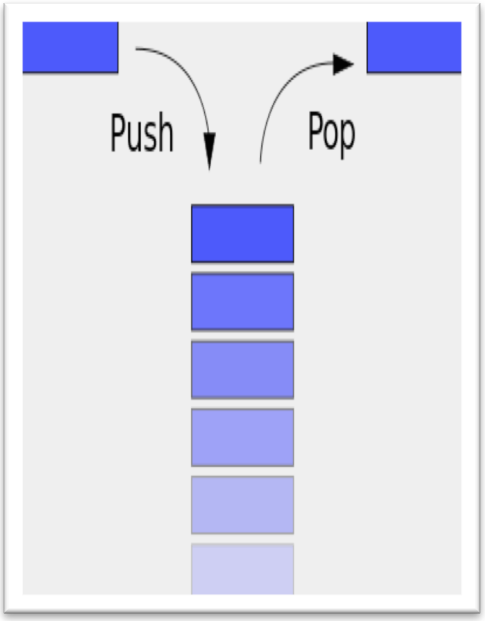
CSS 2100 Project #2
Library Application

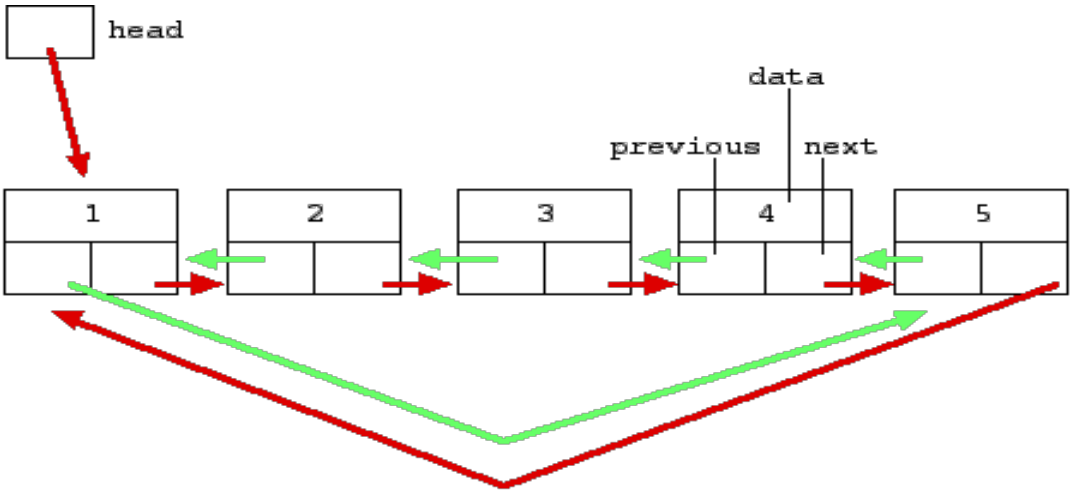
Introduction:

The program will manage a library of a university. The library has **a stack of books** and **doubly linked list of articles**.
The stack is a collection of books where the last element is the only one that can be accessed. The operations that can be used are **pop** and **push**. **push** is used to add an element to the stack and **pop** is used to remove the last element of a stack.



Stack of books

The doubly linked list is a sequential list of articles where each node is connected to the next one and the previous one. The first node is the head of the doubly linked list. The last node is not connected to the first one. The operations that can be done are **removeNode** and **addNode**.



Doubly Linked List of articles

Tasks:

1) First, you will create a header file **LibraryTypes.h** to define the following structured data types:
   a. **Book**: contains the elements: int isbn, string name, string authorName, int pagesNumber.
   b. **BookNode**: contains the elements: Book bk, BookNode *previousNode.
   c. **BookStack**: contains the element: BookNode *lastElement.
   d. **Article**: contains the elements: int articleId, string title, string field.
   e. **ArticleNode**: contains the elements: Article al, ArticleNode *previousNode, ArticleNode *nextNode.
   f. **ArticleDoublyLinkedList**: contains the element ArticleNode *firstElement.
2) Second, you will create two files **BookMgmt.cpp** and **BookMgmt.h** to manage the stack of books by using creating the following functions:
   a. **void pushBook(BookStack *bkStack, BookNode *bkNode):** the function will add the node bkNode to the stack bkStack. If bkNode is the first node, lastElement will take the value of bkNode. Else the element previousNode will take the value of

lastElement and lastElement will take the value of previousNode.

b. **BookNode *popBook(BookStack *bkStack):** the function will remove and return the last element of bkStack. If the stack is empty, the NULL value will be returned. If the stack has one element, lastElement will be returned and it will take the NULL value. Else lastElement will be saved in another variable node, lastElement will take lastElement.previousElement.
c. **bool isEmpty(BookStack *bkStack):** the function will return true if the bkStack is empty and false otherwise.
d. **void displayStackElements(BookStack *bkStack):** the function will display the books' names starting from the last element of bkStack to the first element of bkStack.
e. **void clearStack(BookStack *bkStack):** the function will free the memory used by bkStack.

3) Third, you will create two files **ArticleMgmt.cpp** and **ArticleMgmt.h** to manage the doubly linked list of articles by creating the following functions:
   a. **ArticleNode *getLastArticle(ArticleDoublyLinkedList *articleList):** the function will return the last article in articleList if it exists. The last node has the element NextNode equal to NULL.
   b. **void addArticle(ArticleDoublyLinkedList *articleList, ArticleNode *articleNd):** If articleNd is the first element, firstElement will take its value. Else, the function will link articleNd to the last article of articleList. A node n is linked to a node m of the list in the following way: n.previous will take the value of m and m.next will take the value of n.
   c. **ArticleNode *removeArticle(ArticleDoublyLinkedList *articleList, int index):** If the node doesn't exist, the function will return NULL value. If the $node_{index}$ exists, it will be returned and the links of articleList will be updated given the three following cases: $node_{index}$ doesn't have a next node,

$node_{index}$ doesn't have previous node and $node_{index}$ has a previous node and a next one.

d. **void displayListElements(ArticleDoublyLinkedList *articleList):** the function will display the Articles' names of articleList starting from the first element of articleList to the last element of articleList.
e. **void clearList(ArticleDoublyLinkedList *articleList):** the function will free the memory used by articleList.

4) You will write the code of a **main.cpp** file where you will create a stack of books and doubly linked list of articles and ask the user to enter the information of n books and m articles. After that you will give the option to the user to use a function that you implemented and the choice to exist the program when he is done. After the execution of a function, the collection of data that was used has to be displayed to show the changes.

How to submit your work:

1) You will submit a **compressed file** containing all of your .ccp and .h files. Any other submitted file(s) will not be accepted.
2) The deadline is **Sunday, November 16, 2014**.