Náplň kurzu

- rozsah je přizpůsoben zkušenostem z minulých roků
- Učí se pouze „základy“. Je možné zrychlit výuku a učit i „nástavby“.

Jak hodnotíte své znalosti jazyka C?
Kolik z vás programovalo v C++?
Kolik v objektových jazycích?

- týdenní plán přednášek – orientační

- neobjektové vlastnosti – rozšíření vlastností oproti jazyku C
- objektové vlastnosti – objektové programování, dědění, polymorfizmus
- „nástroje“ – výjimky, šablony, streamy, STL,

Náplň kurzu

- opakování a rozšíření jazyka C
- programátorské dovednosti
- jazyk C++

Proč C++

- vysoký výkon – překlad do spustitelného kódu (assembler), kvalitní optimalizace, jazyk se snaží neimplementovat pomalé mechanizmy, blízké spojení jazyka a výsledného kódu (např. jasně definovaná životnost/zánik proměnných (odložený zánik pomocí garbage collectoru (destruktor) může blokovat zdroje (soubory, linky (seriová, …), handly k zařízení (tiskárna, monitor …)))
- moderní jazyk
- velké rozšíření
- velké množství cílových platforem – PC, mobily, hradlová pole, programovatelné automaty …

Náplň kurzu

Opakování a rozšíření jazyka C, neobjektové vlastnosti

- především na cvičeních než bude možné začít s objektovým C++
- překlad programu, hlavičkové a zdrojové soubory
- rozdíly „čistého C“ oproti možnostem C++ a objektovému programování
- nové datové typy,
- ukazatele x reference

- vstup a výstup - streamy
- knihovny jazyka C, C++

Náplň kurzu

Programátorské dovednosti

- obecná pravidla programování (návrh, tvorba, překlad a testování programu)
- základní programátorské dovednosti a návyky – programátorská kultura, trasování, … použití objektů
- nástroj pro správu (verzí) projektů svn pro spolupráci více autorů na jednom projektu
- nástroj doxygen (+graphviz) pro komentování programů a pro tvorbu dokumentace

Náplň kurzu

Jazyk C++ a objektové programování

- základní teorie, rozdíly oproti C stylu programování
- objektové vlastnosti,
- dědění,
- polymorfizmus
- šablony
- STL
- …

Výuka programovacích jazyků na tomto oboru

… a počátek devadesátých let

- Analogové programování (modelování dynamických soustav), logické obvody a programovatelné automaty, Assembler (práce s programovatelným HW), Pascal (vědecké výpočty), Prolog, Lisp (umělá inteligence)
- podle aktuální situace a oboru činnosti (dostupné překladače, drivery, programová prostředí a jejich možnosti) se některé způsoby programování a jazyky opouštějí a jiné se dostávají do popředí (programování hradlových polí, mikroprocesorů, inteligentních periferií, zpracování dat, komunikace na sítích …)
- základem je stále logické myšlení, znalost základních nástrojů programování a jejich využití

Polovina devadesátých let

- výuka jazyka C/C++ navazující na Pascal. V **jednom semestru** výuka jazyka C, obsluha základního HW (čítače, časovače, přerušení, DMA …), C++.

Dvacáté první století

- po ukončení výuky Pascalu se C/C++ učí ve dvou semestrech.
- počátek grafického programování – "… pište aplikaci bez znalosti jazyka … v našem prostředí napíšete aplikaci bez programování …" – stačí pro "běžného uživatele", náš absolvent by však měl uvažovat i v souvislostech (jak a proč je implementováno, …) a znát mechanizmy hlouběji

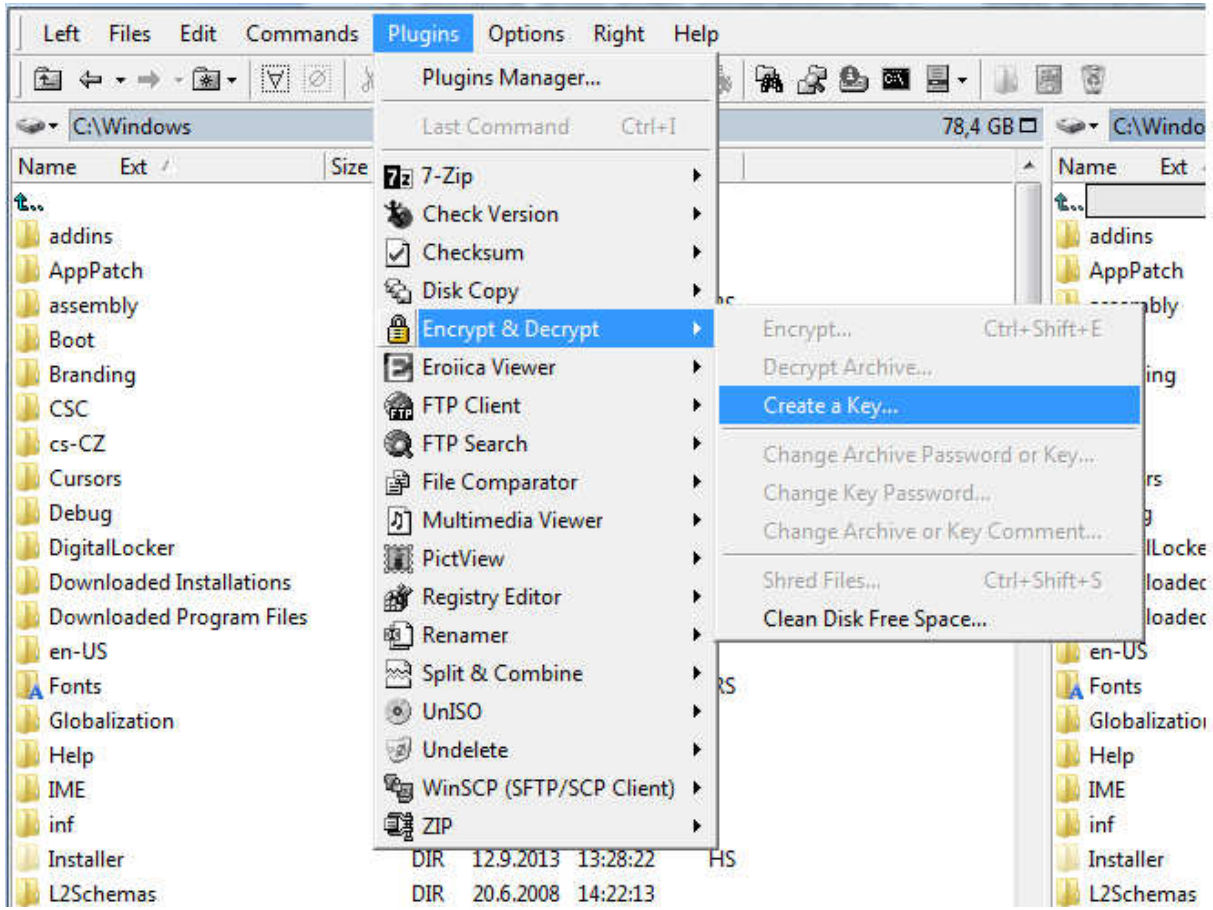
Příklady využití jazyka C++

- vhodné využití: složitější projekty, znovupoužití kódu (dědění, polymorfizmus, šablony)

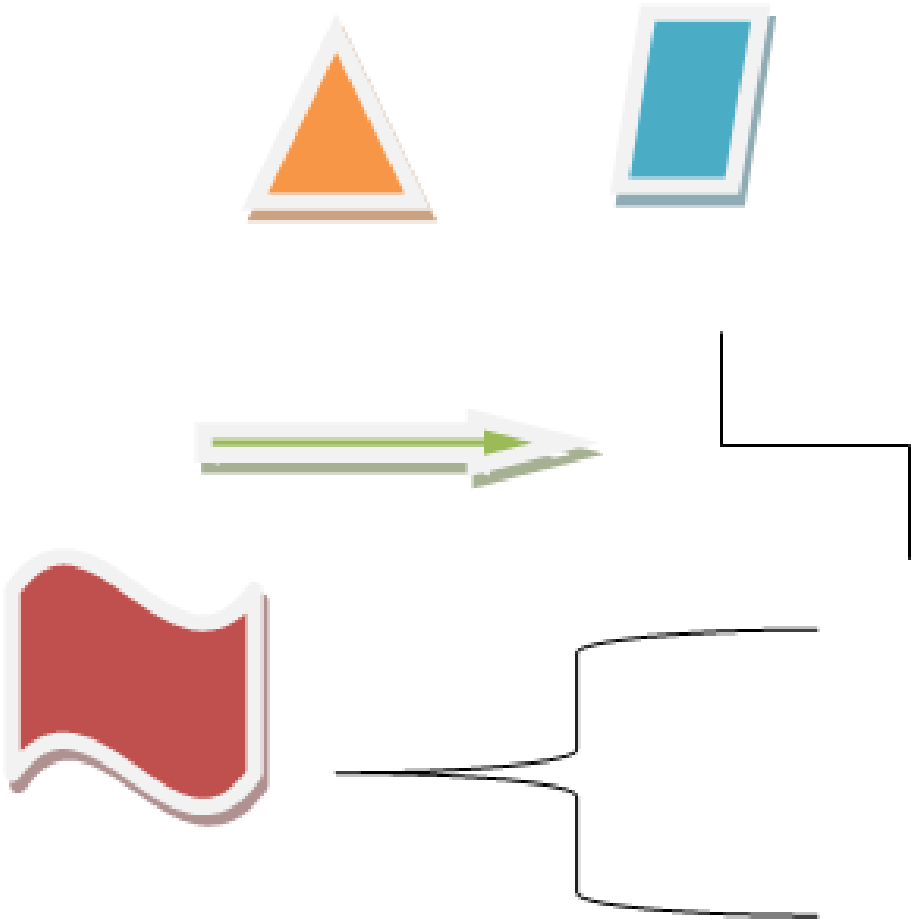
- projekty realizované v C++:
  Adobe Systems (Photoshop, Illustrator, Adobe Premier …)
  Google (Google Chromium, Google file system)
  Mozilla (Firefox, Thunderbird)
  MySQL (využito: Yahoo!, Alcatel-Lucent, Google, Nokia, YouTube, Wikipedia …)
  Autodesk Maya 3D (3D modeling, games, animation, modeling …)
  Winamp Media Player
  Apple OS X (některé části a některé aplikace)
  Microsoft (většina na bázi C++)
  Symbian OS
  Qt framework
  OpenCV

Konkrétní realizace – projekty vhodné pro objektové programování

- *menu* – je struktura obsahující položky (menu, podmenu, akce …). Struktury pro menu, položky, texty, barvy …



- *grafické objekty* – mají společný interface (rozhraní/ přístup). I když jsou různého typu, je možné s nimi pracovat jednotně – program si "zjistí" jakého jsou typu a provede správnou akci pro daný typ (vykreslení, rotace, posun, inverze barev, zjištění který objekt je nejblíže dané pozici (kliknutí myši), …)



- vytváření nových datových typů s operacemi jako typy standardní (ale vlastním chováním)
- například komplexní čísla, zlomky, matice
- znovupoužití kódu pro různé typy …

ukázka kódu pro vlastní typ MATRIX (matice). "Umí" operace jako základní typy, a také je možné ho "naučit" funkce. Pro výpočty je možné zvolit přesnost výpočtu – zde double

```
MATRIX <double> y,x(5,5),a(5,1),A,x1(10,5), ykontrola;
int i,j;

//  řešení rovnice y = x * A pro neznámé A
A = SolveLinEq(x,y);
// kontrola správnosti
ykontrola = x * A;
// vypočtení kvadrátu chyby řešení
chyba = (ykontrola - y);
chyba *= chyba;

// obecné výpočty se standardními operátory a vlastními funkcemi pro
// nový typ (MATRIX)
x1 = x * A * Transp(y) * Inv(x);
```