Name______________________________

Com S 362
Spring 2002

Object-Oriented Analysis and Design

Final Exam: Elaboration and Design Patterns

This test has 4 questions and pages numbered 1 through 9.

Exam Process

Question 1 can be done at any time, and should be turned in at the end of the test along with all of the front matter in the test.

Starting with question 2 on this exam, each question builds on the answer from the previous question. To aid grading and to prevent you from getting too far off track, when you complete an answer for one question, you will trade your answer to that question for our standard solution for that question. You should then use our standard solution when answering the next question. For example, when you finish question 2, you will trade your answer for that question in for our standard solution to question 2, and you will then use that to solve question 3. *For these questions, be sure to put your name on each page you hand in!*

There is some material about the domain, a use case, and a system sequence diagram on the pages following the first question.

Reminders

This test is open book and notes. However, it is to be done individually and you are not to exchange or share materials with other students during the test. So if you have materials on your team project you wish to refer to during the test, please make copies.

If you need more space, use the back of a page. Note when you do that on the front.

This test is timed. We will not grade your test if you try to take more than the time allowed. Therefore, before you begin, please take a moment to look over the entire test so that you can budget your time.

For diagrams and programs, clarity is important; if your diagrams or programs are sloppy and hard to read, you will lose points. Correct syntax also makes some difference.

1. (20 points) This is a problem about design patterns. For each part, write down the name of the design pattern or principle that would be most useful for addressing the situation described.
   a. You are building a persistence framework for the service layer of a system. In this layer, operations such as changing or deleting objects have to be queued for later execution, and it is also necessary to support the ability to rollback (undo) these operations. What pattern would best support this?

   b. You are building a system that relies on a complex algorithm, and that algorithm may be changed often due to marketing pressures. What pattern would best support this?

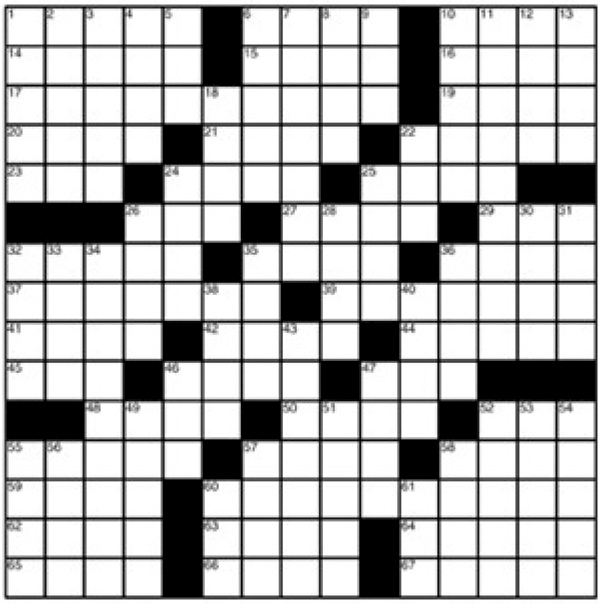
   c. You are building a system in which you want to separate the domain logic layer from the user interface layer. In particular, you want to allow changes in the state of the domain logic layer objects to be reflected in changes to one or more objects in the user interface layer. The domain logic layer shouldn't know about the objects in the user interface layer, or even how many there are. What pattern would best support this?

   d. You find in coding that you have several classes, each of which implements the same interface, and that there is a lot of duplicated code among these classes for a method, call it *m*. You would like to avoid this code duplication and make it easier to add new, similar subclasses in the future that share the common features of method *m*. What pattern would best support this?

A brief explanation of the domain for the following problems

The following is an example of a crossword puzzle. Note the two sets of clues, for words going across and those going down. In each square that is not black, a letter can be placed, which must fit with both words (one across and one down) that use that square.

USENIX Crossword
Volume 1, Number 1



Across
1. Great software property
6. Splotch
10. Surrounds Dennis's head
14. First test
15. On or in successor
16. USENIX OS
17. Bad sysadmin property
19. God doesn't play these!
20. Appear
21. Encrust
22. Pointed
23. Stray
24. Better test
25. Upon
26. Crutch
27. Telomere locale
29. Moray
32. Burn
35. Deed
36. Lazy
37. Georgia city
39. Retribution
41. Sinister
42. Italian desserts
44. Assistants
45. TV ____
46. Wagers
47. Free TV enablers
48. Utterance
50. Taxis
52. Slow x86 pointer type
55. Incense home
57. The name of the game
58. Too
59. Database operation
60. Why the stock market works
62. Follicle folly
63. Plane downers
64. Dear
65. Manipulates dishonestly
66. Pillow places
67. Network servers

Down
1. Poker action
2. Glue magnate
3. Meth
4. Hoax
5. Corrode
6. Lamb plea
7. Way of being united
8. Former
9. Busy one
10. Low frequency signal
11. Not blocked
12. A staple
13. Released
18. Chilled
22. Greek goddess that fostered craziness
24. Offers
25. Pot
26. Pickling substance
28. Corn units
30. If then ____
31. Replacement for more
32. Loses resiliency
33. Remedy
34. Stirring up
35. Having one of these is not an art
36. Bulb flower
38. Cliche dad presents
40. Progenitors
43. Evaded
46. Fu ender
47. Encourage
49. Ocean routes
51. Faulty
52. Bugs
53. Resource
54. Frolics
55. Doors are sometimes ____
56. Places
57. Front
58. Singing voice
60. Taxi
61. Fraxinus tree

Use Case for the following problems

Use Case : Layout Crossword Puzzle

Primary Actor: Author
Stakeholders and Interests:
- Author: wants non-tedious way to create crossword puzzles, automatic numbering, and spell checking.

Success Guarantee (Postconditions): A puzzle with clues is saved.
Main Success Scenario (or Basic Flow):

1. Author enters the number of rows and columns in the desired crossword puzzle.
2. Author enters a word for a particular direction (across or down) and starting square.
3. System places a black square at the end of the word (the next horizontal square in the same row if the word goes across, or the next vertical square in the same column if the word goes down) if the word does not end at the puzzle boundary.
4. Author enters the clue for that word.

   Author repeats steps 2-4 until Author indicates done.
5. System places a black square in any remaining squares without letters.
6. Author saves the completed puzzle.

Extensions (or Alternative Flows):
*a. At any time
  1. Author may request to save the current puzzle.
  2. Author may abandon work on the current puzzle and start anew.
  3. Author may change a word and its clue.
  4. Author may delete a word and its clue.
  5. Author can print the puzzle.

1a. Invalid (0 or negative) number entered for row or columns
  1. System signals an error.
  2. Author enters corrected number of rows and columns.

2a. The letters of the word don't match with one or more letters already assigned to the squares it crosses.
  1. System signals an error, and indicates which letters did not match.
  2. System rejects the word.

2b. The word goes beyond the boundary or across at least one black square
  1. System signals an error, and indicates which letters are in error.
  2. System rejects the word.

2c. The word is misspelled
  1. System suggests various corrections.
  2. Author either accepts a correction or indicates that the word should be considered correctly spelled.

2d. The word would cause a black square to be placed (see step 3) where there are already letters assigned for some other word. (I.e., the word is too short.)
  1. System signals an error, and indicates which squares need letters.
  2. System rejects the word.

2e. The black square that the system would place at the end of the word (see step 3) would violate a design rule.
  1. System signals an error, and explains how the location violates the design rule.
  2. System rejects the word.

4a. The clue is empty.
  1. System signals an error.
  2. Author enters a non-empty clue.

4b. A word in the clue is misspelled
  1. System suggests various corrections.
  2. System rejects the word.

6a. The puzzle cannot be saved.
  1. System signals an error.

Special Requirements

- Display must support basic graphics.
- Interface with external spell checkers

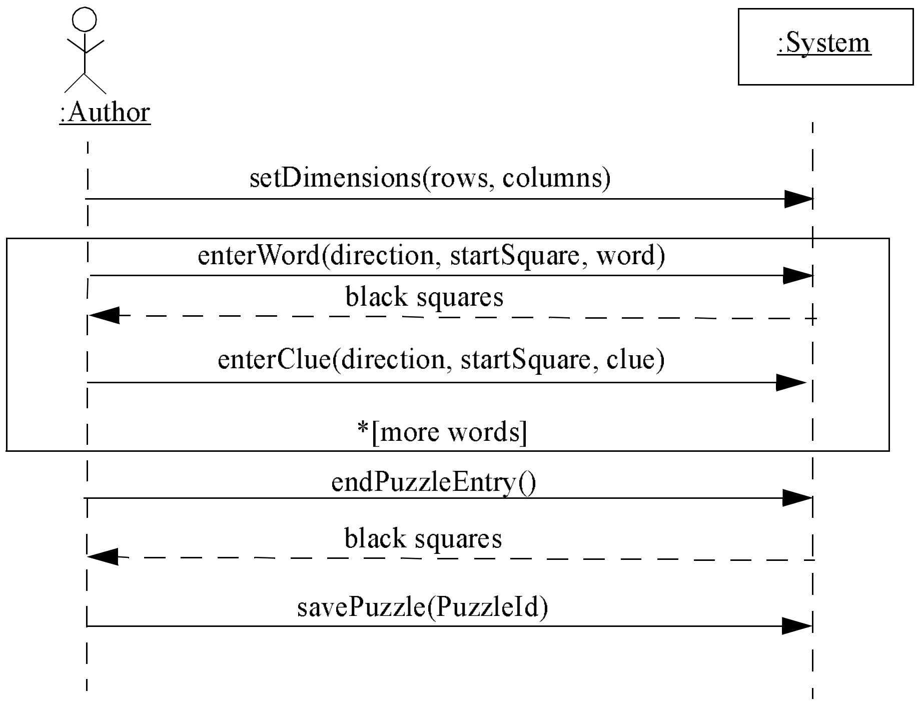
Frequency of Occurrence: one time only per run of the program.

Open Issues:

- More flexible spelling correction.
- Design layout rules.
- Internationalization.

System Sequence Diagram

The following is a system sequence diagram for the above use case.


:Author
:System
setDimensions(rows, columns)
enterWord(direction, startSquare, word)
black squares
enterClue(direction, startSquare, clue)
*[more words]
endPuzzleEntry()
black squares
savePuzzle(PuzzleId)

Name____________________________

2. (25 points) In this problem you will write a domain model, i.e., a conceptual class diagram with associations (and any attributes you feel are useful) for the application logic layer. Consider the entire layout crossword puzzle use case. However, do not include conceptual classes for external systems or for classes that would be expected to be in the programming language.

   Hint: a crossword puzzle has two lists of clues, one for the words going across and one for the words going down. You should think about how the conceptual classes allow one to represent all of the information needed to deal with each of the parts of the use case, along with each of the alternative scenarios.

Name____________________________

3. In this problem you will do the design for two of the system operations from the system sequence diagram we have given you.Your design should be based on the standard solution for the previous problem. Complete both parts (a) and (b) of this problem before turning in your solution for this problem.

   Your designs are to be recorded using interaction diagrams. You may use either the UML sequence diagram or collaboration diagram notation.

   a. (10 points) Write an interaction diagram for the `setDimensions` system operation. (For this part of the problem, you *don't* have to say what design patterns or principles are used to assign responsibilities.)

Name____________________________

b. (20 points) Write an interaction diagram, written either in the UML sequence diagram or collaboration diagram notation, for the `enterWord` system operation. (See the system sequence diagram we have given you.) Your design should be based on the standard solution you received for problem 2.

Use dog-eared commentary boxes to indicate the GRASP/GoF design pattern or principle that justifies the assignment of responsibility where appropriate. This is to be done as in the homework.

Name____________________________

4. (25 points) Based on the standard solution for the previous problems, draw a UML design class diagram for the implementation of the two system operations `setDimensions` and `enterWord`. (That is, you *don't* have to include classes, attributes, or methods needed to implement other system operations. In particular, you don't have to deal with classes for entering clues.) Include types for all method arguments and results (for methods that have results).