Nested Decorators:

In TypeScript, when multiple decorators are applied to a class, they are executed in reverse order. This means that the last decorator specified in the code will be executed first, and the first decorator specified will be executed last.

The reason for this behavior is that the output of each decorator is passed as an input to the next decorator in the chain. So, if the decorators were executed in the order they are specified in the code, the input to each decorator would be the output of the previous decorator, which could cause unintended consequences or errors.

By executing the decorators in reverse order, the input to each decorator is the original class constructor, which allows each decorator to modify the class independently of the other decorators.

Regarding the question of whether decorators act as callbacks, they do share some similarities with callbacks in that they are functions that are called at specific points in the code. However, decorators are a language feature that is built into TypeScript and are executed by the language itself, whereas callbacks are typically functions that are passed as arguments to other functions and are executed by those functions at specific points in their execution. So while there are similarities, the two concepts are distinct.