4. Linked Lists - Add Two Numbers

You are given two non-empty linked lists representing two non-negative integers. Add the two numbers and return the sum as a linked list.

Code:

#include <bits/stdc++.h>

using namespace std;

struct node{

    int data;

    node\* next;

    node(int val){

        data = val;

        next = NULL;

    }

};

node\* addtwoll(node\* l1, node\* l2){

    node\* resl = new node(0);

    node\* head = resl;

    int carry = 0;

    while(l1!=nullptr || l2!=nullptr || carry>0){

        int x = carry;

        if(l1){

            x+= l1->data;

            l1 = l1->next;

        }

        if(l2){

            x+= l2->data;

            l2 = l2->next;

        }

        carry = x/10;

        resl->next = new node(x%10);

        resl = resl->next;

    }

    return head->next;

}

node\* createll(int n) {

    if (n <= 0) return nullptr;

    int val;

    cout << "Enter " << n << " integers:\n";

    cin >> val;

    node\* head = new node(val);

    node\* tail = head;

    for (int i = 1; i < n; ++i) {

        cin >> val;

        tail->next = new node(val);

        tail = tail->next;

    }

    return head;

}

void printll(node\* head) {

    while (head) {

        cout << head->data;

        if (head->next)

            cout << " -> ";

        head = head->next;

    }

    cout << endl;

}

int main(){

    int n1, n2;

    cin >> n1;

    cin >> n2;

    node\* l1 = createll(n1);

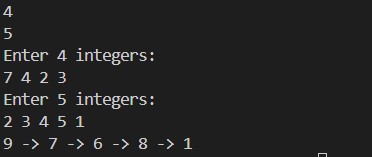
    node\* l2 = createll(n2);

    node\* l3 = addtwoll(l1, l2);

    printll(l3);

}

Output:



5. Reorder List

Reorder a linked list from LOL1...Ln-1-Ln to LO-LnL1-Ln-1L2Ln-2-....

Code:

#include <bits/stdc++.h>

using namespace std;

struct node{

    int data;

    node\* next;

    node(int val){

        data = val;

        next = NULL;

    }

};

node\* reversell(node\* head){

    node\* newl = nullptr;

    while(head){

        node\* tem = head->next;

        head->next = newl;

        newl = head;

        head = tem;

    }

    return newl;

}

void mergell(node\* &head, int n){

    int mid = n/2; //we are using size of linked list to find mid in this case, if size not avalaibale we can find mid point through another function as well

    node\* iter = head;

    node\* midel = head;

    while(mid>0){

        midel = midel->next;

        mid--;

    }

    iter = midel;

    midel = midel->next;

    iter->next = NULL;

    midel = reversell(midel);

    iter = head;

    while(iter && midel){

        node\* tim = iter->next;

        node\* tim2 = midel->next;

        iter->next = midel;

        midel->next = tim;

        iter = tim;

        midel = tim2;

    }

}

node\* createll(int n) {

    if (n <= 0) return nullptr;

    int val;

    cout << "Enter " << n << " integers:\n";

    cin >> val;

    node\* head = new node(val);

    node\* tail = head;

    for (int i = 1; i < n; ++i) {

        cin >> val;

        tail->next = new node(val);

        tail = tail->next;

    }

    return head;

}

void printll(node\* head) {

    while (head) {

        cout << head->data;

        if (head->next)

            cout << " -> ";

        head = head->next;

    }

    cout << endl;

}

int main(){

    int n1;

    cin >> n1;

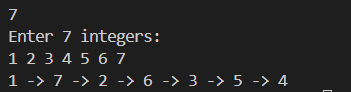
    node\* l1 = createll(n1);

    mergell(l1, n1);

    printll(l1);

}

Output:



6. Remove Nth Node From End of List

Given a linked list, remove the nth node from the end and return its head.

Code:  
#include <bits/stdc++.h>

using namespace std;

struct node{

    int data;

    node\* next;

    node(int val){

        data = val;

        next = NULL;

    }

};

void findnth(node\* &l1, int k){

    node\* front = l1;

    node\* behind = l1;

    for(int i = 0; i<k; i++){

        if(front==nullptr){

            return;

        }

        front = front->next;

    }

    while(front->next){

        front = front->next;

        behind = behind->next;

    }

    node\* tem = behind->next->next;

    behind->next = tem;

}

node\* createll(int n) {

    if (n <= 0) return nullptr;

    int val;

    cout << "Enter " << n << " integers:\n";

    cin >> val;

    node\* head = new node(val);

    node\* tail = head;

    for (int i = 1; i < n; ++i) {

        cin >> val;

        tail->next = new node(val);

        tail = tail->next;

    }

    return head;

}

void printll(node\* head) {

    while (head) {

        cout << head->data;

        if (head->next)

            cout << " -> ";

        head = head->next;

    }

    cout << endl;

}

int main(){

    int n1;

    cin >> n1;

    int k;

    cout << "Enter nth position: ";

    cin >> k;

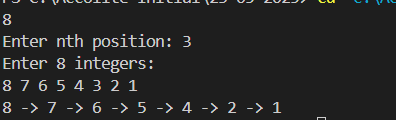
    node\* l1 = createll(n1);

    findnth(l1, k);

    printll(l1);

}

Output:



8. Group Anagrams

Given an array of strings, group the anagrams together.

Code:

#include <bits/stdc++.h>

using namespace std;

vector<vector<string>> groupana(vector<string>& str) {

    unordered\_map<string, vector<string>> mp;

    for (string x : str) {

        string k = x;

        sort(k.begin(), k.end());

        mp[k].push\_back(x);

    }

    vector<vector<string>> res;

    for (auto i : mp) {

        res.push\_back(i.second);

    }

    return res;

}

int main(){

    vector<string> str = {"tame", "meat", "snow", "wons", "care", "star", "race", "rats"};

    vector<vector<string>> ans = groupana(str);

    for(auto x : ans){

        for(auto y : x){

            cout << y << " ";

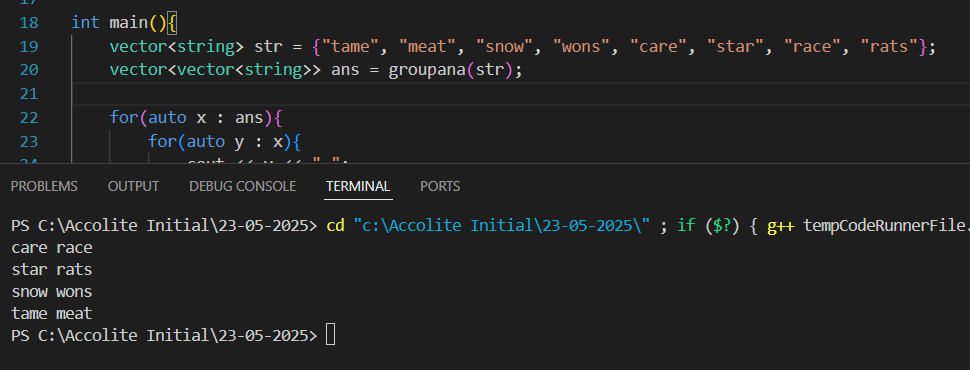
        }

        cout << endl;

    }

}

Output:



9. Rearrange a no to find min possible no in o(n) and constant space.

Code:

#include <bits/stdc++.h>

using namespace std;

int minper(int num){

    vector<int> numcount(10, 0);

    while(num>=1){

        int x = num%10;

        num = num/10;

        numcount[x]++;

    }

    int n = 0;

    for(int i = 1; i<10; i++){

       while(numcount[i]>0){

        n = n\*10 + i;

        numcount[i]--;

       }

    }

    return n;

}

int main(){

    cout << minper(11002201);

}

Output:

