Самостоятельное задание №1

Задача: реализовать метод дизеринга в графической системе

Математическая модель: Для метода Флойда-Штайнберга:

p=0..1, yij=0..1 – цветовые значения пикселей; e – значение квантификационной ошибки для пикселя; y=t(x)=[x / 255] – функция для вычисления похожего цвета; e – значение квантификационной ошибки, вычисляемое как p – t(p).

Список идентификаторов в программе:

|  |  |  |
| --- | --- | --- |
| Имя переменной в программе | Описание переменной | Тип данных |
| y | индекс ряда | int |
| x | индекс столбца | int |
| old\_pixel | текущее значение пикселя | float |
| new\_pixel | новое значение пикселя | float |
| err | значение ошибки | float |
| pixels | список списков пикселей | list |
| similar\_color | функция вычисляющая новый цвет пикселя для расчета квантификационной ошибки | int |
| fs\_dithering | функция, модифицирующая переданный список списков пикселей, применяя к нему дизеринг | list |

Код программы:

def fs\_dithering(pixels):  
 *"""Floyd-Steiner dithering. Affect passed pixels"""* for y in range(len(pixels) - 1):  
 for x in range(len(pixels[y]) - 1):  
 old\_pixel = pixels[x][y]  
 new\_pixel = similar\_color(old\_pixel)  
 pixels[x][y] = new\_pixel  
  
 err = old\_pixel - new\_pixel  
  
 pixels[x + 1][y] = pixels[x + 1][y] + err \* 7 / 16  
 pixels[x - 1][y + 1] = pixels[x - 1][y + 1] + err \* 3 / 16  
 pixels[x][y + 1] = pixels[x][y + 1] + err \* 5 / 16  
 pixels[x + 1][y + 1] = pixels[x + 1][y + 1] + err \* 1 / 16  
  
 return pixels  
  
  
def similar\_color(old\_pixel):  
 return round(old\_pixel / 255, 2)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 m = [  
 [0, 0, 0],  
 [0, 1, 0],  
 [0, 0, 0]  
 ]  
 print(fs\_dithering(m))

Протокол работы программы:

