University of Washington
CSE 331 Software Design & Implementation
Winter 2011

Final exam

Wednesday, March 16, 2011

Name: _____

CSE Net ID (username): _____

UW Net ID (username): _____

This exam is closed book, closed notes. You have **110 minutes** to complete it. It contains 38 questions and 15 pages (including this one), totaling 220 points. Before you start, please check your copy to make sure it is complete. Turn in all pages, together, when you are finished. **Write your initials on the top of *ALL* pages.**

Please write neatly; we cannot give credit for what we cannot read.

Good luck!

Page	Max	Score
2	24	
3	28	
4	28	
5	30	
6	26	
7	14	
8	6	
10	18	
11	10	
12	8	
13	12	
14	20	
Total	224	

1 True/False

(2 points each) Circle the correct answer. T is true, F is false.

1. **T / F** When reasoning about uses of an ADT, it is permissible to ignore the code.
2. **T / F** When reasoning about uses of an ADT, it is permissible to ignore observers.
3. **T / F** Given two MDDs (module dependency diagrams), the one with fewer edges is better.
4. **T / F** `List<Integer>` is a true subtype of `List<Number>`.
5. **T / F** Regular specification fields are used for values represented by a concrete field in the implementation, and derived specification fields are used for values with no representation (as a concrete field) in the implementation.
6. **T / F** For each concrete field in the implementation, there should be a specification field (of some variety) in the abstraction.

2 Multiple choice

Mark each of the following that can be true.

7. (3 points) When is overloading resolved?
   (a) at compile time
   (b) at run time
   (c) none of the above
8. (3 points) Overloading is resolved based on what information?
   (a) declared types
   (b) run-time types (object classes)
   (c) none of the above
9. (3 points) When is overriding resolved?
   (a) at compile time
   (b) at run time
   (c) none of the above
10. (3 points) Overriding is resolved based on what information?
    (a) declared types
    (b) run-time types (object classes)
    (c) none of the above

3 Short answer

11. (3 points) The representation invariant maps _____ to _____. (Fill in the blanks.)

12. (3 points) The abstraction function maps _____ to _____. (Fill in the blanks.)

13. (10 points) In each of the following situations, which is better, a checked or an unchecked exception?

    (a) `Map.getExisting` when the key is not in the map _____

    (b) `Rational.divide` when the key is not in the map _____

    (c) `Object.clone` when the system is out of memory _____

    (d) `File.load` when the file does not exist _____

    (e) `File.load` when there is a disk failure _____

14. (12 points) In one sentence each, give three distinct reasons that you might *not* want to check the rep invariant at the entry and/or exit of a given method. Consider only instance methods that are not a constructor.

    (a) _____

    _____

    _____

    (b) _____

    _____

    _____

    (c) _____

    _____

    _____

15. (12 points) Suppose that you have observed an error in an execution of your program. Give three ways to localize the defect to a small part of the program *while debugging*, and explain (in 1 phrase or sentence each). Give ways that are as different from one another as possible.

    - _____
      _____
    - _____
      _____
    - _____
      _____

16. (10 points) In one sentence each, give an advantage of debugging using a debugger, and an advantage of debugging using print statements. If there are multiple answers, choose the best or most important one. Assume the bug is not timing-related. (Hint: don't give the answer, "you have to learn to use the debugger"; assume you are already proficient with it.)

    Advantage of using a debugger: _____
    _____
    _____

    Advantage of using print statements: _____
    _____
    _____

17. (6 points) Lecture said that the set of all non-negative real numbers is not a valid choice for a decrementing function. Would a closed set of real numbers work? (An example is all numbers x such that $0 \leq x \leq 1$.) This set has a minimum, and you can compare any 2 real numbers. Answer, and give one sentence of explanation.

    _____
    _____
    _____

18. (6 points) Why did Fred Brooks call the man-month "mythical" in his eponymous book and essay? Answer in one sentence.

19. (6 points) In one sentence, what is the scientific method?

20. (6 points) Which is more important to clients of an abstraction? The abstraction function, the representation invariant, or are they of equal importance? Explain in one sentence.

21. (6 points) Which gives more flexibility to a client (a user) of an ADT: an ADT that uses checked or unchecked exceptions? Explain why, in 1 sentence.

22. (6 points) Consider two types, `String` and `@Date String`, where the latter is a String that can be interpreted as a date, such as `"July 4, 1776"` or `"3/16/2011"`. Draw the type hierarchy for these two types. The hierarchy may include a third type, if needed.

23. (6 points) Three ways for module A to depend on module B are for A to be a subclass of B, for A to have a field of type B, or for A to take an argument (or return a value) of type B. Which of these, if any, is the most severe or strongest dependence? Explain in no more than 2 sentences.

24. (12 points) Explain the difference between synchronous and asynchronous callbacks. When is each one appropriate? Give an example of each. (Use no more than one sentence for each part.)

25. (8 points) A Macintosh-style menu item (at the top of the screen, not the top of the window as in a Windows-style menu) has effectively infinite height, because it can be thought of as extending infinitely upward. This infinite size does not affect the "reaction time" part of Fitts's law, but ought to make the "movement time" part of Fitts's law nearly zero (limited only by the fastest possible muscle movement, which is very fast). Explain, in 1-2 sentences, why that is an oversimplification of the benefit.

26. (6 points) Would it be possible for an ADT to contain both producer and mutator methods? Why, or why not, would this make sense? Answer in no more than 2 sentences.

27. (8 points) Suppose that a program calls method M, which satisfies specification S. You have not examined nor verified the program's source code, but it passes an *exhaustive* test suite (the suite tests every legal input and verifies all postconditions). Suppose that you replace the program's use of M by a use of method M', which satisfies specification S', and S' is stronger than S. Does the program still pass its test suite? Explain why or why not in no more than two sentences.

4 Reasoning about code

28. (6 points)

Give the weakest precondition for the following code, with respect to the postcondition $x > y$. Assume that `p` is `boolean` and `x` and `y` are `int`.

```
p = x>y;
if (p) {
  x++;
} else {
  y = x + y;
}
```

Answer: _____

Questions 29 and 30 ask you to prove that this this routine is correct. In other words, you will prove that it terminates with a correct answer.

There is another copy of the code on page 15 that you may tear off.

```
// requires: a is non-null, non-empty, and sorted in increasing order
// requires: val is an element in a
// returns: the index of val in a
int binarySearch(int[] a, int val) {
  int min = 0;
  int max = a.length - 1
  while (min < max) {
    int mid = (min + max) / 2;
    if (val == a[mid]) {
      min = mid;
      max = mid;
    } else if (val > a[mid]) {
      min := mid + 1;
    } else { // val < a[mid]
      max := mid - 1;
    }
  }
  return max;
}
```

29. (6 points) Prove that the `binarySearch` routine (which appears on pages 9 and 15) terminates.

30. (12 points) Prove that the `binarySearch` routine (which appears on pages 9 and 15) gives a correct answer, if it terminates. This property is called "partial correctness".

31. (10 points) Consider the `PriorityQueue.peek` method:

```
class PriorityQueue<E>
  ...
  // Retrieves, but does not remove, the head of this queue,
  // or returns null if this queue is empty.
  public @Nullable E peek() { ... }
}
```

Suppose you want to use it in a fashion like this:

```
int myMethod(PriorityQueue<Date> myQueue) {
  return myQueue.peek().getMonth();
}
```

and you want a compile-time guarantee that no `NullPointerException` will be thrown. The documentation of `peek` states that an exception *might* be thrown, but you know that it will not be thrown in your circumstance.

Describe how you could augment the type system to give this guarantee. Choose two relevant methods of `PriorityQueue`, and state how the modified type-checker would treat them.

5 Code examples

32. (8 points) In Java, `Integer[]` is a subtype of `Number[]`. Assume the following declarations:

```
Number n;
Number[] na;
Integer i;
Integer[] ia;
```

Give code that passes the type-checker but is type-incorrect with respect to true subtyping. Use a maximum of 4 statements (fewer is possible and preferable).

In one sentence, what does your code do at run time?

For the next 4 questions, assume you have this code:

```
class ListNode {
  Object data;
  ListNode next;
  void m(ListNode arg) {
    ...
  }
}
```

33. (2 points) Write a method body for `m` that performs assignment but not mutation to `arg`. (2 lines of code max; the code does not have to do anything sensible, but should not use bad style.)

34. (4 points) How could you prevent such an assignment from occurring, in standard Java? What would happen if a programmer wrote, compiled, and ran such code? (1 sentence each)

35. (2 points) Write a method body for `m` that performs mutation but not assignment to `arg`. (2 lines of code max; the code does not have to do anything sensible, but should not use bad style.)

36. (4 points) How could you prevent such mutation from occurring, in standard Java? What would happen if a programmer wrote, compiled, and ran such code? (1 sentence each)

6 Design patterns

37. (10 points) Why can interning be applied only to immutable classes? (Answer in no more than 3 sentences.)

38. (10 points) The procedural pattern contains uses of `instanceof`. For example, we saw this in lecture:

```
// Example of procedural pattern:  typechecking a programming language expression
Type tcExpression(Expression e) {
  if (e instanceof PlusOp) {
    return tcPlusOp((PlusOp)e);
  } else if (e instanceof VarRef) {
    return tcVarRef((VarRef)e);
  } else ...
```

The visitor pattern achieves the same purpose, without any uses of instanceof. Why not? That is, what happened to them?

This is a duplicate of the code that appears on page 9 and is used in questions 29 and 30. You may tear it off if you find that convenient. You do not need to hand it in.

```
// requires: a is non-null, non-empty, and sorted in increasing order
// requires: val is an element in a
// returns: the index of val in a
int binarySearch(int[] a, int val) {
  int min = 0;
  int max = a.length - 1
  while (min < max) {
    int mid = (min + max) / 2;
    if (val == a[mid]) {
      min = mid;
      max = mid;
    } else if (x > a[mid]) {
      min := mid + 1;
    } else { // x < a[mid]
      max := mid - 1;
    }
  }
  return max;
}
```