**Министерство образования Республики Беларусь**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики и информатики**

**Кочева Ильи Александровича**

Отчет по лабораторной работе № 3,

студента 3-го курса 3-ой группы

2025

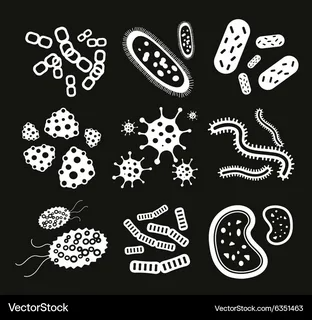
**Отчет по заданию: Построение выпуклой оболочки**

**1. Цель работы**

Разработать программу построения выпуклой оболочки бинарного объекта на основе преобразований утолщения.

**2. Исходные данные**

В качестве исходных данных использовались следующие бинарные изображения:

**Исходное изображение №1**  


**Исходное изображение №2**  


**Исходное изображение №3  
**

**Исходное изображение №4**

****

**Исходное изображение №5**

**3. Код программы**Реализация выполнена на Python в среде Jupyter Notebook Программный код для построения морфологического скелета и восстановления изображения приведен ниже.

import numpy as np

import cv2

from matplotlib import pyplot as plt

from scipy.ndimage import binary\_erosion

def show\_images(images, columns=2, figsize=(10, 10)):

    if isinstance(images, np.ndarray):

        images = [images]

    rows = (len(images) + columns - 1) // columns

    fig, axes = plt.subplots(rows, columns, figsize=figsize)

    axes = np.array(axes).flatten()

    for i, img in enumerate(images):

        axes[i].imshow(img, cmap='gray', interpolation='none')

        axes[i].axis('off')

    for i in range(len(images), len(axes)):

        axes[i].axis('off')

    plt.tight\_layout()

    plt.show()

image\_path = 'tree.png'

original\_image = cv2.imread(image\_path, cv2.IMREAD\_GRAYSCALE)

binary\_image = original\_image > 127

show\_images(binary\_image, columns=1, figsize=(3, 3))

def generate\_structures():

    base = np.zeros(8, dtype=bool)

    base[:4] = True

    order = [0, 1, 2, 7, 8, 3, 6, 5, 4]

    struct\_hit = np.array([

        np.concatenate((np.roll(base, shift), [False]))[order].reshape((3, 3))

        for shift in range(8)

    ])

    struct\_miss = np.zeros((3, 3), dtype=bool)

    struct\_miss[1, 1] = True

    return struct\_hit, struct\_miss

def morphological\_expansion(image, hit, miss):

    erosion\_hit = binary\_erosion(image, hit)

    erosion\_miss = binary\_erosion(~image, miss)

    return image | (erosion\_hit & erosion\_miss)

def thickening\_iteration(image, struct\_hit, struct\_miss):

    for hit in struct\_hit:

        image = morphological\_expansion(image, hit, struct\_miss)

    return image

def convexize(image):

    struct\_hit, struct\_miss = generate\_structures()

    while True:

        new\_image = thickening\_iteration(image, struct\_hit, struct\_miss)

        if np.array\_equal(new\_image, image):

            break

        image = new\_image

    return image

show\_images(convexize(binary\_image), figsize=(3,3), columns=1)

**4. Результаты**

Изображение №1

|  |  |
| --- | --- |
| Исходное изображение | Выпуклая оболочка |
|  |  |

Изображение №2

|  |  |
| --- | --- |
| Исходное изображение | Выпуклая оболочка |
|  |  |

Изображение №3

|  |  |
| --- | --- |
| Исходное изображение | Выпуклая оболочка |
|  |  |

Изображение №4

|  |  |
| --- | --- |
| Исходное изображение | Выпуклая оболочка |
|  |  |

Изображение №5

|  |  |
| --- | --- |
| Исходное изображение | Выпуклая оболочка |
|  |  |