Name:

Sidra Asif

Roll Number:

22011556-096

Class:

IT-22

Section:

B

Submitted to:

Sir Azib

Department:

Information technology

**** **University of Gujrat**

#include <iostream>

Using namespace std;

// Node structure for the linked list

Struct Node {

Int data;

Node\* next;

};

// Function to insert a node at the beginning of the linked list

Void insertAtBeginning(Node\*\* head, int newData) {

Node\* newNode = new Node();

newNode->data = newData;

newNode->next = \*head;

\*head = newNode;

}

// Function to insert a node at a specific position in the linked list

Void insertAtPosition(Node\*\* head, int newData, int position) {

If (position == 0) {

insertAtBeginning(head, newData);

return;

}

Node\* newNode = new Node();

newNode->data = newData;

Node\* temp = \*head;

For (int I = 0; I < position – 1 && temp != nullptr; i++) {

Temp = temp->next;

}

If (temp == nullptr) {

Cout << “Invalid position!” << endl;

Return;

}

newNode->next = temp->next;

temp->next = newNode;

}

// Function to delete a node from the beginning of the linked list

Void deleteFromBeginning(Node\*\* head) {

If (\*head == nullptr) {

Cout << “Linked list is empty!” << endl;

Return;

}

Node\* temp = \*head;

\*head = (\*head)->next;

Delete temp;

}

// Function to delete a node from the end of the linked list

Void deleteFromEnd(Node\*\* head) {

If (\*head == nullptr) {

Cout << “Linked list is empty!” << endl;

Return;

}

If ((\*head)->next == nullptr) {

Delete \*head;

\*head = nullptr;

Return;

}

Node\* temp = \*head;

While (temp->next->next != nullptr) {

Temp = temp->next;

}

Delete temp->next;

Temp->next = nullptr;

}

// Function to delete a node from a specific position in the linked list

Void deleteFromPosition(Node\*\* head, int position) {

If (\*head == nullptr) {

Cout << “Linked list is empty!” << endl;

Return;

}

If (position == 0)

{

Node\* temp = \*head;

\*head = (\*head)->next;

Delete temp;

Return;

}

Node\* temp = \*head;

Node\* prev = nullptr;

Int count = 0;

While (temp != nullptr && count < position) {

Prev = temp;

Temp = temp->next;

Count++;

}

If (temp == nullptr) {

Cout << “Invalid position!” << endl;

Return;

}

Prev->next = temp->next;

Delete temp;

}

// Function to search for a value in the linked list and update it

Void searchAndUpdate(Node\* head, int searchValue, int updateValue) {

Node\* temp = head;

While (temp != nullptr) {

If (temp->data == searchValue) {

Temp->data = updateValue;

Return;

}

Temp = temp->next;

}

Cout << “Value not found in the linked list!” << endl;

}

// Function to display the linked list

Void displayList(Node\* head) {

Node\* temp = head;

While (temp != nullptr) {

Cout << temp->data << “ “;

Temp = temp->next;

}

Cout << endl;

}

Int main() {

Node\* head = nullptr;

// Inserting elements at the beginning

insertAtBeginning(&head, 3);

insertAtBeginning(&head, 2);

insertAtBeginning(&head, 1);

// Inserting elements at a specific position

insertAtPosition(&head, 4, 2);

insertAtPosition(&head, 5, 4);

// Deleting elements from the beginning and end

deleteFromBeginning(&head);

deleteFromEnd(&head);

// Deleting element from a specific position

deleteFromPosition(&head, 2);

// Searching for a value and updating it

searchAndUpdate(head, 2, 10);

// Displaying the final linked list

displayList(head);

return 0;

}

