```
#include <iostream>
using namespace std;

// Node structure for the linked list
struct Node {
    int data;
    Node* next;
};

// Function to add an edge to the graph
void addEdge(Node* adj[], int u, int v) {
    // Add v to u's list
    Node* newNode = new Node{v, adj[u]};
    adj[u] = newNode;

    // Add u to v's list (for undirected graph)
    newNode = new Node{u, adj[v]};
    adj[v] = newNode;
}

// Function to print the adjacency list
void printGraph(Node* adj[], int V) {
    for (int i = 0; i < V; i++) {
        cout << "Vertex " << i << ":";
        Node* temp = adj[i];
        while (temp) {
            cout << " -> " << temp->data;
            temp = temp->next;
        }
        cout << endl;
    }
}

// Function to free memory
void freeGraph(Node* adj[], int V) {
    for (int i = 0; i < V; i++) {
        Node* temp = adj[i];
        while (temp) {
            Node* toDelete = temp;
            temp = temp->next;
            delete toDelete;
        }
    }
}
```

```
int main() {
    int V = 5; // Number of vertices
    Node* adj[V] = {nullptr}; // Array of linked list heads

    // Adding edges
    addEdge(adj, 0, 1);
    addEdge(adj, 0, 4);
    addEdge(adj, 1, 2);
    addEdge(adj, 1, 3);
    addEdge(adj, 1, 4);
    addEdge(adj, 2, 3);
    addEdge(adj, 3, 4);

    // Print the graph
    printGraph(adj, V);

    // Free allocated memory
    freeGraph(adj, V);

    return 0;
}
```