# Lab: List Algorithms

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/426>.

***Note: for this exercise, you are not allowed to use ready functions for doing the algorithm. They are allowed only for reading the input and printing the result.***

## List Contains Item

Read a **list of integers** on the first line of the console and an integer **N** from the second line of the console and print whether the item is **contained** in the list. If it is, print “yes”, otherwise print “no”.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1 2 3 4 5  5 | yes |
| 8 7 7 9 6 2 2  11 | no |
| 99 7 8 6 2314 2  2314 | yes |

### Hints

* Read a text line from the console, split it by space, parse the obtained items as integers and convert them to list of integers.
* Scan through the whole list, item by item, until you either find the item, or reach the end of the list.
* Keep the result of the operation in a Boolean variable such as “isFound”.
* Finally, if the item is found (checking by the variable), print “yes” or “no”.

## Smallest Item in List

Read a **list of integers** on the first line of the console. After that, **find** the smallest item in the list and **print** it on the console.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1 2 3 4 5 | 1 |
| 9 8 7 82 78 13 | 7 |
| 78 77 1268 43 9 | 9 |

### Hints

* Read a text line from the console, split it by space, parse the obtained items as integers and convert them to list of integers.
* Traverse the whole list, item by item, putting the **smallest integer** up to that point into an integer variable called “smallestInt”. Finally, print smallestInt.

## Reverse List In-place

Read a **list of integers** on the first line of the console. After that, **reverse** the list in-place (as in, don’t create a new collection to hold the result, reverse it using only the original list). After you are done, **print** the reversed list on the console.

Note: You are **not** allowed to iterate over the list backwards and just print it,

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1 2 3 4 5 | 5 4 3 2 1 |
| 1 4 2 7 6 1 1 | 1 1 6 7 2 4 1 |
| 11 52 43 12 1 6 | 6 1 12 43 52 11 |

### Hints

* Iterate over **half** of the list (0…length / 2) and swap each item with its opposite index like so:
  + Index 0 gets swapped with -1
  + Index 1 gets swapped with -2,
  + Index 2 gets swapped with -3,
  + and so on…
* When you reach the **middle** of the list, it means you are done swapping items and are ready to print them.

## Sort List Using Bubble Sort

Read a **list of integers** on the first line of the console. After that, **sort** the list, using the [**Bubble Sort**](https://visualgo.net/sorting) algorithm.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5 3 4 1 2 | 1 2 3 4 5 |
| 11 872 673 1 2 | 1 2 11 673 872 |
| 11 52 43 12 1 6 | 1 6 11 12 43 52 |

## Sort List Using Insertion Sort

Read a **list of integers** on the first line of the console. After that, **sort** the list, using the [**Insertion Sort**](https://visualgo.net/sorting) algorithm.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5 3 4 1 2 | 1 2 3 4 5 |
| 11 872 673 1 2 | 1 2 11 673 872 |
| 11 52 43 12 1 6 | 1 6 11 12 43 52 |

## Insertion Sort Using List

Read a **list of integers** on the first line of the console. After that, **sort** the list, using the [**Insertion Sort**](https://visualgo.net/sorting) algorithm, but instead of doing it in-place, add the result **one by one** to a **list**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5 3 4 1 2 | 1 2 3 4 5 |
| 11 872 673 1 2 | 1 2 11 673 872 |
| 11 52 43 12 1 6 | 1 6 11 12 43 52 |

## Largest N Items

Read a **list of integers** on the first line of the console. On the next line, you will receive an **integer N**. After that, find and **print** the **largest N** **items** the list, sorted in **descending order**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5 3 4 1 2  3 | 5 4 3 |
| 11 872 673 1 2  2 | 872 673 |
| 11 52 43 12 1 6  4 | 52 43 12 11 |

### Hints

* A possible solution to this problem is:
  + Sort the list in **descending order**, using a sorting algorithm such as **Insertion Sort**,
  + Extract the first **N items** from it