Computer Science Foundation Exam

August 11, 2006

Computer Science

Section 1A

Name: _____

SSN: _____

Q1		KNW, ANL	
Q2		KNW, ANL	
Q3		CMP	
Q4		KNW	
Q5		DSN	
Q6		KNW	
Total			

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 be neat.

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^2; i++){
   for(j=1; j <= N-3; j++) {
        sum = sum + 1 + 2*j;
   }
}
```

$$sum = \sum_{i=1}^{N^2}\sum_{j=1}^{N-3}(1+2j) \quad = \sum_{i=1}^{N^2}\left(N-3+\frac{2(N-3)(N-2)}{2}\right)$$

(1 PTS) (1 PTS)

$$= \sum_{i=1}^{N^2}\left(N-3+N^2-5N+6\right) = \sum_{i=1}^{N^2}\left(N^2-4N+3\right) \quad \text{(1 PTS)}$$

$$= N^4-4N^3+3N^2 \quad \text{(1 PTS)}$$

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

$$\sum_{i=n-20}^{n}\sum_{j=1}^{2i}3 = \sum_{i=n-20}^{n}6i = 6\sum_{i=n-20}^{n}i \quad \text{(1 PTS)}$$

$$= 6\left(\sum_{i=1}^{n}i - \sum_{i=1}^{n-21}i\right) = 6\left(\frac{N(N+1)}{2} - \frac{(N-21)(N-20)}{2}\right)$$

(1 PTS) (1 PTS)

$$= 3\left(N^2+N-\left(N^2-41N+420\right)\right) = 3(42N-420) \quad \text{(1 PTS)}$$

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 12 milliseconds when the input size is 64. What is the size of the input that makes the algorithm run for 16 milliseconds?

$$\frac{\log_2 N_1}{t_1} = \frac{\log_2 N_2}{t_2} \qquad \frac{\log_2 64}{12} = \frac{\log_2 N_2}{16}$$ (1 PTS)

$$\frac{6}{12} = \frac{\log_2 N_2}{16}$$ (1 PTS) $$\log_2 N_2 = \frac{6*16}{12} = 8$$ (1 PTS)

$$N_2 = 256$$ (1 PTS)

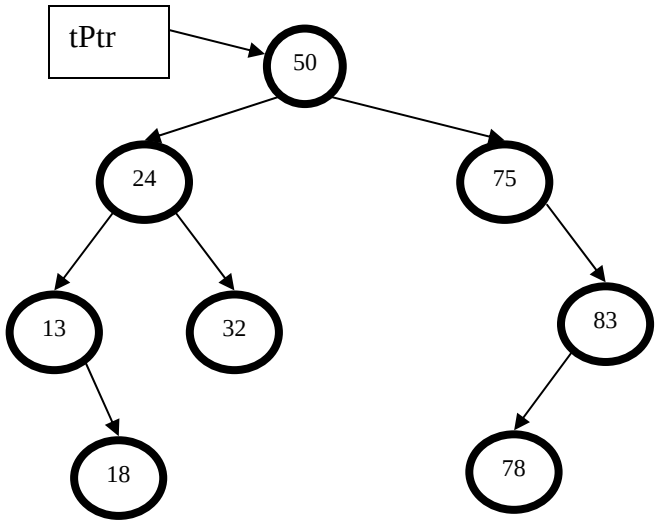
b) **[4 pts]** Assume that an $O(N^2 \log_2 N)$ algorithm runs for 7 milliseconds when the input size is 4. How long does the algorithm run for when the input size is 8?

$$\frac{N_1^2 \log_2 N_1}{t_1} = \frac{N_2^2 \log_2 N_2}{t_2} \qquad \frac{4^2 \log_2 4}{7} = \frac{8^2 \log_2 8}{t_2}$$ (1 PTS)

$$\frac{16*2}{7} = \frac{64*3}{t_2}$$ (1 PTS) $$t_2 = \frac{64*3*7}{16*2}$$

$t_2 = 42$ milliseconds (1 PTS)

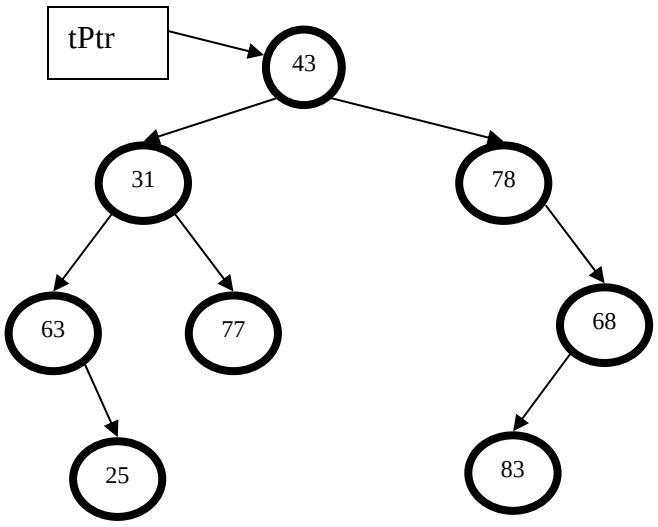
3. **[9 pts]** For the binary tree given below `tPtr` is a pointer to the root of the tree.



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

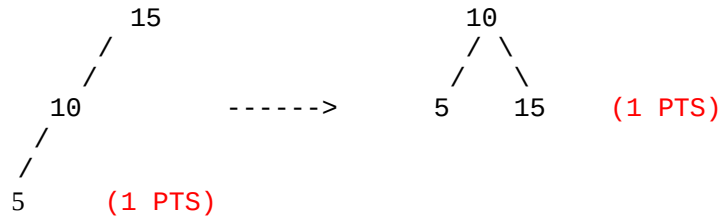
```
struct treeNode{
   int data;
   struct treeNode *left, *right;
};

void modifyT(struct treeNode **node_ptr, int key, int num){
   if (*node_ptr != NULL){
      if ((*node_ptr)->data % 3 == 0){
         (*node_ptr)->data = (*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 = (*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));
      }
   }
}
```

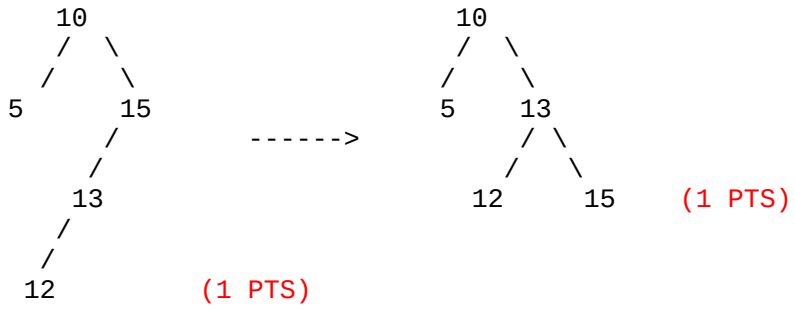
SOLUTION:

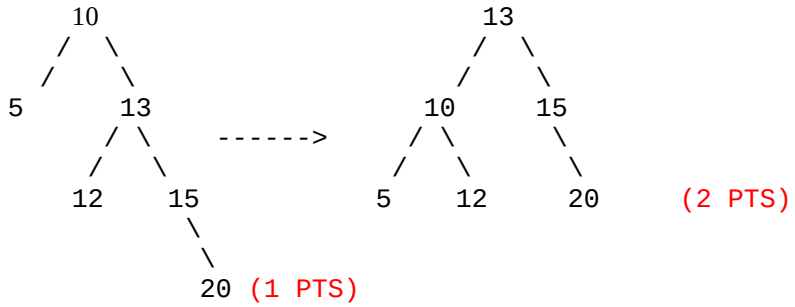


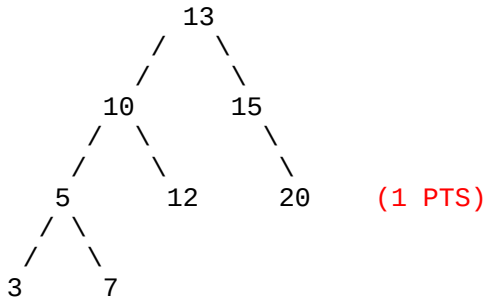
1 PTS FOR EACH CORRECT NODE MODIFICATION
1 PT extra if the entire tree is correct!

4. **[8 pts]** Insert the integers 15, 10, 5, 13, 12, 20, 7, 3, 6 to an initially empty AVL tree in order. Draw the state of the tree before and after each necessary rotation and illustrate the rotation. Draw both steps for double rotations.
SOLUTION:









5. **[8 pts]** Write a recursive function that compares two given binary trees. It returns 1 if two trees are identical, 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) (1 PTS)
          return 1; (1 PTS)
     else if (A != NULL && B != NULL)   (1 PTS)
          if (A->data == B->data) (1 PTS)
               return ( check(A->left, B->left) *
                        check(A->right, B->right)); (3 PTS)
          else return 0; (1 PTS)
     else return 0; (1 PTS)
}
```

6. **[9 points]** Given the array [79, 43, 56, 12, 4, 46, 70, 34, 89, 8, 23, 67, 37] 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

[79, 43, 56, 12, 4, 46, 70, 34, 89, 8, 23, 67, 37]

P1: [43, 79, | 56, 12, 4, 46, 70, 34, 89, 8, 23, 67, 37]

P2: [43, 56, 79, | 12, 4, 46, 70, 34, 89, 8, 23, 67, 37]

P3: [12, 43, 56, 79, | 4, 46, 70, 34, 89, 8, 23, 67, 37]

1 PT FOR EACH CORRECT ARRAY

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

[79, 43, 56, 12, 4, 46, 70, 34, 89, 8, 23, 67, 37]

P1: [4, 79, 43, 56, 12, 8, 46, 70, 34, 89, 23, 37, 67]

P2: [4, 8, 79, 43, 56, 12, 23, 46, 70, 34, 89, 37, 67]

P3: [4, 8, 12, 79, 43, 56, 23, 34, 46, 70, 37, 89, 67]

1 PT FOR EACH CORRECT ARRAY

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

[79, 43, 56, 12, 4, 46, 70, 34, 89, 8, 23, 67, 37]

P1: [4 | 43, 56, 12, 79, 46, 70, 34, 89, 8, 23, 67, 37]

P2: [4, 8 | 56, 12, 79, 46, 70, 34, 89, 43, 23, 67, 37]

P3: [4, 8, 12 | 56, 79, 46, 70, 34, 89, 43, 23, 67, 37]

1 PT FOR EACH CORRECT ARRAY