using L7.dedicnost;

namespace L7.dedicnost;

class Program

{

    public abstract class Zvire //base class for all animals

    {

        public string Jmeno { get; set; }

        public string Zvuk { get; set; }

        public Zvire(string jmeno, string zvuk)

        {

            Jmeno = jmeno;

            Zvuk = zvuk;

        }

        public virtual void VydavejZvuk()

        {

            System.Console.WriteLine(Zvuk);

        }

    }

    public class Kocka : Zvire

    {

        public Kocka(string name, string zvuk) : base(name, zvuk)

        {}

        public override void VydavejZvuk()

        {

            base.VydavejZvuk();

            System.Console.WriteLine($"Jsem kočka.");

        }

    }

    public class Pes : Zvire

    {

        public Pes(string name, string zvuk) : base(name, zvuk)

        {}

    }

    static void Main(string[] args)

    {

        //navrhnete vhodnou strukturu trid, ktera umozni v programu (v Mainu) vytvorit seznam zviratek v zoo koutku (List), ktery pak muzeme jednoduse projit cyklem (foreach) a zadat kazdemu z nich prikaz VydavejZvuk

        //pritom kazde zviratko bude vydavat jiny zvuk (vypise na konzoli Haf, haf nebo Mnau, mnau), v ramci cyklu nechci resit, jake konkretni zviratko to je

        //vytvorte alespon 2 ruzna zviratka

        //hint: budete potrebovat vhodnou bazovou tridu a virtual/abstract a override

        Pes pes = new Pes("Hafik", "Haf haf!");

        Kocka kocka = new Kocka("Micka", "Mnau mnau!");

        List<Zvire> zoo = new List<Zvire>();

        zoo.Add(pes);

        zoo.Add(kocka);

        foreach (Zvire zviratko in zoo)

        {

            zviratko.VydavejZvuk();

        }

    }

}

internal class Program

{

    private static void Main(string[] args)

    {   // Vytvořte abstraktní třídu PohadkovaBytost a v ní abstraktní metodu NapisJakTravisVolnyCas

        // Vytvorte tridy Princezna, Rytir, Carodenice, ktere dedi PohadkovouBytost

        // Naimplementujte metodu NapisJakTravisVolnyCas

        // Vytvorte instance od kazde tridy a vypiste informace o trávení volného času.

        PohadkovaBytost ruzenka = new Princezna("Ruzenka");

        //ruzenka.NapisJakTravisVolnyCas();

        PohadkovaBytost borivoj = new Rytir("Borivoj");

        PohadkovaBytost kouzelnice = new Carodejnice("Kouzelnice");

        Umelec umelec = new Umelec();

        List<PohadkovaBytost> pohadkoveBytosti = new List<PohadkovaBytost>() { ruzenka, borivoj, kouzelnice };

        //List<PohadkovaBytost> ruzneBytosti = new List<PohadkovaBytost>() { ruzenka, borivoj, kouzelnice, umelec }; toto nejde protože Umelec nededi z PohadkovaBytost

        foreach (PohadkovaBytost bytost in pohadkoveBytosti)

        {

            bytost.NapisJakTravisVolnyCas();

        }

    }

}-------------------------------------

public abstract class PohadkovaBytost

{

    public string Jmeno { get; }

    protected PohadkovaBytost(string jmeno)

    {

        Jmeno = jmeno;

    }

    public abstract void NapisJakTravisVolnyCas();

}-------------------------------------

public class Carodejnice : PohadkovaBytost

{

    public Carodejnice(string jmeno) : base(jmeno) { }

    public override void NapisJakTravisVolnyCas()

    {

        System.Console.WriteLine($"{Jmeno}, čarodějnice. Vařím lekvary.");

    }

}-------------------------------------

public class Princezna : PohadkovaBytost

{

    public Princezna(string jmeno) : base(jmeno) { }

    public override void NapisJakTravisVolnyCas()

    {

        System.Console.WriteLine($"{Jmeno}, princezna. Zkouším si šaty.");

    }

}

public class Rytir : PohadkovaBytost

{

    public Rytir(string jmeno) : base(jmeno) { }

    public override void NapisJakTravisVolnyCas()

    {

        System.Console.WriteLine($"{Jmeno}, rytíř. Chodím na turnaje.");

    }

}