Exercises about methods

Exercise 07.01: C# Methods

- https://www.w3schools.com/cs/exercise.asp?filename=exercise_methods1
  - o Exercise 1.
  - o Exercise 2.
  - o Exercise 3.
  - o Exercise 4.

Exercise 07.02: Counting up and down (be lazy)

- Ask a positive number, no decimals, lower than 10.
  - o Ask till a correct input is given.
- Print lines according this rule.
  - o Start printing a line with 1.
  - o Continue printing, every time one digit more, till you have the asked number.
  - o Then continue printing a line, every time one digit less, till 1.
- Use a method "PrintLine" that writes it to the console.
  - o This method should put only one line to the console.

Example of result given a 4

```
1
12
123
1234
123
12
1
```

Notes

....
....
....
....
....
....
....

COPY PASTE)

Exercise 07.03: Overloading with greetings

- Create a method that will greet a person.
- Use that method several times in the Main() method.
- Two parameters, both optional.
  - First parameter is the gender of the person as string.
    - Possible values.
      - "Male".
      - "Female" (default value).
  - Second parameter is an hour.
    - Possible values between 0 and 23 (borders included).
    - Default value is the current hour (Now.Hour).
- When a parameter is not correct.
  - Show the message "Error in greeting".
- When both parameters are correct (or having the default value).
  - The hour.
    - Between 0 and 7 → "Good night".
    - Between 8 and 11 → "Good morning".
    - Between 12 and 17 → "Good afternoon".
    - Between 18 and 23 → "Good evening".
  - The gender.
    - "Male" → "my lord".
    - "Female" → "my lady".
- Sentence ends with a "." and has a comma after the greeting.

Expected result

- Greetings("Male", 15) → Good afternoon, my lord.
- Greetings() at 20 o'clock → Good evening, my lady.
- All different types of calling Greetings() should be tried out.

Notes

....
....
....
....
....
....
....

COPY PASTE)

Exercise 07.04: Get a maximum

- Write a method "GetMaximumOf2" with input 2 decimal numbers.
  - The method should return the biggest number.
- Write a method "GetMaximumOf3" with input 3 decimal numbers.
  - The method should return the biggest number using the method "GetMaximumOf2".
- Write a method "GetMaximumOf4" with input 4 decimal numbers.
  - The method should return the biggest number using the method "GetMaximumOf2".
- Write code in the Main() method that proves that your methods work as planned.

Expected result given the values

- GetMaximumOf2 of 10 and 8 should return 10.
- GetMaximumOf3 of 10; 8 and 12 should return 12.
- GetMaximumOf4 of 10; 8; 12 and 17,5 should return 17,5.

Notes

....

....

....

....

....

....

....

COPY PASTE)

Exercise 07.05: Freezing to death

- Ask a decimal number in the console.
  - Ask till a correct input is given. E.g., a text is not allowed.
- Given that number, we will calculate with it.
- We will create 2 methods that give back a decimal.
  - Method 1: FahrenheitToCelsius.
  - Method 2: CelsiusToFahrenheit.
  - Both results are to 6 decimals accurate.
- You must use both methods to show a result in the console like below:
  - Your given number was xxx.
  - xxx °C is yyy °F.
  - xxx °F is zzz °C.

How to calculate Celsius to Fahrenheit

- Fahrenheit = (Celsius * 9 / 5) + 32.

How to calculate Fahrenheit to Celsius

- Celsius = (Fahrenheit – 32) * 5 / 9.

Expected result given the value 30,5 and 100

```
Your given number was 30,5
30,5 °C is 86,9 °F
30,5 °F is -0,833333 °C

Your given number was 100
100 °C is 212 °F
100 °F is 37,777778 °C
```

Variant 1

- The lowest possible temperature in °C is -273.15, for °F it is -459.67.
- When this is the case in your calculation, show the lowest possible.

Notes

....
....
....
....
....
....
....

COPY PASTE)

Exercise 07.06: Having a nice time

- In the console you will ask a time as a text in the format "00:00".
  - As long as the time is incorrect, show the reason why it is not accepted and re-ask the time.
- What must be checked and in what order?
  - The text is exact 5 characters long.
  - The third character must be a ":".
  - The first 2 digits are between 0 and 23 (borders included).
  - The last 2 digits are between 0 and 59 (borders included).
- Every check is a different method that checks 1 specific criteria.
- Every method returns a boolean value.
  - True, the value is correct.
  - False, the value is incorrect.
- The moment one criteria fails, you show a nice error message and re-ask until a given time is correct.
  - Only one error message is shown at every attempt.

Expected result given the values

- 00:0 → Text is must be exactly 5 characters long.
- 00:120 → Text is must be exactly 5 characters long.
- 00-00 → The separator between hours and minutes must be ":".
- ab:cd → The hours must be between 00 and 23.
- 55:55 → The hours must be between 00 and 23.
- 10:cd → The minutes must be between 00 and 59.
- 10:59 → Thank you.

Notes

....
....
....
....
....
....
....

COPY PASTE)

Exercise 07.07: Getting a maximum again

Starting point is the result of 07.04.

- Use overloading for the method GetMaximum.
- Write a method "GetMaximum" with input 2 decimal numbers.
  - The method should return the biggest number.
- Write a method "GetMaximum" with input 3 decimal numbers.
  - The method should return the biggest number using the method "GetMaximum".
- Write a method "GetMaximum" with input 4 decimal numbers.
  - The method should return the biggest number using the method "GetMaximum".
- Write code in the Main() method that proves that your methods work as planned.

Expected result given the values

- GetMaximum of 10 and 8 should return 10.
- GetMaximum of 10; 8 and 12 should return 12.
- GetMaximum of 10; 8; 12 and 17,5 should return 17,5.

Notes

....
....
....
....
....
....
....

COPY PASTE)

Exercise 07.08: Having a nicer time

This example looks a lot on 07.06, but it is a different way of processing the information, but some parts can be reused.

- Ask hours thru the console.
  - Write a method for asking a correct value through the console.
  - Start with checking if the given hours are numeric. (Use a method).
  - Check also if the entered text is between 0 and the possible maximum.
  - More details are written below.
- Ask minutes thru the console, using the same method as before.
  - Obvious it uses the same numeric checks, but with a different maximum.
- Now you have 2 correct numbers. One for the hour and one for the minutes.
- Show the 2 numbers in this way:
  - The hours in 2 digits.
  - The minutes in 2 digits.
  - A ":" in between.

Expected Result

- When 2 and 5 are entered as correct numbers.
  - The time 02:05 is shown.
- When 0 and 0 are entered as correct numbers.
  - The time 00:00 is shown.
- When 12 and 43 are entered as correct numbers.
  - The time 12:43 is shown.
- When 23 and 59 are entered as correct numbers.
  - The time 23:59 is shown.

How to check if a text is numeric?

There are several ways to do this. In this exercise we will write the code by ourself (normally you don't do it this way).

- Input: Text.
- Action: Check if all characters are digits:
  - Loop thru all the characters.
  - Only 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9 are allowed.
- Output: Boolean value.
  - If all digits, return true.
  - If not all digits, return false.

How to ask a numeric value that is not bigger than a maximum thru the console?

Use a method, that accepts a text, a possible maximum and returns a number.

- Input: The text that is shown on the screen.
- Input: A maximum.
- Action: Is the entered text a numeric value.
- Action: Is the number between 0 and the possible maximum. Borders included.
- Action: Re-ask till you have an acceptable number.
- Output: An acceptable number.

Notes

....
....
....
....
....
....
....
....

COPY PASTE)

Exercise 07.09: The guessing game (I will play it to test)

- Let the software generate a number (no decimals) between 1 and 99 (borders included).
- This is the number that you want to guess in 5 attempts.
- Show on the console this text:
  - Guess the number (attempt xx):
    - xx is the number of the current attempt.
- Write a method to compare the guessed number, with the generated number.
  - Input: guessed number.
  - Input: number to compare with.
  - Action: Is the guessed number a numeric value.

	Use try … catch *Use Convert.ToByte()*

  - Action: Compare the guessed number with the random generated number.
  - Output: When equal, return 9.
  - Output: When bigger, return 1.
  - Output: When smaller, return -1.
  - Output: The input was not a number, return 0.
- When the number of guesses is bigger than 5, show the message that you have lost the game.
- You now have the output of the check, and you must act on it.
- When 9, show a message that you win the game in xx attempts.
    - xx is the number of guesses you needed.
- When 1, show the message that you must guess lower than xx.
    - xx is the last guess.
  - Increment the number of guesses.

- When -1, show the message that you must guess higher than xx.
  - xx is the last guess.
  - Increment the number of guesses.
- When 0, show the message that you must enter a number.
  - Number of guesses is not changed.

Variant 1

- Ask before the game how many times you want to play the game.
- Show your score afterwards.
  - 4 wins on 10 games.

Variant 2

- Remember all the guesses you have made in the game.
- Suppose you have entered 10, and the game told you that the number must be higher.
- If the player now enters 7, the game must respond with:
  - You fool, playing this way, you will never win.
- All situations with wrong entries must be covered.
  - 7 → higher.
  - 40 → lower.
  - 50 → You fool, playing this way, you will never win.
  - 6 → You fool, playing this way, you will never win.
  - 30 → You have lost the game. (If it was not 30)
  - 30 → You have won the game with 5 attempts. (If it was 30)

Variant 3

- Variant 1 and 2 together.

Notes

....
....
....
....
....
....
....

COPY PASTE)

Exercise 07.10: Refactoring stuff to make it more clear

Reorganize / refactor your code of exercise 06.16 using methods.

The goal of this exercise is to compare your solution with only a Main() method, with your new solution.

I want you to learn that working with Methods, should be the standard way to go. What methods do you have? Are there other solutions?

Do you have tests that tests your result of the methods on correctness?

Notes

....
....
....
....
....
....
....

COPY PASTE)

Exercise 07.11: The Euclid GCD (What the F**k)

This algorithm is a technique to calcuate the greatest common divisor (gcd) of two given positive integers. It is called Euclidean algorithm.

- You ask for a number, no decimals.
  - E.g. 12
- You ask for another number, no decimals.
  - E.g. 18
- 12 has the divisors 12, 6, 4, 3, 2 and 1.
- 18 has the divisors 18, 9, 6, 3, 2 and 1.
  - The greatest common divisor is 6.

Below the technique on how you can find the greatest common divisor without dividing at all. Solve this with a method in a loop.

- You start with 2 numbers (12 and 18)
- Loop this routine until both numbers are equal. (use for or while)
  - Subtract the smallest number from the biggest number. (18 – 12 = 6)
  - Change the biggest number of the 2 into the result of the subtraction.
    - So you get 12 and 6.
- When you exit the loop, the numbers you get is the greatest common divisor.

Variant 1

Solve this with a recursive method.

A method that is calling itself. So no loop (for, while) needed.

Notes

....
....
....
....
....
....
....

COPY PASTE)