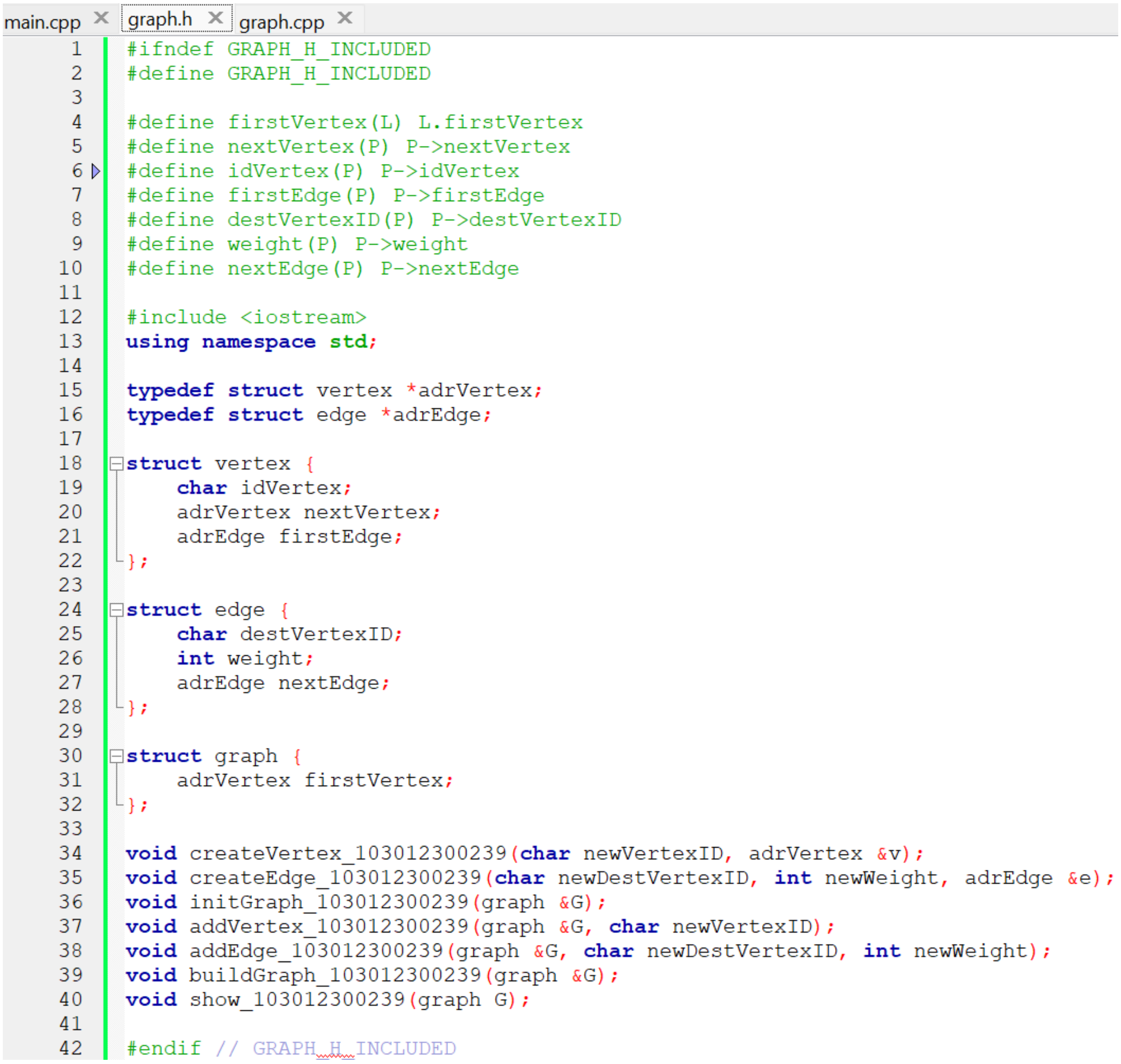
TP MODUL 11

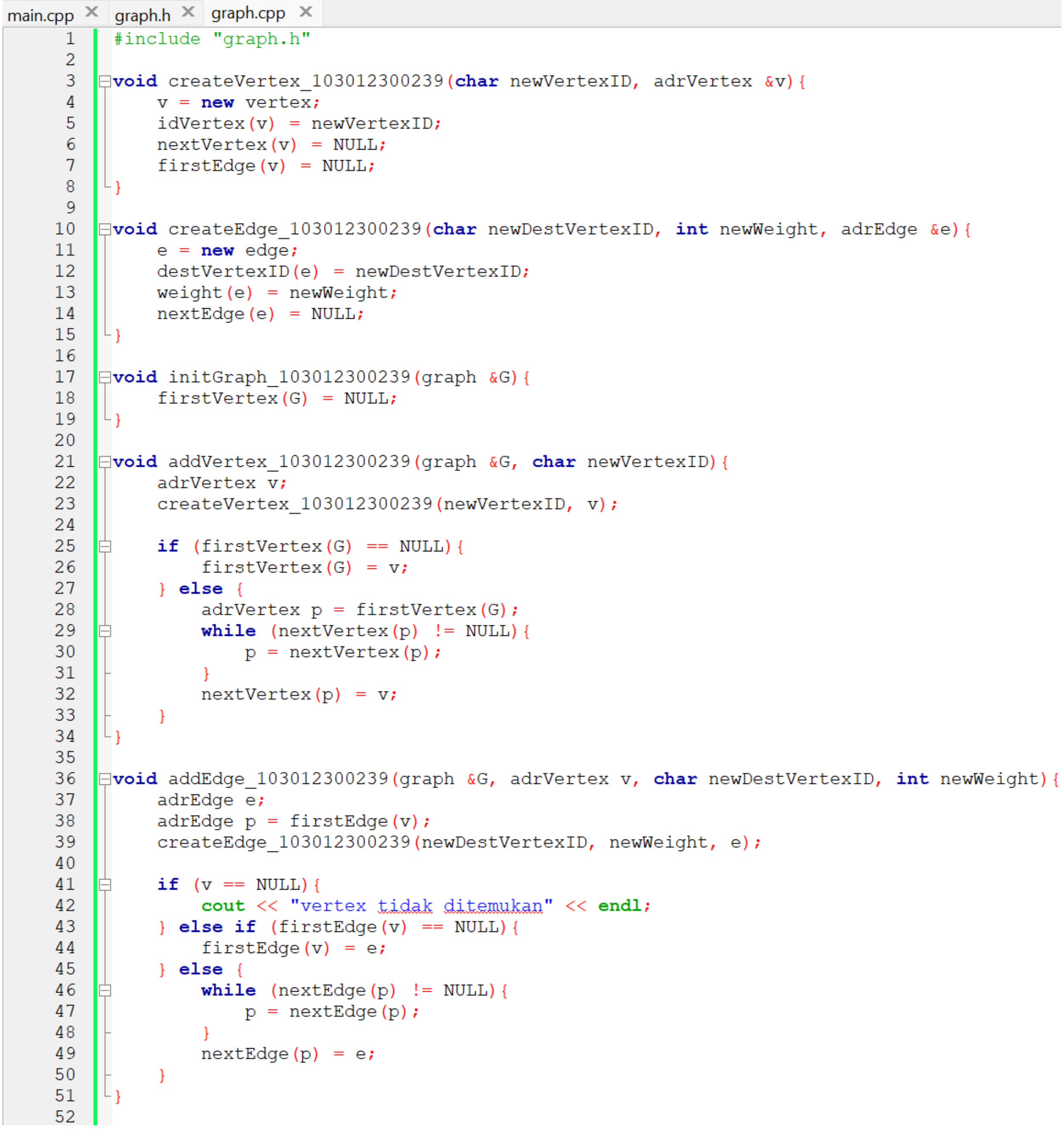
Nama : Ahmad Naufal Ramadhan
NIM : 103012300239
Kelas : IF-47-02
Kode Asprak : YDA

graph.h

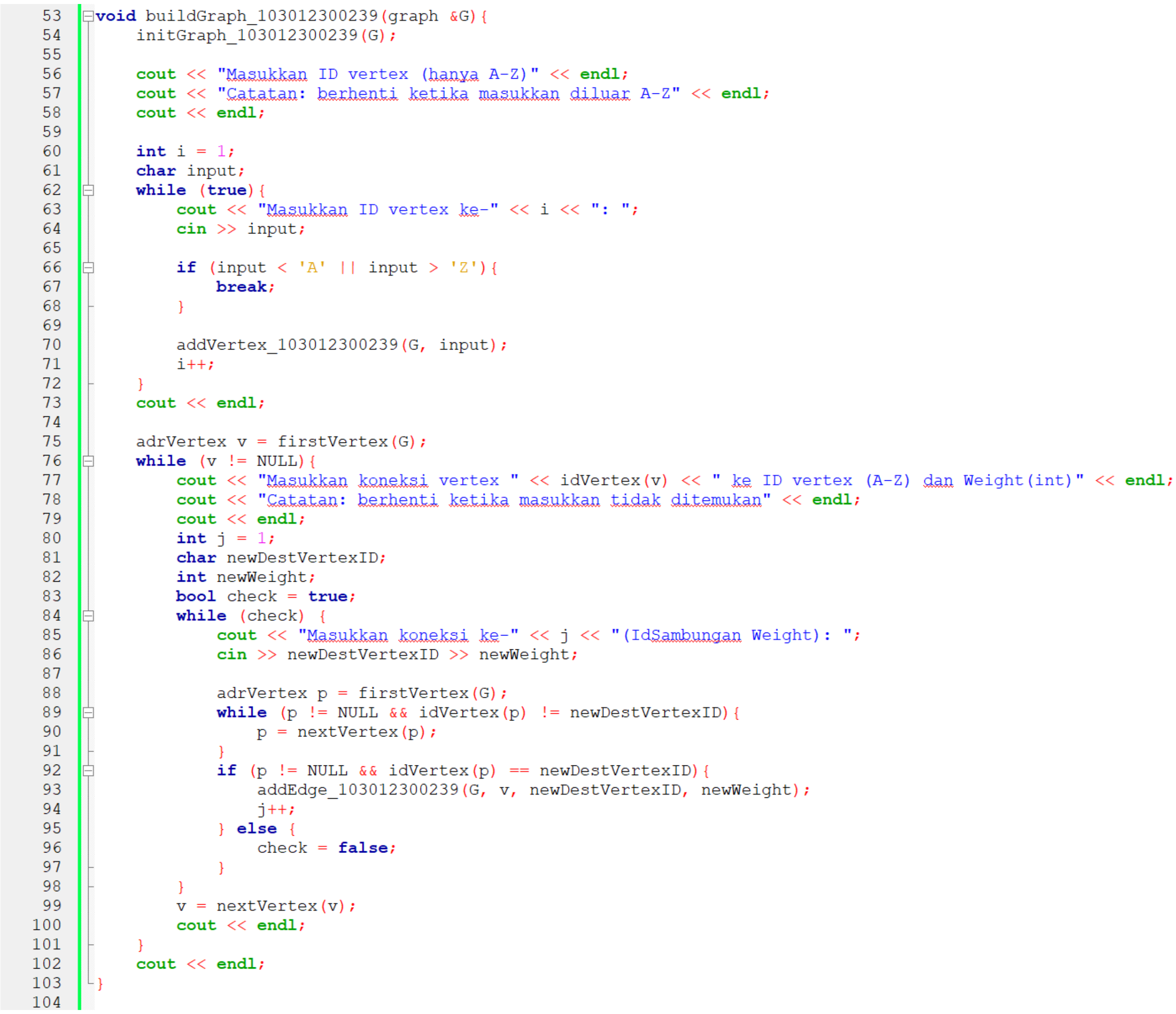


```cpp
#ifndef GRAPH_H_INCLUDED
#define GRAPH_H_INCLUDED

#define firstVertex(L) L.firstVertex
#define nextVertex(P) P->nextVertex
#define idVertex(P) P->idVertex
#define firstEdge(P) P->firstEdge
#define destVertexID(P) P->destVertexID
#define weight(P) P->weight
#define nextEdge(P) P->nextEdge

#include <iostream>
using namespace std;

typedef struct vertex *adrVertex;
typedef struct edge *adrEdge;

struct vertex {
    char idVertex;
    adrVertex nextVertex;
    adrEdge firstEdge;
};

struct edge {
    char destVertexID;
    int weight;
    adrEdge nextEdge;
};

struct graph {
    adrVertex firstVertex;
};

void createVertex_103012300239(char newVertexID, adrVertex &v);
void createEdge_103012300239(char newDestVertexID, int newWeight, adrEdge &e);
void initGraph_103012300239(graph &G);
void addVertex_103012300239(graph &G, char newVertexID);
void addEdge_103012300239(graph &G, char newDestVertexID, int newWeight);
void buildGraph_103012300239(graph &G);
void show_103012300239(graph G);

#endif // GRAPH_H_INCLUDED
```

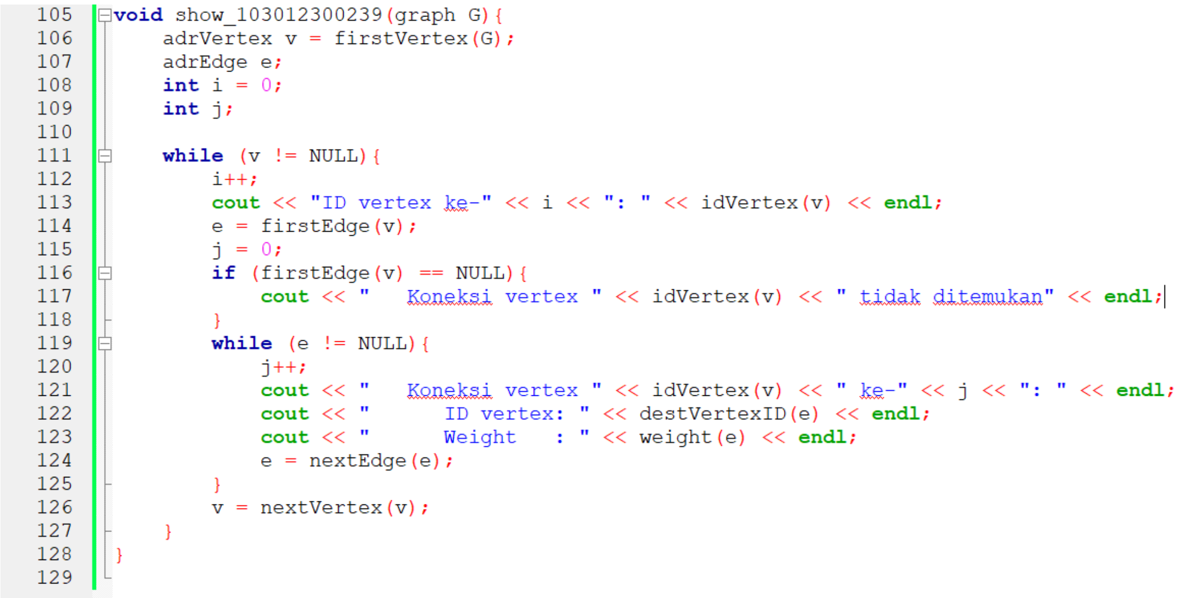
graph.cpp



```cpp
#include "graph.h"

void createVertex_103012300239(char newVertexID, adrVertex &v){
    v = new vertex;
    idVertex(v) = newVertexID;
    nextVertex(v) = NULL;
    firstEdge(v) = NULL;
}

void createEdge_103012300239(char newDestVertexID, int newWeight, adrEdge &e){
    e = new edge;
    destVertexID(e) = newDestVertexID;
    weight(e) = newWeight;
    nextEdge(e) = NULL;
}

void initGraph_103012300239(graph &G){
    firstVertex(G) = NULL;
}

void addVertex_103012300239(graph &G, char newVertexID){
    adrVertex v;
    createVertex_103012300239(newVertexID, v);

    if (firstVertex(G) == NULL){
        firstVertex(G) = v;
    } else {
        adrVertex p = firstVertex(G);
        while (nextVertex(p) != NULL){
            p = nextVertex(p);
        }
        nextVertex(p) = v;
    }
}

void addEdge_103012300239(graph &G, adrVertex v, char newDestVertexID, int newWeight){
    adrEdge e;
    adrEdge p = firstEdge(v);
    createEdge_103012300239(newDestVertexID, newWeight, e);

    if (v == NULL){
        cout << "vertex tidak ditemukan" << endl;
    } else if (firstEdge(v) == NULL){
        firstEdge(v) = e;
    } else {
        while (nextEdge(p) != NULL){
            p = nextEdge(p);
        }
        nextEdge(p) = e;
    }
}

```

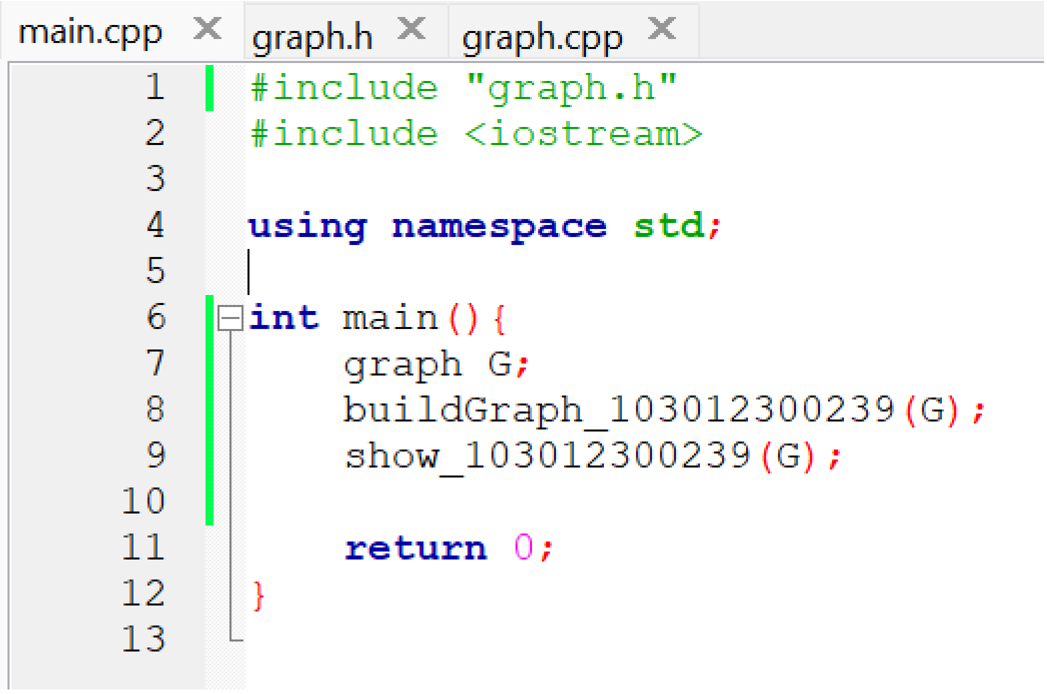


```
void buildGraph_103012300239(graph &G){
    initGraph_103012300239(G);

    cout << "Masukkan ID vertex (hanya A-Z)" << endl;
    cout << "Catatan: berhenti ketika masukkan diluar A-Z" << endl;
    cout << endl;

    int i = 1;
    char input;
    while (true){
        cout << "Masukkan ID vertex ke-" << i << ": ";
        cin >> input;

        if (input < 'A' || input > 'Z'){
            break;
        }

        addVertex_103012300239(G, input);
        i++;
    }
    cout << endl;

    adrVertex v = firstVertex(G);
    while (v != NULL){
        cout << "Masukkan koneksi vertex " << idVertex(v) << " ke ID vertex (A-Z) dan Weight(int)" << endl;
        cout << "Catatan: berhenti ketika masukkan tidak ditemukan" << endl;
        cout << endl;
        int j = 1;
        char newDestVertexID;
        int newWeight;
        bool check = true;
        while (check) {
            cout << "Masukkan koneksi ke-" << j << "(IdSambungan Weight): ";
            cin >> newDestVertexID >> newWeight;

            adrVertex p = firstVertex(G);
            while (p != NULL && idVertex(p) != newDestVertexID){
                p = nextVertex(p);
            }
            if (p != NULL && idVertex(p) == newDestVertexID){
                addEdge_103012300239(G, v, newDestVertexID, newWeight);
                j++;
            } else {
                check = false;
            }
        }
        v = nextVertex(v);
        cout << endl;
    }
    cout << endl;
}
```



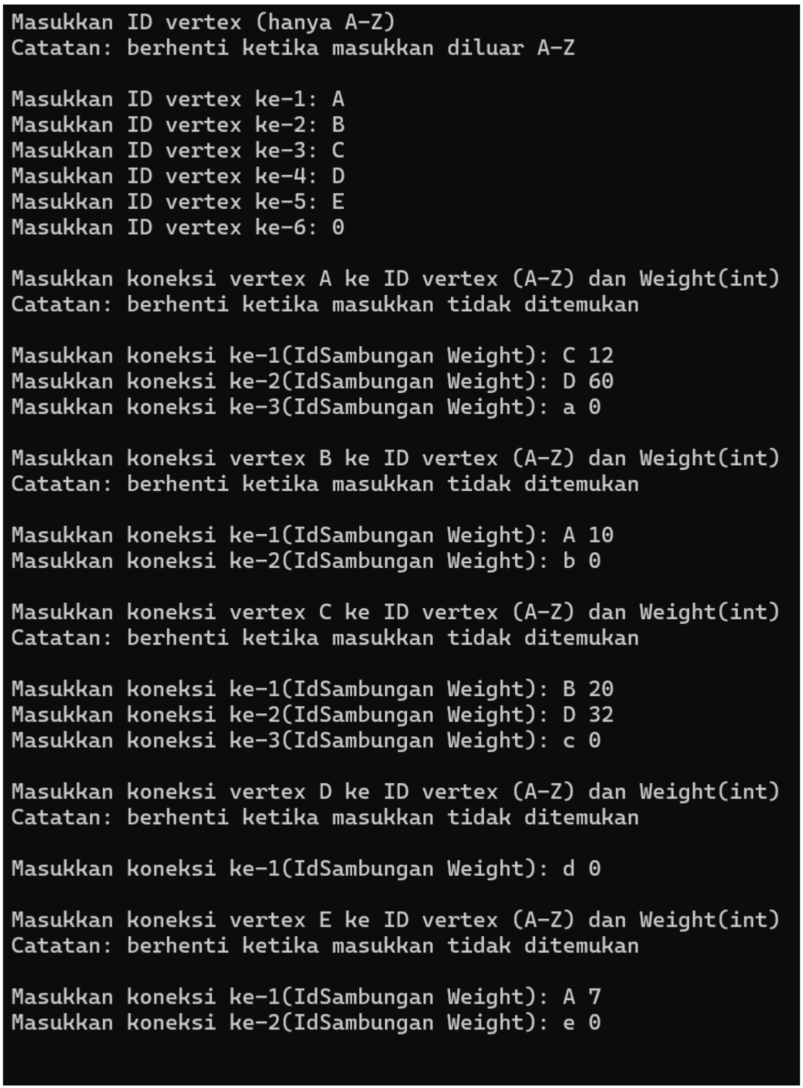
```
void show_103012300239(graph G){
    adrVertex v = firstVertex(G);
    adrEdge e;
    int i = 0;
    int j;

    while (v != NULL){
        i++;
        cout << "ID vertex ke-" << i << ": " << idVertex(v) << endl;
        e = firstEdge(v);
        j = 0;
        if (firstEdge(v) == NULL){
            cout << "   Koneksi vertex " << idVertex(v) << " tidak ditemukan" << endl;
        }
        while (e != NULL){
            j++;
            cout << "   Koneksi vertex " << idVertex(v) << " ke-" << j << ": " << endl;
            cout << "      ID vertex: " << destVertexID(e) << endl;
            cout << "      Weight    : " << weight(e) << endl;
            e = nextEdge(e);
        }
        v = nextVertex(v);
    }
}
```

main.cpp

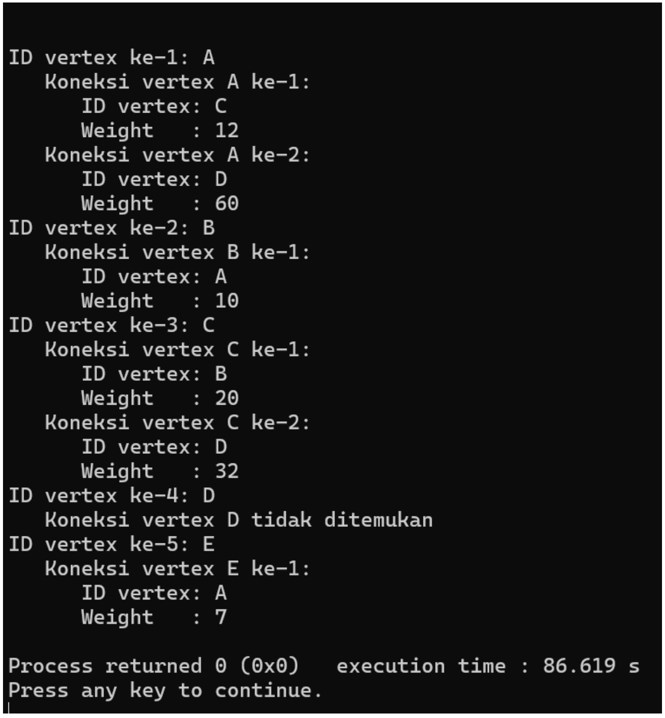


```cpp
#include "graph.h"
#include <iostream>

using namespace std;

int main(){
    graph G;
    buildGraph_103012300239(G);
    show_103012300239(G);

    return 0;
}
```

RUNNING TEST



```
Masukkan ID vertex (hanya A-Z)
Catatan: berhenti ketika masukkan diluar A-Z

Masukkan ID vertex ke-1: A
Masukkan ID vertex ke-2: B
Masukkan ID vertex ke-3: C
Masukkan ID vertex ke-4: D
Masukkan ID vertex ke-5: E
Masukkan ID vertex ke-6: 0

Masukkan koneksi vertex A ke ID vertex (A-Z) dan Weight(int)
Catatan: berhenti ketika masukkan tidak ditemukan

Masukkan koneksi ke-1(IdSambungan Weight): C 12
Masukkan koneksi ke-2(IdSambungan Weight): D 60
Masukkan koneksi ke-3(IdSambungan Weight): a 0

Masukkan koneksi vertex B ke ID vertex (A-Z) dan Weight(int)
Catatan: berhenti ketika masukkan tidak ditemukan

Masukkan koneksi ke-1(IdSambungan Weight): A 10
Masukkan koneksi ke-2(IdSambungan Weight): b 0

Masukkan koneksi vertex C ke ID vertex (A-Z) dan Weight(int)
Catatan: berhenti ketika masukkan tidak ditemukan

Masukkan koneksi ke-1(IdSambungan Weight): B 20
Masukkan koneksi ke-2(IdSambungan Weight): D 32
Masukkan koneksi ke-3(IdSambungan Weight): c 0

Masukkan koneksi vertex D ke ID vertex (A-Z) dan Weight(int)
Catatan: berhenti ketika masukkan tidak ditemukan

Masukkan koneksi ke-1(IdSambungan Weight): d 0

Masukkan koneksi vertex E ke ID vertex (A-Z) dan Weight(int)
Catatan: berhenti ketika masukkan tidak ditemukan

Masukkan koneksi ke-1(IdSambungan Weight): A 7
Masukkan koneksi ke-2(IdSambungan Weight): e 0
```



```
ID vertex ke-1: A
   Koneksi vertex A ke-1:
      ID vertex: C
      Weight   : 12
   Koneksi vertex A ke-2:
      ID vertex: D
      Weight   : 60
ID vertex ke-2: B
   Koneksi vertex B ke-1:
      ID vertex: A
      Weight   : 10
ID vertex ke-3: C
   Koneksi vertex C ke-1:
      ID vertex: B
      Weight   : 20
   Koneksi vertex C ke-2:
      ID vertex: D
      Weight   : 32
ID vertex ke-4: D
   Koneksi vertex D tidak ditemukan
ID vertex ke-5: E
   Koneksi vertex E ke-1:
      ID vertex: A
      Weight   : 7

Process returned 0 (0x0)   execution time : 86.619 s
Press any key to continue.
```

https://drive.google.com/file/d/1H8-6l0V7zYVU9Dbh-1Swu-NKcoeHlhLP/view?usp=sharing