**Лабораторная работа 2. CMake. Сборка приложения с библиотекой.**

**ЗАДАНИЕ**

**#2A**

Написать библиотеку для работы с комплексными числами. Сгенерировать решение при помощи CMake. Комплексные числа можно представить в виде , где  называется вещественной частью числа , а величина

 называется мнимой частью числа . В библиотеке требуется реализовать 4 операции:

1) Сложение двух комплексных чисел  и :



2) Вычитание двух комплексных чисел  и :



3) Умножение двух комплексных чисел  и :



4) Деление двух комплексных чисел  и :



Продемонстрировать работу библиотеки подключив ее в другом проекте(например в WPF-приложении).

Выполнение

Скачиваем CMake

Пишем код для библиотеки комплексных чисел

Листинг кода:

using System;

namespace Complex

{

    public class Complex

    {

        public Complex(double a, double b)

        {

            A = a;

            B = b;

        }

        public double A { get; set; }

        public double B { get; set; }

        public static Complex operator +(Complex first, Complex second)

        {

            return new Complex(first.A + second.A, first.B + second.B);

        }

        public static Complex operator -(Complex first, Complex second)

        {

            return new Complex(first.A - second.A, first.B - second.B);

        }

        public static Complex operator \*(Complex first, Complex second)

        {

            return new Complex((first.A \* second.A) - (second.B \* first.B),

            (first.B \* second.A) + (first.A \* second.B));

        }

        public static Complex operator /(Complex first, Complex second)

        {

            return new Complex(((first.A \* second.A) + (first.B \* second.B)) / (Math.Pow(second.A, 2) + Math.Pow(second.B, 2)), ((first.B \* second.A) - (first.A \* second.B)) / (Math.Pow(second.A, 2) + Math.Pow(second.B, 2)));

        }

        public override string ToString()

        {

            if (B >= 0) return $"{A}+{B}i";

            else return $"{A}{B}i";

        }

    }

}

Создаем текстовый файл CMakeLists.txt и помещаем туда следующий код:

cmake\_minimum\_required(VERSION 3.13 FATAL\_ERROR)

project(WOW LANGUAGES CSharp)

add\_library(Complex SHARED src/Program.cs)

открываем каталог проекта в терминале и пишем команды:

mkdir build

cd build

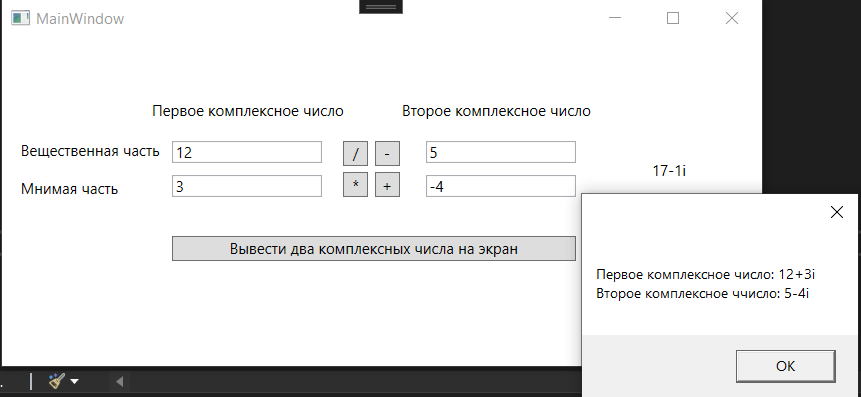
cmake ..

После этого начнется генерирование проекта и в конечном итоге получится следующая структура:

Изображение выглядит как текст

Автоматически созданное описание

WPF



Данная программа демонстрирует работоспособность библиотеки

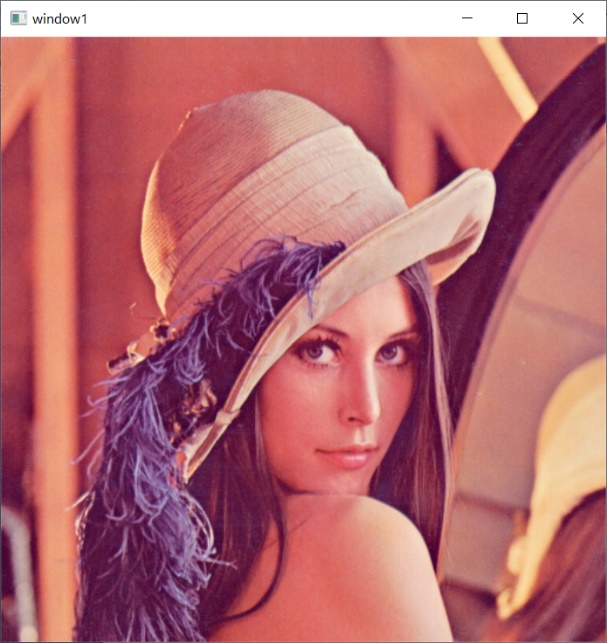
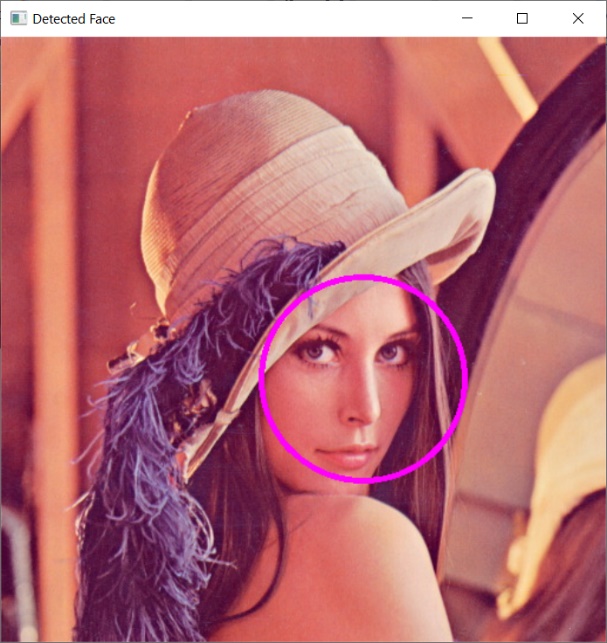
**#2B**

Собрать библиотеку OpenCV с помощью инструмента CMake. Создать проект использующий класс CascadeClassifier для определения лиц на фотографии. Использовать файл haarcascade\_frontalface\_alt2.xml. (доступен в папке install\etc\haarcascades)

Ссылки:

<https://docs.opencv.org/4.5.2/db/d28/tutorial_cascade_classifier.html>

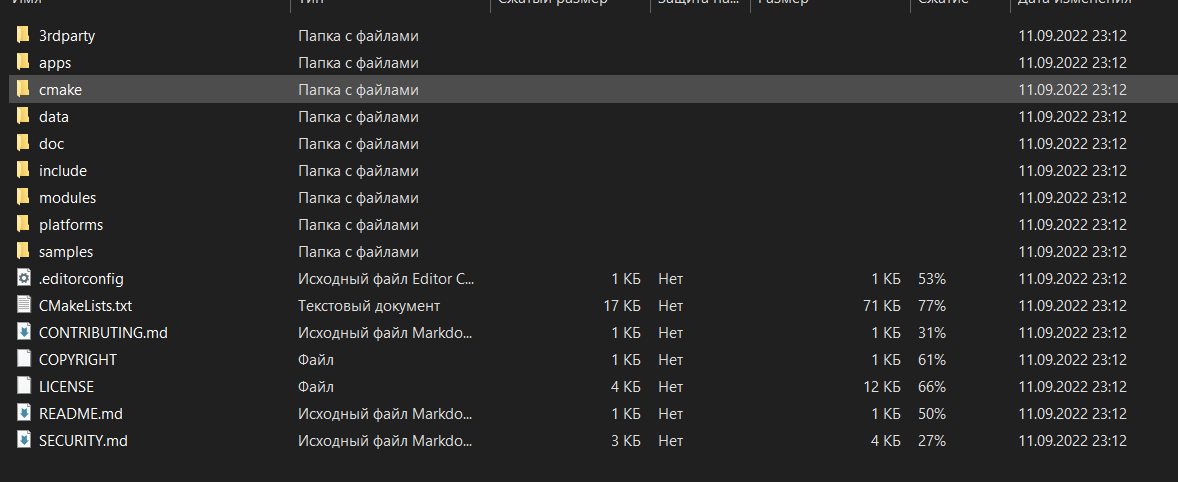
Результат (До и После):

Выполнение:

Скачиваем библиотеку OpenCV

Структура скачанной библиотеки:



Создаем папку build

Запускаем CMake

Результат:

Изображение выглядит как текст

Автоматически созданное описание

Библиотека готова к использованию

Программа компьютерного зрения по обнаружению лиц

Код программы на Python

# для видеокамеры

import cv2

face\_cascade = cv2.CascadeClassifier('haarcascade\_frontalface\_alt2.xml')

capture = cv2.VideoCapture(0)

while(True):

    ret, image = capture.read()

    gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

    faces = face\_cascade.detectMultiScale(gray, 1.1, 4)

    for (x, y, w, h) in faces:

        cv2.rectangle(image, (x, y), (x+w, y+h), (255, 0, 0), 2)

    cv2.imshow('Carmer', image)

    if cv2.waitKey (1) == ord ('q'): # нажмите Q для выхода

        break

#Для фотографии

import cv2

face\_cascade = cv2.CascadeClassifier('haarcascade\_frontalface\_alt2.xml')

image = cv2.imread('face.jpg')

gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

face = face\_cascade.detectMultiScale(gray, 1.5, 5)

for (x,y,w,h) in face:

    cv2.rectangle(image, (x,y), (x+w,y+h), (0,255,0), 3)

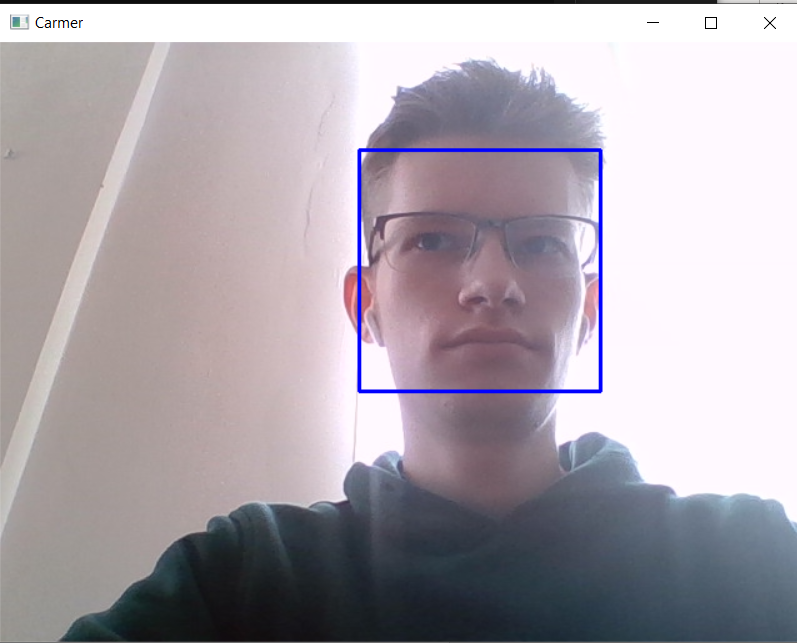
cv2.imshow("image", image)

cv2.waitKey(0)

cv2.destroyAllWindows()

Результаты:

С камеры:



С использованием фото:

Изображение выглядит как зеленый

Автоматически созданное описание