Homework – Notes 11-13

(Solution)

5.59

```
Node remove ( Node h, Item v)
{
      If (h == null) return null;
      If (equals(h.item, v))
      {
                              h.l = null;
                              h.r = null;
      }
                              If (h.l != null) remove (h.l, v);
                              If (h.r != null) remove (h.r, v);

}
```

5.68 Total no. of nodes = NM + 1
No. of external nodes = MN + 1 – N

5.79

Preorder :- DBACFEG, CBADE, ECBADHFGI
Inorder :- ABCDEFG, ABCDE, ABCDEFGHI
Postorder :- ACBEGFD, ABEDC, ABDCGFIHE
Levelorder :- DBFACEG, CBDAE, ECHBDFIAG

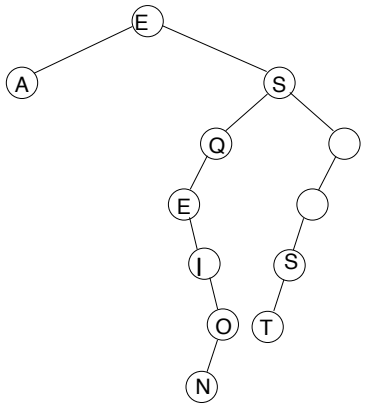
5.86

```
Static int countLeaves (TreeNode node)
{
      If (node == null) return 0;
      Else if ((node.left == null) && (node.right==null))
            return 1;
      Else
            return countLeaves(node.left) + countLeaves(node.right);
}
```

5.87

```
Static int countChild (TreeNode node)
{
        If (node == null) return 0;
        Else if ((node.left == null) && (node.right!=null))
                || ((node.left != null) && (node.right==null))
            return 1;
        Else
            return countChild(node.left) + countChild(node.right);
}
```

12.58



12.70

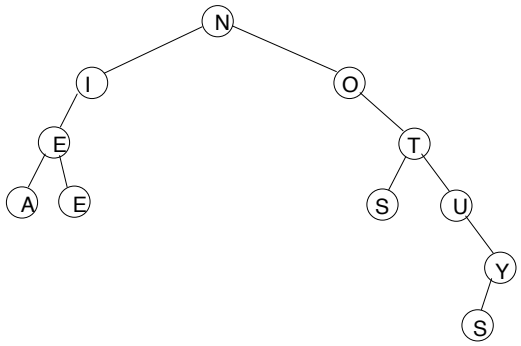
```
Static int height(TreeNode node)
{
        If (node == null) return 0;
        Else if ((node.left == null) && (node.right==null))
            return 0;
```

Else

return max(height(node.left) , height(node.right)) + 1;

}

12.84



12.90

E

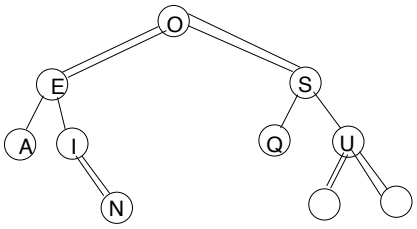
A S

I Y

O U

N T

13.48 *double edges are the "red" edges*



13.53

Because every 3-node can give 2 possible orientations, we can have a total of $\mathbf{2^t}$ combinations.

14.17	**E**	**A**	**S**	**Y**	**Q**	**U**	**T**	**I**	**O**	**N**
	0	**1**	**4**	**0**	**2**	**1**	**0**	**4**	**0**	**4**

0	->	O	->	T	->	Y	->	E	
1	->	U	->	A					
2	->	Q							
3									
4	->	N	->	I	->	S			

14.18

0	->	E	->	O	->	T	->	Y	
1	->	A	->	U					
2	->	Q							
3									
4	->	I	->	N	->	S			

Answer does not depend upon the order of inserting items

14.25	**E**	**A**	**S**	**Y**	**Q**	**U**	**T**	**I**	**O**	**N**
	7	**11**	**1**	**3**	**11**	**7**	**12**	**3**	**5**	**10**

0	
1	S
2	
3	Y
4	I
5	O
6	
7	E
8	U
9	
10	N
11	A
12	Q

13	T
14	
15	

14.26

E	A	S	Y	Q	U	T	I	O	N
5	1	9	5	7	1	0	9	5	4

0	T
1	A
2	U
3	I
4	N
5	E
6	Y
7	Q
8	O
9	S

14.31 **E A S Y Q U T I O N**

M = 16

Hash function 1 = 11 K mod M
Hash function 2 = (K mod 3) + 1

	Hash Table
0	
1	S
2	
3	Y
4	I
5	O
6	
7	E
8	U
9	
10	N
11	A
12	T

Input Symbol Scanned

E = H f1 (E)
= 11 K mod 16
= 11 X 5 mod 16
= 55 mod 16 = 7
A = H f1 (A) = 11 mod 6 = 11
S = H f1 (S) = 11X 19 mod 6 = 1
Y = H f1 (Y) = 25 X 11 mod 16
Q = H f1 (Q) = 17 X 11 mod 16
Q = H f2 (Q) = (17 mod 3) + 1 = 3
Q is inserted in = 14
U = H f1 (U) = 7 (collision)
U = H f2 (U) = (21 mod 3) + 1 = 1
U goes into 7+1 = 8

13	U	T = H f1 (T) = 12 ; I = H f1 (I) = 3 (collision)
14	Q	= H f2 (I) = 1 ; I goes to 3+1 = 4th location
15		O = H f1 (O) = 5 ; N = H f1 (N) = 10

14.32 E A S Y Q U T I O N

M = 10 Hash function 1 = 11 K mod M (Initial probe)

Hash function 2 = (K mod 3) + 1 (Increment)

	Hash Table	Input Symbol Scanned
0	Q	H f1 (E) = 11 X 5 mod 10 = 5
1	A	H f1 (A) = 11 mod 10 = 1
2	U	H f1 (S) = 19X 11 mod 10 = 9
3	T	H f1 (Y) = 25 X 11 mod 10 = 5 (collision)
4	I	H f2 (Y) = (25 mod 3) + 1 = 2; Y goes to 5+2 = 7th location
5	E	H f1 (Q) = 7 (collision) ; H f2 (Q) = 3; Q goes to 0th location
6	O	H f1 (U) = 1 (collision) ; H f2 (U) = 1; U goes to 1+1 = 2
7	Y	Hf1(T)=0 (collision) ; Hf2(T) =(20 mod 3)+1=3
8	N	U goes to 1+1 = 2
9	S	Hf1(I)= 99 mod 10=9 (collision); Hf2(I) =1
		I goes finally to location 4
		Hf1(O)= 15 X 11mod 10= 5 (collision); Hf2(O) =1
		O goes to 6th location
		Hf1(N)= 4 (collisions); Hf2(N) =3;
		Finally gets inserted into location *8* after trying
		7 , 0 , 3, 6 , 9 , 2 , 5
