# 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 – 1

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

## More Task

## Pascal Triangle

The triangle may be constructed in the following manner: In row 0 (the topmost row), there is a unique nonzero entry 1. Each entry of each subsequent row is constructed by adding the number above and to the left with the number above and to the right, treating blank entries as 0. For example, the initial number in the first (or any other) row is 1 (the sum of 0 and 1), whereas the numbers 1 and 3 in the third row are added to produce the number 4 in the fourth row.

If you want more info about it: <https://en.wikipedia.org/wiki/Pascal's_triangle>

Print each row elements separated with whitespace.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 4 | 1  1 1  1 2 1  1 3 3 1 |
| 13 | 1  1 1  1 2 1  1 3 3 1  1 4 6 4 1  1 5 10 10 5 1  1 6 15 20 15 6 1  1 7 21 35 35 21 7 1  1 8 28 56 70 56 28 8 1  1 9 36 84 126 126 84 36 9 1  1 10 45 120 210 252 210 120 45 10 1  1 11 55 165 330 462 462 330 165 55 11 1  1 12 66 220 495 792 924 792 495 220 66 12 1 |

### Hints

* The input number **n** will be **1 <= n <= 60**
* Think about proper **type** for elements in array
* Don’t be scary to use **more and more arrays**

## Recursive Fibonacci

The Fibonacci sequence is quite a famous sequence of numbers. Each member of the sequence is calculated from the sum of the two previous elements. The **first two** elements are 1, 1. Therefore the sequence goes as 1, 1, 2, 3, 5, 8, 13, 21, 34…  
The following sequence can be generated with an array, but that’s easy, so your task is to implement recursively.

So if the function **GetFibonacci(n)** returns the n’th Fibonacci number we can express it using **GetFibonacci(n) = GetFibonacci(n-1) + GetFibonacci(n-2).**

However, this will never end and in a few seconds a StackOverflow Exception is thrown. In order for the recursion to stop it has to have a “**bottom**”. The bottom of the recursion is **GetFibonacci(2)** should return 1 and **GetFibonacci(1)** should return 1.

### Input Format:

* On the only line in the input the user should enter the wanted Fibonacci number.

### Output Format:

* The output should be the n’th Fibonacci number counting from 1.

### Constraints:

* 1 ≤ N ≤ 50

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5 | 5 |
| 10 | 55 |
| 21 | 10946 |

For the Nth Fibonacci number, we calculate the N-1th and the N-2th number, but for the calculation of N-1th number we calculate the N-1-1th(N-2th) and the N-1-2th number, so we have a lot of repeated calculations.



If you want to figure out how to skip those unnecessary calculations, you can search for a technique called