CS/INFO 1182

Test 1 Practice

Read the instructions carefully and answer each question as completely as you can.

Try doing this test as completely as you can; then you can type the code in to Visual Studio to see if what you expect to happen did.

For the test you will be allowed
One – 3x5 index card

I recommend that this is hand written, but it is not required to be so.

1182 – Pretest 1

Use the following code for your test. Yes, I purposely left out comments. I did not say it was good code. Assume the needed reference calls have been done.

```
namespace PracticeTest {
    interface IAttached {
        int countOfAppendages();
    }
    abstract class NamedThing  {
        private String _Name = "";
        public String Name { get { return _Name; } set { _Name = value; } }
    }
    class Animal : NamedThing {
        private int _LegCount = 0;
        private AmountOfHair _Hairyness = AmountOfHair.None;
        public enum AmountOfHair { None, Little, Some, Lots, Very }
        public Animal() { }
        public Animal(String name) { Name = name; }
        public int LegCount { get { return _LegCount; }
                              set { _LegCount = value; } }
        public AmountOfHair Hairyness { get { return _Hairyness; }
                              set { _Hairyness = value; } }
        public virtual String Sound() { return "Grunt"; }
    }
    class Ape : Animal, IAttached {
        private int _ArmCount = 0;
        public Ape() { }
        public Ape(String name) {
            Name = name; LegCount = 2; ArmCount = 2;
            Hairyness = AmountOfHair.Little;
        }
        public int ArmCount { get { return _ArmCount; }
                              set { _ArmCount = value; } }
        public int countOfAppendages() { return ArmCount + LegCount; }
    }
    class Cat : Animal {
        public Cat() { }
        public Cat(String name) { Name = name; LegCount = 4;
                              Hairyness = AmountOfHair.Some; }
        public override String Sound() { return "Meow"; }
        public override String ToString() {
            return String.Format("{0} {2}", Name, Sound(), base.Sound());
        }
    }
}
```

Assuming all of the following code runs in the order presented and using the following code in conjunction with the code on the first page, assuming each line of code happens in the order presented.

```
NamedThing[] things = new NamedThing[3];
Cat felix = new Cat("Felix");
things[1] = felix;
felix.Hairyness = Animal.AmountOfHair.Lots;
Ape george = new Ape("George");
things[0] = george;
Animal ani = new Cat();
ani.Name = "Ralph";
things[2] = ani;
```

1. What will the following code print to the console?
   ```
   Console.Out.Write(felix);
   ```

2. What will the following code print to the console?
   ```
   Console.Out.Write(george.Name);
   ```

3. What will the following code print to the console?
   ```
   Console.Out.Write(felix.Name);
   ```

4. What will the following code print to the console?
   ```
   Console.Out.Write(felix.Sound());
   ```

5. What will the following code print to the console?
   ```
   foreach (NamedThing nt in things)
        Console.Out.Write(nt.Name + ", ");
   ```

6. What will the following code print to the console?
```
Console.Out.Write(things[2].Name + " | " + ani.Name);
```

7. What will the following code print to the console?
```
Console.Out.Write(felix.LegCount);
```

8. What will the following code print to the console?
```
Console.Out.Write(george.countOfAppendages());
```

9. What will the following code print to the console?
```
Console.Out.Write(george.ArmCount + ani.LegCount - felix.LegCount);
```

10. What will the following code print to the console?
```
Console.Out.Write(george.Hairyness);
```

11. What will the following code print to the console?
```
Console.Out.Write(ani.Hairyness);
```

12. What will the following code print to the console?
```
Console.Out.Write(new Cat("Max").Name[2]);
```

Use the following code in conjunction with the code on the first page, assuming it exists before each statement in questions 13-16:

```
Cat cat1 = new Cat("Tom");
Cat cat2 = new Cat("Garfield");
Cat cat3 = new Cat("Heathcliff");
Cat cat4 = new Cat("Fiji");

cat1.LegCount = 3;
cat2.LegCount = 4;
cat2 = cat1;
cat3.LegCount = cat2.LegCount - 2;
cat4.LegCount = 4;

cat4 = cat3;
cat3.Hairyness = Animal.AmountOfHair.Lots;
cat1.LegCount = cat3.LegCount;
```

13. What will the following code print to the console?

```
Console.Out.Write(String.Format("Name: {1}; LegCount: {0}"
                          , cat1.LegCount, cat1.Name));
```

14. What will the following code print to the console?

```
Console.Out.Write(String.Format("Name: {1}; LegCount: {0}"
                          , cat4.LegCount, cat4.Name));
```

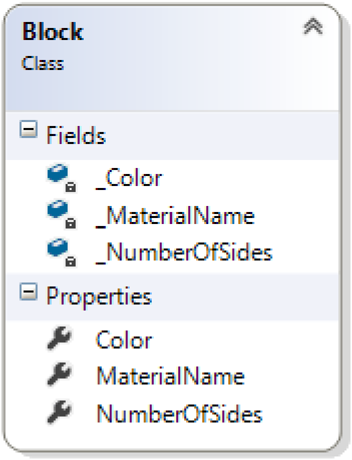
15. What will the following code print to the console?

```
Console.Out.Write(String.Format("Name: {1}; LegCount: {0}"
                          , cat3.LegCount, cat3.Name));
```

16. What will the following code print to the console?

```
Console.Out.Write(String.Format("Name: {1}; LegCount: {0}"
                          ,cat2.LegCount,cat2.Name));
```

17. Using the following UML class diagram, write a C# class.
    If you do not want to use Properties, you may use accessors and mutators.


Block
Class
Fields
_Color
_MaterialName
_NumberOfSides
Properties
Color
MaterialName
NumberOfSides