# Knowledge Test K9

COMP 4081 • Software Engineering • Fall 2019

Name: _____, _____
Last name First name

**Rules:**

- No potty breaks.
- Turn off cell phones/devices.
- Closed book, closed note, closed neighbor.

- WEIRD! Do not write on the backs of pages. If you need more pages, ask me for some.

**Reminders:**

- Verify that you have all pages.
- Don't forget to write your name.
- Read each question carefully.
- Don't forget to answer every question.

1. [2] Which of the following best defines *software design pattern*?

    a) Application programming interface for a code library

    b) General, reusable solution to a commonly occurring design problem

    c) Reusable code library for a particular problem domain

2. [4] List two key benefits that design patterns provide. (Three benefits were discussed in class.)

3. [2] Which book popularized software design patterns?

    a) *Pattern-Oriented Software Architecture* (aka the "POSA" book)

    b) *Design Patterns: Elements of Reusable Object-Oriented Software* (aka the "GoF" book)

    c) *Head First Design Patterns* (aka the "HFDP" book)

4. [2] How is the structure of a design pattern typically expressed?

    a) Component diagram

    b) Dataflow diagram

    c) Class diagram

Consider the Observer design pattern diagram in Figure 1.

5. [2] Which of the following best characterizes a `Subject` in the pattern?

    a) Object that listens for changes to another object and reacts to those changes

    b) Object that encapsulates both subscriber and publisher objects

    c) Object that is being watched for state changes

6. [2] Which of the following best characterizes an `Observer` in the pattern?

    a) Object that listens for changes to another object and reacts to those changes

    b) Object that encapsulates both subscriber and publisher objects

    c) Object that is being watched for state changes

7. [2] Which of the following best describes the purpose of the `Subject#attach` operation?

    a) Operation that a subject object invokes when the subject changes state and that in turn invokes an operation on each of the associated observer objects

    b) Operation that adds an observer object the list of such objects maintained by a subject

    c) Observer-specific operation that defines how a particular observer object reacts to changes in a watched subject object

8. [2] Which of the following best describes the purpose of the `Subject#notify` operation?

    a) Operation that a subject object invokes when the subject changes state and that in turn invokes an operation on each of the associated observer objects

    b) Operation that adds an observer object the list of such objects maintained by a subject

    c) Observer-specific operation that defines how a particular observer object reacts to changes in a watched subject object

9. [2] Which of the following best describes the purpose of the `Observer#update` operation?

    a) Operation that a subject object invokes when the subject changes state and that in turn invokes an operation on each of the associated observer objects

    b) Operation that adds an observer object the list of such objects maintained by a subject

    c) Observer-specific operation that defines how a particular observer object reacts to changes in a watched subject object

10. [6] Describe in plain English all the changes and additions you would need to make in order to apply the Observer Pattern to solve the design problem in Scenario 1. Such changes/additions may involve classes, attributes, methods, associations, generalization relationships, etc.

11. [2] Which objects play the role of subject in the application of the Observer Pattern?

    a) Class `Article`

    b) Class `Editor`

    c) Class `EditorNotifier`

12. [2] Which objects play the role of observer in the application of the Observer Pattern?

    a) Class `Article`

    b) Class `Editor`

    c) Class `EditorNotifier`

## Bonus Problems

13. [5] Scenario: Developer runs `git merge master`. Assume that auto-merge, if used, would complete successfully with no merge conflicts.

```
* 48eb6 (master)
* bafad (HEAD -> iss1)
| * 4030e
|/
* ed379
```

14. [3] Would the scenario in question 13 result in a fast-forward merge? Explain why.

15. [5] Draw a control-flow graph for the function in Figure 2. In addition to the usual CFG features, label each node with the corresponding code-line number.

16. [5] **Statement Coverage:** For each test in the Figure 3 test suite, list the nodes covered by the test.

Test 1)

Test 2)

17. [2] Does the test suite achieve statement coverage? If not, what nodes did the test suite miss?

18. [2] Which, if any, of the tests in Figure 3 would reveal this bug?

## Figures

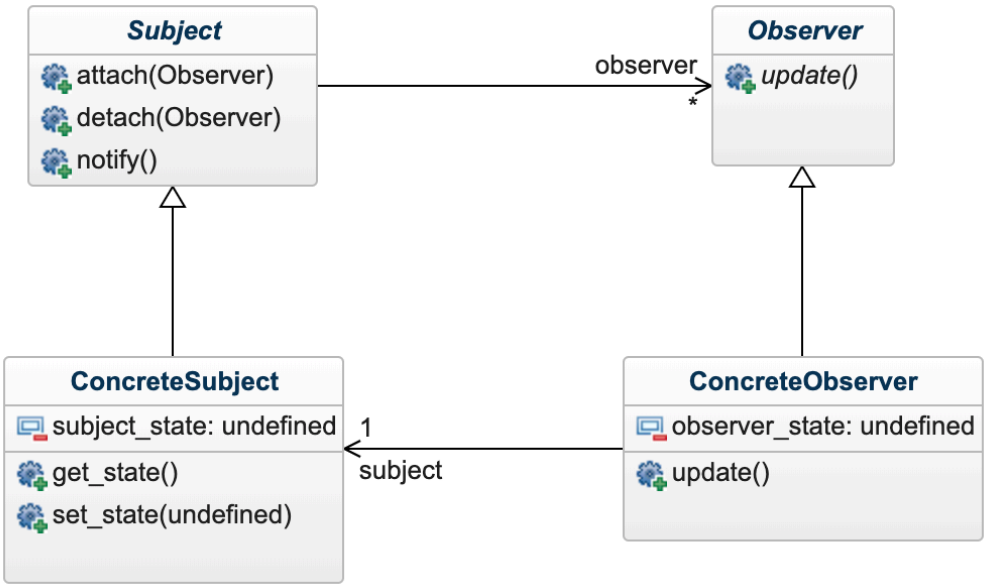
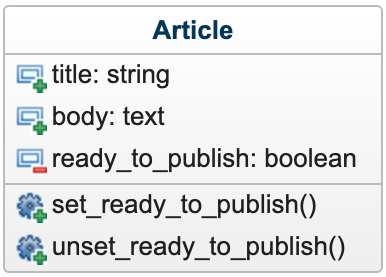



**Figure 1. Observer Pattern diagram.**

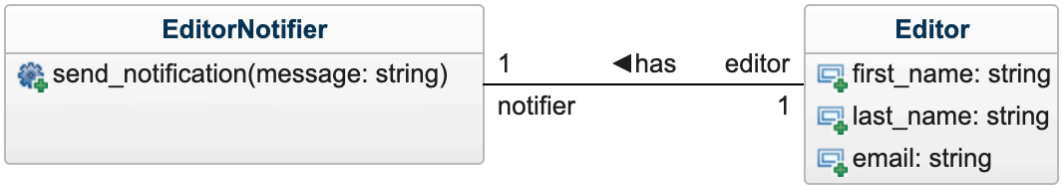
Imagine that you are designing a web app for an online magazine, and you have a design problem that you must solve.

Your app currently has the following model class `Article` that represents a magazine article:




When an article writer first creates an article, the `ready_to_publish` attribute is set to `false`. When the author has completed the article, they use the app to set the attribute to `true`. In particular, the system uses the method `Article#set_ready_to_publish` to set the attribute to `true`.

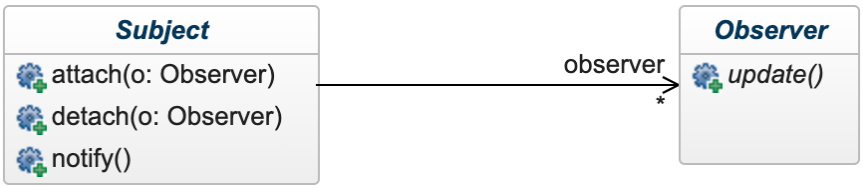
The app also has an editor-notification subsystem:




An editor is the person who reviews an article before it is actually published in the magazine. Each editor is represented by an `Editor` object in the system. Each `Editor` object has an associated `EditorNotifier` object that can be used to send email notifications to the editor by calling the `EditorNotifier#send_notification` method.

Here's the design problem that you must solve: an `EditorNotifier` object must be able to "watch" an `Article` object such that when the `ready_to_publish` attribute is set to `true`, the `EditorNotifier` object will send an email notification to the associated editor.

To solve this problem, you must apply the Observer Pattern. To assist you, a support library for the Observer Pattern has been provided. In particular, the library contains the following:



```
 1  def find_largest(array)
 2      largest = array[0]
 3      i = 1
 4      while i < array.length
 5          if array[i] < largest
 6              largest = array[i]
 7          end
 8          i = i + 1
 9      end
10      return largest
11  end
```

**Figure 2. A function that finds the largest value in an array. The function has a precondition that the array argument must have a length of at least one. Note that this function contains a bug on line 5.**

| Test # | Input | Expected Output |
|---|---|---|
| | `array` | |
| 1 | `[ 1 ]` | `1` |
| 2 | `[ 2, 1 ]` | `2` |

**Figure 3. A test suite for the function in Figure 2.**