Y36PJC Programování v jazyce C/C++

# Jazyk C/C++ - úvod

Ladislav Vagner

# Stručná historie jazyka C

1972 D. M. Ritchie, Bell Laboratories, PDP11
1978 D. M. Ritchie, B. W. Kernighan:
The C Programming Language
1982 ANSI pracovní skupina X3J11
ANSI X3.159-1989 (C89)
převzatá ISO/IEC 9899:1990 (C90)
1999 nová norma ISO 9899:1999 (ISO C99)

- minimalistický jazyk, kompaktní zápis,
- velké množství operátorů,
- využívání ukazatelů, ukazatelová aritmetika,
- vhodný pro programování jádra OS, ovladačů, systémových knihoven.

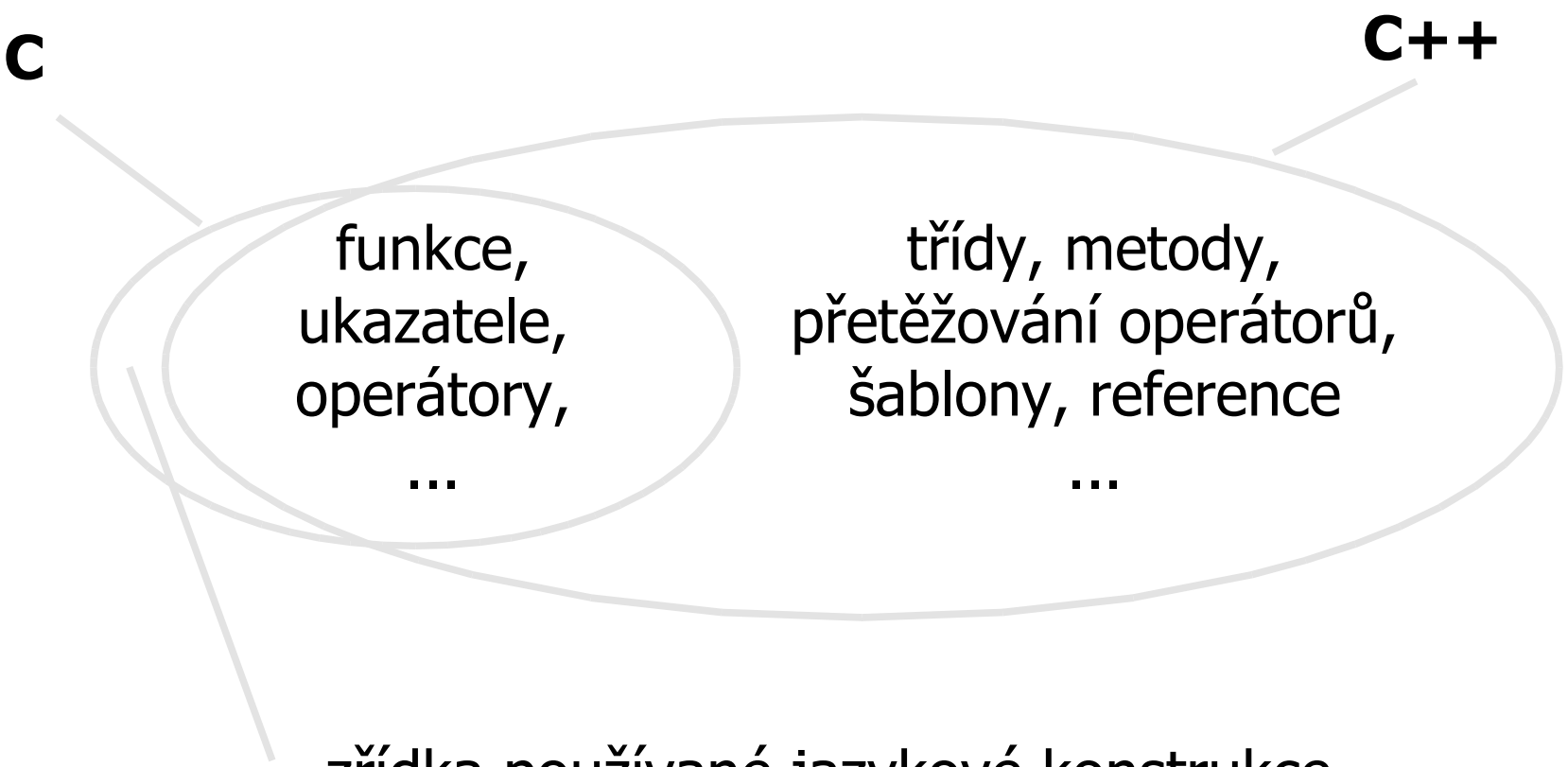
# Stručná historie jazyka C++

1983 B. Stroustrup, AT & T, Bell Laboratories
1986 B. Stroustrup: The C++ Programming Language
1991 B. Stroustrup: The C++ Programming Language (2-nd edition), vyšlo v češtině (1997 BEN)
1998 ISO norma ISO/IEC 14882

Rozšíření jazyka C o:

- třídy a OOP,
- přetěžování funkcí, metod a operátorů,
- šablony (templates),
- výjimky,
- ...

# Vztah C a C++


C
C++
funkce,
ukazatele,
operátory,
...
třídy, metody,
přetěžování operátorů,
šablony, reference
...
zřídka používané jazykové konstrukce
(např. K&R notace zápisu funkcí)

# Od Javy k C/C++

Podobná syntaxe, ALE:

- C/C++ jsou kompilované do nativního kódu procesoru.
- Programy v C/C++ jsou platformně závislé. Nejsou 100% přenositelné ani na úrovni zdrojových kódů, dokonce i když se programátoři snaží dodržet normu:
  - GUI, grafika, zvuk,
  - vlákna, semafory, mutexy,
  - velikosti a reprezentace datových typů.
- Standardní knihovny C/C++ nabízejí ve srovnání s Javou pouze minimum funkcionality.
- C/C++ je navržené pro dosažení výkonu, Java pro přenositelnost, bezpečnost a stabilitu.

# Od Javy k C/C++

- C/C++ umožní alokovat data na haldě i na zásobníku. Více svobody programátora = větší zodpovědnost.
- C/C++ nemá garbage collector. Programátor se stará o uvolnění nepotřebných bloků paměti.
- C++ předává parametry hodnotou i odkazem.
- V C/C++ můžeme přistoupit k libovolné proměnné (objektu, poli, primitivnímu typu) jako k bajtům (lze zjistit adresu i velikost).
- C/C++ nehlídá meze polí.
- V C/C++ lze snadno přepsat data či kód svého programu a ten tím způsobit jeho pád.
- V C/C++ lze pracovat s adresami funkcí a metod.
- C++ nemá mechanismus reflexe.

# Od Javy k C/C++

- V C++ máme možnost pracovat přímo s instancemi tříd (ne jen s referencemi).
- C++ má jak statickou tak dynamickou vazbu.
- V C++ neexistuje předek společný pro všechny třídy (ekvivalent **`java.lang.Object`**).
- C++ nemá interface.
- C++ má vícenásobnou dědičnost.
- C++ můžeme přetížit operátory – definovat jim význam podle našich představ.
- V C/C++ lze realizovat programy:
  - procedurálně,
  - objektově (doporučeno),
  - "s objekty" (nedoporučeno).

# Od Javy k C/C++

- C/C++ vyžaduje po programátorovi pořádek, disciplínu, zodpovědnost a znalost.
- Nabízí rychlost, minimální režii, malou velikost.
- Existuje šance, že špatně napsaný kód v Javě bude nějak fungovat.
- Špatně napsaný kód v C/C++ nebude fungovat vůbec nebo bude nahodile padat.

## C/C++ není kojná.

# Hello world v C

```
#include <stdio.h>

int main ( int argc, char * argv [] )
 {
   printf ( "Hello world !\n" );
   return ( 0 );
 }
```

# Hello world v C++ (starší notace)

```
#include <iostream.h>
#include <iomanip.h>

int main ( int argc, char * argv [] )
 {
   cout << "Hello world !" << endl;
   return ( 0 );
 }
```

# Hello world v C++ (nová notace)

```
#include <iostream>
#include <iomanip>
using namespace std;

int main ( int argc, char * argv [] )
 {
   cout << "Hello world !" << endl;
   return ( 0 );
 }
```

```
// Test, zda je rok prestupny
#include <iostream>
#include <iomanip>
using namespace std;
int main ( int argc, char * argv [] )
 {
   int  y;

   cout << "Zadej rok" << endl;
   cin >> y;

   if ( y % 4==0 && (y % 100!=0 || y % 400==0))
     cout << "Rok " << y << " je prestupny" << endl;
    else
     cout << "Rok " << y << " neni prestupny"<< endl;
   return ( 0 );
 }
```

```
// Test, zda je rok prestupny
#include <iostream>
#include <iomanip>
using namespace std;
int main ( int argc, char * argv [] )
 {
   int  y;

   cout << "Zadej rok" << endl;
   cin >> y;

   cout << "Rok " << y << ((y % 4==0 && (y % 100!=0
        || y % 400==0)) ? " je" : " neni") <<
        " prestupny" << endl;
   return ( 0 );
 }
```

```
// Vypocet poctu dni od zacatku roku
#include <iostream>
#include <iomanip>
using namespace std;
int main ( int argc, char * argv [] )
 {
   int  d, m, y;
   cout << "Zadej den, mesic a rok" << endl;
   cin >> d >> m >> y;
   switch ( m )
    {
      case 1: sum  = d;        break;
      case 2: sum  = 31 + d;   break;
      case 3: sum  = 59 + d;   break;
      …
      case 12: sum = 334 + d; break;
      default: cout <<"Spatny mesic"<<endl; return 1;
    }
```

```
      if ( m > 2 && ( y % 4 == 0 && y % 100 != 0 ||
        y % 400 == 0 )) sum ++;
    cout << d << "." << m << " byl " << sum <<
         "-ty den v roce " << y << endl;
    return ( 0 );
}
```

```
// Vypocet poctu dni od zacatku roku
#include <iostream>
#include <iomanip>
using namespace std;
int main ( int argc, char * argv [] )
 {
   int  d, m, y, sum;
   cout << "Zadej den, mesic a rok" << endl;
   cin >> d >> m >> y;

   sum = d + 31 * (m-1) - 10 * m / 23 - 2 * (m > 2)+
         ((m > 2) && (!(y%4)&& (y%100)|| !(y%400)));
   cout << d << "." << m << " byl " << sum <<
        "-ty den v roce " << y << endl;
   return ( 0 );
 }
```

```
// Celociselne mocneni
#include <iostream>
#include <iomanip>
#include <cmath>
using namespace std;
int main ( int argc, char * argv [] )
 {
   int    n, i;
   double a, res = 1;
   cout << "Zadej a, n:" << endl;
   cin >> a >> n;
   for ( i = 0; i < n; i ++ )
    res *= a;
   cout << a << "^" << n << " = " << res << endl;
   return ( 0 );
 }
```

```
// Vypocet faktorialu rekurzi
#include <iostream>
#include <iomanip>
using namespace std;
static int fact ( int n )
 {
   return ( n > 1 ? n * fact ( n - 1 ) : 1 );
 }
int main ( int argc, char * argv [] )
 {
   int n;

   cout << "Zadej n:" << endl;
   cin >> n;
   cout << n << "! = " << fact ( n ) << endl;
   return ( 0 );
 }
```

```
// Vypocet faktorialu iteraci
#include <iostream>
#include <iomanip>
using namespace std;
int main ( int argc, char * argv [] )
 {
   int i, n, fact = 1;

   cout << "Zadej n:" << endl;
   cin >> n;

   for ( i = 1; i <= n; i ++)
    fact *= i;

   cout << n << "! = " << fact << endl;
   return ( 0 );
 }
```

```
// Vypocet faktorialu Stirlingovym vzorcem (approx.)
#include <iostream>
#include <iomanip>
#include <cmath>
using namespace std;
int main ( int argc, char * argv [] )
 {
   int    n;
   double fact;

   cout << "Zadej n:" << endl;
   cin >> n;
   fact = pow ( (double)n, (double)n ) / exp ( n ) *
          sqrt ( 2 * M_PI * n );
   cout << n << "! = " << fact << endl;
   return ( 0 );
 }
```

```
// Prevod casu na hodiny/minuty/sekundy
#include <iostream>
#include <iomanip>
using namespace std;
void  TimetoHMS ( int Time, int *H, int *M, int *S )
 {
   *H = Time / 3600;
   *M = ( Time / 60 ) % 60;
   *S = Time % 60;
 }
int main ( int argc, char * argv [] )
 {
   int t, h, m, s;
   cout << "Zadej cas v sec" << endl;
   cin >> t;
   TimetoHMS ( t, &h, &m, &s );
   cout <<t<< "=" <<h<< ":" <<m<< ":" <<s<< endl;
   return ( 0 );
 }
```

```
// Prevod casu na hodiny/minuty/sekundy
#include <iostream>
#include <iomanip>
using namespace std;
void  TimetoHMS (int Time, int & H, int & M, int & S)
 {
   H = Time / 3600;
   M = ( Time / 60 ) % 60;
   S = Time % 60;
 }
int main ( int argc, char * argv [] )
 {
   int t, h, m, s;
   cout << "Zadej cas v sec" << endl;
   cin >> t;
   TimetoHMS ( t, h, m, s );
   cout <<t<< "=" <<h<< ":" <<m<< ":" <<s<< endl;
   return ( 0 );
 }
```

```
// Trida pro reprezentaci zlomku
#include <iostream>
#include <iomanip>
using namespace std;
class CRat
 {
   public:
    CRat ( int n = 0, int d = 1 )
     { num = n; den = d; simplify (); }
    friend CRat operator+ ( const CRat & a,
                            const CRat & b );
    friend ostream & operator << ( ostream & os,
                                   const CRat & x );
   private:
    int num, den;
    static int gcd ( int a, int b );
    void simplify ( void )
     { int cd = gcd ( abs (num), abs (den) );
       num /= cd; den /= cd; }
 };
```

```
CRat operator+ ( const CRat & a, const CRat & b )
 {
   CRat r ( a . num * b . den + a . den * b . num,
            a . den * b . den );
   r . simplify ();
   return r;
 }
ostream & operator << ( ostream &os, const CRat &x )
 {
   os << "(" << x . num << "/" << x . den << ")";
   return os;
 }
int CRat::gcd ( int a, int b )
 {
   if ( !a || !b ) return 1;
   while ( a != b )
    if ( a > b )  a -= b; else b -= a;
   return a;
 }
```

```
int main ( int argc, char * argv [] )
 {
   CRat a ( 6, 23 ), b ( 8, 15 );
   CRat c;

   c = a + b + 8;

   cout << a << " + " << b << " + 8 = " << c << endl;
   return ( 0 );
 }
```

```
// Trida pro natahovaci pole - vektor
#include <iostream>
#include <iomanip>
using namespace std;
class CVect
 {
   public:
    CVect ( void ) { m_Max = 0; m_Data = 0; }
    CVect ( const CVect & x );
    ~CVect ( void ) { delete [] m_Data; }
    void    Add ( int x );
    CVect & operator= ( const CVect & x );
    int   & operator [] ( int idx );
   private:
    int   * m_Data;
    int     m_Max;
    friend ostream & operator << ( ostream &os,
                                   const CVect &x );
 };
```

```
CVect::CVect ( const CVect & x )
 {
   m_Max  = x . m_Max;
   m_Data = new int [m_Max];
   for ( int i = 0; i < m_Max; i ++ )
    m_Data[i] = x . m_Data[i];
 }

void CVect::Add ( int x )
 {
   int * tmp;
   tmp = new int [m_Max+1];
   for ( int i = 0; i < m_Max; i ++ )
    tmp[i] = m_Data[i];
   delete [] m_Data;
   m_Data = tmp;
   m_Data[m_Max++] = x;
 }
```

```
CVect & CVect::operator= ( const CVect & x )
 {
   if ( &x != this )
    {
      delete [] m_Data;
      m_Max = x . m_Max;
      m_Data = new int [m_Max];
      for ( int i = 0; i < m_Max; i ++ )
       m_Data[i] = x . m_Data[i];
    }
   return ( *this );
 }

int & CVect::operator [] ( int idx )
 {
   if ( idx < 0 || idx >= m_Max )
    throw "Invalid index";
   return ( m_Data[idx] );
 }
```

```
ostream & operator << ( ostream &os, const CVect &x )
 {
   os << "[";
   for ( int i = 0; i < x . m_Max; i ++ )
    {
      if ( i ) os << ", ";
      os << x . m_Data[i];
    }
   os << "]";
   return os;
 }
int main ( int argc, char * argv [] )
 {
   CVect a;
   a . Add ( 10 );  a . Add ( 20 );
   CVect b = a;
   a . Add ( 30 );  a . Add ( a[1] );  a[0] = 5;
   cout << a << endl << b << endl;
   return ( 0 );
 }
```

# Dotazy...

Děkuji za pozornost.