```
/*************************************************************
* AUTHOR     : Amirarsalan Valipour
* STUDENT ID     : 1103126
* Assignment #5  : DVD Movie ListIntro to OOP
* CLASS          : CS 1B
* SECTION        : MW - 7:30 pm - 9:50 pm
* DUE DATE       : 12/16/2019
*************************************************************/

#include "StackList.h"

                /********************************
                 *   CONSTRUCTOR / DESTRUCTOR   *
                 ********************************/


/****************************************************************
 * StackList ();
 * Constructor: Initialize class attributes
 * Parameters: none
 * Return:     none
 ****************************************************************/

StackList :: StackList()
{
    head = NULL;

    stackCount = 0;
}


/****************************************************************
 * ~StackList ();
 * Destructor: does not perform anything
 * Parameters: none
 * Return: none
 ****************************************************************/

StackList :: ~StackList()
{
    DVDNode *stckPtr;

    stckPtr = head;

    while (stckPtr != NULL)
    {
        head = head -> next;

        delete stckPtr;

        stckPtr = head;
    }
}


                            /******************
                             *    MUTATORS    *
                             ******************/

/
*******************************************************************
 * void Push (DVDNode newDVD);
 *
 * Mutator: This method will add a DVD node to the list to the
front
 *
-------------------------------------------------------------------
 * Parameter: newDVD (DVDNode) //IN - node to be added to list
 *
-------------------------------------------------------------------
 * Return: none
 
*******************************************************************
*/

void StackList :: Push (DVDNode newDVD)
{
    DVDNode *persPtr;

    persPtr = head;

    persPtr = new DVDNode;

    *persPtr = newDVD;


    //QUEUE

    persPtr -> next = head; head = persPtr;


    //UPDATE COUNTER

    stackCount++;

    persPtr = NULL;
}


/*************************************************************************
 * DVDNode Pop ();
 *
 * Mutator: This method will remove a DVD node from the front of the
 *          list and return the DVDNode being removed.
 * ------------------------------------------------------------------
 * Parameter: none
 * ------------------------------------------------------------------
 * Return: dvdPtr (DVDNode)
 *************************************************************************/

DVDNode StackList :: Pop ()
{
    DVDNode dvdPtr;

    DVDNode *persPtr;

    persPtr = head;

    if(IsEmpty())
    {
        cout << "The stack is empty!";

        return dvdPtr;
    }

    dvdPtr = Peek();

    persPtr = persPtr -> next;

    delete persPtr;

    persPtr = NULL;

    return dvdPtr;

}


                            /******************
                             *    ACCESSORS    *
                             ******************/

/*****************************************************************
* bool IsEmpty () const;
*
* Accessor; This method will return the boolean value whether
*           the list is empty or not empty.
* -------------------------------------------------------------
* Parameters: none
* -------------------------------------------------------------
* Return: emptyCheck (bool)
*****************************************************************/

bool StackList :: IsEmpty() const
{
    bool emptyCheck;

    if (stackCount == 0)
    {
        emptyCheck = true;
    }

    else
    {
        emptyCheck = false;
    }

    return emptyCheck;
}


/*****************************************************************
 * DVDNode Peek () const;
 *
 * Accessor; This method will return the DVD node of the first
 *               element on the list
 * -------------------------------------------------------------
 * Parameters: none
 * -------------------------------------------------------------
 * Return: dvdPtr (DVDNode)
 **************************************************************/

DVDNode StackList :: Peek() const
{
    DVDNode dvdPtr;

    dvdPtr.title = "EMPTY";

    if(!IsEmpty())
    {
        dvdPtr = *head;
    }

    return dvdPtr;
}


/**************************************************************
 * int Size () const;
 *
 * Accessor; This method will return the size of the list
 * -------------------------------------------------------------
 * Parameters: none
 * -------------------------------------------------------------
 * Return: stackCount (int)
 **************************************************************/
int StackList :: Size() const
{
    return stackCount;
}
```