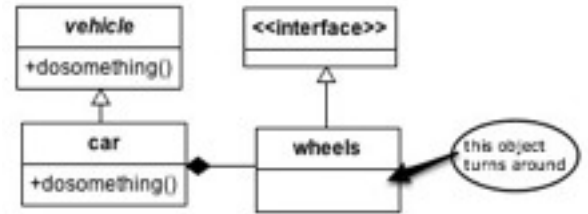


CS330 Design Patterns - Midterm 2

Please read all instructions carefully. The exam is closed book & no laptops / phones / computers shall be present nor be used. Please write your answers in the space provided. Solutions will be graded on correctness and clarity. You can score a total of **100** points on this exam. For every question you **may** use words, class diagrams or java code to illustrate your answer, unless the question asks for a specific format. When you draw a class diagram, indicate the **methods** that make the pattern work also specify the type of **relation** (inheritance / composition) between classes and **type** of class (interface / abstract class / regular) as indicated in the sample UML class diagram on the right. You may also use the terms "is-a" or "has-a" to indicate the relationships. For each class & interface in your diagram indicate what it does. Use a circle to describe the **responsibilities** of each class.



1: (5 points) Assign each design pattern to the right category by placing an 'X' at the right spot. Each pattern belongs to one category.

pattern	structural	creational	behavioral
template method			X
composite	X		
proxy	X		
iterator			X
facade	X		
state			X
adapter	X		

2. (5 points) How is a virtual **proxy** different from a remote **proxy**?

A virtual proxy regulates access to an object that is expensive to create whereas a remote proxy regulates access to an object that is located in a different name space. See page 462 of the book.

CS330 Design Patterns - Midterm 2

3. (5 points) Explain how a class **adapter** and an object **adapter** are different from each other. Draw class diagrams if necessary.

An object adapter uses composition to adapt the adaptee, whereas a class adapter uses inheritance to subclass the adaptee and the target classes See page 243/244

4. (5 points) Explain how the similarities and differences between the **decorator** pattern and the **proxy** pattern. Draw class diagrams if necessary.

Both patterns add functionality to an object, however, the decorator pattern adds behavior to an object while a proxy is more specialized as it controls access.

CS330 Design Patterns - Midterm 2

5. (5 points) What does coupling mean? and how does the **Facade** pattern reduce coupling? Draw class diagrams if necessary.

Coupling is the degree to which each class relies on other classes. The facade provides one uniform interface for a set of complex subsystems (page 264). It reduces coupling as instead of having the client call the numerous interfaces of each subsystem, it calls the interface of the face.

6. (5 points) Explain how the **composite** pattern realizes "Program to interfaces, not to implementations" design principle.

The abstract "Component" class of the abstract layer of the pattern is an interface of the composite structure for the client to access.

7. (5 points) What does cohesion mean? provide an example of a class with low cohesion (you don't have to give implementation details for the methods). What software quality does cohesion affect and how?

Cohesion is a measure of how closely a class supports a single purpose or responsibility. A class has high cohesion when it is design around a a set of related functions, it has low cohesion when the functions are unrelated. Classes with high cohesion are more maintainable. Page 340 lists example classes that have a high cohesion.

CS330 Design Patterns - Midterm 2

8. (5 points) Explain how the **Template Method** pattern works. Draw a class diagram or use source code to illustrate how this pattern works. Explain the purpose of a hook.

The template method pattern defines the skeleton of an algorithm in a method, deferring some steps to subclasses. A hook is a method that is declared in the abstract class, but only given an empty or default implementation. This gives subclasses the ability to hook into the algorithm at various points or ignore it.

A class diagram can be found on Page 289 of the book.

9. (5 points) Mix and match patterns with their object oriented design principle(s). Some patterns may not be connected to a principle.

Pattern		OO principle
template method	→	Only talk to your friends
proxy	→	Favor composition over inheritance
iterator	→	Don't call us, we'll call you
facade	→	A class should have only one reason to change
state	→	

CS330 Design Patterns - Midterm 2

10. (10 points). Given the following Stack class and Queue Interface:

```
public class PegAdapter extends RoundPeg implements ISquarePeg {
    private RoundPeg roundPeg;
    private SquarePeg squarePeg;

    public PegAdapter(RoundPeg peg) {
        this.roundPeg = peg;
    }

    public PegAdapter(SquarePeg peg) {
        this.squarePeg = peg;
    }

    public void insertIntoSquare(String str) {
        roundPeg.insertIntoHole(str);
    }

    public boolean hasPeg() {
        return roundPeg.hasRoundPeg();
    }

    public void insertIntoHole(String str) {
        squarePeg.insertIntoSquare(str);
    }

    public boolean hasRoundPeg() {
        return squarePeg.hasPeg();
    }
}
```

CS330 Design Patterns - Midterm 2

Design Problems

A number of design problems are listed below. Select the most appropriate design pattern to use to address the problem and show how it is applied. In particular, show an appropriate class diagram and enough code fragments to illustrate your use of the pattern to solve the problem.

9. **(15 points)** To improve web page download times, a designer thinks it would be a good idea to have the browser pre-fetch all of the pages referenced from the page the user is currently viewing. But rather than have each individual client pre-fetch the pages from remote web servers, it would be even better if all the pre-fetched pages for a group of clients could be made available locally. Select the most appropriate design pattern to use to address the problem and show how it is applied. In particular, show an appropriate class diagram(s) and enough code fragments to illustrate your use of the pattern to solve the problem.

The solution here is to use the caching proxy pattern.

10. **(15 points)**. We are writing the software for a new electronic hand dryer. The dryer has a light sensor which can determine whenever someone's hands are placed under it. This triggers the dryer to start blowing hot air. The dryer can also be triggered via a push button on the unit. Once the unit starts blowing, we do not want additional pushes of the button or triggers of the light sensor to extend the air blowing cycle.

The solution here is to use the state pattern.

CS330 Design Patterns - Midterm 2

11. (15 points). Your knowledge of design patterns has landed you that dream job at Google. Your goal is to implement a new version of the GoogleBot. Googlebot is a tool that recursively harvests information from web pages to build a searchable index for the [Google search](#) engine. Google bot analyzes websites for tags that are embedded in the various different elements that constitute a webpage. For example `<meta name="description" content="photos of Britney spears"/>` or `` are some of the tags that Google bot seeks to harvest.

The solution here is to use the iterator or the composite pattern, so the search engine can iterate over the elements of a webpage to harvest tags.

