Weekly Homework 4

April 9, 2025

1 Linkedlist

Given the following `Linkedlist` definition:

```
struct NODE{
    int key;
    NODE* p_next;
};
```

```
struct List{
    NODE* p_head;
    NODE* p_tail;
};
```

Complete the following functions to fulfill the given requirements:

1. Initialize a `NODE` from a given integer:
   - `NODE* createNode(int data)`
2. Initialize a `List` from a given `NODE`:
   - `List* createList(NODE* p_node)`
3. Insert an integer to the head of a given `List`:
   - `bool addHead(List* &L, int data)`
4. Insert an integer to the tail of a given `List`:
   - `bool addTail(List* &L, int data)`
5. Remove the first `NODE` of a given `List`:
   - `bool removeHead(List* &L)`
6. Remove the last `NODE` of a given `List`:
   - `void removeTail(List* &L)`
7. Remove all `NODE` from a given `List`:
   - `void removeAll(List* &L)`
8. Remove a `Node` Before with a given value `List`:
   - `void removeBefore(List* &L, int val)`
9. Remove an integer after a value of a given `List`:
   - `void romveAfter(List* &L, int val)`
10. Insert an integer at a position of a given `List`:
    - `bool addPos(List* &L, int data, int pos)`
11. Remove an integer at a position of a given `List`:
    - `void RemovePos(List* &L, int data, int pos)`
12. Insert an integer before a value of a given `List`:
    - `bool addBefore(List* &L, int data, int val)`
13. Insert an integer after a value of a given `List`:
    - `bool addAfter(List* &L, int data, int val)`
14. Print all elements of a given `List`:
    - `void printList(List* L)`
15. Count the number of elements `List`:
    - `int countElements(List* L)`
16. Create a new `List` by reverse a given `List`:
    - `List* reverseList(List* L)`
17. Remove all duplicates from a given `List`:
    - `void removeDuplicate(List* &L)`
18. Remove all key value from a given `List`:
    - `bool removeElement(List* &L, int key)`

2 Doubly Linkedlist

Following is representation of a doubly linked list:

```
struct d_NODE{
    int key;
    d_NODE* pNext;
    d_NODE* pPrev;
};
```

```
struct d_List{
    d_NODE* pHead;
    d_NODE* pTail;
};
```

Implement functions to execute the operations from singly linkedlist section (section 1).

3 Submission Rules

Students must adhere to the following submission guidelines:

1. Each solution must be submitted in a separate file:
   - `Linkedlist.cpp` for the Linkedlist.
   - `DoublyLinkedlist.cpp` for the Doubly Linkedlist.
2. The program should print all the test passed (figure 1).

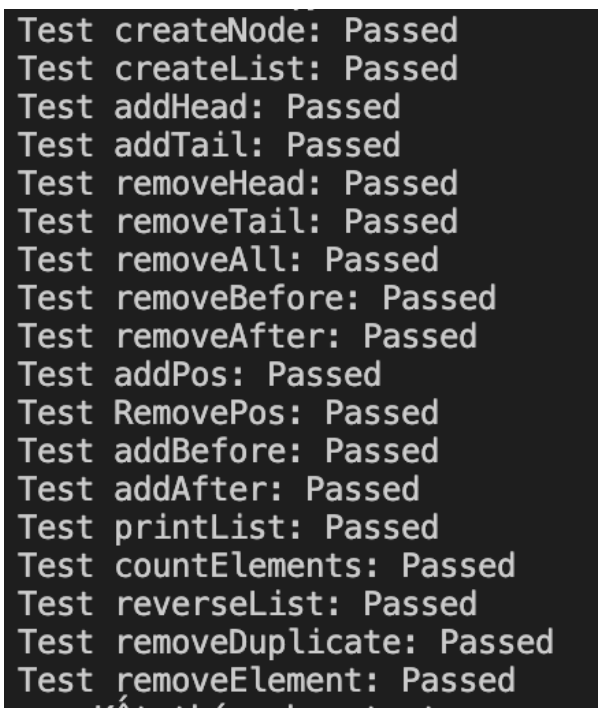

Figure 1: Terminal when execute passed all the test.

3. The submission must be in a **compressed zip file** named **MSSV.zip**, containing:

- The required C++ files. (Linkedlist.cpp, DoublyLinkedlist.cpp).
- A `report.pdf` file describing the approach used in each solution. The image of GitHub home page to verify code is pushed on GitHub
- Don't use $< bits/stdc++.h >$ library