Číslo a název šablony	III/2 Inovace a zkvalitnění výuky prostřednictvím ICT
Číslo didaktického materiálu	EU-OPVK-VT-III/2-ŠR-317
Druh didaktického materiálu	DUM
Autor	RNDr. Václava Šrůtková
Jazyk	čeština
Téma sady didaktických materiálů	Programování v C# v příkladech III
Téma didaktického materiálu	Dědičnost
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	Objekty, metody, třídy, konstruktory, dědičnost, bázová třída, vlastnosti
Anotace	Studenti se seznamují s děděním proměnných a programují systém pro práci s jednoduchými geometrickými objekty
Použité zdroje	ELLER, Frank. *C# - začínáme programovat: podrobný průvodce začínajícího uživatele*. 1. vyd. Praha: Grada, 2002, 240 s. ISBN 80-247-0324-6. OCHRANOVÁ, Renata a Michal KOZUBEK. *Objektově orientované programování v Turbo Pascalu*. 1. vyd. Brno: Masarykova univerzita, 1993, 117 s. ISBN 80-210-0659-5. 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) 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. 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.

317. Dědičnost

Zopakujme si naše znalosti o objektech:

Objekt je v C# označení jakékoliv entity – tlačítka, pera, generátoru náhodných čísel…

Objekt se skládá ze složek různých druhů:

Proměnné – základní charakteristiky objektu, které vždy obsahují určitou hodnotu.

Vlastnosti – navenek se s nimi pracuje jako s proměnnými, ale ve skutečnosti jsou implementovány dvojicí metod: get – čtení vlastnosti, set – zápis do vlastnosti. Tak je možné operace přiřazení či dotaz na hodnotu spojit s potřebnou kontrolou, přepočtem apod. (Viz Poznámka 1)

Proměnné většinou programujeme jako soukromé (private) a příslušné vlastnosti jako veřejné (public).

Metody – jsou akce, které objekt umí provádět, vytváříme je jako podprogramy.

Události – představují zprávy, vysílané objektem do okolí, typicky při nějaké změně jeho stavu.

Příklady pro formulář (okno programu):

Proměnné: button1, textBox1…

Vlastnosti: Text, BackColor

Události: Paint, Load

Metody: Close, Refresh

Třída – označuje datový typ (druh) objektu. O jednotlivých objektech tříd hovoříme jako o jejích instancích, například tlačítka jsou instance třídy Button.

V následujícím příkladu navážeme na naše poslední cvičení – budeme pracovat s třídami pravidelných homogenních geometrických těles – krychle a kvádr. Začněme krychlí – základní údaje bude hrana a hustota. Obě budou privátní proměnné, pro jejich nastavení a čtení použijeme stejnojmennou vlastnost.

Zatím budeme definovat metody pro výpočet objemu, hmotnosti a převedení údajů o krychli na informační řetězec. Třídu vybavíme bezparametrickým i parametrickým konstruktorem.

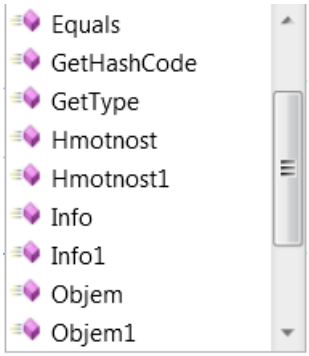
```
class Krychle
    {
        double a; //hrana
        public double A
        {
            get { return a; }
            set { if (value < 0) a = 0; else a = value; }
            //nedovolíme zadat zápornou hodnotu
        }
        double ro; //hustota
        public double Ro
        {
            get { return ro; }
            set { if (value < 0) ro = 0; else ro = value; }
            //nedovolíme zadat zápornou hodnotu
        }
        public Krychle()
        {
            //konstruktor bez parametru
        }
        public Krychle(double a, double ro)
        {
            //parametrický konstruktor
            this.A = a;
            this.Ro = ro;
        }
       public double Objem()
        {
```

```
            return Math.Pow(A, 3);
        }
        public double Hmotnost()
        {
            return Ro * Objem();
        }
public string Info()
        {
            return "Krychle a = " + A.ToString() + " ro = " + Ro.ToString() +
Environment.NewLine + " V = " + Objem().ToString() + " m = " + Hmotnost().ToString();
        }
```

Třídu otestujeme v programu, kde se po stisknutí tlačítka Krychle zobrazí údaje o ní do textboxVystup. Hustotu a hranu zadá uživatel.

```
private void buttonKrychle_Click(object sender, EventArgs e)
        {
            double hrana = Convert.ToDouble(textBoxA.Text);
            double hustota = Convert.ToDouble(textBoxRo.Text);
            Krychle k = new Krychle(hrana, hustota);
            textBoxVystup.Text += k.Info();
        }
```

Když si promyslíme vlastnosti následující třídy kvádr, zjistíme, že je velmi podobná – budou tu navíc dvě hrany a trochu jiné vzorce. Právě v takových situacích využijeme dědičnost – sdílení kódu mezi podobnými třídami.



Společné vlastnosti můžeme ponechat, případně předefinovat, s metodami je to trochu komplikovanější, což si ukážeme v příštích hodinách.

Všimněte si, že metody u kvádru a dalších potomků pojmenováváme jinak (Objem1, Objem2)

Našeptávač nabízí také metody předka, pokud je použijeme, probíhá výpočet tak, jak je definován u předka. (Například se bude počítat objem krychle, i když budeme mít kvádr s různými hranami.)

Potřebná syntaxe je okomentovaná v kódu.

```
class Kvadr: Krychle
    {
        //Tímto zápisem dáváme najevo, že třída Kvádr bude odvozena ze třídy Krychle,
        //takže hranu a a vlastnost A můžeme zdědit bez úprav, stejně jako ro a Ro.
```

```
        double b;
        public double B
        {
            get { return b; }
            set { if (value < 0) b = 0; else b = value; }

        }
        double c;
        public double C
        {
            get { return c; }
            set { if (value < 0) c = 0; else c = value; }

        }
              public Kvadr()
            : base()
        {
         //bezparametrický konstruktor se dědí od Krychle
        }
        public Kvadr(double a,double b, double c, double ro) :base(a,ro)
        {
         //parametrický konstruktor bude přebírat parametrický konstruktor Krychle s
         //inicializací hrany a a hustoty ro, stranu b doplníme.
            this.B =   b;
            this.C = c;
        }
        // metody pro výpočet objemu, hmotnosti a převod na řetězec zatím přepíšeme.
        public double Objem1()
        {
            return A*B*C;
        }
        public double Hmotnost1()
        {
            return Ro * Objem1();
        }
        public string Info1()
        {
            return "Kvadr a = " + A.ToString("F2") + " m, ro = " + Ro.ToString("F2") +
                " kg/m3, b = " + B.ToString("F2") + " m, c = " + C.ToString("F2") +
Environment.NewLine + " m, V = " + Objem1().ToString("F2")
                    + " m3, m = " + Hmotnost1().ToString("F2") + " kg";
        }
```

```
    }
```

Třídu můžeme opět vyzkoušet:

```
private void buttonKvadr_Click(object sender, EventArgs e)
        {
            double a = Convert.ToDouble(textBoxA.Text);
            double b = Convert.ToDouble(textBoxB.Text);
            double c = Convert.ToDouble(textBoxC.Text);
            double hustota = Convert.ToDouble(textBoxRo.Text);
            Kvadr q = new Kvadr(a,b,c, hustota);
            textBoxVystup.Text += q.Info();
        }
```

Poznámka 1: Co kdybychom chtěli změnit jednotky třeba na palce? Právě tady je výhoda vlastností – stačilo by přidat jednu proměnnou a přepracovat příslušnou vlastnost a celý zbytek programu může zůstat tak jak je.

```
       double aInch;

        public double A

       {

          get { return aInch*2.5; }

           set { if (value < 0) aInch = 0; else aInch = value / 2.5; }

       }
```

Poznámka 2: Konstruktor předka se vždy provádí před konstruktorem potomka. Pokud bázová třída nemá bezparametrický konstruktor nebo pokud ho nepotřebujeme, je nutné se na konstruktor předka odvolat pomocí klíčového slova base.

Důležité

Dědičnost umožňuje sdílet kód mezi třídami, přitom platí:

Odvozená třída přebírá všechny složky bázové třídy (předka), další si může přidat a něco zděděného rovněž předefinovat.

Zápis v definici třídy: class Potomek:Předek

Zápis u konstruktorů:
Bezparametrický se dědí beze změny, u parametrického doplníme, co je navíc.

```
   public Kvadr()
            : base()
    {}
  public Kvadr(double a,double b, double c, double ro) :base(a,ro)
```

```
        {

           this.B =   b;
            this.C = c;
        }
```

Pracovní list

Cvičení

1. Doplňte ve třídě Krychle metody pro výpočet povrchu, tělesové úhlopříčky a rozšiřte adekvátně převedení údajů o krychli na informační řetězec.

2. Doplňte třídu Koule jako potomka Krychle, od které zdědí hustotu, rozměr (u krychle hranu, u koule poloměr) a příslušné metody. Zobrazte informace o 10 koulích o poloměrech 1 až 10 m s hustotou 1 kg/m^3.

3. Doplňte třídu Válec jako potomka Koule, od které zdědí hustotu, poloměr (zde podstavy) a příslušné metody. Zobrazte informace o 10 náhodných válcích (hustotu, poloměr i výšku dodá generator náhodných čísel.

Řešení

1.

```
class Krychle
    {
        double a; //hrana
        public double A
        {
            get { return a; }
            set { if (value < 0) a = 0; else a = value; }
            //nedovolíme zadat zápornou hodnotu
        }
        double ro;//hustota
        public double Ro
        {
            get { return ro; }
            set { if (value < 0) ro = 0; else ro = value; }
            //nedovolíme zadat zápornou hodnotu
        }
        public Krychle()
        {
            //konstruktor bez parametru
        }
        public Krychle(double a, double ro)
        {
            //parametrický konstruktor
            this.A = a;
```

```
            this.Ro = ro;
        }
        public double Povrch()
        {
            return 6 * Math.Pow(A, 2);
        }
        public double Objem()
        {
            return Math.Pow(A, 3);
        }
        public double Hmotnost()
        {
            return Ro * Objem();
        }
        public double TUhlopricka()
        {
            return Math.Sqrt(3)*A;
        }
        public string Info()
        {
            return "Krychle a = " + A.ToString() + " ro = " + Ro.ToString() +
Environment.NewLine + " V = " + Objem().ToString() + " S = " + Povrch().ToString() +
                    " u = " + TUhlopricka().ToString() + " m = " +
Hmotnost().ToString();
        }
    }
```

2.

```
class Koule:Krychle
    {
           public Koule()
            : base()
        {
         //konstruktory se dědí od Krychle
        }
        public Koule(double a, double ro)
            : base(a, ro)
        {
        }
        // metody pro výpočet objemu, hmotnosti a převod na řetězec zatím přepíšeme.
        public double Objem2()
        {
```

```
            return 4/3*Math.PI*Math.Pow(A,3);
        }
        public double Hmotnost2()
        {
            return Ro * Objem2();
        }
        public double Povrch2()
        {
            return 4*Math.PI * Math.Pow(A, 2);
        }
        public string Info2()
        {
            return "Koule r = " + A.ToString("F2") + " m, ro = " + Ro.ToString("F2") +
                " kg/m3, " + Environment.NewLine + "V = " + Objem2().ToString("F2")
                + " m3, m = " + Hmotnost2().ToString("F2") + " kg, "+ "S =
"+Povrch2().ToString("F2")+" m2";
        }
    }

private void buttonKoule_Click(object sender, EventArgs e)
        {
            Koule k = new Koule(1, 1);
            for (int i = 1; i < 11; i++)
            {
                textBoxVystup.Text += k.Info2()+Environment.NewLine;
                k.A++;
            }

        }
3.
class Valec:Koule
    {
        double v; //výška
        public double V
        {
            get { return v; }
            set { if (value < 0) v = 0; else v = value; }
            //nedovolíme zadat zápornou hodnotu
        }
             public Valec()
            : base()
        {
```

```
         //konstruktory se dědí od Krychle
        }
        public Valec(double a, double ro,double v)
            : base(a, ro)
        {
            this.V = v;
        }
        // metody pro výpočet objemu, hmotnosti a převod na řetězec zatím přepíšeme.
        public double Objem3()
        {
            return Math.PI*Math.Pow(A,2)*V;
        }
        public double Hmotnost3()
        {
            return Ro * Objem3();
        }
        public double Povrch3()
        {
            return 2*Math.PI * A*(A + V);
        }
        public string Info3()
        {
            return "Válec r = " + A.ToString("F2") + " m, ro = " + Ro.ToString("F2") +
                " kg/m3, v = " + V.ToString("F2")+" m," + Environment.NewLine + "V = "
+ Objem3().ToString("F2")
                + " m3, m = " + Hmotnost3().ToString("F2") + " kg, "+ "S =
"+Povrch3().ToString("F2")+" m2";
        }
    }
private void buttonValce_Click(object sender, EventArgs e)
        {
            Random nahod = new Random();
            Valec val = new Valec();
            for (int i = 1; i < 11; i++)
            {
                double polomer = nahod.Next(1, 10);
                double vyska = nahod.Next(1, 10);
                double hustota = nahod.Next(1, 10);
                val.A = polomer;
                val.V = vyska;
                val.Ro = hustota;
                textBoxVystup.Text +=val.Info3() + Environment.NewLine;
```

```
    }
}
```