**Úvod do C++**

[**http://kmlinux.fjfi.cvut.cz/~fabiadav/cecko/poznamky-k-jazyku-c\_plus\_plus**](http://kmlinux.fjfi.cvut.cz/~fabiadav/cecko/poznamky-k-jazyku-c_plus_plus)

V tomto článku se seznámíme s historií a vývojem C++, tím, proč tento jazyk vznikl, a jak se liší od jazyka C.

**Úvod**

Jazyk C++ se vyvinul z jazyka C jako jeho nadstavba. Touha po možnosti objektového programování přivedla Bjarna Stroustrupa k sepsání první verze jazyka C++. Původní verzi tohoto nového jazyka nazval *C with Classes*. Bjarne postavil svou práci nad jazykem C, který byl dostatečně rychlý, rozšířený a přenositelný. Celý jazyk C++ byla tehdy jen syntakticá obálka nad jazykem C a kód se před vlastním překladem převáděl do zdrojového kódu jazyka C. Teprve později vznikl samostatný překladač a jazyk se osamostatnil.

**Historie jazyka**

Bjarne Stroustrup sepsal první verzi jazyka C++ v roce 1979, když pracoval v Bellových laboratořích. Původní název C with Classes byl změněn na současný C++ v roce 1983. V roce 1985 byla zveřejněna první komerční implementace tohoto jazyka. Od roku 1998 je jazyk standardem ISO stejně jako jazyk C. Posledním vydaným standardem je [C++11](http://kmlinux.fjfi.cvut.cz/~fabiadav/cecko/slovnik/C/c_plus_plus11), který byl uveden v září roku 2011. Přináší velkou řadu změn napříc celým jazykem a zároveň udržuje zpětnou kompatibilitu se staršími verzemi jazyka.

**Odlišnosti od jazyka C**

Jazyk C++ je téměř přímou nadmnožinou jazyka C. Nicméně existuje několik odchylek, které byly do jazyka zavedeny. Některé z nich uvedeme v následujícím seznamu.

1. V jazyce C šlo implicitně konvertovat [ukazatel](http://kmlinux.fjfi.cvut.cz/~fabiadav/poznamky-k-jazyku-c/ukazatele) bez doménového typu na ukazatel na typ. Nejčastěji se tato operace prováděla při použití funkce malloc(). V C++ již toto není možné a je nutné tuto konverzi provést explicitně.
2. [Struktury a výčty](http://kmlinux.fjfi.cvut.cz/~fabiadav/poznamky-k-jazyku-c/struktury-a-vycty) se v jazyce C musí uvádět s plným označením, tzn. s klíčovým slovem struct nebo enum. Tuto nepříjemnost lze obejít použitím typedef. Jazyk C++ nás této povinnosti zbavuje, tudíž lze deklarovat proměnnou struktury nebo výčtu bez příslušného klíčového slova.
3. V jazyce C není žádný typ, který reprezentuje logickou hodnotu. Každý číselný typ lze pro tento účel použít. Naproti tomu v jazyce C++ existuje speciální typ bool, který nabývá hodnot true a false.
4. Jazyk C++ má mnohem větší standardní knihovnu. To se projeví například, když překládáme program využívající některou z matematických funkcí. Překládáme-li překladačem [GCC](http://kmlinux.fjfi.cvut.cz/~fabiadav/poznamky-k-jazyku-c/preklad-pomoci-gcc), musíme uvést parametr -lm. Při použití překladače G++ toto dělat nemusíme.
5. Jazyk C++ zavádí nový způsob předávání parametrů. K předávání odkazem a hodnotou přibylo předávání referencí. Reference jsou v podstatě zástupná jména existujících proměnných a zmiňujeme se o nich níže.

**Reference**

Reference jsou novým způsobem předávání parametrů. Mají blízko k technice předávání parametrů odkazem, mají však některá omezení. Předně nelze řetězit reference, tedy vytvářet reference na reference. Dále nelze vytvořit ukazatel na referenci. Reference se vytvoří následovně.

**int** a = 1; *// vytvoření proměnné a*

**int** &ref = a; *// vytvoření reference na proměnnou a*

ref = 3; *// v a nyní bude hodnota 3*

Jakmile spojíme referenci s nějakou proměnnou, již to nemůžeme změnit. Do konce života reference bude tato spojená s danou proměnnou a veškeré změny v referenci budou provedeny v zastupované proměnné. Není potřeba dereferencovat referenci ani předávat adresu proměnné při přiřazování do reference. Jak vidíme, jde o velmi pohodlný způsob odkazování.