

CS4233-A04 Final Exam

Name: _____

This exam consists of two parts. If you have read the textbook, attended the lectures, done the homework, and studied the notes you are prepared.

Remember that there is a code of academic honesty at WPI and you are expected to do your own work. Good luck.

Part I: General knowledge (60 points)

This questions in this section are fill-in-the-blank, short answers, and so on.

1. What is meant by *stability* with respect to classes?
One class is more stable than another if it has fewer dependencies upon other classes. More exactly, if you compute the ration of the number of other classes that depend upon it with the total number of dependencies, the class with a ratio closer to zero is more stable.
2. Match the pattern with the description. The patterns are:
 - a. Strategy
 - b. Template
 - c. Façade
 - d. Decorator
 - e. Composite

The descriptions are the following. Place the letter of the pattern after the description in the blank provided.

1. _____ This pattern simplifies the use of several classes that make up a subsystem. It provides a single, stable interface to the subsystem.
2. _____ Allows you to treat different type of objects, (i.e., primitive and complex) as a single type and lets you combine them easily.
3. _____ Use this pattern when you have different types of algorithms to use and want to select the proper one based upon the dynamic state of the system.
4. _____ This pattern lets you enhance the behavior of another class without the other class knowing anything about the enhancement.
5. _____ This class lets you implement the step, or flow, of an algorithm without specifying the actual types that are involved. You break the algorithm into primitive steps and let them be determined by the users of the pattern.

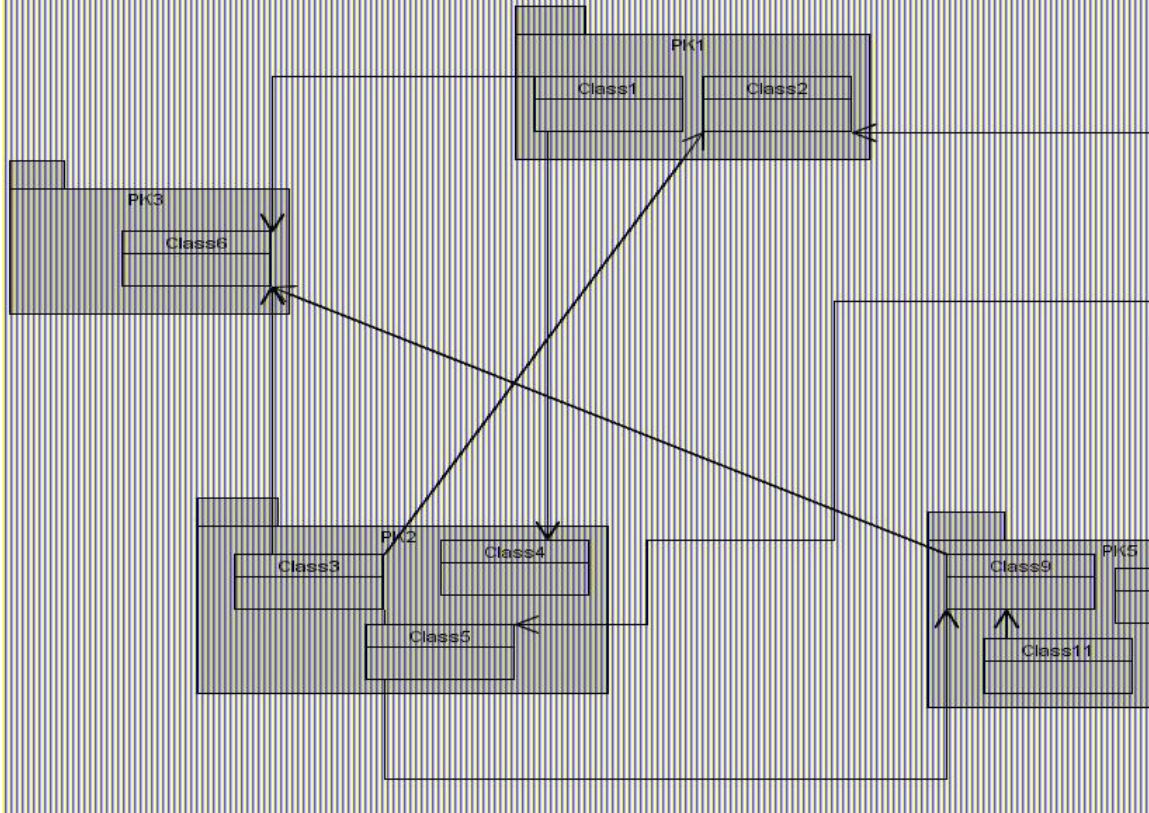
Answers: 1-c, 2-e, 3-a, 4-d, 5-b

CS4233-A04 Final Exam

3. What is the Stable Dependencies Principle (SDP)?
You should depend in the direction of stability. That is, any class should depend on classes that are more stable than it is.
4. In Java, what operator do you use to perform type inquiry? `instanceof`
5. You have just purchased a library to implement direct brainwave translation from a device that gets implanted in your brain. You think this would be a great addition to your MQP where you're implementing a human-intelligence program. But, your program is already written to make use of a much less effective intelligence engine that plugs into your kneecap – different methods, different API. But, the basic functionality is the same. What pattern will help you get out of this predicament and win the No-bell Prize for raising the human IQ?
Adapter
6. What are the three primary characteristics of Java Beans?
It has properties, methods, and events.
7. What is a framework? Give at least two examples of them.
A framework is a set of classes that work together for a specific purpose, almost always in a particular domain. Examples may be things like: Java Swing, Java AWT, a set of Math objects, many of the Eclipse plug-ins, etc.
8. What is the relationship between Java Beans and reflection?
Tools that manipulate Java Beans use reflection to interrogate the bean, modify it, generate code for recognizing events, etc.
9. When would you use an abstract class instead of an interface?
You would use an abstract class in favor of an interface when you have some concrete behavior that is shared among all subclasses and will almost never need to be changed.
10. What is serialization? What do you use it for?
Serialization is a way to take classes from memory and store them in some persistent storage. There are several reasons for doing this, including a simple database that will keep the state of the objects from one session to another.

Part II (40 points)

1. In the following diagram, compute C_e , C_a , and I for each of the packages and write the values above each package.



CS4233-A04 Final Exam

2. Write a small main method that actually runs the following in a separate thread.

```
/**
 * An action that repeatedly prints a greeting.
 */
public class GreetingProducer implements Runnable
{
    /**
     * Constructs the producer object.
     * @param aGreeting the greeting to display
     */
    public GreetingProducer(String aGreeting)
    {
        greeting = aGreeting;
    }

    public void run()
    {
        try
        {
            for (int i = 1; i <= REPETITIONS; i++)
            {
                System.out.println(i + ": " + greeting);
                Thread.sleep(DELAY);
            }
        }
        catch (InterruptedException exception)
        {
        }
    }

    private String greeting;

    private static final int REPETITIONS = 10;
    private static final int DELAY = 100;
}
```