HW2   Tanmay Sardesai   1001094616

Compilation:

```
gcc -o instr instructor_client.c list_hw.c
./instr
```

To run different test cases instructions are given in the client file in main().

Task 1

Function	Test case	Data/code	Does my code handle it?
sublist(list A, list pos_list)	Index out of bounds	A: 10 ->10 ->40 ->20 pos_list: <u>(**-7**)</u> -> 3 or pos_list: 3 -> <u>**80000**</u> -> 3 result: fct returns NULL	Yes
	A is NULL	list A = NULL; result: fct returns NULL	Yes
	A is empty	list A = newList(); result: fct returns NULL	Yes
	pos_list is empty	list pos_list = NULL; result: fct returns NULL	Yes
	pos_list is NULL	list pos_list = newList(); result: fct returns NULL	Yes
	A is not modified by sublist(…) ….	A: 15 -> 100 -> 7 -> 5 -> 100 pos_list: 3 -> 0 ->2 result: A will still be : 15 -> 100 -> 7 -> 5 -> 100	Yes
	Normal data (as in hw writeup)	A: 15 -> 100 -> 7 -> 5 -> 100 -> 7 -> 30 pos_list: 3 -> 0 -> 6 -> 4	yes
	Repeated position	A: 5 pos_list: 0 -> 0 -> 0 result: returns: 5-> 5-> 5	yes

HW2 Tanmay Sardesai 1001094616

deleteOccurrences (list A, int V)	Normal data, V is in A (as in hw write-up)	A: 15 -> 100 -> 7 -> 5 -> 100 -> 7 -> 30 V is 7, Result: A will become: 15-> 100-> 5 -> 100 -> 30	yes
	V does not occur in A	A: 15 -> 100 -> 7 -> 5 V is 9, Result: A does not change: 15-> 100-> 7-> 5	yes
	Repeated consecutive occurrences	A: 15 -> 7 -> 7 -> 5 V is 7, Result: A becomes: 15 -> 100	yes
	A has one item and that is V	A: 7 V is 7 Result: A becomes Empty	yes
	A has only items with value V in it	A: 7->7-> 7 V is 7 Result: A becomes empty	yes
	A is NULL	A = NULL Result: A is not changed	yes
	A is empty	A = newList() Result: A is not changed	yes
insertAtPosition (list A, Item val, int P)	Normal data (as in hw write-up)	A: 15 -> 100 -> 5 -> 100 -> 30 val = 12, P = 0 Result: A will become: 12-> 15-> 100-> 5-> 100-> 30	Yes
	A is NULL	A = NULL Result: A is not changed	Yes

	A is empty	A = newList()
		Val = 12, pos = 0
		Result: A will have item 12.
			yes

	Position is greater than length of A	A: 2->3,val = 12, P = 5 Result: will insert at the end of A. A becomes:	yes

HW2 Tanmay Sardesai 1001094616

		2->3->12	
	Position is negative	A: 2->3,val = 12, P = -2 Result: will insert at the beginning of A. A becomes: 12 -> 2 ->3	yes
moveAllMaxAtEnd (list A)	A is NULL	A = NULL Result: A is not changed	yes
	A is empty	A = newList() Result: A is not changed	yes
	Normal data (as in hw write-up)	A: 15 ->100-> 5 ->100-> 30 Result: A will become: 15 -> 5 -> 30-> 100 -> 100	yes
	A has one item	A: 7 Result: A does not change	yes
	A has only items of the same value in it (all items are MAX).	A: 7-> 7 ->7 Result: A does not change (the order of the nodes does not change either)	yes
	MAX is on first position	A: 100-> 7->20 Result: A: 7->20->100	yes
	MAX is on last position	A: 10-> 7->200 Result: A: 10->7->200	Yes

Task 2: (The answer for this task may also be written in the source code.)

(1) sublist(list A, list pos_list)

Let n = length of A, p = length of pos_list

Then, time complexity = theta(np)

(2) deleteOccurrences(list A, int V)]

Let n = length of A

Then, time complexity = theta(n)

HW2   Tanmay Sardesai   1001094616

(3) insertAtPosition(list A, Item val, int P)

    Let n = length of A

    Then, time complexity = theta(n)

(4) moveAllMaxAtEnd(list A)

    Let n = length of A

    Then, time complexity = theta(n^2)