Reverse a linked list

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

using namespace std;

class SinglyLinkedListNode {
    public:
        int data;
        SinglyLinkedListNode *next;

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

class SinglyLinkedList {
    public:
        SinglyLinkedListNode *head;
        SinglyLinkedListNode *tail;

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

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

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

            this->tail = node;
        }
};

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

```
        node = node->next;

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

void free_singly_linked_list(SinglyLinkedListNode* node) {
    while (node) {
        SinglyLinkedListNode* temp = node;
        node = node->next;

        free(temp);
    }
}

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

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

SinglyLinkedListNode* reverse(SinglyLinkedListNode* llist) {
    SinglyLinkedListNode* prev=NULL;
    SinglyLinkedListNode* curr=llist;
    while(curr!=NULL){
        SinglyLinkedListNode* next=curr->next;
        curr->next=prev;
        prev=curr;
        curr=next;
    }
```

```
    return prev;
}

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

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

    for (int tests_itr = 0; tests_itr < tests; tests_itr++) {
        SinglyLinkedList* llist = new SinglyLinkedList();

        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);
        }

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

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

        free_singly_linked_list(llist1);
    }

    fout.close();

    return 0;
}
```