QUIZ 1
(FALL 2014)

SOEN 343 – Software Design and Architecture

Instructor: Dr. Peter C Rigby

Department of Computer Science and Software Engineering
Concordia University

October 14, 2014
Duration: 45 minutes
The quiz is worth 10% of your grade and is out of 21 marks.

Instructions:

- Closed book and no calculators allowed.
- Clearly state all assumptions.
- Handwriting must be legible. You can use pencil.
- All answers must be written in the **answer book** and not on the question sheet

Part I: Multiple Choice and Fill-in-the-Blanks Questions [4 marks]

Note: Answer all questions in the exam **booklet!**

Question 1: (not covered yet) Name the two classes and interface involved in the observer pattern.

_publisher, producer, subject__ ___subscriber__ __listener, observer_

Question 2: (not covered yet)

SHA1 hash codes are used to __uniquely_ __identify__ commits in git.

Question 3: Which of the following **is** a valid implementation of the class diagram below?

B

a)
```
class X {
   private int b;
}

class Y {
   private int c;
}
```

b)
```
class X {
   private int b;
   private Y a;
}

class Y {
   private int c;
}
```

c)
```
class X {
   private int b;
}

class Y {
   private int c;
   private Y a;
}
```

d)
```
class X {
   private int b;
   protected Y a;
}

class Y {
   private int c;
}
```

Question 4: Assume that the following code for `A` and `B` is valid. What would be the value of `x.get()` if `x` were declared and initialized as follows: `A x = new B();`
(B)

```
class A {
  public String c;

  public A() {
        c = "A";
  }

  public String get() {
    return "A" + c;
  }
}
```

```
class B extends A {
  public B() {
          super();
          c="B";
  }


  public String get() {
    return "B" + c;
  }
}
```

a) AA
b) BB
c) AB
d) BA

Part II: Short Answer [7 marks]

Note: Use the number of marks to guide how much you should write!

Question 1: Summarize Gall's law. What is the implication of Gall's law? What is another way of stating Galls law? **[2 marks]**

-Successful complex systems evolve from a simple system. (1mark)

- A complex system designed from scratch cannot be made to work (.5 marks)

(.5) for any of the following

-write for re-write instead of general reuse

-YAGNI (you aren't going to need it)

-other agile type sayings

Question 2: **(not covered yet)** Provide one advantage and one disadvantage to "working off a named stable base" or "a stable release" **[2 marks]**

- Isolated from unrelated changes
- If you don't merge often enough you can end in merge hell.

Question 3: Mobile Networks [5 marks]

a) Define and contrast bandwidth and latency. **[2 marks]**

Bandwidth: rate of transmitting data (b/sec)
Latency: time delay between sending a message and receiving a response (seconds)

b) Why is a low initial congestion window worse for cellphones than on a wired network? **[3 marks]**
The congestion window is the number of outstanding packets that are allowed until the client/server is no longer allowed to send new data. By default it is initially set to 3. (1 mark)

Cell phones have higher latency than wired networks. On a wired network the number is set low to avoid congestion and grows quickly if the network is uncongested. (1mark)

With cell networks, the latency is so high the client may be waiting for acknowledgements even thought there is no congestion. As a result, high latency and a small initial congestion window can slow reduce the throughput of a connection. Google recommends a congestion window of 10 instead of 3. (1mark)

Part III: Design Question [10 marks]

Question 1: GRASP [10 marks]

Consider the given excerpt of a `Player` class from a simple **board game**. The given code shows the fields and method needed to adjust the amount of money in the player's purse after having landed on a particular square of the board.

```
public class Player {
   private String squareKind;
   ...
   private int purse;

   public void spendOrRecieveMoney() {
      if (squareKind.equals("Treasure")) {
         purse += purse * 100;
      } else if (squareKind.equals("Market")) {
         if (purse > 100)
            purse -= 0.20 * purse;
      } else if (squareKind.equals("Normal")) {
         purse += 1;
      } else {
         throw new IllegalStateException("unknown kind");
      }
   }
   ...
}
```

a) Has the principle of cohesion or coupling been violated in this above code? Explain. [3 marks]

Cohesion: multiple responsibilities are in the same class. The player shouldn't know about the logic behind each square.

b) Name the design pattern that you would use to improve the design [1 mark]

Strategy pattern

c) Provide a ***detailed*** class diagram illustrating your solution (show all type, method and constructor parameters, visibility, static/nonstatic, etc.). [6 marks]

See in class notes