Quest09

Track Bootcamp C Arc 02

Subject

Quest09

Remember to git add && git commit && git push each exercise!

We will execute your function with our test(s), please DO NOT PROVIDE ANY TEST(S) in your file

For each exercise, you will have to create a folder and in this folder, you will have additional files that contain your work. Folder names are provided at the beginning of each exercise under `submit directory` and specific file names for each exercise are also provided at the beginning of each exercise under `submit file(s)` .

Quest09	Reverse Linked List
Submit directory	ex00
Submit file	reverse-linked-list.c

Description

Reverse a singly linked list.

Example:

Control Center

Also working on the project

diallo_m

Type	Project
Group Size	1 Participant
Review system	Test Review (Gandalf)
Difficulty	Initiation
Average duration	1 Week

Project's Metadata

Track	Project
id: 1205	id: 140
name: Bootcamp C Arc 02	name: quest09
visible: True	visible: True

```
Input: 1->2->3->4->5->NULL
Output: 5->4->3->2->1->NULL
```

Function prototype (c)

```c
/*
**
** QWASAR.IO -- reverse_linked_list
**
** @param {listnode*} param_1
**
** @return {listnode*}
**
*/
#ifndef STRUCT_LISTNODE
#define STRUCT_LISTNODE
typedef struct s_listnode
{
    int val;
    struct s_listnode* next;
} listnode;
#endif

listnode* reverse_linked_list(listnode* param_1)
{

}
```

Quest09	Remove Nth Node From End Of List
Submit directory	ex01
Submit file	remove-nth-node-from-end-of-list.c

Description

Given a linked list, remove the *n*-th node from the end of list and return its head.

Example:

```
Given linked list:
1->2->3->4->5, and _n_ = 2.
After removing the second node from the end, the linked list becomes 1->2->3->5.
```

Note:

```
Given _n_ will always be valid.
```

Function prototype (c)

```c
/*
**
** QWASAR.IO -- remove_nth_node_from_end_of_list
**
** @param {listnode*} param_1
** @param {int} param_2
**
** @return {listnode*}
**
*/
#ifndef STRUCT_LISTNODE
#define STRUCT_LISTNODE
typedef struct s_listnode
{
    int val;
    struct s_listnode* next;
} listnode;
#endif


listnode* remove_nth_node_from_end_of_list(listnode* param_1, int param_2)
{

}
```

Quest09	Remove Duplicates From Sorted List

Submit directory	ex02
Submit file	remove-duplicates-from-sorted-list.c

Description

Given a sorted linked list, delete all duplicates such that each element appear only *once*.

Example 00:

```
Input: 1->1->2
Output: 1->2
```

Example 01:

```
Input: 1->1->2->3->3
Output: 1->2->3
```

Function prototype (c)

```c
/*
** 
** QWASAR.IO -- remove_duplicates_from_sorted_list
**
** @param {listnode*} param_1
**
** @return {listnode*}
**
*/
#ifndef STRUCT_LISTNODE
#define STRUCT_LISTNODE
typedef struct s_listnode
{
    int val;
    struct s_listnode* next;
} listnode;
#endif


listnode* remove_duplicates_from_sorted_list(listnode* param_1)
{

}
```

Quest09	Merge K Sorted Lists
Submit directory	ex03
Submit file	merge-k-sorted-lists.c

Description

Merge *k* sorted linked lists and return it as one sorted list. Analyze and describe its complexity.

Example:

```
Input: [1->4->5, 1->3->4, 2->6]
Output: 1->1->2->3->4->4->5->6
```

Function prototype (c)

```c
/*
**
** QWASAR.IO -- merge_k_sorted_lists
**
** @param {listnode_array*} param_1
**
** @return {listnode*}
**
*/
#ifndef STRUCT_LISTNODE
#define STRUCT_LISTNODE
typedef struct s_listnode
{
    int val;
    struct s_listnode* next;
} listnode;
#endif

#ifndef STRUCT_LISTNODE_ARRAY
#define STRUCT_LISTNODE_ARRAY
typedef struct s_listnode_array
{
    int size;
    listnode **array;
} listnode_array;
#endif


listnode* merge_k_sorted_lists(listnode_array* param_1)
{

}
```