graph.h

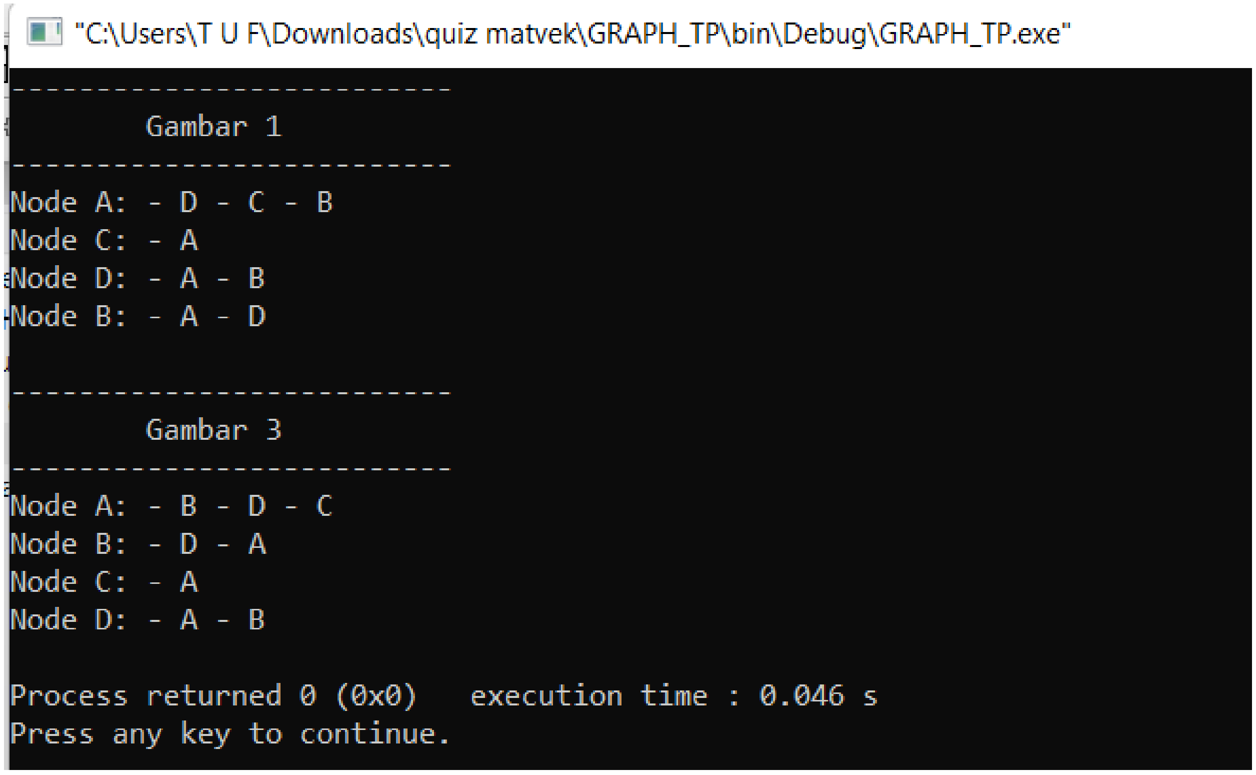
```
#ifndef GRAPH_H_INCLUDED
#define GRAPH_H_INCLUDED
#include <iostream>

#define info(P) (P)->info
#define next(P) (P)->next
#define dest(P) (P)->dest
#define nil NULL

using namespace std;

typedef char infotype;
typedef struct Node *adrNode;
typedef struct Edge *adrEdge;

struct Node {
    infotype info;
    adrNode next;
    adrEdge dest;
};

struct Edge {
    infotype info;
    adrEdge next;
};

adrNode newNode_1301210336(char x);
void addNode_1301210336(adrNode &G, adrNode p);
adrNode findNode_1301210336(adrNode G, char x);
void addEdge_1301210336(adrNode &G, char x, char y);
bool isConnected_1301210336(adrNode G, char x, char y);
void printGraph_1301210336(adrNode G);


#endif // GRAPH_H_INCLUDED
```

graph.cpp

```
#include "graph.h"

adrNode newNode_1301210336(char x){
    adrNode P = new Node;
    info(P) = x;
    next(P) = nil;
    dest(P) = nil;
    return P;
}

void addNode_1301210336(adrNode &G, adrNode p){
    if(next(G) == nil){
        next(G) = p;
    }else{
        while(next(G) != nil){
            G = next(G);
        }
        next(G) = p;
    }
}

adrNode findNode_1301210336(adrNode G, char x){
    if (G != nil) {
        while(info(G) != x){
            G = next(G);
        }
        return G;
    }
    return nil;
}

void addEdge_1301210336(adrNode &G, char x, char y){
    adrNode node = findNode_1301210336(G, x);

    if (node != nil) {
        adrEdge edge = new Edge;
        info(edge) = y;
        next(edge) = nil;
        if (dest(node) == nil) {
            dest(node) = edge;
        } else {
            adrEdge q = dest(node);
            while (next(q) != nil) {
                q = next(q);
            }
            next(q) = edge;
        }
    }

}

bool isConnected_1301210336(adrNode G, char x, char y){
    if(info(G) == x && dest(G) != nil){
        adrEdge p = dest(G);
        while(p != nil){
            if(info(p) == y){
                return true;
            }
            p = next(p);
        }
    }
    return false;
}

void printGraph_1301210336(adrNode G){
    if(next(G) == nil){
        cout << "GRAPH NOT FOUND" << endl;
    }else {
        G = next(G);
        while(G != nil){
            cout << "Node " << info(G) << ":";
            adrEdge edge = dest(G);
            while(edge != nil){
                cout << " - " << info(edge);
                edge = next(edge);
            }
            cout << endl;
            G = next(G);
        }
    }
}
```

main.cpp

```
#include "graph.h"


int main()
{
    adrNode P, adrP;
    P = new Node;
    next(P) = nil;

    adrP = newNode_1301210336('A');
    addNode_1301210336(P, adrP);
    addNode_1301210336(adrP, newNode_1301210336('C'));
    addNode_1301210336(adrP, newNode_1301210336('D'));
    addNode_1301210336(adrP, newNode_1301210336('B'));
    addEdge_1301210336(P, 'A', 'D');
    addEdge_1301210336(P, 'A', 'C');
    addEdge_1301210336(P, 'A', 'B');
    addEdge_1301210336(P, 'C', 'A');
    addEdge_1301210336(P, 'D', 'A');
    addEdge_1301210336(P, 'D', 'B');
    addEdge_1301210336(P, 'B', 'A');
    addEdge_1301210336(P, 'B', 'D');

    cout << "--------------------------" << endl;
    cout << "\tGambar 1" << endl;
    cout << "--------------------------" << endl;
    printGraph_1301210336(P);
    cout << endl;

    adrNode Q, adrQ;
    Q = new Node;
    next(Q) = nil;

    adrQ = newNode_1301210336('A');
    addNode_1301210336(Q, adrQ);
    addNode_1301210336(adrQ, newNode_1301210336('B'));
    addNode_1301210336(adrQ, newNode_1301210336('C'));
    addNode_1301210336(adrQ, newNode_1301210336('D'));
    addEdge_1301210336(Q, 'A', 'B');
    addEdge_1301210336(Q, 'A', 'D');
    addEdge_1301210336(Q, 'A', 'C');
    addEdge_1301210336(Q, 'B', 'D');
    addEdge_1301210336(Q, 'B', 'A');
    addEdge_1301210336(Q, 'C', 'A');
    addEdge_1301210336(Q, 'D', 'A');
    addEdge_1301210336(Q, 'D', 'B');

    cout << "-------------------------" << endl;
    cout << "\tGambar 3" << endl;
    cout << "-------------------------" << endl;
    printGraph_1301210336(Q);

    return 0;
}
```



"C:\Users\T U F\Downloads\quiz matvek\GRAPH_TP\bin\Debug\GRAPH_TP.exe"

```
--------------------------
        Gambar 1
--------------------------
Node A: - D - C - B
Node C: - A
Node D: - A - B
Node B: - A - D

--------------------------
        Gambar 3
--------------------------
Node A: - B - D - C
Node B: - D - A
Node C: - A
Node D: - A - B

Process returned 0 (0x0)   execution time : 0.046 s
Press any key to continue.
```