NAME AHMED AZIZ

SAP ID 55223

SECTION SE3-2

TASK1:

```
#include <iostream>

using namespace std;

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

    Node(int val) {
        data = val;
        next = nullptr;
    }
};

class LinkedList {
private:
    Node* head;

public:
    LinkedList() {
        head = nullptr;
```

```
}

void insert(int data) {
    Node* newNode = new Node(data);
    if (!head) {
        head = newNode;
    } else {
        Node* current = head;
        while (current->next) {
            current = current->next;
        }
        current->next = newNode;
    }
}

void deleteNode(int key) {
    Node* current = head;
    Node* previous = nullptr;

    while (current && current->data != key) {
        previous = current;
        current = current->next;
    }

    if (!current) {
        cout << key << " not found in the list." << endl;
        return;
    }
```

```
    if (previous == nullptr) {
        head = current->next;  // Deleting head
    } else {
        previous->next = current->next;  // Bypass the node to be deleted
    }

    delete current;  // Free memory
    cout << key << " deleted from the list." << endl;
}

bool search(int key) {
    Node* current = head;
    while (current) {
        if (current->data == key) {
            return true;
        }
        current = current->next;
    }
    return false;
}

void display() {
    Node* current = head;
    while (current) {
        cout << current->data << " -> ";
        current = current->next;
    }
    cout << "nullptr" << endl;
}
```

```
};

int main() {
    LinkedList linkedList;
    int value;

    // Taking user input for 5 nodes
    cout << "Enter 5 integers to create the linked list:" << endl;
    for (int i = 0; i < 5; ++i) {
        cin >> value;
        linkedList.insert(value);
    }

    // Display the list
    cout << "Linked List: ";
    linkedList.display();

    // Insert a new node
    cout << "Enter a value to insert into the linked list: ";
    cin >> value;
    linkedList.insert(value);
    cout << "Linked List after insertion: ";
    linkedList.display();

    // Delete a node
    cout << "Enter a value to delete from the linked list: ";
    cin >> value;
    linkedList.deleteNode(value);
    cout << "Linked List after deletion: ";
```

```
    linkedList.display();

    // Search for a value
    cout << "Enter a value to search in the linked list: ";
    cin >> value;
    if (linkedList.search(value)) {
        cout << value << " found in the list." << endl;
    } else {
        cout << value << " not found in the list." << endl;
    }

    return 0;
}
```

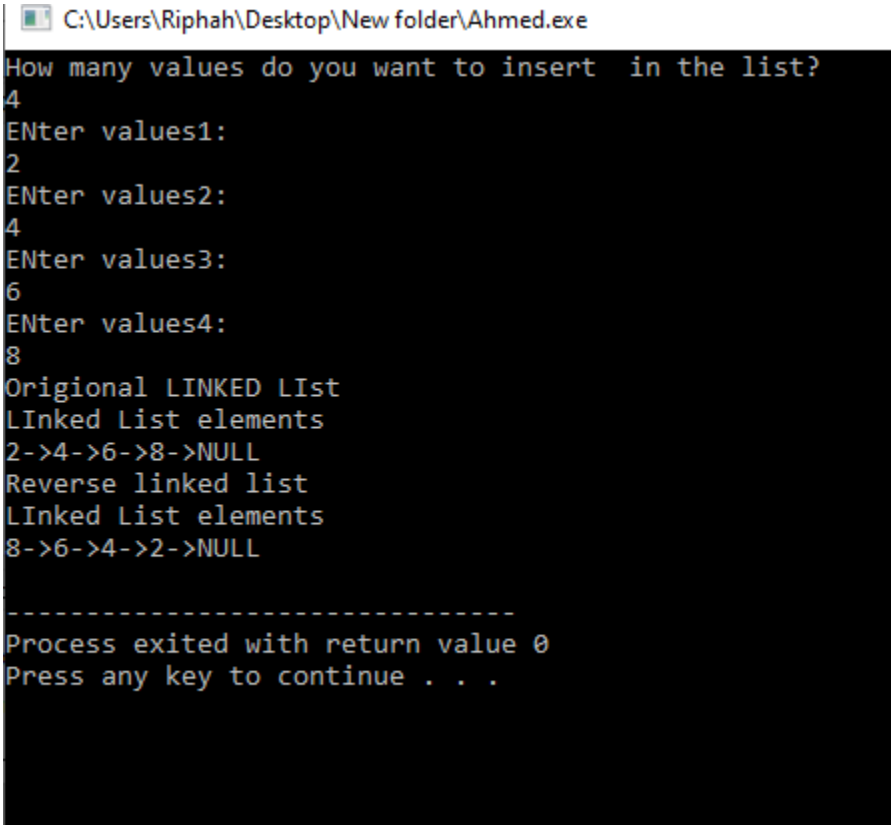
OUTPUT:

TASK2:

```
#include <iostream>
using namespace std;
struct Node{
	int data;
	Node* next;
};
void insertAtEnd(Node** head, int newData){
	Node* newNode=new Node();
	newNode->data= newData;
	newNode->next= NULL;
	if(*head == NULL){
		*head= newNode;
		return;
	}
	Node* last = *head;
	while(last->next!=NULL){
		last=last->next;
	}
	last->next=newNode;
}
void reverseList(Node** head){
	Node* prev= NULL;
	Node* current = *head;
	Node* next= NULL;
	while(current != NULL){
		next = current ->next;
		current->next= prev;
		prev= current;
		current= next;
```

```
        }
        *head = prev;
}
void printList(Node* node){
        cout<<"LInked List elements"<<endl;
        while(node!=NULL){
                cout<<node->data<<"->";
                node=node->next;
        }
        cout<<"NULL"<<endl;
}
int main(){
        Node* head= NULL;
        int n, value;
        cout<<"How many values do you want to insert  in the list?"<<endl;
        cin>>n;
        for(int i=0;i<n;i++)
        {
                cout<<"ENter values"<<i+1<<":"<<endl;
                cin>>value;
                insertAtEnd(&head,value);
        }
        cout<<"Origional LINKED LIst"<<endl;
        printList (head);
        reverseList(&head);
        cout<<"Reverse linked list"<<endl;
        printList(head);
        return 0;
}
```

OUTPUT:



TASK3:

TASK4:

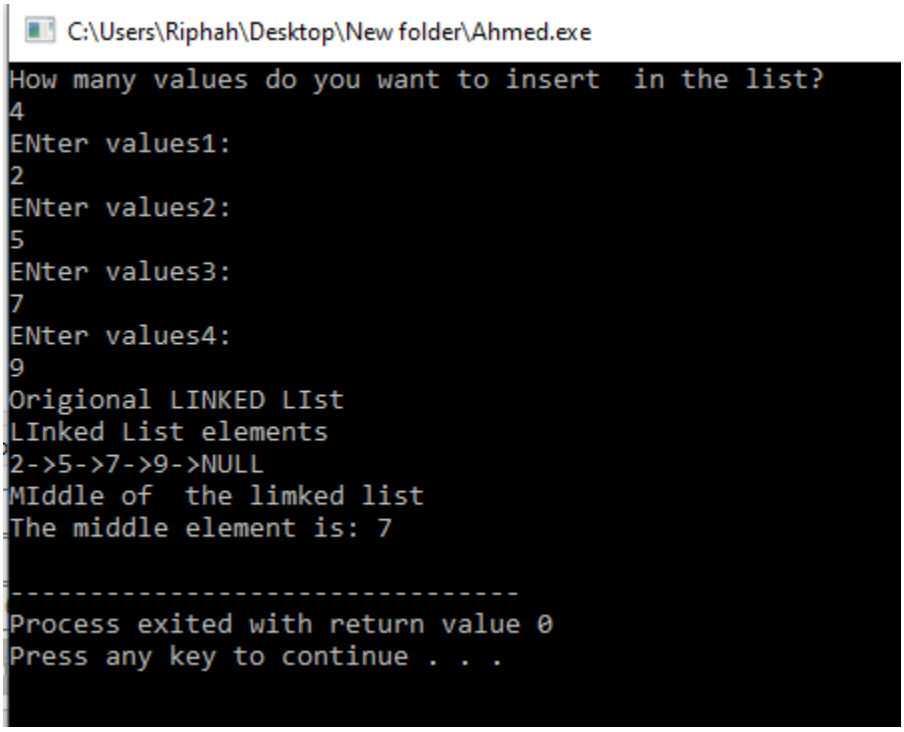
```
#include <iostream>
using namespace std;
struct Node{
	int data;
	Node* next;
};
void insertAtEnd(Node** head, int newData){
	Node* newNode=new Node();
	newNode->data= newData;
	newNode->next= NULL;
	if(*head == NULL){
		*head= newNode;
		return;
	}
	Node* last = *head;
	while(last->next!=NULL){
		last=last->next;
	}
	last->next=newNode;
}
void findMiddle(Node* head){
	if(head==NULL){
		cout<<"The list is  empty"<<endl;
		return;
	}
	Node* slow = head;
	Node* fast = head;
	while(fast!=NULL && fast->next !=NULL){
```

```
            slow = slow->next;
            fast= fast->next->next;
        }
        cout<<"The middle element is: "<<slow->data<<endl;
}
void printList(Node* node){
        cout<<"LInked List elements"<<endl;
        while(node!=NULL){
            cout<<node->data<<"->";
            node=node->next;
        }
        cout<<"NULL"<<endl;
}
int main(){
        Node* head= NULL;
        int n,count, value;
        cout<<"How many values do you want to insert  in the list?"<<endl;
        cin>>n;

        for(int i=0;i<n;i++)
        {
            cout<<"ENter values"<<i+1<<":"<<endl;
            cin>>value;
            insertAtEnd(&head,value);
        }
        cout<<"Origional LINKED LIst"<<endl;
        printList (head);
        cout<<"MIddle of  the limked list "<<endl;
        findMiddle(head);
```

```
        return 0;
}
```

OUTPUT



TASK5:

```
#include <iostream>
using namespace std;
int counter;
struct Node
{
   int data;
   Node* next;
};
```

```
void insertAtEnd(Node** head, int newData)
{
   Node* newNode = new Node();
   newNode->data = newData;
   newNode->next  = NULL;
   if(*head == NULL)
   {
      *head = newNode;
      return;
   }
Node* last = *head;
while(last->next != NULL)
{
   last = last->next;
}
   last->next = newNode;
}
void findMiddle(Node* head)
{
   if(head == NULL)
   {
      cout << "The list is empty." << endl;
      return;
   }
   Node* slow = head;
   Node* fast= head;
   while(fast != NULL && fast->next != NULL)
   {
      slow = slow->next;
```

```
        fast = fast->next->next;
    }
    cout << "The middle Element is: " << slow->data << endl;
}
void middle(Node* node, int counter)
{
  int count1 = counter/2;
for (int i=1; i<=count1; i++)
{
   node = node -> next;
}
cout << "Middle node Value is: " << node->data << endl;
}
void printList(Node* node)
{
   cout << " linked list elements: ";
   while( node != NULL)
   {
      cout  << node->data << " -> ";
      node = node -> next;
   }
   cout << "NULL" <<endl;
}
int main()
{
    Node* head = NULL;
    int n, value;
    cout<<" How many values do yo want to insertr in this list: ";
    cin>>n;
```

```
    for( int i=0; i<n; i++)
    {
       cout << " Enter value "<< i+1 <<" : ";
       cin >> value;
       counter++;
       insertAtEnd(&head, value);
    }
    printList(head);
    findMiddle(head);
    middle(head,counter);
    return 0;
}
```

OUTPUT: