Linked List

* Operations in a Singly Linked List

# Insert At Head of Singly Linked List

#include <iostream> using namespace std;

class Node

{

public:

int data;

Node\* next;

Node(int data)

{

this -> data = data; this -> next = NULL;

}

};

void insertAtHead(Node\* &head, int value)

{

Node\* newNode = new Node(value); newNode -> next = head;

head = newNode;

}

void display(Node\* &head)

{

Node\* temp = head;

while (temp != NULL)

{

cout << temp->data << " -> "; temp = temp->next;

}

cout << "NULL";

}

int main()

{

Node\* head = NULL;

insertAtHead(head, 50);

insertAtHead(head, 40);

insertAtHead(head, 30);

insertAtHead(head, 20);

cout << "Current Linked List : "; display(head);

cout << endl; int newValue;

cout << "Enter a Value to be insert at Head : "; cin >> newValue;

insertAtHead(head, newValue);

cout << "Linked List After Inserting a Node at Head : "; display(head);

return 0;

}

# Insert at Tail of Singly Linked List

#include <iostream> using namespace std;

class Node

{

public:

int data;

Node\* next;

Node(int data)

{

this -> data = data; this -> next = NULL;

}

};

void insertAtTail(Node\* &head, int value)

{

Node\* newNode = new Node(value); if(head == NULL)

{

head = newNode; return;

}

Node\* temp = head; while(temp -> next != NULL)

{

temp = temp -> next;

}

temp -> next = newNode;

}

void display(Node\* &head)

{

Node\* temp = head; while(temp != NULL)

{

cout << temp -> data << " -> "; temp = temp -> next;

}

cout << "NULL";

}

int main()

{

Node\* head = NULL; insertAtTail(head, 50);

insertAtTail(head, 40);

insertAtTail(head, 30);

insertAtTail(head, 20);

cout << "Current Linked List : "; display(head);

cout << endl; int newValue;

cout << "Enter a value to be insert at the Tail : "; cin >> newValue;

insertAtTail(head, newValue);

cout << "After Inserting a Node at The Tail : "; display(head);

return 0;

}

# Insert before a given Node of Singly Linked List

#include <iostream> using namespace std; class Node

{

public:

int data;

Node\* next;

Node(int data)

{

this -> data = data; this -> next = NULL;

}

};

void insertAtTail(Node\* &head, int value)

{

Node\* newNode = new Node(value);

if(head == NULL)

{

head = newNode; return;

}

Node\* temp = head; while(temp -> next != NULL)

{

temp = temp -> next;

}

temp -> next = newNode;

}

void insertBefore(Node\* &head, int ibv, int value)

{

Node\* newNode = new Node(value); Node\* temp = head;

Node\* previous = NULL;

if(temp != NULL && temp -> data == ibv)

{

newNode -> next = temp; head = newNode;

return;

}

while(temp != NULL && temp -> data != ibv)

{

previous = temp;

temp = previous -> next;

}

if(temp == NULL)

{

endl;

cout << "Node with Value " << ibv << " Not Found" <<

delete newNode; // Free up the newNode space if not

Inserted

return;

}

previous -> next = newNode; newNode -> next = temp;

}

void display(Node\* &head)

{

Node\* temp = head; while(temp != NULL)

{

cout << temp -> data << " -> "; temp = temp -> next;

}

cout << "NULL" << endl;

}

int main()

{

Node\* head = NULL;

insertAtTail(head, 10);

insertAtTail(head, 20);

insertAtTail(head, 40);

insertAtTail(head, 50);

insertAtTail(head, 60);

cout << "Current Linked List : ";

display(head);

int ibv, newValue;

cout << "Enter the Value to insert Before : "; cin >> ibv;

cout << "Enter a New Value to be Insert : "; cin >> newValue;

insertBefore(head, ibv, newValue);

cout << "After Inserting, New Linked List : "; display(head);

return 0;

}

# Insert After a Node in Singly Linked List

#include <iostream> using namespace std; class Node

{

public:

int data;

Node\* next;

Node(int data)

{

this -> data = data; this -> next = NULL;

}

};

void insertAtTail(Node\* &head, int value)

{

Node\* newNode = new Node(value); if(head == NULL)

{

head = newNode; return;

}

Node\* temp = head; while(temp -> next != NULL)

{

temp = temp -> next;

}

temp -> next = newNode;

}

void insertAfter(Node\* &head, int iav, int value)

{

Node\* newNode = new Node(value); Node\* temp = head;

while(temp != NULL)

{

if(temp -> data == iav)

{

newNode -> next = temp -> next; temp -> next = newNode; return;

}

temp = temp -> next;

}

cout << "Node with Value " << iav << " not Found." << endl;

}

void display(Node\* &head)

{

Node\* temp = head; while(temp != NULL)

{

cout << temp -> data << " -> "; temp = temp -> next;

}

cout << "NULL" << endl;

}

int main()

{

Node\* head = NULL;

insertAtTail(head, 10);

insertAtTail(head, 20);

insertAtTail(head, 40);

insertAtTail(head, 50);

insertAtTail(head, 60);

cout << "Current Linked List : "; display(head);

int iav, newValue;

cout << "Enter the Value to insert After : "; cin >> iav;

cout << "Enter a New Value to be Insert : "; cin >> newValue;

insertAfter(head, iav, newValue);

cout << "After Inserting, New Linked List : "; display(head);

return 0;

}

# Insert at Specific Position in Singly Linked List

#include <iostream> using namespace std;

class Node

{

public:

int data;

Node\* next;

Node(int data)

{

this -> data = data; this -> next = NULL;

}

};

void insertAtTail(Node\* &head, int value)

{

Node\* newNode = new Node(value); if(head == NULL)

{

head = newNode; return;

}

Node\* temp = head; while(temp -> next != NULL)

{

temp = temp -> next;

}

temp -> next = newNode;

}

void insertSpecific(Node\* &head, int pos, int newValue)

{

Node\* newNode = new Node(newValue); Node\* ptr = head;

newNode -> data = newValue; newNode -> next = NULL;

pos--;

while(pos != 1)

{

ptr = ptr -> next; pos--;

}

newNode -> next = ptr -> next; ptr -> next = newNode;

}

void display(Node\* &head)

{

Node\* temp = head; while(temp != NULL)

{

cout << temp -> data << " -> "; temp = temp -> next;

}

cout << "NULL" << endl;

}

int main()

{

Node\* head = NULL;

insertAtTail(head, 10);

insertAtTail(head, 20);

insertAtTail(head, 30);

insertAtTail(head, 50);

insertAtTail(head, 60);

cout << "Current Linked List : " << endl; display(head);

int pos, newValue;

cout << "Enter a Value to be Insert : "; cin >> newValue;

cout << "Enter the Position : "; cin >> pos;

insertSpecific(head, pos, newValue);

cout << "After Inserting, Linked List : " << endl; display(head);

return 0;

}

# Delete from Head in Singly Linked List

#include <iostream> using namespace std;

class Node

{

public:

int data;

Node\* next;

Node(int data)

{

this -> data = data; this -> next = NULL;

}

};

void insertAtTail(Node\* &head, int value)

{

Node\* newNode = new Node(value); if(head == NULL)

{

head = newNode; return;

}

Node\* temp = head; while(temp -> next != NULL)

{

temp = temp -> next;

}

temp -> next = newNode;

}

void deleteFromHead(Node\* &head)

{

Node\* temp = head; // Node to be Deleted head = head -> next;

free(temp);

}

void display(Node\* &head)

{

Node\* temp = head; while(temp != NULL)

{

cout << temp -> data << " -> "; temp = temp -> next;

}

cout << "NULL" << endl;

}

int main()

{

Node\* head = NULL;

insertAtTail(head, 5);

insertAtTail(head, 10);

insertAtTail(head, 15);

insertAtTail(head, 20);

insertAtTail(head, 25);

insertAtTail(head, 30);

insertAtTail(head, 35); display(head); deleteFromHead(head);

cout << "After Deleting : " << endl; display(head);

return 0;

}

1. Delete from Tail in Singly Linked List

#include <iostream> using namespace std;

class Node

{

public:

int data;

Node\* next;

Node(int data)

{

this -> data = data; this -> next = NULL;

}

};

void insertAtTail(Node\* &head, int value)

{

Node\* newNode = new Node(value); if(head == NULL)

{

head = newNode; return;

}

Node\* temp = head; while(temp -> next != NULL)

{

temp = temp -> next;

}

temp -> next = newNode;

}

void deleteFromTail(Node\* &head)

{

Node\* secondLast = head;

while(secondLast -> next -> next != NULL)

{

secondLast = secondLast -> next;

}

Node\* temp = secondLast -> next; secondLast -> next = NULL; free(temp);

}

void deleteSpecific(Node\* &head, int position)

{

Node\* temp = head; if (temp == NULL)

{

return;

}

if (position == 1)

{

head = temp -> next; free(temp);

return;

}

for (int i = 1; temp != NULL && i < position - 1; i++)

{

temp = temp -> next;

}

if (temp == NULL || temp -> next == NULL)

{

return;

}

Node\* nodeToDelete = temp->next; temp -> next = temp -> next -> next; free(nodeToDelete);

}

void countNodes(Node\* &head)

{

int count = 0; Node\* temp = head;

while(temp != NULL)

{

count++;

temp = temp -> next;

}

cout << "The Number of Nodes in the Linked List is : " << count << endl;

}

void display(Node\* &head)

{

Node\* temp = head; while(temp != NULL)

{

cout << temp -> data << " -> "; temp = temp -> next;

}

cout << "NULL" << endl;

}

int main()

{

Node\* head = NULL;

insertAtTail(head, 5);

insertAtTail(head, 10);

insertAtTail(head, 15);

insertAtTail(head, 20);

insertAtTail(head, 25);

insertAtTail(head, 30);

insertAtTail(head, 35); display(head); deleteFromTail(head);

cout << "After Deleting, New Linked List will be : " << endl;

display(head); int position;

cout << "Enter the Position from where you want to delete the node : ";

cin >> position; deleteSpecific(head, position);

cout << "After Deleting from the specified position new Linked List : ";

display(head); countNodes(head);

return 0;

}

# Delete from Specific Position in Singly Linked List

#include <iostream> using namespace std;

class Node

{

public:

int data;

Node\* next;

Node(int data)

{

this -> data = data; this -> next = NULL;

}

};

void insertAtTail(Node\* &head, int value)

{

Node\* newNode = new Node(value); if(head == NULL)

{

head = newNode; return;

}

Node\* temp = head; while(temp -> next != NULL)

{

temp = temp -> next;

}

temp -> next = newNode;

}

void deleteSpecific(Node\* &head, int pos)

{

Node\* temp = head;

if(temp == NULL)

{

return;

}

if(pos == 0)

{

head = temp -> next; free(temp);

return;

}

for(int i = 0; temp != NULL && i < pos - 1; i++)

{

temp = temp -> next;

}

if (temp == NULL || temp -> next == NULL)

{

return;

}

Node\* toDelete = temp -> next;

temp -> next = temp -> next -> next;

free(toDelete);

}

void display(Node\* &head)

{

Node\* temp = head; while(temp != NULL)

{

cout << temp -> data << " -> "; temp = temp -> next;

}

cout << "NULL" << endl;

}

int main()

{

Node\* head = NULL; insertAtTail(head, 10);

insertAtTail(head, 20);

insertAtTail(head, 90);

insertAtTail(head, 30);

insertAtTail(head, 40);

insertAtTail(head, 50);

insertAtTail(head, 60);

cout << "Current Linked List : " << endl; display(head);

int pos;

cout << "Enter the Position from where you want to Delete

: ";

cin >> pos; deleteSpecific(head, pos);

cout << "After Deleting, New Linked List : " << endl; display(head);

return 0;

}

# Search a Node in a Singly Linked List

#include <iostream> using namespace std;

class Node

{

public:

int data;

Node\* next;

Node(int data)

{

this -> data = data; this -> next = NULL;

}

};

void insertAtTail(Node\* &head, int value)

{

Node\* newNode = new Node(value); if(head == NULL)

{

head = newNode; return;

}

Node\* temp = head; while(temp -> next != NULL)

{

temp = temp -> next;

}

temp -> next = newNode;

}

void search(Node\* &head, int searchElement)

{

Node\* curr = head; int flag = 0;

while(curr != NULL)

{

if(searchElement == curr -> data)

{

cout << "Element Found..." << endl; flag = 1;

break;

}

curr = curr -> next;

}

if(flag == 0)

{

cout << "Element Not Found in Linked List..." << endl;

}

}

void display(Node\* &head)

{

Node\* temp = head;

while(temp != NULL)

{

cout << temp -> data << " -> "; temp = temp -> next;

}

cout << "NULL";

}

int main()

{

Node\* head = NULL;

insertAtTail(head, 16);

insertAtTail(head, 17);

insertAtTail(head, 18);

insertAtTail(head, 19);

insertAtTail(head, 20); display(head);

cout << endl;

int searchElement;

cout << "Enter the Element you want to search in Linked List : ";

cin >> searchElement; search(head, searchElement);

return 0;

}

# Count the Number of Nodes in Singly Linked List

#include <iostream> using namespace std;

class Node

{

public:

int data;

Node\* next;

Node(int data)

{

this -> data = data; this -> next = NULL;

}

};

void countNodes(Node\* &head)

{

int count = 0;

Node\* temp = head; while(temp != NULL)

{

count++;

temp = temp -> next;

}

cout << "The Number Of Nodes in the Linked List : " << count << endl;

}

void insertAtTail(Node\* &head, int value)

{

Node\* newNode = new Node(value); if(head == NULL)

{

head = newNode; return;

}

Node\* temp = head; while(temp -> next != NULL)

{

temp = temp -> next;

}

temp -> next = newNode;

}

void display(Node\* &head)

{

Node\* temp = head; while(temp != NULL)

{

cout << temp -> data << " -> "; temp = temp -> next;

}

cout << "NULL";

}

int main()

{

Node\* head = NULL; insertAtTail(head, 50);

insertAtTail(head, 40);

insertAtTail(head, 30);

insertAtTail(head, 20);

insertAtTail(head, 10);

cout << "Current Linked List : "; display(head);

cout << endl; countNodes(head);

return 0;

}