Given NANMA
1BM18CS021

Insert Operation on b-tree

```
function
  insert (int k)
      if root == NULL
          ~~root = new~~
          //allocate memory for root
          root -> keys[0] = k
          root -> n = 1
      else
          if root -> n == 2*t -1
              //allocate memory for new root
              s -> C[0] = root
              s -> splitChild (0, root)
              int i=0
              if (s->keys[0] < k)
                  i++
              s -> C[i] -> insertNonFull(k)
              root = s
          else
              root -> insertNonFull(k)
function
  insertNonFull (k)
      int i = n-1
      if leaf == true
          while i>=0 && keys[i] >k
              keys[i+1] = keys[i];
              i--;
          keys [i+1] = k
          n = n+1
```

GURU MANNA
1BM18CS031

```
else
    while (i> = 0 && keys[i] >k
          i--
    if (C[i+1]-> n == 2*t-1
          Split Child(i+1 , C[i+1])
          if (keys[i+1]<k
                i++
       C[i+1] -> insertNonFull(k)
function
  Splitchild (i , BTreeNode *y) {
        //New node which is going to store t-1 keys of y
  *z = new BTreeNode(y->t, y-> leaf)
     z->n = t-1
     for j=0 to t-1 ; j++
          z -> keys[j] = y -> keys[j+t]
     if y->leaf ==false
          for j=0 to t ; j++
               z -> C[j] = y -> C[j+t)
     y->n = t-1 //Reducing number of keys in y
     for j=n >= i+1 ; j--
          C[j+1) = C[j]
     c [i+1] = z
    for j = n-1 to i ; j--
          keys [j+1] = keys[j]
    key[i] = y-> keys[t-1] ; n = n+1
```