1. [15%] Short answer:

1.1 Give 2 benefits of using an interface over an abstract class:

1.2 Two benefits of using an abstract class over of an interface:

1.3 If given

```java
public interface I { public int m(); }
public class A implements I { public int m() { return 1; } }
public class A1 extends A { public int m() { return 2; } }
```

I i = new A1();

What does i.m() return? and why (explain the rule)?

Give code that would give access to the version of m() in class A
2. [15%] Regarding recursion and iteration.

2.1 Define and distinguish recursive method versus recursive data structure. Give a specific example of each.

2.2 Describe the benefits of using an Iterator on a List class.

2.3 Define and give a specific example of a wrapper class (they were useful in several patterns).
3. [70%] Consider the design domain of a **dog**, which watches his **ball**. When a **person** goes over to the ball and **throws** it, the (initially **lazy**) dog gets **excited**, and **chases** the ball.

Use that domain to help answer the following. First decide what methods and responsibilities comprise this domain. Plan on using the **State**, **Visitor**, and **Observer** patterns. Provide a concise but complete description of the pattern, including in your answer:

a) The prototypical **design goals** for which this pattern is intended.

b) The **benefits and limitations** of this design pattern.

c) Its structural components (use a **UML class diagram**).

d) Provide the **driver code** for the above.

e) Describe in text, code fragments, and a **UML sequence diagram** how each of the three patterns work (what methods in what classes call what other methods in other classes, and in what order).

You may combine your answers to parts c-e as appropriate (but indicate which parts relate to which patterns, perhaps by circling them in the UML class diagrams). Provide sufficient code to make your design clear, but you need not code the entire domain (but in fact that is not a very large task).