**15.- TERAN ROBAYO CAMILA**

**Clean Code**

**A Handbook of Agile**

**Software Craftsmanship**

"In short, a programmer who creates clean code is an artist who can transform a blank canvas into an elegant code system." (Martin, 2009)

**Chapter 1 and 2**

Being a programmer and being a good programmer, no doubt both are on the same path but there is a big difference between the two, and we clearly notice it in their code. A code that is understandable to other programmers must be rigorous, formal, precise, detailed, that is, a clean code that meets the given requirements.

Like a clean code there is an incorrect code, where it is not possible to comply with an understandable structure, it does not comply with certain standards and the respective tests are not performed, thus generating a great loss of time and productivity. What causes this code is that other programmers cannot then modify it. Why at the end of your program does it end with a bad code? People are experts in looking for excuses and one of them are the requirements that change, the delivery time, etc. and we don't realize that the programmer must know how to act in all these situations and look for a solution, focus on his code and deliver in a correct and functional way; Without fear and without excuses, defend your code until the end and prove that it is a good code. What causes an incorrect code is more errors, if the problem is not solved from the beginning it will continue until all the code is finished.

A good programmer knows how to create a code suitable for changes and improvements (clean code), taking into account that the code must be detailed, understandable, organized and essential; We speak in this way of an elegant, efficient, simple and direct code, it fulfills what we expected. Having this type of code helps us so that over time the necessary changes can be made without any conflict or error and thus improve its version.

The first factor that we must take into account when creating a code is to write the names properly, this will allow other programmers to have no difficulty understanding the code. Their names must be clear, precise and specific so as not to cause any confusion and limit comments. When giving a name this should give us information about what we are doing, what is that variable, what function it is, what name I am looking for, enter other things and it is important to comply with a good structure when giving names, that is, use uppercase if it is a space or underscore, avoid encoders, use names with computer terms, the names must be different from each other to avoid confusion and thus help other programmers understand the context of the program and

be readable.

**Chapter 3 and 4**

Using functions in a code allows it to not be too long and makes it easy to verify that the code meets your requirements, and also allows the programmer to better understand the code. To be able to create really useful functions, we must take into account that the name must cover everything that the function will perform and that makes it easier for the programmer to understand the context of the code.

It is important not to mix details with essential concepts, it is better to raise from the beginning the direct concept that is going to be made and forget the details since the programmer just by reading the precise name of the function will understand. The descending rule explains that you should read from top to bottom to have a better understanding of the steps to follow that the code will perform, in other words, it is like descending on a ladder and going step by step. The extension of a function is very important since they cannot be extensive, the less lines they have, the better for the programmer, that is why a function must be concrete, to perform a single task. The correct name of a function improves its explanation and avoids confusion at the time of programming.

The use of comments may be unnecessary in our code, if the name of each variable and function is well described, adding comments is creating a bad code. If comments are used, they should be the minimum possible and these should be brief and precise.

If we need to return to a function to improve or modify, we use a reminder comment with the word TODO.