```
/*
	Mendoza, Moises
	Alaya, Jason
	Almashash Swed, Jawad

	CS A200
	October 25, 2017

	Lab 5
*/

#include "BST.h"

// Definition function insert (non-recursive)
void BST::insert(int item)
{
	if (root == nullptr)
	{
		root = new Node;
		root->data = item;
	}
	else
	{
		Node* child = root;
		Node *trail = child;
		bool equal = false;

		while (child != nullptr && !equal)
		{
			if (child->data == item)
			{
				cerr << "The item to insert is already in the list,
				duplicates are not allowed." << endl;
				equal = true;
			}
			else if (child->data > item)
			{
				trail = child;
				child = child->llink;
			}
			else if (child->data < item)
			{
				trail = child;
				child = child->rlink;
			}
		}

		if (trail->data > item && !equal)
		{
			trail->llink = new Node();
			trail->llink->data = item;
		}
		else if (trail->data < item && !equal)
		{
			trail->rlink = new Node();
			trail->rlink->data = item;
		}
```

```
	}
}

// Definition function totalNodes
int BST::totalNodes() const
{
	if (root == nullptr)
		return 0;
	else
		return totalNodes(root);
}

// Definition function totalNodes (recursive)
int BST::totalNodes(const Node* p) const
{
	int count = 1;

	if (p != nullptr)
	{
		if(p->llink != nullptr)
			count += totalNodes(p->llink);
		if (p->rlink != nullptr)
			count += totalNodes(p->rlink);
	}
	return count;
}

// Definition overloaded function preorderTraversal
void BST::preorderTraversal() const
{
	if (root == nullptr)
		cerr << "There is no tree.";
	else
		preorderTraversal(root);
}

// Definition overloaded function preorderTraversal (recursive)
void BST::preorderTraversal(const Node* p) const
{
	if (p != nullptr)
	{
		cout << p->data << " ";
		preorderTraversal(p->llink);
		preorderTraversal(p->rlink);
	}
}

// Definition overloaded function postorderTraversal
void BST::postorderTraversal() const
{
	if (root == nullptr)
		cerr << "There is no tree.";
	else
		postorderTraversal(root);
}

// Definition overloaded function postorderTraversal (recursive)
void BST::postorderTraversal(const Node* p) const
```

```
{
	if (p != nullptr)
	{
		postorderTraversal(p->llink);
		postorderTraversal(p->rlink);
		cout << p->data << " ";
	}
}
```