namespace OOP

{

internal interface IMove

{

void Move();

void Move(int distance);

}

}

namespace OOP

{

internal abstract class Animal : IMove

{

public string Name { get; set; }

public int Age { get; set; }

public abstract void MakeSound();

public virtual void Move()

{

Console.WriteLine("Animals are Moving");

}

public virtual void Move(int distance)

{

Console.WriteLine($"Animals are moving with distance {distance}");

}

}

}

namespace OOP

{

internal class Dog:Animal

{

public string Color { get; set; }

public override void MakeSound()

{

Console.WriteLine("Dogs are Barking");

}

public override void Move()

{

base.Move();

}

public override void Move(int distance)

{

Console.WriteLine($"Dogs are Moving with Distance {distance}");

}

}

}

namespace OOP

{

internal class Cat:Animal

{

public override void MakeSound()

{

Console.WriteLine("Cats are xalling with Meow");

}

}

}

namespace OOP

{

internal class Zoo

{

private List<Animal> animals;

public Zoo()

{

animals = new List<Animal>();

}

public void AddAnimal(Animal animal)

{

animals.Add(animal);

}

public void MakeAllSound()

{

foreach (Animal in animals)

{

Console.WriteLine(animal.Name);

Console.WriteLine(animal.Age);

animal.MakeSound();

Console.WriteLine("-----------------------------");

Console.WriteLine();

}

}

}

}

namespace OOP

{

internal class Program

{

static void Main(string[] args)

{

// Dog

Dog = new Dog();

dog.Name = "Fido";

dog.Age = 1;

dog.Color = "White";

dog.Move();

dog.Move(5);

Console.WriteLine("------------------------------");

Console.WriteLine();

// Cat

Cat = new Cat();

cat.Name = "Boxer";

cat.Age = 2;

cat.Move();

cat.Move(3);

Console.WriteLine("------------------------------");

Console.WriteLine();

// Zoo

Zoo = new Zoo();

zoo.AddAnimal(dog);

zoo.AddAnimal(cat);

zoo.MakeAllSound();

Console.ReadLine();

}

}

}