# Exercise: Arrays

Problems for exercises and homework for the [“Technology Fundamentals” course @ SoftUni](https://softuni.bg/trainings/3212/java-fundamentals-january-2021).

You can check your solutions in [Judge](https://judge.softuni.bg/Contests/1247).

## Train

You will be given a count of wagons in a train **n**. On the next **n** lines, you will receive how many people are going to get on that wagon. At the end print the whole train and after that the sum of the people in the train.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 3  13  24  8 | 13 24 8  45 |
| 6  3  52  71  13  65  4 | 3 52 71 13 65 4  208 |
| 1  100 | 100  100 |

## Common Elements

Write a program, which prints common elements in two arrays. You have to compare the elements of the second array to the elements of the first.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Hey hello 2 4  10 hey 4 hello | 4 hello |
| S of t un i  of i 10 un | of i un |
| i love to code  code i love to | code i love to |

## Zig-Zag Arrays

Write a program which creates 2 arrays. You will be given an integer **n**. On the next **n** lines, you get 2 integers. Form 2 arrays as shown below.

### Examples

|  |  |
| --- | --- |
| **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 receives an array and number of rotations you have to perform (first element goes at the end) Print the resulting array.

### Examples

|  |  |
| --- | --- |
| **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 |

## Top Integers

Write a program to find all the top integers in an array. A top integer is an integer which is **bigger** than all the elements to its right.

### Examples

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

## Equal Sums

Write a program that determines if there **exists an element in the array** such that the **sum of the elements on its left** is **equal** to the **sum of the elements on its right**. If there are **no elements to the left / right**, their **sum is considered to be 0**. Print the **index** that satisfies the required condition or **“no”** if there is no such index.

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 1 2 3 3 | 2 | At a[2] -> left sum = 3, right sum = 3  a[0] + a[1] = a[3] |
| 1 2 | no | At a[0] -> left sum = 0, right sum = 2  At a[1] -> left sum = 1, right sum = 0  No such index exists |
| 1 | 0 | At a[0] -> left sum = 0, right sum = 0 |
| 1 2 3 | no | No such index exists |
| 10 5 5 99 3 4 2 5 1 1 4 | 3 | At a[3] -> left sum = 20, right sum = 20  a[0] + a[1] + a[2] = a[4] + a[5] + a[6] + a[7] + a[8] + a[9] + a[10] |

## Max Sequence of Equal Elements

Write a program that finds the **longest sequence of equal elements** in an array of integers. If several longest sequences exist, print the leftmost one.

### Examples

|  |  |
| --- | --- |
| **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 |

## Magic Sum

Write a program, which prints all unique pairs in an array of integers whose sum is equal to a given number.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1 7 6 2 19 23  8 | 1 7  6 2 |
| 14 20 60 13 7 19 8  27 | 14 13  20 7  19 8 |

## \*Kamino Factory

The clone factory in Kamino got another order to clone troops. But this time you are tasked to find **the best DNA** sequence to use in the production.

You will receive the **DNA length** and until you receive the command **"Clone them!"** you will be receiving a **DNA sequences of ones and zeroes, split by "!"** **(one or several).**

You should select the sequence with the **longest subsequence of ones**. If there are several sequences with **same length of** **subsequence of ones**, print the one with the **leftmost** **starting index**, if there are several sequences with same **length and starting index**, select the sequence with the **greater sum** of its elements.

After you receive the last command "Clone them!" you should print the collected information in the following format:

"Best DNA sample {bestSequenceIndex} with sum: {bestSequenceSum}."

"{DNA sequence, joined by space}"

### Input / Constraints

* The **first line** holds the **length** of the **sequences** – **integer in range [1…100];**
* On the next lines until you receive **"Clone them!"** you will be receiving sequences (at least one) of ones and zeroes, **split by "!"** (one or several).

### Output

The output should be printed on the console and consists of two lines in the following format:

"Best DNA sample {bestSequenceIndex} with sum: {bestSequenceSum}."

"{DNA sequence, joined by space}"

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 5  1!0!**1!1**!0  0!**1!1**!0!0  Clone them! | Best DNA sample 2 with sum: 2.  0 1 1 0 0 | We receive 2 sequences with **same length** **of subsequence of ones**, but the second is printed, because its subsequence starts at **index[1].** |
| **Input** | **Output** | **Comments** |
| 4  **1!1**!0!**1**  1!0!0!1  **1!1**!0!0  Clone them! | Best DNA sample 1 with sum: 3.  1 1 0 1 | We receive 3 sequences. Both 1 and 3 **have same length** of subsequence of ones -> 2, **and both start from index[0]**, but the first is printed, because its **sum is greater.** |