// Biểu diễn đồ thị bằng danh sách kề dùng danh sách liên

#include <iostream>

using namespace std;

// Data structure to store adjacency list nodes

struct Node

{

int val;

Node\* next;

};

// Data structure to store a graph edge

struct Edge {

int src, dest;

};

class Graph

{

// Function to allocate a new node for the adjacency list

Node\* getAdjListNode(int dest, Node\* head)

{

Node\* newNode = new Node;

newNode->val = dest;

// point new node to the current head

newNode->next = head;

return newNode;

}

int N; // total number of nodes in the graph

public:

// An array of pointers to Node to represent the

// adjacency list

Node \*\*head;

// Constructor

Graph(Edge edges[], int n, int N)

{

// allocate memory

head = new Node\*[N]();

this->N = N;

// initialize head pointer for all vertices

for (int i = 0; i < N; i++) {

head[i] = nullptr;

}

// add edges to the directed graph

for (unsigned i = 0; i < n; i++)

{

int src = edges[i].src;

int dest = edges[i].dest;

// insert at the beginning

Node\* newNode = getAdjListNode(dest, head[src]);

// point head pointer to the new node

head[src] = newNode;

}

}

// Destructor

~Graph() {

for (int i = 0; i < N; i++) {

delete[] head[i];

}

delete[] head;

}

};

// Function to print all neighboring vertices of a given vertex

void printList(Node\* ptr)

{

while (ptr != nullptr)

{

cout << "—" << ptr->val;

ptr = ptr->next;

}

cout << endl;

}

// Graph implementation in C++ without using STL

int main()

{

// an array of graph edges as per the above diagram

Edge edges[] =

{

// pair {x, y} represents an edge from `x` to `y`

{0, 1}, {1, 2}, {2, 0}, {2, 1}, {3, 2}, {4, 5}, {5, 4}

};

// total number of nodes in the graph (labelled from 0 to 5)

int N = 6;

// calculate the total number of edges

int n = sizeof(edges)/sizeof(edges[0]);

// construct graph

Graph graph(edges, n, N);

// print adjacency list representation of a graph

for (int i = 0; i < N; i++)

{

// print given vertex

cout << i;

// print all its neighboring vertices

printList(graph.head[i]);

}

return 0;

}