```cpp
#include <iostream>
using namespace std;

class Node {
  int *keys;
  int t;
  Node **C;
  int n;
  bool leaf;

  public:
  Node(int tt, bool lleaf);

  void insertNonFull(int k);
  void splitChild(int i, Node *y);
  void traverse();

  friend class BTree;
};

class BTree {
  Node *root;
  int t;

  public:
  BTree(int tt) {
    root = NULL;
    t = tt;
  }

  void traverse() {
    if (root) root->traverse();
  }

  void insert(int k);
};

Node::Node(int tt, bool lleaf) {
  t = tt;
  leaf = lleaf;

  keys = new int[2 * t - 1];
  C = new Node *[2 * t];

  n = 0;
}

void Node::traverse() {
```

```
  int i;
  for (i = 0; i < n; i++) {
    if (leaf == false) C[i] -> traverse();
    cout << " " << keys[i];
  }

  if (leaf == false) C[i]->traverse();
}

void BTree::insert(int k) {
  if (root == NULL) {
    root = new Node(t, true);
    root -> keys[0] = k;
    root -> n = 1;
  } else {
    if (root -> n == 2 * t - 1) {
      Node *s = new Node(t, false);

      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);
  }
}

void Node::insertNonFull(int 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;
  } else {
    while (i >= 0 && keys[i] > k) i--;

    if (C[i + 1] -> n == 2 * t - 1) {
      splitChild(i + 1, C[i + 1]);
```

```
      if (keys[i + 1] < k) i++;
    }
    C[i + 1] -> insertNonFull(k);
  }
}

void Node::splitChild(int i, Node *y) {
  Node *z = new Node(y -> t, y -> leaf);
  z -> n = t - 1;

  for (int j = 0; j < t - 1; j++) z -> keys[j] = y -> keys[j + t];

  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;
}

int main() {
  int n;
  cout << "Enter B Tree Order \n";
  cin >> n;
  BTree t(n);

  int k;
  cout << "Enter Elements \n";
  cin >> k;

  while (k--) {
    int m;
    cin >> m;
    t.insert(m);
  }

  cout << "The B-tree is: ";
  t.traverse();
}
```

```
Enter B Tree Order
2
Enter Elements
10
20
5
34
48
59
49
43
432
56
349
The B-tree is:  5 20 34 43 48 49 56 59 349 432
```