Program 5

2-3 trees

```
Class TwoThreeTree {
    TwoThreeNode root;
    int k;
    TwoThreeTree() {
        root = NULL;
        k = 0;
    }
}
void insert (int k)
{
    if (root)
    {
        root = new TwoThreeNode();
        root -> key[0] = k;
        root -> n = 1;
    }
    else if (root -> n == 2 * t - 1) {
        TwoThreeNode s = new TwoThreeNode();
        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 = new TwoThreeNode(k);
    }
}

void insertNonFull(int k)
{
    int i = n-1;
    if (leaf) {
        while (i >= 0 && keys[i] > k) {
            keys[i+1] = keys[i];
            i--;
        }
        keys[i+1] = k;
        n += 1;
    }
    else {
        while (i >= 0 && keys[i] > k)
            i--;
        if (C[i+1]->n == 2*t-1)
            split(i+1, C[i+1]);
        else if (keys[i+1] < k)
            i++;
    }
    C[i+1]->insertNonFull(k);
}

void split(int i, TwoThreeNode *y)
{ TwoThreeNode *z = new TwoThreeNode(
    y->leaf);
```

```
z->n = t-1;
for (int j=0; j<t-1; j++)
    z->keys[j] = y->keys[j+1];
if (y->leaf == false)
{ for (int j=0; j<t; j++)
      z->C[j] = y->C[j+t];
}
y->n = t-1;
for (int j=n; j>=i+1; j--)
    C[j+1] = C[j];
C[i+1] = z;
for (int j=n-1; j>=i; j--)
    keys[j+1] = keys[j];
keys[i] = y->keys[t-1];
n = n+1;
}
void remove(int k){
if (!root)
{ cout << "Tree empty";
  return;
root = root->remove(k);
if (root->n == 0)
{ Node* temp = root;
if (root->leaf)
    root = NULL;
else
```

```
delete keys[i];
    }
}
void removeFromLeaf (int idx){
    for (int i = idx + 1; i < n; ++i)
        keys[i-1] = keys[i];
    n--;
    return;
}
void removeFromNonLeaf (int idx){
    int k = keys[idx];
    if (C[idx]->n >= t){
        int pred = getPred(idx);
        keys[idx] = pred;
        C[idx]->remove(pred);
    }
    else if (C[idx+1]->n >= t){
        int succ = getSucc(idx);
        keys[idx] = succ;
        C[idx+1]->remove(succ);
    }
    else{
        merge(idx);
        C[idx]->remove(k);
    }
    return;
}
```