Workshop CS

Requirements:

To compile and run C#:

- On linux:

```
sudo dnf/apt install mono-update
mcs -out:a.exe test.cs
mono a.exe
```

- On Windows:
  - https://learn.microsoft.com/en-us/visualstudio/get-started/csharp/tutorial-console?view=vs-2022
  - Stop at step 4 and select .NET 6.0

Exercise n°0: Hello World!

What would be learning a new language without starting with a 'Hello World!' ?

File: ex00.cs

Exercise n°1: Simple Concatenate

Create a simple program to :

- Takes two string as a **user input**.
- Prints the **length** of the **two concatenated strings**.

File : ex01.cs

> Hint : Remember to use google :)

Exercise n°2: Introduction to classes

You did a great job for now, but what you did could have been done in C, or any other languages. The difference with C#, is you can implement ✨classes✨.

Classes are used to define *Objects*.

>> "But what's an object ????"

Let's take an example, with a *player* in a video game.

A player has health points, xp, a username, etc. In other words, individual players all have the same attributes, but theses attributes are not the same for each player.

Here is an example of a class:

```
class Player
{
    public string username;
    public int health;
    public double xp;
}
```

What you have to do:

Complete the following code to:

- Create a new player called "Francis", with 20 health points, and 10.0 xp.
- Print the attributes of the player: Username, health, xp.

File ex02.cs

> Hint : If you are struggling, maybe there is something missing in the Player class, linked to "private"...

Bonus for geniuses:

If you have completed all the previous tasks, well done!

If there is some time left, you can try doing this bonus exercise, which might be a little harder than the rest :)

Good luck!

What you have to do:

You are provided with a text file named "words.txt" containing a series of words separated by spaces. Your task is to write a C# program that reads this file, and calculates the following statistics:

- Total number of words
- Number of unique words
- Word frequency

Example:

words.txt :

```
banana orange lemon banana kiwi banana orange
```

output :

```
Total number of words: 7
Number of unique words: 4
Word frequency:
 - banana: 3
 - orange: 2
 - lemon: 1
 - kiwi: 1
```

Files: bonus/bonus.cs & bonus/words.txt