# Exercises: Arrays and Lists

Test your solutions in the **judge system**: <https://judge.softuni.org/Contests/4437>

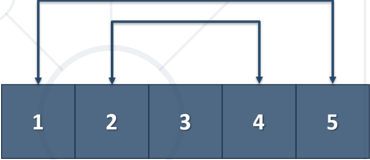
**Note: Problems marked with star (\*) are not included in the final homework bonus. These tasks are not mandatory, because of their difficulty level.**

## Gauss' Trick

Write a program that:

* Reads a **sequence with integer numbers** from the console
* **Sums all numbers in a list** in the following order:

**first + last, first + 1 + last - 1, first + 2 + last - 2, … first + n, last – n**



### Example

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 1 2 3 4 5 | 6 6 3 | 1 2 3 4 | 5 5 |

## Condense Array to Number

Write a program that:

* Read **a sequence of integer numbers** from the console
* **Condense** them by **summing** adjacent couples of elements until a **single integer** is obtained

**Example:** If we have 3 elements {2, 10, 3}. We sum the first two and the second two elements and obtain

{2+10, 10+3} = {12, 13}. Then we sum again all adjacent elements and obtain {12+13} = {25}.

### Example

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 2 10 3 | 25 | 2 10 3 🡪 2+10 10+3 🡪 12 13 🡪 12 + 13 🡪 25 |
| 5 0 4 1 2 | 35 | 5 0 4 1 2 🡪 5+0 0+4 4+1 1+2 🡪 5 4 5 3 🡪 5+4 4+5 5+3 🡪 9 9 8 🡪 9+9 9+8 🡪 18 17 🡪 18+17 🡪 35 |
| 1 | 1 | 1 is already condensed to number |

## Merging Lists

Write a program that:

* Reads two **sequences with integer numbers** from the console
* Create a result list that **contains the numbers from both** of the sequences
* The **first element should be from the first sequence**, the **second from the second sequence**, and so on
* If the length of the two **sequences is not equal**, just add the remaining elements at the end of the sequence

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| 3 5 2 43 12 3 54 10 23  76 5 34 2 4 12 | 3 76 5 5 2 34 43 2 12 4 3 12 54 10 23 |
| 76 5 34 2 4 12  3 5 2 43 12 3 54 10 23 | 76 3 5 5 34 2 2 43 4 12 12 3 54 10 23 |

## \*Sum Adjacent Equal Numbers

Write a program that:

* Reads a **sequence of integer numbers** from the console
* **Sum all adjacent equal numbers** in a list of decimal numbers, starting from **left to right**
  + After two numbers are summed, the obtained result could be equal to some of its neighbors and should be summed as well
  + Always sum the **leftmost** two equal neighbors (if several couples of equal neighbors are available)

### Example

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Explanation** |
| 3 3 6 1 | 12 1 | **3 3** 6 1 🡪 **6 6** 1 🡪 12 1 |
| 8 2 2 4 8 16 | 16 8 16 | 8 **2 2** 4 8 16 🡪 8 **4 4** 8 16 🡪 **8 8** 8 16 🡪 16 8 16 |
| 5 4 2 1 1 4 | 5 8 4 | 5 4 2 **1 1** 4 🡪 5 4 **2 2** 4 🡪 5 **4 4** 4 🡪 5 8 4 |

## Bomb Numbers

Write a program that:

* Reads a **sequence of integer numbers** from the first line of the console
* Read a **special bomb number (integer)** and its **power (integer)** from the second line of the console
* Detonate **every occurrence of the special bomb number** and according to its power - **his neighbors from left and right**
* Detonations are performed from left to right, and all detonated numbers disappear
* Print the **sum of the remaining elements** in the sequence

### Example

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 1 2 2 4 2 2 2 9  4 2 | 12 | The special number is 4 with power 2. After detonation, we left with the sequence [1, 2, 9] with sum 12. |
| 1 4 4 2 8 9 1  9 3 | 5 | The special number is 9 with power 3. After detonation, we left with the sequence [1, 4] with sum 5. Since the 9 has only 1 neighbor from the right, we remove just it (one number instead of 3). |
| 1 7 7 1 2 3  7 1 | 6 | Detonations are performed from left to right. We could not detonate the second occurrence of 7 because it's already destroyed by the first occurrence. The numbers [1, 2, 3] survive. Their sum is 6. |
| 1 1 2 1 1 1 2 1 1 1  2 1 | 4 | The red and yellow numbers disappear in two sequential detonations. The result is the sequence [1, 1, 1, 1]. Sum = 4. |
| 1 2 3 4 5 6 7  2 2 | 18 | [5, 6, 7] |
| 1 2 3 4 5 6 7  1 4 | 13 | [6, 7] |

## \*Cards Game

You will be given two hands of cards, which will be **integer** numbers. Assume that you have two players. You must find the winning deck and, respectively, the winner.

You start from the beginning of both hands. Compare the cards from the first deck to those from the second. The player, who has a bigger card, takes both cards and puts them on the **back** of his hand - **the second player's card is last, and the first person's card (the winning one) is before it (second to last),** and the player with the smaller card must **remove** the **card** from his deck. If both players' cards have the same values - no one wins, and the two cards must be **removed** **from** the **decks**.

The game is over when one of the decks is left without any cards. You have to print the winner on the console and the sum of the left cards:

**"{First/Second} player wins! Sum: {sum}"**.

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| 20 30 40 50  10 20 30 40 | First player wins! Sum: 240 |
| 10 20 30 40 50  50 40 30 30 10 | Second player wins! Sum: 50 |

## Zig-Zag Arrays

Write a program that:

* Creates **two empty integer arrays**
* Reads an **integer number N** from the console
* Reads two **integer numbers** on the next N lines
* Form **two integer arrays** as shown below

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| 4  1 5  9 10  31 81  41 20 | 1 10 31 20  5 9 81 41 |
| 2  80 23  31 19 | 80 19  23 31 |

## Array Rotation

Write a program that:

* Reads **sequence of integer numbers** from the first line of the console
* Reads **an integer number (rotations you have to perform)** from the second line of the console
* Each rotation is when the **first element goes at the end**
* Print the **resulting sequence**

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| 51 47 32 61 21  2 | 32 61 21 51 47 |
| 32 21 61 1  4 | 32 21 61 1 |
| 2 4 15 31  5 | 4 15 31 2 |

## \*Max Sequence of Equal Elements

Write a program that:

* Reads **sequence of integer numbers** from the first line of the console
* Find the **longest sequence of equal elements** in a sequence of integers

**Note:** If several longest sequences exist, **print the leftmost one.**

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| 2 1 1 2 3 3 **2 2 2** 1 | 2 2 2 |
| **1 1 1** 2 3 1 3 3 | 1 1 1 |
| **4 4 4 4** | 4 4 4 4 |
| 0 **1 1** 5 2 2 6 3 3 | 1 1 |

## \*Top Integers

Write a program that:

* Reads **sequence of integer numbers** from the first line of the console
* Find all the top integers in a sequence
* Top integer is an integer that is **bigger** than all the elements to its right

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1 4 3 2 | 4 3 2 |
| 14 24 3 19 15 17 | 24 19 17 |
| 27 9 42 2 13 45 48 | 48 |