1.(a) Draw the Binary Tree for the following array using the strategy defined below:

(i) root is at A[1]

(ii) left child of node at A[i] is at A[2i]

(iii) left child of node at A[i] is at A[2i+1]

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 4 | 1 | 3 | 2 | 16 | 9 | 10 | 14 |

1.(b) Write down the pseudo-code for Min-Heapify(A,i) operation and then run Min-Heapify(A,2) for the initial tree you obtained in question 1(a).

2. Analyse the time complexity of Merge-Sort. Note that you need to write down the recurrence relation of it and solve that relation using recurrence tree method. While solving the problem you need to answer the following queries.

(i) Draw the Recurrence Tree.

(ii) State the height, no. of levels and no. of nodes at the bottom level of the tree in terms of input size ***n***. Mention these terms with respect to the Array A [given in 1(a)]

3.(a) Describe running time of following Code Segments w.r.t to Best Case, Worst Case.

Search\_and\_Sum (item, B, n) ►B[1 . . n]{

intfun(intkey,int[] A)

{

int sum = 0;

**for**(**int** i=0;i<=m+m+10;i++)

{

**int** k=B.length;

**while**(k>=1 &&item != B[k])

{

sum=\*item;

k/=2;

}

}

return sum;

}

(b) Order the following terms in ascending order.

O(100),O(nlogn),O(logn),O(n!),O(2n),O(n2)