Jazyk C# a platforma .NET

Katedra softwarového inženýrství
Fakulta informačních technologií
České vysoké učení technické v Praze

Syntaxe jazyka C# - 2. část
BI-DNP

```
// Zaklady jazyka C# (druha cast)

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
// pridano!!
using System.Collections;   // kvuli ArrayList

namespace JayzkCS2 {
    public partial class Form1 : Form {
        public Form1() {
            InitializeComponent();
        }

        private void btnPole_Click(object sender, EventArgs e) {

            // pole v C#

            // deklarace pole
            int[] pole;
            // pole[0] = 2;     // nelze - pole NEMA alokovanu pamet
            pole = new int[10];
            pole[3] = 11;

            // deklarace kompaktněji (zahrnuta alokace pameti)
            int[] p = new int[10];

            // inicializovane pole
            int[] primes = new int[4] { 2, 5, 7, 11 }; // delka shodna!!

            // inicializace zkracene
            string[] jmena = { "Franta", "Eva", "Karel" };  // triprvkove pole

            // vicerozmerna pole
            int[,] matice = new int[3, 4];
            matice[1, 1] = 25;

            // vicerozmerne s inicializaci
            int[,] m = {{1,2,3},{4,5,6}};   // pole o 2 radcich, 3 sloupcich

            // nektere vlastnosti a metody poli
            txtVystup.Text = jmena.Length.ToString();         // delka pole
            txtVystup.Text = matice.Length.ToString();        // vsechny prvky
            txtVystup.Text = matice.GetLength(1).ToString(); // delka rozmeru
            txtVystup.Text = m.Rank.ToString();               // pocet rozmeru
```

```
        // "zubata" pole
        int[][] zubate = new int[3][];

        zubate[0] = new int[5];
        zubate[1] = new int[2];
        zubate[2] = new int[3];

        zubate[1][1] = 15;


        // ukazka kolekce
        ArrayList seznam = new ArrayList();

        seznam.Add("Prvni");          // pridani prvku do kolekce
        seznam.Add("Druhy");
        seznam.Add("Treti");
        seznam.Add("Paty");
        seznam.Add("Sesty");

        txtVystup.Clear();

        // cyklus pres prvky kolekce (u ArrayList možno pouzit indexy)
        for (int i = 0; i < seznam.Count; i++) {
            txtVystup.Text += seznam[i] + "\r\n";
        }

        seznam.Insert(3, "Ctvrty");    // vlozi do kolekce

        txtVystup.Clear();

        //  foreach cyklus pres prvky kolekce
        foreach (object prvek in seznam) {
            txtVystup.Text += prvek + "\r\n";
        }
    }
```

```
private void btnFunkce_Click(object sender, EventArgs e) {

    // funkce v jazyce C# - volani funkci

    Pozdrav("Franta");

    int i = Soucet(2, 3);

    int n;
    n = 1;

    // predavani parametru hodnotou
    ZvysHodnotou(n);
    txtVystup.Text = n.ToString();

    // predavani parametru odkazem (adresou, referenci)
    n = 1;
    ZvysReferenci(ref n);
    txtVystup.Text = n.ToString();

    // vystupni parametr
    int k;           // pro VYSTUPNI parametr nemusim inicializovat
    Nastav(out k);
    txtVystup.Text = k.ToString();

    // pouziti named parametrs
    Uloz("Karolina", "Svetla");
    // ...
    Uloz("Nemcova", "Bozena");                  // chybne poradi
    Uloz(Prijmeni: "Nemcova", Jmeno: "Bozena"); // named parameters OK

    // pouziti optional parametru
    Pojistka("Karel", 35, "muz");
    // Pojistka("Eva",,"zena"); // nelze
    Pojistka("Eva", Pohlavi: "zena"); // OK
    Pojistka("Cosi",25);
    Pojistka("Jakesi");

    // pole parametru stejneho typu na konci seznamu
    txtVystup.Text = Suma("Franta",8,6,2).ToString();
    txtVystup.Text = Suma("Jirka",8,9,8,8,6,10,8,9,8).ToString();

}
```

```
        // definice volaných funkci

        // funkce typu procedury (nevraci nic - void)
        void Pozdrav(string Jmeno) {
            MessageBox.Show("Zdravime " + Jmeno);
        }

        // Funkce vracejici hodnotu typu integer
        int Soucet(int p1, int p2) {
           return p1 + p2;
        }

        // predavani parametru hodnotou
        void ZvysHodnotou(int p) {
            p += 2;
        }

        // predavani parametru adresou (odkazem, referenci)
        void ZvysReferenci(ref int p) {
            p += 2;
        }

        // vystupni parametr
        void Nastav(out int p) {
            int t;
            // t = p;   // nelze - out parametr NEMUSI byt inicializovan!!
            p = 5;
        }

        // named parameters (pojmenovane parametry)
        // deklarace je standardni, jmena se pouziji při volani
        void Uloz(string Jmeno, string Prijmeni) {
            // ...
        }

        // optional (volitelne) parametry - MUSI by na konci seznamu parametru
        // a MUSI mit definovanu implicitni hodnotu
        void Pojistka(string Jmeno, int Vek = 80, string Pohlavi = "zena") {
            // ...
        }

        // pole parametru stejneho typu (zde int) na konci seznamu
        int Suma(string Jmeno, params int[] Cisla) {
            int Total = 0;

            for (int i = 0; i < Cisla.Length; i++) {
                Total += Cisla[i];
            }
            return Total;
        }
    }
}
```