# Console Input - Output – Exercise

The goal of this exercise is to practice **printing values to the console and reading from it**. Your task is to create a solution containing C# projects.

## Printing to the Console

Write a program that reads 3 numbers: an integer a (0 ≤ a ≤ 500), a floating-point b and a floating-point c and **prints them in 4 virtual columns** on the console. Each column should have a width of 10 characters. The number a should be printed in **hexadecimal, left aligned**; then the number a should be printed in binary form, padded with zeroes, then the number b should be **printed with 2 digits after the decimal point, right aligned**; the number c should be **printed with 3 digits after the decimal point, left aligned**. Examples:

|  |  |  |  |
| --- | --- | --- | --- |
| **a** | **b** | **c** | **result** |
| 254 | 11.6 | 0.5 | |FE |0011111110| 11.6|0.500 | |
| 499 | -0.5559 | 10000 | |1F3 |0111110011| -0.56|10000.000 | |
| 0 | 3 | -0.1234 | |0 |0000000000| 3|-0.123 | |

1. Declare 3 variables: an **integer** '**a'** and two **double** variables '**b'** and '**c'**.
2. **Parse** the **input** from the console by calling the **double.Parse()** method and pass the **Console input** as parameter (**Console.ReadLine()** ).
3. **Print** the **values** of the variables with one single **Console.WriteLine()** method and use **formatted strings**:
   1. To format a number to **hexadecimal** use {**0:X**};
   2. To format a number to **binary** use **Convert.ToString(number, 2)**. Then to add zeros to the left side of the number use **.PadLeft(10, '0')**;
   3. To format a number **rounded** to **2/3 decimal** points use {**0:F2/3**} or {**0.00/0**};
   4. To align numbers **10 digits** to the **right** or to the **left** use {**0, ±10**};