20  99  5 | 5 99 20 -1 |

### **Упътване**

* Първо, въведете числото n.
* Създайте масив от n цели числа.
* Въведете с цикъл for числата.
* Вместо да обръщате масива, можете просто да изведете елементите му като го обходите от последния до първия

## Обръщане на масив от символни низове

Напишете програма, която да прочете масив от символни низове, обръща масива и печата на неговите елементи. Входът се състои от поредица от низове, разделени с интервал. Отпечатва резултата на един ред с разделител интервал.

### **Примери**

|  |  |
| --- | --- |
| Вход | Изход |
| a b c d e | e d c b a |
| -1 hi ho w | w ho hi -1 |

### **Упътване**

* Въведете масив от символни низове
* Разменете първият елемент (с индекс 0) с последния елемент (с индекс n--1)
* Продължете с тези размени с останалите елементи докато стигнете средата на масива



* Друг, по-кратък подход е да се ползва готовия extension метод **.Reverse()** от **System.Linq**.

## Завъртане и сумиране

“Завъртане на масив на дясно” означава да преместим неговия последен елемент на първо място: {1, 2, 3} 🡪 {3, 1, 2}.

Напишете програма, която въвежда масив от n цели числа (разделени с интервал на един ред) и цяло число k, завърта k пъти надясно и сумира получените масиви след всяко завъртане както е показано по-долу:

### **Примери**

|  |  |  |
| --- | --- | --- |
| Вход | Изход | Коментари |
| 3 2 4 -1  2 | 3 2 5 6 | rotated1[] = -1 3 2 4  rotated2[] = 4 -1 3 2  sum[] = 3 2 5 6 |
| 1 2 3  1 | 3 1 2 | rotated1[] = 3 1 2  sum[] = 3 1 2 |
| 1 2 3 4 5  3 | 12 10 8 6 9 | rotated1[] = 5 1 2 3 4  rotated2[] = 4 5 1 2 3  rotated3[] = 3 4 5 1 2  sum[] = 12 10 8 6 9 |

### **Упътване**

* След r завъртания, елементът на позиция i отива на позиция (i + r) % n.
* Масивът sum[] може да бъде изчислен с два вложени цикъла : for r = 1 … k; for i = 0 … n-1.

## Сгъни и събери

Въведете масив от 4\*k цели числа, сгънете го както е указано по-долу и изведете сумата на горния и долния ред (всеки, съдържащ 2 \* k цели числа):



### **Примери**

|  |  |  |
| --- | --- | --- |
| Вход | Изход | Коментари |
| 5 2 3 6 | 7 9 | 5 6 +  2 3 =  7 9 |
| 1 2 3 4 5 6 7 8 | 5 5 13 13 | 2 1 8 7 +  3 4 5 6 =  5 5 13 13 |
| 4 3 -1 2 5 0 1 9 8 6 7 -2 | 1 8 4 -1 16 14 | -1 3 4 -2 7 6 +  2 5 0 1 9 8 =  1 8 4 -1 16 14 |

### **Упътване**

* Създайте първия ред след сгъването: първите k числа обърнати, последвани от последните k числа, също обърнати.
* Създайте втория ред след сгъването, като вземете средните 2\*к числа
* Сумирайте първи и втори ред