04/30/17 04:58:40 /Users/hostname/Desktop/CSE 330/Lab4/[Lab4] Adnar Lozano.cpp

```
1  // Adnar Lozano
2  // CSE 330 Data Structures
3  // Lab 4 (Linked List 2)
4  // 4/26/17
5
6  #include  <iostream>
7  using namespace std;
8
9  class Node {
10 public:
11         int data;
12         Node* next;
13 };
14 Node* Insert_Tail (Node* head, int data) {
15         Node* temp = new Node();
16         temp->data = data;
17         temp->next = NULL;
18         if (head == NULL) head = temp;
19         else {
20                 Node* temp1 = new Node();
21                 temp1 = head;
22                 while ( temp1->next != NULL) temp1 = temp1->next;
23                 temp1->next = temp;
24         }
25         return head;
26 }
27 Node* Insert_Head (Node* head, int data) {
28         Node* temp = new Node();
29         temp->data = data;
30         temp->next = NULL;
31         if(head != NULL) temp->next = head;
32         head = temp;
33         return head;
34 }
35 Node* Insert_Nth(Node* head, int data, int position) {
36     Node* temp1 = new Node();
37     temp1->data = data;
38     temp1->next = NULL;
39     if (position == 1) {
40         temp1->next = head;
41         head = temp1;
42         return head;
43     }
44     Node* temp2 = head;
45     for (int i = 0; i < position-2; i++)
46         temp2 = temp2->next;
47     temp1->next = temp2->next;
48     temp2->next = temp1;
49     return head;
50 }
51 Node* Delete(Node* head, int position) {
52         Node* temp1 = head;
53         if (position == 1) {
54                 head = temp1->next;
55                 delete temp1;
56                 return temp1;
57         }
58         for(int i = 0; i < position-2; i++)
59                 temp1 = temp1->next;
60         Node *temp2 = temp1->next;
61         temp1->next = temp2->next;
62         delete temp2;
63         return temp1;
64 }
65 void Print(Node* head) {
66         cout<<"List is:\n";
67         while ( head != NULL) {
68                 cout << head->data << endl;
69                 head = head->next;
70         }
71         cout << endl;
72 }
```

```
 73  void ReversePrint(Node *head) {
 74          if(head == NULL) return;
 75          ReversePrint(head->next);
 76          cout << head->data << endl;
 77  }
 78  int CompareLists(Node *headA, Node* headB) {
 79          while (headA != NULL && headB != NULL) {
 80                  if (headA->data != headB->data)
 81                          return 0;
 82                  headA = headA->next;
 83                  headB = headB->next;
 84          }
 85          if (headA == NULL && headB == NULL)
 86                  return 1;
 87          else
 88                  return 0;
 89  }
 90  int main() {
 91
 92          Node* headA = NULL;
 93          Node* headB = NULL;
 94          int n,x;
 95          cout << "Getting values for List A using Insert_Head:\nU";
 96          cout << "how many numbers? \n";
 97          cin >> n;
 98          for (int i = 0; i < n; i++) {
 99                  cout <<"enter the number: \n";
100                  cin>>x;
101                  headA = Insert_Head(headA, x);
102                  Print(headA);
103          }
104          cout << "Getting values for List B using Insert_Tail:\n";
105          cout << "how many numbers? \n";
106          cin >> n;
107          for (int i = 0; i < n; i++) {
108                  cout << "enter the number: \n";
109                  cin >> x;
110                  headB = Insert_Tail(headB, x);
111                  Print(headB);
112          }
113          cout << "Printing List A:\n";
114          Print(headA);
115          cout << "Printing List B:\n";
116          Print(headB);
117          cout << "Inserting 0 at position 4 of List A :\n";
118          headA = Insert_Nth(headA, 0, 4);
119          Print(headA);
120          cout << "Inserting 0 at position 4 of List B :\n";
121          headB = Insert_Nth(headB, 0, 4);
122          Print(headB);
123          cout << "Deleting position 3 of List A :\n";
124          Delete(headA, 3);
125          Print(headA);
126          cout << "Deleting position 3 of List B :\n";
127          Delete(headB, 3);
128          Print(headB);
129          cout << "Reverse List A is:\n";
130          ReversePrint(headA);
131          cout << endl;
132          cout << "Reverse List B is:\n";
133          ReversePrint(headB);
134          cout << endl;
135          cout << "Comparing List A and List B\n";
136          int r = CompareLists(headA, headB);
137          if (r == 0) cout << "result = " << r << "\nTherefore the lists are not the same\n";
138          else cout << "result = " << r << "\nTherefore the lists are the same\n";
139          return 0;
140  }
```