CSC 172 MID-TERM EXAM

March 3rd, 2011

Please write answers on the exam sheet.

Put your name and on the front (this) page.

Closed book, closed notes
Hand calculators are allowed.
Laptop computers are not allowed.

10 Questions
100 points total.

75 min

Meliora.

NAME :_____

QUESTION	GRADE	POINT VALUE
Q0 (name)		1
Q1		11
Q2		11
Q3		11
Q4		11
Q5		11
Q6		11
Q7		11
Q8		11
Q9		11
TOTAL		**100**

1. Prove the following relationship **by induction**. Be sure to explicitly state the induction parameter, the induction hypothesis and its use in the proof, and appropriately identify the salient cases.

$$\sum_{i=0}^{n-1} ar^i = \frac{(ar^n - a)}{(r-1)}$$

BASIS: n == 1

$$\sum_{i=0}^{1-1} ar^i = \sum_{i=0}^{0} ar^i = ar^0 = a = \frac{(ar^1 - a)}{(r-1)} = \frac{a(r-1)}{(r-1)}$$

Induction :

$$\sum_{i=0}^{n} ar^i = \sum_{i=0}^{n-1} ar^i + ar^n$$

BTIH

$$\sum_{i=0}^{n} ar^i = \frac{(ar^n - a)}{(r-1)} + ar^n = \frac{(ar^n - a)}{(r-1)} + \frac{ar^n(r-1)}{(r-1)} = \frac{ar^n - a + ar^{n+1} - ar^n}{(r-1)} = \frac{(ar^{n+1} - a)}{(r-1)}$$

QED

2. A standard deck of cards contains thirteen ranks of cards {2,3,4,5,6,7,8,9,10,J,Q,K,A} and four suits {C,H,D,S} for a total of fifty-two cards. A standard "hand" consists of 5 cards drawn from such a deck. A "pair" of cards is any two cards of the same rank. A hand containing "two pairs" has two such pairs of different ranks and a fifth card which is of a third rank. For example {3H,3D,QC,QD, AS}, and {AH,AD,8C,8S,JH} are both "two pair hands". Explain how many possible "two pair" hands can be made from a standard deck of cards. You do not need to give an exact number, an expression explaining your reasoning will be sufficient. You do need to give an expression and an explanation of your reasoning.

For the pairs
There are 4 ways to pick the first
There are 3 ways to pick the second
but order is not important, so (4 * 3) / 2 == 6
There are 4C2 == 6 ways to get a pair of any rank

There are 13 ways to pick the first pair
There are 12 ways to pick the 2nd pair
but order is not important, so (13 * 12) / 2 == 78 == "13 choose 2"

There are 52 – 8 == 44 remaining cards for the kicker

so (13*6*12*6*44) / 2 == 78 * 6 * 6* 44 = 123,552 possible hands
(exact number not required if the proof is sound)

3. Show (prove) that the runtime of the code below. To do this, you need two steps : (1) express the runtime as a series – relating lines in the code to expressions in the series (sigma notation), a drawing may help with this (2) prove the closed form of the series.

```
public int void foo (int n)  {          // line #1
                                        //  2
     int x = 0, k = 1;                  //  3
                                        //  4
     for (int i = 0 ; i < n ; i++) {    //  5
         for (int j = 0 ; j < k ; j++)  //  6
              x++;                      //  7
         k *= 2 ;                       //  8
     }                                  //  9
                                        // 10
     return x ;                         // 11
                                        // 12
    }                                   // 13
}                                       // 14
```

The outer loop runs n times
the first time, the inner loop runs 1 times
the 2nd time, the inner loop runs 2 times
the 3rd time, the inner loop runs 4 times
the 4th time, the inner loop runs 8 times
…

the n-th time, the inner loop runs 2^(n-1) times

$\sum_{i=0}^{n-1} 2^i = 2^n - 1$ Basis $\sum_{i=0}^{1-1} 2^i = 2^0 = 2^1 - 1$

INDUCTION

$\sum_{i=0}^{(n+1)-1} 2^i = \sum_{i=0}^{n-1} 2^i + 2^n$

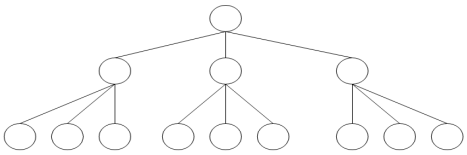
BTIH:

$\sum_{i=0}^{(n+1)-1} 2^i = 2^n - 1 + 2^n = 2^{(n+1)} - 1$

So, it is O(2^n)

(extra page for Q#4)

4. A *trinary* tree is a tree made up of nodes that can have 0,1,2, or 3 children. A ***full trinary*** tree is a fully balanced trinary tree. [i.e. All the non-leaf nodes have exactly three children and the bottom layer of leaf nodes is full.] Prove by (structural) induction that any full trinary tree of height h has $\frac{3^{h+1}-1}{2}$ nodes. (Drawing pictures helps)



make a new tree with height h+1 from three such sub-trees each of height h
(you have to add one more node for the new root)

Show that this new tree has (3^((h+1) + 1) -1)/2 nodes

BTIH : Each sub tree has (((3^(h+1)) – 1)/2) nodes, and there are three of them

3* (((3^(h+1)) – 1)/2) == (((3^(h+2)) – 3)/2)

add one node for the new root

(((3^(h+2)) – 3)/2) + (2/2) = (((3^(h+2)) – 1)/2)

5. Given the recurrence T(1) = 1, T(n) = 2T(n-1)
   Derive (prove) the closed form of T(n)

   T(n) = 2 T(n-1)
   T(n-1) = 2 T(n-2)
   T(n-2) = 2 T(n-3)

   . . . .

   T(n-(n-1)) = T(1) = 1

   T(n) = 2(2(2(. . .2(1)))))

   // we can count n-1 2's

   T(n) = 2^(n-1)

6. Show the result of inserting the following keys into an AVL tree. Assume alphabetical order. Redraw the tree after each balance operation.
{ R, S, D ,K, B, C, L, G, H, E, F, M}
{18,19, 4, 11, 2, 3, 12, 7, 8, 5, 6, 13}

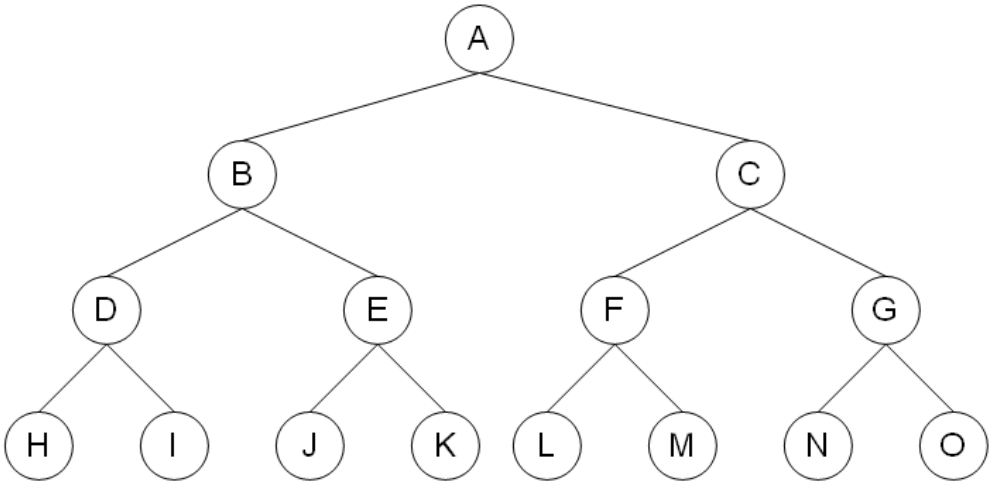
{R{D{B{}{}}{K{}{}}}{S{}{}}}
C
{R{D{B{}{C{}{}}}{K{}{}}}{S{}{}}}
rotate right
{D{B{}{C{}{}}}{R{K{G{}{}}{L{}{}}}{S{}{}}}}
H
{D{B{}{C{}{}}}{R{K{G{}{H{}{}}}{L{}{}}}{S{}{}}}}
rotate right
{D{B{}{C{}{}}}{K{G{E{}{}}{H{}{}}}{R{L{}{}}{S{}{}}}}}
F
{D{B{}{C{}{}}}{K{G{E{}{F{}{}}}{H{}{}}}{R{L{}{}}{S{}{}}}}}
rotate left
{G{D{B{}{C{}{}}}{E{}{F{}{}}}{K{H{}{}}{R{L{}{}}{S{}{}}}}}
M
{G{D{B{}{C{}{}}}{E{}{F{}{}}}{K{H{}{}}{R{L{}{}}{S{}{}}}}}
rotate left
{G{D{B{}{C{}{}}}{E{}{F{}{}}}{L{K{H{}{}}{}}{R{M{}{}}{S{}
{}}}}}

pictures help

(extra sheet)

7. Write the sequence in which the nodes of the following tree would be visited using a PreOrder, InOrder, and PostOrder traversal.

PostOrder : HIDJKEBLMFNOGCA



InOrder : HDIBJEKALFMCNGO

PreOrder : ABDHIEJKCFLMGNO

8. What is the output of the following code ? (It compiles and runs.)

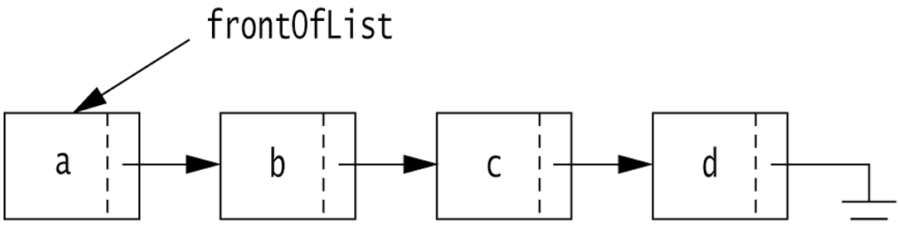
```
#include <stdio.h>
#include <stdlib.h>

struct nlst { struct nlst * nxt ; int val ; };

#define HSZ 9
static struct nlst *htab[HSZ];
unsigned hsh(int i) {  return i % HSZ ; }

void ajou(int s) {
  struct nlst * np;
  unsigned hval;

    np = (struct nlst *) malloc (sizeof(struct nlst));
    hval = hsh(s);
    np->nxt = htab[hval];
    np->val = s;
    htab[hval] = np;
}

void plst(struct nlst * start) {
  struct nlst *np;
  for (np = start; np != NULL; np = np->nxt)
    printf(" -> %d ",np->val );
}

void phtb() {
  int i;
  for (i = 0 ; i < HSZ; i++){
      printf("%d : ",i);
      plst(htab[i]);
      printf("\n");
  }
}

int main(int argc, char * argv[]) {
  int i;
  for (i = 0 ; i < 30; i+=3) ajou(i);
  phtb();
}
```

0 : -> **27** -> **18** -> **9** -> **0**
1 :
2 :
3 : -> **21** -> **12** -> **3**
4 :
5 :
6 : -> **24** -> **15** -> **6**
7 :
8 :

(extra sheet)

9. Assume a “bare bones” implementation of a linked list

```
class Node {
     Object data;
     Node next;
}
```



Write a JAVA method to remove the second element of the list (the “b” element in the figure above). If there is no first element or not second element, the method should do nothing. Only a bare bones implementation is required, you need not use Generics or exception handling.

```
public void remove2nd(Node headOfList) {

  if (headOfList == null) return;
  if (headOfList.next == null) return;
  headOfList.next = (headOfList.next).next;

}
```

(extra sheet)