SOLUTION to Foundation exam (Part B of Computer Science I)

December 2004

1. How many multiplications are being performed in the following code? Show your complete work.. an answer alone is not sufficient to earn full credit. **[8 pts]**

```
for (k =11; k<= 20;  k++){
     for (i= 1; i <=k; i++) {
          w = t * k;
          p = z * i;
          for ( j = 1; j <= 14; j++)
               s = k * t * j;
     }
}
```

$$\sum_{k=11}^{20} \sum_{i=1}^{k} (2 + \sum_{j=1}^{14} 2)$$ [2 pts]

$$\sum_{k=11}^{20} 30k$$ [1 pt]

$$(\sum_{k=1}^{20} 30\,k - \sum_{k=1}^{10} 30k)$$

[2 pt]

$$= 30\ (20)\ 21/2 - 30(10)11/2$$ [2 pts]

$$= 4650$$ [1 pt]

2. Using summations find the value of count in terms of p after the following segment has been executed. Note that p is an even integer. Show your complete work.. an answer alone is not sufficient to earn full credit **[10 pts]**

```
count = 0;
for (i= 0 ;    i <  16 * p + 16 ;   i++) {
     for ( j = p/2;   j <= p ;  j++) {
          count+ = p - j ;
     }
}
```

$M = 16\,p + 16$

$$\sum_{i=0}^{M} \sum_{j=p/2}^{p} (p - j) \qquad \text{[2 pts]}$$

$$\sum_{i=0}^{M} \Big[\sum_{j=p/2}^{p} p - \sum_{j=p/2}^{p} j \Big]$$

$$\sum_{i=0}^{M} \Big[p\,(p - p/2 + 1) - \Big(\sum_{j=1}^{p} j - \sum_{j=1}^{p/2-1} j \Big) \Big] \qquad \text{[4 pts]}$$

$$\sum_{i=0}^{M} [\, p(1 + p/2) - p(p+1)/2 + (p/2 - 1)(p/2)/2 \qquad \text{[2 pts]}$$

$$\sum_{i=0}^{M} p(p+2)/8$$

$$= 16(p+1)p\,(p+2)/8$$

$$= 2p(p+1)(p+2) \qquad \text{[2 pts]}$$

3. Write a recursive function **struct node * largest(struct node * B)** which returns a pointer to the node containing the largest element in a BST (binary search tree). The node structure is as follows:

```
struct node {
     int node_value;
     struct node * left, *right;
};
```

[8 pts]

```
struct node* largest(struct node *B){
      if ( B==NULL)
            return NULL;
      else if (B->right ==NULL)
            return B;
      else return largest(B->right);
}
```

Grading: 4 pts for an iterative solution

4. In a binary tree, each node may have a single child, two children, or no child. Write a recursive function **`int one (struct tree_node *p)`** for a binary tree which returns the number of nodes with a single child.

Use the node structure

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

[10 pts]

```
int one  ( struct tree_node *p){
      if ( p != NULL)
```

```
        {
            if( p->left == NULL)
                if( p->right != NULL)
                    return 1+ one(p->right);
            else if( p->right == NULL)
                 if( p->left != NULL)
                    return 1+ one(p->left);
            else
                    return one (p->left) + one(p->right) ;
        }
}
```

Solution 2:

```
int one  ( struct tree_node *p){
        if ( p != NULL)
         {
            if( p->left == NULL && p->right != NULL
                ||  p->right == NULL &&  p->left != NULL)
                    return 1+ one(p->left)+ one(p->right);
            else
                    return (one (p->left) + one(p->right)) ;
        }
}

        return 0;
}
```

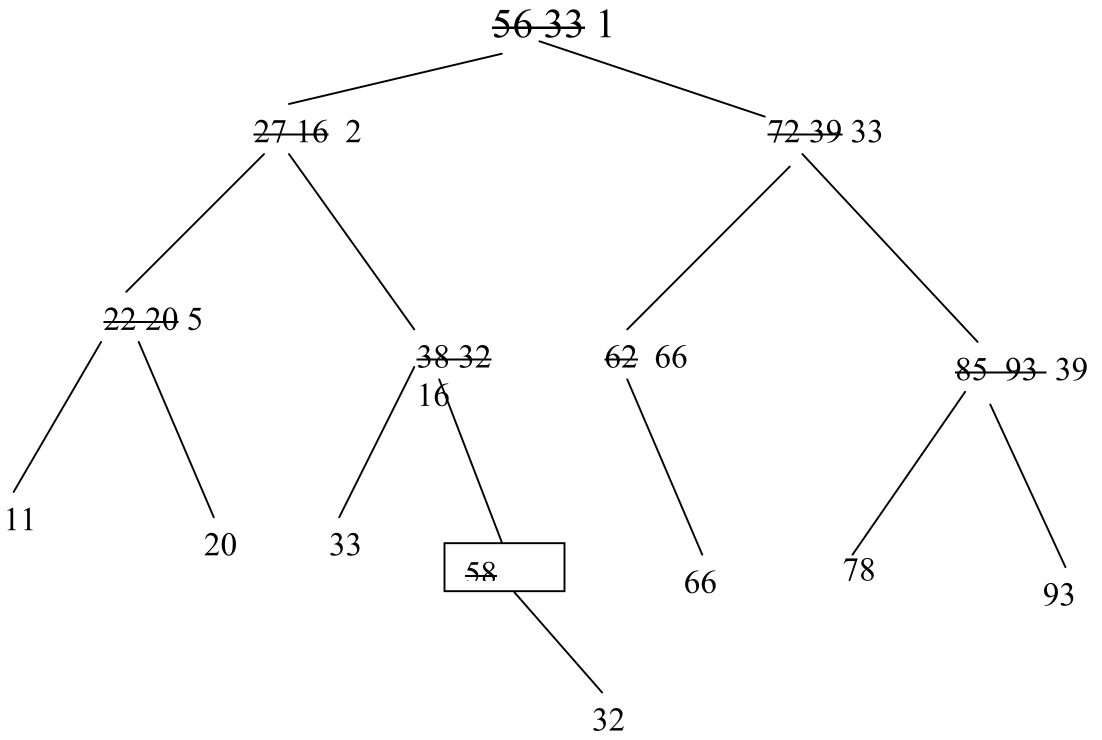
5. The following code is applied on the tree shown below with p pointing to the root of the tree. Show each change on the tree by crossing out the old value and replacing with the new value. **[14 pts]**

```
struct node {
     int data;
     struct node * left, *right:
};
func( struct node *p)
{
     if ( p == null)
          return;
     func(p ->right);
     func( p->left);
     if (p->right != null)
          p ->data = p ->right -> data;

     if (p->left != null)
          p ->data = (p ->left -> data)/2;

}
```

Grading : One point for each change in value


~~56 33~~ 1
~~27 16~~ 2
~~72 39~~ 33
~~22 20~~ 5
~~38 32~~
16
~~62~~ 66
~~85 93~~ 39
11
20
33
~~58~~
66
78
93
32