Answers to Midterm questions on 10/08/2015

OAKLAND UNIVERSITY, School of Engineering and Computer Science
Object Oriented Computing II
CSE 231 002/CSE 506 002

Chapter 1:

1. Waterfall
2. Unified Modeling language
3. B
4.

   The public access modifier allows access from anywhere. The Protected access modifier allows access from within the class, or within subclasses in the same package, or within subclasses in other packages. Package access allows access from within the class or within subclasses in the same package. Finally, private access only allows access from within the class.

5. A
6. B
7. Extends (extend (without s) will not get point)
8. Import
9. D
10. A
11. False ($O(N^3)$
12. $O(N^4)$
13. $O(N^2)$
14. (6 points)
    best case: 2N
    worst case: 64N
    average: 32N
    O(N)

Chapter 2

15. Abstract Data Type or ADT

16. Application (or user or client) level. Logical (or abstract) level. Implementation (or concrete) level. Application level.

17. Precondition

18. constants (*final* variables) and *abstract* methods

19. C,B,A,D,E [B, A, D] can mix in any order, but must be in between C and E. NO partial point for this one
20. O(N)
21. A

22. It is not correct, the while loop will go infinite if it finds since it only update the found flag. There are several ways to correct it, e.g. put found as part of while termination condition (&& !found), or just do a return true when found.

Chapter 3 Stack

23.

Answer: Last-In-First-Out, stack

24.

A. **Defining the exception** – usually as a subclass of Java's ***Exception*** class
B. **Generating (raising) the exception** – by recognizing the **exceptional situation** and then using Java's ***throw*** statement to "announce" that the exception has occurred
C. **Handling the exception** – using Java's ***try – catch*** statement to discover that an exception has been thrown and then take the appropriate action

Step B

25.

3 5 4
5
16
5
1
0

26.

```
  if (!isFull())
  {
    topIndex++;
    stack[topIndex] = element;
  }
  else
    throw new StackOverflowException("Push attempted on a full
stack.");
```

Chapter 4 Recursion

27.

Answer:

1. *The Base-Case Question:* Is there a nonrecursive way out of the algorithm, and does the algorithm work correctly for this base case?
2. *The Smaller-Caller Question:* Does each recursive call to the algorithm involve a smaller case of the original problem, leading inescapably to the base case?
3. *The General-Case Question:* Assuming the recursive call(s) to the smaller case(s) works correctly, does the algorithm work correctly for the general case?

For factorial recursion: For question 1: statement at line 4 is the base case and will lead to nonrecursive way out of the algorithms; For question 2: each call to factorial use (n-1) as parameter, which will eventually lead to the base-base; For question 3: if factorial(n-1) is correct, the n*factorial(n-1) is the correct way to calculate factorial(n).

28.

When a method is invoked, it needs space to keep its parameters, its local variables, and the return address (the address in the calling code to which the computer returns when the method completes its execution). This space is called an activation record or stack frame.

A function call will push the action record of the caller to the stack and the execution will be working on the callee's action record. When a function return, the action record stack pops to the caller's action record.

Chapter 5

29. True

30. B

31. A

32. The one in the *BoundedQueueInterface* is defined as potentially throwing a *QueueOverfowException* while the one in the *UnboundedQueueInterface* is not.

33.

4 6 3

10
9
7
6

34. Implement the floating front design of array-based implementation of queue, i.e. the access to the array element wrapped around to the beginning of the array when it reach the end.