四電子二B 曾皓暐 B10213147 Homework#1

Graph1:

V5

V3

V6

V6

V2

V4

V1

V2

V3

V4

V5

V6

V9

V7

V8

∞

∞

2

2

1

0

1

∞

∞

V1

V2

V3

V4

V5

V6

V9

V7

V8

∞

∞

2

2

1

0

1

2

∞

V2

V4

V7

V1

V2

V3

V4

V5

V6

V9

V7

V8

∞

∞

∞

∞

1

0

1

∞

∞

V1

V2

V3

V4

V5

V6

V9

V7

V8

∞

∞

∞

∞

∞

0

∞

∞

∞

V1

V2

V3

V4

V5

V6

V9

V7

V8

∞

∞

∞

∞

∞

∞

∞

∞

∞

V1

V2

V3

V4

V5

V6

V9

V7

V8

∞

3

2

2

1

0

1

2

∞

V1

V7

V4

V1

V2

V3

V4

V5

V6

V9

V7

V8

∞

3

2

2

1

0

1

2

∞

V1

V7

V1

V2

V3

V4

V5

V6

V9

V7

V8

3

3

2

2

1

0

1

2

3

V9

V8

V1

V1

V2

V3

V4

V5

V6

V9

V7

V8

3

3

2

2

1

0

1

2

3

V9

V8

V1

V2

V3

V4

V5

V6

V9

V7

V8

3

3

2

2

1

0

1

2

3

V9

V1

V2

V3

V4

V5

V6

V9

V7

V8

3

3

2

2

1

0

1

2

3

ф

Search path

v5→v3→v6→v2→v4→v7→v1→v8→v9→Done

Graph2:

∞

∞

∞

∞

∞

∞

∞

∞

∞

∞

V1

V2

V3

V10

V7

V4

V5

V6

V8

V9

∞

∞

∞

∞

∞

∞

∞

∞

∞

0

V1

V2

V3

V10

V7

V4

V5

V6

V8

V9

v7

∞

1

1

∞

∞

∞

∞

∞

1

0

V1

V2

V3

V10

V7

V4

V5

V6

V8

V9

v8

v4

v6

2

1

1

∞

∞

∞

2

∞

1

0

V1

V2

V3

V10

V7

V4

V5

V6

V8

V9

v5

v1

v8

v6

2

1

1

∞

∞

∞

2

∞

1

0

V1

V2

V3

V10

V7

V4

V5

V6

V8

V9

v1

v5

v8

2

1

1

2

∞

∞

2

∞

1

0

V1

V2

V3

V10

V7

V4

V5

V6

V8

V9

v10

v1

v5

2

1

1

2

∞

3

2

3

1

0

V1

V2

V3

V10

V7

V4

V5

V6

V8

V9

v3

v2

v10

v5

2

1

1

2

∞

3

2

3

1

0

V1

V2

V3

V10

V7

V4

V5

V6

V8

V9

v3

v2

v10

2

1

1

2

3

3

2

3

1

0

V1

V2

V3

V10

V7

V4

V5

V6

V8

V9

v9

v3

v2

2

1

1

2

3

3

2

3

1

0

V1

V2

V3

V10

V7

V4

V5

V6

V8

V9

v3

v9

2

1

1

2

3

3

2

3

1

0

V1

V2

V3

V10

V7

V4

V5

V6

V8

V9

v9

2

1

1

2

3

3

2

3

1

0

V1

V2

V3

V10

V7

V4

V5

V6

V8

V9

ф

Search path:

v7→v4→v6→v8→v1→v5→v10→v2→v3→v9→Done

Graph3:

∞

∞

∞

∞

∞

V1

V2

V3

V4

V5

∞

∞

∞

∞

0

V1

V2

V3

V4

V5

V3

∞

1

1

1

0

V1

V2

V3

V4

V5

V4

V2

V1

∞

1

1

1

0

V1

V2

V3

V4

V5

V2

V4

∞

1

1

1

0

V1

V2

V3

V4

V5

V4

2

1

1

1

0

V1

V2

V3

V4

V5

V5

2

1

1

1

0

V1

V2

V3

V4

V5

ф

Search path:

V3→V1→V2→V4→V5→Done

Program code

#include <stdio.h>

#include <stdlib.h>

#define White 0

#define Gray 1

#define Black 2

#define Empty -1

#define Size 15

/\* run this program using the console pauser or add your own getch, system("pause") or input loop \*/

const int ASCII0=48;

int Table[Size][Size];

int Queue[Size],Color[Size],Path[Size],D[Size],Up[Size];

int u,p=0,Qn=0;

int main(int argc, char \*argv[]) {

int i,s,out,v;

/\*Initialize\*/

for(i=0;i<Size;i++){

Queue[i]=Empty;

D[i]=Empty;

Up[i]=Empty;

Path[i]=Empty;

Color[i]=White;

}

open\_file();

modify\_Table();

/\*Setting start position\*/

s=Table[0][0];

Color[s]=Gray;

D[s]=0;

Up[s]=Empty;

EnQueue(s);

while(Queue[0]!=Empty){

u=DeQueue(out);

i=1;

while(Table[u][i]!=Empty){

v=Table[u][i]; //Connected vertex of u

if(Color[v]==White){

Color[v]=Gray;

D[v]=D[u]+1;

Up[v]=u;

EnQueue(v);

}

i++;

}

Color[u]=Black; //Black means the vertex is searched

show\_Status(u);

}

show\_Result();

return 0;

}

void open\_file(void) {

FILE \*fp;

int i,count=0;

char fname[50],line[Size];

/\*Enter a file name\*/

printf("Enter a file:");

gets(fname);

/\*convert txt to array\*/

fp = fopen(fname, "r");

if(fp!=NULL) {

printf("Opening file:[%s]\n", fname);

while(fgets(line,Size,fp)!=NULL) {

printf("%s",line);

for(i=0; i<Size; i++) {

Table[count][i]=line[i];

}

count++;

}

fclose(fp);

}

else{

printf("Open [%s] file fail.", fname);

exit(1);

}

printf("\n");

}

void modify\_Table(void){

int i,j,n,sum=0;

/\*Convert ASCII to integer and remove the control characters\*/

for(i=0;i<Size;i++){

n=0;sum=0;

for(j=0;j<Size;j++){

if(Table[i][j]!=0){

if(Table[i][j]>=ASCII0 && Table[i][j]<=ASCII0+9)

sum=sum\*10+Table[i][j]-ASCII0; //For the value is bigger than 10

else{

Table[i][n]=sum;

sum=0;

n++;

}

}

else break;

}

for(;n<Size;n++)Table[i][n]=Empty; //Fill "Empty" to unused space

}

}

void EnQueue(int n){

Queue[Qn]=n;

Qn++;

}

int DeQueue(int out){

int i;

/\*Pop first vertex out\*/

out=Queue[0];

Path[p]=Queue[0];

for(i=1;i<=Qn;i++) Queue[i-1]=Queue[i]; //Left shift a space

Qn--;p++;

return out;

}

void show\_Status(int now){

int i;

printf("Choosen vertex is %d\n",now);

printf("Next vertex is\n");

i=1;

while(Table[now][i]!=Empty){

printf("node=%d ",Table[now][i]);

printf("Distance is %d ",D[Table[now][i]]);

if(Color[Table[now][i]]==Black) printf("Visited Vertex -> Throw Away");

printf("\n");

i++;

}

printf("\n");

}

void show\_Result(void) {

int i;

printf("The final path is\n");

for(i=0;i<p;i++) printf("%d -> ",Path[i]);

printf("Done!\n");

printf("BFS is finished!\n");

}



