

Programování II

Martin Pergel

3. března 2021

Užitečné informace

- Navážeme na Programování 1,
- změníme jazyk na C#,
- podrobnosti o C# budou v navazujícím kurzu,
- soustředíme se na totéž, co v zimě, tedy:
- technické programování, návrh a implementace algoritmů, datové struktury a související teorii.

Pragmatické informace

- Zápočet: zápočtový program (s dokumentací), aktivní účast na cvičení (body v CodExu), praktický test, další dle pokynů cvičících.
- Zkouška: Za oba semestry.

Další informace

- C# je představitelem jazyka z rodiny jazyku C.
- Uvidíme jazyk úplně jinak navržený nežli v zimě.
- Zjistíme, co všechno se dá v programovacích jazycích dělat jinak.
- Budeme masívně stavět na Pythonu.

- Budeme používat ReCodEx,
- využívat budeme buďto Microsoft Visual Studio (v dostupné verzi),
- nebo prostředí MONO (obojí je k dispozici legálně zdarma aspoň v omezené verzi).
- Tvořit budeme převážně konzolové aplikace (o psaní formulářových si taktěž něco řekneme).

Literatura

- M. Virius: C# pro zelenáče, Neocortex Praha, 2002,
- E. Gunnerson: Začínáme programovat v C#, Computer Press Praha, 2001,
- J. Kent: Visual C# 2005 bez předchozích znalostí, Computer Press Brno, 2007,
- J. Liberty, D. Xie: Programming C# 3.0, Fifth Edition, O'Reilly Media Inc., 2007.

Objektové programování

aneb proč nebyl C# už v zimě?

- Vše je objekt, vše má metody a atributy se všemi důsledky, které z toho plynou.
- Objekty a třídy, metody a atributy.
- I program je objekt (s metodami). Vždy je definováno, jak se s objektem programu zachází.
- Program neběží odnikud nikam, ale spustí se mu metoda `Main`.

Krok stranou

Rodina jazyka C

- B. Kernighan a D. Ritchie napřed navrhli jazyk A...
- tým byl spokojen až s jazykem C.
- Ač byl tento jazyk navrhován údajně jako recese, velmi se ujal.
- Za jeho potomky lze prohlásit C++, Java, C#, Javascript, PHP, Python a další.

Společné vlastnosti

- Jsou case-sensitive,
- mají podobnou syntax,
- u neobjektových jazyků se zpravidla spustí funkce `main` (až na velikost písmen),
- příkazy se ukončují středníkem.

Krok stranou

Jazyk C

hello.c

- `#include <stdio.h>`
- `int main()`
- `{ printf("Hallo, world!\n");`
- `}`

Krok stranou

Jazyk C

k_nicemu.c

```
■ #include <stdio.h>
■ void f()
■ {    printf("Hallo, world!\n");
■ }
■ int main()
■ {    f();
■ }
```

Proměnné

a jejich definice

- Proměnné je třeba definovat, tedy...
- určit místo (kde mají fungovat)
- a datový typ.

Příklad v C, C++, C# i Javě

definice proměnných a funkce, základní řídicí struktury

- *Datový typ se píše napřed!*
- `int faktorial(int a)`
- `{` `int b=1;`
- `while(a>1)b*=a--;`
- `return b;`
- `}`

Nejdůležitější typy

- `void` – prázdný datový typ – zahození hodnoty,
- `char` – obvykle jeden byte – měřák velikosti,
- `short` – celočíselný typ,
- `int` – nejobvyklejší celočíselný typ,
- `long` – celočíselný typ,
- `float` – neceločíselný typ,
- `double` – neceločísl. typ s dvojnásobnou (doublovou) přesností,
- **C#** `string` – řetězec **C** – **ne C#** `pointery`.

Příklad

- `int a,b,c=10,d=100;`
- `double x=1.15;`
- `char a='a',b='x';`
- `string text="nic";`
- `void nanic(int a, int b, int c);`
- **parametry funkce při definici se oddělují čárkou.**
- **konstanty modifikátorem `const`:** `const double pi=3.1415926;`

Základní operátory

- + (binární) sčítání,
- - odečítání, podobně * a /
- = přiřazení,
- == porovnání na rovnost, !=, >, <, >=, <= (nerovnosti),
- logické && (and), || (or), ! (not), & (and), | (or), ^ (xor),
- první dvě vyhodnocují líně (je-li výsledek jasný, přestanou), 4. a 5. vyhodnocují úplně (takto v C#, v C to bylo trochu jinak).
- unární ++, -- (prefixové a postfixové, tedy a++ vs ++a).
- Pozor na priority!
- Řešení jako v Pythonu, tedy závorkovat.
- Pozor na porovnání a přiřazení!
- Přiřazovací příkazy: +=, -=, *=, /=, &= apod.

Komentáře, bloky

- Jednořádkové `//` do konce řádku
- Víceřádkové `/*` komentář hodně dlouhý dokud ho neukončíme explicitně `*/`
- Složené závorky označují blok (jako v Pythonu indentace), tedy `{ blok; příkazů; }`
- programy je tedy možno psát slušně nepřehledně.

Základní řídicí struktury

- `if (podm) příkaz;`
- `if (podm) příkaz; else příkaz;`
- `if (podm) { blok } [else {blok}]`
- `while(podm) příkaz_nebo_blok`
- `for(init;podm;inkr) tělo`
- *init je inicializační kód, podm je podmínka, inkr je inkrementační výraz, tedy příklad:*
- `for(a=1;a<10;a++)b+=a;`
- *Co je tohle?* `if(a=5) a_je_pet();`
- `do příkazy(); while(podm);`
- *Poznámka:* `a=b=c=1;`

Konstrukce switch

lepší než elif

- `switch(i)`
- `{` `case 1: i_je_jedna(); break;`
- `case 2: case 3: i_je_2_nebo_3(); break;`
- `default: vsechno_je_jinak(); break;`
- `}`

Break ukončuje jednotlivé větve, v C se vykonával kód, dokud se nenarazí na `break`, C# toto neumožňuje (jednu větev je třeba ukončit před začátkem druhé).

Definice funkcí

```
typ jmeno(parametry)
{ telo }
```

Chceme-li vrátit návratovou hodnotu, použijeme klíčové slovo `return` a za ně umístíme vrácený výraz: `return 0;`

Definice funkcí

podrobnosti

- procedura je funkce, která vrací void,
- void vracíme pomocí return;;
- kulaté závorky jsou operátor zavolání (nebo operátor definice funkce).
- Parametry jsou předávány podobně jako v Pythonu (hodnotové a referenční typy). Jazyk C předání referencí neměl (místo toho byly pointery).
- C# umí předávat parametry též referencí a výsledkem:
- `void divna(ref int a, out int vysledek);`
- proměnná `a` je předána referencí, proměnná `vysledek` výsledkem, což je téměř totéž (tedy je řešena též odkazem, ale hodnota proměnné předávané výsledkem není funkci vůbec předána).

Objektové programování I

- Objekty, třídy, metody a atributy.
- Svět sestává z objektů, většina věcí je objekt.
- Objekt student se na přednášce vyskytuje několikrát,
- jednotlivci přísluší do třídy student.
- Jsou vybaveni atributy a metodami (aby mohli studovat).

Objektové programování II

- Objekty jsou instance třídy.
- Statické metody (resp. atributy) přísluší třídě (ne jednotlivým objektům).
- Modifikátory `public`, `private`, `protected` řídí přístup k jednotlivým položkám (může každý / může jen metoda dané třídy / smí současná třída a potomek).
- Třidu (v C++, Javě, C# a dalších) vytváříme pomocí klíčového slova `class`.

Rituál programování v C#

```
using System;
using System.Collections.Generic;
using System.Text;
namespace ConsoleApplication1
{
    class Program
    {
        static void Main(strings[] args)
        {
            Console.WriteLine("Sem se bude psát
ten program!");
        }
    }
}
```

Operátor tečky má podobný význam jako v Pythonu.

Eukleidův algoritmus

```
static void Main(string[] args)
{
    Console.WriteLine("Dvě čísla sem:");
    int a=int.Parse(Console.ReadLine());
    int b=int.Parse(Console.ReadLine());
    while(a!=b)
        if(a>b) a-=b;
        else b-=a;
    Console.Write("NSD je: ");
    Console.WriteLine(a);
}
```

Praktická ukázka Visual Studia...

- ... a to bude pro dnešek všechno.
- Děkuji za pozornost.