CS1 LAB - More Linked List

This code has to be submitted by the deadline. It is highly recommended that you complete it during the lab time.

Adding more functions to Linked List

In the lecture, we have learned how to insert an item to the beginning or to the end or between nodes (for sorted) of a linked list. In this lab, you will add some more functionalities to the code.

Before starting, copy the code SinglyLinkedListInsert_Delete.c file from webcourses and remove all the menus and modify the code to have only the following menu :

Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit:

Remove all unnecessary functions such as delete and insert to the end functions. Then add the following functions and call them appropriately based on the menu in the main function.

1. **node* reverse(node* head):** Write a function that takes the head of a linked list and then reverse the list. Finally, it returns the new head of the linked list. For example, if you pass the following linked list to the function: 10->20->15->17->NULL, the linked list will be converted into this: 17->15->20->10->NULL. Here is the hint:
    a. Let's say the list passed to the function is called main_list
    b. Use another node pointer called reversed_list and put the head the main list to the reversed_list and put NULL next to it as the head of main_list will be at the end of the reversed_list
    c. Then keep moving the main_list to forward and for each of the node in the main_list, put it to the reverse_list's front using a temp node.

2. **void insertToPlace(node* head, int val, int place):** Write a function that takes in a pointer to the head of a linked list, a value to insert into the list, *val*, and a location in the list in which to insert it, *place*, which is guaranteed to be greater than 1, and does the insertion. If *place* number of items aren't in the list, just insert the item in the back of the list. You can assume that the linked list into which the inserted item is being added is not empty. Here is the hint:
    a. Before insertion create a temp node and fill-up the data
    b. Use a counter variable and traverse the list until you reach to the end or you reach to the place
    c. As soon as you stop traversing, add the temp node after that.
    d. In the main function, display the list after reversing it by calling the display function.

Sample input/output 1:

(this is a blank line)
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 1

Enter data (an integer): 1

Printing your linked list.....1
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 1

Enter data (an integer): 2

Printing your linked list.....2 1
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 1

Enter data (an integer): 6

Printing your linked list.....6 2 1
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 1

Enter data (an integer): 4

Printing your linked list.....4 6 2 1
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 2
List reversed.

Printing your linked list.....1 2 6 4
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 3

Enter data (an integer) and place (>1) separated by space: 10 3

Printing your linked list.....1 2 10 6 4
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 0

GOOD BYE>>>>
(a blank line)

Sample Input/Output 2:

(this is a blank line)
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 1

Enter data (an integer): 4

Printing your linked list.....4
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 1

Enter data (an integer): 6

Printing your linked list.....6 4
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 1

Enter data (an integer): 2

Printing your linked list.....2 6 4
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 1

Enter data (an integer): 1

Printing your linked list.....1 2 6 4
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 3

Enter data (an integer) and place (>1) separated by space: 10 1

List is empty or place is not valid
Printing your linked list.....1 2 6 4
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 3

Enter data (an integer) and place (>1) separated by space: 10 8

Printing your linked list.....1 2 6 4 10
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 2
List reversed.

Printing your linked list.....10 4 6 2 1
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 0

GOOD BYE>>>>
(a blank line)

Sample input/output 3:

(a blank line)
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 1

Enter data (an integer): 15

Printing your linked list.....15
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 2
List reversed.

Printing your linked list.....15
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 3

Enter data (an integer) and place (>1) separated by space: 60 2

Printing your linked list.....15 60
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 0

GOOD BYE>>>>
(a blank line)

Sample input/output 4 (testing seg fault in reverse function):

Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 2
List reversed.

Printing your linked list.....
Menu: 1. insert at front, 2. reverse list 3. Insert to place 0. exit: 0

GOOD BYE>>>>