CSIS-10B FINAL REVIEW SOLUTIONS

Closed Computer and Book
1 Double-sided sheet of notes allowed – staple to your test when done

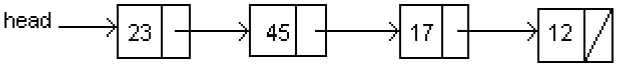
Sample Exam Problems

Problems 1 and 2 refer to the following Node class:

```
class Node {
    String data; // data field -- the data stored in this particular node
    Node next;   // next field -- reference to next Node in list, or null

         // Constructor
     Node (String data, Node next)
     {  this.data = data;
        this.next = next;
     }
```

Now suppose we have a simple linked list of nodes.



What will be printed by the following statements?

1) Given the above linked list, show the output for **each** of the following statements or indicate why an **error** will happen:

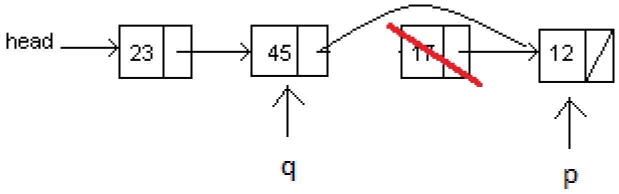
	OUTPUT
a) System.out.println(head.next.data);	a) 45

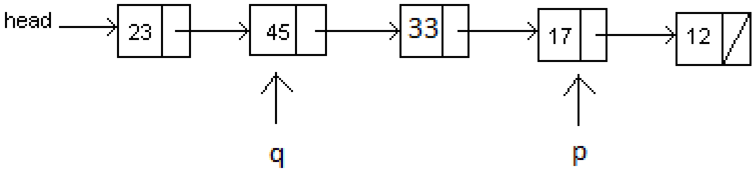
b) Node p=head.next;
p=p.next.next;
System.out.println(p.data);
	b) 12

c) Node p=head.data;
System.out.println(p.data);
	c) error, p cannot refer to String (data)

d) for (Node p=head; p!=null; p=p.next) System.out.println(p.data);	d) 23 45 17 12

2) Redraw the picture of the list above 1) after the following statements are performed. Also show the position of pointers p and q and draw an X through any deleted nodes (nodes without references that will be garbage collected):

a) Node p= head.next.next;
 	
Node q= head.next;
q.next=p.next;
p=q.next;
	Picture of list after code runs
	

b) (list reverts back to original above 1)
Node p= head.next.next;
 	
Node q= head.next;
q.next=new Node("33",p);
	

3) What will be shown if we run the following statements on our QueueLink class from lab10

```
QueueLink q = new QueueLink();
for (int k = 0; k<4; k++){
    if (k<3)
        q.enqueue(""+k);
    else
        q.enqueue( q.dequeue());
}
System.out.println("q = "+q);       // show q data from front to rear
```

<u>OUTPUT:</u> 1 2 0

4) draw the data structure you get when inserting the values "c" "x" "e" "g" "s" into a:

a) binary search tree c \\ x / e \\ g \\ s	b) a min heap (such as we did in lab) c / \\ g e / \\ x s Or in vector form: c g e x s

5. Here is a small binary tree:

```
    14
   /  \
  2    11
```

```
    / \   /
   1   3  10
          /
         7
```

Write the order of the nodes visited in:
A. An in-order traversal: 1, 2, 3, 14, 7,10,11
B. A pre-order traversal: 14,2,1,3,11,10,7
C. A post-order traversal: 1,3,2,7,10,11,14

6. Redraw the following heap after three removal operations:

```
     Original      First removal      Second             Third

         1                  3                 4                  7
      /    \             /    \             /    \             /    \
     3      4           7      4           7      13          10    13
    / \    / \         / \    / \         / \    / \         / \    /
   7  11 44  13       10  11 44  13      10  11 44  25      25  11 44
  / \                /
10  25             25
```

7. Draw a hash table of size 10 using the hash function k % 10 and linear probing to insert the keys 5, 29, 20, 11, 0, 18, 44, 27, 15, 19

cell	value
0	20
1	11
2	0
3	19
4	44
5	5
6	15
7	27
8	18
9	29

8. determine the number of items checked in order to retrieve the following keys (including the item itself) in problem 7:

5	ANS: 1
15	ANS: 2
19	ANS: 5

9. Suppose we have two algorithms, one is O(N) the other is O(M $\log_{10}$ N) which would be best to use when

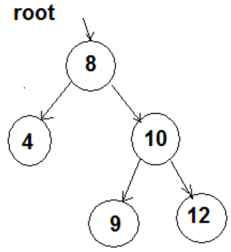
a) N=10000000, M = 20 ?	O(M log N) would be just O(140)
b) N= 10000000, M = 10000000 ?	O(N) …still bad but multiplying by 7 would be worse

10. Suppose we make a mistake in our inOrder method for the BST class:

```
public void inOrder()
{
   inOrderAux(root);
}

private void inOrderAux(TreeNode subNode)
{
   if (subNode != null)
   {
          inOrderAux(subNode.left);
          System.out.print(subNode.data + " ");
          inOrderAux(subNode.left);  ←  OOPS!  Should be right
   }
}
```

Show the program output if we invoke the buggy inOrder method from on the Binary Search Tree shown:

	4 8 4

The actual final will not allow use of a computer
It will be entirely paper based. You may find the following helpful.

11) Download and expand the zip file LinkTest.zip from the website. Open the project in BlueJ or Eclipse and solve the following problem. Suppose we need to implement a new method for the StrLinkedList class called clearToEnd(int index). The purpose of this method is to remove all the nodes in the list starting at index through to the end of the list.

For example, if a StrLinkedList test began with the values:

test:
"a"-"c"-"g"-"w"-"r"-"f"-"u"

the statement list.clearToEnd(4) would leave the list in the following state:

test:
"a"-"c"-"g"-"w"

Code this method and test it using the demo app provided.

```
public void clearToEnd(int index){
    SLLNode pred = first;
    for (int count = 0; count<index-1; count++)
        pred = pred.next;
    pred.next=null;
}
```