Milwaukee School of Engineering

Electrical Engineering and Computer Science Department

CS-2852 – Test 1 – Dr. Durant

Wednesday 2 April 2014

No notes, calculators, or other reference materials may be used.

Good luck!

Name: Answers

Page 2:	(15 points)	______
Page 3:	(20 points)	______
Page 4:	(15 points)	______
Page 5:	(15 points)	______
Page 6:	(35 points)	______
Total:	(100 points)	______

4:07
:28
21

15

1. (5 points) Explain a situation where using the Collection interface is preferred over the List interface, ***and*** vice versa. Recall that one key difference is that List has a get method, whereas Collection does not.

   Collection is preffered over List when you want maximum flexibility to choose the implementing type OR you want to prevent potentially inefficient methods such as LinkedList<E>.get(int) from being called.
   Vice versa: need access to derived methods such as get(), perhaps for efficiency (e.g., using ArrayList).

2. (5 points) Given that the Collection interface does not have a get method, how do you retrieve objects via it?

   Collection.iterator() gives you an Iterator.

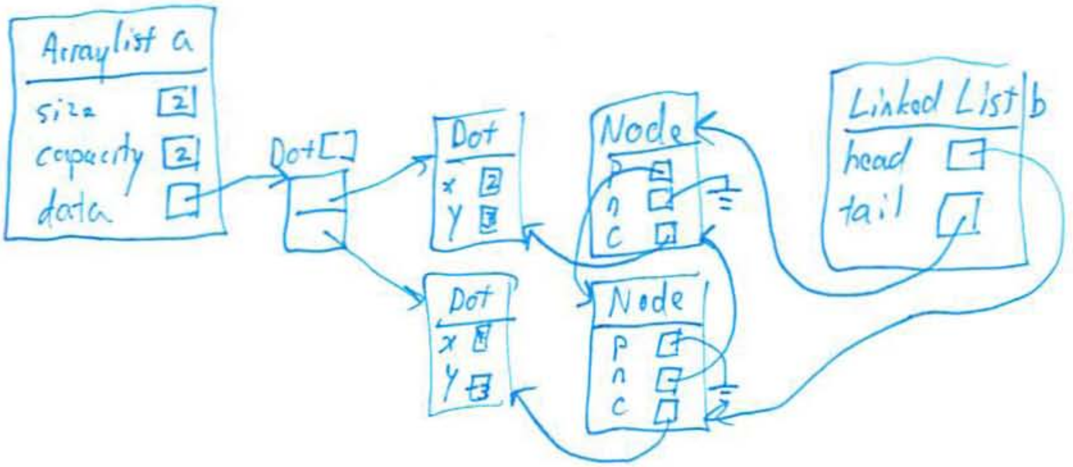
   Related: Use a foreach loop.

3. (5 points) Explain the problem that is solved by having the concepts of capacity and size in an ArrayList (instead of just size).

   Reallocating on growth is expensive, so do larger than needed growing, keeping some excess capacity on hand, so that overall growth cost is reduced.

4. (20 points) ***Draw a diagram*** illustrating an ArrayList<Dot> named "a" of size (and capacity) 2 containing two Dot objects with coordinates (2,3) and (4,-3), in that order. Add a List<Dot> (doubly linked) named "b" that contains the ***same*** Dot objects in the ***reverse*** order. Recall that lists store their object references in separate nodes.

∴ LinkedList



15

5. (10 points) Write the list of steps (not code) that the remove(int) method must take when implemented on an ArrayList. You ***do not*** need to include handling an out of bounds argument.

+ keep a reference to object to remove
- copy all higher elements, if any, 1 lower
- -- size
- set element @ size to null ← { required if removing final element. good practice regardless
+ return the kept reference

6. (5 points) Write the list of steps (not code) that the set(int pos, E) method must take when implemented on a List (doubly linked). You ***must*** include handling of an out of bounds argument.

alternative: check for >= size while walking → more efficient if size not known in advance

- throw exception if pos < 0 or pos >= size
- access the 0th node via the head reference
- walk forward a node pos times
- set ~~no~~ the node's content to the ~~1st~~ second argument

this doesn't include returning the old element → I did not deduct for missing ~~that~~

15

7. *(5 points)* A method contains an if statement whose condition is true roughly half of the time. The first (true) block has O(1) complexity, while the second (false) block has O(N) complexity. What is the overall complexity? ***Explain*** your answer.

$p \cdot 1 + (1-p) \cdot N \qquad p \doteq 0.5 \neq f(N)$

$\doteq 0.5 + \underbrace{0.5\,N}$

↑
greater growth rate
↓

$\boxed{O(N)}$

alternate: choose N
since it is worst
case (∵ conditional probability
is <u>not</u> a function of N)

8. (5 points) A method contains an O(N) operation followed by an O(N^2) operation. What is the method's complexity? ***Explain*** your answer.

$N + N^2$

↑
greater growth rate
↓

$\boxed{O(N^2)}$

9. (5 points) A method contains an O(log N) loop and the body of the loop contains an O(N) operation. What is the method's complexity? ***Explain*** your answer.

both are growing, the O(N) ~~[illegible]~~ inner steps are all run O(log N) times,
so $(\log N) \cdot N = N \log N$
↓
$O(N \log N)$

35

10. (10 points) What is an advantage ***and*** a disadvantage of using a singly linked list relative to a doubly linked list?

ad: smaller in memory / some updates simpler (add() for example)

dis: can't go backward... more complex remove logic

11. (5 points) Explain why get(int) can be O(1) on an ArrayList but at best O(N) on a LinkedList.

AL O(1): computer multiplies size of elements by index to get memory location of desired element
LL O(N): computer must walk through each of the O(n) previous node

12. (5 points) What is the purpose of the Iterable<E> interface?

It's a promise to provide an iterator through a well defined interface.

13. (5 points) What are the 3 methods of the Iterator<E> interface? (This does not include the constructor).

- .hasNext()
- .next()
- .remove()

14. (5 points) Define acceptance test.

Testing by end user to make sure software meets their needs.

15. (5 points) Define unit test.

Testing of a small # of classes (often 1) to make sure it works correctly. Often done before integration & system test.