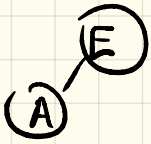
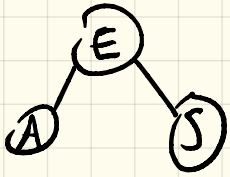
① EASYQUTION

1.

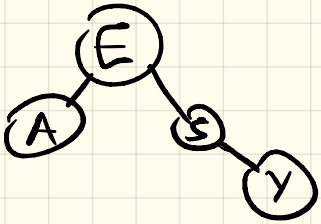
2.



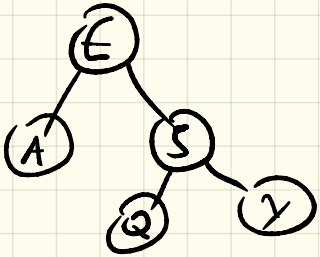
3.



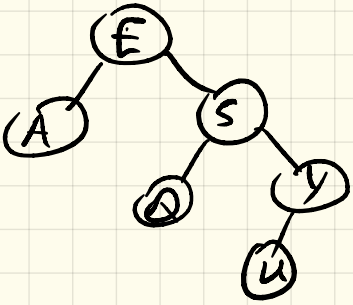
4.



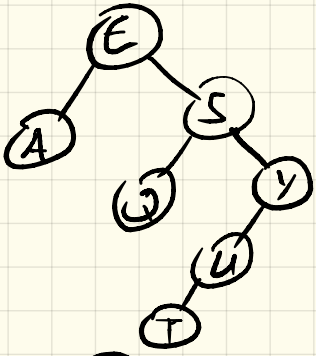
5.



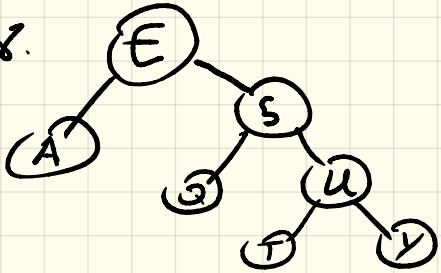
6.



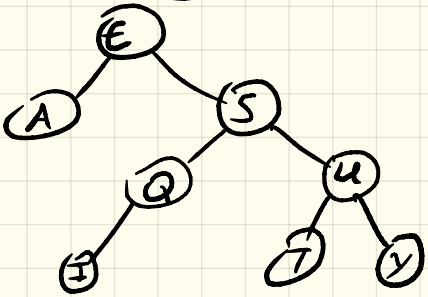
7.



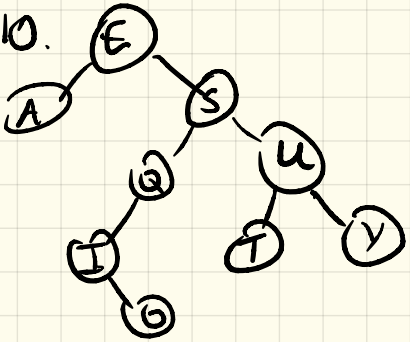
8.



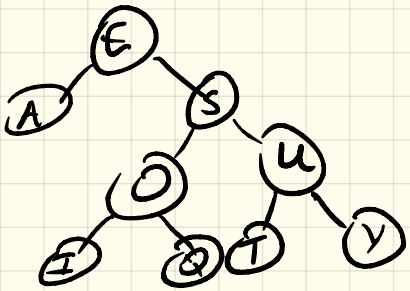
9.



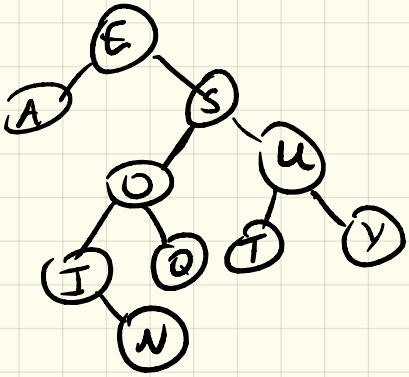
10.



11.



12.



(2) a) Exist (Root, a, b):

```
if (root = null)
 then false
elif ( root.data > b)
   then Exist ( root.left , a, b)
elif (root.data < a)
  then exist (root.right , a, b)
elif (root.data >= a & root.data <= b)
  then true
end if
```

b) Time complexity $O(\log n)$

We are skipping to either the left or right at each step, or returning.

(3)

0	91
1	
2	
3	81
4	147
5	265
6	162
7	202
8	21
9	48
10	75
11	37
12	77

b) 7 total collisions

(4)

0	75
1	91
2	77
3	48
4	94
5	
6	81
7	37
8	
9	189
10	265
11	
12	162
13	
14	

Collisions = 8

While it doesn't work for this data set, in general the increase in number of cells will lead to less collisions as there's more locations.

(5)

0	27
1	16
2	89
3	24
4	25
5	95
6	45
7	
8	
9	12
10	14
11	38
12	31

a) 11 cells

b) 17 collisions

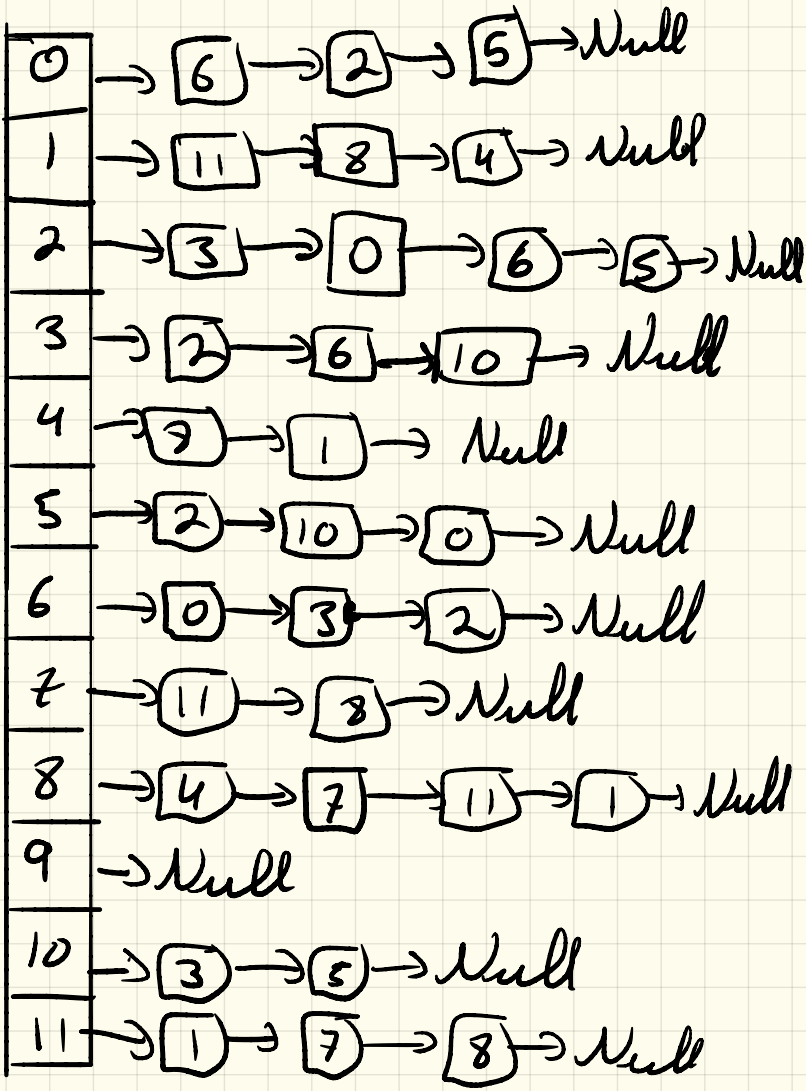
(26) S E A R C H X M P L

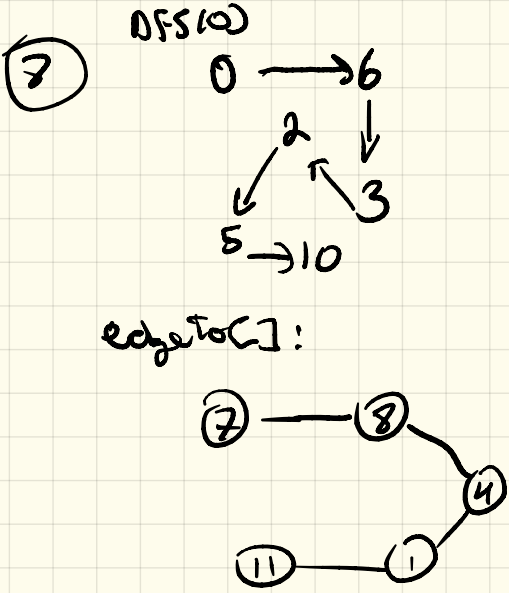
arr{19, 5, 1, 18, 3, 8, 24, 13, 16, 12}

check = true

```
for ( M=10, M<=26 ; M++)
    for ( a=1 ; a<=100 ; a++)
        for ( i=0; i<10 ; i++)
            arr[a] = arr[i*a] % M
        check = true;
        for ( i=0; i<10; i++)
            for(j=1+i; j<10 ;j++)
              if ( arr(j) = arr(i))
                check = false
                break;
        if ( check = false)
          break;
    if( flag = true)
      //Output for all k (0 to 10)
```

(7)


0 → 6 → 2 → 5 → Null
1 → 11 → 8 → 4 → Null
2 → 3 → 0 → 6 → 5 → Null
3 → 2 → 6 → 10 → Null
4 → 9 → 1 → Null
5 → 2 → 10 → 0 → Null
6 → 0 → 3 → 2 → Null
7 → 11 → 8 → Null
8 → 4 → 7 → 11 → 1 → Null
9 → Null
10 → 3 → 5 → Null
11 → 1 → 7 → 8 → Null


DFS(0)
7
0
6
2
3
5
10
edgeTo[]:
7
8
4
11
1