Reverse a doubly linked list

```
#include <bits/stdc++.h>

using namespace std;

class DoublyLinkedListNode {
    public:
        int data;
        DoublyLinkedListNode *next;
        DoublyLinkedListNode *prev;

        DoublyLinkedListNode(int node_data) {
            this->data = node_data;
            this->next = nullptr;
            this->prev = nullptr;
        }
};

class DoublyLinkedList {
    public:
        DoublyLinkedListNode *head;
        DoublyLinkedListNode *tail;

        DoublyLinkedList() {
            this->head = nullptr;
            this->tail = nullptr;
        }

        void insert_node(int node_data) {
            DoublyLinkedListNode* node = new
DoublyLinkedListNode(node_data);

            if (!this->head) {
                this->head = node;
            } else {
                this->tail->next = node;
                node->prev = this->tail;
            }

            this->tail = node;
        }
};
```

```
void print_doubly_linked_list(DoublyLinkedListNode* node, string
sep, ofstream& fout) {
    while (node) {
        fout << node->data;

        node = node->next;

        if (node) {
            fout << sep;
        }
    }
}

void free_doubly_linked_list(DoublyLinkedListNode* node) {
    while (node) {
        DoublyLinkedListNode* temp = node;
        node = node->next;

        free(temp);
    }
}

/*
 * Complete the 'reverse' function below.
 *
 * The function is expected to return an
INTEGER_DOUBLY_LINKED_LIST.
 * The function accepts INTEGER_DOUBLY_LINKED_LIST llist as
parameter.
 */

/*
 * For your reference:
 *
 * DoublyLinkedListNode {
 *     int data;
 *     DoublyLinkedListNode* next;
 *     DoublyLinkedListNode* prev;
 * };
 *
 */

DoublyLinkedListNode* reverse(DoublyLinkedListNode* llist) {
    DoublyLinkedListNode* curr=llist;
    DoublyLinkedListNode* temp=NULL;
    while(curr!=NULL){
```

```
        temp=curr->prev;
        curr->prev=curr->next;
        curr->next=temp;
        llist=curr;
        curr=curr->prev;
    }
    return llist;
}

int main()
{
    ofstream fout(getenv("OUTPUT_PATH"));

    int t;
    cin >> t;
    cin.ignore(numeric_limits<streamsize>::max(), '\n');

    for (int t_itr = 0; t_itr < t; t_itr++) {
        DoublyLinkedList* llist = new DoublyLinkedList();

        int llist_count;
        cin >> llist_count;
        cin.ignore(numeric_limits<streamsize>::max(), '\n');

        for (int i = 0; i < llist_count; i++) {
            int llist_item;
            cin >> llist_item;
            cin.ignore(numeric_limits<streamsize>::max(), '\n');

            llist->insert_node(llist_item);
        }

        DoublyLinkedListNode* llist1 = reverse(llist->head);

        print_doubly_linked_list(llist1, " ", fout);
        fout << "\n";

        free doubly linked list(llist1);
    }

    fout.close();

    return 0;
}
```