Practical Exercise 6

Ομάδα : 3

Συμμετέχοντες : Μιχάλης Μιχαήλ

Σώτος Βασιλείου

Πασιουρτίδης Κώστας

A person holding a card

Description automatically generated with low confidence

Άσκηση 1

// \*\*\*\*\*\*\*\*\*\*\* έχω πρόβλημα στην ENQUEUE του queue γιαυτό στο κώδικα θα το βρείτε με στατική δέσμευση και την αναφορά σε δυναμική

Στην συγκεκριμένη άσκηση διαβάζουμε από αρχείο τους κόμβους και τις διαδρομές και δημιουργούμε το γράφον .Καλούμε την συνάρτηση BFS . Εσσωτερικά της BFS

δημιουργούμε δυναμικά της ουρά και εκτελούμε την παρακάτω διαδικασία μέχρι να αδειάσουμε όλη την ουρά . Διαδικασία : για όλες τις κορυφές Εάν αυτή η κορυφή δεν έχει επισκεφθεί την προσθέτουμε στην ουρά και την εκτυπώνουμε και εκτυπώνουμε τον πατέρα της που ήδη είμαστε σε αυτόν .

Graph\_init

Int descover={};

Bfs{

Queue\_init // malloc

Enqueue(start)

While(!queue\_is\_empty){

Dequeue

For(όλες τις πιθανες διαδρομές ([j])){

If(υπάρχει διαδρομή και δεν έχει επισκευθεί){

Descover[j]

Enqueue([j])

printf

}

}

}

//graph ίδιες συναρτήσεις και struct από δεύτερη άσκηση

//queue δυναμική δέσμευση

typedef struct node {

int data;

struct node \*next;

} Node;

typedef struct queue{

int size;

Node \*head;

Node \*tail;

}Queue;

Queue \* InitQueue(){

Queue \*newQueue;

return newQueue;

}

Node \* InitQueueNode(){

Node \*newNode = (Node \*) malloc(sizeof(Node));

newNode->next=NULL;

return newNode;

}

Άσκηση 2

DFS

Αρχικα στη main ανοιγω αρχειο το διαβαζω παιρνω τους κομωους και τις ακμες και στη συνεχεια παιρνω τις κορυφη 1 και την κορυφη 2 τις κάθε ακμης καθως και το βαρος.

Graph g;

ifstream fin

fin.open("ECE325\_graph.txt")

int nodes,akmes,kor1,kor2,weight;

fin>>nodes

//cout <<nodes<<endl

fin>>akmes

//cout <<akmes<< endl

while (!fin.eof()) {

fin>>kor1>>kor2>>weight

g.addEdge(kor1, kor2)

}

Μετα ξεκινα η κυρια διαδικασια DFS του γραφου ξεκινώντας από το μηδεν όπως λεει και στην ασκηση.

Αρχικα εχω μια συναρτηση void Graph::addEdge(int v, int w) στην οποια βαζω τις κορυφες των ακμων που συνδεουν το γραφο μου.

Στη συνεχεια η συναρτηση void Graph::DFS(int v) που παιρνει σαν παραμετρο π.χ. την κορυφη 0 στην οποια από αυτή θα ξεκινισει. Αν ο παρον κομβος είναι επισκεπτομενος τοτε του βαζουμε true και τον τυπωνουμε. Μετα επανερχομαστε σε ολους τους κομβους με for loop και παιρνοντας την λιστα μας από την αρχη μεχρι το τελος. ΑΝ ο κομβος δεν είναι επισκεπτόμενος τοτε ξαναγινεται η αναδρομη DFS.

void Graph::addEdge(int v, int w)

{

adj[v].push\_back(w); // Add w to v’s list.

}

void Graph::DFS(int v)

{

// simiono to paron komvo kai ton kano print

visited[v] = true;

cout << v<< " ";

//epanerxomai gia oles tis korifes

list<int>::iterator i;

//cout << v;

for (i = adj[v].begin(); i != adj[v].end(); ++i){

if (!visited[\*i]){

cout<<"(" <<v<< ")";

DFS(\*i);

}}

}