Číslo a název šablony	III/2 Inovace a zkvalitnění výuky prostřednictvím ICT
Číslo didaktického materiálu	EU-OPVK-VT-III/2-ŠR-210
Druh didaktického materiálu	DUM
Autor	RNDr. Václava Šrůtková
Jazyk	čeština
Téma sady didaktických materiálů	Programování v C# v příkladech II
Téma didaktického materiálu	Pole jako úložiště dat, řetězce
Vyučovací předmět	Seminář z informatiky
Cílová skupina (ročník)	Žáci ve věku 16–17 let
Úroveň žáků	Mírně pokročilí
Časový rozsah	1–2 vyučovací hodiny
Klíčová slova	Disky, řetězce, pole znaků
Anotace	Studenti si prohlubují znalosti o poli prací s polem disků počítače a programováním některých operací s řetězci.
Použité zdroje	TÖPFEROVÁ, Dana a Pavel TÖPFER. *Sbírka úloh z programování*. Vyd. 1. Praha: Grada, 1992, 98 s. Educa '99. ISBN 80-854-2499-1. VYSTAVĚL, Radek. *Moderní programování: sbírka úloh k učebnici pro středně pokročilé*. 1. vyd. Ondřejov: moderníProgramování, 2008-2009, 2 sv. ISBN 978-80-903951-3-8. VYSTAVĚL, Radek. *Moderní programování: učebnice pro středně pokročilé*. Ondřejov: moderníProgramování s.r.o, 2008. ISBN 978-80-903951-2-1.
Typy k metodickému postupu učitele, doporučené výukové metody, způsob hodnocení, typy k individualizované výuce apod.	Text je možno využít ke společné práci, samostatné přípravě studentů, domácímu studiu apod. Při společné práci je vhodné nejprve obtížnější úlohy rozebrat, potom společně se studenty implementovat na počítači. (Rozbor nejlépe na tabuli, synchronní řešení s promítáním) Prezentace obsahuje stručné shrnutí poznatků potřebných pro řešení příkladů. V pracovním listu je zadání cvičení – většinou se jedná o úlohy, které by měli studenti naprogramovat samostatně. Není nutné, aby všichni zpracovali všechno, vhodné je diferencovat podle jejich zájmu a schopností. Obtížnější úlohy jsou označeny hvězdičkou. Součástí materiálu je zdrojový kód těchto příkladů. Návrh způsobu hodnocení: ohodnocení samostatné práce během hodiny např. podle volby a počtu úloh a elaborace

	řešení (efektivnost, komentáře…).

Metodický list k didaktickému materiálu

Prohlášení autora

Tento materiál je originálním autorským dílem. K vytvoření tohoto didaktického materiálu nebyly použity žádné externí zdroje s výjimkou zdrojů citovaných v metodickém listu.

Obrázky (schémata a snímky obrazovek) pocházejí od autora.

210. Pole jako sklad dat, řetězec jako pole znaků

Pole mohou sloužit také jako kontejnery vstupních a výstupních dat. Potom se mohou vyskytovat jako parametry metod. Dosud jsme používali metody, jejichž smyslem byla akce – např. refresh() a metody, které měly jedinou návratovou hodnotu – ToInt32()…

Pokud potřebujeme, aby metoda vracela několik dat stejného druhu, bude její návratová hodnota zřejmě pole.

(Pro data různého typu je třeba vytvořit vlastní objekt)

Jako příklad si vyzkoušíme statickou metodu GetLogicalDrives třídy Environment. Její návratová hodnota je pole řetězců, které obsahuje seznam diskových jednotek počítače.

Příklad 1

Při stisknutí tlačítka ButtonDisky zobrazte do textového pole diskové jednotky počítače.

```
string[] Disky = Environment.GetLogicalDrives();

            MessageBox.Show("Počet disků je " + Disky.Length.ToString());

            for (int i = 0; i < Disky.Length; i++)

            {

                textBoxDisky.Text += Disky[i] + Environment.NewLine;

            }
```

Na řetězce se můžeme dívat také jako na pole znaků. Znak má datový typ char, jednotlivé znaky řetězce jsou ale pouze ke čtení. (dá se eventelně obejít přes typ char [], ale přetypování není implicitní). Můžeme si ale pomoci pomocným řetězcem. (viz Příklad 3) Počet znaků udává délka řetězce – Length.

Příklad 2

Zjistěte počet znaků *a* ve větě zadané v textovém políčku

```
int pocet = 0;
```

```
string veta = textBoxVeta.Text;
for (int i = 0; i < veta.Length; i++)
       {
       if (veta[i]=='a')
{
    pocet++;
}
}
MessageBox.Show("Počet znaků a je " + pocet.ToString());
```

Příklad 3

Nahraďte všechny znaky *a* ve větě zadané v textovém políčku znakem *.

```
string veta = textBoxVeta.Text;
string veta2 = null; ;

for (int i = 0; i < veta.Length; i++)
{
    if (veta[i] == 'a')
        veta2+= "*";
    else
      veta2 += Convert.ToString(veta[i]);
}
textBoxVeta.Text = veta2;
```

Poznámka: Všimněte si, že do věty veta2 vkládáme jednoznakové řetězce `"*"`, nikoliv znaky.

Důležité

Metoda GetLogicalDrives třídy Environment vrací pole řetězců, které obsahuje seznam diskových jednotek počítače.

Řetězce jako pole znaků – retezec[i] je i-tý znak řetězce, vlastnost je jen ke čtení.

Pracovní list

Cvičení

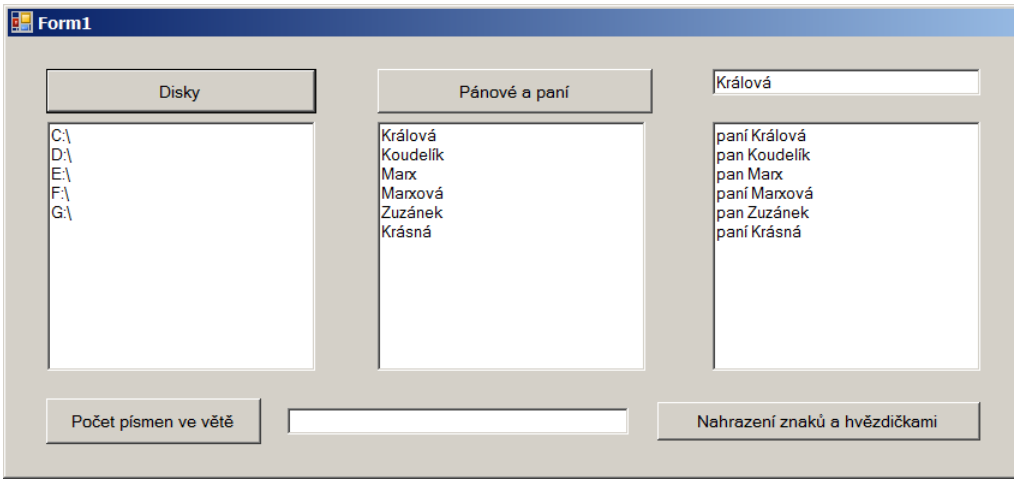
1. Vstupem je příjmení v textBoxu. Pro jednoduchost budeme předpokládat, že všechna česká ženská příjmení končí na *á*. Zjistěte, zda se jedná o muže nebo ženu

2. Vstupem je pole příjmení ženských a mužských (alespoň 6) – zadáno při deklaraci pole. Do jednoho textBoxu vypište toto pole, do druhého pole upravené – před příjmením bude oslovení pan nebo paní.

3. (*) Vstupem je jméno a příjmení v textBoxu, oddělené mezerou. Zobrazte samotné jméno a příjmení.

4. Vyhledejte nejdelší příjmení z pole v příkladu 2.

Poznámka: smyslem cvičení je cvičit. Podobně jako v Excelu najdete na dané problémy šikovné funkce v dokumentaci. (Řetězce mají například metodu split, s parametrem znak, která je rozdělí na pole řetězců, kde daný znak funguje jako oddělovač.)



Řešení

```
 string[] osoby = new string[]

   {"Králová", "Koudelík","Marx","Marxová", "Zuzánek", "Krásná"

   };
```

1, 2

```
//je pan nebo paní?

int posled=textBoxKdo.Text.Length-1;

string jmeno=textBoxKdo.Text;

if (jmeno[posled] == 'á')

    MessageBox.Show("paní");

else
```

```
MessageBox.Show("pan");
        for (int i = 0; i < 6; i++)
        {
            textBoxVstup.Text += osoby[i] + Environment.NewLine;
            posled = osoby[i].Length - 1;
            if (osoby[i][posled] == 'á')
                osoby[i] = "paní " + osoby[i];
            else
                osoby[i] = "pan " + osoby[i];
            textBoxVystup.Text += osoby[i] + Environment.NewLine;
        }
```

3.

```
        //oddělení jména a příjmení
        string celeJmeno = textBoxKdo.Text;
        string jmeno=null;
        string prijmeni=null;;
        int i = 0; //1. znak
        while (celeJmeno[i] != ' ')
        { //dokud nenarazíme na mezeru, ukládáme znaky do jména
            jmeno += celeJmeno[i];
            i++;
        }
        i++; //přeskočíme mezeru a zbytek naskládáme do příjmení
        for (int k = i; k < celeJmeno.Length; k++)
        {
            prijmeni += celeJmeno[k];
        }
        MessageBox.Show(jmeno + Environment.NewLine + prijmeni);
```

4.

```
    string nejdelsi = osoby[0];
```

```
for (int i = 1; i < 6; i++)
{

    if (osoby[i].Length>nejdelsi.Length)
        nejdelsi=osoby[i];

}
MessageBox.Show(nejdelsi);
```