LAPORAN PRAKTIKUM
STRUKTUR DATA

MODUL 12
GRAPH

Disusun Oleh :
NAMA : Muhammad Azzam Satria
NIM : 103112400112

Dosen
FAHRUDIN MUKTI WIBOWO S. Kom., M. Eng

PROGRAM STUDI STRUKTUR DATA
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2025

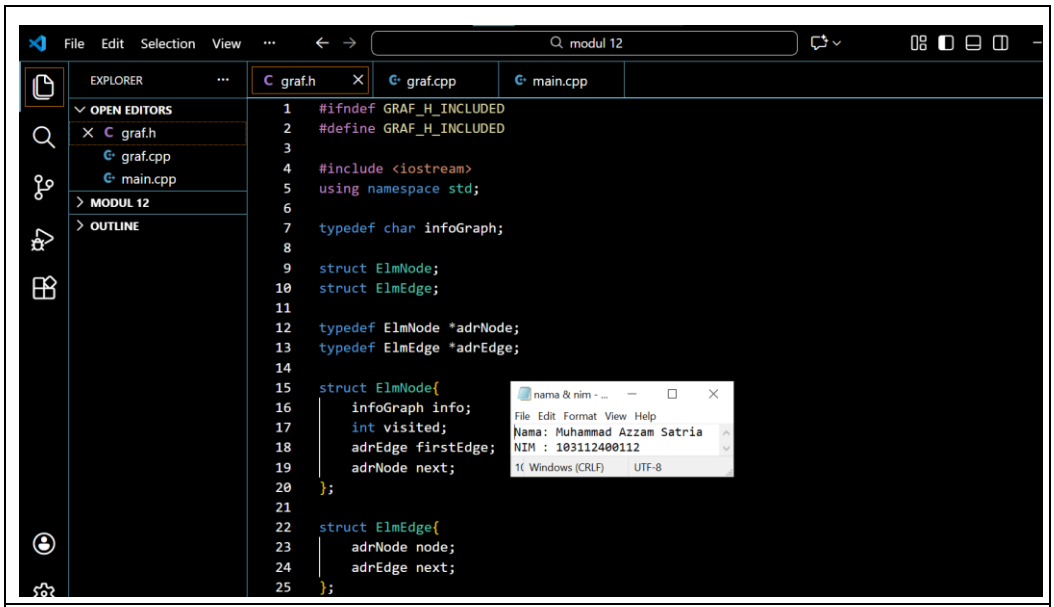
A. Dasar Teori

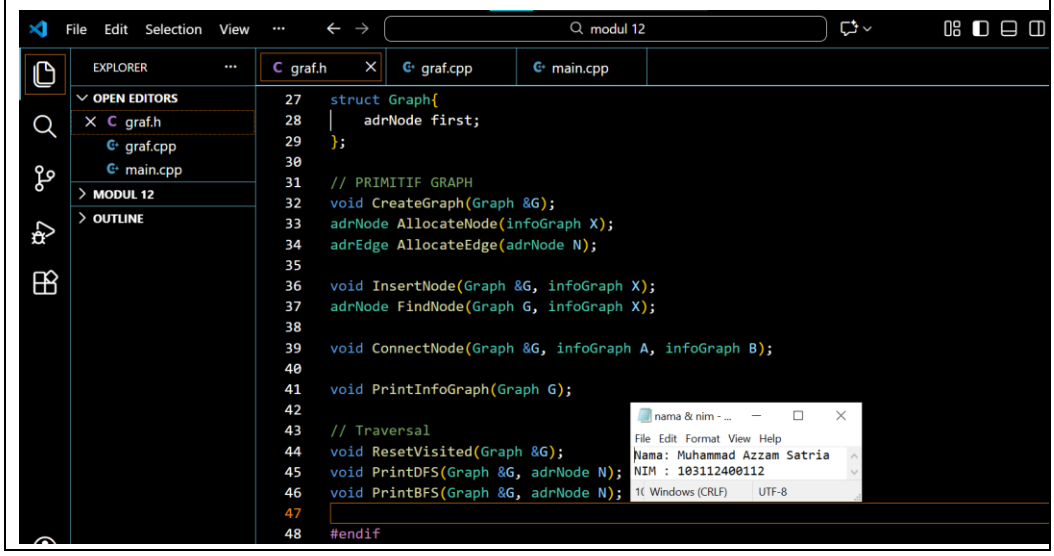
Graph merupakan struktur data yang berupa kumpulan node (simpul) yang dihubungkan oleh edge (sisi) yang menggambarkan hubungan antar data dari setiap simpulnya. Simpul pada graf adalah representasi dari suatu objek seperti orang dan tempat, sedangkan sisi merepresentasikan hubungan langsung antara dua simpul tertentu. Berdasarkan karakteristiknya, graph memiliki 2 jenis yaitu directed graph (graph berarah) dan undirected graph (graph tidak berarah). Pada graph berarah setiap sisi hanya memiliki satu arah yang jelas di setiap antar simpulnya, sedangkan graph tidak berarah setiap sisinya tidak mempunyai arah tertentu, karena setiap sisi yang menghubungkan dua simpul akan bersifat dua arah (saling menghubungkan). Algoritma graph banyak digunakan di berbagai operasi pada graf seperti analisis jaringan, optimisasi rute, pengenalan pola dan hubungan antar data.

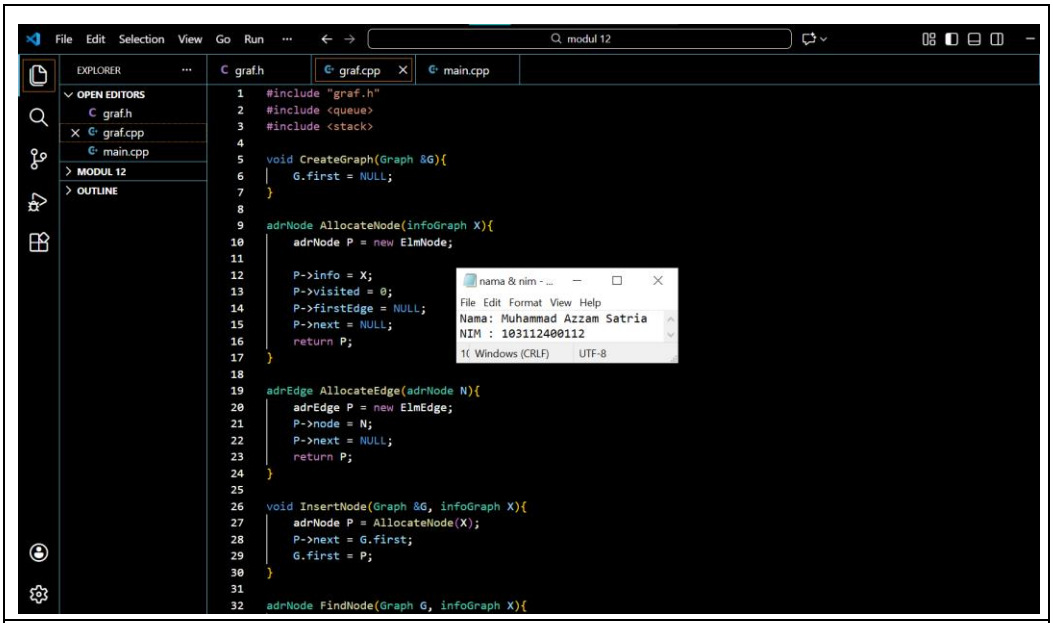
Pada penerapannya, graph dapat direpresentasikan dengan memakai multilist berbasis pointer karena sifatnya yang dinamis. Setiap simpul menyimpan daftar sisi yang menampilkan simpul-simpul lain saat terhubung secara langsung. Representasi graph seperti ini dapat ditemukan pada media sosial, yang mana pengguna direpresentasikan sebagai simpul dan hubungan seperti followers dan following direpresentasikan sebagai sisi. Selain itu, ketetanggaan antar node mempunyai peran penting dalam proses penelusuran graph. Metode penelusuran yang sering digunakan adalah Breadth First Search (BFS) dan Depth First Search (DFS). BFS menjalankan penelusuran berdasarkan tingkat atau jarak terdekat, sementara itu DFS melakukan penelusuran dengan mendalami satu cabang terlebih dahulu kemudian setelah satu cabang selesai penelusuran baru akan berpindah ke cabang lainnya pada graph.

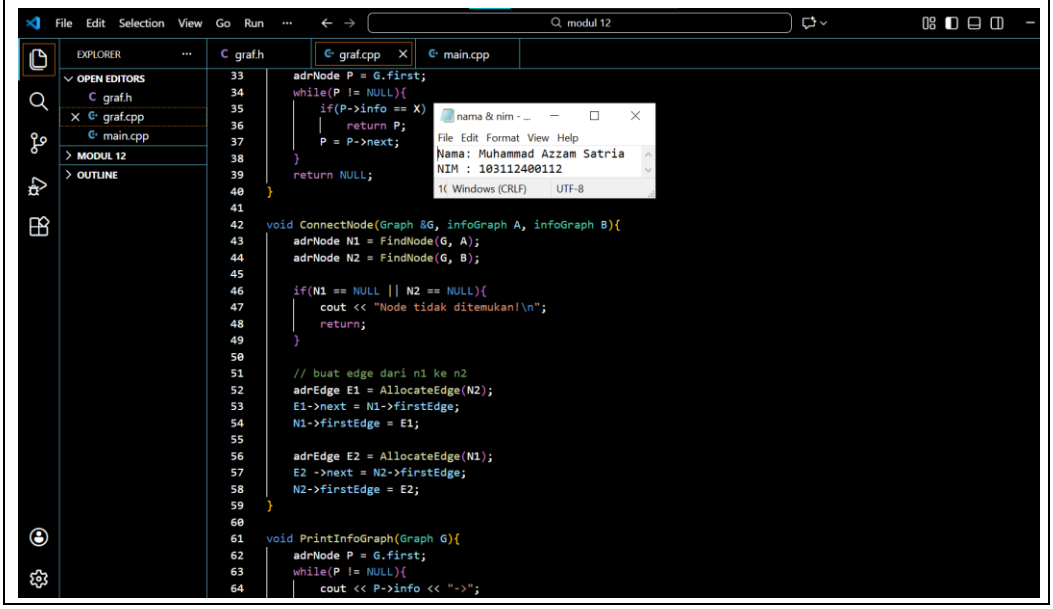
B. Guided (berisi screenshot source code & output program disertai penjelasannya)

Guided 1

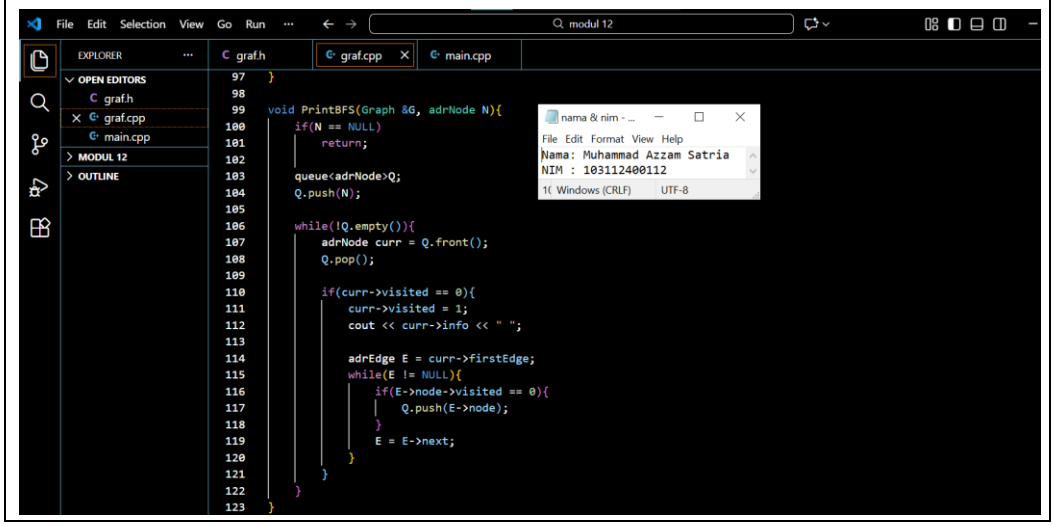

File Edit Selection View
modul 12
EXPLORER
OPEN EDITORS
graf.h
graf.cpp
main.cpp
MODUL 12
OUTLINE
graf.h
graf.cpp
main.cpp
#ifndef GRAF_H_INCLUDED
#define GRAF_H_INCLUDED
#include <iostream>
using namespace std;
typedef char infoGraph;
struct ElmNode;
struct ElmEdge;
typedef ElmNode *adrNode;
typedef ElmEdge *adrEdge;
struct ElmNode{
infoGraph info;
int visited;
adrEdge firstEdge;
adrNode next;
};
struct ElmEdge{
adrNode node;
adrEdge next;
};
nama & nim - ...
File Edit Format View Help
Nama: Muhammad Azzam Satria
NIM : 103112400112
Windows (CRLF)
UTF-8

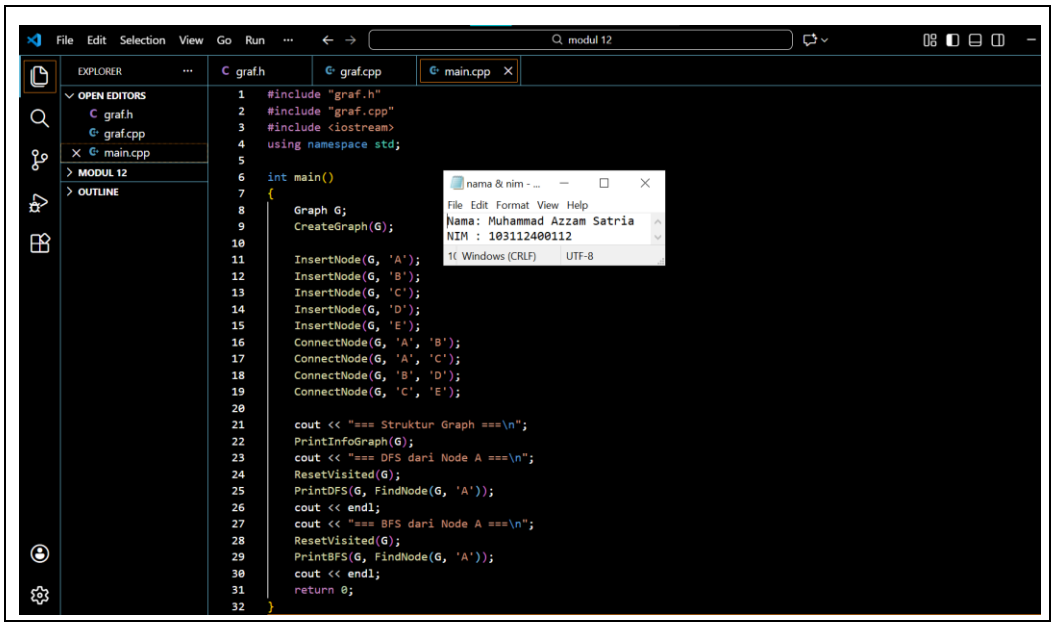

File Edit Selection View
modul 12
EXPLORER
OPEN EDITORS
graf.h
graf.cpp
main.cpp
MODUL 12
OUTLINE
graf.h
graf.cpp
main.cpp
struct Graph{
adrNode first;
};
// PRIMITIF GRAPH
void CreateGraph(Graph &G);
adrNode AllocateNode(infoGraph X);
adrEdge AllocateEdge(adrNode N);
void InsertNode(Graph &G, infoGraph X);
adrNode FindNode(Graph G, infoGraph X);
void ConnectNode(Graph &G, infoGraph A, infoGraph B);
void PrintInfoGraph(Graph G);
// Traversal
void ResetVisited(Graph &G);
void PrintDFS(Graph &G, adrNode N);
void PrintBFS(Graph &G, adrNode N);
#endif
nama & nim - ...
File Edit Format View Help
Nama: Muhammad Azzam Satria
NIM : 103112400112
Windows (CRLF)
UTF-8


File Edit Selection View Go Run
modul 12
EXPLORER
graf.h
graf.cpp
main.cpp
OPEN EDITORS
graf.h
graf.cpp
main.cpp
MODUL 12
OUTLINE
#include "graf.h"
#include <queue>
#include <stack>
void CreateGraph(Graph &G){
G.first = NULL;
}
adrNode AllocateNode(infoGraph X){
adrNode P = new ElmNode;
P->info = X;
P->visited = 0;
P->firstEdge = NULL;
P->next = NULL;
return P;
}
adrEdge AllocateEdge(adrNode N){
adrEdge P = new ElmEdge;
P->node = N;
P->next = NULL;
return P;
}
void InsertNode(Graph &G, infoGraph X){
adrNode P = AllocateNode(X);
P->next = G.first;
G.first = P;
}
adrNode FindNode(Graph G, infoGraph X){
nama & nim - ...
File Edit Format View Help
Nama: Muhammad Azzam Satria
NIM : 103112400112
1(Windows (CRLF) UTF-8

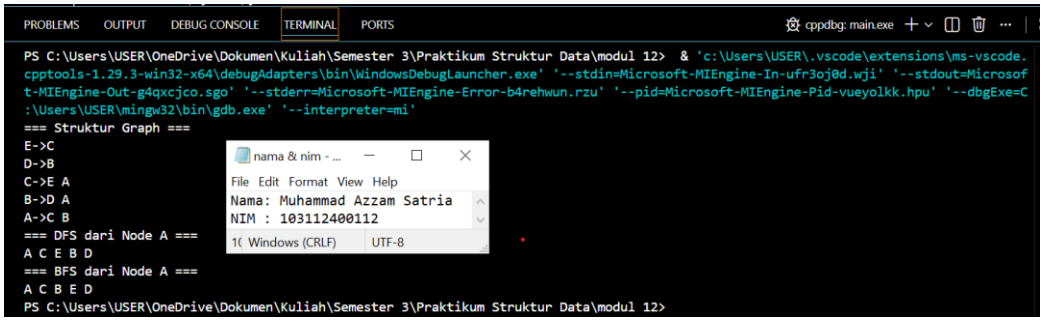

File Edit Selection View Go Run
modul 12
EXPLORER
graf.h
graf.cpp
main.cpp
OPEN EDITORS
graf.h
graf.cpp
main.cpp
MODUL 12
OUTLINE
adrNode P = G.first;
while(P != NULL){
if(P->info == X)
return P;
P = P->next;
}
return NULL;
}
void ConnectNode(Graph &G, infoGraph A, infoGraph B){
adrNode N1 = FindNode(G, A);
adrNode N2 = FindNode(G, B);
if(N1 == NULL || N2 == NULL){
cout << "Node tidak ditemukan!\n";
return;
}
// buat edge dari n1 ke n2
adrEdge E1 = AllocateEdge(N2);
E1->next = N1->firstEdge;
N1->firstEdge = E1;
adrEdge E2 = AllocateEdge(N1);
E2 ->next = N2->firstEdge;
N2->firstEdge = E2;
}
void PrintInfoGraph(Graph G){
adrNode P = G.first;
while(P != NULL){
cout << P->info << "->";
nama & nim - ...
File Edit Format View Help
Nama: Muhammad Azzam Satria
NIM : 103112400112
1(Windows (CRLF) UTF-8


File Edit Selection View Go Run
modul 12
EXPLORER
graf.h
graf.cpp
main.cpp
OPEN EDITORS
graf.h
graf.cpp
main.cpp
MODUL 12
OUTLINE
adrEdge E = P->firstEdge;
while(E != NULL){
cout << E->node->info << " ";
E = E->next;
}
cout << endl;
P = P->next;
}
}
void ResetVisited(Graph &G){
adrNode P = G.first;
while (P != NULL){
P->visited = 0;
P = P->next;
}
}
void PrintDFS(Graph &G, adrNode N){
if(N == NULL)
return;
N->visited = 1;
cout << N->info << " ";
adrEdge E = N->firstEdge;
while(E != NULL){
if(E->node->visited == 0){
PrintDFS(G, E->node);
}
E = E->next;
}
nama & nim - ...
File Edit Format View Help
Nama: Muhammad Azzam Satria
NIM : 103112400112
Ln 1, Col 1
Windows (CRLF)
UTF-8


File Edit Selection View Go Run
modul 12
EXPLORER
graf.h
graf.cpp
main.cpp
OPEN EDITORS
graf.h
graf.cpp
main.cpp
MODUL 12
OUTLINE
}
void PrintBFS(Graph &G, adrNode N){
if(N == NULL)
return;
queue<adrNode>Q;
Q.push(N);
while(!Q.empty()){
adrNode curr = Q.front();
Q.pop();
if(curr->visited == 0){
curr->visited = 1;
cout << curr->info << " ";
adrEdge E = curr->firstEdge;
while(E != NULL){
if(E->node->visited == 0){
Q.push(E->node);
}
E = E->next;
}
}
}
}
nama & nim - ...
File Edit Format View Help
Nama: Muhammad Azzam Satria
NIM : 103112400112
Ln 1, Col 1
Windows (CRLF)
UTF-8



Screenshots Output



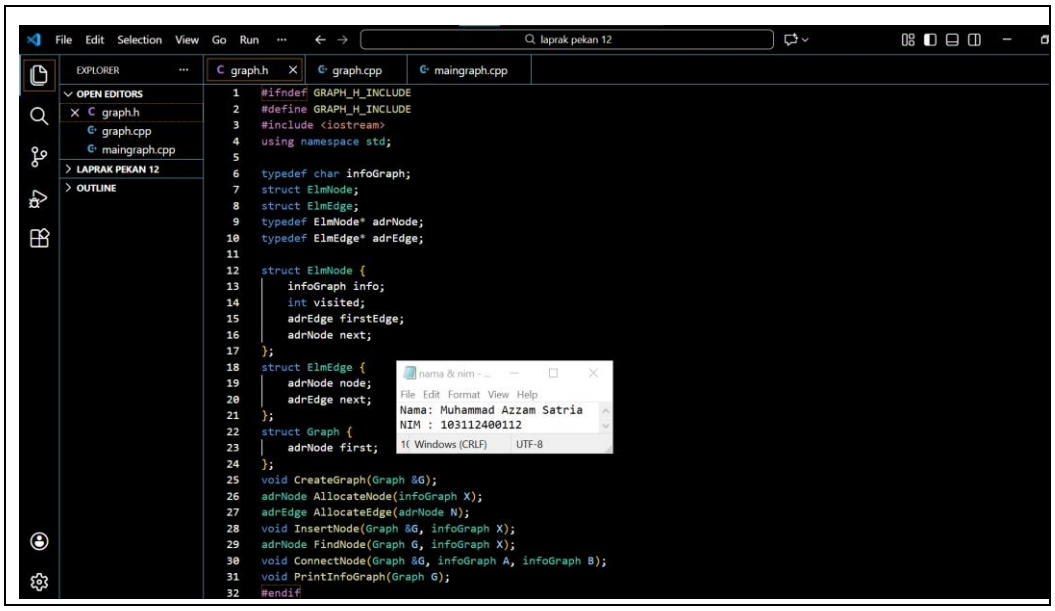
Deskripsi:

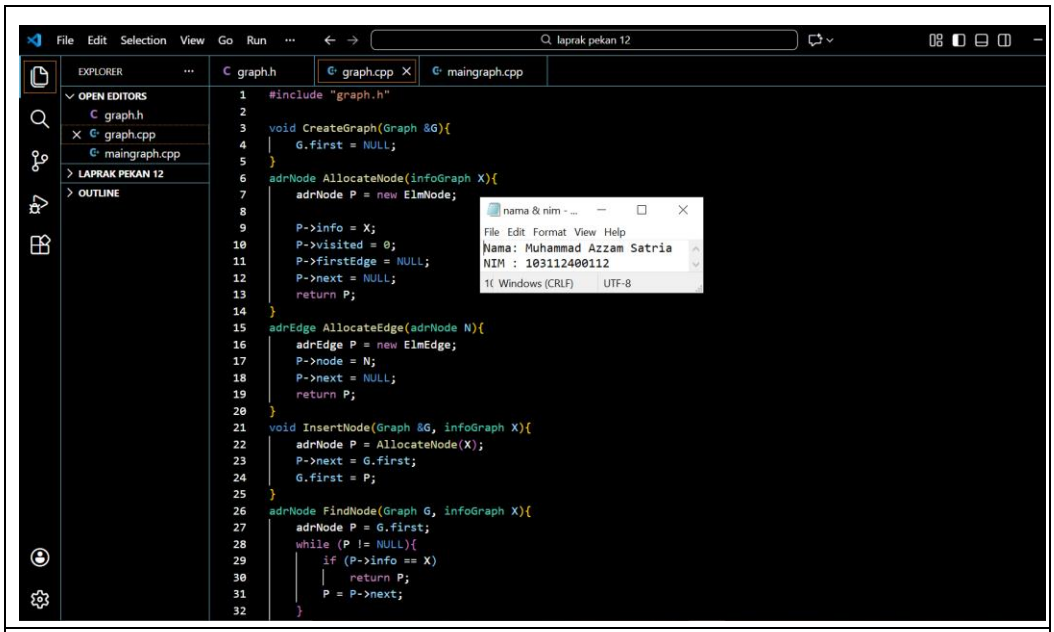
Program ini di dalam penerapannya menggunakan struktur data graf dengan konsep ADT (Abstract Data Type). Pada file garf.h struktur data graf didefinsikan dengan list linear yang terdiri dari node dan edge. Struktur ElmNode mempunyai fungsi untuk menyimpan data simpul berupa karakter, int visited, pointer ke edge pertama dan pointer ke edge selanjutnya dalam satu list. Struktur ElmEdge digunakan untuk menjadi penghubung antar node dengan menyimpan alamat node tujuan dan pointer ke edge selanjutnya. Di dalam header juga terdapat beberapa fungsi dan prosedur seperti CreateGraph, AllocateNode, AllocateEdge, InsertNode, FindNode, ConnectNode, ResetVisited, PrintInfoGraph, PrintDFS dan PrintBFS.

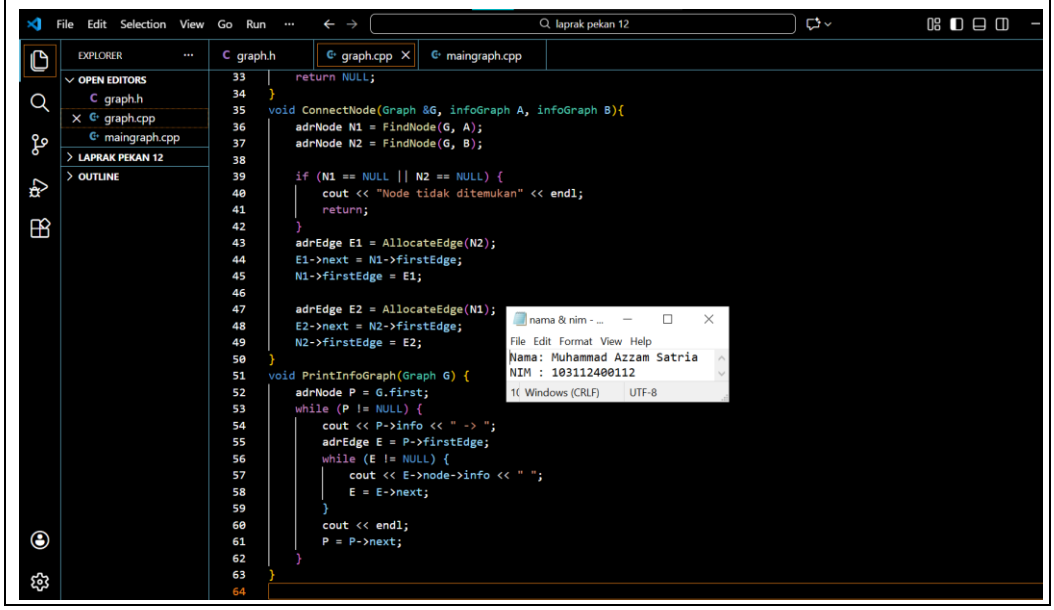
Pada file graf.cpp semua fungsi harus sesuai dengan yang sudah

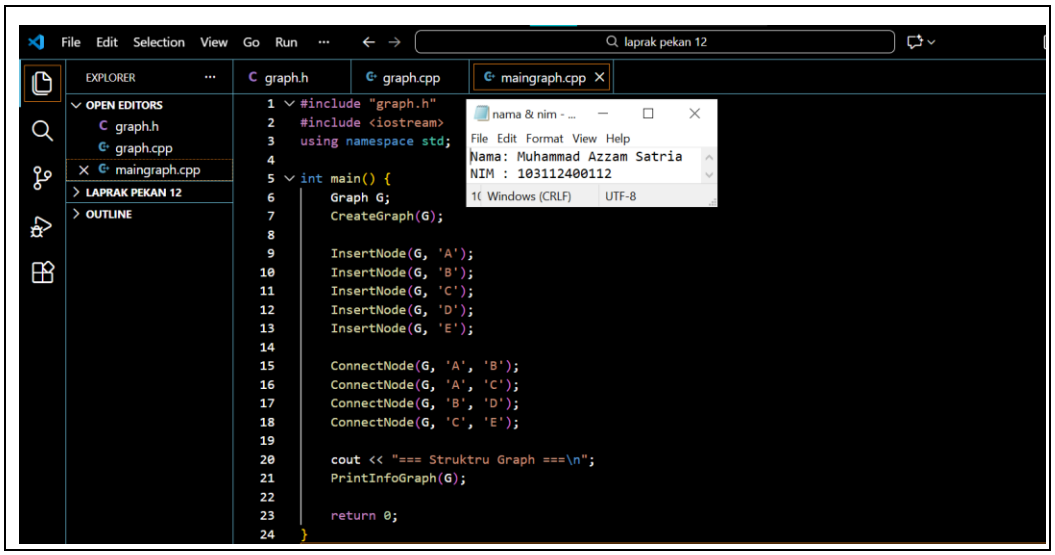
dideklarasikan pada file header. Prosedur CreateGraph akan mengatur ulang graf dengan mengosongkan node pertama. Fungsi AllocateNode dan AllocateEdge akan membuat node dan edge baru pada memori. Prosedur InsertNode melakukan penambahan node ke dalam graf di awal linked list. Fungsi FindNode yaitu akan mencari node berdasarkan informasi yang dicari. Prosedur ConnectNode memiliki fungsi sebagai penghubung dua node dengan menambahkan edge dua arah sehingga graf memiliki sifat tak berarah. Proses penelusuran graf akan melalui prosedur DFS dan BFS yang menggunakan variabel visited untuk menandai node yang telah dikunjungi, kemudian fungsi ResetVisited akan mengatur ulang status visited semua node sebelum traversal dijalankan pada program utama.

C. Unguided 1

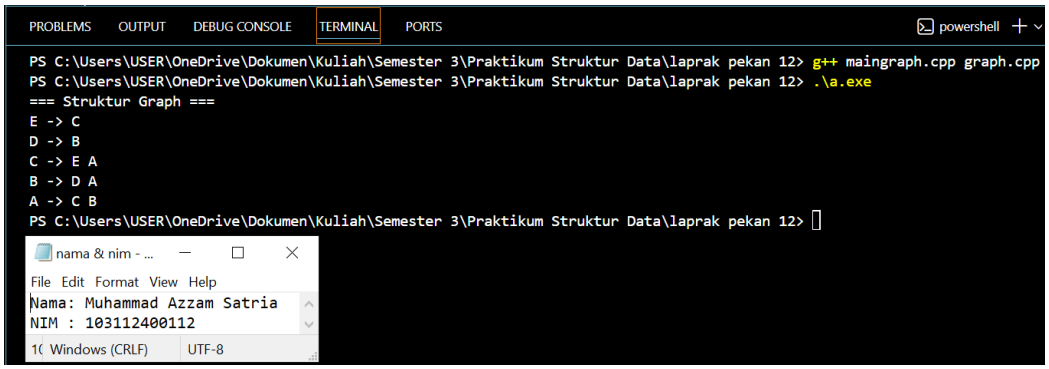



File Edit Selection View Go Run
laprak pekan 12
EXPLORER
graph.h
graph.cpp
maingraph.cpp
OPEN EDITORS
graph.h
graph.cpp
maingraph.cpp
LAPRAK PEKAN 12
OUTLINE
#include "graph.h"
void CreateGraph(Graph &G){
G.first = NULL;
}
adrNode AllocateNode(infoGraph X){
adrNode P = new ElmNode;
P->info = X;
P->visited = 0;
P->firstEdge = NULL;
P->next = NULL;
return P;
}
adrEdge AllocateEdge(adrNode N){
adrEdge P = new ElmEdge;
P->node = N;
P->next = NULL;
return P;
}
void InsertNode(Graph &G, infoGraph X){
adrNode P = AllocateNode(X);
P->next = G.first;
G.first = P;
}
adrNode FindNode(Graph G, infoGraph X){
adrNode P = G.first;
while (P != NULL){
if (P->info == X)
return P;
P = P->next;
}
nama & nim - ...
File Edit Format View Help
Nama: Muhammad Azzam Satria
NIM : 103112400112
Windows (CRLF)
UTF-8


File Edit Selection View Go Run
laprak pekan 12
EXPLORER
graph.h
graph.cpp
maingraph.cpp
OPEN EDITORS
graph.h
graph.cpp
maingraph.cpp
LAPRAK PEKAN 12
OUTLINE
return NULL;
}
void ConnectNode(Graph &G, infoGraph A, infoGraph B){
adrNode N1 = FindNode(G, A);
adrNode N2 = FindNode(G, B);
if (N1 == NULL || N2 == NULL) {
cout << "Node tidak ditemukan" << endl;
return;
}
adrEdge E1 = AllocateEdge(N2);
E1->next = N1->firstEdge;
N1->firstEdge = E1;
adrEdge E2 = AllocateEdge(N1);
E2->next = N2->firstEdge;
N2->firstEdge = E2;
}
void PrintInfoGraph(Graph G) {
adrNode P = G.first;
while (P != NULL) {
cout << P->info << " -> ";
adrEdge E = P->firstEdge;
while (E != NULL) {
cout << E->node->info << " ";
E = E->next;
}
cout << endl;
P = P->next;
}
}
nama & nim - ...
File Edit Format View Help
Nama: Muhammad Azzam Satria
NIM : 103112400112
Windows (CRLF)
UTF-8



Screenshots Output



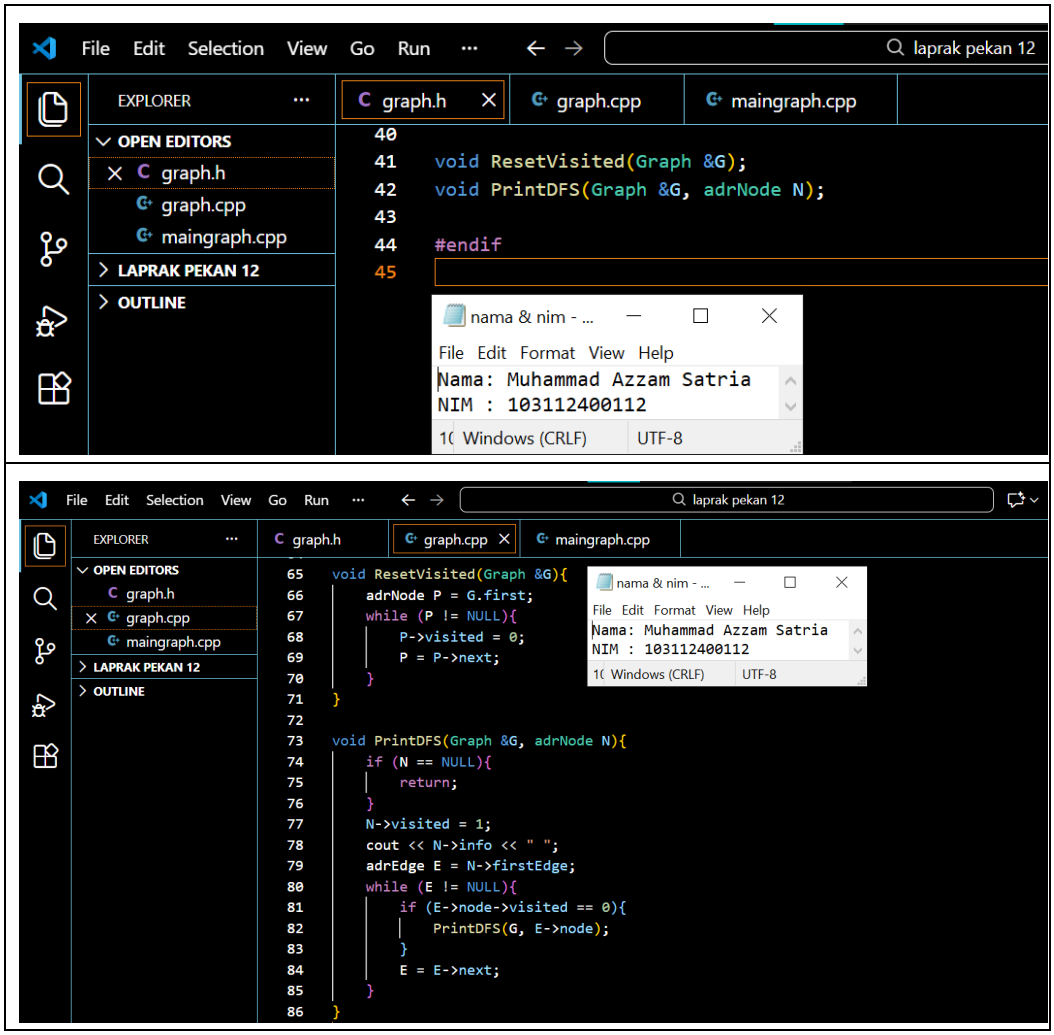
Deskripsi:

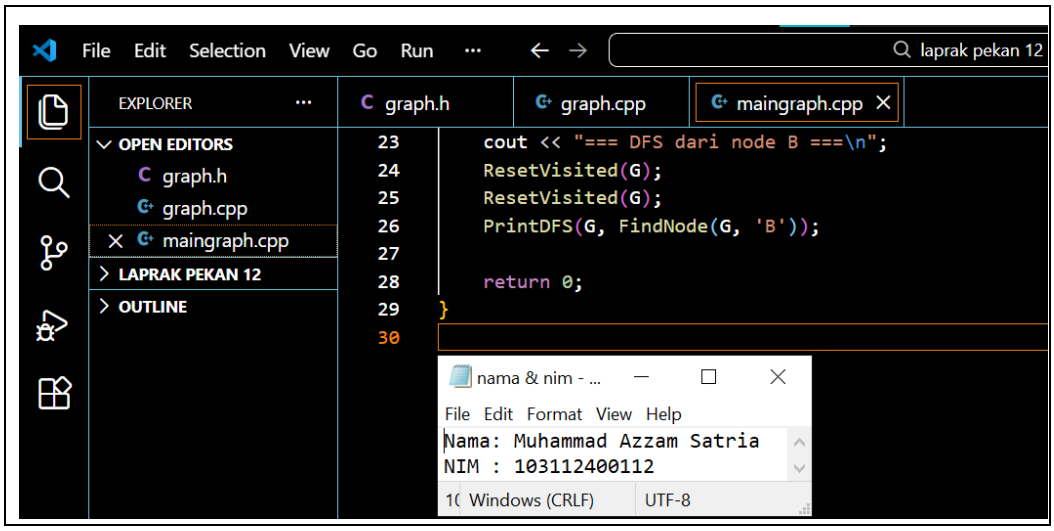
Pada soal nomor 1 program dibuat dengan struktur data graph tidak berarah yang menggunakan konsep ADT dan didalamnya terdapat representasi struktur multilist berbasis pointer. Pada file graph.h struktur ElmNode berfungsi untuk menyimpan data simpul dengan tipe char, int visited, pointer ke edge awal dan pointer ke edge selanjutnya. Struktru ElmEdge fungsinya yaitu untuk menyimpan hubungan antar node dengan menyimpan alamat node tujuan dan pointer ke edge berikutnya. Di dalam file header juga terdapat fungsi dan prosedur diantaranya yaitu CreateGraph, AllocateNode, AllocateEdge, InsertNode, FindNode, ConnectNode dan PrintInfoGraph.

Pada file graph.cpp prosedur CreateGraph fungsinya untuk mengiecek apakah graph dalam kondisi kosong. Fungsi AllocateNode dan AllocateEdge

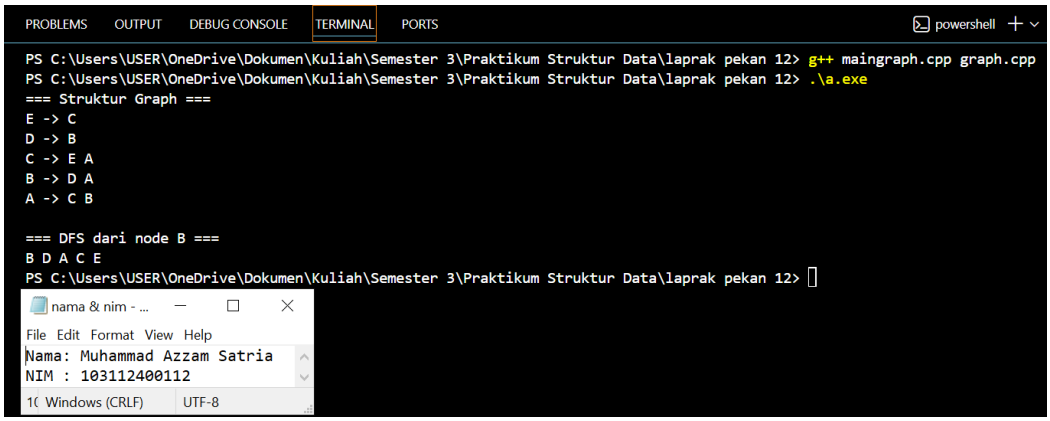
berperan untuk Membuat node dan edge baru. Prosedur InsertNode fungsinya untuk menambahkan node ke dalam graph dengan menyisipkan di awal list node. Fungsi FindNode adalah untuk mencari node tertentu berdasarkan data yang disimpan. Prosedur ConnectNode fungsinya untuk menghubungkan dua node dengan menambahkan edge dua arah agar graph menjadi bersifat tidak berarah. Pada file utama (graphmain.cpp) sejumlah node ditambahkan ke dalam graph, kemudian node-node tersebut dihubungkan menggunakan prosedur ConnectNode dan terakhir program mencetak output dengan menggunakan PrintInfoGraph.

Unguided 2





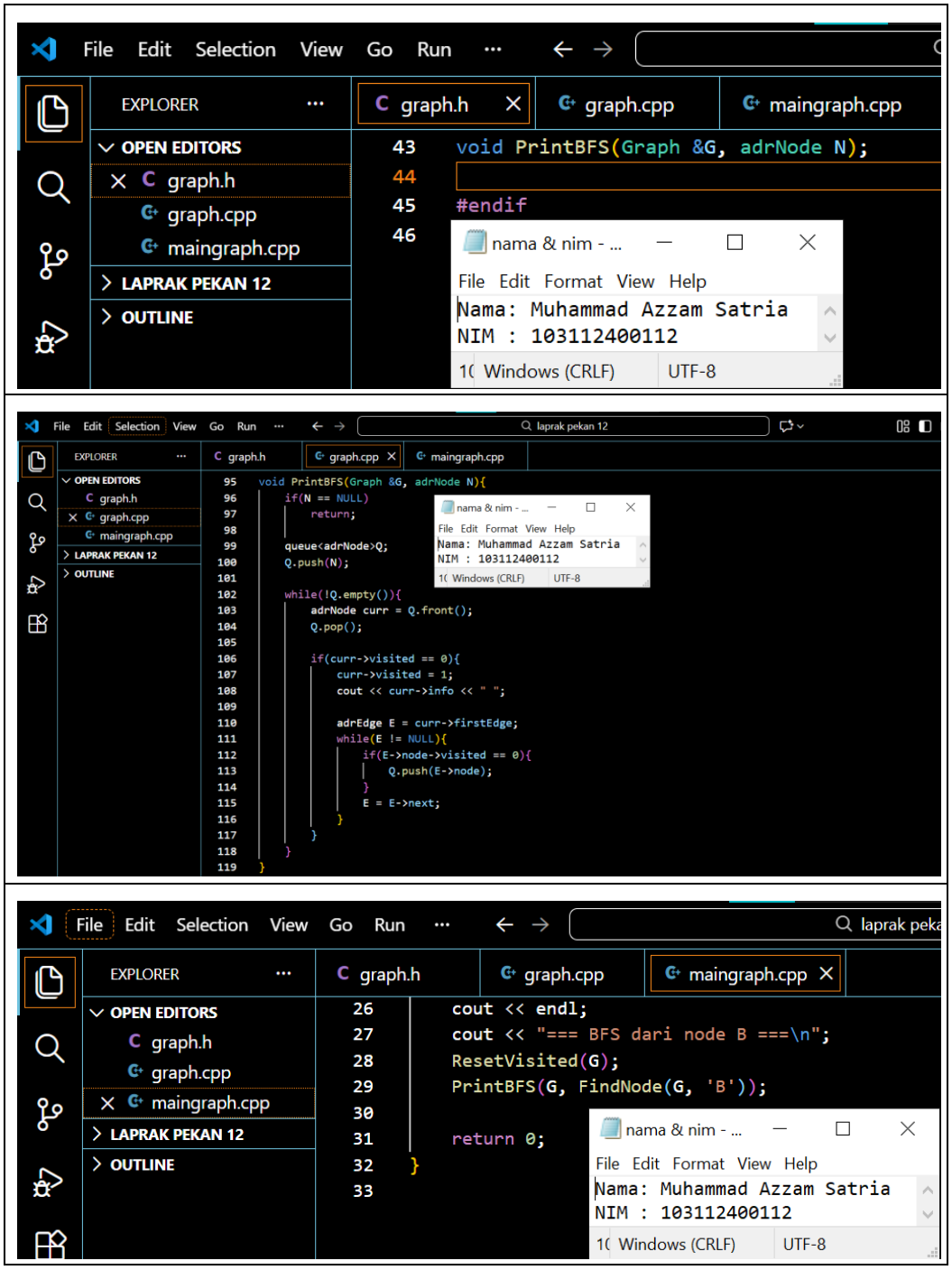
Screenshots Output



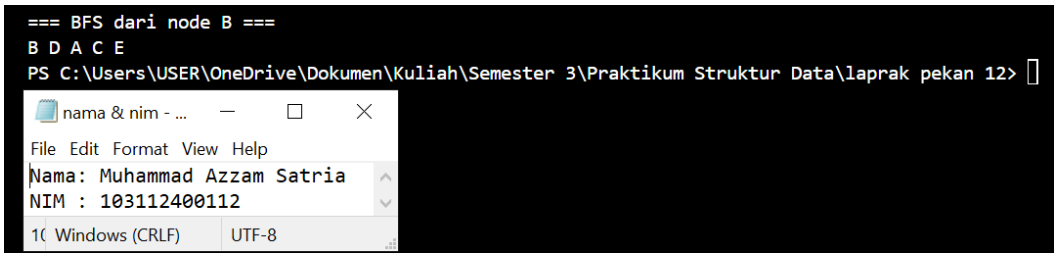
Deskripsi:

Pada soal nomor 2 terdapat tambahan prosedur yaitu ResetVisited dan PrintDFS. Prosedur ResetVisited fungsinya untuk mengatur ulang nilai penanda kunjungan di setiap simpul dengan cara menelusuri semua simpul yang ada di dalam graph dan menggunakan perulangan untuk mengecek setiap node melalui pointer next. Prosedur PrintDFS fungsinya yaitu untuk menerima parameter graph dan alamat node awal penelusuran. Cara kerja prosedur ini yaitu simpul yang sedang diproses akan ditandai sebagai telah dikunjungi, kemudian data di dalam simpul dicetak di output. Proses penelusuran tetap berlanjut ke semua simpul yang terhubung melalui sisi selama simpul tersebut belum perna dikunjungi.

Unguided 3



Screenshots Output



Deskripsi:

Pada soal nomor 3 terdapat penambahan prosedur PrintBFS yang fungsinya untuk menelusuri graf secara melebar dari kiri ke kanan dimulai dari simpul awal. Prosedur ini memakai struktur antrean untuk menyimpan simpul yang akan di proses berikutnya. Simpul awal ditandai sebagai telah dikunjungi dan dimasukkan ke dalam antrean. Setiap simpul yang keluar dari antrean akan ditampilkan dan simpul tetangganya akan diperiksa satu per satu Begitu seterusnya. Simpul yang belum dikunjungi akan ditandai dan dimasukkan ke dalam antrean. Proses dilakukan secara berulang sampai antrean kosong.

D. Kesimpulan

Struktur data graf pada modul ini direpresentasikan menggunakan simpul dan sisi yang saling terhubung melalui pointer. Dengan adanya guided dan modul graf, saya belajar bagaimana membangun struktur data graf menggunakan konsep Abstract Data Type (ADT). Pada program terdapat pembuatan node, penghubung antar node serta menampilkan struktur graf sesuai dengan output yang diinginkan. Selanjutnya program dikembangkan dengan menambahkan fungsi penelusuran graf dengan metode Depth First Search (DFS) dan Breadth First Search (BFS).

Penelusuran DFS cara kerjanya yaitu menggunakan pendekatan rekursif dan memanfaatkan penanda kunjungan pada setiap node. Sementara itu, BFS menggunakan struktur antrean untuk menelusuri simpul secara bertahap berdasarkan jarak dari node awal. Penambahan prosedur resetvisited yaitu untuk penanda kunjungan agar setiap penelusuran dilakukan dalam kondisi awal yang sama. Jadi hasil penelusuran dipengaruhi oleh dua faktor yaitu struktur graf dan titik awal yang digunakan.

E. Referensi

Andriati, D. A., Dariato, E., & Hafizh, R. (2025). Implementasi teori graf dan optimisasi algoritma Dijkstra, BFS, dan DFS dalam menentukan rute terpendek jaringan bengkel di Jakarta berbasis Google Maps. *Jurnal Multimedia dan Teknologi Informasi (Jatilima)*, 7(3), 674–684.

Ginting, S. H. N., Effendi, H., Kumar, S., Marsisno, W., Sitanggang, Y. R. U., Anwar, K., & Smrti, N. N. E. (2024). *Pengantar struktur data* (Vol. 1, No. 01). Penerbit Mifandi Mandiri Digital.

Ulandari, S., & Zahra, A. (2023). Representasi teori graf dalam jejaring sosial Instagram. *JEMST (Journal of Education in Mathematics, Science, and Technology)*, 6(2), 97–107.