COMP 4081

Exam 2

Fall 2016

Name: Solutions , _____

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. [2pts] Which term is best defined by the following text?

   Development of a system through repeated cycles and in smaller portions at a time, allowing software developers to take advantage of what was learned during development of earlier parts or versions of the system

   a) Verification and validation

   b) Waterfall development process

   c) Configuration management

   d) Iterative development process

   e) None of the above

2. [2pts] Which term is best defined by the following text?

   Development of a system whereby progress is seen as flowing steadily downwards through the phases of conception, analysis, design, construction, testing, production, and maintenance, and wherein one should move to a phase only when its preceding phase is reviewed and verified

   a) Verification and validation

   b) Waterfall development process

   c) Configuration management

   d) Iterative development process

   e) None of the above

3. [2pts] Which of the following is an issue associated with the waterfall development process?

   a) Can lead to "analysis paralysis" wherein a considerable investment of time and effort is put into a project before any code is written

   b) System defects are often discovered late in the development process

   c) Falsely assumes that requirements are stable and can be known from the start

   d) All of the above

   e) None of the above

4. [2pts] In iterative development, how long should an iteration generally be?

   a) 6 months to a year

   b) 2–4 months

   (c) 2–6 weeks

   d) 1 week

   e) None of the above

5. [6pts] Think of the *Piazza* system that we've used in class. Reverse engineer one user story that records a requirement for the system. 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's the templates.

Title: <verb><noun>

Description: As a <who>, I want <what> <why>.

Here are the other key attributes:

- Describe one thing
- Use customer's language
- Not be long essay
- Not use technical terms

6. [4pts] Describe two things that are wrong with this user story for GitHub functionality.

> AJAX Profile Form
> The update-profile form should use AJAX so that when the user presses the "Update profile" button, the data is saved, but the page does not reload.

(1) Mentions specific implementation technology (AJAX)

(2) User technical jargon (AJAX) that the customer might not understand

(3) Mentions specific features of the UI design (button)

7. [2pts] Which of the following activities should not be done by the developers?

a) US creation

b) US corrections

c) Add estimations

(d) Set priorities of USs

e) None of the above

8. [2pts] T or F? The larger the estimate, the more likely it is to be accurate.

a) True

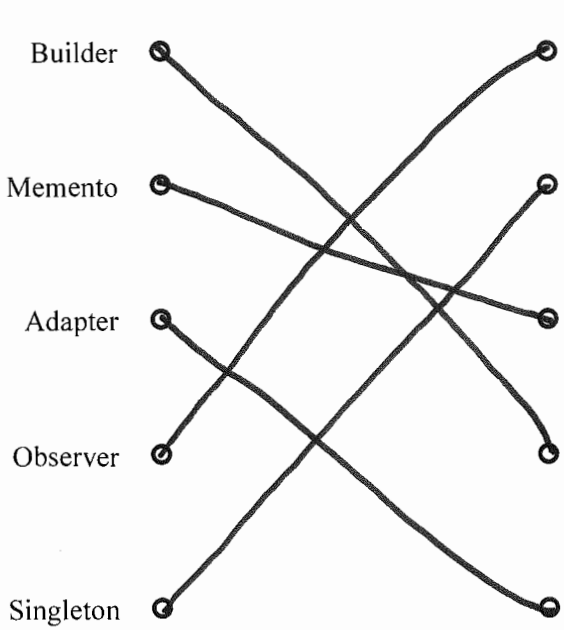
(b) False

9. [2pts] T or F? To estimate work, developers commonly use their own past performance and/or the "wisdom of the crowd."

(a) True

b) False

10. [6pts] Match the Design Pattern to an example usage of the pattern.



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

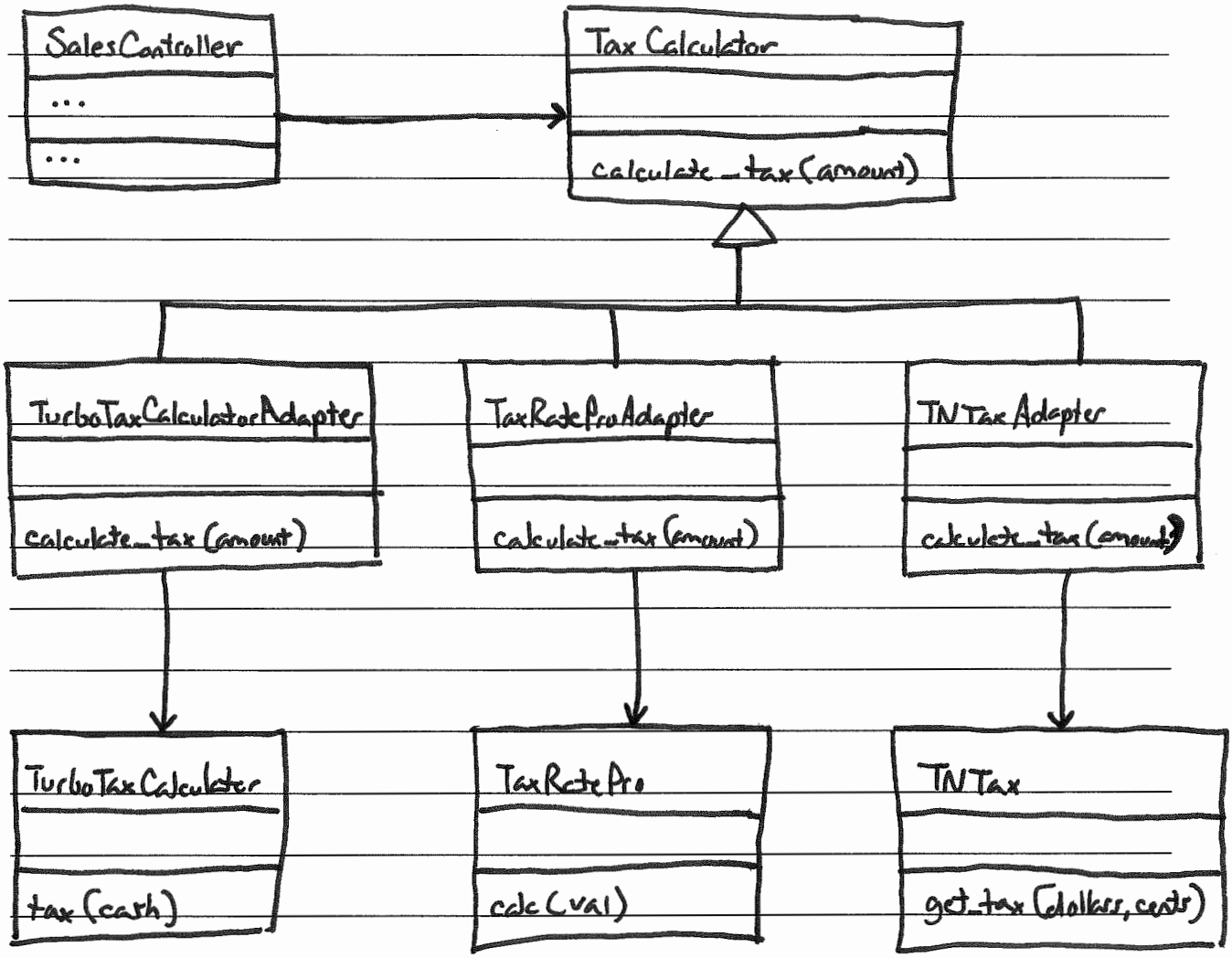
Your web app must log data from many different places in its code, but only one Logger object should do the logging for the entire system.

You want your application to save its state so that if it crashes, then it can auto-recover.

Your application needs to generate large, complex XML files (from scratch).

Your program must support switching among several different database management system libraries (e.g., MySQL, PostgreSQL, SQLite), but each one has a slightly different interface.

11. [8pts] Recall the Adapter Design Pattern depicted in Figure 1. Imagine that you are designing a web app for on-line sales. Figure 2 depicts the classes that you have so far. In particular, you have designed a SalesController class that records sales. As part of this controller's responsibilities, it must calculate the sales tax. In your design, the responsibility of performing the tax calculation will be delegated to one of several off-the-shelf tax calculators (e.g., TurboTaxCalculator, TaxRatePro, and TNTax). Different customers prefer different calculators, so you must be able to quickly swap them in and out of your application. Draw a class diagram that applies the Adapter Design Pattern to solve this problem. Make it so that the controller interacts with the calculators via the TaxCalculator interface. You must include all the classes from Figure 2 in your diagram (i.e., your changes should be additive).



12. [2pts] Which of the following is true of *exhaustive testing*?

a) Tests all possible inputs

b) Typically results in an intractably large set of test cases even for small programs

c) Generally infeasible in practice

(d) All of the above

e) None of the above

13. [2pts] Which of the following is <u>not</u> a difference between black-box and white-box testing?

(a) White-box tests are made by programmers, whereas black-box tests are made by ordinary users

b) Black-box tests often focus on boundary cases, whereas white-box tests tend not to

c) Black-box tests are based only on the interface of a component, whereas white-box tests are based on the implementation

d) White-box tests often aim to achieve particular levels of code-coverage, whereas black-box tests do not

e) None of the above (they are all differences)

14. [2pts] Which of the following is <u>not</u> a difference between unit tests and integration tests?

a) Unit tests should not perform I/O, whereas integration tests may do so

b) Unit tests should be deterministic, whereas integration tests may have non-determinism

c) Unit tests should be fast (less than half a second), whereas integration tests may be slower

(d) Unit tests must be black-box tests, whereas integration tests must be white-box tests

e) None of the above (they are all differences)

15. [8pts] Consider these code fragments:

```
a) user = users(:one)
b) include Devise::Test::IntegrationHelpers
c) assert_response :error
d) assert_response :success
e) assert_select "h1", "Book"
f) assert_select "td", "Jaws"
g) sign_in user
h) class BooksControllerTest < ActionDispatch::IntegrationTest
i) assert_template :index
j) assert_template :books
k) test "should display books" do
l) get books_url
m) end
```

Using the above fragments, create a functional test (class and method) for the "index" page of a book-themed web app (illustrated in Figure 3). 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 correct ERB is rendered (index.html.erb), and (5) check that the rendered HTML table contains a cell with everyone's favorite shark book. Some fragments may be used more than once in your solution. Some fragments may not be used at all

h

b

k

a

g

l

d

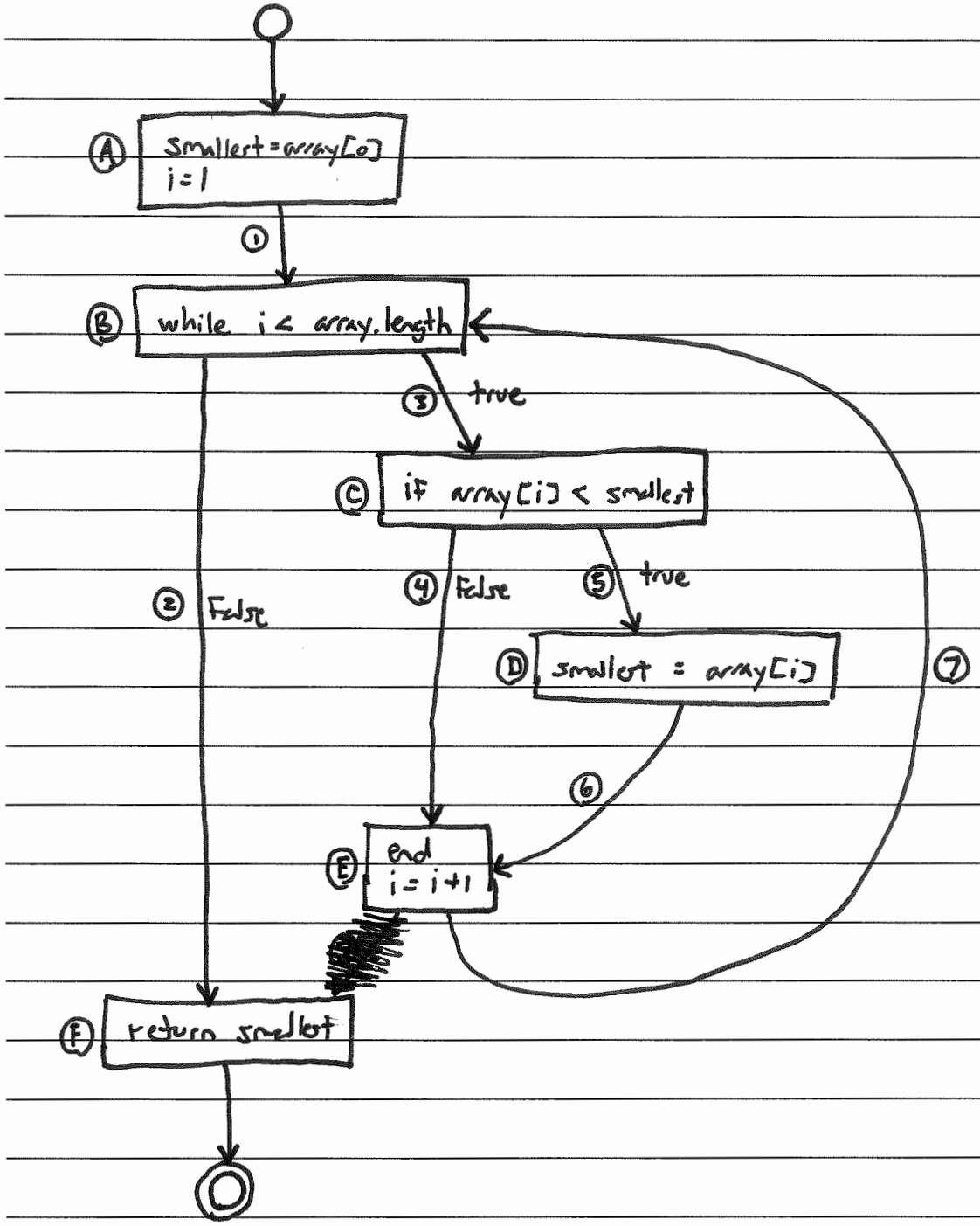
i

f

m

m

16. [5pts] Draw a control-flow graph (CFG) for the function in Figure 4. 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 4 to answer the following questions.

17. [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 array	Expected Output	Covers
[1,0]	0	A, B, C, D, E, B, F

18. [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 array	Expected Output	Covers
[1,0,2]	0	3, 5, 3, 4, 2

19. [4pts] Fill in the table below with a test suite that provides path coverage. In the Covers column, list the number labels (1, 2, 3, etc.) of the edges covered by each test case. You need only cover executions that involve at most 1 iteration of each loop (if there are any). Before you fill in the table, list all the paths to be covered.

Paths:

- 1, 2
- 1, 3, 5, 6, 7, 2
- 1, 3, 4, 7, 2

Input array	Expected Output	Covers
[0]	0	1,2
[1,0]	0	1,3,5,6,7,2
[0,1]	0	1,3,4,7,2

20. [2pts] How do you prevent SQL injection attacks?

a) Escape queries

b) Interrupt requests

c) Merge tables

d) All of the above

e) None of the above

21. [2pts] What is an effective way to prevent cross-site scripting attacks?

a) Place your server behind a firewall

b) Disable all form input

c) Escape any input text that would be displayed on a webpage

d) All of the above

e) None of the above

Figures

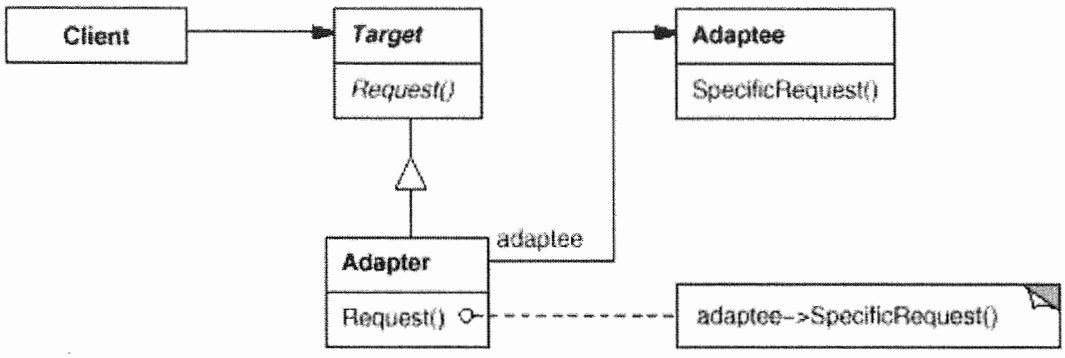


Figure 1. Adapter Pattern from the "Gang of Four" book. (Note that the book uses an outdated class diagram notation.)

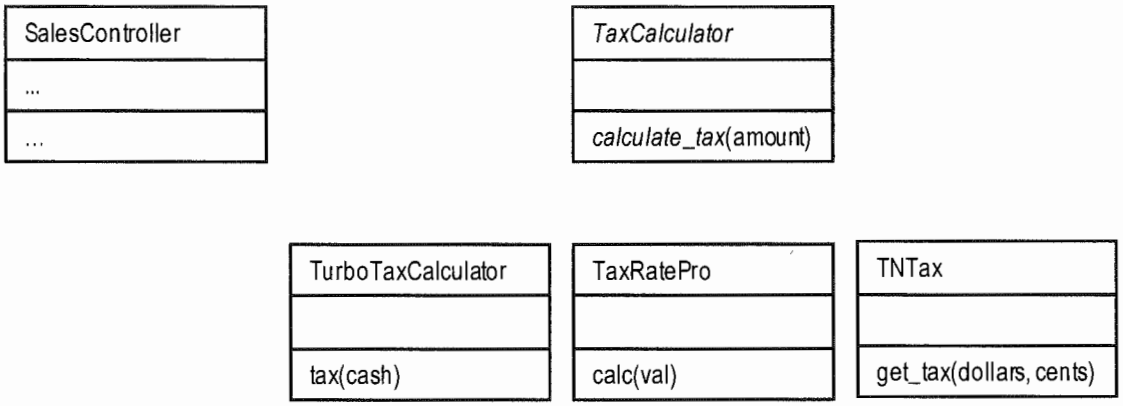


Figure 2. Classes for on-line sales web app.

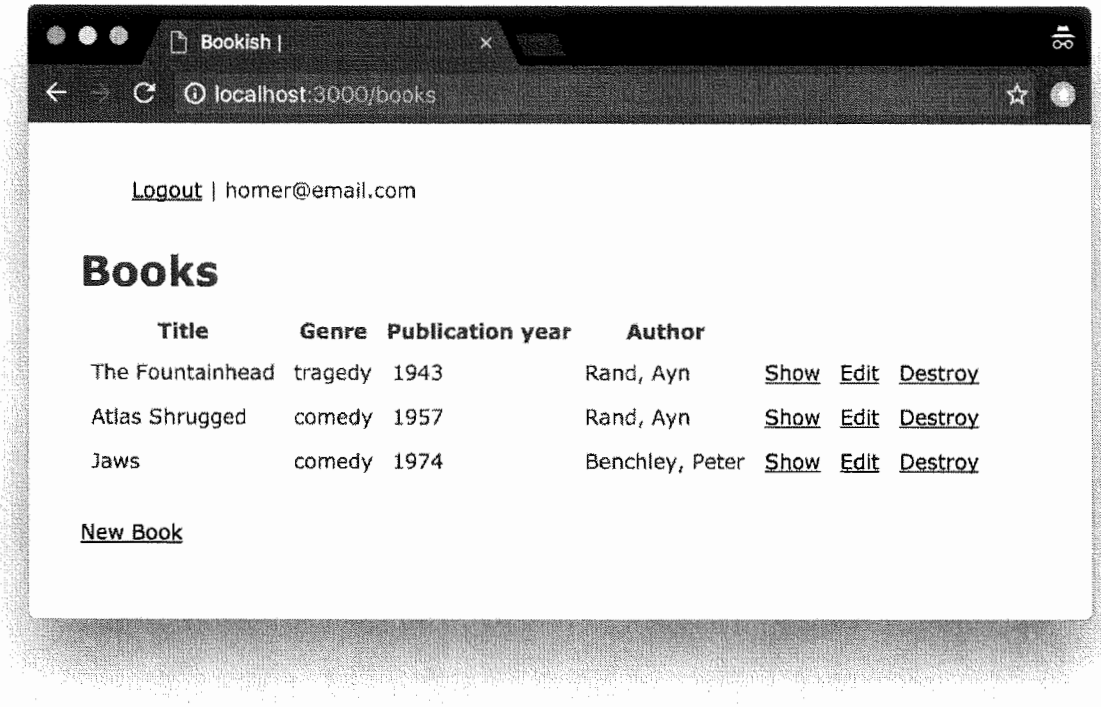


Figure 3. Books index page.

```
def find_smallest(array)
  smallest = array[0]
  i = 1
  while i < array.length
    if array[i] < smallest
        smallest = array[i]
    end
    i = i + 1
  end
  return smallest
end
```

Figure 4. Function that finds the smallest value in an array.