Week days are working days
Weekends are hard working days for 'Winners'

SSS - Saturday Sunday Specials

Saturday Sunday Specials of 20 - 9 - 2020.

Write Functions (recursive / non-reccursive) for the following:

Assume all LLs are having data values as integers.

1. Check if a given LL is a palindrome.

. Remove duplicates from a given sorted LL (recursive)

3. Remove duplicates from a given unsorted LL.

4. Delete last occurrence of a duplicate data value node from a given LL (duplicate data values can be there)

Example : input : L = { 1 , 3, 2, 4, 5, 7, 8, 3, 4 , 2, 7, 4, 2 }

output : L = { 1 , 3, 2, 4, 5, 7, 8 , 4 , 2 }

5. Segregate even data values to left and odd data values right of a given LL

input : L = { 4 , 7 , 8, 2, 5, 4, 9, 1, 6 }

output : L = { 4, 8, 2, 4, 6, 7 5 9 1 }

6. insertion sort on a give LL with unique data values

7. reverse a given LL (try for recursion)

8. Swap Kth node from the beginning with Kth node from end in a given LL

input : L = { 4, 9 , 8, 7 , 2 , 6, 1, 5 , 3 }

k = 3 .

output : L = { 4, 9, 1, 7, 2, 6, 8, 5, 3 }

9. Check whether L2 is present in L1.

Example : L1 = { 5, 8, 2, 9, 5, 8, 2, 3, 7, 1 , 6}

L2 = { 8, 2, 3 } output : 1 if L2 = { 8, 2, 7 } output : 0 if L2 = { 7 , 1} output :

10. Given two linked lists, merge their nodes together to make one list, taking nodes alternatively between the two lists. If either list runs out, all the nodes should be taken from the other list.

Example :

Input : L1 = { 1 , 3, 5 } L2 = { 2, 4, 6, 8, 9}

Output : L3 = { 1, 2, 3, 4, 5, 6, 8, 9 }

Sequence (cin>>   , cout <<) to be followed to pass test case:

Create a Linked list with given numbers termination of input is -1 (you have to read with cin>>)

1. Check if the LL is a palindrome or not . for 'Yes' cout <<   1 for 'No' cout<< 0

Create a Linked list with given numbers termination of input is -1 (you have to read with cin>>)

.2. Remove duplicates from the sorted LL (recursive)

Print all data values (you have to use cout<<)

Create a Linked list with given numbers termination of input is -1 (you have to read with cin>>)

3. Remove duplicates from a given unsorted LL.

Print all data values (you have to use cout<<)

Create a Linked list with given numbers termination of input is -1 (you have to read with cin>>)

4. Delete last occurrence of a duplicate data value node from a given LL (duplicate data values can be there)

Print all data values (you have to use cout<<)

Create a Linked list with given numbers termination of input is -1 (you have to read with cin>>)

5. Segregate even data values to left and odd data values right of a given LL

Print all data values (you have to use cout<<)

Create a Linked list with given numbers termination of input is -1 (you have to read with cin>>)

6. insertion sort on a give LL with unique data values

Print all data values (you have to use cout<<)

Create a Linked list with given numbers termination of input is -1 (you have to read with cin>>)

7. reverse a given LL (try for recursion)

Print all data values (you have to use cout<<)

Create a Linked list with given numbers termination of input is -1 (you have to read with cin>>)

Cin >> k

8. Swap Kth node from the beginning with Kth node from end in a given LL

Print all data values (you have to use cout<<)

Create two Linked lists with given numbers termination of input is -1 (you have to read with cin>> till -1 for L1, and you have to read with cin>> till -1 for L2)

9. Check whether L2 is present in L1.

if L2 is present in L1 cout<< 1 if L2 is not present cout<< 0

Create two Linked lists with given numbers termination of input is -1 (you have to read with cin>> till -1 for L1, and you have to read with cin>> till -1 for L2)

10. Given two linked lists, merge their nodes together to make one list, taking nodes alternatively between the two lists. If either list runs out, all the nodes should be taken from the other list.

Print all data values of L3 (you have to use cout<<)