Спецификация программы

На вход программы подается:

image – изображение,

rate – калибровка,

threshold – порог бинаризации

Результат преобразований и вычислений выводится на экран. Результат преобразований выводится в отдельном окне, результат вычислений выводится в отдельную таблицу.

Описание требований к параметрам:

Исходное изображение – цветовое(24 бита на цвет), прямоугольное(например 800\*600), размер не более 1920\*1080. Калибровка – вещественное число больше нуля. Порог бинаризации – целое число от 0 до 255.

Определим классы эквивалентности для входных данных:

Для image:

Изображение отсутствует – ошибочно.

Изображение размером больше 1920\*1080 – ошибочно.

Изображение с отрицательным размером – ошибочно.

Изображение размером до 1920\*1080 – корректно.

Для rate:

Rate < 0 – ошибочно

Rate – не число – ошибочно

Rate > 0 – корректно

Для threshold:

Threshold < 0 – ошибочно

Threshold > 255 – ошибочно

Threshold – вещественное число – ошибочно

Threshold – не число – ошибочно

Threshold in [0,255] – корректно.

Спецификации модулей:

*А. private void SquareToolforStack(Stack<Point> needCheckedPixel, int [,] matrix, int newNum, int x, int y)*

Входные данные:

1. needCheckedPixel – рабочий стек для проверки пикселей типа «точка»

2. matrix – рабочая матрица размером равным размеру исходного изображения

3. newNum – номер заполняемой частицы

4. x,y – координаты на матрице

Классы эквивалентности:

1. needCheckedPixel не инициализирован – ошибочно

needCheckedPixel пуст – корректно

needCheckedPixel не пуст – корректно

2. matrix не существует – ошибочно

Matrix не инициализирован – ошибочно

Matrix – размерность больше 1920\*1080 – ошибочно

Matrix – размерность меньше 0\*0 – ошибочно

Matrix инициализирована и размерностью от 0\*0 до 1920\*1080 – корректно

3. newNum < 0 – ошибочно

newNum не число – ошибочно

newNum вещественное – ошибочно

newNum > 1920\*1080 – ошибочно

0 < newNum < 1920\*1080 – корректно

4. x,y не числа – ошибочно

x,y < 0 – ошибочно

x,y вещественные – ошибочно

X > 1920 – ошибочно

Y > 1080 – ошибочно

0<x<1920, 0<y<1920 – корректно

*Б. int SquareCalculate(int x, int y, int num, int[,] matrix, int h, int w)*

Входные данные:

1. x,y – координаты на рабочей матрице

2. num – номер частицы

3. matrix – рабочая матрица размером равным размеру исходного изображения

4. h – высота изображения

5. w – ширина изображения

Классы эквивалентности:

1,2,3 аналогично 4,3,2 модуля А.

4. h – не число – ошибочно

H < 0 – ошибочно

H > 1920 – ошибочно

H – вещественное – ошибочно

H in [0,1920] – корректно.

5. w – не число – ошибочно

w < 0 – ошибочно

w > 1080 – ошибочно

w – вещественное – ошибочно

w in [0,1080] – корректно.

*В. private void GetRGB(Bitmap img, int x, int y, int r, int g, int b)*

Входные данные:

1. Img – исходное изображение размером до 1920\*1080

2. X,Y – координаты на изображении

3. R,g,b – значения цветовых компонент.

Классы эквивалентности:

1 аналогично классам эквивалентности matrix в спецификации программы

2. аналогично 4. Модуля А

3. r,g,b – не числа – ошибочно

R < 0 || G < 0 || B < 0 – ошибочно

R > 255 || G > 255 || B > 255 – ошибочно

R,g,b – вещественные – ошибочно

R in [0,255] && G in [0,255] && B in [0,255] – корректно

*Г. private double ColourComponent(Bitmap img, int x, int y)*

Входные данные:

1. Img – исходное изображение размером до 1920\*1080

2. X,Y – координаты на изображении

Классы эквивалентности:

1,2 аналогично 1,2 модуля В.

*Д. private void FuseToolforStack(Stack<Point> needCheckedPixel, Bitmap img, int x, int y)*

Входные данные:

1. needCheckedPixel – рабочий стек для проверки пикселей типа «точка»

2. Img – исходное изображение размером до 1920\*1080

3. X,Y – координаты на изображении

Классы эквивалентности:

2,3 аналогично 1,2 модуля В. 1 аналогично 1 модуля А.

*Е. private void Fuse(int x, int y, Bitmap img, int h, int w)*

Входные данные:

1. X,Y – координаты на изображении

2. Img – исходное изображение размером до 1920\*1080

3. h – высота изображения

4. w – ширина изображения

Классы эквивалентности:

2,1 аналогично 1,2 модуля В. 3,4 аналогично 4,5 модуля Б.

*Ж. private int ExtremateColor(int source, int changeval)*

Входные данные:

1. source – значение цветовой компоненты

2. changeval – прибавляемое значение компоненты.

Классы эквивалентности:

1. source < 0 – ошибочно

Source > 255 – ошибочно

Source – не число – ошибочно

Source – вещественное – ошибочно

Source in [0,255] – корректно

2. changeval < 0 – ошибочно

Changeval > 0 – корректно

Changeval – не число – ошибочно

Changeval – вещественное – ошибочно.

Разработка тестов для модулей:

*А. private void SquareToolforStack(Stack<Point> needCheckedPixel, int [,] matrix, int newNum, int x, int y)*

1. Point == null;

2. некорректные x,y,newNum

3. matrix == null

*Б. int SquareCalculate(int x, int y, int num, int[,] matrix, int h, int w)*

1. Point == null;

2. некорректные x,y,num

3. matrix == null

*Г. private double ColourComponent(Bitmap img, int x, int y)*

1. некорректные img,x,y

*Д. private void FuseToolforStack(Stack<Point> needCheckedPixel, Bitmap img, int x, int y)*

1. некорректные входные needcheckedPixel, img, x,y

2. Colour=255

3. Point = null

*E. private void Fuse(int x, int y, Bitmap img, int h, int w)*

1. некорректные входные данные

2. needCheckedPixel == null

*Ж.private void BorderProcessingHelper(Bitmap img, int height, int width, int x, int y)*

1. некорректные входные данные

2. colour == 255

*З. private void BorderProcessing(Bitmap img)*

1. некорректные входные данные

2. некорректные h,w

*И. private void Negativ(Bitmap img)*

1. некорректные входные данные

2. r = 255 – r < 0 (аналогично для g,b)

*К. private int ExtremateColor(int source, int changeval)*

1. некорректные входные данные

*Л. public void ConvertImage(Bitmap image)*

1. некорректные входные данные

2. в матрице не только нули и единицы.

*М. !!! public void MethodOtsu()*

1. некорректно создались массивы

2. некорректно заполнились массивы

3. res not in [0,255]

4. total = 0

5. q1,q2 == 0

*Н. private bool IsBorder(int[,] a, int I, int J, int h, int w)*

1. пиксель внутри границ

2. пиксель вне границ