Question1.cpp

```cpp
#include <iostream>
using namespace std;
// A binary tree node has data, pointer to left child and a pointer to right child
struct Node
{
    int data;
    struct Node *left, *right;
};
// Utility function to create a new tree node
Node *newNode(int data)
{
    Node *temp = new Node;
    temp->data = data;
    temp->left = temp->right = NULL;
    return temp;
}

void printPreorder(struct Node *node)
{
    if (node == NULL)
    {
        return;
    }
    cout << node->data << " ";
    printPreorder(node->left);
    printPreorder(node->right);
}

int main()
{
    struct Node *root = newNode(1);
    root->left = newNode(2);
    root->right = newNode(3);
    root->left->left = newNode(4);
    root->left->right = newNode(5);

    // Function call
    cout << "\npreorder traversal of binary tree is \n";
    printPreorder(root);

    return 0;
}
```

Question2.cpp

```cpp
#include <iostream>
using namespace std;
// A binary tree node has data, pointer to left child and a pointer to right child
struct Node
{
    int data;
    struct Node *left, *right;
};
// Utility function to create a new tree node
Node *newNode(int data)
{
    Node *temp = new Node;
    temp->data = data;
    temp->left = temp->right = NULL;
    return temp;
}

void printPostorder(struct Node *node)
{
    if (node == NULL)
    {
        return;
    }
    printPostorder(node->left);
    printPostorder(node->right);
    cout << node->data << " ";
}
int main()
{
    struct Node *root = newNode(1);
    root->left = newNode(2);
    root->right = newNode(3);
    root->left->left = newNode(4);
    root->left->right = newNode(5);

    // Function call
    cout << "\nPostorder traversal of binary tree is \n";
    printPostorder(root);
    return 0;
}
```

Question3.cpp

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

class node
{
public:
    int data;
    node *left;
    node *right;

    node(int d)
    {
        this->data = d;
        this->left = NULL;
        this->right = NULL;
    }
};
node *buildTree(node *root)
{
    cout << "Enter data" << endl;
    int data;
    cin >> data;
    root = new node(data);
    if (data == -1)
    {
        return NULL;
    }
    cout << "Enter data for inserting in left " << data << endl;
    root->left = buildTree(root->left);
    cout << "Enter data for inserting in right " << data << endl;
    root->right = buildTree(root->right);
};

void inorder(node *root)
{
    if (root == NULL)
    {
        return;
    }
    inorder(root->left);
    cout << root->data <<" ";
    inorder(root->right);
}

void preorder(node *root)
{
    if (root == NULL)
    {
        return;
    }
    cout << root->data <<" ";
    preorder(root->left);
    preorder(root->right);
```

```cpp
}

void postorder(node *root)
{
    if (root == NULL)
    {
        return;
    }
    postorder(root->left);
    postorder(root->right);
    cout << root->data <<" ";
}

int main()
{
    node *root = NULL;

    // creating of binary tree
    root = buildTree(root);

    cout << "inorder traversal is:  ";
    inorder(root);

    cout << endl
         << "preorder traversal is:  ";
    preorder(root);

    cout << endl
         << "postorder traversal is:  ";
    postorder(root);

    return 0;
}
```