#include "pch.h"

#include <iostream>

#pragma once

#include <string>

#include <fstream>

#include <cstdint>

#include "ecrireimage.h"

using namespace std;

void test();

int borneDansIntervalle();

bool estDansIntervalle();

double aleatoireZeroUn();

int aleatoireSelonDistribution();

static constexpr unsigned tailleX = 500, tailleY = 500;

int borneDansIntervalle(int valeurEntier, int borneSup, int borneInf) //valeurEntier INOUT \*/

{

if (valeurEntier < borneInf)

{

valeurEntier = borneInf;

}

if (valeurEntier > borneSup)

{

valeurEntier = borneSup;

}

else

{

valeurEntier = valeurEntier;

}

return valeurEntier;

}

bool estDansIntervalle(int valeurEntier, int borneSup, int borneInf) // valeurEntier INOUT \*/

{

if (valeurEntier >= borneInf && valeurEntier <= borneSup)

{

return true;

}

else

{

return false;

}

return valeurEntier;

}

double aleatoireUn() // rand()%10/10 OUT \*/

{

srand(time(NULL));

return rand() % 10 / 10.0;

}

int aleatoireSelonDistribution(double tableau[]) //tableau[] IN \*/

{

int i = 0;

double valeur = aleatoireUn();

while (tableau[i] != 1)

{

if (valeur < tableau[i])

{

return i;

}

i++;

}

return 0;

}

double dessinePoint(double x, double y, int intensite, int image[tailleX][tailleY]) // x,y, INOUT \*/

{

int pixel[tailleX][tailleY];

int result[tailleX][tailleY];

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

for (int y = 0; y <= tailleY; y++)

result[i][y] = image[i][y] - intensite;

}

}

for (int z = 0; z <= tailleX; z++) {

for (int a = 0; a <= tailleY; a++) {

if (result[z][a] > 255)

result[z][a] = 255;

if (result[z][a] < 0)

result[z][a] = 0;

}

}

int aleatoireSelonDistribution(double tableau[]) {

int i = 0;

double valeur = aleatoireUn();

while (tableau[i] != 1)

{

if (valeur < tableau[i])

{

return i;

}

i++;

}

return 0;

}

double transformePoint(double x, double y) // x,y INOUT \*/

{

double transformation[3][2];

double a, b;

a = (transformation[0][0] \* x + transformation[0][1] \* y) + transformation[2][0];

b = (transformation[1][0] \* x + transformation[1][1] \* y) + transformation[2][1];

return transformePoint;

}

void test() {

}

const int nbr\_texte = 5;

struct CalculerImage {

int nbr\_point\_depart;

int Nbr\_transformation\_depart;

int Nbr\_transformation\_app;

int intensite;

};

struct IFS {

double transformation[6];

double probabilite;

};

struct calculerImage { CalculerImage texte[nbr\_texte]; IFS textes[nbr\_texte]; int nombreTexte; };

string fichiernom;

cout << "le nom du fichier a lire";

cin >> fichiernom;

ifstream fichier(fichiernom);

if (fichier.is\_open()) {

int nbrTexte = 0;

while (!ws(fichier).eof()) {

calculerImage num\_texte;

IFS textes;

int nbr\_valeur, nbr\_chiffre;

nbr\_valeur++;

fichier >> num\_texte.texte[nbrTexte].nbr\_point\_depart >>

num\_texte.texte[nbrTexte].Nbr\_transformation\_depart >>

num\_texte.texte[nbrTexte].Nbr\_transformation\_app >>

num\_texte.texte[nbrTexte].intensite >>

num\_texte.textes[nbrTexte].transformation[6] >>

num\_texte.textes[nbrTexte].probabilite;

nbrTexte++;

nbr\_chiffre = nbr\_valeur - 10;

for (int i = 0; i < (num\_texte.texte[nbrTexte].nbr\_point\_depart)\*2; i++)

{

cout << aleatoireZeroUn() << endl;

for (int i = 0; i < (num\_texte.texte[nbrTexte].Nbr\_transformation\_app)\*2;i++)

{

cout << aleatoireSelonDistribution() << endl;

for (int i=0; i<(num\_texte.texte[nbrTexte].nbr\_point\_depart)\*2; i++)

{

cout << transformePoint(double 2 \* i, double 2 \* i + 1) << endl;

}

int main() {

test();

calculerImage();

return 0;

}