# Lab: Arrays

Problems for in-class lab for the ["C# Fundamentals" course @ SoftUni](https://softuni.bg/trainings/3836/programming-fundamentals-with-csharp-september-2022)  
You can check your solutions in [Judge](https://judge.softuni.org/Contests/1202/Arrays-Lab)

## Day of Week

Enter a number in range 1-7 and print out the **word** representing it or "**Invalid Day!**". Use an **array of strings**.

using System;

namespace \_01.\_Day\_of\_Week

{

class Program

{

static void Main(string[] args)

{

string[] daysOfWeek = new string[7]

{

"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday","Sunday"

};

int day = int.Parse(Console.ReadLine());

if(day<1||day>7)

{ Console.WriteLine("Invalid day!"); }

else { Console.WriteLine(daysOfWeek[day-1]); }

}

}

}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1 | Monday |
| 2 | Wednesday |
| 10 | Invalid day! |

## Print Numbers in Reverse Order

Read **n** numbers and print them in reverse order, separated by a single space.

using System;

namespace \_02.\_Print\_Numbers\_in\_Reverse\_Order

{

class Program

{

static void Main(string[] args)

{

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

int[] numbers = new int[n];

for (int i = 0; i <n; i++)

{

numbers[i] = int.Parse(Console.ReadLine());

}

for (int i = numbers.Length-1; i >=0; i--)

{

Console.Write(numbers[i]+" ");

}

}

}

}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 3  10  20  30 | 30 20 10 |
| 3  30  20  10 | 10 20 30 |
| 1  10 | 10 |

### Hints

First, we need to read **n** from the console.



Create an **array of integers** with **n** size.



Read **n** numbers using for loop.



**Set** number to the corresponding **index**.



Print the array in reversed order.



## Rounding Numbers

Read an array of real numbers (space separated), round them in "**away from 0**" style and print the output as in the examples:

using System;

using System.Linq;

namespace \_03.\_Rounding\_Numbers

{

class Program

{

static void Main(string[] args)

{

double[] numbers = Console.ReadLine()

.Split()

.Select(double.Parse)

.ToArray();

for (int i = 0; i < numbers.Length; i++)

{

Console.WriteLine($"{numbers[i]} => {(int)Math.Round(numbers[i],MidpointRounding.AwayFromZero)}");

}

}

}

}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 0.9 1.5 2.4 2.5 3.14 | 0.9 => 1  1.5 => 2  2.4 => 2  2.5 => 3  3.14 => 3 |
| -5.01 -1.599 -2.5 -1.50 0 | -5.01 => -5  -1.599 => -2  -2.5 => -3  -1.50 => -2  0 => 0 |

## Reverse Array of Strings

Create a program that reads an **array of strings**, reverses it, and **prints** its elements. The input consists of a sequence of space-separated **strings**. Print the output on a single line (space separated).

using System;

using System.Linq;

namespace \_04.\_Reverse\_Array\_of\_Strings

{

class Program

{

static void Main(string[] args)

{

string[] input = Console.ReadLine()

.Split()

.ToArray();

for (int i = input.Length - 1; i >= 0; i--)

{

Console.Write(input[i]+" ");

}

}

}

}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| a b c d e | e d c b a |
| -1 hi ho w | w ho hi -1 |

## Sum Even Numbers

### Read an array from the console and sum only its even values.

using System;

using System.Linq;

namespace \_05.\_Sum\_Even\_Numbers

{

class Program

{

static void Main(string[] args)

{

int[] numbers = Console.ReadLine()

.Split()

.Select(int.Parse)

.ToArray();

int sum = 0;

for (int i = 0; i < numbers.Length; i++)

{

if (numbers[i] % 2 == 0)

{

sum += numbers[i];

}

}

Console.WriteLine(sum);

}

}

}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1 2 3 4 5 6 | 12 |
| 3 5 7 9 | 0 |
| 2 4 6 8 10 | 30 |

### Hints

First, we need to read the array.



We will need a variable for the sum.



Iterate through all elements in the array with for loop.



Check if the number at the current index is even.



Print the total sum.



## Even and Odd Subtraction

Create a program that calculates the difference between the **sum of the even** and the **sum of the odd** numbers in an array.

using System;

using System.Linq;

namespace \_06.\_Even\_and\_Odd\_Subtraction

{

class Program

{

static void Main(string[] args)

{

int[] numbers = Console.ReadLine()

.Split()

.Select(int.Parse)

.ToArray();

int sumEven = 0,sumOdd=0;

for (int i = 0; i < numbers.Length; i++)

{

if (numbers[i] % 2 == 0)

{

sumEven += numbers[i];

}

}

for (int i = 0; i < numbers.Length; i++)

{

if (numbers[i] % 2 != 0)

{

sumOdd += numbers[i];

}

}

Console.WriteLine(sumEven-sumOdd);

}

}

}

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 1 2 3 4 5 6 | 3 | Even: 2 + 4 + 6 = 12  Odd: 1 + 3 + 5 = 9  Result: 12 – 9 = 3 |
| 3 5 7 9 | -24 | Even: 0  Odd: 3 + 5 + 7 + 9 = 24  Result: 0 – 24 = -24 |
| 2 4 6 8 10 | 30 | Even: 2 + 4 + 6 + 8 + 10 = 30  Odd: 0  Result: 30 – 0 = 30 |

### Hints

First, we need to read the array.



We will need two variables – even and odd sum.



Iterate through all elements in the array with for loop.



Check the current number – if it is even, add it to the even sum, otherwise add It to the odd sum.



Print the difference.



## Equal Arrays

Read two arrays and determine whether they are identical or not. The arrays are **identical**, if all their **elements are equal**. If the arrays are identical, find the **sum of the elements of one of them** and print the following message to the console: "**Arrays are identical. Sum: {sum}**"

Otherwise, find the first index where the arrays differ and print the following message to the console: "**Arrays are not identical. Found difference at {index} index.**"

using System;

using System.Linq;

namespace \_07.\_Equal\_Arrays

{

class Program

{

static void Main(string[] args)

{

int[] A = Console.ReadLine()

.Split()

.Select(int.Parse)

.ToArray();

int[] B = Console.ReadLine()

.Split()

.Select(int.Parse)

.ToArray();

int diffInd = 0,sum=0;

bool areSame = true;

for (int i = 0; i < A.Length; i++)

{

if(A[i]!=B[i])

{

diffInd = i;

areSame = false;

break;

}

else

{

sum += A[i];

}

}

if(!areSame)

{ Console.WriteLine($"Arrays are not identical. Found difference at {diffInd} index"); }

else

{ Console.WriteLine($"Arrays are identical. Sum: {sum}"); }

}

}

}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 10 20 30  10 20 30 | Arrays are identical. Sum: 60 |
| 1 2 3 4 5  1 2 4 3 5 | Arrays are not identical. Found difference at 2 index |
| 1  10 | Arrays are not identical. Found difference at 0 index |

### Hints

First, we need to read two arrays.



Iterate through arrays and compare elements. If the elements are not equal print the required message and break the loop.



Think about how to solve the other part of the problem.

## Condense Array to Number

Create a program to read **an array of integers** and condense them by **summing** all adjacent couples of elements until a **single integer** remains.

For **example**, let us say we have **3** elements - **{2, 10, 3}**. We sum the first two and the second two elements and get **{2 + 10, 10 + 3} = {12, 13}**, then we sum all adjacent elements again. This results in **{12 + 13} = {25}.**

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 2 10 3 | 25 | 2 10 3 🡪 2+10 10+3 🡪 12 13 🡪 12 + 13 🡪 25 |
| 5 0 4 1 2 | 35 | 5 0 4 1 2 🡪 5+0 0+4 4+1 1+2 🡪 5 4 5 3 🡪 5+4 4+5 5+3 🡪 9 9 8 🡪 9+9 9+8 🡪 18 17 🡪 18+17 🡪 35 |
| 1 | 1 | 1 is already condensed to number |

### Hints

While we have more than one element in the array **nums[]**, repeat the following:

* Allocate a new array **condensed[]** of size **nums.length**.
* Sum the numbers from **nums[]** to **condensed[].**
  + condensed[i] = nums[i] + nums[i+1]
* nums[] = condensed[]

The process is illustrated below:



