Semantics vs Syntax

> Tags: Programming Languages, OOP, Visual Programming, Compiler, Interpreter

Programming Languages

set of keywords, commands, symbols and a system of rules for constructing statements. (allows humans to communicate instructions to a computer)

Semantics - refers to the meaning of the actions that occur in an algorithm
Syntax - is a set of rules associated with a programming language

Programming Language Generations and Paradigm

Ada Lovelace first published computer program for a mechanical computer program by **Charles Babbage**

"Programming languages image.png" could not be found.

High Level Language

- easy for programmer to understand
- contains English words

Low Level Language

- the computer's own language
- binary numbers, codes in 1's and 0's

"Programming Languages Image 2.png" could not be found.

Visual, Object-Oriented and Artificial Intelligence Languages

Structured Programming

uses the structured control flow structures of selection and repetition, block structure and subordinates.

- aimed at improving the clarity, quality and development time of a computer-program

Object-oriented programming (OOP)

Languages that are based on objects

Visual Programming

Uses a graphical or "visual" interface combined with text-based commands

Fifth Generation Language (5GLs)

- also called natural languages
- More English-like syntax than 4LGs

Object-Oriented Programming

in OOP, classes inherit properties and behaviors of the previous classes

- **class** - car
- **object** - Audi

class - is a definition of structure that contain properties and functionality

object - is a copy of **Class** with the certain values

Compiler vs Interpreter

Compiler - a special software program that converts the programmer's source code into the machine-language instructions. First, the compiler translates the program into a machine language; Second, the CPU executes that program

Interpreter - a language translator that carries out the operations called for by the source code. An **interpreter** does not produce a complete machine-language program. After the statement executes the machine language statement is discarded, the process continues for the next statement

Compiler - gets all text, goes through all of it, turns it into machine code

interpreter - translates the code portion by portion(line-by-line). If there is a problem in the middle of the code, it says:
*"The code produced by **interpreter** is not very much optimized, when look at the whole program, can optimize it a lot"*