NET3004 **Carleton University**
Data Structures - Midterm Test **Winter 2010**

Name : _____

Student#: _____

NOTES
1. The exam is closed book. No calculators are allowed.
2. The paper is not to be taken from the classroom
3. PRINT NEATLY! IF I CAN'T READ IT
4. YOU WILL NOT GET THE MARKS!
5. PLEASE DO NOT ASK QUESTIONS UNLESS YOU THINK THERE IS A TYPO ON THE EXAM
6. Leave difficult questions to the end. Do the easy ones first.

DO NOT WRITE IN THIS SPACE	
1-	/5
2-	/11
3-	/13
4-	/5
5-	/10
6-	/4
	/48

Question 1. [5 marks] Java/Eclipse familiarity.

Suppose you have the following Java code.

```
1. public class Example {
2.   private int anInt=2;
3.   public static void think () {
4.     System.out.println("The value is "+anInt);
5.   }
6.   public static void main(String[] args) {
7.     think();
8.   }
9. }
```

Eclipse complains about line 4 saying '`Cannot make a static reference to the non-static field anInt`'. When you comment out line 4, everything compiles and runs fine, but doesn't print anything of course.

Describe precisely two ways to fix the problem and get the expected output: `The value is 2`

i) remove static from method think
-> instantiate example in main (Example e new Example() ..... e.think()

ii) add static to definition of anInt

Question 2. [10 marks] Time complexity, Binary search vs linear search

a) [2 mark] To use a binary search on an array of objects, what must be done to the array first?

In a sorted array, you determine the middle index and check its value. If the value is the one you are looking for you are done, otherwise it checks the low/high range depending on the value as compared to the middle.

b) [2 marks] Give the Big Oh expression for the (worst case) number of comparisons needed to search an array of n objects

i) If a binary search is used: O(log N)

ii) If a linear search is used: O(N)

c) [4 marks] If an array contains 5000 values, how many comparisons will be required in the worst case according to part b? Give a number in each case, and a *brief *explanation if you think it is necessary.

i) Binary search: O(log N) ~12/13

2^12 = 4096
2^13 = 8192

$\log_2 n$ = # that n must be raised in order to get n

2^0 = 1 $\log_2 1 = 0$
2^1 = 2 $\log_2 2 = 1$
2^2 =4

ii) Linear search: O(N) ~5000

d) [3 marks] Suppose an array contains the integer values shown below. List the values in the order they will be accessed from the array during a binary search for the value 30

index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
value	5	10	20	30	40	50	60	70	80	90	100	110	120	130	140

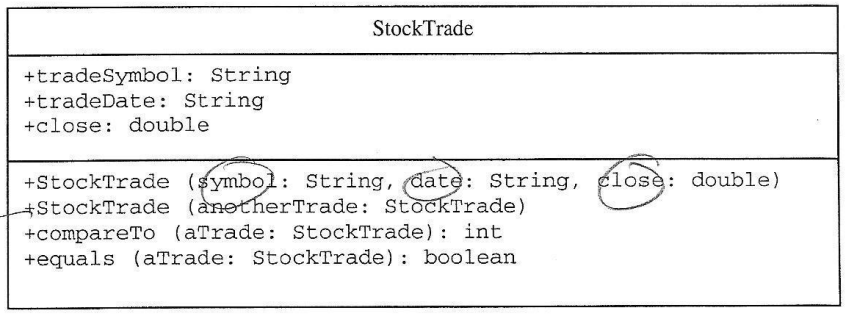
Index 0-14
mid = 6.5

30 != 60 (30 < 60)

index 0-6
mid 3
.....

Question 3. (13 marks) compareTo, equals conventions

Suppose you have the following UML for a StockTrade. NOTE IT IS DIFFERENT THAN THE ASSIGNMENT.



a) [4 marks] Write the copy constructor for `StockTrade`.

```
public StockTrade (StockTrade anotherTrade)
{
        this.symbol = tradeSymbol;
        this.date = tradeDate;
        this.close = close;
}
```

PRINT NEATLY! IF I CAN'T READ IT YOU WILL NOT GET THE MARKS!

b) [5 marks] Write Java code for the `compareTo()` method of StockTrade. Assume that StockTrades are to be ordered by `tradeSymbol+tradeDate`, similar to the assignment.

```
public int compareTo (StockTrade aTrade)
{
        String S1 = tradeSymbol + tradeDate;
        String S2 = aTrade.TradeSymbol + aTrade.tradeDate;

        return s1.compareTo(S2);
}
```

c) [4 marks] Write an `equals` method for `StockTrade`. Watch out for the common trap with comparing Strings.

```
public boolean equals ( StockTrade aTrade)
{
        String S1 = aTrade.Symbol + aTrade.date +aTrade.close;
        String S2 = this.sybom + this.date + this.close ;

        if (S1.equalTo(S2))
        return 1;
        else
        return -1;
}
```

Question 4. [5 marks] Arrays

Suppose that you are given the following UML description.

TradeArrayList
-trades:StockTrade[] // an unsorted array -actualSize: int // # StockTrades in array
+TradeArrayList (size:int) +insert(newTrade:StockTrade):int

The constructor looks like this:

```
public TradeArrayList (int size) {
      actualSize = 0;
      trades = new StockTrade[size];
}
```

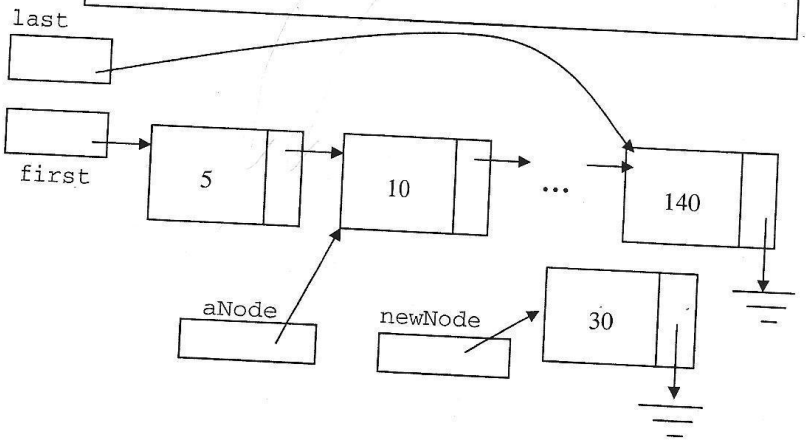
a) [5 marks] `+insert(s:StockTrade):int` inserts `s` in the first unused position (**at the end**) of the array `trades`. It returns `-1` if the new element could not be inserted, and returns the index of the newly added array element otherwise. Complete the code below.

```
public int insert ( StockTrade newTrade)
{
        if (actualSize >= trades.length)
        {
                return -1;
        else
        {
                trades[actualSize] = new StockTrade(s);
                actualSize++;
                return (actualSize-1);
        }
}
```

Question 5. [10 marks] Linked Lists

Consider a linked list of `Node` objects, where the `Node` class is defined as follows

```
public class Node {
      public int  value;
      public Node next;      //forward link
}
```



In the above, the variables `last, first, aNode, newNode` are all references to an object of type `Node`. Here is SOME of the code to create the situation in the picture.

```
Node last, first, aNode, newNode;
newNode = new Node(); newNode.value=30; newNode.next=null;
```

a) [5 marks] Write Java-like statements (as in the class notes) that insert the node `newNode` into the list AFTER the node at `aNode`. You can assume that the node `aNode` exists in the list, but you should account for the possibility that it is the last node.

```
newNode.next = aNode.next
aNode.next = newNode;
if (newNode.next == NULL)
last = newNode;

if(aNode != last)
{
        newNode.next = aNode.next;
        aNode.next = newNode;
} else {
        aNode.next = newNode;
        last = newNode;
    }
}
```

b) [5 marks] Suppose you have a recursive method (in some class whose name I forget) defined in UML syntax as follows:

```
+addup (Node start): int
```

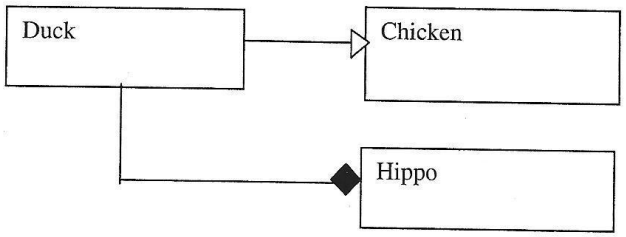
The method `addup` returns the sum of the `value` fields in the list of `Nodes` that starts at `start`. Write Java code for `addup`.

```
public int addup (node start)
{
        if (start.next == NULL)
        {
                return start.value;
        else
                return start.value + addup(start.next)
}
```

```
if (start == NULL)
{
        return 0;
else
        return start.value + addup(start.next);
}
```

Question 6. [4 marks] More on UML class diagrams

Suppose you have the following UML diagram.



Fill in the blanks on the following statements.

Hippo has-a Duck.

Duck is-a Chicken.