12214994_SaurabhRana

Title: Animal Sound Simulator with Interactive Input

Description:

You are tasked with creating a program that simulates the sounds made by different animals. The program utilizes a hierarchy of classes to represent various animals and their corresponding sounds. The program is designed to be interactive, allowing the user to continuously enter data for different animals until they choose to quit. Write the solution within the **Program.cs** file.

The program consists of the following classes:

All the classes must be in public.

- **Animal**: The base class representing an animal. It contains a virtual method `MakeSound()` which outputs **"Some generic animal sound"** when called.
- **Dog**: A derived class from Animal representing a dog. It overrides the `MakeSound()` method to output **"Bark"** when called.
- **Cat**: A derived class from Animal representing a cat. It overrides the `MakeSound()` method to output **"Meow"** when called.
- **Program**: The main program class containing the `Main()` method. The program then creates an array of Animal objects based on the user input and calls the `MakeSound()` method for each animal.

Tasks:

1. Implement the `Main()` method to prompt the user to input the number of animals they want to simulate.
2. Ensure that the input is a valid integer greater than zero. If the input is invalid, prompt the user again until a valid input is provided.
3. For each animal, prompt the user to input the type of animal (**animal, dog, or cat**). If the user enters an invalid type, prompt them again until a valid type is provided.
4. After the user has entered all the animals, output the sounds made by each animal stored in the array.
5. Ensure that the program can handle varying numbers and types of animals, and that it provides clear feedback for invalid inputs.

Constraints:

- Ensure error handling for invalid user inputs.

- Use appropriate inheritance and method overriding concepts to achieve the desired behavior.
- Implement the program logic efficiently to handle varying numbers and types of animals.

Input Format

- The first input should be the number of animals.
- The next lines of input should be animals (**animal, dog, cat**).

Output Format

If number of animals > 0 :

- The first line of output should be **"Sounds of the animals in the array:"**
- The next lines of output should be displayed based on input, sounds made by each animal.

If number of animals <= 0 :

- The first line of output should be **"Please enter a valid positive integer."**

The output should be in the format of sample outputs.

Sample Input 1

```
4
animal
dog
cat
dog
```

Sample Output 1

```
Sounds of the animals in the array:
Some generic animal sound
Bark
Meow
Bark
```

Sample Input 2

```
0
```

Sample Output 2

```
Please enter a valid positive integer.
```

Commands to Run the Project:

- `cd dotnetapp`
  Select the dotnet project folder
- `dotnet run`
  To run the application
- `dotnet build`
  To build and check for errors
- `dotnet clean`
  If the same error persists clean the project and build again

---

Note:

The project will not be submitted if **'Submit Project'** is not done at least once.

```
class Animal
{
   public virtual void MakeSound()
   {
      Console.WriteLine("Some generic animal sound");
   }
}
class Cat : Animal
{
   public override void MakeSound()
   {
```

```
        Console.WriteLine("Meow");
    }
}
class Dog : Animal
{
    public override void MakeSound()
    {
        Console.WriteLine("Bark");
    }
}
class Program
{
    static void Main()
    {

        Console.Write("Enter number of animals: ");
        int number = Convert.ToInt32(Console.ReadLine());
        while (number <= 0)
        {
            Console.WriteLine("Please enter a valid positive integer ");
            number = Convert.ToInt32(Console.ReadLine());
        }

        Console.WriteLine("Enter the type of animals: ");
        Animal[] animals = new Animal[number];
```

```
for(int i = 0; i < number; i++)
{
   Animal animal;
   while (true)
   {
      string input = Console.ReadLine().ToLower();
      if (input == "animal")
      {
         animals[i] = new Animal();
         break;
      }
      else if (input == "cat")
      {
         animals[i] = new Cat();
         break;
      }
      else if (input == "dog")
      {
         animals[i] = new Dog();
         break;
      }
      else
      {
         Console.WriteLine("Invalid animal type. Please enter animal, dog, or cat.");
      }
```

```
        }


    }



    Console.WriteLine("Sounds of the animals in the array:");
    foreach (Animal animal in animals)
    {
        animal.MakeSound();
    }


  }
}
```