Computer Science Foundation Exam

May 2, 2003

COMPUTER SCIENCE I

Section I B

No Calculators!

KEY

Name:____

SSN: ____

Score: /50

In this section of the exam, there are four (4) problems

You must do all of them.

The weight of each problem in this section is indicated with the problem. The algorithms in this exam are written in C. Any algorithms that you are asked to produce should use a syntax that is clear and unambiguous. Partial credit cannot be given unless all work is shown.

As always, be complete, yet concise, and above all be neat. Credit cannot be given when your results are unreadable.

1. **(10 points)**
   Write a recursive function that will reverse the pointers of a singly-linked list while traversing the list only once. Initially, each node points to its logical successor, after calling the reverse function each node points to its logical predecessor.

   Assume the following definitions and declarations have been made:

```
  struct node {
          int data;
          struct node *next;
  }
```

```
mylist = (struct node *) malloc(sizeof (struct node));
//mylist points to the head of the list to be reversed
```

   The initial call is: **mylist = reverse(mylist).**

SOLUTION

```
struct node * reverse(struct node *head)
{
     if (head == NULL)
         return NULL;
     else
         return(reverse(NULL, head));
}

struct node * reverse (struct node * p, struct node *q)
{
        struct node *t;

        if (q -> next == NULL){
           q -> next = p;
           return q;
        }
        else {
           t = q -> next;
           q -> next = p;
           return(reverse (q, t));
        }
}
```

2. **(10 points – 9pts(a), 1pt(b))**
Answer questions (a) and (b) for the recursive function shown below.
   (a) Provide a trace of the execution of this function when it is initially called with the statement: **x = f(1,3)**.
   (b) What is the final value that is assigned to **x** by this function?

```
int  f (int x, int y)
{
    if (x == 0  && y >= 0)
          return y + 1;
    else if (x > 0 && y == 0)
          return (f(x–1,1));
    else if (x > 0 && y > 0)
          return (f(x–1, f(x, y–1)));
}
```

Solution to (a):

x = f(1, 3)
= f(0, f(1, 2))
= f(0, f(0, f(1, 1)))
= f(0, f(0, f(0, f(1,0))))
= f(0, f(0, f(0, f(0,1))))
= f(0, f(0, f(0, 2)))
= f(0, f(0, 3))
= f(0,4)
= 5

Solution to (b): 5

3. **(15 points – 5 points each)**

Find the closed form expression in terms of the parameter N or an exact value if the summation limits are known, for each of the following summations. Show all of your work; an answer alone is not sufficient to receive full credit.

a) $\sum_{i=0}^{N}(6i-3) =$

Solution: 3N^2 - 3

$$\sum_{i=0}^{N}(6i-3) = 6\sum_{i=0}^{N} i - 3\sum_{i=0}^{N} 1 \quad = \quad \frac{6(N)(N+1)}{2} - 3N - 3 = 3N^2 - 3$$

---

b) $\sum_{i=1}^{3N+5}(4\,i+2) =$

Solution: 18N^2 + 72N + 70

$$\sum_{i=1}^{3N+5}(4i+2) = 4\sum_{i=1}^{3N+5} i + 2\sum_{i=1}^{3N+5} 1 \quad = \quad \frac{4(3N+5)(3N+6)}{2} + 2(3N+5)$$

$$= \frac{4(3N+5)(3N+6)}{2} + 2(3N+5) = \frac{4\left(9N^2 + 18N + 15N + 30\right)}{2} + 6N + 10$$

$$= 18N^2 + 36N + 30N + 60 + 6N + 10 = 18N^2 + 72N + 70$$

---

c) $\sum_{i=15}^{38}(4i-3) =$

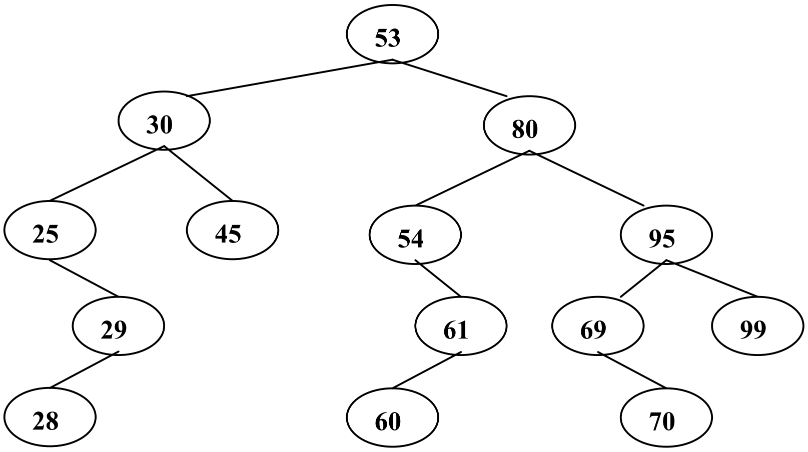
Solution: 2472

$$\sum_{i=15}^{38}(4i-3) = \left(4\sum_{i=1}^{38} i - 54\sum_{i=1}^{14} i\right) - \left(3\sum_{i=1}^{38} 1 - 3\sum_{i=1}^{14} 1\right) = \frac{4(38)(39)}{2} - \frac{4(14)(15)}{2} - 3(38) + 3(14)$$

$$= 2(38)(39) - 2(14)(15) - 3(38) + 3(14) = 2964 - 420 - 114 + 42 = 2472$$

4. (15 points – 2pts(a), 4pts(b), 9pts(c))

Given the following Binary Tree, answer the questions (a) through (c) below :



(a) Is this a valid Binary Search Tree? (circle one) [✓ YES] NO

(b) List the nodes of this tree in the order that they are visited in a postorder traversal:

28	29	25	45	30	60	61	54	70	69	99	95	80	53

1st node visited

last node visited

(c)Execute the algorithm shown below using the tree shown above. Show the exact contents of both the stack and the queue when the algorithm completes execution. Assume that the initial call is: **P4(root, 55)** and that the tree nodes and pointers are defined as shown.

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

void P4(struct tree_ptr *node_ptr,  int key) {
   if (node_ptr != NULL){
        if (node_ptr->data >= key){
             push(node_ptr->data);
             P4(node_ptr->left,  (key - 10));
             P4(node_ptr->right,  (key + 10));
        }
        else{
             enqueue(node_ptr->data);
             P4(node_ptr->right, (key + 15));
             P4(node_ptr->left,  (key - 15));
        }
}
```

top

contents of the stack:

28	29	25	99	95	60	80				

head

contents of the queue

53	54	61	69	70	30	45				