# Exercises: Arrays

Please submit your solutions (source code) of all below-described problems in [Judge](https://judge.softuni.org/Contests/2940/Arrays-Exercises)

## Zig-Zag Arrays

Write a program that creates two arrays:

* Read an integer number **N (N < 100) from the first line of the console**, which represents **size of the arrays**
* On the next **N** lines, read **two integer numbers**
* **Form two arrays** as shown below
* Print two arrays, **each on the separate line**
* Elements in the arrays have to be printed, **separated by single space**

### 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   1. 31 |

## Longest Sequence

Write a program that:

* Read an **integer number** **N (N < 100) from the first line of the console**, which represents **size of the array**
* Read an **integer array** **with the given size** from the second line of the console
* Finds the **longest sequence of equal elements** in the given integer array
* Prints that **sequence** on the console (integer numbers are separated by single space on a single line)

**Note:** If there is more than one such sequence, print the last one.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 7  13 10 10 1 4 2 10 | 10 10 |
| 5  13 42 19 21 103 | 103 |

## Above Average

Write a program that:

* Read an **integer number** **N (N < 100) from the first line of the console**, which represents **size of the array**
* Read an **integer array** **with the given size** from the second line of the console
* Find all numbers which are **larger than or equal to the mathematical average (rounded to the smallest integer number)** of the numbers in the array
* The numbers should be printed on a single line, separating the output number by spaces.

**Note:** The output numbers should be in the same order as they were in the input.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  1 2 3 4 5 | 3 4 5 |
| 6  5 4 3 8 9 0 | 5 4 8 9 |

## Most Frequent Number

Write a program that:

* Read an **integer number** **N (N < 100) from the first line of the console**, which represents **size of the array**
* Read an **integer array** **with the given size** from the second line of the console
* **Integer numbers in the array will be in the range [0, 9]**
* Find the **most frequent number** in the given integer array
* Print the **most frequent number**

**Note:** In case of multiple numbers with the same maximal frequent, print all of them, ordered from smallest to largest, separated by space.

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 13  4 1 1 4 2 3 4 4 1 2 4 9 3 | 4 | The number 4 is the most frequent |
| 8  2 2 2 2 1 2 2 2 | 2 | The number 2 is the most frequent |

## Cartesian Product

Write a program that:

* Read an **integer number** **N (N < 100) from the first line of the console**, which represents **size of the array**
* Read an **integer array** **with the given size** from the second line of the console
* Find and print the **product of each of its elements with all elements**

**Example:**

For the array {1, 7, 3} the result would be: **{1\*1, 1\*7, 1\*3, 7\*1, 7\*7, 7\*3, 3\*1, 3\*7, 3\*3},** which gives us the array **{1, 7, 3, 7, 49, 21, 3, 21, 9},** so for the input **1 7 3,** the program should print **1 7 3 7 49 21 3 21 9.**

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 3  1 7 3 | 1 7 3 7 49 21 3 21 9 |
| 2  -1 4 | 1 -4 -4 16 |

## Closest Numbers

Write a program that:

* Read an **integer number** **N (N < 100) from the first line of the console**, which represents **size of the array**
* Read an **integer array** **with the given size** from the second line of the console
* Finds the **two closest (by value) integer numbers** in the array
* Prints the **absolute differenc**e between them

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 5  1 105 10 100 3 | 2 | The closest numbers are 1 and 3  abs(1 - 3) = abs (-2) = 2 |
| 9  1 2 3 4 5 6 7 8 9 | 1 | All numbers are exactly 1 unit apart |

## Array Rotation

Write a program that:

* Read an **integer number** **N (N < 100) from the first line of the console**, which represents **size of the array**
* Read an **integer array** **with the given size** from the second line of the console
* Read an **integer number** **from the third line of the console**, which represents **count rotations you have to perform**
* **One rotation is when the first element goes at the end (first element becomes last element)**
* Print the **resulting array elements**, separated by single space

### Examples

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

## Top Integers

Write a program that:

* Read an **integer number** **N (N < 100) from the first line of the console**, which represents **size of the array**
* Read an **integer array** **with the given size** from the second line of the console
* Find all the top integers in an array
* **Top integer is an integer that is bigger than all the elements to its right**
* Print **all top integers**, separated by single space

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 4  1 4 3 2 | 4 3 2 |
| 6  14 24 3 19 15 17 | 24 19 17 |
| 7  27 9 42 2 13 45 48 | 48 |

## Magic Sum

Write a program that:

* Read an **integer number** **N (N < 100) from the first line of the console**, which represents **size of the array**
* Read an **integer array** **with the given size** from the second line of the console
* Read an **integer number** **from the third line of the console**, which represents **magic sum**
* Print all unique pairs in an array of integers whose sum is equal to the given **magic sum**

**Note:** Here's how to generate all pairs for the first two numbers of an array. Use the same logic for the whole array:

A number on a black background

Description automatically generated

### Examples

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