void SinglyLinkedList::addFront(const Book& e)

{

Node\* temp = new Node; // Create new node.

temp->book = e; // Store data.

temp->next = head; // Current head is now next of our new node.

head = temp; // Our new node is now the new head.

}

Draw pictures to fully illustrate code

|  |
| --- |
| Node\* temp = new Node;  Could this be rewritten without the new keyword? |
| temp->book = e;  How could this line of code be rewritten? |
| temp->next = head;  Will temp’s next continue to point to the head of the list? |
| head = temp;  Can any of these steps be omitted in the adding of this data to the list? |

void SinglyLinkedList::removeFront()

{

if ( !empty() )

{

Node\* temp = head; // temp saves the current head.

head = temp->next; // Our new head is the old head's next.

delete temp; // Delete the previous head.

}

}

Draw pictures to fully illustrate code

|  |
| --- |
| if ( !empty() )  Is this condition necessary? |
| Node\* temp = head;  Where is temp in terms of the program’s memory? |
| head = temp->next;  Could this be rewritten as:  head = head->next; |
| delete temp;  Why are we deleting temp? |