Computer Science Foundation Exam

August 8, 2008

Computer Science

Section 1A

Name: _____

PID: _____

	Max Pts	Passing Pts	Category	Score
Q1	8	6	KNW, CMP	
Q2	8	6	ANL	
Q3	9	6	CMP	
Q4	8	6	KNW	
Q5	8	6	DSN	
Q6	9	6	KNW	
Total	50	36		

You have to do all the 6 problems in this section of the exam.
Partial credit cannot be given unless all work is shown and is readable.

Be complete, yet concise, and above all <u>be neat</u>.

1. **[8 pts]** Show all your work and indicate your final answer.

   a) **[4 pts]** Determine the value of ***sum*** in terms of ***N***, when the following code segment is executed:

```
sum = 0;
for(i=1; i <= N*N; i++){
   for(j=1; j <= N-5; j++) {
         sum = sum + 3 + 4*j;
   }
}
```

$$sum = \sum_{i=1}^{n^2}\sum_{j=1}^{n-5}(3+4j) \qquad = \sum_{i=1}^{n^2}\left(3n-15+\frac{4(n-5)(n-4)}{2}\right)$$

(1 point) (1 point)

$$= \sum_{i=1}^{n^2}\left(3n-15+2n^2-18n+40\right) = \sum_{i=1}^{n^2}\left(2n^2-15n+25\right)$$ (1 point)

$$= 2n^4-15n^3+25n^2$$ (1 point)

There are many equivalent ways to approach this problem. Please be understanding.

   b) **[4 pts]** Evaluate the following expression using summation rules and find the closed form in terms of *n*.

$$\sum_{i=n-10}^{n}\sum_{j=1}^{2i}5$$

$$= \sum_{i=n-10}^{n}10i = 10\sum_{i=n-10}^{n}i \qquad = 10\left(\sum_{i=1}^{n}i-\sum_{i=1}^{n-11}i\right) \qquad = 10\left(\frac{n(n+1)}{2}-\frac{(n-11)(n-10)}{2}\right)$$

(1 point) (1 point) (1 point)

$$= 5\left(n^2+n-\left(n^2-21n+110\right)\right) = 5(22n+110)$$

(1 point)

2. **[8 pts]** Answer each of the following "timing" questions concerning an algorithm of a particular order and a data set of a particular size. Assume that the run time is affected only by the size of the data set and not its composition and that *N* is an arbitrary integer. Show your work for full credit.

   a) **[4 pts]** Assume that an O($\log_2 N$) algorithm runs for 10 milliseconds when the input size (*N*) is 32. What is the size of the input that makes the algorithm run for 14 milliseconds?

$\frac{\lg N_1}{t_1} = \frac{\lg N_2}{t_2}$ $\frac{\lg 32}{10ms} = \frac{\lg N_2}{14ms}$ (1 point) $\frac{5}{10ms} = \frac{\lg N_2}{14ms}$ (1 point)

$\lg N_2 = \frac{5 * 14ms}{10ms} = 7$ (1 point) $N_2 = 2^7 = 128$ (1 point)

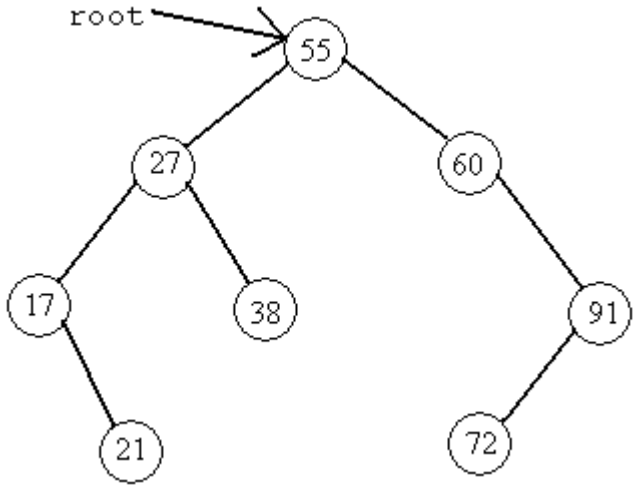
Don't deduct points if the student omits units. Solution keys should aspire to a higher standard, though.

   b) **[4 pts]** Assume that an O($N^2 \log_2 N$) algorithm runs for 6 milliseconds when the input size (*N*) is 4. How long does the algorithm run for when the input size is 8?

$\frac{N_1^2 \lg N_1}{t_1} = \frac{N_2^2 \lg N_2}{t_2}$ $\frac{4^2 \lg 4}{6ms} = \frac{8^2 \lg 8}{t_2}$ (1 point) $\frac{16 \cdot 2}{6ms} = \frac{64 \cdot 3}{t_2}$ (1 point)

$t_2 = \frac{64 \cdot 3 \cdot 6ms}{16 \cdot 2}$ (1 point) $t_2 = 36ms$ (1 point)

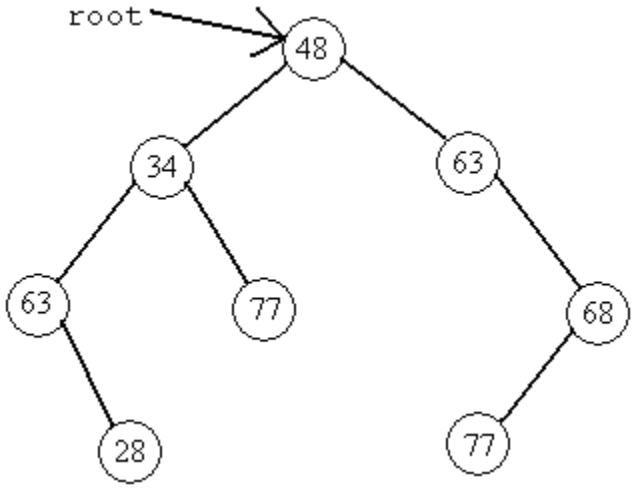
3. **[9 pts]** For the binary tree given below **`root`** is a pointer to the root of the tree.



Redraw (on the following page) the tree shown above when the following function is executed. Assume that the initial call is `modifyT(root, 7, 65)`.

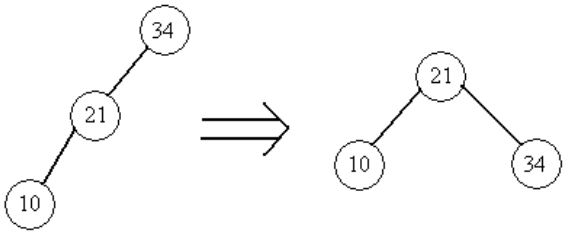
```
struct treeNode
{
   int data;
   struct treeNode *left;
   struct treeNode *right;
};

void modifyT(struct treeNode* node_ptr, int key, int num)
{
   if (node_ptr != NULL)
   {
      if (node_ptr->data % 3 == 0)
      {
         node_ptr->data += key;
         modifyT(node_ptr->left, key + 2, num - key);
         modifyT(node_ptr)->right, key - 3, num + key);
      }
      else if (node_ptr->data % 5 == 0)
      {
         node_ptr->data -= key;
         modifyT(node_ptr->right, key - 4, num);
         modifyT(node_ptr->left, key, num + 5);
      }
      else
      {
         node_ptr->data = num;
         modifyT(node_ptr->right, key - 2, num + 10);
         modifyT(node_ptr->left, key + 5, num - 7);
      }
   }
}
```

Answer for Problem 3

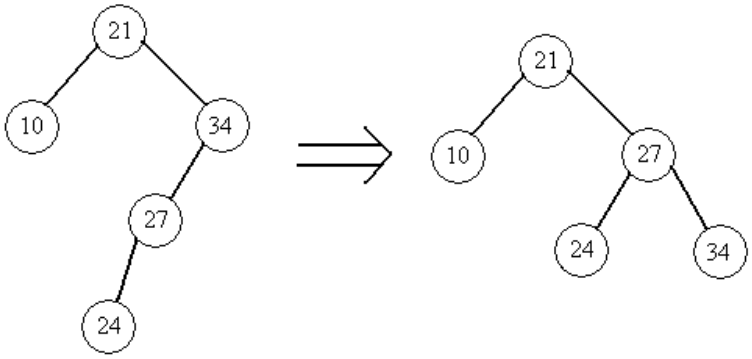


1 point for each correct node modification
1 point extra if the entire tree is correct

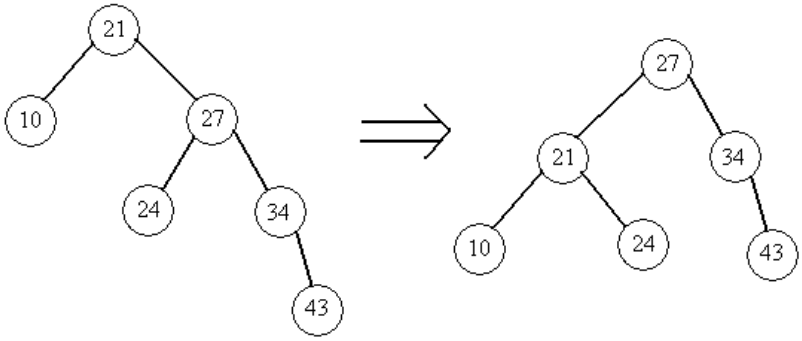
4. **[8 pts]** Insert the integers 34, 21, 10, 27, 24, 43, 15, 6 to an initially empty AVL tree in order. Draw the state of the tree before and after each necessary rotation. Be sure to draw the final state of the tree.



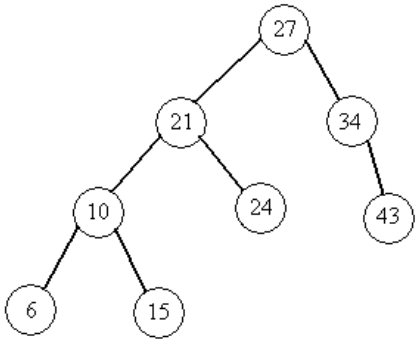
1 point 1 point



1 point 1 point



1 point 1 point



2 points for correct final state

5. **[8 pts]** Write a recursive function that compares two given binary trees. It returns 1 if two trees are different and it returns 0 otherwise. Use the node structure and the function prototype provided below:

```
struct treeNode {
      int data;
      struct treeNode * left;
      struct treeNode * right;
};

int check(struct treeNode *A, struct treeNode *B)
```

One possible solution:

```
int check(struct treeNode *A, struct treeNode *B)
{
      if(A == NULL && B == NULL)
            return 0;
      else if(A == NULL || B == NULL)
            return 1;
      else if(a->data != b->data)
            return 1;

      if(check(A->left, B->left) || check(A->right, B->right))
            return 1;
      else
            return 0;
}
```

Grading:
Base cases:
The trees are the same if both trees are NULL (1 point)
The trees are different if one is NULL, the other isn't (1 point)
The trees are different if neither is NULL, but the data differs (1 point)

Recursive cases:
The two trees are different if either the left or right is different (3 points)
The two trees are the same only if both the left and right are the same (2 points)

6. **[9 points]** Given the array [37, 21, 28, 6, 2, 23, 35, 17, 44, 4, 11, 33, 18] show the state of the array after each pass when the following sorting algorithms ((a) Insertion Sort, (b) Bubble Sort, and (c) Selection Sort) are applied on the original array for three (3) passes.

a) **[3 points]** Insertion Sort

Pass 1:
[21, 37 | 28, 6, 2, 23, 35, 17, 44, 4, 11, 33, 18]

Pass 2:
[21, 28, 37 | 6, 2, 23, 35, 17, 44, 4, 11, 33, 18]

Pass 3:
[6, 21, 28, 37 | 2, 23, 35, 17, 44, 4, 11, 33, 18]

b) **[3 points]** Bubble Sort

Pass 1:
[2 | 37, 21, 28, 6, 4, 23, 35, 17, 44, 11, 18, 33]

Pass 2:
[2, 4 | 37, 21, 28, 6, 11, 23, 35, 17, 44, 18, 33]

Pass 3:
[2, 4, 6 | 37, 21, 28, 11, 17, 23, 35, 18, 44, 33]

c) **[3 points]** Selection Sort

Pass 1:
[2, 21, 28, 6, 37, 23, 35, 17, 44, 4, 11, 33, 18]

Pass 2:
[2, 4, 28, 6, 37, 23, 35, 17, 44, 21, 11, 33, 18]

Pass 3:
[2, 4, 6, 28, 37, 23, 35, 17, 44, 21, 11, 33, 18]

Grading Criteria:
1 point for each correct array