COMP 7012

Exam 2

Spring 2018

Name: Solutions ,
Last name First name

Rules:

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

- <u>WEIRD!</u> 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 <u>carefully</u>.
- Don't forget to answer <u>every</u> question.

1. [2pts] In software engineering, defects that are discovered ______ are ______ to fix.

   a) by customers; less expensive

   b) by developers; more expensive

   c) earlier; more expensive

   [✓ d)] later; more expensive

   e) None of the above

2. [2pts] Following a ______ software engineering process tends to reveal defects early in development.

   a) Waterfall

   [✓ b)] Iterative

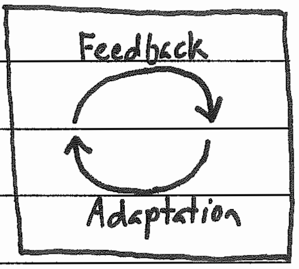
   c) Sequential

   d) All of the above

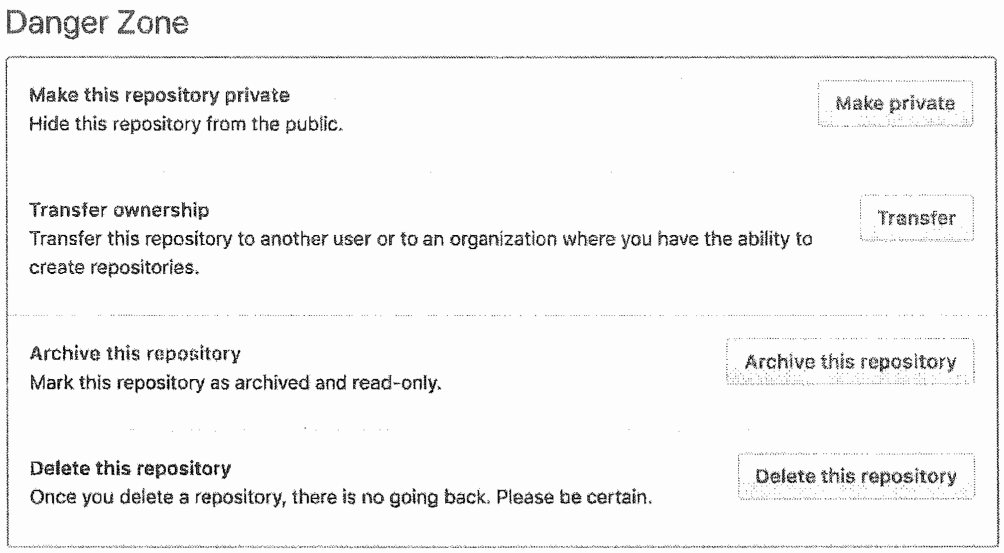
   e) None of the above

3. [4pts] What are the key features of an *empirical process model*, does it effectively address *unstable requirements* (i.e., ones that are prone to change), and if so, how?

Key Features of an empirical process model is that it iterates between Feedback and adaption.
Yes, it effectively addresses unstable requirements.
It does so via the Feedback part of the process: as requirements change, the changes are revealed From the Feedback. Because they are discovered quickly, they can be attended to early on when the cost of Fixing them is lowest.



Consider the following GitHub repository settings:


Danger Zone

Make this repository private
Hide this repository from the public.

Transfer ownership
Transfer this repository to another user or to an organization where you have the ability to create repositories.

Archive this repository
Mark this repository as archived and read-only.

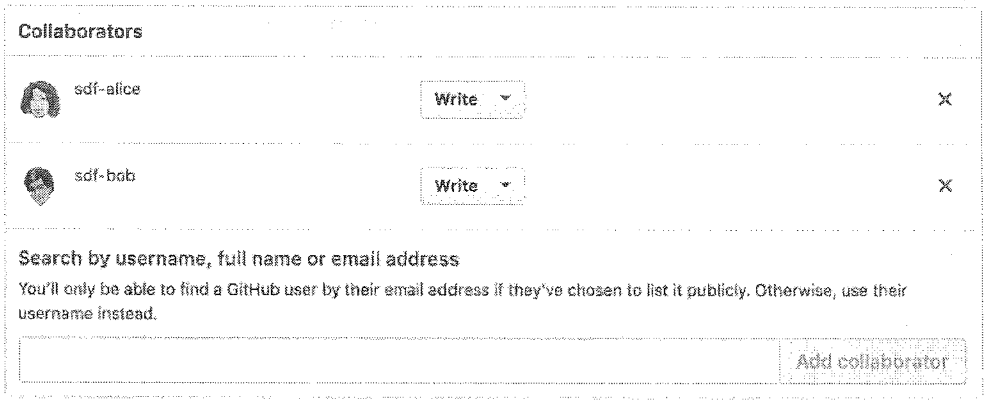
Delete this repository
Once you delete a repository, there is no going back. Please be certain.

4. [6pts] Reverse engineer <u>one</u> user story that records a requirement of your choice for the above settings. You must apply the templates described in class, and your US must have the other attributes of good user stories, which we discussed in class. (You may omit the US's estimate and priority.)

Many possible answers. Here are the templates:
Title: <verb><noun>
Description: As a <who>, I want to <what> <why>.
Some Key attributes are describing one thing, using the customer's language, not being a long essay, and not using technical jargon.
Here's an example:

Title: Make repository private
Description: As a repository owner, I want to make the repository private, so I can keep my work private until such a time that I choose to release it.

Consider this GitHub repository interface for adding collaborators to a project:


Collaborators

sdf-alice — Write — ×

sdf-bob — Write — ×

Search by username, full name or email address

You'll only be able to find a GitHub user by their email address if they've chosen to list it publicly. Otherwise, use their username instead.

Add collaborator

Here is a user story related to the above interface:

AJAX Remove-Collaborator Button
The Collaborators page must have an "X" button next to each collaborator, and the button must use AJAX such that clicking the button causes the collaborator to be removed immediately.

5. [4pts] Describe two things that make this a poor-quality user story.

Three possible things:
(1) Mentions specific implementation technology (AJAX)
(2) Uses technical jargon that the customer might not understand (AJAX again)
(3) Mentions specific features of the user-interface design ("X" buttons)

6. [2pts] T or F? In general, the larger the estimate, the less likely it is to be accurate.

   [✓ a)] True

   b) False

7. [2pts] Which of the following approaches/techniques leverages the collective opinion of a group of individuals rather than that of a single expert? Circle <u>all</u> answers that apply.

   a) Black-box testing

   [✓ b)] Planning Poker

   c) Writing user stories

   [✓ d)] Wisdom of the Crowd

   e) None of the above

8. [2pts] In the agile development process taught in class, the ______ estimate each user story, the ______ decide the priority for each story, and the ______ choose which user stories to implement in the next iteration.

   a) developers; customers; customers

   b) customers; developers; customers

   c) customers; customers; developers

   d) customers; developers; developers

   [✓ e)] developers; customers; developers

9. [2pts] Exhaustive testing is ______ and, in general, is ______ performed in practice.

   a) a black-box technique; often

   b) a white-box technique; never

   c) writing a test for every possible output; often

   [✓ d)] writing a test for every possible input; never

   e) writing a test for every user story; often

10. [2pts] For each piece of text below, place a "B" or a "W" next to it if it corresponds to *Black-Box* or *White-Box* testing, respectively.

__B__ Tests focus on boundary cases

__B__ Tests based only on the interface of a component

__W__ Tests based on the implementation of a component

__W__ Tests aim to achieve particular levels of code-coverage

11. [2pts] For each piece of text below, place a "U" and/or "I" and/or "S" next to it if it corresponds to *Unit* and/or *Integration* and/or *System* tests, respectively.

__U, I__ Tests something less than the whole system

__I, S__ May perform I/O

__U__ Should not have non-determinism

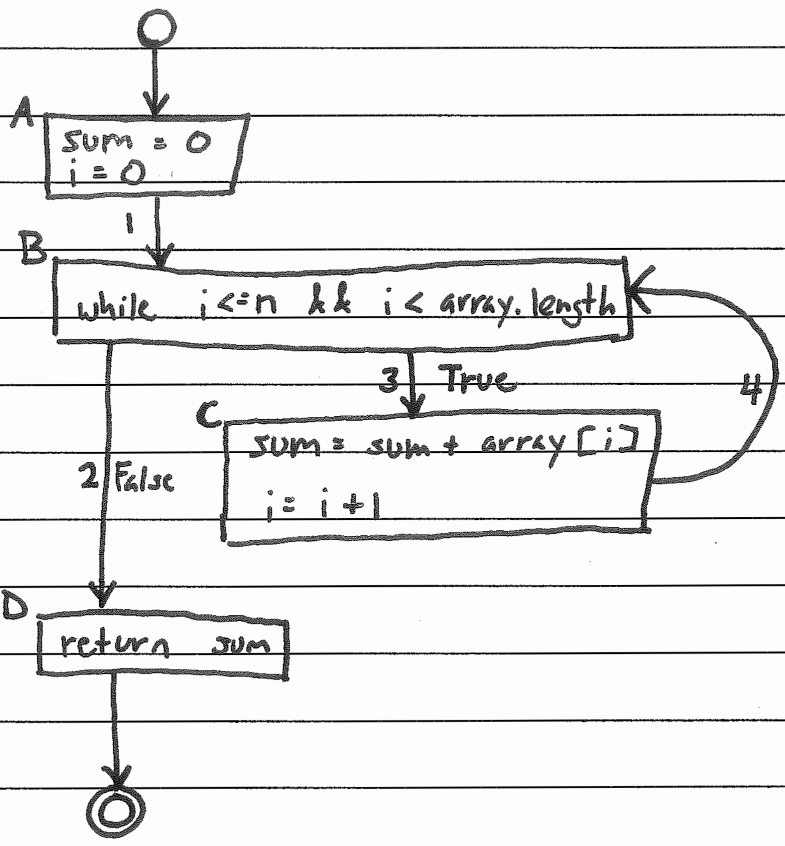
__U__ Should be fast (less than half a second)

12. [8pts] Using the fragments below, create a functional test (class and method) for the “index” page of a movie-themed web app (which has the typical scaffold layout). The test should do the following in this order (1) retrieve user fixture “one” and sign in the user, (2) simulate an HTTP request for the index page, (3) check that the HTTP response does not report an error, (4) check that the rendered HTML table contains a cell with the most famous shark movie of all time, and (5) check that the correct ERB is rendered (index.html.erb). Some fragments may be used more than once in your solution. Some fragments may not be used at all.

a) `album = albums(:one)`
b) `assert_response :error`
c) `assert_response :success`
d) `assert_select "h1", "Movie"`
e) `assert_select "td", "Jaws"`
f) `assert_template :index`
g) `assert_template :movies`
h) `class Movie < ApplicationRecord`
i) `class MoviesControllerTest < ActionDispatch::IntegrationTest`
j) `end`
k) `get movies_url`
l) `get movies_url(@movie)`
m) `include Devise::Test::IntegrationHelpers`
n) `sign_in user`
o) `test "should display movies" do`
p) `user = users(:one)`

i
m
o
p
n
k
c
e
f
j
j

13. [4pts] Draw a control-flow graph (CFG) for the function in Figure 1. In addition to the usual CFG features, label the nodes with capital letters (A, B, C, etc.), and label the edges with numbers (1, 2, 3, etc.). Don't forget to include entry and exit points.



Use the CFG you created for the function in Figure 1 to answer the following questions.

14. [2pts] Fill in the table below with a test suite that provides statement coverage. In the Covers column, list the letter labels (A, B, C, etc.) of the nodes covered by each test case.

Input		Expected Output	Covers
array	n		
[1, 1]	1	1	A, B, C, D

15. [3pts] Fill in the table below with a test suite that provides branch coverage. In the Covers column, list the number labels (1, 2, 3, etc.) of the edges covered by each test case (only true/false edges needed).

Input		Expected Output	Covers
array	n		
[1, 1]	1	1	2, 3

16. [4pts] Fill in the table below with a test suite that provides path coverage. Before you fill in the table, first list all the paths to be covered, and label each path ("P1", "P2", "P3", etc.). You need only cover executions that involve at most 1 iteration of each loop (if there are any). In the Covers column, list the path labels covered by each test case.

Paths:

P1: 1,2

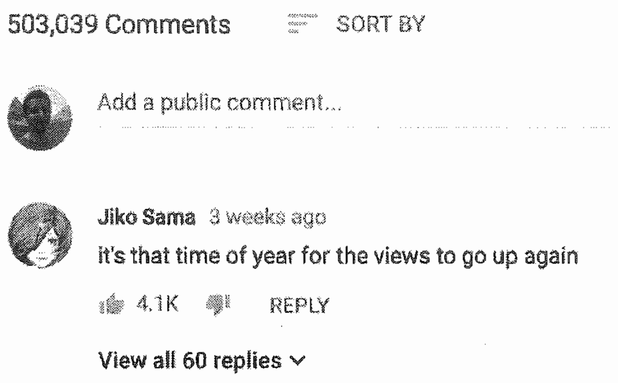
P2: 1,3,4,2

Input		Expected Output	Covers
array	n		
[ ]	0	0	P1
[1]	1	1	P2

17. [2pts] Which, if any, of your above three test suites would have caught the bug in the function?

The statement- and branch-coverage suites would have caught the bug (but not the path-coverage one).

Consider the YouTube comment section interface:


503,039 Comments SORT BY

Add a public comment...

Jiko Sama 3 weeks ago

it's that time of year for the views to go up again

18. [2pts] Which of the following attacks might some attempt to perpetrate by adding a public comment? Circle all that apply.

    a) Packet sniffing

    [✓ b)] Cross-site scripting

    c) Eavesdropping

    d) Man-in-the-middle attack

    [✓ e)] SQL injection

19. [2pts] How do you prevent the above attack(s)? Circle all that apply.

    [✓ a)] Sanitize inputs

    b) Redirect requests

    [✓ c)] Escape input characters

    d) Disable cookies

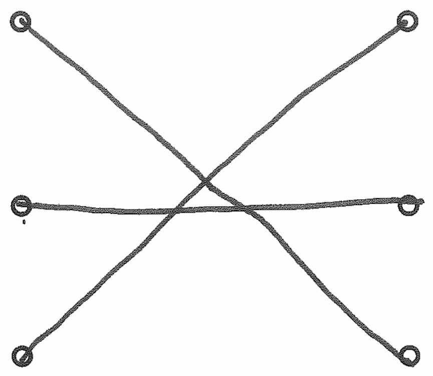
    e) Authenticate users

20. [6pts] Match the Design Pattern to an example usage of the pattern.

Observer

Singleton

Adapter



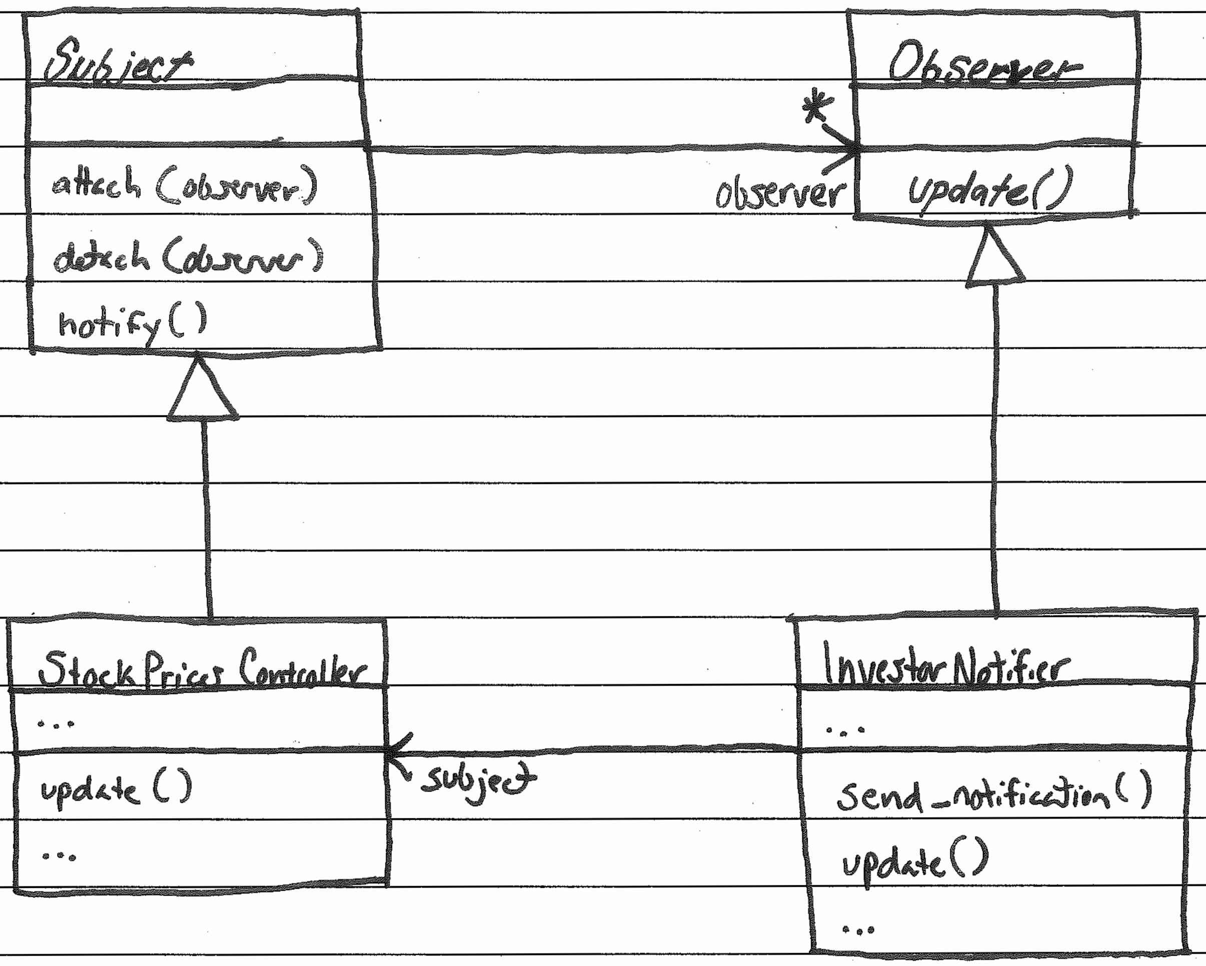
Your program must support switching among several different foreign language translation libraries (e.g., FrenchToX, HindiTranslatePro, English4Me), but each one has a slightly different interface.

Your point-of-sales application must log data from many different places in its code, but only one Logger object should do the logging for the entire system.

Your *Doom* program needs to listen for keyboard and mouse clicks to manipulate how a player character moves, shoots, etc.

(One more question to go on the next page!)

21. [7pts] Recall the Observer Design Pattern depicted in Figure 2. Imagine that you are designing a web app for an investment company. Figure 3 depicts the classes that you have so far. In particular, you have designed a StockPricesController class that records price changes to stocks. As part of this controller's responsibilities, it must update stock price entries as they change. You have also designed a InvestorNotifier that is capable of sending notification messages to investors. The design problem you need to solve is how to make a InvestorNotifier "listen" for when a StockPricesController updates a stock price, and to send a notification to affected investors whenever that happens. Draw a class diagram that applies the Observer Design Pattern to solve this problem. Use the same names used in the design pattern as much as possible (except make Ruby style). You must include all the classes from Figure 3 in your diagram (i.e., your changes should be additive). In particular, I expect that you will be adding classes, operations, inheritance relationships, and associations.



Figures

```
def sum_the_first_n(array, n)
  sum = 0
  i = 0
  while i <= n && i < array.length
    sum = sum + array[i]
    i = i + 1
  end
  return sum
end
```

Figure 1. Buggy function that sums the first *n* numbers in an array.

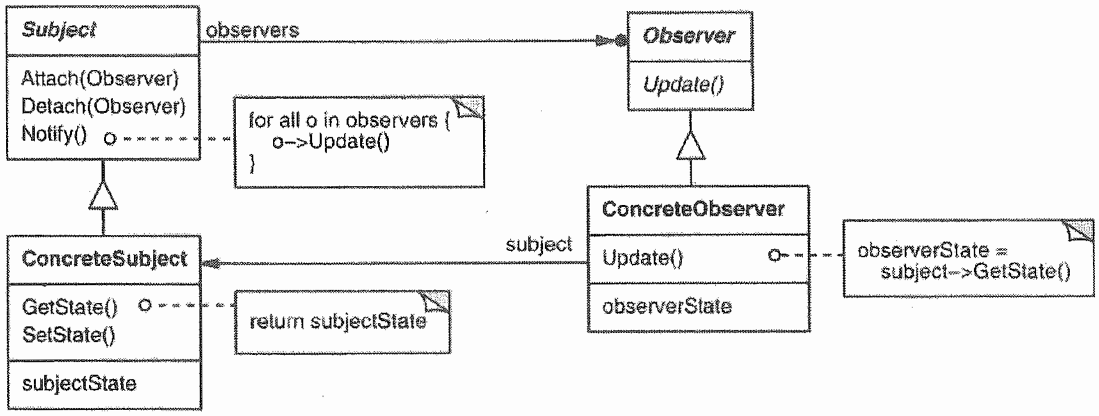


Figure 2. Observer Pattern from the "Gang of Four" book. (Note that the book uses an outdated class diagram notation.)

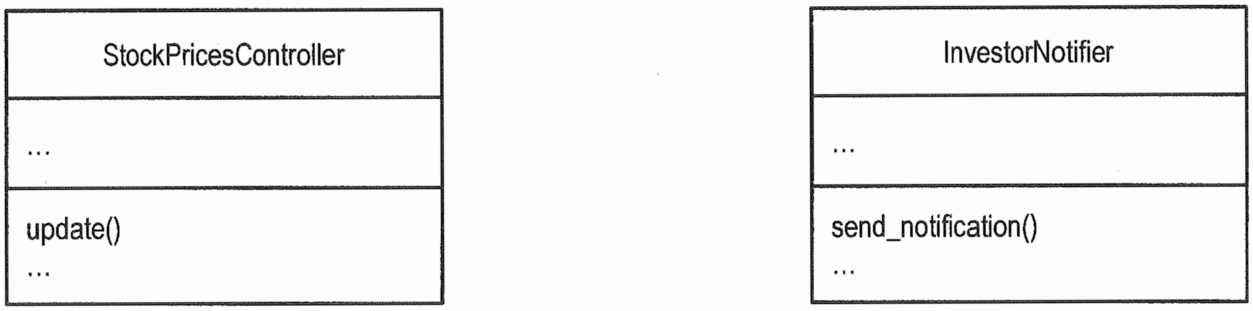


Figure 3. Classes for investment company web app.