Jazyk C# a platforma .NET

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

Functory
BI-DNP

```
using System;
using System.Collections.Generic;
using System.Windows.Forms;

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

        // Mejme kolekci cisel ze ktere chceme vybirat podle
        // urcitych podminek - napr.:
        public List<int> cisla = new List<int>() { 3, 4, 7, 1, 8,
                                    10, 21, 5, 9, 11, 14, 19 };

        // mozno definovat funkci pro jeden vyber (zde nepouziji)
        public List<int> Vyber1(List<int> numbers) {
            List<int> selected = new List<int>();  // prazdny seznam
            foreach (int number in numbers)
                if (number > 10) selected.Add(number);
            return selected;
        }

        // potom definuji druhou fci pro vyber (take nepouziji)
        public List<int> Vyber2(List<int> numbers)
        {
            List<int> selected = new List<int>();
            foreach (int number in numbers)
                if ((number / 5) > 1) selected.Add(number);
            return selected;
        }

        // a dalsi funkce pro vybery ...

        // obecne (prvni) reseni pomoci definice
        // "standardniho" delegata (delegat definovan "natvrdo")
        public delegate bool PodminkaDelegate(int p);

        public List<int> VyberDelegatem(List<int> numbers,
                                    PodminkaDelegate numberResolver) {
            List<int> selected = new List<int>();
            foreach (int number in numbers)
                if (numberResolver(number)) selected.Add(number);
            return selected;
        }
        // a volat s parametrem (delegatem) pro prislusne funkce
        // (zde neni realizovano)
```

```
// obecne (druhe) reseni pomoci functoru - BEZ
// explicitni definice delegata

// Functory jsou vlastne preddefinovane parametricke
// (generické - ve smyslu generiky) delegaty.
// Maji 0 - 16 parametru, ktere odpovidaji parametrum metod,
// pro kterou vytvori delegáta. Poslední parametr (out),
// odpovidajici typu, vracenemu z dane metody

// priklady ruznych funkci o nula, jednom, ... parametru
/*
int VratMaxID()
string VratPosledniJmeno() ...

int Mocnina(int x)
string NaVelka(string Nazev) ...
*/

// definice funktoru v .NET:
// funktor pro funkci bez parametru:
// delegate TResult Func <out TResult> ();
// funktor pro funkci s jednim parametrem:
// delegate TResult Func <in T, out TResult> (T arg);
// funktor pro funkci se dvema parametry:
// delegate TResult Func <in T1, in T2, out TResult>
//                                   (T1 arg1, T2 arg2);
// ... az do T16

public List<int> VyberFunctorem(List<int> numbers,
                        Func<int, bool> numberResolver) {
    List<int> selected = new List<int>();

    foreach (int number in numbers)
       if (numberResolver(number)) selected.Add(number);
    return selected;
}

private void btnProved_Click(object sender, EventArgs e)
{
    txtVystup.Clear();

    // parametr typu delegate (podminka) je v obou
    // pripadech zadan jako lambda vyraz

    // prvni vyber
    List<int> prvni = VyberFunctorem(cisla, x => x > 10);
```

```
            // vypis vysledku
            foreach (int i in prvni) txtVystup.Text += i + "\r\n";

            txtVystup.Text += "\r\n";

            // druhy vyber
            List<int> druhy = VyberFunctorem (cisla, n => (n/5) > 1);
            // vypis vysledku
            foreach (int i in druhy) txtVystup.Text += i + "\r\n";
        }
    }
}
```