TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

KHOA CÔNG NGHỆ THÔNG TIN

---------------o0o---------------



**BÀI TẬP LỚN MÔN HỌC**

**CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT**

Giảng viên hướng dẫn: TS. Phạm Đình Phong

Sinh viên thực hiện: Nguyễn Quỳnh Anh – MSV 201200014 - Lớp CNTT 2 -K61

**Hà Nội tháng 12 năm 2021**

1. **Đề bài** (Bài 17)
2. Xây dựng cấu trúc dữ liệu hàng đợi
3. Xây dựng cấu trúc dữ liệu biểu diễn đồ thị
4. Áp dụng cấu trúc dữ kiệu hàng đợi thiết kế thuật toán tìm kiếm theo chiều rộng (BFS)
5. Tìm kiếm đường đi trong đồ thị có hướng
6. Thuật toán loang tìm đường đi trong mê cung
7. Một bài toán cụ thể khác sinh viên tự đề xuất

(Bài toán: Kiểm tra tính liên thông của đồ thị vô hướng.)

1. Viết hàm main thực hiện các công việc trên
2. **Phân tích bài toán**
3. Yêu cầu của bài toán:

* Xây dựng cấu trúc dữ liệu hàng đợi, từ đó áp dụng thuật toán tìm kiếm đường đi trong đồ thị có hướng và trong mê cung.
* Một bài toán cụ thể: Kiểm tra tính liên thông của đồ thị vô hướng.
* Viết hàm main để thực hiện thuật toán BFS.

1. Chương trình gồm: Queue.cpp, Graph.cpp, Tim\_Duong\_Di.cpp, Me\_Cung.cpp, Thuat\_Toan.cpp và Main.cpp. Ba file phụ: input.txt để lưu trữ đường đi trong mê cung, dulieu.txt để chứa đồ thị đường đi có hướng và file.txt để lưu trữ đường đi của đồ thị vô hướng.

* **File Queue.cpp:** Bao gồm các hàm như hàm tạo, hàm hủy, bool empty(), int size(), T &front(), T &back(), void pop(), void push(T x).
* **Thuộc tính:**

**+** Một con trỏ kiểu dữ liệu T tên là buff dùng để lưu trữ các phần tử kiểu T.

**+** Các biến kiểu số nguyên: num chứa giá trị số phần tử của hàng đợi trong mảng, cap là dung lượng của mảng và F,L là hai con trỏ để lưu trữ phần tử đầu, cuối của hàng đợi.

* **Các phương thức của lớp:**

**+** Hàm tạo Queue(): Dùng để khởi tạo một hàng đợi.

**+** Hàm hủy ~Queue(): Dùng để xóa một hàng đợi.

**+** Hàm bool empty(): Trả lại giá trị kiểu boolen để xác định có phần tử được lưu trữ trong queue không? (Để kiểm tra xem queue có rỗng không. Nếu kiểu trả về là true thì queue rỗng, ngược lại kết quả trả về false thì queue không rỗng.)

**+** Hàm int size(): Trả về số phần tử hiện đang được lưu trữ trong queue.

**+** Hàm T &front(): Trả lại phần tử đầu tiên trong queue.

**+** Hàm T &back(): Trả lại phần tử ở cuối của queue.

**+** Hàm void pop(): Xóa đi phần tử đầu của queue.

**+** Hàm void push(T x): Thêm một phần tử có kiểu dữ liệu T vào cuối queue.

* **File Graph.cpp:**
* **Thuộc tính:**

**+** Các biến kiểu số nguyên: nV là số đỉnh của đồ thị, nE là số cạnh của đồ thị và type là loại đồ thị(0 là vô hướng – 1 là có hướng).

**+** Mảng hai chiều số nguyên mtk[100][100] dùng để lưu trữ ma trận kề của đồ thị.

* **Phương thức:**

**+** Hàm void Nhap() dùng để đọc các thông tin của đồ thị(số đỉnh, số cạnh và loại đồ thị).

**+** Hàm void Xuat() biểu diễn ma trận kề của đồ thị.

* **File Tim\_Duong\_Di.cpp:**
* **Thuộc tính:**

**+** Số nguyên n dùng để lưu trữ bậc của ma trận.

**+** Mảng hai chiều số nguyên a[100][100] dùng để lưu trữ ma trận của đồ thị có hướng.

* **Các phương thức:**

**+** Hàm void LoadFile(string fn) dùng để đọc ma trận n hàng, m cột từ file du\_lieu.txt được gán tên là fn.

**+** Hàm int BFS(int s, int f) dùng để tìm đường đi từ gốc s đến đích f của đồ thị.

**+** Hàm void inkq(int\*, int, int) để in kết quả theo yêu cầu bài toán.

**+** Hàm void Out() để nhập đỉnh xuất phát s và đỉnh kết thúc f của đường đi cần tìm. Nếu không có đường đi thì sẽ thông báo “Không có đường đi từ gốc đến đích”.

* **File Me\_Cung.cpp:**
* **Thuộc tính:**

**+** Hai mảng kiểu nguyên hh[]={0,-1,0,1} và hc[]={1,0,-1,0} để kiểm tra vị trí xung quanh điểm cần xét là 0 hay 1.

**+** Mảng số nguyên hai chiều a[100][100] dùng lưu trữ ma trận của mê cung dưới dạng bit 0-1.

**+** Các kiểu số nguyên: n là số hàng của ma trận, m là số cột của ma trận và d là số đường đi được thỏa yêu cầu bài toán.

* **Các phương thức:**

**+** Hàm void LoadFile(string fn) dùng để đọc ma trận n hàng, m cột từ file input.txt được gán tên là fn.

**+** Hàm void Write() dùng để in ma trận vừa đọc ra màn hình.

**+** Hàm void Print() dùng để in kết quả tìm được.

**+** Hàm Try\_MeCung(pair<int,int> bd, pair<int,int> f) dùng để tìm đường đi thỏa yêu cầu bài toán.

**+** Hàm void Path() để nhập tọa độ đầu và cuối của đường cần đi trong mê cung.

* **File Thuat\_Toan.cpp:**
* **Thuộc tính:**

**+** Các kiểu số nguyên: n là số đỉnh, m là số đường đi của đồ thị vô hướng.

**+** Một cấu trúc dữ liệu hàng đợi S dùng lưu lại các đỉnh để duyệt.

**+** Một map tên là Dau với key có kiểu dữ liệu là kiểu số nguyên int và giá trị là bool.

**+** Mảng pair gồm 100 phần tử với cả first và second trong đó có kiểu dữ liệu số nguyên int.

* **Các phương thức:**

**+** Hàm void LoadFile(string fn) dùng để số đỉnh, số đường đi và các đường đi lấy từ file.txt.

**+** Hàm void BFS() dùng để duyệt xem đồ thị có liên thông hay không.

**+** Hàm Write() dùng để in các đường đi vừa đọc ra màn hình.

**+** Hàm Check() dùng để thông báo kết quả kiểm tra đồ thị có liên thông hay không.

* **File Main.cpp:**

Dùng để chạy các file mà bài toán yêu cầu.

* **Phân tích thời gian chạy của từng phương thức có trong các lớp**

1. **File Queue.cpp**

- Queue(): O(1)

- ~ Queue(): O(1)

- bool empty(): O(1)

- int size(): O(1)

- T &front(): O(1)

- T &back(): O(1)

- void pop(): O(1)

- void push(T x): O(n)

1. **File Graph.cpp**

* void Nhap(): O(n^2)
* void Xuat(): O(n^2)

1. **File Tim\_Duong\_Di.cpp**

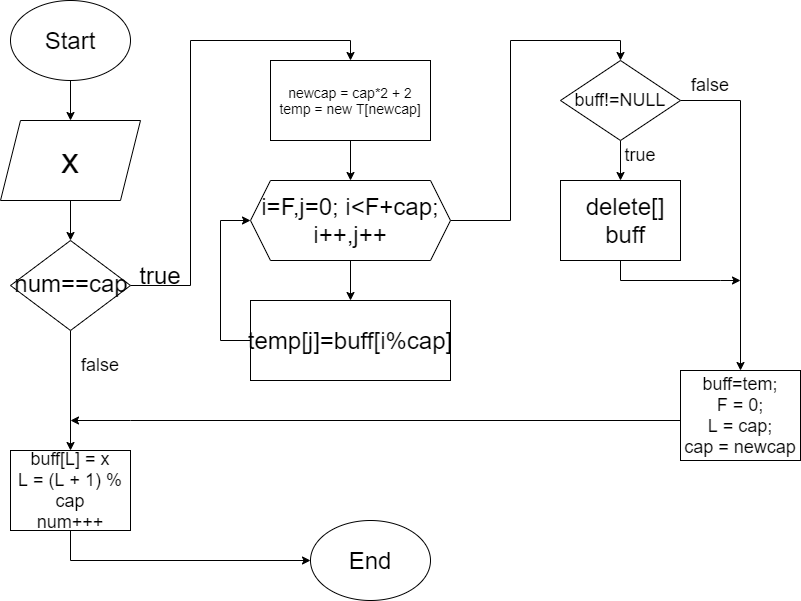
* void LoadFile(string fn): O(n^2)
* int BFS(int s, int f): O(n^2)
* void inkq(int\*, int, int): O(1)
* void Out(): O(n^2)

1. **File Me\_Cung.cpp**

* void LoadFile(string fn): O(n^2)
* void Write(): O(n^2)
* void Print(): O(n)
* void Try\_MeCung(pair<int,int> bd ,pair<int,int> f): O(1)
* void Path(): O(1)

1. **File Thuat\_Toan.cpp**

* void LoadFile(string fn): O(n)
* void BFS(): O(n^2)
* void Write(): O(n)
* void Check(): O(n)
* **Sơ đồ khối**

****

1. **Cài đặt các lớp và hàm main bằng C++**
2. **File Queue.cpp**

#include<bits/stdc++.h>

using namespace std;

#ifndef Queue\_cpp

#define Queue\_cpp

template<class T>

class Queue

{

T \*buff;

int num, cap, F, L;

public:

Queue() {

buff=0;

num=cap=0;

F=L=0;

}

~Queue() {

if(buff) delete[]buff;

buff=0;

}

bool empty() { return num==0; }

int size() { return num; }

T &front() { return buff[F]; }

T &back() {

return L==0?buff[cap-1]:buff[L-1];

}

void pop() {

F=(F+1)%cap;

num--;

}

void push(T x)

{

if(num==cap){

int newcap=cap\*2+2;

T \*tem=new T[newcap];

for(int i=F,j=0;i<F+cap;i++,j++)

tem[j]=buff[i%cap];

if(buff) delete[]buff;

buff=tem;

F=0;

L=cap;

cap=newcap;

}

buff[L]=x;

L=(L+1)%cap;

num++;

}

};

#endif

1. **File Graph.cpp**

#include<bits/stdc++.h>

using namespace std;

#ifndef \_\_Graph\_\_cpp\_\_

#define \_\_Graph\_\_cpp\_\_

template<class G>

class Graph{

private:

int nV ;

int nE ;

int type ;

int mtk[100][100];

public:

void Nhap(){

int dd, dc;

for(int i=1;i<=G.nV;i++)

for(int j=1;j<=nV;j++)

G.mtk[i][j]=0;

for(int k=1;k<=G.nE;k++)

G.mtk[dd][dc]=1;

if(G.type==0)

G.mtk[dd][dc]=1;

}

void Xuat(){

for(int i=1;i<=G.nV;i++)

for(int j=1;j<=G.nV;j++)

G.mtk[i][j];

}

};

#endif

1. **File Tim\_Duong\_Di.cpp**

* **File du\_lieu.txt:**

6

0 1 0 0 0 1

0 0 0 0 1 0

0 1 0 0 0 0

1 0 1 0 0 0

0 0 0 1 0 1

0 1 0 0 0 0

* **Tim\_Duong\_Di.cpp**

#include<bits/stdc++.h>

#include"Queue.cpp"

#ifndef \_\_Tim\_Duong\_Di\_\_cpp\_\_

#define \_\_Tim\_Duong\_Di\_\_cpp\_\_

using namespace std;

class graph{

int n;

int a[100][100];

public:

void LoadFile(string fn);

int BFS(int s, int f);

void inkq(int\*, int, int);

void Out();

};

void graph::Out(){

int f, s;

LoadFile("du\_lieu.txt");

cout<<"Ma tran bieu dien do thi co huong"<<endl;

for(int i=0;i<=n;i++){

a[0][i]=a[i][0]=i;

}

for(int i=0;i<=n;i++){

for(int j=0;j<=n;j++){

cout<<a[i][j]<<" ";

}

cout<<"\n";

}

cout<<"Nhap dinh xuat phat s: ";

cin>>s;

cout<<"Nhap dinh ket thuc f: ";

cin>>f;

if(BFS(s,f));

else cout<<"Khong co duong di tu "<<s<<" den " << f;

}

void graph::inkq(int\* sign, int s, int f){

if(sign[f]==s) cout<<"Co duong di la"<<s<<"->"<<f;

else{

inkq(sign,s,sign[f]);

cout<<"->"<<f;

}

}

int graph::BFS(int s, int f){

Queue<int>Q;

int\* sign=new int[n+2];

int u, v, i;

for(i=1;i<=n;i++) sign[i]=0;

Q.push(s);

sign[s]=1;

while(!Q.empty()){

u=Q.front();

Q.pop();

for(v=1;v<=n;v++)

if(sign[v]==0 && a[u][v]==1){

Q.push(v);

sign[v]=u;

if (v == f){

inkq(sign, s, f);

return 1;

}

}

}

return 0;

}

void graph::LoadFile(string fn){

ifstream f(fn);

f>>n;

for(int i=1;i<=n;i++)

for(int j=1;j<=n;j++){

int t;

f>>t;

a[i][j]=t;

}

}

#endif

1. **File Me\_Cung.cpp**

* **File input.txt:**

7 10

1 1 1 0 0 1 0 0 1 1

1 1 1 0 1 0 0 0 0 0

0 1 0 0 0 1 1 1 0 1

0 0 0 1 0 1 1 1 0 1

0 1 0 0 0 0 0 1 0 0

1 1 0 1 0 1 0 0 0 0

0 1 0 0 0 0 0 1 1 1

* **Me\_Cung.cpp:**

#include<bits/stdc++.h>

#include"Queue.cpp"

#ifndef \_\_Me\_Cung\_\_cpp\_\_

#define \_\_Me\_Cung\_\_cpp\_\_

#define H first

#define C second

using namespace std;

int hh[]={0, -1, 0, 1};

int hc[]={1, 0, -1, 0};

class MeCung{

int n,m;

int a[100][100];

int d;

vector<pair<int, int>> Toado;

public:

void LoadFile(string fn);

void Write();

void Print();

void Try\_MeCung(pair<int,int> bd, pair<int,int> f);

void Path();

};

void MeCung::LoadFile(string fn){

ifstream f(fn);

f>>n>>m;

for(int i=1;i<=n;i++){

for(int j=1;j<=m;j++){

f>>a[i][j];

}

}

f.close();

for(int i=0;i<=n+1;i++){

a[i][0]=a[i][m+1]=1;

}

for(int j=0;j<=m+1;j++){

a[0][j]=a[n+1][j]=1;

}

}

void MeCung::Write(){

for(int i=0;i<=n+1;i++){

for(int j=0;j<=m+1;j++){

cout<<a[i][j]<<" ";

}

cout<<"\n";

}

}

void MeCung::Try\_MeCung(pair<int, int> bd,pair<int,int> f){

queue<pair<int, int>> Q;

Q.push(bd);

while(!Q.empty()){

pair<int, int> u = Q.front();

Q.pop();

Toado.push\_back(u);

bool check = true;

for(int t=0; t<=3; t++){

pair<int,int>x;

x.H = u.H + hh[t];

x.C = u.C + hc[t];

if(x == f){

cout << "Co duong di." << endl;

return;

}

if(a[x.H][x.C] == 0){

Q.push(x);

a[x.H][x.C] = 1;

check = false;

}

}

if(check) Toado.pop\_back();

}

}

void MeCung::Print(){

cout<<"Cach di "<<d+1<<":";

int n = Toado.size();

for(int i = 0; i < n - 1; i++){

cout<<"("<<Toado[i].H<<","<<Toado[i].C<<")->";

}

cout<<"("<<Toado[n-1].H<<","<<Toado[n-1].C<<")";

}

void MeCung::Path(){

pair<int,int> s,f;

d=0;

LoadFile("input.txt");

cout<<"\n\nMe cung"<<endl;

Write();

cout<<"(0 la duong - 1 la tuong)" ;

cout<<"\nNhap toa do xuat phat: ";

cin>>s.H>>s.C;

cout<<"Nhap toa do ket thuc: ";

cin>>f.H>>f.C;

a[s.H][s.C]=1;

Try\_MeCung();

}

#endif

1. **File Thuat\_Toan.cpp**

* **File file.txt:**

8 16

1 2

1 6

2 1

2 3

3 2

3 4

4 3

4 5

5 4

5 6

6 1

6 5

6 7

7 6

7 8

8 7

* **Thuat\_Toan.cpp:**

#include<bits/stdc++.h>

#ifndef \_\_Thuat\_Toan\_\_cpp\_\_

#define \_\_Thuat\_Toan\_\_cpp\_\_

using namespace std;

map<int, bool>Dau;

pair<int, int> P[100];

class Thuat\_Toan{

int n, m;

Queue<int> S;

public:

void LoadFile(string fn);

void BFS();

void Write();

void Check();

};

void Thuat\_Toan::Check(){

cout<<"\n\nCac duong cua do thi vo huong\n";

LoadFile("file.txt");

for(int i=0; i<m; i++)

cout<<P[i].first <<" " <<P[i].second<<endl;

S.push(1);

BFS();

Write();

}

void Thuat\_Toan::LoadFile(string fn){

ifstream f(fn);

f >> n >> m;

int i=0;

while(i<m){

f >> P[i].first >> P[i].second;

i++;

}

}

void Thuat\_Toan::BFS(){

while(!S.empty()){

int u = S.front();

S.pop();

Dau[u]= true;

for(int i=0; i<m; i++){

if(u == P[i].first) {

if(Dau[ P[i].second ] == false){

S.push(P[i].second);

Dau[P[i].second] = true;

}

}

else if(u == P[i].second) {

if(Dau [P[i].first ] == false){

S.push(P[i].first);

Dau[P[i].first] = true;

}

}

}

}

}

void Thuat\_Toan::Write(){

for(int i=1; i<=n; i++){

if(Dau[i] == false){

cout<<"=> DO THI KHONG LIEN THONG";

return ;

}

}

cout<<"=> DO THI CO LIEN THONG";

}

#endif

1. **File Main.cpp**

#include<bits/stdc++.h>

#include"Tim\_Duong\_Di.cpp"

#include"Me\_Cung.cpp"

#include"Thuat\_Toan.cpp"

using namespace std;

int main()

{

//Tim duong di trong do thi co huong

graph G;

G.Out();

//Tim duong di trong me cung

MeCung M;

M.Path();

//Kiem tra tinh lien thong cua do thi vo huong

Thuat\_Toan T;

T.Check();

}

1. **Danh sách tài liệu tham khảo**

* **Giáo trình Cấu trúc dữ liệu và giải thuật.**
* <https://www.geeksforgeeks.org/>
* <https://tek4.vn/khoa-hoc>
* <https://expressmagazine.net/>
* <https://www.facebook.com/profile.php?id=100044571123353>
* **Mạng Internet.**