// A complete working C++ program to delete

// a node in a linked list at a given position

#include <iostream>

using namespace std;

// A linked list node

class Node {

public:

int data;

Node\* next;

};

// Given a reference (pointer to pointer) to

// the head of a list and an int inserts a

// new node on the front of the list.

void push(Node\*\* head\_ref, int new\_data)

{

Node\* new\_node = new Node();

new\_node->data = new\_data;

new\_node->next = (\*head\_ref);

(\*head\_ref) = new\_node;

}

// Given a reference (pointer to pointer) to

// the head of a list and a position, deletes

// the node at the given position

void deleteNode(Node\*\* head\_ref, int position)

{

// If linked list is empty

if (\*head\_ref == NULL)

return;

// Store head node

Node\* temp = \*head\_ref;

// If head needs to be removed

if (position == 0) {

// Change head

\*head\_ref = temp->next;

// Free old head

free(temp);

return;

}

// Find previous node of the node to be deleted

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

temp = temp->next;

// If position is more than number of nodes

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

return;

// Node temp->next is the node to be deleted

// Store pointer to the next of node to be deleted

Node\* next = temp->next->next;

// Unlink the node from linked list

free(temp->next); // Free memory

// Unlink the deleted node from list

temp->next = next;

}

// This function prints contents of linked

// list starting from the given node

void printList(Node\* node)

{

while (node != NULL) {

cout << node->data << " ";

node = node->next;

}

}

// Driver code

int main()

{

// Start with the empty list

Node\* head = NULL;

push(&head, 7);

push(&head, 1);

push(&head, 3);

push(&head, 2);

push(&head, 8);

cout << "Created Linked List: ";

printList(head);

deleteNode(&head, 4);

cout << "\nLinked List after Deletion at position 4: ";

printList(head);

return 0;

}