Assignment – 4

Akshay raj singh

BT22CSH054

CODE:-

#include <iostream>

#include <vector>

using namespace std;

class Node {

public:

int data;

Node\* next;

Node(int data) {

this->data = data;

this->next = NULL;

}

};

Node\* list(Node\* head, int d) {

Node\* t = new Node(d);

if (head == NULL)

head = t;

else {

Node\* t1 = head;

t->next = head;

head = t;

}

return head;

}

Node\* create(Node\* head, const string& num) {

int len = num.length();

for (int i = len - 1; i >= 0; i -= 3) {

int d = 0;

int factor = 1;

for (int j = 0; j < 3 && i - j >= 0; j++) {

d += (num[i - j] - '0') \* factor;

factor \*= 10;

}

head = list(head, d);

}

return head;

}

Node\* reverse(Node\* head) {

Node\* current = head;

Node\* prev = NULL;

Node\* Next = NULL;

while (current != NULL) {

Next = current->next;

current->next = prev;

prev = current;

current = Next;

}

head = prev;

return head;

}

Node\* addition(Node\* h1, Node\* h2) {

Node\* add = NULL;

Node\* t1 = h1;

Node\* t2 = h2;

int carry = 0;

while (t1 != NULL || t2 != NULL) {

int sum = carry;

if (t1 != NULL) {

sum += t1->data;

t1 = t1->next;

}

if (t2 != NULL) {

sum += t2->data;

t2 = t2->next;

}

carry = sum / 1000;

add = list(add, sum % 1000);

}

if (carry > 0) {

add = list(add, carry);

}

return reverse(add);

}

void print(Node\* head) {

Node\* temp = head;

while (temp != NULL) {

cout << temp->data;

temp = temp->next;

}

cout << endl;

}

int main() {

Node\* n1 = NULL;

Node\* n2 = NULL;

n1 = create(n1, "123456789123456789123456789");

n2 = create(n2, "987654321987654321987654321");

print(n1);

print(n2);

n1 = reverse(n1);

n2 = reverse(n2);

Node\* add = addition(n1, n2);

add = reverse(add);

cout << "Sum:" << endl;

print(add);

return 0;

}

Time complexity :- O(max(m, n)) , where 'm' and 'n' are the lengths of the input linked lists