

Quiz 2

1.

Which of the following statements is **FALSE** in relation to generics?

<input type="radio"/>	A generic class used without type arguments is known as a <i>raw type</i>
<input type="radio"/>	You cannot instantiate an array of a generic type using new operator e.g., <code>T[] anArray = new T[100]</code>
<input type="radio"/>	Consider the following method, <code>someMethod(Box<Number> n) { /*.... */ }</code> , this method can take in a <code>Box<Integer></code> or <code>Box<Double></code>

2.

Consider the following code:

```
public interface xyz {  
    void abc() throws IOException;  
}  
public interface pqr {  
    void abc() throws FileNotFoundException;  
}  
public class Implementation implements xyz, pqr {  
    // insert code  
    { /*implementation*/ }  
}
```

Which of the following statement(s) can you insert in place of “// insert code” comment?

<input type="radio"/>	<code>public void abc() throws FileNotFoundException, IOException</code>
<input type="radio"/>	<code>public void abc() throws IOException</code>
<input type="radio"/>	<code>public void abc() throws FileNotFoundException</code>

3.

A design pattern used to enhance a component’s functionality dynamically at run-time is:

<input type="radio"/>	Composite Pattern
-----------------------	-------------------

<input type="radio"/>	Decorator Pattern
<input type="radio"/>	Abstract Factory Pattern
<input type="radio"/>	Observer Pattern

4.

- Which Design Pattern should you use when....
 - there is a language to interpret, and you can represent statements in the language as abstract syntax trees.

<input type="radio"/>	Singleton
<input type="radio"/>	State
<input type="radio"/>	Composite
<input type="radio"/>	Factory

5.

Identify the pattern used in this code:

```
LineNumberReader lnr = new LineNumberReader(
    new BufferedReader(
        new FileReader("./test.c")));
String str = null;
while((str = lnr.readLine()) != null)
    System.out.println(lnr.getLineNumber() + ": " + str);
```

<input type="radio"/>	Strategy
<input type="radio"/>	State
<input type="radio"/>	Factory
<input type="radio"/>	Decorator

6.

Which of the following exceptions must be handled by a try-catch block or declared?

<input type="radio"/>	NullPointerException
<input type="radio"/>	MalformedURLException
<input type="radio"/>	ClassCastException
<input type="radio"/>	ArithmeticException

7.

Which of the following statements is NOT true?

<input type="radio"/>	The Builder Pattern is a violation of the law of demeter
<input type="radio"/>	Decorators provide a flexible alternative to inheritance for extending functionality.
<input type="radio"/>	The observer pattern provides a design where subjects and observers are loosely coupled
<input type="radio"/>	The Factory Method Design Pattern uses inheritance to solve the problem of creating objects without specifying their exact object classes

✓ Submit