2) (10 pts) DSN (Linked Lists)

Complete the following user defined function that reconstructs the structure of a singly linked list. The function will take a pointer to the head of some list along with the number of nodes `n` and move the first half the nodes to the back of the list while the other half moves up to the front. **For grading purposes, please write your solution iteratively, with NO recursion.** The function will return the new head of the list. **You may assume that the original list has an even number of elements and is not empty.** The following figure shows the scenario.

Before flipHalf

After flipHalf

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

node_t * flipHalf(node_t * head, int n) {


    node_t* end = head;
    for (int i=0; i<n/2-1; i++)
        end = end->next;

    node_t* newfront = end->next;

    end->next = NULL;

    node_t* tmp = newfront;
    while (tmp->next != NULL)
        tmp = tmp->next;

    tmp->next = head;
    return newfront;
}
```

Grading: 5 pts total – identify beginning and end of list to be returned.
1 pt – putting NULL at the end of the list to be returned.
2 pts – iterating to end of the original list.
1 pt – linking end of original list to beginning of original list.
1 pt – returning a pointer to the new front of the list.