LAB 4

Name	ID
FAIRS ALGHAMDI	1847152

1- The memory leak occurs within the remove() function. After removing an element from the beginning of the list, the head_ pointer is updated to a new node, but the old node is not deleted. To resolve this issue, a delete statement should be added as follows:

```
if (marker->next() == 0) {
    head_ = 0;
    delete marker;
    marker = 0;
} else {
    head_ = new Node(marker->value(), marker->next());
    delete marker;
    marker = 0;
}
```

2- Another bug is present when removing an element, as the next pointer of the previous node is not being updated. To address this issue, the following fix should be implemented:

After deleting the node, both when removing from the beginning and middle of the list, include the line

```
temp->next(marker->next());
```

If removing from the beginning of the list:

```
if (marker->next() == 0) {
    head_ = 0;
    delete marker;
    marker = 0;
} else {
    head_ = new Node(marker->value(), marker-
>next());
    delete marker;
    temp->next(marker->next()); // ADD THIS LINE
    marker = 0;
}
```

If removing from the middle of the list:

```
temp->next(marker->next()); // ADD THIS LINE
delete temp;
temp = 0;
```