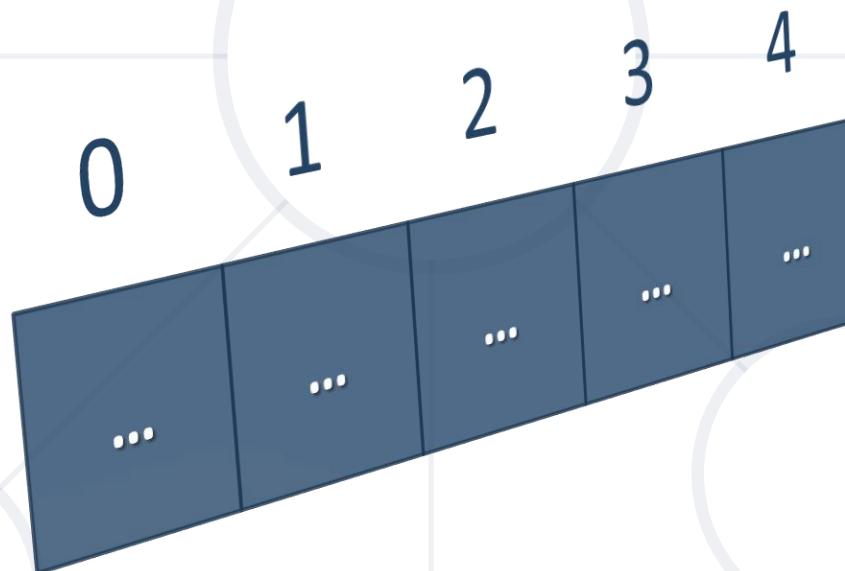


# Lists

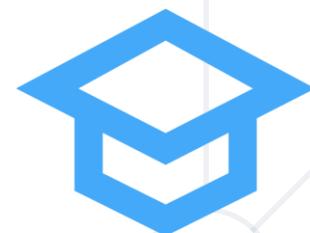
Processing Variable-Length Sequences of Elements



SoftUni Team

Technical Trainers

 Software University



SoftUni



Software University

<https://softuni.bg>

Have a Question?



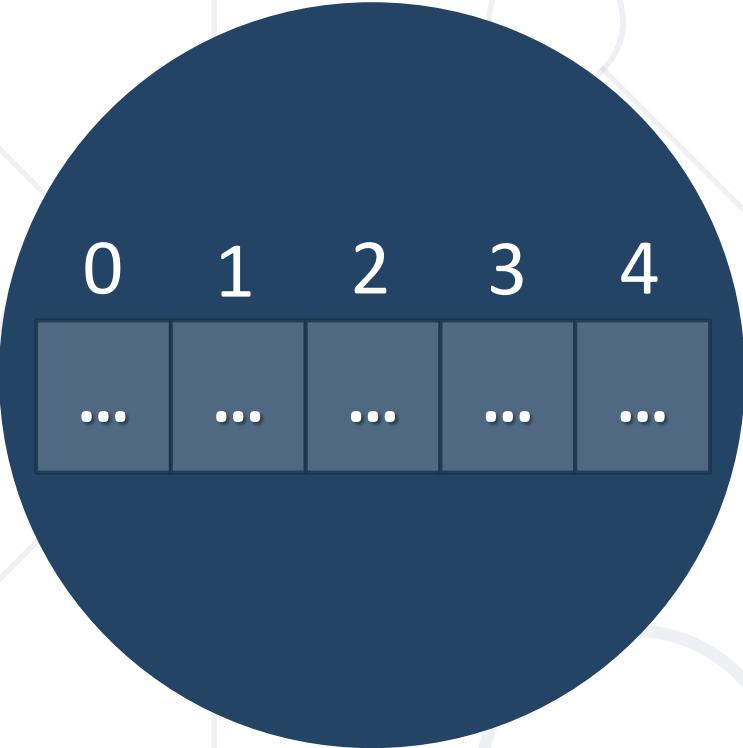
**sli.do**

**#fund-csharp**

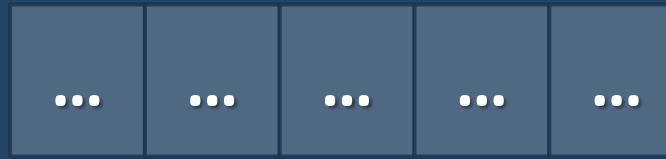
# Table of Contents

1. Lists
2. Reading Lists from the Console
3. Sorting Lists and Arrays





0 1 2 3 4



# Lists

# List<T> – Overview

- List<T> holds a list of elements of the same type



```
List<string> names = new List<string>();  
// Create an empty list of strings  
  
names.Add("Peter");  
  
names.Add("Maria");  
// Add elements  
  
foreach (var name in names)  
    Console.WriteLine(name);  
  
Console.WriteLine(string.Join(", ", names));  
// Print elements
```

# Creating Lists

- Use the **new** keyword
  - Create an empty list of integers

```
List<int> numbers = new List<int>();
```

- Using a target-type **new** expression

```
List<string> names = new() {"Peter", "Ana", "Maria"};
```



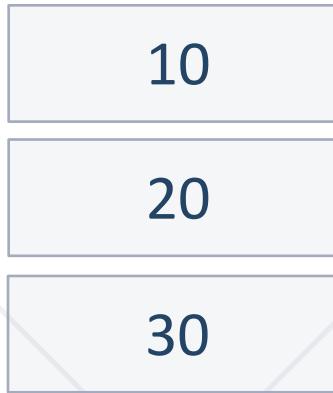
# List<T> – Basic Methods

- Provides operations to **add / insert / remove / find** elements
  - **Add(element)** – adds an element to the **List<T>**
  - **Count** – number of elements in the **List<T>**
  - **Remove(element)** – removes an element (returns **true / false**)

- **Insert(index, element)** – inserts an element to a given index
- **Contains(element)** – determines whether an element is in the list
- **Sort()** – sorts the array/list in ascending order

# Add() – Appends an Element

- We create an empty list and start adding elements
- The count increases each time we add an element



List<int>

Count:



# Remove() – Deletes an Element

- We remove an element from the List
- The count decreases each time we remove an element

List<int>

10
20
30

Count:

3

# Insert() – Inserts an Element at Position

- We insert an element at index 1
- Other elements' indices are changed upon insertion

-10

List<int>

10
30

Count:

3

# List<T> – Basic Methods Example

```
List<int> nums = new List<int>
{
    10, 20, 30, 40, 50, 60
};

nums.Remove(30);

nums.Add(100);

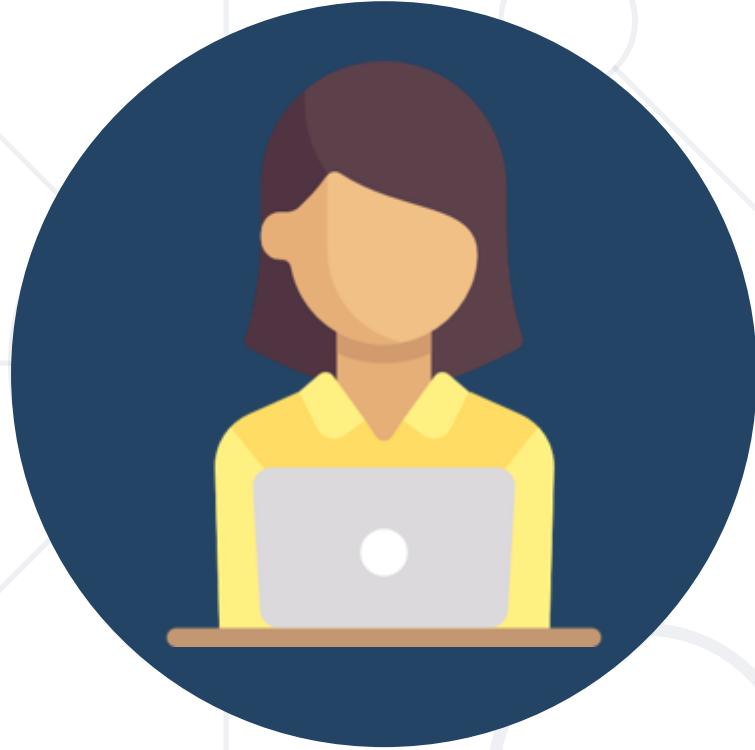
nums.Insert(0, -100);

Console.WriteLine(string.Join(", ", nums));

Console.WriteLine($"Count: {nums.Count}");
```



```
-100, 10, 20, 40, 50, 60, 100
Count: 7
```



# Reading Lists from the Console

## Using for Loop or String.Split()

# Reading Lists from the Console

- First, read from the console the list's **length**

```
int n = int.Parse(Console.ReadLine());
```

- Next, create a list of a given size **n** and read its **elements**

```
List<int> list = new List<int>();  
for (int i = 0; i < n; i++)  
{  
    int number = int.Parse(Console.ReadLine());  
    list.Add(number);  
}
```

# Reading List Values from a Single Line

- Lists can be read from a **single line of space separated values**

```
2 8 30 25 40 72 -2 44 56
```

```
string values = Console.ReadLine();
List<string> items = values.Split(' ').ToList();
List<int> nums = new List<int>();
for (int i = 0; i < items.Count; i++)
    nums.Add(int.Parse(items[i]));
```

Convert a collection into List

```
List<int> items = Console.ReadLine()
    .Split(' ')
    .Select(int.Parse)
    .ToList();
```

Read a List of integers

# Printing Lists On the Console

- Printing a list using a **for** loop

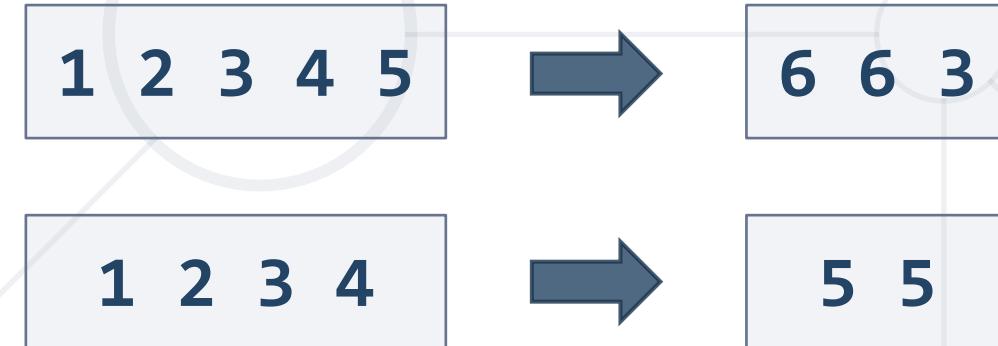
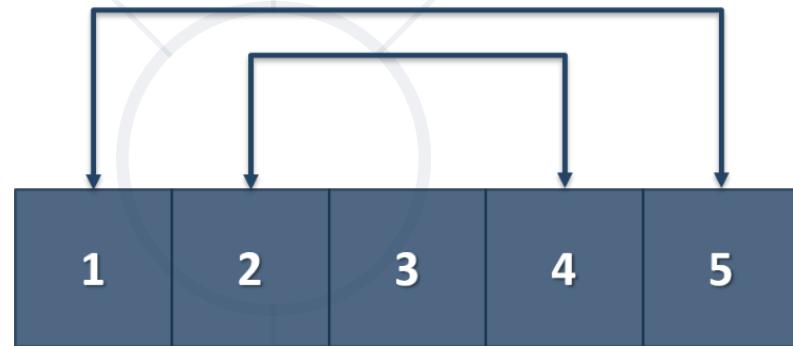
```
List<string> list = new List<string>() {  
    "one", "two", "three", "four", "five", "six"};  
for (int index = 0; index < list.Count; index++)  
    Console.WriteLine("arr[{0}] = {1}", index, list[index]);
```

- Printing a list using a **string.Join(...)**

```
List<string> list = new List<string>() {  
    "one", "two", "three", "four", "five", "six"};  
Console.WriteLine(string.Join(";", ", ", list));
```

# Problem: Gauss' Trick

- Write a program that sums all numbers in a list in the following order
  - $\text{first} + \text{last}, \text{first} + 1 + \text{last} - 1, \text{first} + 2 + \text{last} - 2, \dots, \text{first} + n, \text{last} - n$
- Examples



Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1210#1>

# Solution: Gauss' Trick

```
List<int> numbers = Console.ReadLine()
    .Split().Select(int.Parse).ToList();

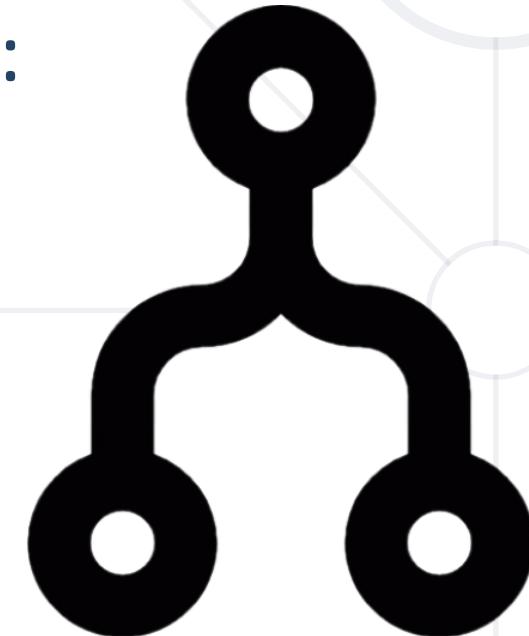
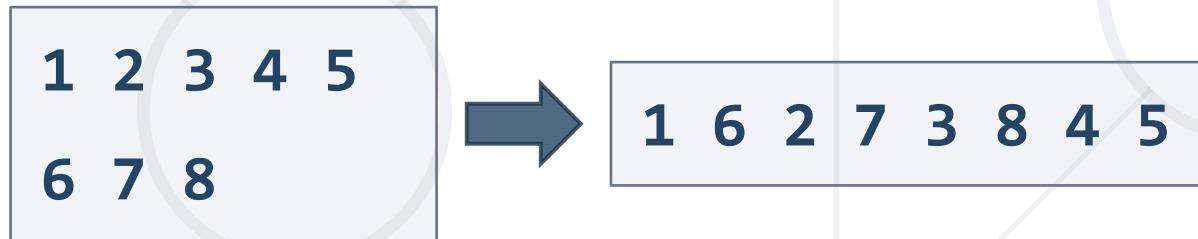
int originalLength = numbers.Count;

for (int i = 0; i < originalLength / 2; i++)
{
    numbers[i] += numbers[numbers.Count - 1];
    numbers.RemoveAt(numbers.Count - 1);
}

Console.WriteLine(string.Join(" ", numbers));
```

# Problem: Merging Lists

- You receive two lists with numbers. Print a result list, which contains the numbers from both of the lists
  - If the length of the two lists are not equal, just add the remaining elements at the end of the list:
  - `list1[0], list2[0], list1[1], list2[1], ...`



Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1210#2>

# Solution: Merging Lists

```
// TODO: Read the input

List<int> resultNums = new List<int>();

for (int i = 0; i < Math.Min(nums1.Count, nums2.Count); i++)
    // TODO: Add numbers in resultNums
    if (nums1.Count > nums2.Count)
        resultNums.AddRange(GetRemainingElements(nums1, nums2));
    else if (nums2.Count > nums1.Count)
        resultNums.AddRange(GetRemainingElements(nums2, nums1));
    Console.WriteLine(string.Join(" ", resultNums));
```

# Solution: Merging Lists

```
static List<int> GetRemainingElements(List<int> longerList,  
List<int> shorterList)  
{  
    List<int> nums = new List<int>();  
    for (int i = shorterList.Count; i < longerList.Count; i++)  
        nums.Add(longerList[i]);  
    return nums;  
}
```

Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1210#2>



# Live Exercises

Reading and Manipulating Lists



# Sorting Lists and Arrays

# Sorting Lists

- Sorting a list == reorder its elements incrementally: **Sort()**
- Items must be **comparable**, e.g., numbers, strings, dates, ...

```
List<string> names = new List<string>()
    {"Peter", "Michael", "George", "Victor", "John" };
names.Sort();           Sort in natural
                        (ascending) order
Console.WriteLine(string.Join(", ", names));
// George, John, Michael, Peter, Victor
names.Sort();
names.Reverse();        Reverse the sorted result
Console.WriteLine(string.Join(", ", names));
// Victor, Peter, Michael, John, George
```

# Problem: List of Products

- Read a number n and n lines of products. Print a numbered list of all the products ordered by name.
- Examples:

4  
Potatoes  
Tomatoes  
Onions  
Apples



1.Apples  
2.Onions  
3.Potatoes  
4.Tomatoes

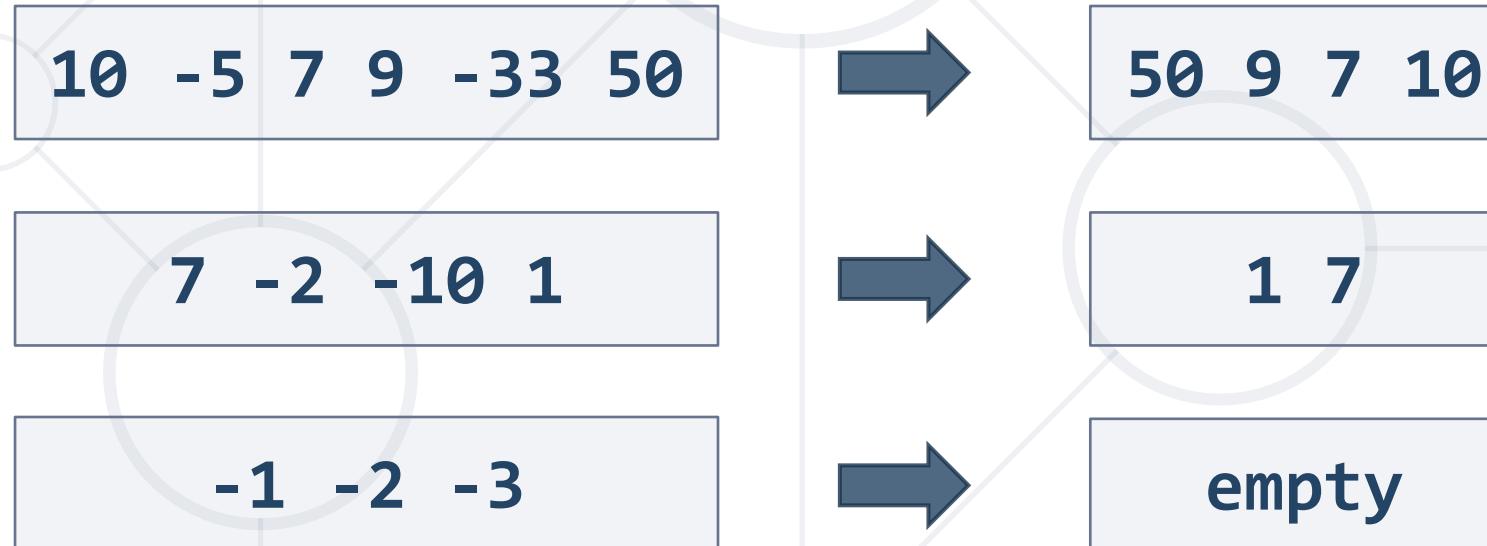
A  
Z

# Solution: List of Products

```
int n = int.Parse(Console.ReadLine());  
  
List<string> products = new List<string>();  
  
for (int i = 0; i < n; i++)  
{  
    string currentProduct = Console.ReadLine();  
    products.Add(currentProduct);  
}  
products.Sort();  
  
for (int i = 0; i < products.Count; i++)  
    Console.WriteLine($"{i + 1}.{products[i]}");
```

# Problem: Remove Negatives and Reverse

- Read a list of integers, remove all negative numbers from it.
  - Print the remaining elements in reversed order
  - In case of no elements left in the list, print "empty"



Check your solution here: <https://judge.softuni.org/Contests/Practice/Index/1210#4>

# Solution: Remove Negatives and Reverse

```
List<int> nums = // TODO: Read the List from the console.  
  
for (int i = 0; i < nums.Count; i++)  
    if (nums[i] < 0) { nums.RemoveAt(i--); }  
  
nums.Reverse();  
if (nums.Count == 0)  
    Console.WriteLine("empty");  
else  
    Console.WriteLine(string.Join(" ", nums));
```



# Live Exercises

## Sorting Lists

# Summary

- Lists hold a sequence of elements (variable-length)
- Can **add** / **remove** / **insert** elements at runtime
- Creating (allocating) a list: **new List<T>()**
- Accessing list elements by index
- Printing list elements: **string.Join(...)**



# Questions?



# SoftUni Diamond Partners



THE CROWN IS YOURS



# Trainings @ Software University (SoftUni)



- Software University – High-Quality Education, Profession and Job for Software Developers
  - [softuni.bg](http://softuni.bg), [about.softuni.bg](http://about.softuni.bg)
- Software University Foundation
  - [softuni.foundation](http://softuni.foundation)
- Software University @ Facebook
  - [facebook.com/SoftwareUniversity](https://facebook.com/SoftwareUniversity)



Software  
University



- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – <https://about.softuni.bg/>
- © Software University – <https://softuni.bg>

