# Programovací jazyky – vlastnosti, srovnání, popis způsobů tvorby i běhu program

Programovací jazyk je prostředek pro zápis algoritmů, jež mohou být provedeny na počítači. Programovací jazyky využíváme tvorbu programů … posloupnost instrukcí, které popisují realizaci dané úlohy (algoritmu) počítačem. Základní program počítače je jádro, který řídí jeho chod a uživatel pak pracuje s aplikačním softwarem.

Každý programovací jazyk je něčím specifický a liší se od jiných například: syntaxí, složitostí, usecasem, popularita, čitelnost atd. Hlavní vlastností je míra abstrakce. Programovací jazyky dělíme na vyšší a nižší.

**hexadecimální kód je strojový kód**

## Vyšší programovací jazyk

Je označení pro programovací jazyk s vyšší mírou abstrakce. Tím je míněno přiblížení zdrojového kódu programu v danem programovacím jazyce k tomu, jak problémy zpracovává svým myšlením člověk. Počítač vlastně neví, co je for each, procesor ho nezná. Kompilátor tedy zkompiluje kód do instrukcí.

Příklady: Java, Python, C#

* Vyšší programovací jazyky mají obvykle vyšší úroveň sémantiky, což umožňuje programátorům vytvářet složitější algoritmy s méně kódovými řádky.
* vyšší programovací jazyky se obvykle kompilují do mezikódu, který je následně interpretován.
* Vyšší programovací jazyky jsou obvykle pomalejší a mají větší nároky na paměť kvůli vyšší úrovni abstrakce.
* Vyšší programovací jazyky jsou obvykle více přenositelné než nižší programovací jazyky, protože jsou méně závislé na konkrétní architektuře a hardwaru.

## Nižší programovací jazyky

jsou obvykle blíže strojovému kódu a hardwaru, což znamená, že jsou úzce spojeny s konkrétními architekturami a hardwarovými platformami. Tyto jazyky jsou často používány pro účely programování přístupu k hardwaru, jako jsou ovladače, operační systémy a firmware.

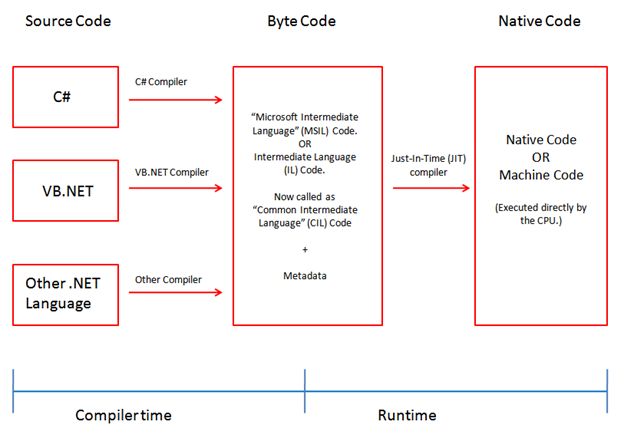
Příklad: Jazyk Symbolických Adres – Assembler, C

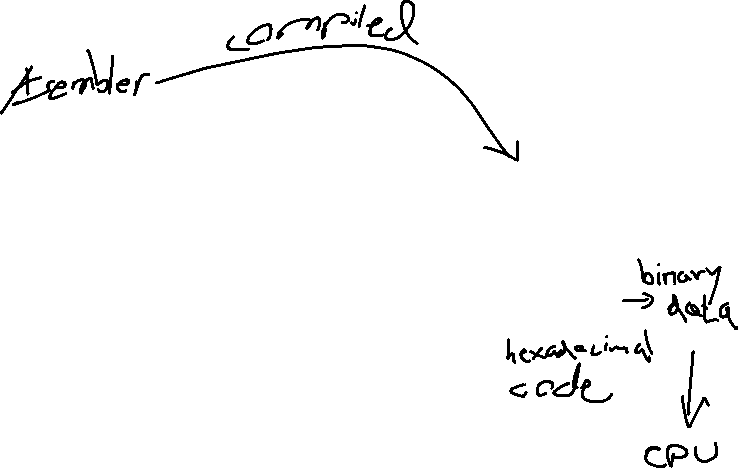
* Nižší programovací jazyky mají tendenci být méně abstraktní a vyžadovat více manuálního řízení.
* Nižší programovací jazyky se obvykle kompilují přímo do strojového kódu
* Nižší programovací jazyky jsou obecně rychlejší a mají menší nároky na paměť než vyšší programovací jazyky, protože jsou blíže strojovému kódu.
* Nižší programovací jazyky jsou obvykle závislé na konkrétní architektuře a hardwar

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.

### Asmebler

Kód psaný v asembleru je kompilován pomocí překladače assembler na strojový kód. Assemblery jsou programovací jazyky, které umožňují programátorům psát instrukce, které jsou přímo srozumitelné procesoru. Tyto instrukce jsou obvykle zapsány ve formě mnemonik, což jsou zkrácené názvy pro strojové instrukce, např. MOV pro přesunutí dat. Assembler kód je tedy v podstatě jen lidsky čitelnou verzí strojového kódu. Stále je ale potřeba ho převést do strojového kódu.





Kompilované jazyky

Kompilované jazyky jsou programovací jazyky, ve kterých se zdrojový kód překládá na strojový kód jednou kompilací. Výsledný strojový kód pak může být spuštěn na cílovém počítači. Některé příklady kompilovaných jazyků jsou C či C++.

* Kompilace může zabrat mnoho času, ale po zkompilování je program celkem rychlý
* Pokud kód obsahuje chybu (odhalitelnou kompilátorem), tak kompilátor hodí chybu. u. 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.

Interpretované jazyky

Interpretované jazyky jsou programovací jazyky, ve kterých se zdrojový kód interpretuje postupně za běhu programu. Interpreter čte zdrojový kód a provádí ho bez překladu na strojový kód. Některé příklady interpretovaných jazyků jsou Python, Ruby a JavaScript.

* Výhoda interpretovaných je to, že oproti kompilovaným je přenositelný mezi platformami.
* Nejprve je ale potřeba si stáhnout „tlumočníka“ který kód bude překládat
* Není potřeba třeba dávat i datové typy, protože interpret při běhu programu automaticky rozpoznává typ každé proměnné podle její hodnoty a provádí příslušné operace v souladu s tímto typem. Můžou se měnit protože interpret sleduje typ každé proměnné na základě její aktuální hodnoty.

Hybridní jazyky či Jazyky s virtuálním strojem

Jsou programovací jazyky, které jsou kompilované i interpretované. Tyto jazyky mohou mít překladový krok při kompilaci, ale také mohou být interpretovány při běhu programu. Příkladem hybridního jazyka je například C#. Například v jave je vytvořen **mezikód** v Jave pojmenován bytecode. Tím pádem proběhne proces kompilace, ale zároveň se následně bude vytvořený kód interpretovat tím pádem nejsem závislí na daném hardwaru. Navíc mezikód bude interpretován rychleji než zdrojový kód.

## Paradigmata

Jsou způsoby jak zařadit programovací jazyky na základě jejich features. Programovací jazyk mohou být zařazeny do více paradigmat.

Dělíme je na imperativní a deklarativní programování.

Imperativní programování (též procedurální programování) je založeno na příkazech, které programu říkají, jak provádět určité kroky a jakým způsobem změnit stav proměnných.

* Strukturované … Strukturovaný řeší problém pomocí funkce.
* OOP … je program vytvořen z objektů, objekt je instancí třídy, což je zapouzdření dat (tzv. Pole) a procedur (nazývaných metody), které s nimi manipulují.

Deklarativní programování (též neprocedurální programování) je založeno na myšlence programování aplikací pomocí definic co se má udělat, a ne jak se to má udělat.

* Funkcionální programování chápe výpočet jako vyhodnocení matematických funkcí. Funkcionální programování má své kořeny v lambda-kalkulu, formálním systému vyvinutém v 30. letech k vyšetřování definicí funkcí, jejich aplikace a rekurze
* Logické programování je v širším významu použití matematické logiky jako prostředku pro programování.
* Reaktivní programování orientované kolem datových toků a šíření změn. To znamená, že by mělo být možné vyjádřit statické nebo dynamické datové toky v programovacích jazycích jednoduše a že základní provedení modelu bude automaticky kopírovat změny prostřednictvím datového toku.

## 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.