/\*

This code can be compiled and run ok.

Given a map of color tiles which we try to eliminate the same color tiles from the blank is the key.

Without the same color tiles we can eliminate, the game is over.

usage (how to run):

fc (then input n and alpha)

input file:

map

output files:

none

compile (how to compile):

g++ -o fc hw3.cpp

pseudocode:

While still some tiles can be eliminated

For each row in map

For each column in map

If map[row][column] is a space, then

Find up neighboring tiles, and save to NT (Step1)

Find down neighboring tiles, and save to NT (Step1)

Find left neighboring tiles, and save to NT (Step1)

Find right neighboring tiles, and save to NT (Step1)

Count number of each color, and save to SUM (Step2)

Eliminate tiles which have duplicated color (Step2)

coded by Wei-Ming Hou, ID: h34996158, email: jacob0701@hotmail.com.tw

date: 2011.03.29

\*/

#include <iostream> //for cout, cin

#include <fstream> //for ifstream, ofstream

#include <cstdio> //for printf

#include <windows.h> //for hConsole, if the operating system you use is not Windows, please comment this line

using namespace std;

//predefined function prototype

void print\_map(int\*\*,int,int);

void color\_print\_map(int\*\*,int,int); //if the operating system you use is not Windows, please comment this line

int main(){

//read map file and store into dynamic allocation of arrays------

ifstream inFile("map.txt");

int map\_height, map\_width;

inFile >> map\_height >> map\_width;

int \*\*map = new int\*[map\_height+1];

int i,j;

for(i = 1; i <= map\_height; i++){

map[i] = new int[map\_width+1];

for(j = 1; j <= map\_width; j++)

inFile >> map[i][j];

}

inFile.close();

//---------------------------------------------------------------

//print map------------------------------------------------------

cout << "Origin map:" << endl;

print\_map(map, map\_height, map\_width);

cout << endl;

//---------------------------------------------------------------

//answer section-------------------------------------------------

int NT[4][3] = {{0},{0}}; //統計每點移動上下左右之座標

int SUM[10] = {0}; //統計上下左右之座標之顏色

int ux=0, uy=0, dx=0, dy=0, lx=0, ly=0, rx=0, ry=0; //上下左右之座標

int count=0, test=0; //統計總共所消之tiles和最後檢驗

while(1)

{

for(i = 1; i <= map\_height; i++) //自(1,1)開始每格尋找可消之方格

{

for(j = 1; j <= map\_width; j++)

{

for(int u=0; u<=9; u++) //clean

SUM[u]=0;

if(map[i][j]==0)

{

for(ux=i, uy=j; ux > 0; ux--) //往上檢查

{

NT[0][0] = map[ux][uy];

NT[0][1] = ux;

NT[0][2] = uy;

SUM[map[ux][uy]]+=1;

if(map[ux][uy]!=0)

break;

}

for(dx=i, dy=j; dx <= map\_height; dx++) //往下檢查

{

NT[1][0] = map[dx][dy];

NT[1][1] = dx;

NT[1][2] = dy;

SUM[map[dx][dy]]+=1;

if(map[dx][dy]!=0)

break;

}

for(lx=i, ly=j; ly > 0; ly--) //往左檢查

{

NT[2][0] = map[lx][ly];

NT[2][1] = lx;

NT[2][2] = ly;

SUM[map[lx][ly]]+=1;

if(map[lx][ly]!=0)

break;

}

for(rx=i, ry=j; ry <= map\_width; ry++) //往右檢查

{

NT[3][0] = map[rx][ry];

NT[3][1] = rx;

NT[3][2] = ry;

SUM[map[rx][ry]]+=1;

if(map[rx][ry]!=0)

break;

}

}

for(int s=1; s<=9; s++) //檢查SUM中是否有相同顏色出現兩次以上

{

if(SUM[s]>=2)

{

for(int t=0; t<=3; t++) //檢查NT中顏色是否相同

{

if(NT[t][0]==s)

{

switch(t) //當有顏色相同時，值變為0

{

case 0: map[ux][uy]=0;

count++;

break;

case 1: map[dx][dy]=0;

count++;

break;

case 2: map[lx][ly]=0;

count++;

break;

case 3: map[rx][ry]=0;

count++;

break;

}

}

}

}

}

}

}

if(test==count) //檢測是否還有未消除之tiles

break;

test=count;

}

cout << "After map:" << endl;

print\_map(map, map\_height, map\_width);

cout << "Eliminate tiles : " << count << endl;

//---------------------------------------------------------------

return 0;

}

//predefined functions

void print\_map(int \*\*map, int map\_height, int map\_width){

int i,j;

for(i = 1; i <= map\_height; i++){

for(j = 1; j <= map\_width; j++){

printf("%3d",map[i][j]);}

cout << endl;}

}

void color\_print\_map(int \*\*map, int map\_height, int map\_width){ //if the operating system you use is not Windows, please comment this function

int color[]={247,231,215,199,183,167,151,103,87,71,55};

HANDLE hConsole;

hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);

int i,j;

for(i = 1; i <= map\_height; i++){

for(j = 1; j <= map\_width; j++){

SetConsoleTextAttribute(hConsole, color[map[i][j]]);

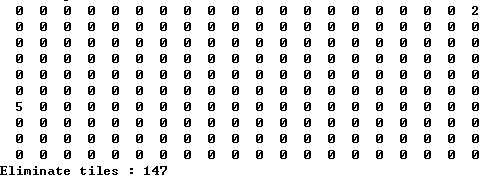
printf("%3d",map[i][j]);

}cout << endl;

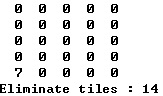
}SetConsoleTextAttribute(hConsole, 15);

}

map.txt



map1.txt



map2.txt

