Final Exam Practice Questions

Question 1. Which design pattern is most appropriate for each of the following situations?

A. You need to call a subsystem using a long series of complex lines of code.

- a. Builder
- **b. Façade**
- c. Memento
- d. Factory

B. Your code has to decide at runtime which class to instantiate.

- a. Builder
- b. Interpreter
- c. Memento
- **d. Factory**

C. Your code will need to save the state of an object and then reload it later.

- a. Builder
- b. Façade
- **c. Memento**
- d. Observer

D. Your system will need to support scriptability

- a. Builder
- **b. Interpreter**
- c. Memento
- d. Factory

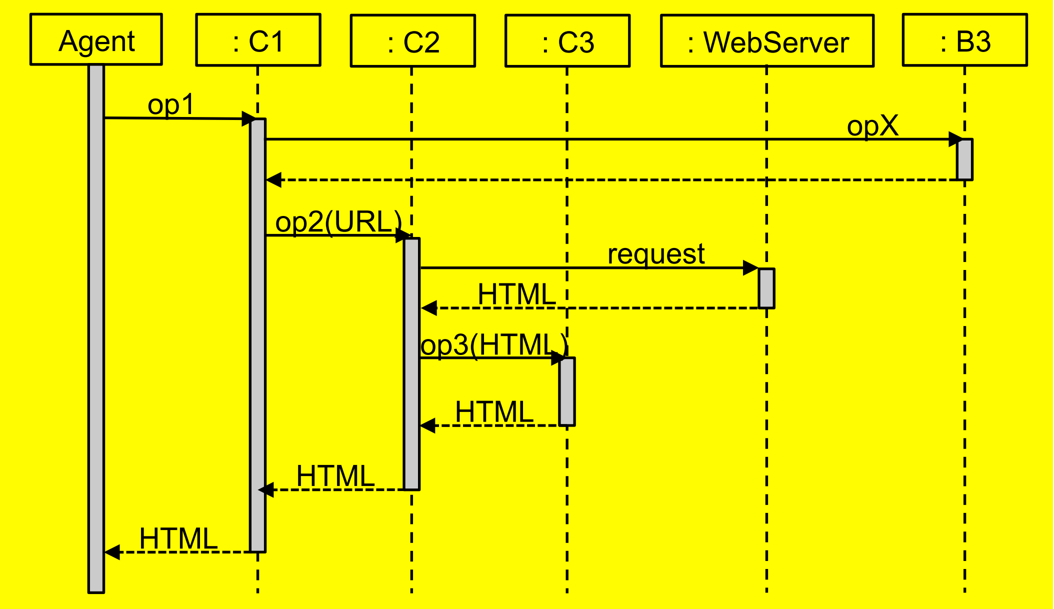
E. You want to encapsulate the code for constructing and filling in a complex composite object.

- **a. Builder**
- b. Façade
- c. Interpreter
- d. Observer

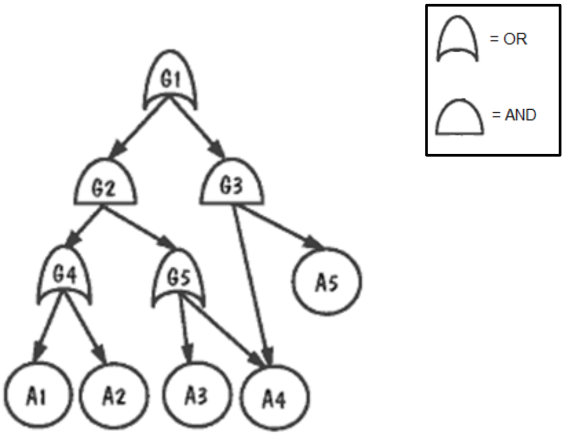
Question 2. Here is how a system implements a use case (from steps A–H):

A. An instance of class user agent is already active and invokes operation op1 on an instance of class C1.
B. The instance of C1 invokes opX on an instance B3.
C. The instance of C1 invokes op2, passing an argument URL, on an instance of class C2.
D. The instance C2 invokes operation request on the web server specified by the URL, in order to retrieve the HTML at that URL.
E. The web server returns HTML to the instance of C2.
F. Now that HTML has been retrieved from the web server, the instance of C2 invokes op3 on an instance of class C3, passing the HTML as an argument.
G. The instance C3 does some cleanup on the HTML (like removing advertisements from the HTML) and then returns the HTML to the instance of C2.
H. C2 returns the HTML to C1, and then C1 returns the HTML to the user agent.

Complete the following message sequence diagram showing these events. Remember the life lines.



Question 3. The CIA wants you to ensure that it is impossible for a terrorist to plant false information into the system. So you do a fault tree analysis. Here is the tree that you draw (where event G1 means that the terrorists successfully plant false information)…

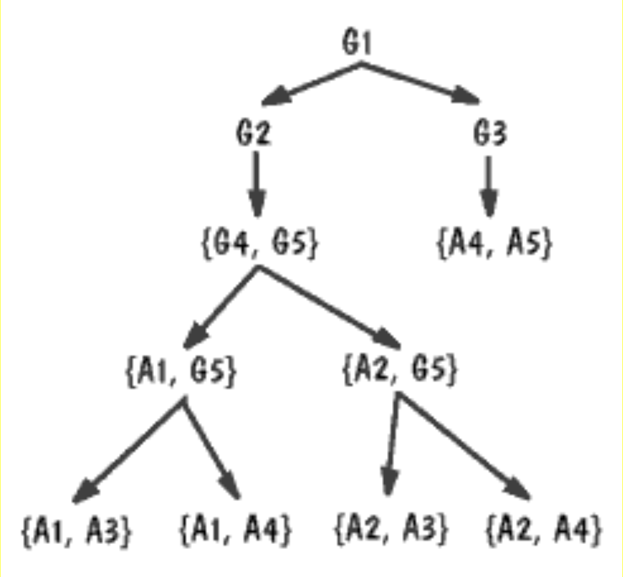


A. If event A1 occurs (but A2, A3, A4, and A5 do not), will event G1 occur?
   a. Yes
   b. No

B. If events A1, A3, and A5 occur (but A2 and A4 do not), will event G1 occur?
   a. Yes
   b. No

C. If event A4 occurs (but A1, A2, A3, and A5 do not), will event G1 occur?`
   a. Yes
   b. No

D. Use the fault tree on the previous page to draw a corresponding cut-set tree.



E. Suppose that it would cost $1 million to prevent event A1, $2 million to prevent event A2, $3 million to prevent event A3, $4 million to prevent event A4, and $5 million to prevent event A5. What is the minimum amount of money required to prevent G1?

$7 million (by preventing A1, A2, and A4, or by preventing A3 and A4)

Question 4. The table below describes the complexity of each item in this implementation. (Note that Classes D1, D2, D3, and D4 are together just one single 3GL component, so the system has a total of four 3GL components.)

Component	**Type**
A	3GL component
B1	Screen, simple complexity
B2	Screen, simple complexity
Z	3GL component
D1–D4	3GL component
X	3GL component
M	Report, difficult complexity

A. How many application points does this system have, in total?

10×4 + 1×2 + 8×1 = 50 a.p.

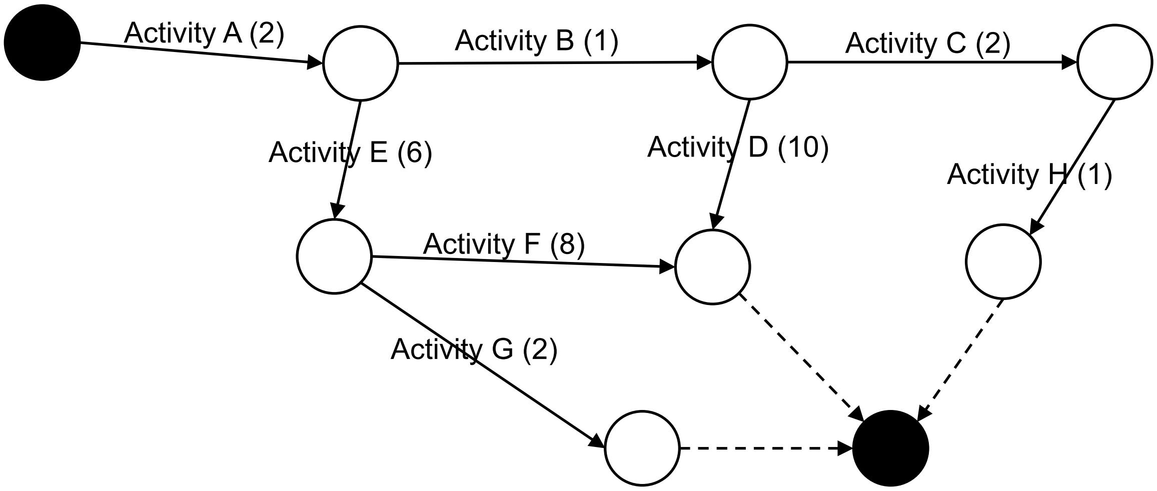
B. Suppose that your team has nominal experience and capability with creating this kind of system, and your team has very low CASE maturity and capability. What is the expected productivity of each team member, in application points per month?

(13 + 4)/2 = 8.5 a.p./mo.

C. How many person-months would this system take to implement?

50/8.5 = 5.88 person-months

Question 5. Suppose that your team builds the system by performing the activities shown in the graph below. All estimates of effort are shown in person-weeks.



A. What activities are on the critical path?

A → E → F (followed by null activity)

B. What is the slack time for Activity G?

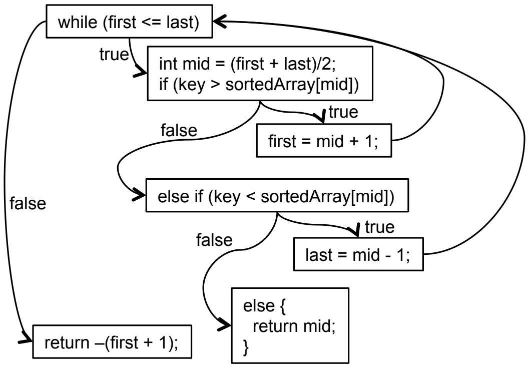
The earliest that G can start is A + E = 2 + 6 = 8. But it doesn't need to finish until simultaneously with the end of F, so the latest start is A + E + F – G = 2 + 6 + 8 – 2 = 14. The difference is the slack: 14 – 8 = 6 person-weeks.

C. What is the length of the critical path, in weeks?

A + E + F = 2 + 6 + 8 = 16

Question 6. Consider the following function and associated control flow graph.

```
int binarySearch(int sortedArray[], int first, int last, int key)
{
  while (first <= last) {
    int mid = (first + last)/2;
    if (key > sortedArray[mid]) {
      first = mid + 1;
    }
    else if (key < sortedArray[mid]) {
      last = mid - 1;
    }
    else {
      return mid;
    }
  }
  return -(first + 1);
}
```



A. What inputs would provide statement coverage?

Test case 1: sortedArray={}, first=1, last=0, key=0
Test case 2: sortedArray={1,2,3,4,5,6,7}, first=0, last=6, key=5

B. Decision coverage?

The test cases from A work.

C. Path coverage?

For 0 or 1 loop iterations:
Test case 1: sortedArray={}, first=1, last=0, key=0
Test case 2: sortedArray={8}, first=0, last=0, key=9
Test case 3: sortedArray={8}, first=0, last=0, key=7
Test case 4: sortedArray={8}, first=0, last=0, key=8

D. Mutation coverage with mutation operators that replace comparison operators with “not” versions (e.g., <= becomes >)?

Mutated operator			**Test case**	**Expected**	**Actual**
<=	**>**	**<**			
1	**1**	**1**	**sortedArray={8}, first=0, last=0, key=8**	**0**	**-1**
1	**1**	**0**	“	“	“
1	**0**	**1**	“	“	“
1	**0**	**0**	“	“	“
0	**1**	**1**	“	“	**-2**
0	**1**	**0**	“	“	**-2**
0	**0**	**1**	“	“	**-1**

Question 7. Which of the following are bad code smells (circle all that apply)?

a. **Method that is ≥ 1 screen in length**

b. **Chunks of code that appear in ≥ 3 places**

c. An "if...else if...else" statement with ≥ 5 "else if"s

d. **Class that has ≥ 7 member variables and/or ≥ 50 methods**

e. **Method that has a long list of parameters**

Question 8. Define a contract (i.e., preconditions, postconditions, and invariants) for the following class.

Class PriorityQueue

A. Invariants:

Inv: highest priority item is at the front of the queue (or the queue is empty).

B. enqueue(item : Item)

Pre: true

Post: item has been added to the priority queue.

C. dequeue()

Pre: size of queue > 0.

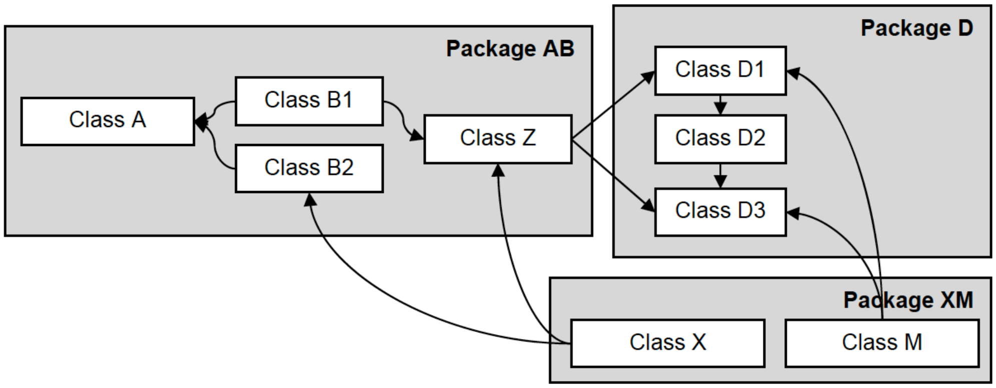
Post: The highest priority item is removed from the queue.

D. getFront() : Item

Pre: size of queue > 0.

Post: result is the highest priority item in the queue.

Question 9. Consider the following design.



- Package AB
  - The concern of Package AB is to provide a user interface.
  - Inside Package AB, classes A, B1, B2, and Z all work together to present the screen that a user sees.
- Package D
  - The concern of Package D is to perform database search.
  - Inside Package D, Class D1 tells D2 to run, which tells D3 to run; these constitute the three steps of a process.
- Package XM
  - The concern of Package XM is to do some automated testing and report generation.
  - Inside package XM, either Class X's code is run or Class M's code is run.
  -

A. Sort the packages from most to least cohesive. Justify your answer.

Most-to-least cohesive: AB, D, XM

Justification: Based on the description, AB appears to be functionally cohesive, D appears to be procedurally cohesive, and XM appears to be coincidentally cohesive.

B. Class X instantiates Class B2. It modifies this instance of Class B2 to fill it in with some information. It then instantiates Class Z. It calls a method of Class Z. When it does this, it passes the instance of B2 that it had previously instantiated.

Would it be good or bad to move Class X into Package AB? Why?

It could be good to move X into AB because it would reduce coupling between packages AB and XM. Because X "modifies" B2 by "filling" it with information, there is content coupling (bad!) between AB and XM.