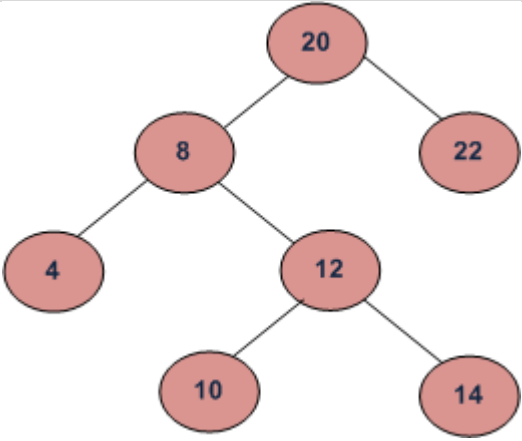


Demo Duyệt cây BST theo mức

Cây BST	Kết quả: Duyệt theo mức
	Output: 20 8 22 4 12 10 14

Code

```
/* Duyệt cây nhị phân theo mức*/

#include<iostream>
#include<queue>
#include<stdlib.h>
using namespace std;

// cấu trúc cây nhị phân BST
struct node {
    int data;
    node* pLeft;
    node* pRight;
};

// duyệt cây BST theo mức dùng hàng đợi
void printLevelOrder(node* root) {

    if(root == NULL) return;
    // tạo hàng đợi rỗng cho duyệt theo mức
    queue<node* > q;
    // Thêm root vào hàng đợi
    q.push(root);
    while (q.empty() == false)
    {
        //số node tại mức đang xét
        int nodeCount = q.size();

        // Lấy ra tất cả các node mức đang xét
        // Thêm vào tất cả các node ở mức kế tiếp
        while (nodeCount > 0)
        {
            node* node = q.front();
            cout << node->data << " ";
            q.pop();
            if(node->pLeft != NULL)
                q.push(node->pLeft);
            if(node->pRight != NULL)
                q.push(node->pRight);
            nodeCount--;
        }
        cout << endl;
    }
}
```

```

        if(node->pRight != NULL)
            q.push(node->pRight);
        nodeCount--;
    }
    cout << endl;
}

//Hàm tạo một node mới
node* newNode(int data) {
    node* tmp = new node;
    tmp->data = data;
    tmp->pLeft = NULL;
    tmp->pRight = NULL;
    return tmp;
}

// test
int main() {
    //tạo cây BST
    node* root = newNode(20);
    root->pLeft = newNode(8);
    root->pRight = newNode(22);
    root->pLeft->pLeft = newNode(4);
    root->pLeft->pRight = newNode(12);
    root->pLeft->pRight->pLeft = newNode(10);
    root->pLeft->pRight->pRight = newNode(14);

    printLevelOrder(root);

    system("pause");

    return 0;
}

```

===== * * * =====