Full Name: _____ NetID: _____

Midterm, 10/14/2014

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

Please write and/or mark your answers clearly and neatly; answers that are not readable are not considered. Please answer in appropriate details if needed and you may get partial points for the intermediate steps even if a final answer is not correct.

Each question is of 1 point, unless otherwise explicitly specified.

Chapter 1 (22 points):

1. The software life-cycle model where one carefully follows a predetermined succession of stages, one after the other, is called the _____ model.

2. UML is an acronym for _____

3. Which kind of variable holds a value that cannot be changed?

   (A) instance variable
   (B) final variable
   (C) private variable
   (D) class variable
   (E) protected variable

4. List the four types of access modifiers provided by Java and describe what each controls access to. (4 points)

5. If the Java compiler cannot find a method defined in an object's class definition it will look in the definition of the class's _____

   (A) superclass
   (B) subclass

6. Java supports double inheritance, i.e., a class can have more than one _____

   (A) superclass
   (B) subclass

7. The Java reserved word _____ indicates inheritance, i.e., we code *class subclass* _____ *superclass*.

8. To access the contents of a package from within a program, you must _____ the package into your program (answer with a Java reserved word).

9. Which structure is "first in, first out".

   (A) Array
   (B) Linked List
   (C) Stack
   (D) Queue
   (E) Tree

10. In Java, method arguments are passed by

    (A) value
    (B) reference
    (C) creating a copy of the argument and passing it
    (D) using an array
    (E) none of these

11. $3N^3 + 5N^2 + 27N + 17$ is $O(3N^3)$, True or False _____

12. What is the order of magnitude of the following function, using Big-O notation: $3N^4 + 17N^2$? _____

13. Describe the order of magnitude of the following code section using Big(O) notation. _____

```
k = 0;
for (i = 0; i < N/2; i++)
  for (j = (2 * N); j > 0; j--)
    k = k + 1;
```

14. Calculate the number of iterations in the following double-nested loop in best case complexity, worst case complexity and the average case complexity, and then describe the order of magnitude of the code using Big(O) notation. (6 points)

```
k = 0;
for (i = 0; i < N % 32; i++)
  for (j = (2 * N); j > 0; j--)
    k = k + 1;
```

Chapter 2 (16 points):

15. A data type whose properties (domain and operations) are specified independently of any particular implementation is a(n) _____ (Full name, not acronym).

16. What are the three perspectives, or levels, with which we deal with ADTs? On which level do we just need to know how to use the ADT? (4 points)

17. Assumptions that must be true upon entry into a method for it to work correctly are _____ .

18. What kinds of constructs can be declared in a Java interface? (2 points)

19. Put the following steps we follow when implementing the StringLog ADT in the order of our development (the normal steps we will do to implement any ADT): (2 points)

    (A) Implement StringLogInterface using Array
    (B) Create UseStringLog application
    (C) Define StringLogInterface
    (D) Implement StringLogInterface using LinkedList
    (E) Create test program and test cases for our implementation

20. For the *ArrayStringLog* class, assuming a list size of N, what is the Big-O complexity for contains method?

21. Testing a method by itself is called

    (A) Unit testing
    (B) Inspection
    (C) Desk checking
    (D) Black box testing
    (E) Walkthrough

22. For *contains* methods of *LinkedStringLog* implementation in the following code, is the implementation correct? If not, what is wrong with the code and modify it to make it correct. (4 point)

```
public class LLStringNode {

        private String info;
        private LLStringNode link;
         ……
}
public class LinkedStringLog {

        protected LLStringNode log;.

        public boolean contains(String element) {
                // Returns true if element is in this StringLog, otherwise false.
                LLStringNode node;
                node = log;
                boolean found = false;

                while (node != null) {
                        if (element.equalsIgnoreCase(node.getInfo()))
                                found = true;
                        else node = node.getLink();
                }

                return found;
        }
         ……..
}
```

Chapter 6 (28 points):

23. Java 5.0 introduce Generics, e.g. one way of defining a class using Generics is “public class Log<T>”, what is the best way to describe symbol T:

    (A) Object variable
    (B) Class variable
    (C) Type variable
    (D) Template
    (E) Transformer

24. Any class that implements the *Comparable* interface must provide a _____ method.

25. For the following code segment, what is the value for flag1 and flag2 and explain how you arrive your result? (2 points)

```
public class Person { …… }

Person p1 = new Person();
Person p2 = new Person();
boolean flag1 = (p1 == p2);
boolean flag2 = p1.equals(p2);
```

26. We say that a list is a collection that exhibits a linear relationship among its elements. Define linear relationship.

27. The Java _____ construct is used to create a formal specification of our List ADT.

28. The *add* method is available of each of our list varieties (unsorted, sorted and index). The add method of our sorted list uses the *compareTo* method to compare two objects – how can we be sure that the two objects on the list support the *compareTo* method? (2 point)

29. The following code segment is the implementation of the *add* method for *ArraySortedList<T>* class, please explain how this algorithm works and how the order is maintained? What is the purpose of the *while* loop and the *for* loop? (4 points)

```
public void add(T element)
// Adds element to this list.
{
  T listElement;
  int location = 0;

  if (numElements == list.length)
    enlarge();

  while (location < numElements)
  {
    listElement = (T)list[location];
    if (((Comparable)listElement).compareTo(element) < 0)
       location++;
    else
      break;
  }

  for (int index = numElements; index > location; index--)
    list[index] = list[index - 1];

  list[location] = element;
  numElements++;
}
```

30. True or False? The binary search algorithm is an efficient way for us to search our unsorted lists.

31. Briefly describe the binary search algorithm. What is the difference between binary search algorithm and regular search algorithm in terms of their complexity? (4 points)

32. Complete the following table by entering the Big-O execution time of the indicated operations when using the *RefUnsortedList* class, assuming a maximum list size of N. (4 points)

method	efficiency
add	_____
remove	_____
size	_____
contains	_____

33. True or False? A class that implements the Serializable interface must provide *read* and *write* methods for reading and writing objects of the class.

34. The following code segment is for the implementation of find method for Reference-based implementation of *List* ADT. When a call to this function returns, what is the possible values of *previous* variable (reference to which element of the list)? Please also describe the condition for each of the value. (6 points)

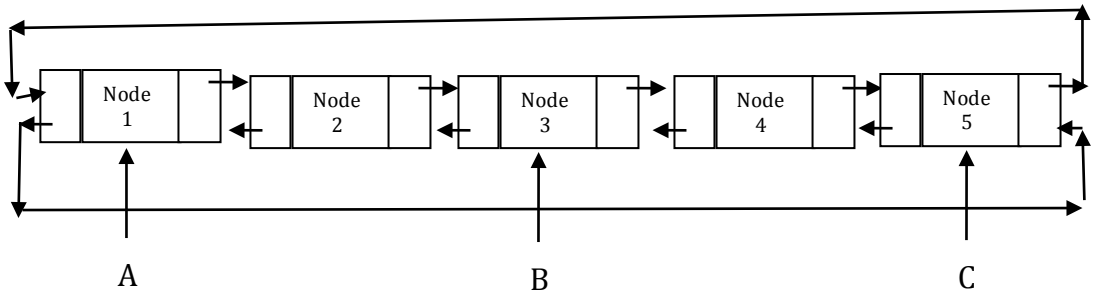
```
protected void find(T target)
{
  location = list;
  found = false;
  previous = null;

  while (location != null)
  {
    if (location.getInfo().equals(target))
    {
      found = true;
      return;
    }
    else
    {
      previous = location;
      location = location.getLink();
    }
  }
}
```

Chapter 7 (24 points):

35. Explain the difference between a linked list and a circular linked list. (2 points)

36. True or False? In our doubly linked list, the back reference of the first node on the list points to the last element on the list.

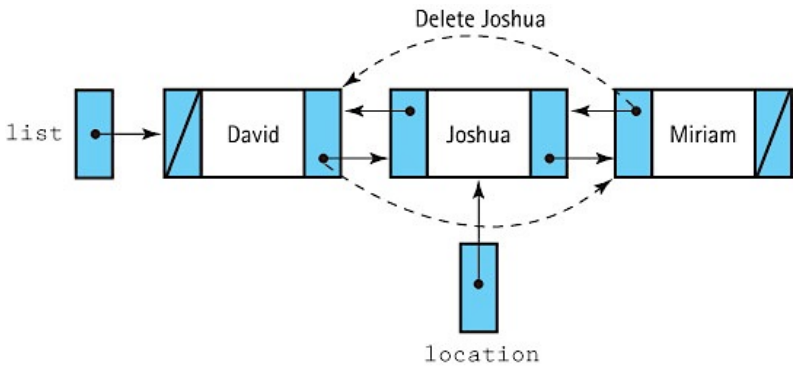
37. Give a name for the following type of list according to the way we name the different kinds of list in the textbook.



38. For the three types of lists we discussed, given a reference to a node A that belong to the list, we have two operations to find the adjacent node of node A. Fill in the table using Big-O notion of the complexity for each operation: (6 points)

List type	Find A's previous node	Find A's next node
Linked list		
Circular Linked List		
Doubly Linked list		

39. Considering removing a node from a doubly linked list as show in the following figure, please add the missing statement to finish the removing operation: (2 points)



location.back.link = location.link;

_____ = _____;

40. What is the purpose of header and trailer nodes in a list?

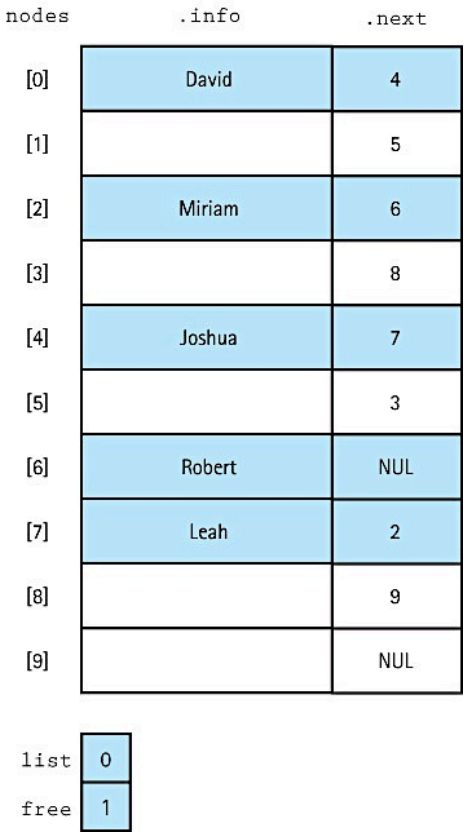
41. When implementing a linked list using an array, we use the array _____ of the “linked to” node as the value of the link.

42. What are the benefits of using array of nodes to implement a linked list? (3 points)

43. The following code is the class in the textbook to implement linked list using array of nodes, how can you modify this code to make it implement a doubly linked list? (2 points)

```
public class ArrayRefSortedStringList implements ListInterface<String>
{
  protected static final int NUL = -1;   // End of list symbol
  protected class AListNode
  {
    private String info;        // The info in a list node
    private int next;           // A link to the next node on the list
  }

  protected AListNode[] nodes;  // Array of AListNode holds the linked list

  protected int list;           // Reference to the first node on the list
  protected int free;           // Reference to the first node on the free list

  protected int numElements;    // Number of elements in the list
  protected int currentPos;     // Current position for iteration
```

44. The following diagram shows the current state of a linked list using an array of nodes; please draw a new diagram showing the state of the list after removing element “Joshua” (5 points)



Chapter 5 (15 points):

45. True or False? The first element to be stored in a queue is also the first element removed from the queue.

46. The *QueueUnderflowException* is potentially thrown by the following queue method(s).

A. enqueue only
B. dequeue only
C. isEmpty
D. the constructors
E. enqueue and dequeue

47. The *QueueOverFlowException* is potentially thrown by the following queue method(s).

A. enqueue only
B. dequeue only
C. isEmpty
D. the constructors
E. enqueue and dequeue

48. What is the difference between the *enqueue* method defined in our *UnboundedQueueInterface* and the *enqueue* method defined in our *BoundedQueueInterface*? (2 points)

49. Show what will be written by running following segment of code, given that item1, item2, and item3 are int variables, and queue is an object that fits our abstract description of a queue. Assume that you can store and retrieve variables of type int on queue. (6 points)

```
item1 = 6;
item2 = 3;
item3 = 4;
queue.enqueue(item2);
queue.enqueue(item1);
queue.enqueue(item3);
queue.enqueue(item1 + item3);
item3 = queue.dequeue();
queue.enqueue (item3*item3);
queue.enqueue(7);
item2 = queue.dequeue();
queue.enqueue(item2);
item1 = queue.dequeue();
System.out.println(item1 + " " + item2 + " " + item3);
while (!queue.isEmpty())
{
 item1 = queue.dequeue();
 System.out.println(item1);
}
```

50. In the following enqueue method for ArrayUnbndQueue class, what is the purpose of the statement in bold font (rear = (rear + 1) % queue.length)? (4 points)

```
public void enqueue(T element)
// Adds element to the rear of this queue.
{
  if (numElements == queue.length)
    enlarge();
  rear = (rear + 1) % queue.length;
  queue[rear] = element;
  numElements = numElements + 1;
}
```

---------- The end of the Test ----------

Full Name: _____ NetID: _____