#include <stdio.h>

#include <iostream>

#include <opencv2/highgui/highgui.hpp>

#include <opencv2/imgproc/imgproc.hpp>

#include <opencv2/core/core.hpp>

#include <queue>

using namespace cv;

using namespace std;

struct p

{

int i;

int j;

}obj, obj1;

queue<p>S;

Mat img = imread("paint.png",CV\_LOAD\_IMAGE\_GRAYSCALE);

Mat img1(img.rows, img.cols, CV\_8UC1, Scalar(0));

int c = 0, th=50;

void dfs(void)

{

//int flag = 0;

img1.at<uchar>(obj.i, obj.j) =c;

for (int l = obj.i-1; l <=obj.i+1;l++){

for (int k = obj.j - 1; k <= obj.j + 1; k++)

{

if (l >= 0 && l < img.rows&&k >= 0 && k < img.cols)

{

if (img.at<uchar>(l, k) < (255 - th) && (l != obj.i || k != obj.j) && (img.at<uchar>(l, k) - img.at<uchar>(obj.i, obj.j)) < th && (img1.at<uchar>(l,k)==0))

{

// flag = 1;

//cout << c << " searching for neighb of i,j " << obj.i << " " << obj.j << "\n";

obj.i = l;

obj.j = k;

S.push(obj);

//cout << c<<"found neighb i,j "<<l<<" "<<k<<"\n";

img1.at<uchar>(obj.i, obj.j) = c;

//dfs(l, k);

}

}

}

}}

//if (flag == 0)

//S.pop();

int main()

{

for (int i = 0; i < img.rows;i++)

for (int j = 0; j < img.cols;j++)

if (img.at<uchar>(i, j) < (255 - th) && img1.at<uchar>(i, j) == 0)

{

cout << i << " " << j<<" ";

obj.i = i;

obj.j = j;

S.push(obj);

c++;

//mark.at<uchar>(obj.i, obj.j) = c;

while (!S.empty())

{

//cout << "popping " << S.top().i << " " << S.top().j << "\n";

S.pop();

if (!S.empty())

obj = S.front();

dfs();

//mark.at<uchar>(obj.i, obj.j) = c;

}

//cout << obj.i << " " << obj.j<<" \n";

}

for (int i = 0; i < img.rows; i++)

for (int j = 0; j < img.cols; j++)

if (img1.at<uchar>(i,j)!=0)

img1.at<uchar>(i, j) = 255.0 / img1.at<uchar>(i, j);

imshow("Orig", img);

imshow("Dfs: ", img1);

cout << c;

waitKey(0);

}