quiz no solution w3

Question 1

What does MVC Stand for? Use spaces between each word, no upper case letters, and no punctuation.

[model view controller]

Correct

Correct! The model view controller pattern is important for user-interface applications, and it previews some software architectures that we will talk about in the next course.

Question 2

Select the **two** elements of the open/closed principle:

0 / 1 point

- ☐ **Open for modification**
- ☐ **Open for extension**
- ☐ **Open for maintenance**
- ☐ **Closed for modification**
- ☐ **Closed for extension.**
- ☐ **Closed for maintenance.**

Question 3

What is the best description of the Dependency Inversion principle?

0 / 1 point

- ☐ **Client objects are dependent on a service interface that directs their requests.**

- [ ] **Service objects subscribe to their prospective client objects as Observers, watching for a request.**
- [ ] **Client objects depend on an Adaptor Pattern to interface with the rest of the system.**
- [ ] **Client objects depend on generalizations instead of concrete objects.**

Question 4

Which of these statements is true about the Composing Objects principle?

1. it provides behaviour with aggregation instead of inheritance
2. it leads to tighter coupling

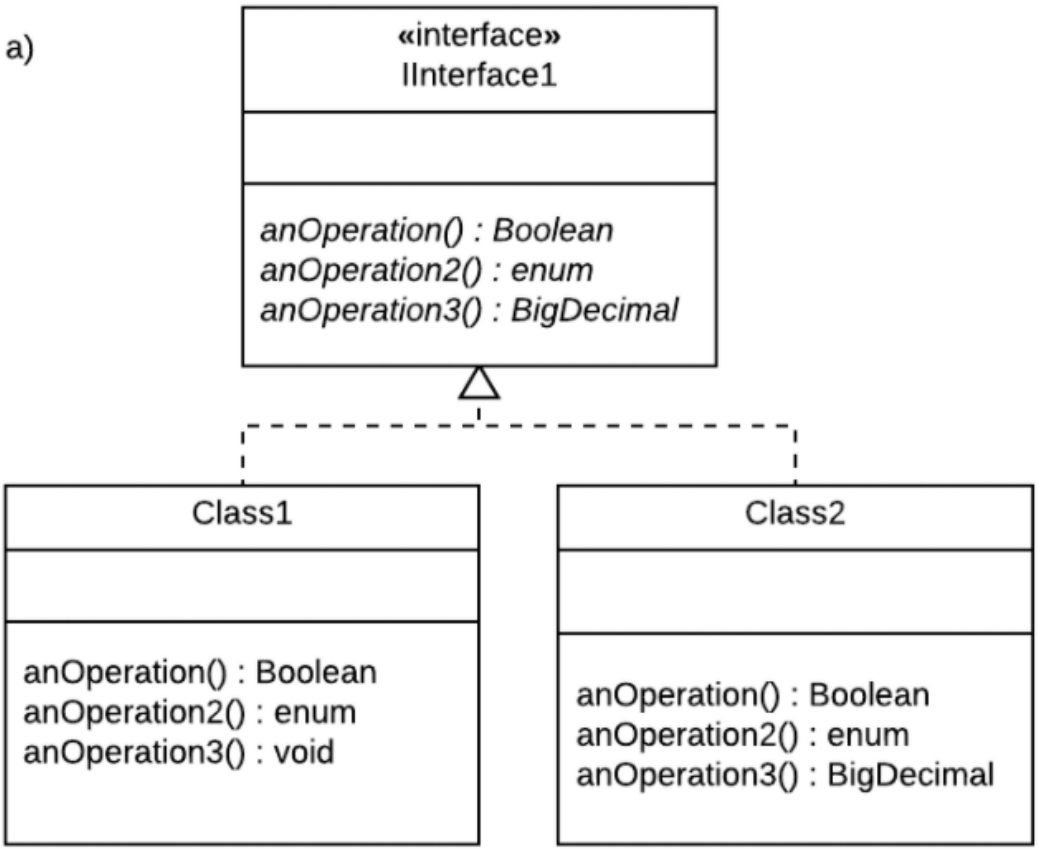
1 / 1 point

- [ ] **The first statement is true**
- [ ] **The second statement is true**
- [ ] **Neither statement is true**
- [ ] **Both statements are true**

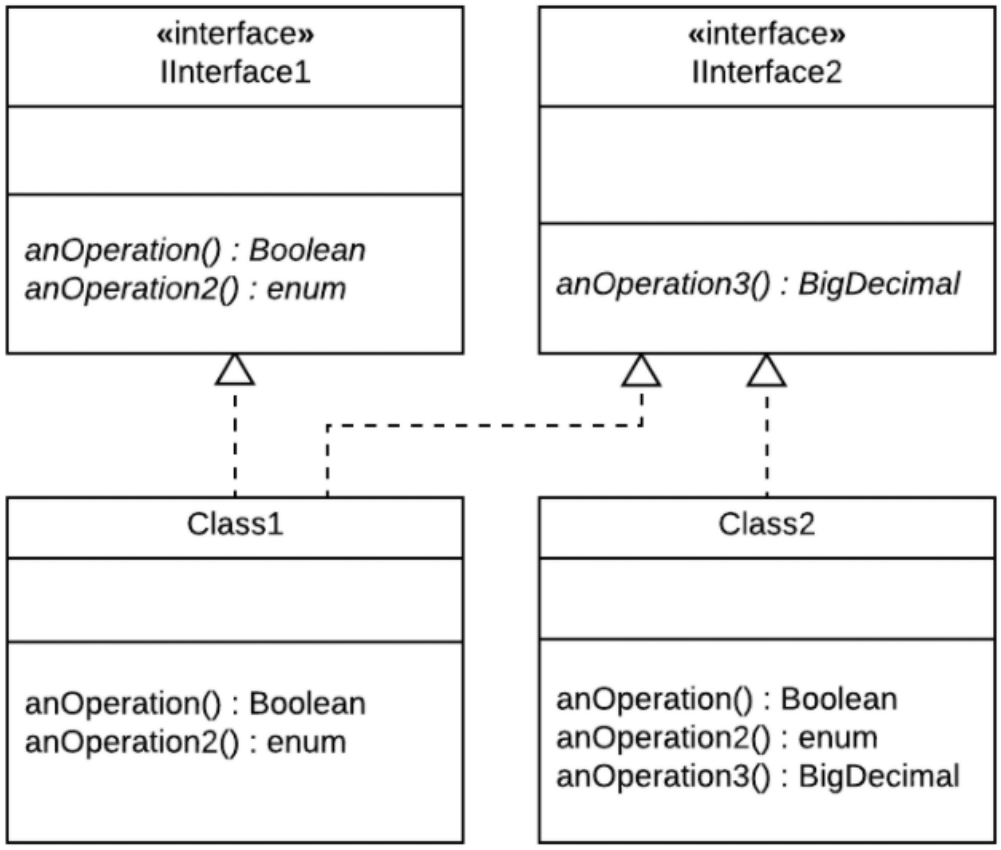
Question 5

Which of these UML diagrams demonstrates the Interface Segregation principle?

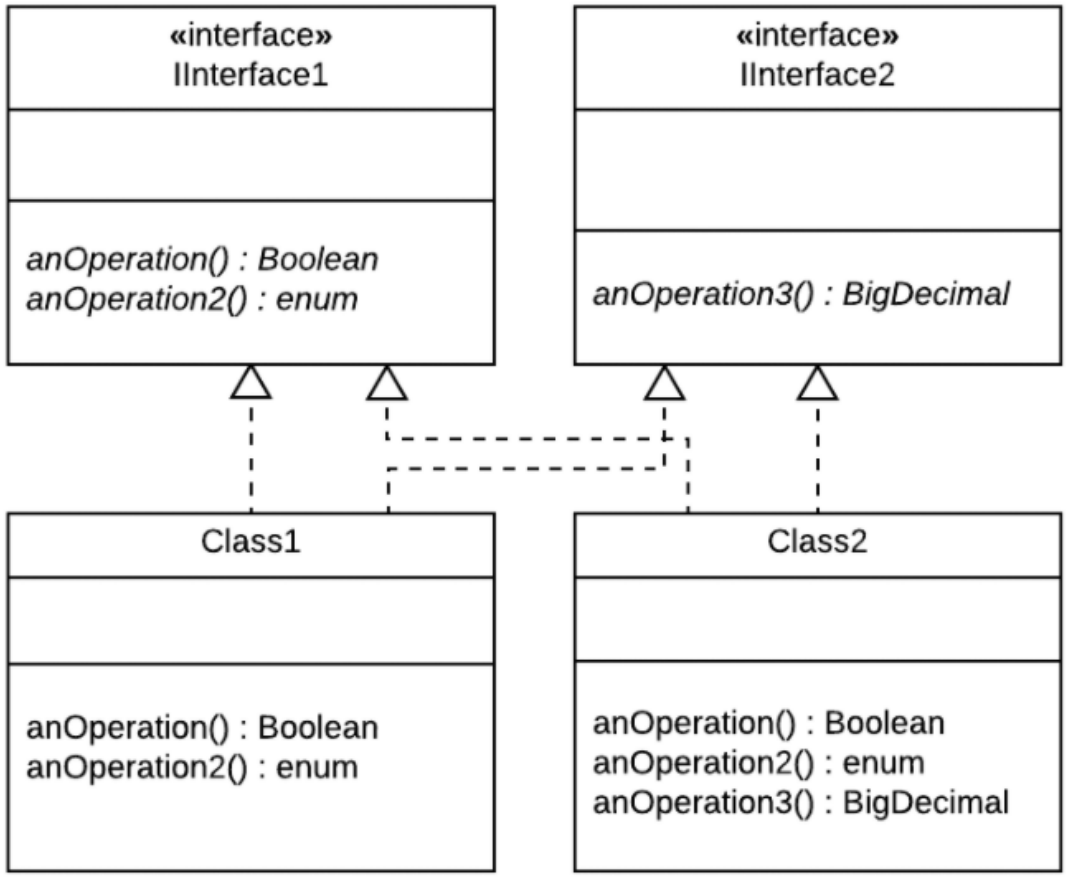
a)


«interface»
IInterface1
anOperation() : Boolean
anOperation2() : enum
anOperation3() : BigDecimal
Class1
anOperation() : Boolean
anOperation2() : enum
anOperation3() : void
Class2
anOperation() : Boolean
anOperation2() : enum
anOperation3() : BigDecimal

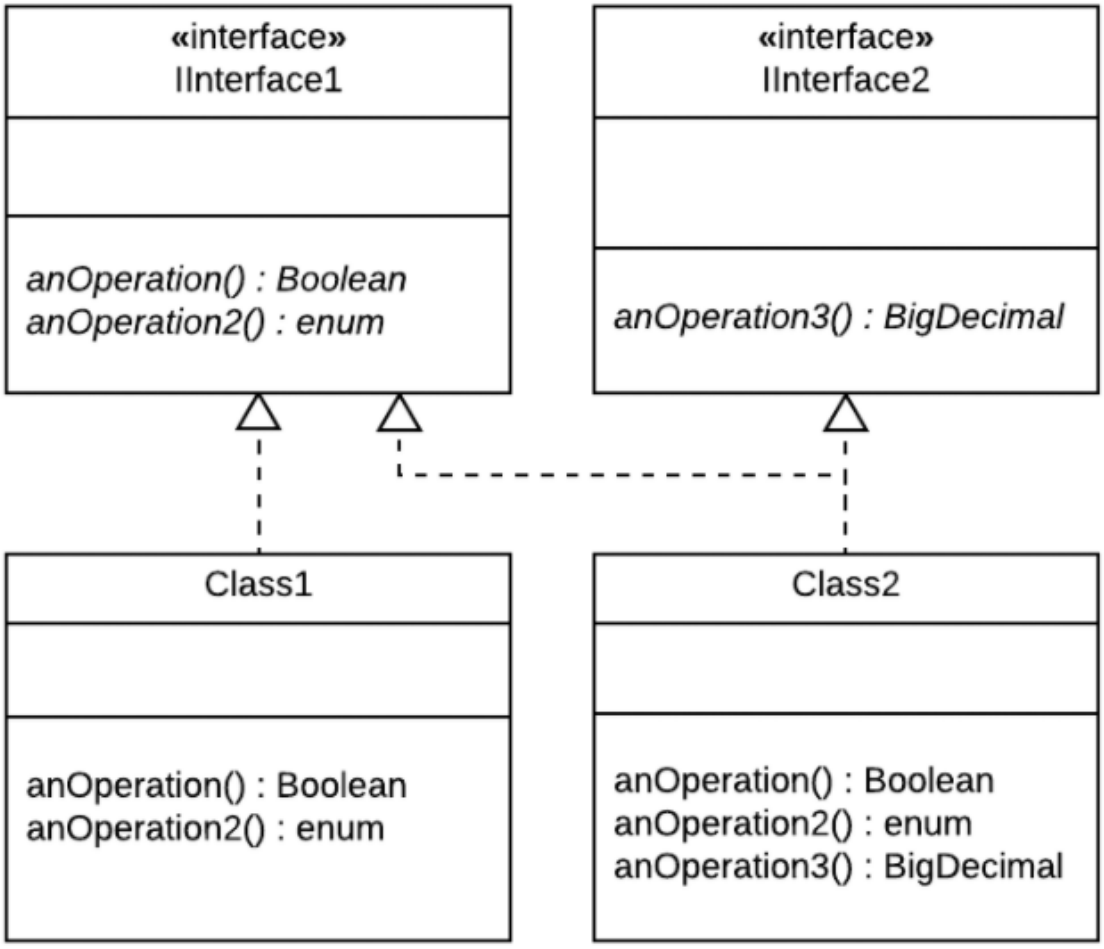
b)


«interface»
IInterface1
anOperation() : Boolean
anOperation2() : enum
«interface»
IInterface2
anOperation3() : BigDecimal
Class1
anOperation() : Boolean
anOperation2() : enum
Class2
anOperation() : Boolean
anOperation2() : enum
anOperation3() : BigDecimal

c)


«interface»
IInterface1
anOperation() : Boolean
anOperation2() : enum
«interface»
IInterface2
anOperation3() : BigDecimal
Class1
anOperation() : Boolean
anOperation2() : enum
Class2
anOperation() : Boolean
anOperation2() : enum
anOperation3() : BigDecimal

d)



- [ ] a
- [ ] b
- [ ] c
- [ ] d

Question 6

Which of these code examples **violates** the Principle of Least Knowledge, or Law of Demeter?

- [ ] 1

```
public class O {
      M I = new M();

      public void anOperation2() {
        this.I.N.anOperation();
      }
    }
```

☐ 2

```
public class Class1 {
      public void N() {
        System.out.println("Method N invoked");
      }
    }

    public class Class2 {
      public void M(Class1 P) {
        P.N();
        System.out.println("Method M invoked");
      }
    }
```

☐ 3

```
public class O {
       public void M() {
         this.N();
         System.out.println("Method M invoked");
       }

       public void N() {
         System.out.println("Method N invoked");
       }
    }
```

☐ 4

```
public class P {
      public void N() {
        System.out.println("Method N invoked");
      }
    }
```

```
public class O {
  public void M() {
P I = new P();
I.N();
    System.out.println("Method M invoked");
  }
}
```

Question 7

How can Comments be considered a code smell?

0 / 1 point

- ☐ **They can't! Comments help clarify code.**
- ☐ **Excessive commenting can be a coverup for bad code**
- ☐ **When a comment is used to explain the rationale behind a design decision**
- ☐ **Too many comments make the files too large to compile.**

Question 8

What is the primitive obsession code smell about?

0 / 1 point

- ☐ **Code that contains many low-level objects, without using OO principles like aggregation or inheritance.**
- ☐ **Overuse of primitive data types like int, long, float**
- ☐ **Using many different primitive types instead of settling on a few that together capture that appropriate level of detail for your system.**
- ☐ **Using key-value pairs instead of abstract data types.**

Question 9

You have a class that you keep adding to. Whenever you add new functionality, it just seems like the most natural place to put it, but it is starting to become a

problem! Which code smell is this?

0 / 1 point

- ☐ **Long Method**
- ☐ **Large Class**
- ☐ **Divergent Change**
- ☐ **Speculative generality**

Question 10

Why is it important to avoid message chains whenever possible?

0 / 1 point

- ☐ **If an unexpected object is returned, this could easily lead to runtime errors.**
- ☐ **They lower cohesion in your class.**
- ☐ **It's a workaround to get to private methods, which are important for encapsulation.**
- ☐ **The resulting code is usually rigid and complex.**

Question 11

Look at the code snippet. Which code smell do you detect?

```
public class Class1 {

  ...

    public void M(Class2 C) {
      C.doSomething(x);
      C.foo(y);
      C.foo2(z, i);
    }
  }
```

- ☐ **Long Parameter List**

- ☐ **Feature Envy**
- ☐ **Inappropriate Intimacy**
- ☐ **Divergent Change**

Question 12

Joseph was developing a class for his smartphone poker game, and decided that one day he would like to be able to change the picture on the backs of the cards, so he created a Deck superclass. Since his app does not have that feature yet, Deck has only one subclass, RegularDeck. What code smell is this?

0 / 1 point

- ☐ **Refused Bequest**
- ☐ **Divergent Change**
- ☐ **Speculative Generality**
- ☐ **Primitive Obsession**