Name _____

Answer 9 questions. Indicate which 9 questions you are answering. Do **not** answer more than 9 questions. Each question is worth 10 points.

The following might be names of patterns: Abstract Class, Abstract Factory, Adapter, Bridge, Builder, Chain of Responsibility, Collaborator, Command, Command Processor, Composite, Decorator, Façade, Factory Method, Flyweight, Interpreter, Iterator, Mediator, Memento, Null Object, Observer, Prototype, Proxy, Singleton, Specification, State, Strategy, Template Method, Visitor.

1. In the last assignment you created a sequence or chain of objects to match a pattern in a string. Was this an example of chain of responsibility or object-oriented recursion? Justify your answer.

2a. What is a robust iterator?

b. What is a fail-safe iterator?

3. Explain the benefits and disadvantages of using the Null object pattern in implementing a linked list.

4. In assignments you have implemented code to find the strings in a linked list that start with a vowel. Most of you have done this three different ways: directly in the linked list, in a visitor and using a filter iterator. Compare and contrast two of these methods in terms of coupling and cohesion.

5. Give two difficulties that can occur in implementing the adaptor pattern.

6. What is the difference between internal coupling and cohesion? Provide a concrete example.

7. Explain one of the following types of coupling: Data Coupling, Control Coupling, Inside Internal Object Coupling

8. Explain one of the following types of cohesion: Logical, Temporal, Procedural, Communication, Sequential. Give an example.

9. Explain how the observer pattern reduces tight coupling.

10. Both the Memento and Command patterns can be used to implement undo. What are the disadvantages and advantages of each pattern for implementing undo.

11. (Smalltalk version) The following methods are from classes A & B that have a common superclass C. Modify (that is refactor) the methods using Template method and Factory Method. Show all the methods you would create and which classes they would be in. (Java/C++/C# version on next page. Do at most one version of problem 11)

```
Class A
send: aRequest to: anAddress
    | sender result |
    self log: 'send in class A called'.
    sender := HttpClient new.
    sender address: anAddress.
    result := sender process: aRequest.
    result is404 ifTrue: [self log: 'Error in sending message']

Class B
send: aRequest to: anAddress
    | sender result |
    self log: 'send in class B called'.
    sender := FtpClient new.
    sender address: anAddress.
    result := sender process: aRequest.
    result isEmpty ifTrue: [self log: 'Error in sending message']
```

11. (Java/C++/C# Version) The following methods are from classes A & B that have a common superclass C. Modify (that is refactor) the methods using Template method and Factory Method. Show all the methods you would create and which classes they would be in. (Do at most one version of problem 11)

```
Class A
send( Request aRequest, URL anAddress) {
    log( "send in class A called");
    NetClient sender = new HttpClient().
    sender address(anAddress).
    Repsonse result = sender process( aRequest).
    if (result is404()) {log("Error in sending message")};
}

Class B
send( Request aRequest, URL anAddress) {
    log( "send in class B called");
    NetClient sender = new FtpClient ().
    sender address(anAddress).
    Repsonse result = sender process( aRequest).
    if (result isEmpty ()) {log("Error in sending message")};
}
```

12. Select one of the patterns Strategy or State and discuss the consequences (good and bad) of the pattern.

13. Select one of the patterns Visitor or Composite and explain how the pattern works.

14. The design patterns text uses two main design principles. Select one pattern that uses both principles. Show how it uses both.

In the follow 2 problems two patterns are listed. The paired patterns are similar in some fashion. Explain how the patterns are different. You should cover when each pattern would be used over the other, the differences in intent of the patterns and any structural differences.

16. Prototype verses Factory Method

17. Decorator verses the Chain of Responsibility