Číslo a název šablony	III/2 Inovace a zkvalitnění výuky prostřednictvím ICT
Číslo didaktického materiálu	EU-OPVK-VT-III/2-ŠR-315
Druh didaktického materiálu	DUM
Autor	RNDr. Václava Šrůtková
Jazyk	čeština
Téma sady didaktických materiálů	Programování v C# v příkladech III
Téma didaktického materiálu	Knihovny DLL
Vyučovací předmět	Seminář z informatiky
Cílová skupina (ročník)	Žáci ve věku 17–18 let
Úroveň žáků	Středně pokročilí
Časový rozsah	1–2 vyučovací hodiny
Klíčová slova	Dynamicky připojené knihovny, statické metody
Anotace	Studenti vytvářejí vlastní DLL, procvičují si přitom práci s třídami a jejich metodami
Použité zdroje	DRÓZD, Januš a Rudolf KRYL. *Začínáme s programováním*. 1.vyd. Praha: Grada, 1992, 306 s. ISBN 80-854-2441-X. ELLER, Frank. *C# - začínáme programovat: podrobný průvodce začínajícího uživatele*. 1. vyd. Praha: Grada, 2002, 240 s. ISBN 80-247-0324-6. KUKAL, Jaromír. *Myšlením k algoritmům*. Praha: Grada, 1992, 131 s. ISBN 80-854-2447-9. 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 pokročilé*. Ondřejov: moderníProgramování, 2011, 149 s. ISBN 978-80-903951-7-6. 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.

315. DLL

Jako knihovna se označuje soubor obsahující třídy a podprogramy, které mohou používat jiné programy. V současnosti se používá hlavně dynamické připojování knihoven (Dynamic Link Library), kdy se kód knihovny nepřipojuje do .EXE souboru (statické propojení), ale soubor existuje v počítači v jediném exempláři pro všechny programy, které ho používají a zavádí se do paměti při startu programu nebo požadavku na své podprogramy. Většinu knihoven patformy .NET můžeme najít ve složce C:\Windows\Microsoft.NET\Framework\v2.0.50727.

Vyrobíme si vlastní knihovnu, která zatím bude obsahovat převod celého čísla z desítkové do dvojkové soustavy a ve cvičení ji rozšíříme o další převody.

Vytvoření knihovny

Připravíme nový projekt, tentokrát ale ne Windows Forms application, ale ze šablony **Class Library**. Novou knihovnu pojmenujeme (Name) Matematika.

Vývojové prostředí otevře kód třídy:

```
namespace Matematika
```

```
{
    public class Class1
    {
    }
}
```

Pomocí místní nabídky (Refactor-Rename) přejmenujeme třídu na Matika.

V Solution Exoloreru přejmenujeme zdrojový kód Class1.cs na Mat.Cs.

Naprogramujeme metodu. Vstupem bude její parametr – desítkové celé číslo, výstupem řetězec, skládající se z nul a jedniček.

Použijeme známý algoritmus: vstupující číslo postupně dělíme dvěma a zbytky přidáváme zezadu k výstupnímu řetězci, dělení opakujeme, dokud výsledek není nula.

6 : 2 = 3 : 2 =1 : 2 = 0, takže $6_{10} = 110_2$

0 1 1

Knihovnu uložíme a sestavíme:

Knihovna se nedá spustit, můžeme ji ale sestavit – Build/Build Solution nebo klávesou F6.

```
namespace Matematika
{
    public class  Matika
    {
        public string DecBin(int n)
        {
            string s = null;
            do
            {
                int zbytek = n % 2;
                s = zbytek.ToString() + s;
                n /= 2;
            }
            while (n == 0);
            return s;
        }
```

```
    }
}
```

V podsložce bin\Release vznikl soubor knihovny Matematika.dll.

Poznámka: ve všech dnešních příkladech využíváme pouze bezparametrických konstruktorů, které poskytuje C#. Parametrické si můžete doprogramovat sami.

Použití knihovny

Otevřeme nový projekt s jedním textovým polem pro vstup a tlačítkem k výpočtu.

Přidáme do svého projektu příslušný odkaz: Project/Add Reference – tlačítkem Browse vyhledáme soubor Matematika.dll a potvrdíme OK.

V novém programu přidáme nahoře řádek: `using Matematika;`

a můžeme program vyzkoušet.

Nejprve vytvoříme instanci třídy Matika a pak můžeme používat její metody.

```
public partial class Form1 : Form
    {
        Matika M = new Matika();
```

…

```
private void button1_Click(object sender, EventArgs e)
        {
            int x = Convert.ToInt32(textBoxVstup.Text);
             string s = M.DecBin(x);
              MessageBox.Show(s);
        }
```

Definujme ve jmenném prostoru Matematika ještě třídu vektor s metodou pro výpočet jeho velikosti a vyzkoušejme si ji v programu.

```
namespace Matematika
{
public class Vektor
    {
        public double x, y;
        public double velikost()
```

```
        {
            return Math.Sqrt(x * x + y * y);
        }
    }
…
private void buttonVV_Click(object sender, EventArgs e)
        {
            Vektor v = new Vektor();
            v.x = Convert.ToDouble(textBoxu1.Text);
            v.y = Convert.ToDouble(textBoxu2.Text);
            double velikost = v.velikost();
            MessageBox.Show(velikost.ToString());
        }
```

Statické složky třídy

To, že chceme-li převádět číslo do dvojkové soustavy, musíme vytvořit novou instanci třídy matika, nám není příliš pohodlné. Naprogramujme si tedy ještě statickou metodu i pro převod desítkového čísla do soustavy o základu z<10. Postup je stejný jako u převodu do dvojkové soustavy, dělí se základem. Základ bude také druhým vstupním parametrem metody. V definici je třída označena **static,** v programu ji pak voláme přes název třídy. (Jako třeba Color.FromARGB…)

```
public static string DecZ(int n, int z)
        {
            string s = "";
            do
            {
                int zbytek = n % z;
                s = zbytek.ToString() + s;
                n /= z;
            }
            while (n != 0);
            return s;
        }
    }
```

```
private void buttonDZ_Click(object sender, EventArgs e)
        {
            int x = Convert.ToInt32(textBoxVstup.Text);
            int z = Convert.ToInt32(textBoxZaklad.Text);
            string s = Matika.DecZ(x,z);
            MessageBox.Show(s);
        }
```

Pracovní list

Cvičení:

1. (*) Přidejte dále pro převod desítkového čísla do šestnáctkové soustavy. Opět funguje stejný algoritmus, jen zbytky 10,11, až 15 nahradíme písmeny A, až F.

2. Připojte další metody vektoru – skalární součin s jiným vektorem (vstupní parametr) a zjištění, zda je kolmý k jinému vektoru.

3. Připojte třídu kruh s metodami obvod a obsah.

Všechny nové metody vyzkoušejte ve svém programu.

(vstupní parametr)

Řešení

1.

Matematika:

```
public static string DecHex(int n)
        {
            //mohli bychom řešit stejně jako v předchozích případech
            //a u zbytků větších než deset program vhodně rozvětvit,
            //ale také pro zbytky můžeme použít pole znaků.
            string[] zbytky = new string[16]{"0","1","2","3","4","5","6","7",
                "8","8","A","B","C","D","E","F"};
            string s = "";
            do
            {
                int zbytek = n % 16;
```

```
            s = zbytky[zbytek] + s;
            n /= 16;
        }
        while (n != 0);
        return s;
    }
```

Program:

```
private void buttonDS_Click(object sender, EventArgs e)
    {
        int x = Convert.ToInt32(textBoxVstup.Text);
        string s = Matika.DecHex(x);
        MessageBox.Show(s);
    }
```

2.

Matematika

```
public double SkalSoucin(Vektor v)
    {
        return x * v.x + y *v.y;
    }
    public bool Kolmost(Vektor v)
    {
        return (SkalSoucin(v)==0);
    }
```

Program:

```
private void buttonSS_Click(object sender, EventArgs e)
    {
        Vektor v = new Vektor();
        Vektor u = new Vektor();
```

```
            v.x = Convert.ToDouble(textBoxu1.Text);
            v.y = Convert.ToDouble(textBoxu2.Text);
            u.x = Convert.ToDouble(textBoxv1.Text);
            u.y = Convert.ToDouble(textBoxv2.Text);
            double ss = v.SkalSoucin(u);
            MessageBox.Show(ss.ToString());
            if (v.Kolmost(u))
                MessageBox.Show("Jsou kolmé");
            else
                MessageBox.Show("Nejsou kolmé");
        }
```

3.

Matematika

```
    class Kruh
    {
        public double r;
        public double Obvod()
        {
            return Math.PI * r * 2;
        }
        public double Obsah()
        {
            return Math.PI * r * r;
        }
    }
```

Program:

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

            Kruh k = new Kruh();
            k.r=Convert.ToDouble(textBoxR.Text);
            MessageBox.Show("S = "+k.Obsah().ToString()+Environment.NewLine+
                     "O = " + k.Obvod().ToString());

        }
```