TRENT UNIVERSITY DURHAM GREATER TORONTO AREA

COIS 2020 H: Data Structures and Algorithms

2024, winter

Assignment 1

Fill out the following table. You must paste **screenshots** (images), NOT text (copy/paste).

YOUR NAME: Dikshith Reddy Macherla (0789055)

Paste a screenshot showing the `position` class here.

```
7 references
public class Position
{
    private double x, y, z;

    2 references
    public Position(double x, double y, double z)
    {
        SetX(x);
        SetY(y);
        SetZ(z);
    }

    2 references
    public double X { get => x; set => SetX(value); }
    2 references
    public double Y { get => y; set => SetY(value); }
    2 references
    public double Z { get => z; set => SetZ(value); }

    2 references
    private void SetX(double value) => x = Clamp(value);
    2 references
    private void SetY(double value) => y = Clamp(value);
    2 references
    private void SetZ(double value) => z = Clamp(value);

    3 references
    private double Clamp(double value) => Math.Max(-15.0, Math.Min(15.0, value));

    1 reference
    public void Move(double dx, double dy, double dz)
    {
        X += dx;
        Y += dy;
        Z += dz;
    }

    0 references
    public override string ToString() => $"Position: ({X}, {Y}, {Z})";
}
```

Paste a screenshot showing the `animal` class here.

```
public class Animal
{
    2 references
    public int ID { get; set; }
    2 references
    public string Name { get; set; }
    2 references
    public double Age { get; set; }
    3 references
    public Position Pos { get; set; }

    2 references
    public Animal(int id, string name, double age, Position pos)
    {
        ID = id;
        Name = name;
        Age = age;
        Pos = pos;
    }

    1 reference
    public void Move(double dx, double dy, double dz) => Pos.Move(dx, dy, dz);

    4 references
    public override string ToString() => $"ID: {ID}, Name: {Name}, Age: {Age}, Position: {Pos}";
}
```

Paste a screenshot showing the `cat` class here.

```
public class Cat : Animal
{
    3 references
    public enum Breed { Abyssinian, BritishShorthair, Bengal, Himalayan, Ocicat, Serval }
    2 references
    public Breed CatBreed { get; set; }

    1 reference
    public Cat(int id, string name, double age, Position pos, Breed breed)
        : base(id, name, age, pos) => CatBreed = breed;

    3 references
    public override string ToString() => $"{base.ToString()}, Breed: {CatBreed}";
}
```

Paste a screenshot showing the `Snake` class here.

```
public class Snake : Animal
{
    2 references
    public double Length { get; set; }
    2 references
    public bool Venomous { get; set; }

    1 reference
    public Snake(int id, string name, double age, Position pos, double length, bool venomous)
        : base(id, name, age, pos)
    {
        Length = length;
        Venomous = venomous;
    }

    3 references
    public override string ToString() => $"{base.ToString()}, Length: {Length}, Venomous: {Venomous}";
}
```

Paste a screenshot showing the `main` method here.

```
static void Main()
{
    // Assuming you have methods to read names from files: ReadCatNames(), ReadSnakeNames()
    List<string> catNames = ReadCatNames(); // Implement this
    List<string> snakeNames = ReadSnakeNames(); // Implement this

    var animals = new List<Animal>();
    var rand = new Random();

    // Generate 3 cats
    for (int i = 0; i < 3; i++)
    {
        animals.Add(new Cat(i, catNames[i], rand.NextDouble() * 15, new Position(rand.NextDouble() * 30 - 15, rand.NextDouble() * 30 - 15, rand.NextDouble() * 30 - 15), (Cat.Breed)rand.Next(6)));
    }

    // Generate 3 snakes
    for (int i = 3; i < 6; i++)
    {
        animals.Add(new Snake(i, snakeNames[i - 3], rand.NextDouble() * 15, new Position(rand.NextDouble() * 30 - 15, rand.NextDouble() * 30 - 15, rand.NextDouble() * 30 - 15), rand.NextDouble() * 10, rand.Next(2) == 0)
    }

    // Use an array or list to store and manage these objects as required
    Animal[] animalArray = animals.ToArray();
    List<Animal> animalList = new List<Animal>(animals);

    // Traverse and print properties
    foreach (var animal in animalArray)
    {
        Console.WriteLine(animal);
    }

    // Move all animals
    foreach (var animal in animalList)
    {
        double dx = rand.NextDouble() * 4 - 2;
        double dy = rand.NextDouble() * 4 - 2;
        double dz = rand.NextDouble() * 4 - 2;
        animal.Move(dx, dy, dz);
    }

    // Print off all objects and positions again
    foreach (var animal in animalList)
    {
        Console.WriteLine(animal);
    }

}
```

Paste a screenshot showing the code that traverses the *array*, along with another screenshot showing the terminal (output) after traversing the array and printing out the information of all `animals`.

```
// Traverse and print properties
foreach (var animal in animalArray)
{
    Console.WriteLine(animal);
}
```

Paste a screenshot showing the code that traverses the *list*, along with another screenshot showing the terminal (output) after traversing the list and printing out the information of all `animals`.

```
// Print off all objects and positions again
foreach (var animal in animalList)
{
    Console.WriteLine(animal);
}
```

Paste a screenshot showing the code that moves all animals in the *array*, along with another screenshot of the terminal (output) showing the animals' information before AND after moving them. [See step 4 under the Main method description]

```
// Move all animals
foreach (var animal in animalList)
{
    double dx = rand.NextDouble() * 4 - 2;
    double dy = rand.NextDouble() * 4 - 2;
    double dz = rand.NextDouble() * 4 - 2;
    animal.Move(dx, dy, dz);
}
```

Paste a screenshot showing the code that moves all animals in the *list*, along with another screenshot of the terminal (output) showing the animals' information before AND after

moving them. [See step 4 under the Main method description]

```
// Move all animals
foreach (var animal in animalList)
{
    double dx = rand.NextDouble() * 4 - 2;
    double dy = rand.NextDouble() * 4 - 2;
    double dz = rand.NextDouble() * 4 - 2;
    animal.Move(dx, dy, dz);
}
```