Lab 05: Single Link List

In this assignment you are going to implement a set functions for a single link list.

Note: ***Students must demonstrate their lab in the following week of November 23rd, 2020 to get the grade. This lab is worth 20marks. A Missed demonstration will result in 25% grade deduction. Late submission and late demonstration to the lab is welcomed but with a loss of marks. That is, you do not earn full marks for late submission. 10% penalty for every day late up to 50%. Work will not be accepted after 10 days.***
You must talk to your lab professor in case of any unprecedented situation which may result in late submission.
You are required to submit the properly commented (as per assignment submission standards doc on BrightSpace) text file of your code along with a screenshot of the output.

Given the following data type:

```
typedef struct node {
 int data;
 struct node *next;
}node_t;
```

and the given functions:

```
/*************************************************************/
/* Provided functions
 * llist_push:  Add an element to the head of the list
 * llist_pop:   Removes first element of the list
 * llist_size:  Returns number of elements in the list
 *************************************************************/
```

Implement the following function by the end of the lab:

```
/*************************************************************/
/* Functions you need to implement
 * 1> EASY functions: no need to modify the list
 * llist_contains:  Returns 1 if an element is contained in the list
 *                  0 otherwise
 * llist_count:     Returns the number of times an element is in the list
 * llist_find:      Returns the index of the first element given it finds,
 *                        -1 if not found
 * llist_is_equal:  Returns 1 if lists are the same, 0 if not
 *                  Two lists are the same if
 *                  they have the same number of elements
 *                  and the elements are presented in the same order
 *
```

```
* 2> MEDIUM functions: needs to modify the list
* llist_addlast:    Adds an element to the tail of the list
* llist_insert:     Adds an element at index
* llist_remove:     Removes first instance of value
* llist_free:       Free all elements in the list
*****************************************************************/
```

You MUST implement the ***llist_free( )*** function as it is required in all tests.

By the end of the lab you must demonstrate the passing of at least **5** tests, including at least 2 easy function and 2 medium functions.

Starting code, ***makefile*** and tests are provided.