Střední škola pedagogická, hotelnictví a služeb, Litoměřice, příspěvková organizace

Předmět:	**Učební praxe**
Téma:	**Visual Studio**
Vyučující:	**Ing. Milan Káža**
Třída:	**EK3**
Hodina:	**průběžně**
Číslo:	**V/5**

Programování v jazyce C a C#

Visual Studio

9. Další příkazy pro C#

V rámci Visual Studia se používá mnoho příkazů pro rychlé a efektivní zpracování dat, které do programu vkládá uživatel. Ne všechny příkazy, které se používají ve Visual Studio, se dají použít čistě v C#, to je právě výhoda podpory Windows.

9.1. Příkazy pro práci se soubory

O základních příkazech pro soubory bylo již napsáno nyní se uvedou ty, které se používají s podporou Visual Studia a hlavně Windows.

Všechny příkazy se nalézají v knihovně System.IO. Tu je nutné zavést na začátku programu. Nyní budou příkazy uvedeny.

Zjištění existence souboru

```
File.exists( cesta k souboru)
```

Smazání souboru

```
File.Delete(cesta k souboru)
```

Přesunutí souboru z jedné složky do druhé

```
File.Move(cesta k souboru, kam soubor vložit)
```

Vytvoření nového souboru

```
File.Create(cesta k soubor vytvořit)
```

Zkopírování souboru

```
File.Copy(jaký soubor, kam se má soubor zkopírovat, potvrzení přepsání)
```

9.2. Práce s datumem a časem

Nyní budou uvedeny některé příkazy pro práci s datumem a časem. Ty se využívají při ukládání důležitých dat např. při měření, nebo vytvoření souboru, jeho poslední aktualizace datum a čas se dá také použít při kodóvání dat v souboru.

Zjištění aktuálního datumu

```
string dnesje= DateTime.Now.Date.ToString();
```

Zjištění, kolikátý den je v roce

```
int denvroce=DateTime.Now.DayOfYear;
```

Zjištění počtu dnů v daném měsíci

```
int pocetdni= DateTime.Now.DaysInMonth( aktuální rok, měsíc);
```

9.3. Práce s obrázky a grafikou

Obrázky se velmi často objevují v různých programech buď jako pozadí, nebo při vysvětlení, správném výsledku apod.

Osobní zkušenost: *Dělal jsem jednoduchý program pro výpočty z počtů pro první třídu a pokud dítě dobře vypočítalo zadané příklady, tak se ojevil dalmatín (v té době letěli). Proto je zobrazení obrázku velmi důležité.*

Barvy (Color) jsou deklarovány v knihovně **System.Drawing** a je nutné požít název **Color**.

Přístup k barvám

```
Label.BackColor=System.Color.Windows;
```

Tyto barvy jsou předefinované ve Windows a je možné si je měnit. Tento princip byl již popsán.

V barvách je možné vytvořit i pero, které se pak používá při kreslení čar. Per je definované v knihovně **System.Drawing.Pen**

Zavedení pera

```
Pen pero = new Pen(Color.Red);
```

Pero bude kreslit čeveně.

Dále je možné pro kreslení vyžít aplikaci Štětec (Brush). Štětcem se vyplňují geometrické útvary.
Vytvoření štětce černé barvy

```
SolidBrus testbrush =new SolidBrush(Color.Black);
```

Vytvoření štětce se vzorkem

```
HatchBrush hb= new HatchBrush(HatchStyle.Cross, Color.White);
 Graphics g =this.CreateGraphics();
 g.FillRectangle(hb, 100, 100, 200, 200 );
```

HatchBrush připravuje štětec na základě vzorku. Štětec a práce s ním je deklarován knihovně **System.Drawing.Brushes**.

V rámci Visual Studia je možné vytvořit i různé geometrické objekty jako je elipsa, kružnice apod. Pro názornost bude uvedena kresba elipsy.

```
private void Elipsa()
        {
         Graphics eg = this.CreateGraphics();
         eg.DrawEllipse(Pens.Black, 50, 50, 40, 20);
        }
```

Rozbor Elipsy: *Zavolejte knihovnu* `Graphics` *a zadejte proměnnou, poté zadejte* `this.CreateGraphics()` **this** *zajistí vytvoření instance. Potom zavedete proměnnou a vyberete z nabídky jaký geometrický útvar požadujete, vložíte příkaz pera pro barvu útvaru a zadáte číselné hodnoty, které vyjadřují: první dvě (50,50) zadávají kde se zobrazí Elipsa,první 50 jeod horního okraje, druhá 50 je od levého okraje, další dvě číslice (40,20) udávají velikost Elipsy.*

Kód pro práci s perem, pomocí tohoto kódu se vytvoří přerušovaná čára o zvolené tloušťce, kterou je možné měnit.

```
private void Pera()
    {
       Graphics gt=this.CreateGraphics();
       Pen pero = new Pen(Color.Red, 5);
       gt.DrawLine(System.Drawing.Pens.Blue ,200,200,250,250);
       pero.DashStyle = System.Drawing.Drawing2D.DashStyle.Dash;
      gt.DrawLine(pero, 50, 50, 250, 150);

    }
```

Kód pro práci s textem, často se zobrazený text natáčí, pro efekt nebo z důvodů vertikálního popisu, zde je ukázka aplikace pro natočení textu.

```
private void HryTexty()
{
    string mujtext = "To je muj text";
    Graphics tgt = this.CreateGraphics();
    SolidBrush txbr = new SolidBrush(Color.Blue);
    tgt.TranslateTransform(200, 10);
    tgt.RotateTransform(32);
    tgt.DrawString(mujtext, this.Font, txbr, 150, 50);

}
```

`tgt.RotateTransform(32);` Tento příkaz zajišťuje natočení textu na úhel v tomto případě 32°.

Kód pro natočení obrázku, opět vzniká efekt obrázku a z něj mohou např. vystupovat Buttony.

```
        private void ObrazSikmo()
        {
            Image img = Image.FromFile(@"obr1.png");
            Graphics gimg = this.CreateGraphics();

            gimg.DrawImage(img, new Point[] { new Point(50, 5), new Point(280, 150),
new Point(50, 350) });

      }
```

Kód pro vytvoření jednoduché křivky.

```
        private void Krivka()
        {
            Pen pero=new Pen(Color.Blue,3);
            Point[] body= new Point[]{new Point(250,150),new Point(150,50),new
Point(255,35),new Point(25,48) };

            Graphics gkr = CreateGraphics();
            gkr.DrawCurve(pero, body);
            gkr.Dispose();


        }
```

V případě vytvoření křivky a šikmo vloženého obrázku jsou důležité Pointy a jejich nastavení. Vše je vhodné si vyzkoušet a pohrát si s těmito aplikacemi.

9.4. Triky pro chybové hlášení

Programátor musí program ošetřit tak, aby v případě chyby, kterou udělá uživatel, byl upozorněn a měl možnost chybu opravit, jedná se o chyby, jako překlepnutí kdy místo číslice vloží obsluha písmeno. Pokud by došlo k takovéto chybě, program upozorní na chybu. Upozornění se vytváří kombinací příkazů **try – catch**, tak jak je ukázáno na příkladu, vkládat se mají číslice, které jsou nazačátku programu deklarovány, vloží-li uživatel číslo systém si ověří že je to číslo a pokračuje na další příkazy za příkazem **try**, pokud ne příkaz **catch** zobrazí chybu hlášením

```
try
            {  //ano je to číslo vypočítej
                vzdm = Double.Parse(txbvzd.Text);
                vzdalen = vzdm / 1000;
                lbvzdl.Text = "" + vzdalen+" km";
                Vypocet_FresZony();
            }
            catch
            {
                MessageBox.Show("Zadejte hodnoty v metrech");
            }
```

Další z triků je vyhledání znaku v řetězci a také jeho převod do číselné podoby, číselnou hodnotu, je možné najít v ASCII tabulce, kterou je možné získat na internetu.

První možnost jak načíst znak ze souboru je pomocí příkazu **foreach.** Program postupně prochází znaky a hledá jeden, který uživatel potřebuje.

```
private void NajdiZnak(string znak)
        {

            foreach (int w in znak)
```

```
rtkr.Text += w + "";
```

Další možností je načítání přes cyklus while a opět dojde k porovnání znaku. Zde je vytvořen kód pro šifrování znaků, šifru je možné vytvořit přičtením čísla, zvoleného programátorem k číslu znaku, který je dán ASCII abecedou. Např. *a* má charakteristcký číselný znak 97 a k této hodnotě se připočte hodnota pevně voleného klíče (23) a klíče zvoleného uživatelem(432). Tedy pak vyjde číselný znak, který se odešle, zde tedy je nutné číst znak po znaku.

```
private void SifraKod()
        {
           char mezi = ',';
            int p = 0;
            string retez1 = " ";
            string retez2;
         string Text = rtb.Text;
            int Pocetznaku = Text.Length;
            lbpocet.Text = "" + Pocetznaku;
            while (p < Pocetznaku)
            {
                char znak = Text[p];
                int cislo = (int)znak;
                cislo = cislo + sifrklic + kod_klic;
                string hodn = cislo.ToString();
                retez2 = hodn + mezi+"" ;
                retez1 = retez1 + retez2;
                rtkr.Text = retez1;
                p = p + 1;
            }
        }
```

```
 private void Klic()
        {
            int pk = 0;
            int znklice;
            int vys_kod = 0;
            // int klickod = Int32.Parse( txbklic.Text);
            string klickod = txbklic.Text;
            //znklice = txbklic.Text.Length;
            znklice = klickod.Length;
            lbpocet.Text = "" + klickod;
            while (pk < znklice)
            {
                char kod = klickod[pk];
                char kodes = kod;
                vys_kod = vys_kod + (kodes + kod);

                lbkod.Text = "" + vys_kod;
                pk = pk + 1;
            }
            kod_klic = vys_kod;
    }
private void Desifruj()
        {
            int po_des = 0;
            int nacti;
            char mezera = ' ';
```

```
char carka = ',';
string retez22;
// string num_ret;
string ret1 = rtbdes.Text;  //vkladam puvodni text

//string ret2 = txbsif.Text;   //nacti sifrovany
string ret2_2 = rtkr.Text;
nacti = ret2_2.Length;
while (po_des != nacti)
{
    char puv_znak = ret2_2[po_des];

    if (puv_znak != mezera)
    {
        retez22 = puv_znak.ToString();
        txbsif.Text += retez22;


  if (Char.IsDigit(puv_znak))
        {

            num_ret = txbsif.Text;
        }
        else
        {

            int cislo = int.Parse(num_ret);
            cislo = cislo - sifrklic - kod_klic;
```

```
                    string desf_txt = cislo.ToString();
                    char znak = (char)cislo;
                    ret1 = ret1 + znak;
                    rtbdes.Text = ret1;
                    txbsif.Text = "";
                }
            }

            else
            {
                MessageBox.Show("Desifruji");
                rtbdes.Text = "";
            }

            po_des = po_des + 1;
        }
    }
//*************************************************
```

Použitá literatura:

Miroslav Virius: od C++ k C# ,1.vydání, KOOP České Budějovice, 2002
Eric Gunnerson: Začínáme programovat v C# , 1. vydání, Computer Press, 2001
John Sharp, Jon Jagger: Microsoft Visual C# .NET krok za krokem, 1. vydání, Knihy.iDnes
Miroslav Virius: C# Hotová řešení, 1. vydání, Computer Press, 2006
Amadeo Mareš: 1001 tipů a triků pro C#