**ПРИЛОЖЕНИЕ А**

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

# КАФЕДРА ИНТЕЛЛЕКТУАЛЬНЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Проектирование системы параллельной обработки с использованием MPI

**ТЕКСТ ПРОГРАММЫ**

КП.ПО4.190348-03 81 00

## Листов 3

|  |  |
| --- | --- |
| Руководитель | Ю.В. Савицкий |
| Выполнил | Д.А Синяк |
| Консультант |  |
| по ЕСПД | Ю.В. Савицкий |
|  |  |

2021

**СОДЕРЖАНИЕ**

**Код программы:**

from mpi4py import MPI

import time

from PIL import Image, ImageDraw

from multiprocessing import Pool

from math import floor

():

draw = ImageDraw.Draw(image)

width = image.size[0]

height = image.size[1]

pix = image.load()

start\_time = time.time()

for i in range(width): # инверсия

for j in range(height):

a = pix[i, j][0]

b = pix[i, j][1]

c = pix[i, j][2]

draw.point((i, j), (255 - a, 255 - b, 255 - c))

for i in range(1, width - 1):

for j in range(1, height - 1):

a = int((pix[i - 1, j - 1][0] + pix[i - 1, j][0] + pix[i - 1, j + 1][0] +

pix[i, j - 1][0] + pix[i, j][0] + pix[i, j + 1][0] +

pix[i + 1, j - 1][0] + pix[i + 1, j][0] + pix[i + 1, j + 1][0]) / 9)

b = int((pix[i - 1, j - 1][1] + pix[i - 1, j][1] + pix[i - 1, j + 1][1] +

pix[i, j - 1][1] + pix[i, j][1] + pix[i, j + 1][1] +

pix[i + 1, j - 1][1] + pix[i + 1, j][1] + pix[i + 1, j + 1][1]) / 9)

c = int((pix[i - 1, j - 1][2] + pix[i - 1, j][2] + pix[i - 1, j + 1][2] +

pix[i, j - 1][2] + pix[i, j][2] + pix[i, j + 1][2] +

pix[i + 1, j - 1][2] + pix[i + 1, j][2] + pix[i + 1, j + 1][2]) / 9)

draw.point((i, j), (a, b, c))

image.save("NewLion.bmp", "BMP")

print(time.time() - start\_time)

del draw

def parral(tuple):

img = Image.open("Lion.bmp")

DImage = img.copy()

draw = ImageDraw.Draw(DImage)

pix = DImage.load()

step, startSt = tuple

height = img.size[1]

for i in range(startSt, step+startSt):

for j in range(0, height):

a = pix[i, j][0]

b = pix[i, j][1]

c = pix[i, j][2]

draw.point((i, j), (255 - a, 255 - b, 255 - c))

for i in range(startSt + 1, step+startSt - 1):

for j in range(1, height - 1):

a = int((pix[i - 1, j - 1][0] + pix[i - 1, j][0] + pix[i - 1, j + 1][0] +

pix[i, j - 1][0] + pix[i, j][0] + pix[i, j + 1][0] +

pix[i + 1, j - 1][0] + pix[i + 1, j][0] + pix[i + 1, j + 1][0]) / 9)

b = int((pix[i - 1, j - 1][1] + pix[i - 1, j][1] + pix[i - 1, j + 1][1] +

pix[i, j - 1][1] + pix[i, j][1] + pix[i, j + 1][1] +

pix[i + 1, j - 1][1] + pix[i + 1, j][1] + pix[i + 1, j + 1][1]) / 9)

c = int((pix[i - 1, j - 1][2] + pix[i - 1, j][2] + pix[i - 1, j + 1][2] +

pix[i, j - 1][2] + pix[i, j][2] + pix[i, j + 1][2] +

pix[i + 1, j - 1][2] + pix[i + 1, j][2] + pix[i + 1, j + 1][2]) / 9)

draw.point((i, j), (a, b, c))

DImage = DImage.crop((startSt, 0, startSt + step + 1, height))

return startSt, DImage

if name == 'main':

image = Image.open("Lion.bmp")

NewImage = image.copy()

DoIt()

p = Pool(4)

start\_time = time.time()

parts = floor(image.size[0] / 4)

arr = []

for i in range(4):

arr.append((parts, parts \* i))

dic = dict()

dic.update(p.map(parral, arr))

for i in range(0, 4 \* parts, parts):

NewImage.paste(dic[i], (i, 0))

NewImage.save("NewLion.bmp", "BMP")

print(time.time() - start\_time)