FINAL EXAMINATION KEY
DECEMBER 2013
CSC 212 ♦ SECTION 01
INSTRUCTOR: NICHOLAS R. HOWE

YOU MAY USE TWO 8.5"x11" SHEETS OF NOTES ON THIS EXAM.
YOU MAY NOT USE THE TEXTBOOK, A COMPUTER, OR ANY OTHER INFORMATION SOURCE BESIDES YOUR TWO PAGES OF NOTES.

All work should be written in the exam booklet. Partial credit will be granted where appropriate if intermediate steps are shown.

Vocabulary (12 points)

Give a brief yet precise definition of each of the following terms.

a.) Instance
Each time a class specification is put into use, the associated data in memory is called an instance

b.) Scope (of a variable or other symbol)
Scope refers to the extent within a program where a particular symbol has meaning and can be used

c.) Declaration
A declaration associated a symbol in the program with memory storage of a particular type

d.) Allocation
Allocation is the process of creating and initializing memory for an instance of some class

e.) Complexity (of a program)
The complexity refers to the number of operations it will take to run a program, as a function of problem size

f.) Abstract class
An abstract class is one whose methods have been specified but not completely defined

g.) Reference type
A reference type (as opposed to one of the eight primitive types) is stored by reference on the heap

h.) Recursion
Recursion is a style of programming where a method loops by calling itself repreatedly

i.) Interface
An interface defines a set of methods that a class can promise to implement

j.) Override (as a method)
When defining a subclass, if a new method is defined that has the same call signature as the some method inherited from the parent class, the parent method is said to be overridden

k.) Member (of a class)
A member is one of the elements appearing within a class definition – either a field, a method, or a nested class

l.) Overloading (regarding a method)
An overloaded method is one in which the same method name is defined more than once, with a different call signature each time

Graphs (8 points)

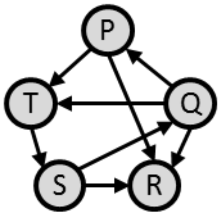
Draw the directed graphs represented by the following:

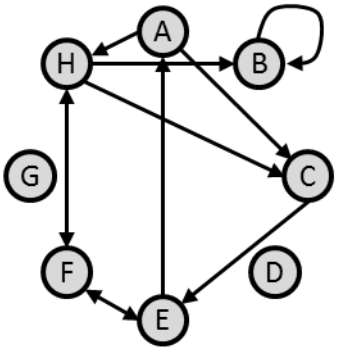
a.) Adjacency graph. Label the nodes A through H. →

b.) Edge matrix. Label the nodes P through T. →

-1	0	1	0	0
-1	0	0	0	1
1	-1	0	0	0
0	-1	1	0	0
0	-1	0	0	1
0	1	0	-1	0
0	0	1	-1	0
0	0	0	1	-1

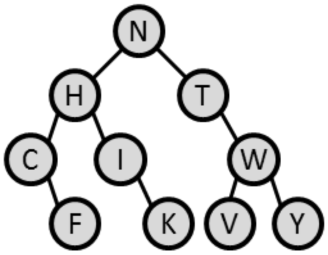
0	0	1	0	0	0	0	1
0	1	0	0	0	0	0	0
0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0
0	0	0	0	1	0	0	1
0	0	0	0	0	0	0	0
0	1	1	0	0	1	0	0





Trees (12 points)

Consider the tree at right in answering the following questions.



a.) Predict the output of the following algorithm performed on the root:

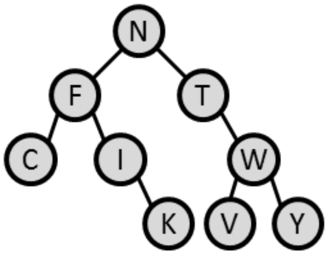
```
procedure: tour(node)
    output data
    if right not null then tour(right)
    output data
    if left not null then tour(left)
    output data
```

NTWYYYWVVVWTTNHIKKKIIHCFFFCCHN

b.) Suppose that the tree is a binary search tree using alphabetic order on the nodes. Where in the tree would the value M have to be inserted?

Right child of K

c.) Again assuming that the tree is a BST, draw what the tree would look like if H is deleted, with a copy-left protocol to fill in holes.



Recursion (12 points)

Consider the following recursive procedure for determining the winner of a sporting tournament in answering the questions that follow. Note that $\lfloor x \rfloor$ indicates a *floor* operation (round down)

```
procedure:  champion(team_0,team_1,…, team_{n-1})
    if n == 1 then
        return team_0
    else
        p = ⌊n/2⌋
        return winnerOfGame(champion(team_0, …, team_{p-1} ),champion(team_p,..., team_{n-1}))
    endif
```

a.) In a tournament with 7 teams, how many times will the *winnerOfGame* method be called?

Six times

b.) In a tournament with 12 teams, some teams will need to win three games in order to be declared champion, while others will need to win four. Which teams (e.g., *team*$_0$, etc.) will only have to play three games?

Teams 0, 3, 6, and 9 play only three games. The rest play 4.

c.) Five teams are playing a tounament: the **Holyhead Harpies**, **Wimbourne Wasps**, **Chudley Cannons**, **Tutshill Tornados**, and **Puddlemere United** (numbered in that order). Assuming that all games are won by the team whose name is first in alphabetical order, list the opponents in every game of the tournament, in the order the games would be played.

HH play WW
TT play PU
CC play PU
CC play HH

Stacks & Queues (8 points)

For each of the following, identify whether their behavior is stack-like, queue-like, or neither.

a.) A bag of candy *Neither*

b.) Students by class year at Smith (e.g., class of ’14, class of ’15, etc.) *Queue*

c.) Nodes already seen but waiting to be processed during breadth-first traversal *Queue*

d.) Method calls in Java *Stack*

e.) Coats of paint on a house *Stack*

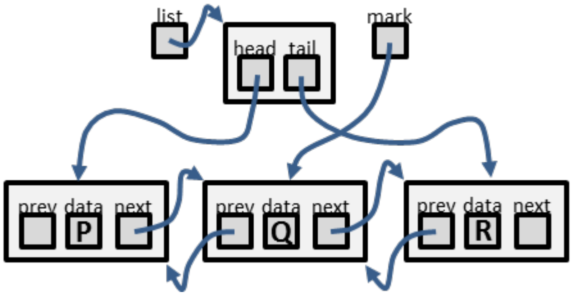
f.) Layers of skin *Queue*

g.) A heap *Neither*

h.) Cars in a parking lot *Neither*

Linked List (24 points)

Consider the diagram at right, showing a doubly linked list in memory. For each question below, assume that the starting configuration is as shown.



a.) Write a line of code to print out the data of the first element of the list, using mark as the starting point.

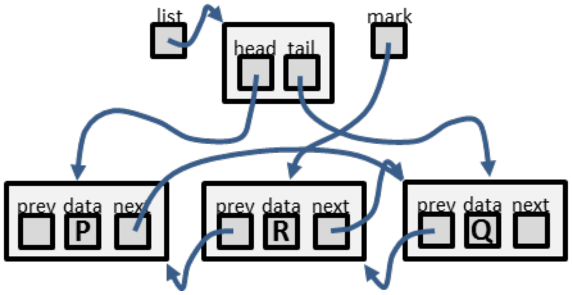
System.out.println(mark.prev.data);

b.) Write one or more lines of code to remove the first element from the list, leaving the data structure in a consistent form.

list.head.next.prev = null;
list.head = list.head.next;

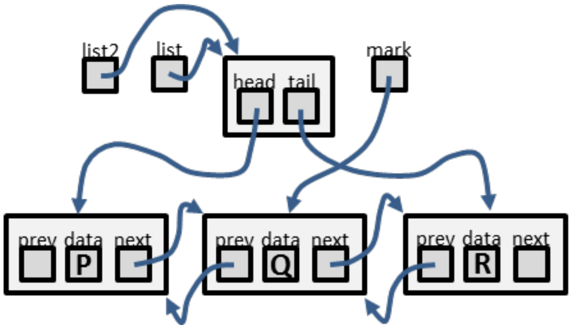
c.) Draw the memory structure that would result from execution of the following code:

```
mark.next.prev = mark.prev;
mark.next.next = mark;
mark.prev = mark.next;
mark.next = null;
list.tail = mark;
```

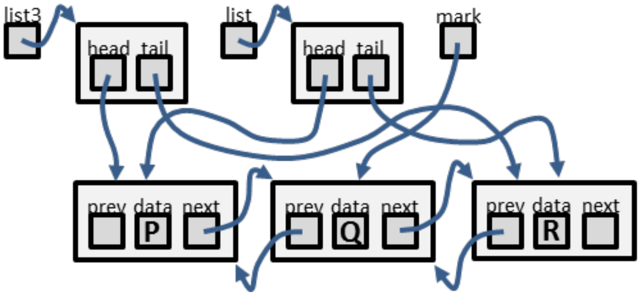


d.) Is the resulting structure from part c a consistent data structure? Describe what has happened to the list in words. *Almost, but not quite: the last two elements have swapped places, but the next link of the first element has not been updated.*

e.) Draw a diagram showing a new variable list2 that is a reference copy or alias of list.



f.) Draw a diagram showing a new variable list3 that is a shallow copy of list.



Programming Practice (8 points)

Consider the program below. Identify any Java errors or instances where the style does not meet the standards expected in thie course.

```
/** Class for the exam
 *  @author N. Howe
 *  @version December 2013
 */
public class Name {
    /** Parts of the name */
    private String firstName, lastName;
      // should have separate line & javadoc for each field

    /** Constructor for first name */
    public Name(String firstName) {
        this.firstName = firstName;
    }

    /** Constructor for last name */
    public Name(String lastName) {
        lastName = lastName;
        // as written the line above has no effect
    }
      // constructors should initialize every field, rather than one constructor per field
    /** Accessor for first name */
    public String getFirstName() {
        return this.firstName;
    }

    /** Accessor for last name */
    public String getLastName() {
        return lastName;
    }

    /** Manipulator for first name */
    public void setFirstName(String name) {
        this.firstName = name;
    }

    /** Manipulator for last name */
    public void setLastName(String name) {
        lastName = name;
    }

    /** Convert to full name
     *  @override
     */
    public String toString() {
        return firstName+" "+lastName;
    }
}
```

Program Analysis (8 points)

Consider the method below, which implements a proposed sorting algorithm.

```
public static void cocktail_sort(int[] arr) {
    for (int i = 1; i < arr.length/2; i++) {
        for (int j = i-1; j < arr.length-i; j++) {
            if (arr[j] > arr[j+1]) {
                int tmp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = tmp;
            }
        }
        for (int j = arr.length-i-1; j > i-1; j--) {
            if (arr[j] < arr[j-1]) {
                int tmp = arr[j];
                arr[j] = arr[j-1];
                arr[j-1] = tmp;
            }
        }
    }
}
```

a.) Determine the exact number of comparisons this algorithm would make, in terms of the number of elements in the array ***n***.
Outer loop (n-1)/2 times; inner loops n times on average gives n(n-1)/2

b.) How does this performance relate to other sorting algorithms we have studied?
It is O(n^2), which is similar to insertion sort and selection sort, but slower than merge sort and heap sort.

Programming Principles (8 points)

Write a few paragraphs describing what you know about the **`static`** modifier in Java. You should consider its use with both fields and methods, in terms of motivation and effects. What are the restrictions imposed, and the benefits gained, from its use? Try to be as specific as possible, preferably with examples, while considering the topic in its entirety.

Static indicates that the field or method is to be associated with the class as a whole rather than any particular instance. This means that static fields are set up before the program runs, and are essentially "shared" by all the instances of that class. Likewise, static methods do not require an instance of the class in order to run, and cannot therefore refer to any of the nonstatic fields or methods of the class (which would require such an instance).