Standard of code

In order to have an understandable code, there is standard of code in the group.
The standard has been adopted at the end of the project, that was maybe not the best strategy and that is also why not all codes are translated using these standards. In fact in order to understand the meaning of translated codes in standard these will be explained here.

A brief recall

The standard of code is used for many advantages and is proper too each groups or team. It makes the code understandable and also coherent. All team members should have the same language to be understood and we should adopt the same way of thinking and of writing like this when the code is released, everyone can see the group as only one member writing the code. This is also a way of having a good cohesion in the group.
In fact this is not only for aesthetic format, it's also for optimisation and for readability. If someone want to work on the project, in one overview he can know what is concerned and why we use such a variable, in what type of function and what is the usage of such a variable / function.

Variable Names

Instead of using meaningless variable names, or too long, we need to describe where it come from and also what is the type of variable:

- is it a member of a class?
- a user input?
- a database class input?

For each possibility there is a variable name assigned with a prefix that makes the code understandable.

Member of a class should start with m_ and then the name of the variable.
User input should start with u_ and then the name of the variable.
Database class input should start with d_ and then the name of the variable.

All the variables name should be in lower case and if needed the word should be separated by underscores.

Function Names

in the same way of thinking, function names should be fun lower case and with underscores separating the words if needed.

If the function is an initialisation step, then it should be prefixed by an _ and the optimal name is __init__. It is more readable and also more understandable for someone coming from other languages using preprocessors.

In the classes, the functions should all be in public , but sometimes protected is useful, not to protect the function but to explain a special behaviour, such as initiation which is proper to this class or special running process.

Classes

The classes should be used as soon as we can. This is a way of separating the code and make it more readable. It doesn't affect the way of the program run but it increase readability adaptability and portability.

Comments

Every lines, or almost every line should be commented is they have special behaviour, type or way of processing data.
The comment is here to give an overview on what it is happening in the code.

Every classes should be commented inside for constructors and desctructors, mutators and accessors.
Before each class, we should have a big comment explaining, what is the class, the usage, and also explaining every functions and especially constructors.

Before each files, we should have copyright and also the name of the developper. In a way of thinking as professional developer, if a future developer want to contact you because something is wrong or he is unable to understand the code (for example a code without comment), at least he can contact you by email just by looking at the first lines of a file.