What is abstraction and why is it important?

Abstraction is one of the principles of Object-Oriented Programming.

This principle establishes that our design must simplify code complexity by separating large classes, functions, or methods into simpler, more focused ones. The main objective of abstraction is to “hide” complex implementation and give the user/developer simple methods to work with.

The benefits of abstraction are:

- Simplify methods: When writing our functions/methods, we implement abstraction by thinking: “How many tasks this method is doing”. The answer to that question must be “one”. That way, we will divide complex methods into various related methods.

- Improve code readability: By hiding complex implementations and avoiding multi-task methods our code is more organized and easier to read.

- Better scalability: When we separate our code complexity into smaller and simpler parts, we can make changes to each of those parts individually without affecting the overall functionality of the program.

**Example of abstraction**

Original implementation:

1 – Into the Program class, create a List variable (options)

2 – Add each menu option to the list

3 – Instantiate a Menu class and pass the options as arguments

4 – Display the menu options

<<Program.cs>>

Text, letter

Description automatically generated

Abstraction applied:

* Options removed from the Program class.
* Added a Dictionary containing the options into the Menu class.
* Use a method to display the options.
* Now we can manage our options from the dictionary and our Main program will not be affected.

<<Program.cs>>

Graphical user interface, text, application, email

Description automatically generated

<<Menu.cs>>

Text

Description automatically generated