Rhea Mae Edwards

Assignment #5 Design/Testing

Understanding the Problem

This problem is asking me to create a program that would take the required code that was given to me in the information for the assignment, and create the code for the given functions that have been previously determined, and the code must also supplement the given and required struct and code of the main function for the main file of this program. Overall, the assignments focuses on the basics of creating, sorting, and deleting a general linked list in the program language of C.

Devising a Plan/Design

Create a header file named list.h, which will contain the struct named node containing the members of an integer named val and a struct node pointer named next, and which will also contain the function declaration that have also been given that are the following:

int length(struct node *)
void print(struct node *)
struct node * push(struct node *, int)
struct node * append(struct node *, int)
struct node * clear(struct node *)
struct node * remove1(struct node *, int)
struct node * sort_ascending(struct node *)
struct node * sort_descending(struct node *)
struct node * insert_middle(struct node *, int, int)

Create a C file named list.c that will contain all the required functions along with their needed functionality, the list.h file, and the needed libraries for the functions.

int length(struct node *n) function will determine the length of the linked list with a while loop, and return the determined value as an integer.

void print(struct node *n) function will have the functionality to print out the passed linked list by using a while loop.

struct node * push(struct node *n, int v) function will take the inputted integers from the user and include them as a part of the linked list being created to the begin of the list.

struct node * append(struct node *n, int v) function will take the inputted integers from the user and include them as a part of the linked list being created to the end of the list by using an if statement if the length of the list is zero, and an else statement to account for a list with a greater length, which will contain a while loop to locate the end of the list to include the inputted integer to the list. This function will also include the code that will commented out in order not to confused the complier in the end, that will provided the person who will be grading my assignment the functionally the function could have when the linked list includes a tail pointer at the end of the list.

struct node * clear(struct node *n) function will have the functionality to clear the final linked list created by the inputs of the user, when the user decides to create another linked list. In order to do this will require a while loop in order to delete the list by going through each node of the list.

struct node * remove1(struct node *n, int v) function will have the functionality to delete a certain node or nodes that contain an integer that user would like to remove from the linked list by using a while statement and an if-else statement.

struct node * sort_ascending(struct node *n) function will have the functionality to sort the linked list in ascending order with an algorithm that will compare the values of a selected node and the node following that initial node of the list, and switching those values in a sense when the second node contains a value smaller than the first node's value, which will be done with a couple of while statements and an if statement.

struct node * sort_descending(struct node *n) function will have the functionality to sort the linked list in ascending order with an algorithm that will compare the values of a selected node and the node following that initial node of the list, and switching those values in a sense when the second node contains a value larger than the first node's value, which will be done with a couple of while statements and an if statement.

struct node * insert_middle(struct node *n, int i, int j) function will have the functionality to insert a value to a certain position given by the user to the linked list by using an if-else statement, a for statement, and the previously declared push function and other previously declared functionalities.

Create a C file named test_list.c that will contain the main function of the program, which was also given as a part of the information for the assignment, which gives the functionality of what the program will do overall, providing a face to the user, and what I as the programmer will have to work around in order in creating the functionalities for each of the required functions for this program.

How did your design for the Linked List in Assignment #5 change during implementation?

Certain previously thought of designing did not account for the relationships between each function and the main function of the program. Previously thought of statements for certain functions weren't necessary or needed to be change slightly or a lot in order for the functionality of what it was supposed to do to work. Overall, the design initial to the final stages of the program had its fair amount of changes and non-changes at the end.

Testing

What were the actual values from your testing? Did these match your expected values?

Input Values	Expected Output	Did Actual Meet Expected?
num = 5	Continue on prompting the user if they would like to enter another number to add to the linked list.	Yes
ans = y	Allow user to input another number.	Yes
num = 7	Continue on prompting the user if they would like to enter another number to add to the linked list.	Yes
ans = n	Continue on prompting the user how they would like the linked list to be sorted.	Yes
ans = a	Print out the linked list in ascending order.	Yes
ans = d	Print out the linked list in descending order.	Yes
ans = y	Allow user to go through the process of inputting values for a linked list again by starting off by prompting the user to enter a number.	Yes
ans = n	Exit out of the program.	Yes

num = f	Will not allow the user to input anymore integers.	Yes
ans = 1	Will not allow the user to input anymore integers.	Yes
ans = 2	Print out the linked list as it was inputted by the user (unsorted), which will only depend on if the program was coded to push or append the values as they were being inputted by the user.	Yes
ans = 3	Exit out of the program.	Yes

What did you do to make sure you get the expected values?

I would take another look at the code to initially see what the logical error to it was, and if I was unable to figure it out just by looking through it, I would consult the pdf and notes found online or through lecture notes that have been gone through in class, and I was unable to make progress by doing that, I would ask a peer to help me out or go to a TA during office hours or during my lab time to get the expected values by resolving the issue at hand.