Question: Implement a singly linked list in c++ for storing 10 numbers and implement the following operation of linked list:   
1. Insertion of node at beginning of the list   
2. Insertion of node at end of the list   
3. Insertion of node at a specific node position.   
4. Deletion of node at beginning of the list   
5. Deletion of node at end of the list   
6. Deletion of node at a specific node position   
7. Display a singly linked list stored item.

#include <iostream>

using namespace std;

class node{

public:

int data;

node\* next;

node\* head = NULL;

}; //for data insertion

class list : public node {

public:

int count = 1;

node\* create\_node() {

node\* newnode = new node;

return newnode;

}

void read(node\* newnode) {

cout << "ENTER THE DATA : - "; cin >> newnode->data; cout << endl;

}

void insert\_at\_head(node\* newnode) {

node\* temp = head;

if (head == NULL) {

head = newnode;

newnode->next = NULL;

}

else {

newnode->next = temp;

head = newnode;

}

count++;

}

void insert\_at\_end(node\* newnode){

node\* temp = head;

if (head == NULL) {

head = newnode;

newnode->next = NULL;

}

else {

while (temp->next != NULL) {

temp = temp->next;

}

temp->next = newnode;

newnode->next = NULL;

}

count++;

}

void insert\_at\_specific\_node(node\* newnode, int n\_n) {

node\* temp = head;

int count\_=1;

if (head == NULL) {

cout << "\n NO NODES INSERTED YET" << endl;

}

else {

if (count >= count\_) {

while (count\_ != n\_n) {

temp = temp->next;

count\_++;

}

if (count\_ == n\_n) {

node\* temp\_p = temp;

newnode->next = temp->next;

temp\_p->next = newnode;

}

}

}

}

void swap(node\* prev) {

node\* curr = head;

if (head == NULL) {

cout << "\n NOTHING TO SWAP" << endl;

}

else if (head->next == NULL) {

cout << "\nNOTHING TO SWAP" << endl;

}

else {

while (curr->next != NULL) {

if(curr->data>curr->next->data){

int data\_temp = curr->data;

curr->data = curr->next->data;

curr->next->data = data\_temp;

curr = curr->next;

}

else {

curr = curr->next;

}

}

}

}

void display() {

cout << "DISPLAYING DATA" << endl;

node\* temp = head;

if (head == NULL) {

cout << "\n NOTHING TO DISPLAY" << endl;

}

else {

while (temp != NULL) {

cout << temp->data << endl;

temp = temp->next;

}

cout << "\n \n";

}

}

void delete\_at\_head() {

node\* temp = head;

if (head == NULL) {

cout << "\n NOTHING TO DELETE" << endl;

}

else {

head = temp->next;

}

count--;

}

void delete\_at\_end() {

node\* temp = head;

if (head == NULL) {

cout << "\n NOTHING TO DELETE" << endl;

}

else {

while (temp->next->next != NULL) {

temp = temp->next;

}

temp->next = NULL;

}

count--;

}

void delete\_at\_node(int n\_n) {

node\* temp = head;

node\* prev = temp;

int count\_ = 0;

if (head == NULL) {

cout << "NOTHING TO DELETE" << endl;

}

else {

if (n\_n < count) {

while (count\_ != n\_n) {

prev = temp; //to store the prev node of temp->next

temp = temp->next; //moves temp to next but the current temp before moving is stored in prev :)

count\_++;

}

if (count\_ == n\_n) {

if(temp == NULL){ //i.e temp is prev->next

head = NULL;

}

else {

prev->next = temp->next;

}

}

}

}

count--;

}

}; //for insertion of node

int main() {

list l;

cout << "WELCOM" << endl;

while(1) {

int option;

node n;

cout << "ENTER 1 TO INPUT USERS DATA" << endl;

cout << "ENTER 2 DELETE A NODE" << endl;

cout << "ENTER 3 TO DISPLAY DATA" << endl;

cin >> option;

switch(option) {

case 1:

{

int o1;

cout << "\n ENTER 1 TO INSERT AT HEAD" << endl;

cout << "\n ENTER 2 TO INSERT AT END" << endl;

cout << "\n ENTER 3 TO INSERT AT SPECIFIC NODE POISITION" << endl;

cin >> o1;

node\* newnode;

switch (o1) {

case 1:

{

newnode = l.create\_node();

l.read(newnode);

l.insert\_at\_head(newnode);

break;

}

case 2:

{

newnode = l.create\_node();

l.read(newnode);

l.insert\_at\_end(newnode);

break;

}

case 3:

{

newnode = l.create\_node();

l.read(newnode);

int n\_n;

cout << "ENTER THE NODE NUMBER : "; cin >> n\_n; cout << endl;

l.insert\_at\_specific\_node(newnode, n\_n);

break;

}

}

break;

}

case 2:

{

int opt;

cout << "ENTER 1 TO DELETE AT BEGINNING" << endl;

cout << "ENTER 2 TO DELETE AT END" << endl;

cout << "ENTER 3 TO DELETE AFTER SPECIFIC NODE" << endl;

cin >> opt; cout << endl;

switch (opt) {

case 1:

{

l.delete\_at\_head();

break;

}

case 2:

{

l.delete\_at\_end();

break;

}

case 3:

{

int inp;

cout << "\n ENTER THE NODE NUMBER AFTER WHICH YOU WANT TO DELETE" << endl;

cin >> inp; cout << endl;

l.delete\_at\_node(inp);

break;

}

}

break;

}

case 3:

{

l.swap(n.head);

l.display();

break;

}

}

}

}

