B-Tree (Insertion)

Ankur Mishra
IIBM18CS013

```
public void insert(int key)
{   Node r = root;
    if (r.n == 2*T-1)
    {   Node s = new Node();
        root = s;
        s.leaf = false;
        s.n = 0;
        s.child[0] = r;
        split(s, 0, r);
        _insert(s, key);
    }
    else
    _insert(r, key);
}
void _insert(Node x, int k)
{   if(x.leaf)
    {   int i=0;
        for(i = x.n-1; i>=0 && k<x.key[i]; i--)
            x.key[i+1] = x.key[i];
```

Amber Mishra
IBM18CS013

```
        r.key[i+1] = k
        r.n = r.n+1;
    }
    else
    {   i = 0
        for i = r.n-1 to 0.
        i++
        Node t = r.child[i];
        if (t.n == 2*T-1)
          split (r, i, t)
        if (k > r.key[i])
            i ++
      - insert(r.child[i], k);
    }
}

void split (Node r, int pos, Node y)
    {   Node z = new Node();
        z.leaf = y.leaf;
        z.n = T-1;
      for j → 0 to T-1
          z.key[j] = y.key[j+T];
```

Amber Mishra
IBM18CS013

```
if (!y.leaf)
{
  for j -> 0 to T
     z.child[j] = y.child[j+T];
  }
  y.n = T-1
  for j -> x.n to loc+1, j--
     x.child[j+1] = x.child[j];
  x.child[loc+1] = z;

class Node
{
  int n;
  int key[] = new int[2*T-1]
  Node child[] = new Node[2*T]
  boolean leaf = true;
}
```