EiABC
Computer Programming

Part 1

Contents

List of tables

List of figures

Fundamentals of programming / introductory information.

Even if this brief lecture note is prepared for design students, the lecture note is intended to provide general facts about programming to students before generally embarking on using programing for design endeavors. Nevertheless, students will be provided with design integrated lecture notes on further lecture notes.

1. What is programming?

To "program" means to organize the work of the computer through sequences of instructions. These commands (instructions) are given in written form and are implicitly followed by the computer (respectively by the operating system, the CPU and the peripheral devices). **To "program" (programing) means writing sequences of instructions in order to organize the work of the computer to perform something. These sequences of instructions are called "computer programs" or "scripts". A sequence of steps to achieve, complete some work or obtain some result is called an algorithm**. This is how programming is related to algorithms. Programming involves describing what you want the computer to do by a sequence of steps, by algorithms. **Programmers** are the people who create these instructions, which control computers. These instructions are called **programs** (Sofia, Svetlin Nakov & Co., 2013).

2. Types of programming languages:

There is no perfect programming language, they all offer something a little different, and there are hundreds of programming languages with new ones being created every day (excelwithbusiness.com).
Numerous programs exist, and they are created using different kinds of **programming languages**. Each language is oriented towards **controlling the computer on a different level**. There are languages oriented towards **the machine level** (the lowest) – Assembler for example. Others are most useful at the **system level** (interacting with the operating system), like C. There are also high level languages used to **create application programs**. Such languages include **C#, Java, C++, PHP, Visual Basic, Python, Ruby, Perl, JavaScript** and others (Sofia, Svetlin Nakov & Co., 2013).
There are of course so many other programing languages: ALGOL; ASPECTJ; APPLESCRIPT; ASSEMBLY LANGUAGE; BASH (UNIX SHELL); BASIC; C; C++; C#; CAML (OCAML); CLOJURE (CLOJURESCRIPT); COBOL; COFFEESCRIPT; DART; DBASE (FOXPRO); DELPHI (OBJECT PASCAL); EIFFEL; ERLANG; ELIXIR; F#; FORTRAN; GO; GROOVY (RUBY); HASKELL; IBM RPG; JAVA; JAVASCRIPT (ECMASCRIPT); LISP; LOGO; LUA; MACHINE

CODE; MATHEMATICA (WOLFRAM LANGUAGE); MATLAB; ML; NODE.JS; OBJECTIVE-C; PASCAL; PERL; PHP; POWERSHELL; PYTHON; R; RPG; RUBY; RUST; SCALA; SCHEME; SCRATCH; SMALLTALK; SWIFT; TCL and TYPESCRIPT (EXCELWITHBUSINESS.COM)

2.1. Comparison between programing languages

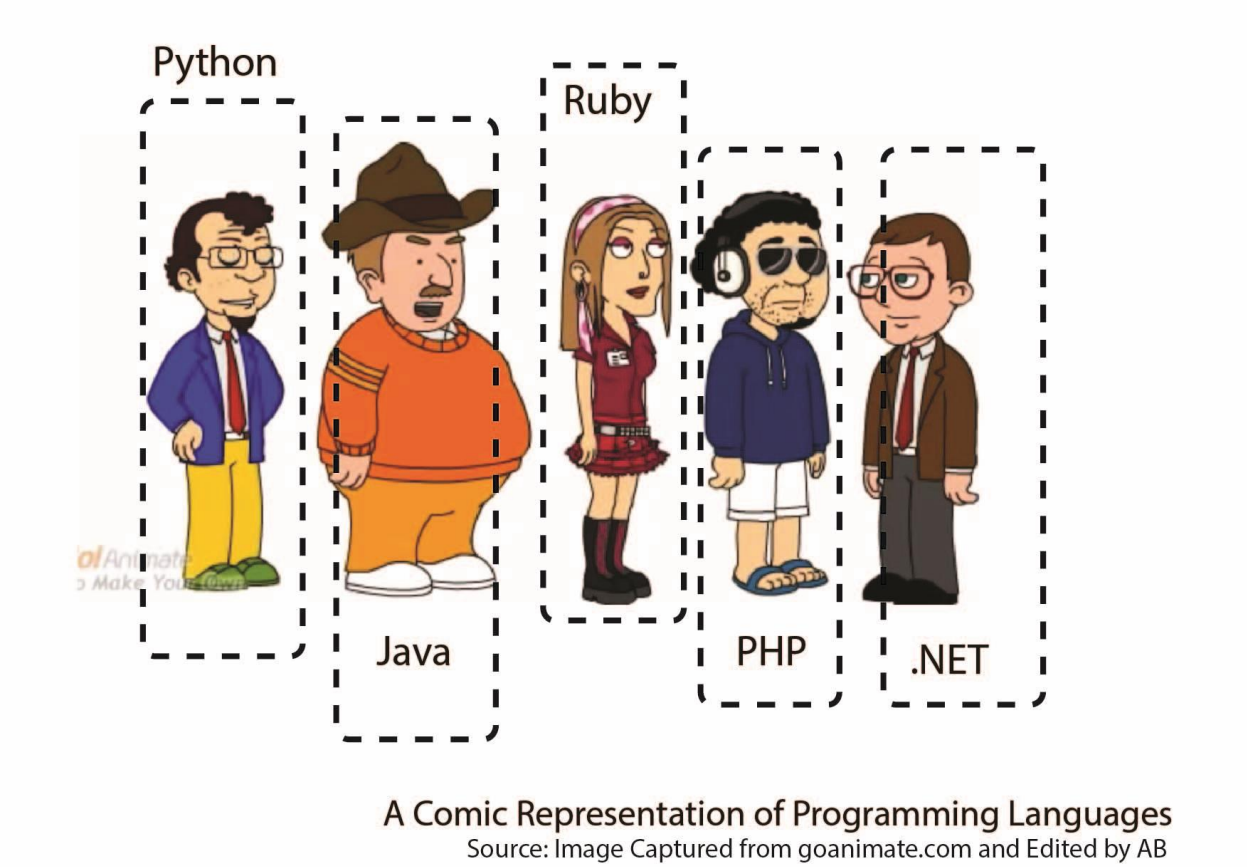

Figure 1: Programming languages

Table 1: Comparison between programing languages

Source: http://www.jvoegele.com/software/langcomp.html

	Eiffel	Smalltalk	Ruby	Java	C#	C++	Python	Perl	Visual Basic
Object-Orientation	Pure	Pure	Pure	Hybrid	Hybrid	Hybrid / Multi-Paradigm	Hybrid	Add-On / Hybrid	Partial Support
Static / Dynamic Typing	Static	Dynamic	Dynamic	Static	Static	Static	Dynamic	Dynamic	Static
Generic Classes	Yes	N/A	N/A	No	No	Yes	N/A	N/A	No

	Eiffel	Smalltalk	Ruby	Java	C#	C++	Python	Perl	Visual Basic
Inheritance	Multiple	Single	Single class, multiple "mixins"	Single class, multiple interfaces	Single class, multiple interfaces	Multiple	Multiple	Multiple	None
Feature Renaming	Yes	No	Yes	No	No	No	No	No	No
Method Overloading	No	No	No	Yes	Yes	Yes	No	No	No
Operator Overloading	Yes	Yes?	Yes	No	Yes	Yes	Yes	Yes	No
Higher Order Functions	Agents (with version 5)	Blocks	Blocks	No	No	No	Lambda Expressions	Yes (???)	No
Lexical Closures	Yes (inline agents)	Yes (blocks)	Yes (blocks)	No	No	No	Yes (since 2.1)	Yes	No
Garbage Collection	Mark and Sweep or Generational	Mark and Sweep or Generational	Mark and Sweep	Mark and Sweep or Generational	Mark and Sweep or Generational	None	Reference Counting	Reference Counting	Reference Counting
Uniform Access	Yes	N/A	Yes	No	No	No	No	No	Yes
Class Variables / Methods	No	Yes	Yes	Yes	Yes	Yes	No	No	No
Reflection	Yes (as of version 5)	Yes	Yes	Yes	Yes	No	Yes	Yes?	No
Access Control	Selective Export	Protected Data, Public Methods	public, protected, private	public, protected, "package", private	public, protected, private, internal, protected internal	public, protected, private, "friends"	Name Mangling	None	public, private
Design by Contract	Yes	No	Add-on	No	No	No	No	No	No
Multithreading	Implementation-Dependent	Implementation-Dependent	Yes	Yes	Yes	Libraries	Yes	No	No
Regular Expressions	No	No	Built-in	Standard Library	Standard Library	No	Standard Library	Built-in	No
Pointer Arithmetic	No	No	No	No	Yes	Yes	No	No	No
Language Integration	C, C++, Java	C	C, C++, Java	C, some C++	All .NET Languages	C, Assembler	C, C++, Java	C, C++	C (via DCOM)
Built-In Security	No	No?	Yes	Yes	Yes	No	No?	Yes (perlsec)	No
Capers Jones Language Level*	15	15	N/A	6	N/A	6	N/A	15	11

All free and proprietary software you know are written with different combinations of the programming languages mentioned in the above table and others that are not. Please refer to excelwithbusiness.com for further look at different programing languages.

Further comparison between programing languages:

These are the five different ways to differentiate between different programming languages:
Source: Garry Explains

a. If a program is compiled or interpreted

Interpreted: python

This is when a program runs through lines of codes you have just written and interprets it as it goes. Here your code does not need to be compiled, which means you wrote the code on the program you are running it in.

Compiled: c

In the first place the code you have written on a text editor or anywhere needs to be compiled in compiler. You compile it to produce a binary or executable program for a platform.

Intermediate language:

There is an intermediate kind of language, java, c#

b. A program is Strong and weak typing

Strong typing languages need the programmer to write his/her information thoroughly while weak typing languages work with less information from the coder.

c. Garbage collection:

Here the difference between the programs is that, some programs learn the staff you are using such as tables, strings, etc and workout if you are using them or not any more after which the program decides to free up the memory. While in other programs you will have to tell the programs which of the staffs you need to be cleared. For example C do not have garbage collection you have to manually allocate and free up any memory. But in programs such as java, when you are not using elements anymore, they get removed automatically.

d. Is it object oriented programming?

This is like saying can you create classes, can those classes be inherited, etc.?
C++, c#, java are object oriented while C is note.

e. If a program supports concurrency

If a coder can order a program to run two things at a time, it means the program supports concurrency. C does not have this function while GOLANG has concurrency concept. But for instance in C you can call operating level functions that can support concurrency even if the concept of concurrency is not built in to it as a specification.

3. Six Different Programing styles:

All programing styles are not mutually exclusive.

Imperative programming

: is just lists of instructions without blocks and loops.

Structural Programming

: based on imperative programing uses: **if blocks, loops**

Procedural Programming

: based on imperative programing and structural programming uses: **if blocks, loops, functions**. several programs such as C basic stop here.

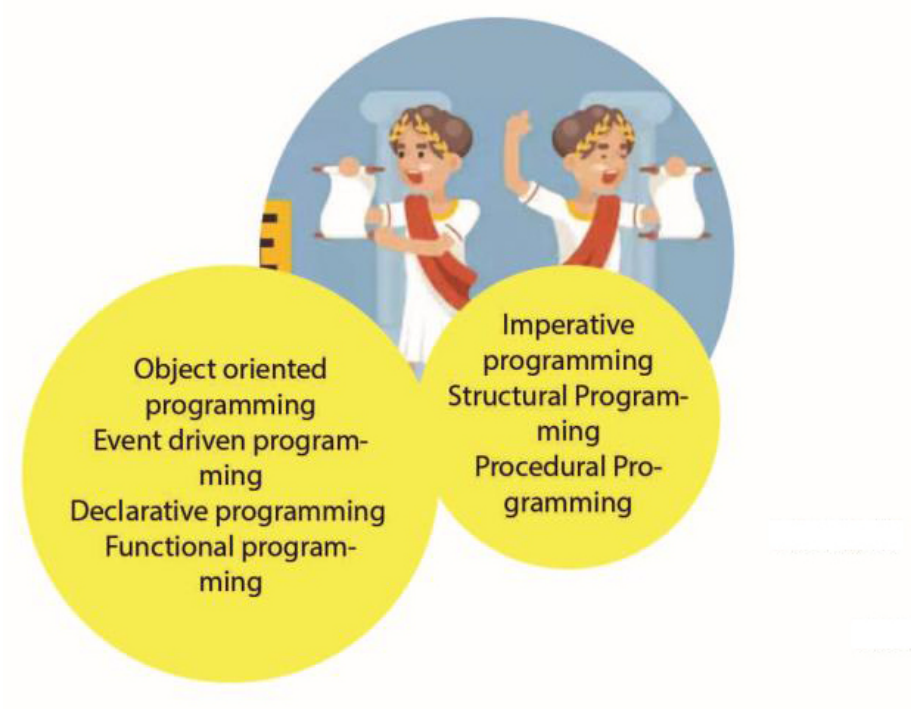

Figure 2: Comic image showing programing styles in two categories.
Source: Image from tylermcginnis.com edited by AB

Object oriented programming

: Based on imperative programing and structural programming uses: **if blocks, loops, functions, objects**, several programs such as java and C++ use this style.

Event driven programming

: Example GUIs(Java, Java Script . . .)

Declarative programming

: On the above styles we tell the program how to do a task whereas in this style, we tell the program what we want. Examples of this program styles can be: Markup language: HTML, Database language: SQL, Logic language: Prolog.

Functional programming

We write with this style based on declarative style. It avoids side effects that may be are seen on the above styles. Side effects are avoided because nothing happens outside of the scope of the functions used. Since it is declarative, we do not use loops. To replace loops we use recursions.

Box 1: a further lists of programming styles

Source: medium.com (Bradley Nice)

Interpreted Programming Languages; Functional Programming Languages; Compiled Programming Languages; Procedural Programming Languages; Scripting Programming Languages; Markup Programming Languages; Logic-Based Programming Languages; Concurrent Programming Languages; Object-Oriented Programming Languages

To revise about comparison between programing languages briefly Refer back to figure 1 figure 2 and table 1

End of lecture note!