Midterm Exam 2 Solutions

1. Using the counting sort, answer the following:

a. Using the counting sort, show the state of the sort after each of the first three passes. Show array A, B, and C after every pass, where C is cumulative sums and B is semi-sorted array

0,1, 5, 2, 4, 1, 2, 6, 4, 9, 8, 9, 1, 1

b. What is the running time of counting sort for this example? Describe any variables/parameters you used

Write the result after the first pass in a table like:

C=

0	1	2	3	4	5	6	7	8	9
1	5	7	7	9	10	11	11	12	14

B=

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
					1									

Write the result after the second pass in a separate table like:

0	1	2	3	4	5	6	7	8	9
1	4	6	7	9	10	11	11	12	14

B=

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
				1	1									

Write the result after the third pass in a separate table like:

0	1	2	3	4	5	6	7	8	9
1	3	6	7	9	10	11	11	12	14

B=

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
				1	1									9

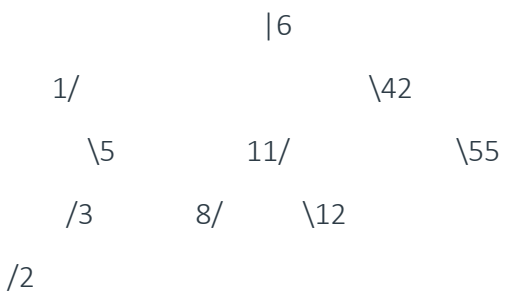
b) O((r+n) ; r= range of numbers(here it's 9-0=9), n=number of input(here it is 14)

2. Populate the following list in a binary search tree:

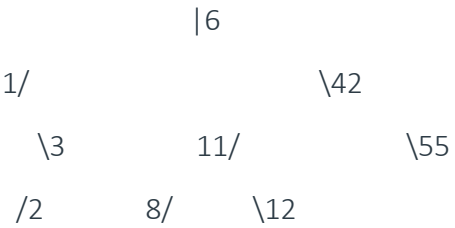
S = {6, 1, 5, 3, 42, 11, 55, 12, 8, 2}

A. Populate the tree with a root of 6
B. Delete node 5 from the tree built in part "A"
C. Insert 43 in the tree built in part "A"
D. Provide the pre-order travers for the tree built in part "A"

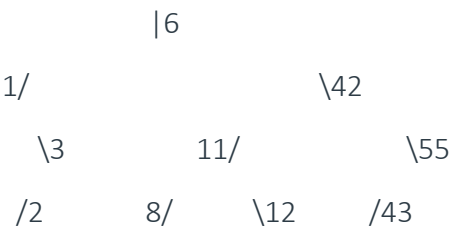
a)



b)



c)



d) 6, 1, 3, 2, 42, 11, 8, 12, 55, 43

3. For a Hash Tables, draw the contents of the hash table in the boxes below given the following conditions:

A. The size of the hash table is 11
B. Open addressing and double hashing is used to resolve collisions.
C. The hash function used is H1(k) = k mod 11
D. The second hash function is: H2(k) = 6 – (k mod 6)

What values will be in the hash table after the following sequence of insertions?

45, 9, 27, 34, 12, 56

Draw the values in the boxes below and show your work.

45 % 11 = 1; empty

9 % 11 = 9; empty

27 % 11 = 5; empty

34 % 11 = 1; collision.

(1 + [1*(6-(34%6)])%11 = (1 + [1*(6-4)])%11 = (1+2)%11 = 2%11 = 3; empty

12 % 11 = 1; collision.

(1 + [1*(6-(12%6)])%11 = (1 + [1*(6-0)])%11 = (1+6)%11 = 7; empty

56 % 11 = 1; collision.

(1+ [1*(6-(56%6)])%11 = (1 + [1*(6-2)])%11 = (1+(1*4))%11 = 5; collision.

(1+ [2*(6-(56%6)])%11 = (1 + [2*(6-2)])%11 = (1+(2*4))%11 = 9; collision.

(1+ [3*(6-(56%6)])%11 = (1 + [3*(6-2)])%11 = (1+(3*4)%11 = 13%11 = 2; empty

0	
1	45
2	56
3	34
4	
5	27
6	
7	12
8	
9	9
10	

4. In the following recurrence, K is a random number in each iteration.

$T(n) = T(k) + T(n - k) + O(n)$

A. What is the K value that can intrigue the worst case?
B. What is the best-case time complexity?
C. Explain and justify your answer to part "B" using input boundaries

A) $K = n/2$

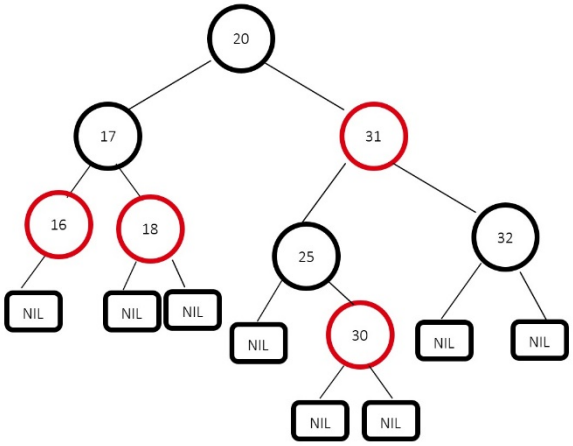
B) $O(n \log n)$

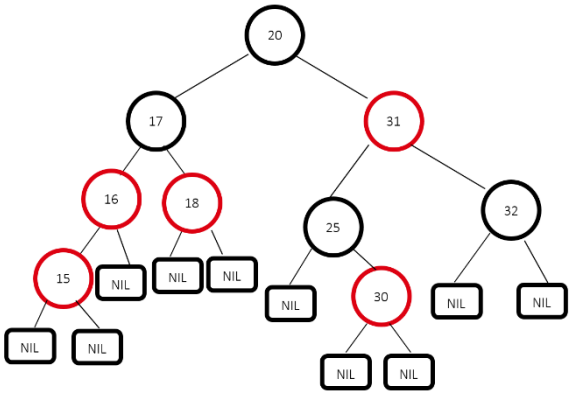
C) Dividing the input size by two in every step will create a balanced binary tree which will result in the shortest possible height for the algorithms recursion tree. Hence it will result in best case time complexity.

5. Refer to the given Red-Black Tree to answer the fill in the blanks:

In image (a), the black height of node 32 is -------1------ and height of node 20 is --------4----- and the black height of the tree is ---------2----------.

In image (b), a new node 15 is inserted, after fixing the violation, the color of node 18 will be ----------black---------





Answers: 1 4, 2, Black

6.

a) What does the following Stack "S" look like after calling Push (Q, 6) and Pop(Q) twice? Assume the head of the stack is at index 2.

0	1	2	3	4	5	6
		7	2	3		

b) Explain the main differences between a stack and a queue?

a)

0	1	2	3	4	5	6
				3	6	

b) A queue is FIFO(First In First Out) whereas a Stack is LIFO(Last In First Out). Queue operations are enqueue and dequeue whereas Stack operations are Push And Pop

7. In a decision tree for the following array A, answer the following questions:

A=[X 1, X 2, X 3, X 4]:

a) The minimum number of leaves is [a]

b) The maximum number of leaves is [a2]

c) Every internal node represents [a3]

d) The height of the tree represents [a4]

a) minimum: n!;

b) maximum: 2^h

b) A comparison between elements.

c) Worst case comparison

8.

Fill in the blank for the following partitionSort function:

```
partitionSort(int Arr[],int low, int high)
{
int pivot,i,j;
pivot=Arr(low);
i= [          ]
j= [          ] ;
while (true){
do{
j--;
}while( [          ] >pivot);
do{
i++;
}while( Arr(I)<pivot );
if (i<j){
int t = Arr(i);
[          ] ;
Arr(j) = t;
}
else{
return [          ] ;
}
```

```
int partitionSort(int Arr[],int low, int high)
{
    int pivot,i,j;
    //median
//    int mid,median;
//    mid=(low+high)/2;
//    median=(Arr[low]+Arr[mid]+Arr[high])/3;
//    pivot=median;
    pivot=Arr[low];
    i=low-1;
    j=high+1;
    while (true){
                do{
                    j--;
                }while(Arr[j]>pivot);
```

```
                do{
                    i++;
                }while( Arr[i]<pivot );
            if (i<j){
                int t = Arr[i];
                Arr[i] = Arr[j];
                Arr[j] = t;
            }
            else{
                return j;
            }
    }
}
```