# Lab: Lists: Simple List Processing

Problems for exercises and homework for the [“Programming Fundamentals Extended” course @ SoftUni](https://softuni.bg/courses/programming-fundamentals).

You can check your solutions here: [https://judge.softuni.bg/Contests/384/Lists-Simple-List-Processing-Lab](https://judge.softuni.bg/Contests/384/Arrays-Simple-Array-Processing-Lab).

## Sum List Items

Write a program, which reads a **list** of integers, calculates its **sum** and **prints** it.

The input consists of a **number** n (the number of items) + n integers, each as a separate line.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| **4**  1  2  3  4 | 10 |
| **5**  1  1  1  1  1 | 5 |
| **4**  2  -1  -2  8 | 7 |

### Hints

* First, read the number n.
* Read the integers in a for-loop.

## Multiply an List of Floats

Write a program to read **a list of floats**, a float p, multiply each item by p and print the resulting list.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1.2 3.0 12.3 4.56  4 | 4.8 12 49.2 18.24 |
| 6.0 8.8 1.2 -9.6  0.5 | 3 4.4 0.6 -4.8 |

### Hints

* Read the list
* **Loop through** the list, **multiplying each item** by **p**
* Finally, **print** the resulting list, using a **for** loop

## Smallest Item in List

Write a program to read **a list of integers**, find the **smallest item** and **print** it.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| **1** 2 3 4 | 1 |
| 3 2 9 **-9** 6 1 | -9 |
| -6 0 **-17** -1 | -17 |

### Hints

* **Loop through** the **integer list** until you find the **smallest item**

## Rotate List of Strings

Write a program to read **a list of strings**, **rotate** it to the right and **print** its rotated items.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| a b c d e | e a b c d |
| soft uni hi | hi soft uni |
| i r a b | b i r a |

### Hints:

* You can store the rotated list in a **second list** alongside the first one

## Count of Odd Numbers in List

Write a program to read **a list of integers** and find **how many odd items it holds**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| **1** -2 **3** 4 | 2 |
| **3** **9** **-9** -6 **1** -2 | 4 |
| 66 0 2 1 | 1 |

### Hints:

* You can check if a number is **odd** if you **divide them by 2** and check whether you get **a remainder of 1**.
* Odd numbers, which are negative, have a **remainder** of **-1**.

## Odd Numbers at Odd Positions

Write a program to read a list of integers and find how many **odd numbers** at **odd positions** the list holds. If there are no numbers, which match this criteria, **do not print anything**

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Explanation** |
| 2 **3** 5 2 7 **9** -1 **-7** | Index 1 -> 3  Index 5 -> 9  Index 7 -> -7 | Indexes: 0 **1** 2 3 4 **5** 6 **7**  Numbers: 2 **3** 5 2 7 **9** **-1** -7  Odd positions with odd numbers: 1, 5 and 7 |
| 2 **3** 55 2 4 **1** | Index 1 -> 3  Index 5 -> 1 | Indexes: 0 **1** 2 3 4 **5**  Numbers: 2 **3** 55 2 4 **1**  Odd positions with odd numbers: 1 and 5 |
| 5 0 1 2 | *(no output)* | Indexes: 0 1 2 3  Numbers: 5 0 1 2  Odd positions with odd numbers: **none** |

### Hints

* Positions are counted **from** **0** from left to right, so if for example the second item (**index 1**) is **odd**, then we **should** count it, and so on…
* Do **NOT** count odd numbers, which are at **even** positions (0, 2, 4, etc…)