# Lab: Arrays

You can check your solutions here: https://judge.softuni.bg/Contests/3171/Additional-Exercises.

## 1. Day of Week

Enter a **day number** [1…7] and print the **name** (in English) or "**`Invalid day!`**"

### Examples

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

## 2. Print Numbers in Reverse Order

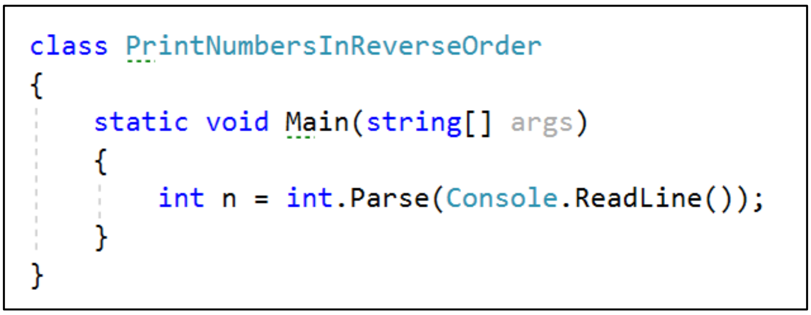
Read **n** numbers and print them in **reversed order**.

### Examples

| Input | Output | Input | Output |
|---|---|---|---|
| 3<br>10<br>20<br>30 | 30 20 10 | 1<br>10 | 10 |

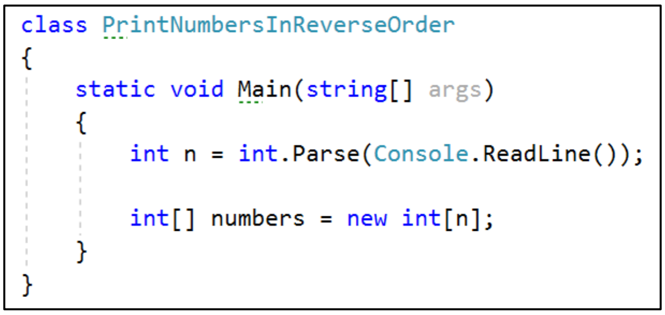
### Hints

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



```
class PrintNumbersInReverseOrder
{
    static void Main(string[] args)
    {
        int n = int.Parse(Console.ReadLine());
    }
}
```

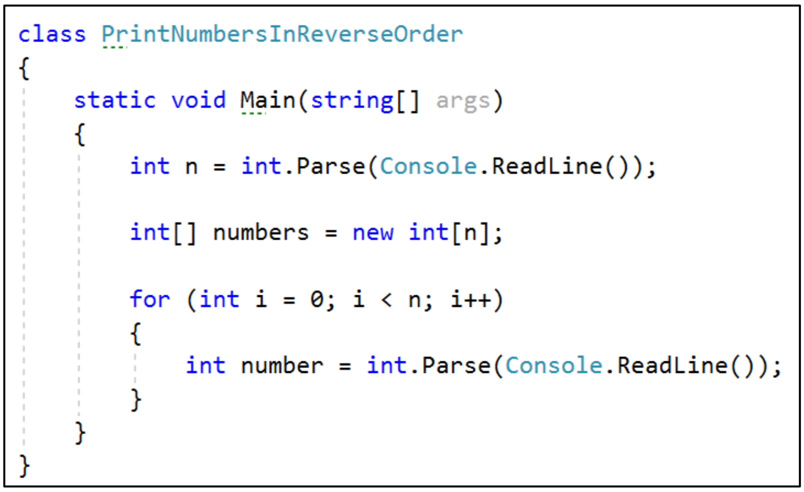
Create an **array of integer** with **n** size.



```
class PrintNumbersInReverseOrder
{
    static void Main(string[] args)
    {
        int n = int.Parse(Console.ReadLine());

        int[] numbers = new int[n];
    }
}
```

Read **n** numbers using for loop.

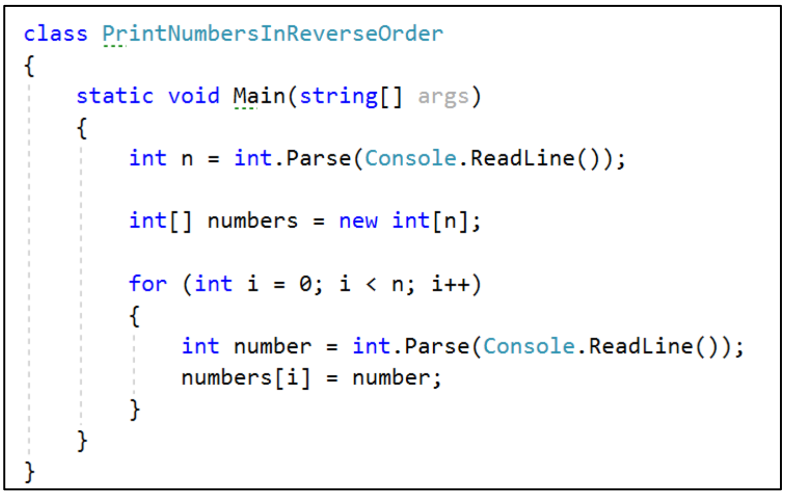


```
class PrintNumbersInReverseOrder
{
    static void Main(string[] args)
    {
        int n = int.Parse(Console.ReadLine());

        int[] numbers = new int[n];

        for (int i = 0; i < n; i++)
        {
            int number = int.Parse(Console.ReadLine());
        }
    }
}
```

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

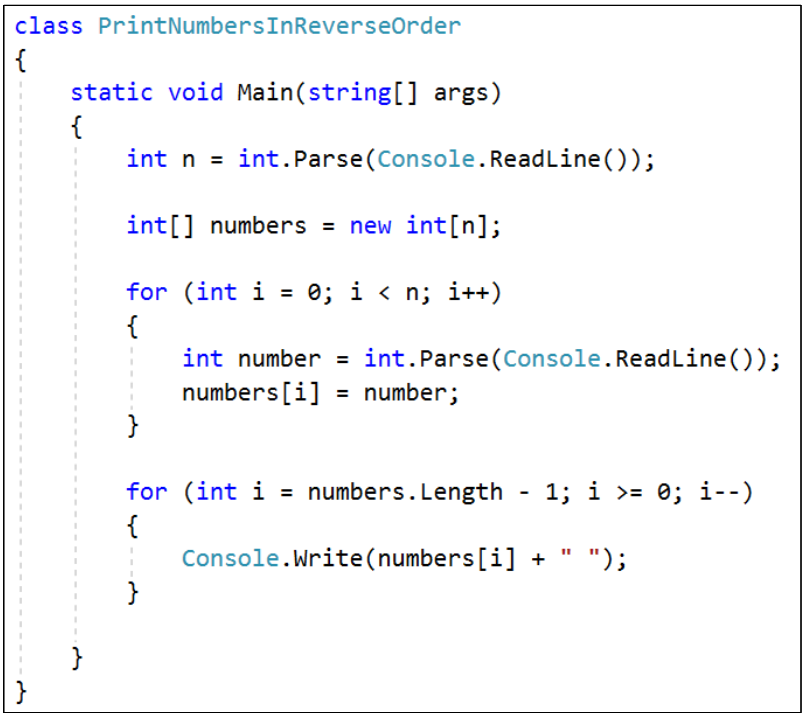


```
class PrintNumbersInReverseOrder
{
    static void Main(string[] args)
    {
        int n = int.Parse(Console.ReadLine());

        int[] numbers = new int[n];

        for (int i = 0; i < n; i++)
        {
            int number = int.Parse(Console.ReadLine());
            numbers[i] = number;
        }
    }
}
```

Print the array in **reversed order**.



```
class PrintNumbersInReverseOrder
{
    static void Main(string[] args)
    {
        int n = int.Parse(Console.ReadLine());

        int[] numbers = new int[n];

        for (int i = 0; i < n; i++)
        {
            int number = int.Parse(Console.ReadLine());
            numbers[i] = number;
        }

        for (int i = numbers.Length - 1; i >= 0; i--)
        {
            Console.Write(numbers[i] + " ");
        }

    }
}
```

# 3. 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:

## Examples

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

# 4. Reverse Array of Strings

Read an **array of strings** (**space** separated values), **reverse it** and **print** its elements:

## Examples

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

# 5. Sum Even Numbers

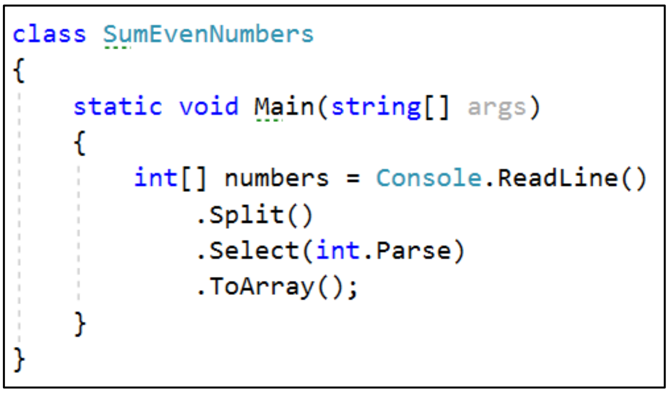
**Read an array** from the console and **sum** only the **even numbers**.

## Examples

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

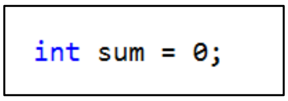
## Hints

First, we need to **read the array**.



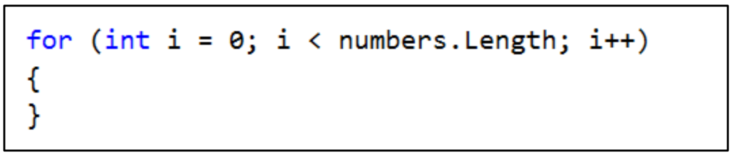
```
class SumEvenNumbers
{
    static void Main(string[] args)
    {
        int[] numbers = Console.ReadLine()
            .Split()
            .Select(int.Parse)
            .ToArray();
    }
}
```

We will need a **variable** for the **sum**.



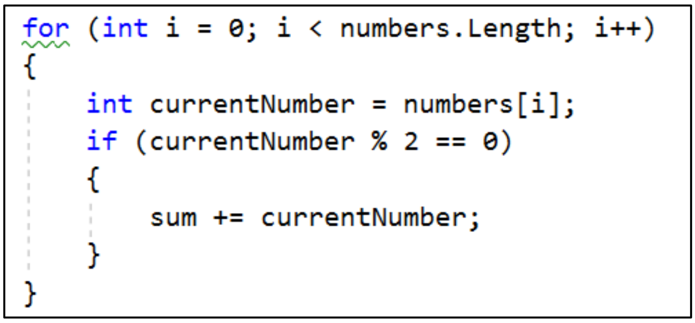
```
int sum = 0;
```

**Iterate** through all elements in the array with **for loop**.



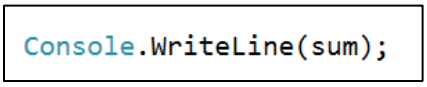
```
for (int i = 0; i < numbers.Length; i++)
{
}
```

Check if the number at **current index** is **even**.



```
for (int i = 0; i < numbers.Length; i++)
{
    int currentNumber = numbers[i];
    if (currentNumber % 2 == 0)
    {
        sum += currentNumber;
    }
}
```

Print the **total sum**:



```
Console.WriteLine(sum);
```

## 6. Even and Odd Subtraction

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

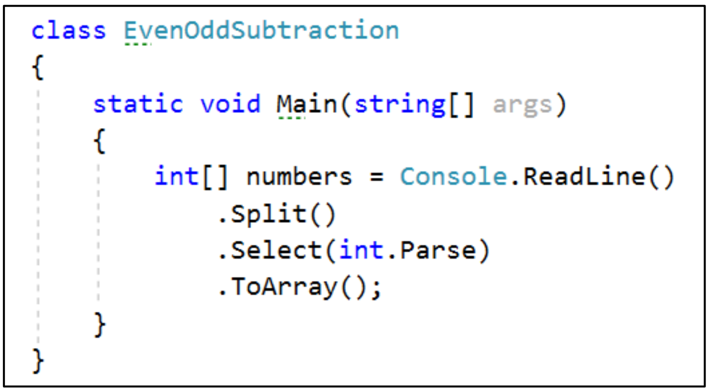
### Examples

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

| | | |
|---|---|---|
| | | Odd: 0<br>Result: 30 – 0 = 30 |

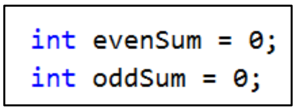
## Hints

First, we need to **read the array**.



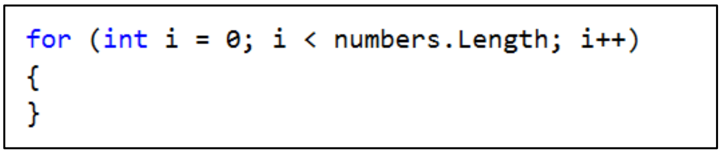
```
class EvenOddSubtraction
{
    static void Main(string[] args)
    {
        int[] numbers = Console.ReadLine()
            .Split()
            .Select(int.Parse)
            .ToArray();
    }
}
```

We will need **two variables** – **even** and **odd sum**.



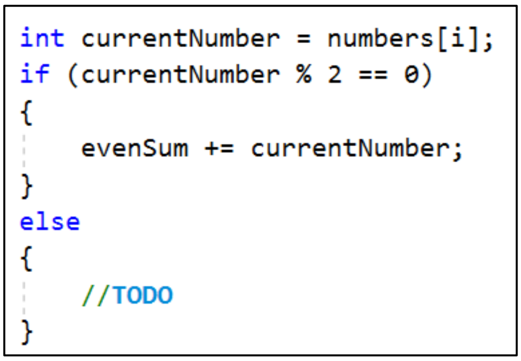
```
int evenSum = 0;
int oddSum = 0;
```

**Iterate** through all elements in the array with **for loop**.



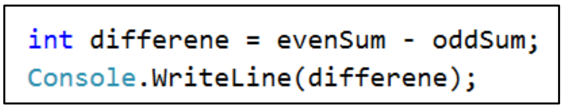
```
for (int i = 0; i < numbers.Length; i++)
{
}
```

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



```
int currentNumber = numbers[i];
if (currentNumber % 2 == 0)
{
    evenSum += currentNumber;
}
else
{
    //TODO
}
```

**Print the difference**.



```
int differene = evenSum - oddSum;
Console.WriteLine(differene);
```

# 7. Condense Array to Number

Write a program to read **an array of integers** and **condense** them by **summing** adjacent couples of elements until a **single integer** is obtained. For example, if we have 3 elements {2, 10, 3}, we sum the first two and the second two elements and obtain {2+10, 10+3} = {12, 13}, then we sum again all adjacent elements and obtain {12+13} = {25}.

## Examples

| Input | Output | Comments |
|---|---|---|

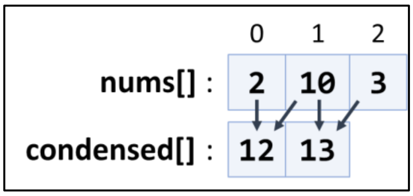
|  |  |  |
|---|---|---|
| `2 10 3` | `25` | `2 10 3 → 2+10 10+3 → 12 13 → 12 + 13 → 25` |
| `5 0 4 1 2` | `35` | `5 0 4 1 2 → 5+0 0+4 4+1 1+2 → 5 4 5 3 → 5+4 4+5 5+3 → 9 9 8 → 9+9 9+8 → 18 17 → 18+17 → 35` |
| `1` | `1` | 1 is already condensed to number |

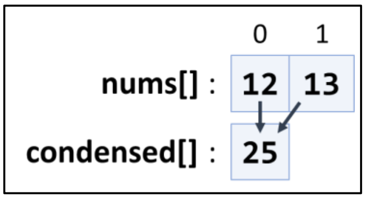
## Hints

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

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

The process is illustrated below:


0
1
2
nums[] :
2
10
3
condensed[] :
12
13


0
1
nums[] :
12
13
condensed[] :
25