CSE251 System Programing (Fall 2020)
Instructors: Prof. Sam H. Noh
TAs: Hyojae Yun, Jungbin Kim, Shean Kim and Ethan Park
Due date: 11:59 PM, Sep 18, 2020.

Assignment 1: Double Linked List

In this assignment, you need to implement a double linked list in `list.c`. Struct Node is already implemented for you. Here is a description of functions you need to implement. Default type will be `double`.

- void init (struct List* list) : init the member variables of list
- struct Node* front (struct List* list) : access the first element
- struct Node* back (struct List* list) : access the last element
- int empty (struct List* list) : check whether the list is empty
- int size (struct List* list) : return the number of elements
- void insert (struct List* list, int pos, Data val) : create a node with the element and inserts it before pos
- void erase (struct List* list, int pos) : erase an element
- void push_front (struct List* list, Data val) : add an element at the front
- void pop_front (struct List* list) : erase the first element
- void push_back (struct List* list, Data val) : add an element at the back
- void pop_back (struct List* list) : erase the last element
- void clear (struct List* list) : erase the all elements
- struct Node* find (struct List* list, Data data) : find an element with given data
- void forward_sum (struct List* list, int n) : go forward and sum every n nodes. If the remainders are less than n, sum the remaining ones. Output format is described in Figure 3.
- void backward_sum (struct List* list, int n) : go backward and sum every n nodes. If the remainders are less than n, sum the remaining ones. Output format is described in Figure 3.

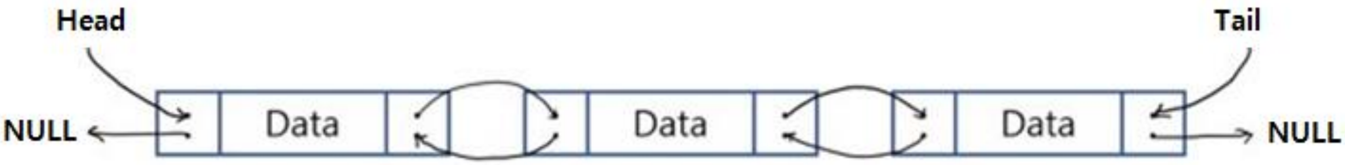


Figure 1 Double linked list

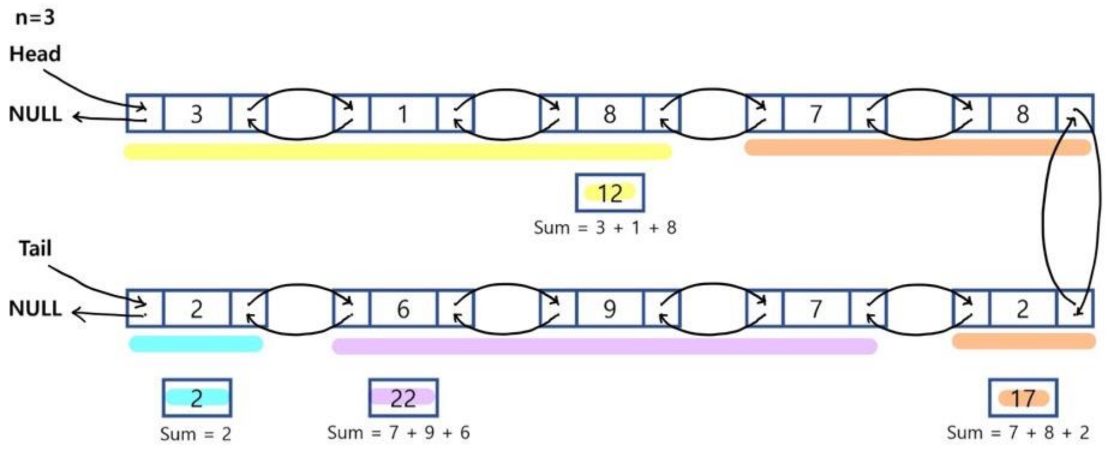
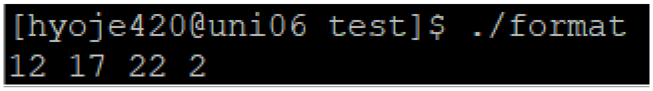


Figure 2 forward sum (n=3)



```
[hyoje420@uni06 test]$ ./format
12 17 22 2
```

Figure 3 Output format of forward_sum and backward_sum (in case of Figure2)

IMPORTANT:

- Check correctness of your program **on uni server** before final submission.
  - Compile option: `gcc –o main main.c list.c`
  - **No Windows** for final correctness validation.
  - **No Code::Blocks** for final correctness validation.
- Address any **segmentation fault** or **compile error** you encounter.
- You **MUST include comments** in the program.
- Your program must not print anything other than what this document asks it to do so.
- **No additional headers** allowed for this assignment.
- You can only edit where we explicitly allowed you to do so. The places you can edit are marked with a comment and you must follow the direction.
- However, we will not use main.c when grading, so feel free to edit it in any way you want.
- All descriptions on this document are subject to be clarified, supplemented, and/or changed later via announcements on Piazza.
- **Do not** ask us via email. Post your questions on Piazza and share them with your colleagues. Also, you **MUST express** your point in the title so that other students can find your question helpful.