15. - Programovací jazyky – vlastnosti, srovnání, popis způsobů tvorby i běhu programů

Co je programovací jazyk?

Programovací jazyky jsou jazyky pro tvorbu programů. Programem myslíme sadu instrukcí pro počítač, jak vyřešit nějaký problém – algorytmizujeme problémy.

Každý jazyk má určité diferencei, jak se liší od ostatních jazycích.Také je zde popularita, čitelnost, kompatibility. Hlavní vlastnost je míra abstrakce. Rozdělujeme je na vyšší a nižší.

Nižší

Jsou mnohem primitivnější, nemají komplexní syntaxi, píšeme podobný způsob jako píšeme rovnou příkazy pro procesor.

Procesor dělá určitou sadu instrukcí. Takže kód, který napíšeme, se nejdříve zpracuje do hexadecimálního kódu – strojový kód a poté do binárního. Tyto binární informace jsou poslány do procesoru.

Příklad: Jazyk Symbolických Adres - Assembler

Vyšší

Více zkušeností programátora, potřebují znalostu CPU, hardware.

Díky tomu, že nepracujeme s primitivním kódem jako v nižších, můžeme dělat složitější informace. Počítač vlastně neví, co je for each, procesor ho nezvládne. Kompilátor tedy zkompiluje kód do instrukcí.

Příklad: Java, Python, C#

Dělení dle kompilace

Kompilované jazyky

Příkladem kompilovaných jazyků je například C, C++. Výsledkem kompilací jsou hexa soubory – strojový kód. Jsou poměrně dost rychlé. Pokud kód obsahuje chybu, tak kompilátor hodí chybu. Díky tomu můžeme zjistit, kde je chyba před tím, než začneme spouštět kód. Nevýhoda je ale to, že zkompilovaný program je závislý na platformě a na typu procesoru nebo operačního systému. Kompilace občas může zabrat spoustu času v případě komplexního kódu. Jakmile máme zkompilovaný program, pokdu chceme udělat nějakou úpravu, tak musíme celou kompilaci duělat znovu.

Interpretované jazyky

Kompiluje jenom určité části a kompiluje během toho, jak jsou potřeba. Jedná se o tlumočníka. Člověk mluví anglicky, on to překládá do angličtiny.

Příklad je PHP.

Výhoda interpretovaných je to, že oproti kompilovaným je přenositelný mezi platformami. Jde o to, že interpreter je tlumočník. Proto je potřeba stáhnout v případě PHP exe soubor a tlumočníky, které dostanou na vstup

Není potřeba třeba dávat i datové typy, protože on je interpretuje postupně a můžou se měnit.

Jazyky s virtuálním strojem

Třeba Java – místo toho, aby on vytvářel rovnou hexadecimální – strojový kód, tak vytváří mezi binární soubor. V případě Javy to je bytecode. Řeší to problémy s hledáním chyb. Zde je zpracování jednodušší a rychlejší, interpretor to rychleji zpracuje.

Paradigmata

Paradigma zančí základní programovací styl – Alfa a omega toho, jak to má vypadat.

Dělá se na procedurální / imperatuvní a neprocedurální / deklarativní

Procedurální mají dva poddruhy – OOP a strukturované. Strukturovaný řeší problém pomocí funkce.

Neprocedurální jsou k tomu, abychom neřešili přesný postup, ale čeho má program dosáhnout, jakého cíle – funkcionální a logický.

Funkcionální je program vytvořený z matematických funkcích - LIPS
Logické, ten řeší pomocí matematiky a logických výroků to, jak to vypadat.

Dělení podle účelu

Není žádný jazyk, který by dokázal řešit všechny problémy a nebyl by naprosto unvierzální.

Díky tomu se třeba Python zaměřuje na umělou inteligenci, analytiku a vědeckou činnost, Java zase se využívá na bankovní systémy, assembler je vhodný pro programování k hardware, C vhodný pro práci se stroji jako třeba Raspberry PI atd.

Statické a dynamické typování

U statických je pevně definován datový typ při zakládání. Na začátku je jasné, jaký datový typ může proměnná mít a nemůžeme ho měnit. Při špatném datovém typu spadne program.

Příklad: C# a Java

Dynamické typování – Proměnná se může měnit. Datový typ je tedy určen hodnotou jako takovou. Hodnoty vznikají během programu a nevýhoda je, že se nedá během kompilace zjistit, kde je chyba.

Syntaxe

Jazyky mají jinou syntaxi, Python třeba řeší odsazení místo složených závor při dělání kódu v bloku jako v ostatních jazycích. Poté třeba dědičnost, Python se píše do závorek vedle třídy, v C a C# se dávají vedle třídy po dvojtečce.