**МИНИСТЕРСТВО ОБРАЗОВАНИЯ МОСКОВСКОЙ ОБЛАСТИ**

**Государственное бюджетное профессиональное образовательное учреждение Московской области «Люберецкий техникум имени Героя Советского Союза, летчика-космонавта Ю.А.Гагарина»**

**ОТЧЕТ**

**УЧЕБНОЙ ПРАКТИКИ**

Ершова Владислава Владимировича

(Фамилия, имя, отчество студента)

по профессиональному модулю

**\_\_\_\_\_\_\_\_***ПМ.01 «Разработка модулей программного обеспечения***\_\_\_\_\_\_\_\_**

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ для компьютерных систем»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Специальность \_*09.02.07 "Информационные системы и программирование"*\_

Код, название

Курс 3 Группа № 195

Период практики с «12» ноября 2021 г. по «24» ноября 2021г.

Руководитель практики

от техникума Ревнивцев Максим Валерьевич /\_\_\_\_\_\_\_\_\_\_\_\_\_\_/  
 Жирнова Юлия Витальевна /\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

Люберцы 2021

**Оглавление**

1. [Введение 4](#_Toc88700115)
2. [Изучение предметной области 6](#_Toc88700116)
3. [Изучение программных средств 7](#_Toc88700117)
4. [OpenCV 10](#_Toc88700118)
   1. [Захват видео с Веб-Камеры и Оператор Собеля 11](#_Toc88700119)
   2. [Распознавание фигур 12](#_Toc88700120)
   3. [Распознавание лиц 15](#_Toc88700121)
   4. [Распознавание дорожной разметки с видеорегистратора 16](#_Toc88700122)
   5. [Распознавание лиц с помощью нейронной сети на Python 16](#_Toc88700123)
5. [Курс «Мини-приложения C#» 19](#_Toc88700124)
6. [WPF 20](#_Toc88700125)
7. [Выводы 28](#_Toc88700126)

**2.Дневник прохождения практики**

Студента \_\_\_\_курса

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ФИО

специальность \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Дата** | **Краткое содержание выполнения работ** | **Отметка руководителя практики**  **(оценка, подпись)** |
| 1 | 2 |  | 3 |
|  |  |  |  |

Руководитель практики

от техникума \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

подпись ФИО

# Введение

Практика как элемент учебного процесса проводится с целью *закрепления* и *расширения* знаний, полученных студентами в учебном учреждении; приобретения необходимых практических навыков работы по специальности в условиях производства; овладения передовыми методами технологии и труда. Практика способствует развитию самостоятельной работы студентов. В процессе прохождения практики студенты учатся самостоятельно отбирать и систематизировать информацию в рамках поставленных перед ними задач; применять полученные знания на практике; изучать технологию и оборудование, используемые в рамках конкретного производства; развивать навыки работы в коллективе; осуществлять самоконтроль.

Целью учебной практики является реализация компьютерного зрения с помощью библиотеки OpenCV, Создание мини-приложения с помощью прохождения курса «По использованию технологии Windows Forms Net.Framework на языке C#» и подключение базы данных к C# WPF приложению с помощью SQL Server Management Studio.

Для достижения данной цели требуется решить следующие задачи:

1. Знакомство с литературой
2. Пройти курс «Мини-приложения C#»
3. Изучить библиотеку OpenCV
4. Реализация приложения на WPF и подключение базы данных

В ходе учебной практики планируется освоение следующего вида деятельности: «Разработка модулей программного обеспечения для компьютерных систем » (ПК):

ПК 1.1 Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.

ПК 1.2 Разрабатывать программные модули в соответствии с техническим заданием.

ПК 1.3 Выполнять отладку программных модулей с использованием специализированных программных средств

ПК 1.4 Выполнять тестирование программных модулей

ПК 1.5 Осуществлять рефакторинг и оптимизацию программного кода

ПК 1.6 Разрабатывать модули программного обеспечения для мобильных платформ.

ПК 2.1. Разрабатывать требования к программным модулям на основе анализа проектной и технической документации на предмет взаимодействия компонент.

ПК 2.2. Выполнять интеграцию модулей в программное обеспечение.

ПК 2.3. Выполнять отладку программного модуля с использованием специализированных программных средств.

ПК 2.4. Осуществлять разработку тестовых наборов и тестовых сценариев для программного обеспечения.

ПК 2.5. Производить инспектирование компонент программного обеспечения на предмет соответствия стандартам кодирования.

# Изучение предметной области

* **Предметная область** – множество всех предметов, свойства которых и отношения, между которыми рассматриваются в научной теории.
* **OpenCV** – это набор типов данных, функций и классов для обработки изображений алгоритмами компьютерного зрения.

В версии 2.2 структура библиотека реорганизована — теперь вместо больших универсальных модулей (cxcore, Cvaux, Highgui, Cvaux) библиотека OpenCV разделена на небольшие модули по функциональному использованию.

* **Windows Presentation Foundation** (WPF) — это платформа пользовательского интерфейса для создания клиентских приложений для настольных систем. Платформа разработки WPF поддерживает широкий набор компонентов для разработки приложений, включая модель приложения, ресурсы, элементы управления, графику, макет, привязки данных, документы и безопасность.
* Был пройден курс «**Мини-Приложения C#**», где было рассказано, как разработать утилиты, которые послужат основой дальнейшего обучения искусству программирования.

# Изучение программных средств

Использовавшиеся программные средства:

* **IDE Visual Studio Community Edition**

Достоинства:

*Встроенный Web-сервер.* Для обслуживания Web-приложения ASP.NET необходим Web-сервер, который будет ожидать Web-запросы и обрабатывать соответствующие страницы. Наличие в Visual Studio интегрированного Web-сервера позволяет запускать Web-сайт прямо из среды проектирования, а также повышает безопасность, исключая вероятность получения доступа к тестовому Web-сайту с какого-нибудь внешнего компьютера, поскольку тестовый сервер может принимать соединения только с локального компьютера.

*Поддержка множества языков при разработке*. Visual Studio позволяет писать код на своем языке или любых других предпочитаемых языках, используя все время один и тот же интерфейс (IDE).

*Меньше кода для написания*. Для создания большинства приложений требуется приличное количество стандартного стереотипного кода. Например, добавление Web-элемента управления, присоединение обработчиков событий и корректировка форматирования требует установки в разметке страницы ряда деталей. В Visual Studio такие детали устанавливаются автоматически.

*Интуитивный стиль кодирования*. По умолчанию Visual Studio форматирует код по мере его ввода, автоматически вставляя необходимые отступы и применяя цветовое кодирование для выделения элементов типа комментариев. Такие незначительные отличия делают код более удобным для чтения и менее подверженным ошибкам.

*Высокая скорость разработки*. Многие из функциональных возможностей Visual Studio направлены на то, чтобы помогать разработчику делать свою работу как можно быстрее. Удобные функции, вроде функции IntelliSense (которая умеет перехватывать ошибки и предлагать правильные варианты), функции поиска и замены (которая позволяет отыскивать ключевые слова как в одном файле, так и во всем проекте) и функции автоматического добавления и удаления комментариев (которая может временно скрывать блоки кода), позволяют разработчику работать быстро и эффективно.

*Возможности отладки*. Предлагаемые в Visual Studio инструменты отладки являются наилучшим средством для отслеживания загадочных ошибок и диагностирования странного поведения. Разработчик может выполнять свой код по строке за раз, устанавливать интеллектуальные точки прерывания, при желании сохраняя их для использования в будущем, и в любое время просматривать текущую информацию из памяти.

Visual Studio также имеет и множество других функций: возможность управления проектом; встроенная функция управления исходным кодом; возможность рефакторинга кода; мощная модель расширяемости.

* **PyCharm**

Достоинства:

Имеет удобный редактор кода со всеми полезными функциями: *подсветкой синтаксиса, автоматическим форматированием, дополнением и отступами*. PyCharm позволяет проверять версии интерпретатора языка на совместимость, а также использовать шаблоны кода.

Тем, кто часто использует документацию, будет удобно смотреть ее прямо в окне редактора (для элементов) либо в браузере (для внешней документации).

PyCharm позволяет быстро производить *рефакторинг кода*, а также использовать удобный графический *отладчик*.

Утилита поддерживает все свежие версии Django, а также IronPython, Jython, Cython, PyPy wxPython, PyQt, PyGTK и многие другие инструменты.

В PyCharm можно проводить *интегрированное* Unit тестирование, использовать интерактивные консоли для Python, Django, SSH, отладчика и баз данных.

PyCharm имеет большую коллекцию плагинов, и его можно использовать в связке с разными трекерами вроде JIRA, Youtrack, Lighthouse, Redmine, Trac и так далее.

PyCharm *кроссплатформенная* среда разработки: можно использовать на Linux, Windows и Mac OS.

* **Microsoft SQL Server Management Studio**

Достоинства:

СУБД *масштабируется*, поэтому работать с ней можно на портативных ПК или мощной мультипроцессорной технике. Процессор может одновременно обрабатывать большой объем запросов.

*Размер страниц* – до 8 кб, поэтому данные извлекаются быстро, подробную и сложную информацию хранить удобнее. Система позволяет обрабатывать транзакции в интерактивном режиме, есть динамическая блокировка.

Рутинные административные *задачи* *автоматизированы*: это управление блокировками, памятью, редактура размеров файлов. У системы продуманы настройки, можно создать профили пользователей.

Реализован *поиск* по фразам, тексту, словам, можно создавать ключевые индексы.

В SQL Server есть *репликации* через интернет, предусмотрена *синхронизация*. Есть полноценный веб-ассистент для форматирования страниц.

В систему интегрирован сервер интерактивного анализа для принятия решений, создания корпоративных отчетов. Есть службы преобразования информации.

Запросы можно формулировать на английском языке, без программирования.

СУБД поддерживает работу с другими продуктами Microsoft: Access, MS Excel.

# OpenCV

1. **OpenCV** (Open Source Computer Vision) библиотека компьютерного зрения с открытым исходным кодом) — библиотека алгоритмов компьютерного зрения, обработки изображений и численных алгоритмов общего назначения с открытым кодом. Реализована на C/C++, также разрабатывается для *Python, Java, Ruby, Matlab, Lua* и других языков. Может свободно использоваться в академических и коммерческих целях — распространяется в условиях лицензии BSD.

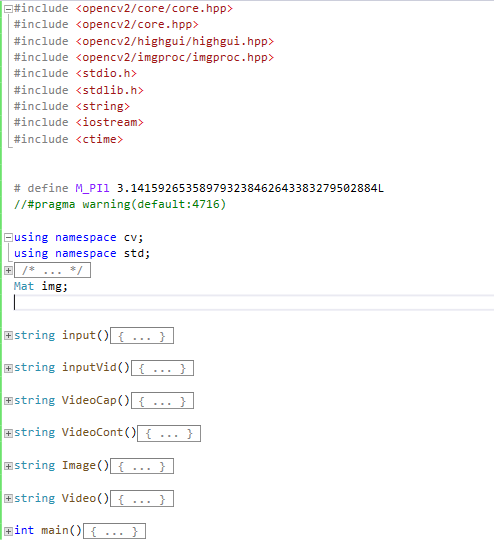
Но наиболее часто она используется для написания приложений машинного обучения на языке Python, особенно в сфере компьютерного зрения.

Для работы c OpenCV был создан отдельный репозиторий на GitHub: educational-practice. Весь код и результаты расположены в папке OpenCV - <https://github.com/vlayerrt/educational-practice>

1. Файл **Source.cpp** - код программы, где есть:

Функция **VideoCap()** - оператор Собеля с захватом веб-камеры

Функция **VideoCont()** - нахождение центров фигур на видео



## Захват видео с Веб-Камеры и Оператор Собеля

Для запуска программы необходимо выполнить функцию VideoCap().

С помощью OpenCV был произведен захват видео с Веб-Камеры, видео с которой впоследствии обработано **оператором Собеля.**

**Оператор Собеля** — дискретный дифференциальный оператор, вычисляющий приближённое значение градиента яркости изображения. Результатом применения оператора Собеля в каждой точке изображения является либо вектор градиента яркости в этой точке, либо его норма. Используется в области обработки изображений, в частности, часто применяется в алгоритмах выделения границ.

VideoCapture cap – оператор для захвата веб-камеры

detectMultiScale — общая функция для распознавания как лиц, так и объектов. Чтобы функция искала именно лица, мы передаём ей соответствующий каскад.

Функция detectMultiScale принимает 4 параметра

Обрабатываемое изображение в градации серого.

**Код** находится в файле educational-practice/opencv/Source.cpp (в классе **VideoCap()** ) на репозитории, **результат** прикреплен скриншотом в той же папке.

## Распознавание фигур

Для запуска программы необходимо выполнить функцию **VideoCont().**

Для распознавания фигур были использованы функции findContours и drawContours:

**findContours** (кадр, режим\_группировки, метод\_упаковки [, контуры[, иерархия[, сдвиг]]])

***кадр*** — должным образом подготовленная для анализа картинка. Это должно быть 8-битное изображение. Поиск контуров использует для работы монохромное изображение, так что все пиксели картинки с ненулевым цветом будут интерпретироваться как 1, а все нулевые останутся нулями. На уроке про поиск цветных объектов была точно такая же ситуация.

***режим\_группировки*** — один из четырех режимов группировки найденных контуров:

CV\_RETR\_LIST — выдаёт все контуры без группировки;

CV\_RETR\_EXTERNAL — выдаёт только крайние внешние контуры.

CV\_RETR\_CCOMP — группирует контуры в двухуровневую иерархию. На верхнем уровне — внешние контуры объекта. На втором уровне — контуры отверстий, если таковые имеются. Все остальные контуры попадают на верхний уровень.

CV\_RETR\_TREE — группирует контуры в многоуровневую иерархию.

***метод\_упаковки*** — один из трёх методов упаковки контуров:

CV\_CHAIN\_APPROX\_NONE — упаковка отсутствует и все контуры хранятся в виде отрезков, состоящих из двух пикселей.

CV\_CHAIN\_APPROX\_SIMPLE — склеивает все горизонтальные, вертикальные и диагональные контуры.

CV\_CHAIN\_APPROX\_TC89\_L1,CV\_CHAIN\_APPROX\_TC89\_KCOS — применяет к контурам метод упаковки (аппроксимации) Teh-Chin.

контуры — список всех найденных контуров, представленных в виде векторов;

***иерархия*** — информация о топологии контуров. Каждый элемент иерархии представляет собой сборку из четырех индексов, которая соответствует контуру[i]:

* иерархия[i][0] — индекс следующего контура на текущем слое;
* иерархия[i][1] — индекс предыдущего контура на текущем слое:
* иерархия[i][2] — индекс первого контура на вложенном слое;
* иерархия[i][3] — индекс родительского контура.

***сдвиг*** — величина смещения точек контура.

Иерархия, возвращаемая findContours, имеет следующий вид: hierarchy[idx][{0,1,2,3}]={next contour (same level), previous contour (same level), child contour, parent contour}

Вы в основном проходите через hierarchy[idx] и рисуете что-либо с hierarchy[idx][3]>-1.

Полученные с помощью функции *findContours* контуры хорошо бы каким-то образом нарисовать в кадре. Машине это не нужно, зато нам это поможет лучше понять, как выглядят найденные алгоритмом контуры. Для этого используетяс следующая функция:

**drawContours**( кадр, контуры, индекс, цвет[, толщина[, тип\_линии[, иерархия[, макс\_слой[, сдвиг]]]]])

***кадр*** — кадр, поверх которого мы будем отрисовывать контуры;

***контуры*** — те самые контуры, найденные функцией findContours;

***индекс*** — индекс контура, который следует отобразить.

-1 — если нужно отобразить все контуры;

***цвет*** — цвет контура;

***толщина*** — толщина линии контура;

***тип\_линии*** — тип соединения точек вектора;

***иерархия*** — информация об иерархии контуров;

***макс\_слой*** — индекс слоя, который следует отображать.

Если параметр равен 0, то будет отображен только выбранный контур. Если параметр равен 1, то отобразится выбранный контур и все его дочерние контуры.

Если параметр равен 2, то отобразится выбранный контур, все его дочерние и дочерние дочерних! И так далее.

***сдвиг*** — величина смещения точек контура.

**Оператор Canny**

Получил свое название в честь математика, разработавшего фильтр.

Состоит из этапов:

* Преобразование в оттенки серого - rgb2gray
* Сглаживание blur()
* Гаусс: отсекает Передали градиент - направление. рисует вместо каждого пикселя стрелочку

Они считаются контуром, если имеют одно направление и максимальную яркость - подавление

* vector - тип данных который позволяет хранить координаты (x и y начало контура | x и y конца контура) Есть вектор момента. Его использование позволяет определить нулевой(m00) момент (кол-во точек в контуре). Первый момент(m10)
* **Класс Mat** библиотеки OpenCV используется для хранения значений изображения. Он представляет собой n-мерный массив и используется для хранения данных изображения в градациях серого или цветных изображений, объемов вокселей, векторных полей, облаков точек
* **Dilate** растягивает (операция расширения) изображение с использованием фильтра(ядра) один или несколько раз, если element == NULL используется ядро 3х3 (изображение формируется из локальных максимумов — т.е. будут увеличиваться светлые области)
  + *src* — исходное изображение
  + *dst* — получаемое изображение
  + *element* — структурирующий элемент (ядро) по умолчанию NULL — соответствует ядру 3x3 с якорем по центру.

Программа определяет на картинке следующие фигуры:

* Треугольник,
* квадрат/прямоугольник,
* круг.
* Определяет центр масс.

## Распознавание лиц

Для распознавания лиц была написана отдельная программа, которая находится в папке educational-practice/opencv/python/

*(*[*https://github.com/vlayerrt/educational-practice/tree/main/opencv/python*](https://github.com/vlayerrt/educational-practice/tree/main/opencv/python)*)*

Файл **main.py** – код программы для распознавания лиц с картинки

Файл **faces.py -** код программы для распознавания лиц с веб-камеры

* Параметр scaleFactor. Некоторые лица могут быть больше других, поскольку находятся ближе, чем остальные. Этот параметр компенсирует перспективу.
* Алгоритм распознавания использует скользящее окно во время распознавания объектов. Параметр minNeighbors определяет количество объектов вокруг лица. То есть чем больше значение этого параметра, тем больше аналогичных объектов необходимо алгоритму, чтобы он определил текущий объект, как лицо. Слишком маленькое значение увеличит количество ложных срабатываний, а слишком большое сделает алгоритм более требовательным.

minSize — непосредственно размер этих областей.

* Сохранение изображения

import cv2

image = cv2.imread("./импорт/путь.расширение")

cv2.imwrite("./экспорт/путь.расширение", image)

Рисование прямоугольников

import cv2

output = image.copy()

cv2.rectangle(output, (2600, 800), (4100, 2400), (0, 255, 255), 10)

viewImage(output, "Обводим прямоугольником лицо пёсика")

Эта функция принимает 5 параметров:

* Само изображение.
* Координата верхнего левого угла (x1, y1).
* Координата нижнего правого угла (x2, y2).
* Цвет прямоугольника (GBR/RGB в зависимости от выбранной цветовой модели).
* Толщина линии прямоугольника.

## Распознавание дорожной разметки с видеорегистратора

Для распознавания дорожной разметки с видеорегистратора была написана отдельная программа, которая находится в папке educational-practice/opencv/road/

*(*[*https://github.com/vlayerrt/educational-practice/tree/main/opencv/road*](https://github.com/vlayerrt/educational-practice/tree/main/opencv/road)*)*

Чтобы распознать полосы разметки необходимо:

1. Преобразовать исходное изображение в grayscale.
2. Затемнить полученное изображение (это помогает уменьшить контраст от обесцвеченных участков дороги).
3. Преобразовать исходное изображение в цветовое пространство HLS (Hue, Lightness, Saturation — тон, свет, насыщенность).
4. Изолировать жёлтый цвет из HLS для получения маски (для жёлтой разметки).
5. Изолировать белый цвет из HLS (для белой разметки).
6. Выполнить побитовое «ИЛИ» жёлтой и белой масок для получения общей маски.
7. Выполнить побитовое «И» маски и затемнённого изображения.
8. Применить Гауссово размытие.
9. Применить детектор границ Canny (пороги устанавливаются методом проб и ошибок).
10. Определить область интереса (помогает отсеять нежелательные края, обнаруженные детектором Canny).
11. Получить линии Хафа.
12. Объединить и экстраполировать линии Хафа; отобразить их на исходном изображении.

Сначала на вход функции подается 3-канальное изображение формата RGB, которое затем фильтруется, преобразовывается, а внутри функции обновляются объекты Line и Lane. Затем поверх самого изображения рисуются все необходимые элементы.

## Распознавание лиц с помощью нейронной сети на Python

Для распознавания лиц на видео с использованием нейронной сети из библиотеки openCV была написана отдельная программа, которая находится в папке educational-practice/opencv/neural\_network/

*(*[*https://github.com/vlayerrt/educational-practice/tree/main/opencv/neural\_network*](https://github.com/vlayerrt/educational-practice/tree/main/opencv/neural_network)*)*

Ссылка на программу: [https://colab.research.google.com/drive/1Uqze8waoZ49CyxUlgI5Hw8EKe0na5Y2T](https://colab.research.google.com/drive/1Uqze8waoZ49CyxUlgI5Hw8EKe0na5Y2T?usp=sharing)

В качестве примера работы программы использовано видео с интервью Team Spirit после победы на The International 2021 по Dota 2.

В видео всего 9 участников и я сделал шаблоны только для 6 из них, чтобы показать, как именно работает программа.

Суть в том, что нейронная сеть ищет совпадения во всем видео только по подготовленным шаблонам, называя «квадраты» для лиц эквивалентно названию файла-шаблона.

Все файлы шаблоны находятся в папке “photos”. Они делаются заранее из исходного видео, чтобы программа искала совпадения.

Файл **main.py –** код программы

Исходный файл с видео находится по ссылке: <https://goo.su/9XOM>

Файл **output.mp4 –** результат работы программы.

Весь результат работы программы также прикреплен скриншотами к readme.md файлу в репозитории по ссылке выше.

**Введение:**

За последние годы компьютерное зрение набрало популярность и выделилось в отдельное направление. Разработчики создают новые приложения, которыми пользуются по всему миру.

В этом направлении меня привлекает концепция открытого исходного кода. Даже технологические гиганты готовы делиться новыми открытиями и инновациями со всеми, чтобы технологии не оставались привилегией богатых.

Одна из таких технологий — **распознавание лиц**. При правильном и этичном использовании эта технология может применяться во многих сферах жизни.  
Примеры использования:

* Распознавание лиц в соцсетях
* Распознавание лиц в сфере безопасности
* Распознавание лиц для подсчета количества людей

Для написания программы нам необходимо:

1. Подключить библиотеку OpenCV к проекту на Python

Для этого используется команда:

*pip3 install opencv-python*

1. Установить face\_recognition API

Для этого используются команды:

*pip install dlib*

*pip install face\_recognition*

Далее пишется сам код, в котором мы загружаем исходное видео, обрабатываем его, ищем совпадения и экспортируем видео, в котором будут распознанные лица.

Программа работает по принципу поиска совпадений готовых шаблонов с лицами на видео.

Мы сделали захват видео в реальном времени кадр за кадром.

Затем обработали каждый кадр и извлекли местонахождение всех лиц на изображении.

В итоге, воспроизвели эти кадры в форме видео вместе с указанием на то, где расположены лица.

Скриншоты выполнения всех программ расположены по ссылкам:

[educational-practice/opencv](https://github.com/vlayerrt/educational-practice/tree/main/opencv) – Оператор Собеля, захват видео с виртуальной веб-камеры и фигуры

[educational-practice/opencv/python](https://github.com/vlayerrt/educational-practice/tree/main/opencv/python) - Распознавание лиц на языке Python с помощью OpenCV

[educational-practice/opencv/road](https://github.com/vlayerrt/educational-practice/tree/main/opencv/road) - Разметка дороги

[educational-practice/opencv/neural\_network/](https://github.com/vlayerrt/educational-practice/tree/main/opencv/neural_network) - Распознавание лиц с помощью нейронной сети OpenCV на Python

# Курс «Мини-приложения C#»

Презентация выложена в папке educational-practice/course по [ссылке](https://github.com/vlayerrt/educational-practice/blob/main/course/%D0%9C%D0%B8%D0%BD%D0%B8%20%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F%20C%23%20l77.pptx).

[*(https://github.com/vlayerrt/educational-practice/tree/main/course)*]((https:/github.com/vlayerrt/educational-practice/tree/main/course))

* Сначала установлена рабочая среда IDE Visual Studio 2019, после чего создан WPF Проект (см. слайд 3).
* Подготовлена рабочая среда: настроена панель элементов, создана форма, подготовлен обозреватель решений. (см. слайд 4, 5, 6)
* Далее созданы и настроены элементы формы, добавлены события по нажатию (см слайд 7, 8, 9, 10)
* Создание счетчика: добавлены кнопки и текст, после чего прописан код к событиям. (см слайд 11, 12, 13, 14, 15, 16).
* Создание генератора случайных чисел: с помощью панели элементов созданы NumericUdDown, label и кнопка, где NumericUdDown задает диапазон генерации, кнопка генерирует число. Далее прописан код с использованием переменной класса Random. Следующим шагом мы создали обработчик событий на созданную кнопку, где добавили новую переменную и присвоили ей случайное целое число с заданным диапазоном, в который входят значения numericUpDown1, заданные пользователем. К последнему значению добавили +1, чтобы крайнее значение тоже входило в диапазон. И вывели результат в наш label (см слайд 17, 18, 19, 20, 21).
* Создание списка сгенерированных чисел: добавлен textbox и создан обработчик событий для кнопки. Чтобы очистить список, создана кнопка и событие к ней . Чтобы скопировать результат, также понадобится кнопка.

Далее добавлена часть кода, отвечающая за отсутствие повторения результатов в списке сгенерированных чисел, а также возможность ее включения/отключения.(см слайд 22, 23, 24, 25, 26, 27, 28, 29, 30, 31).

* Создание блокнота: в форме делается еще одна вкладка, и добавляется новый элемент – RichTextBox. Далее добавляется возможность вставить дату и время, для этого создается новый элемент меню и обработчик событий к нему. Для этих функций можно добавить горячий клавиши.

Созданы функции сохранения и загрузки текстового документа. Написаны исключения для обработки возможных ошибок (отсутствие файла) (см слайд 32, 33, 34, 35, 36, 37, 38, 39, 40, 41).

* Создание генератора паролей: добавлена новая вкладка в форме и создан элемент CheckBoxList для задания критерий генерации пароля. В коллекции прописаны сами критерии, которые пользователь может выбрать. Добавлены также: кнопка генерации, текст, возможность задать длину пароля и поле с созданным паролем. Прописан код для обработчика событий для кнопки, где будет написана сама программа генератора паролей. Для каждого критерия прописан код.

Заключение:

* Созданы программы под Windows;
* Ознакомлен с наиболее часто используемыми элементами Windows Forms;
* Изучены примеры использования классов Net.Framework;
* Изучено, что представляет из себя программа средней сложности;

# WPF

* Для прохождения практики установлен **SQL Server Management Studio** и **IDE Visual Studio 2019 Community Edition**
* Создан проект WPF .NET Framework

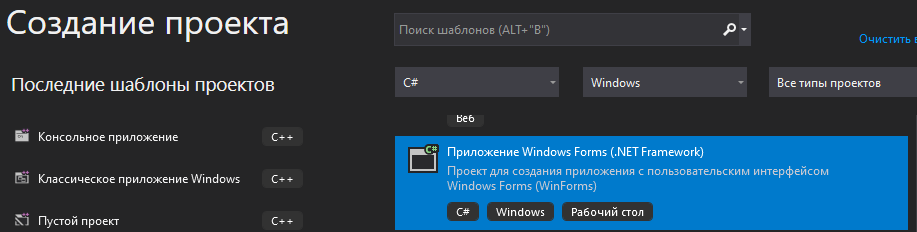


Рисунок 1. Создание проекта WPF

* Написано приложение с окном Регистрации

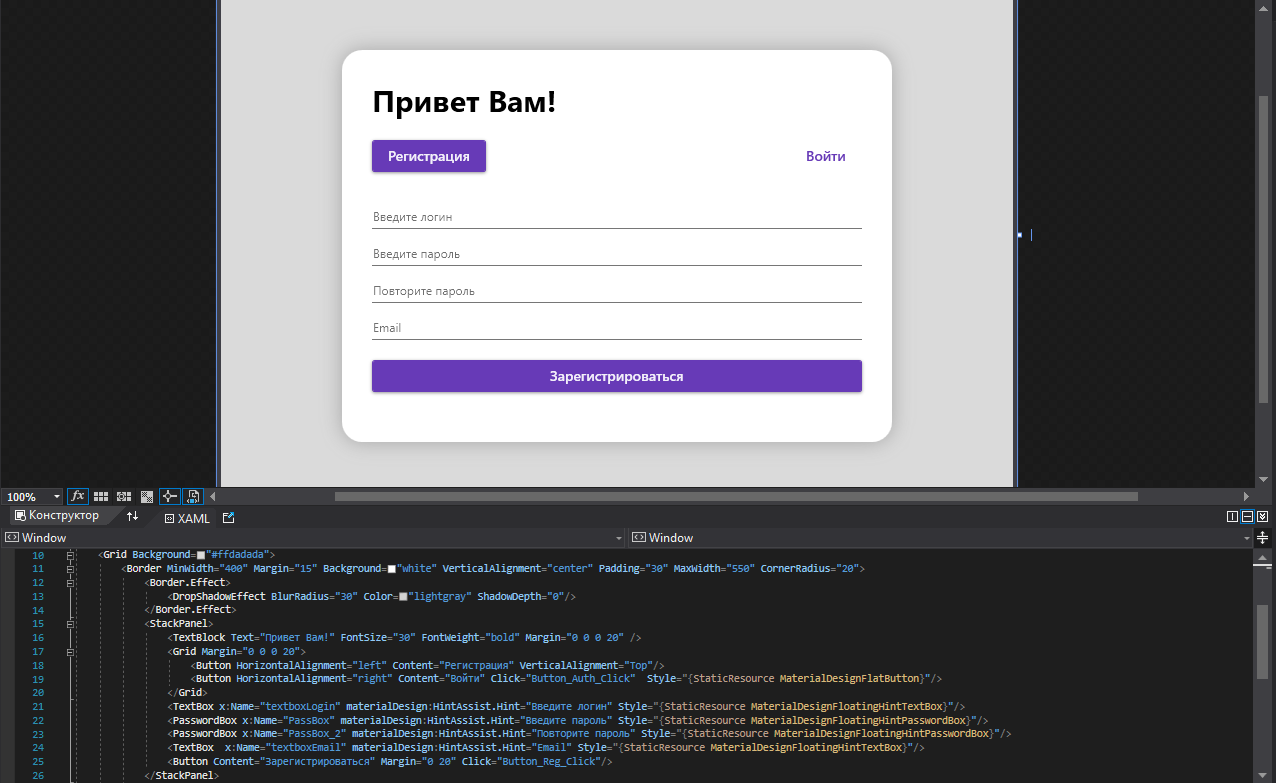


Рисунок 2. Окно с регистрацией

* Создание базы данных на локальном сервере

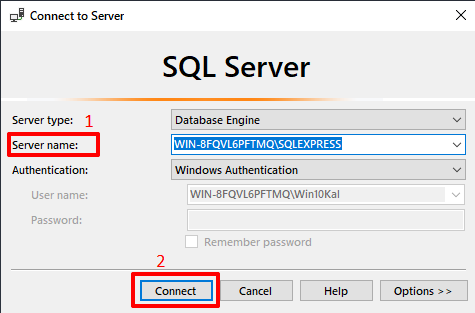


Рисунок 3. Подключение к серверу

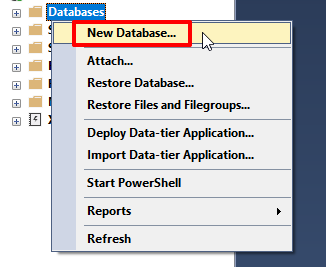


Рисунок 4. Создание базы данных

* Создание таблицы Users
* База данных была подключена к приложению с помощью добавления класса **модели данных ADO.NET Entity**

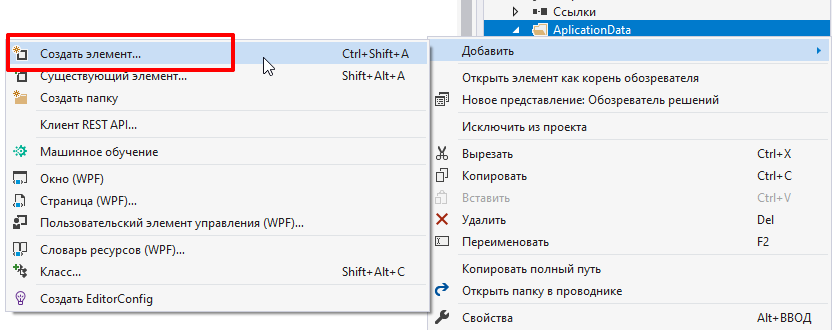


Рисунок 5. Подключение бд к WPF приложению

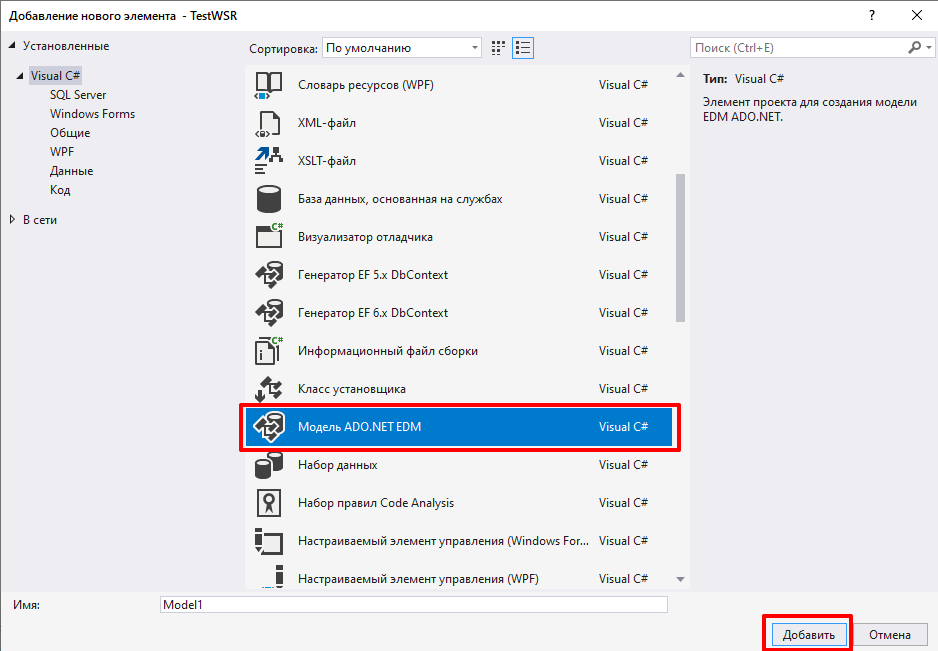


Рисунок 6. Выбор модели

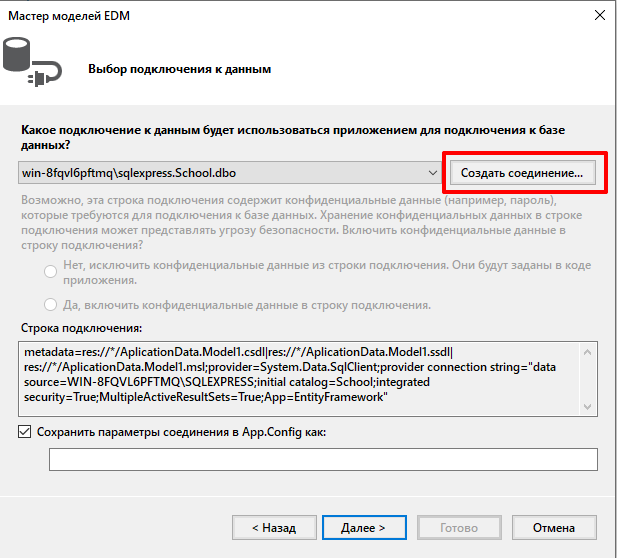


Рисунок 7. Создание подключения

Выбраны таблицы и добавлены в проект с названием пространства имен модели SFEntities.

* Написан код в файле Model1.Context.cs, записывающий результаты в таблицу базы данных.

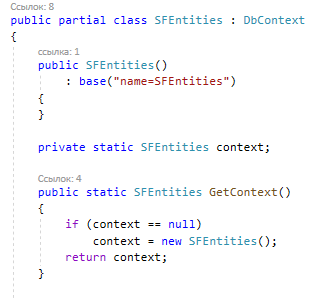


Рисунок 8. Код GetContext()



Рисунок 9. Код для сохранения в таблицу

* Прописаны исключения

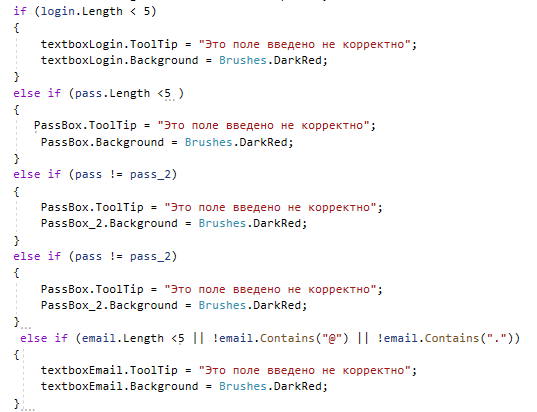


Рисунок 10. Создание условий

* Создано окно с авторизацией

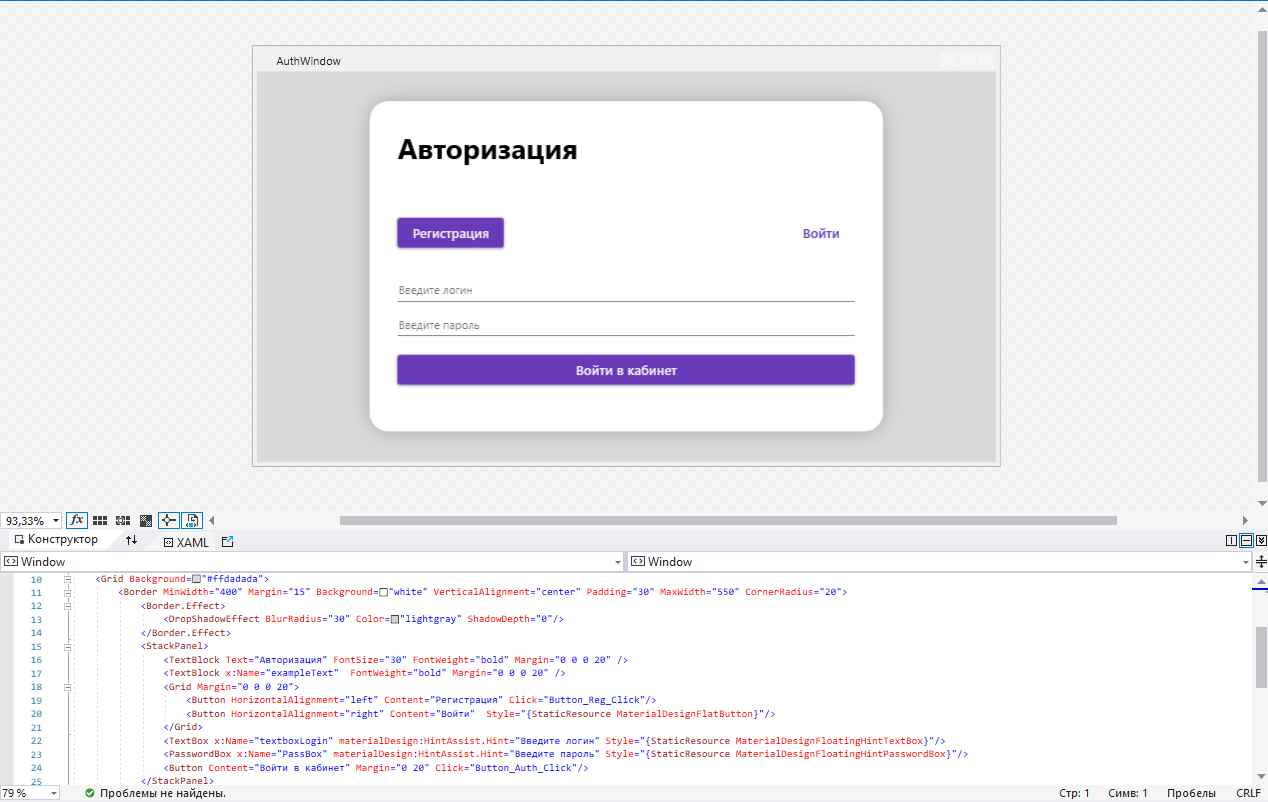


Рисунок 11. Окно авторизации пользователя

* Написан обработчик событий для проверки данных при авторизации.



Рисунок 12. Обработчик событий на кнопку авторизации

* Создано окно с Личным Кабинетом, где написаны все логины и пароли из базы данных

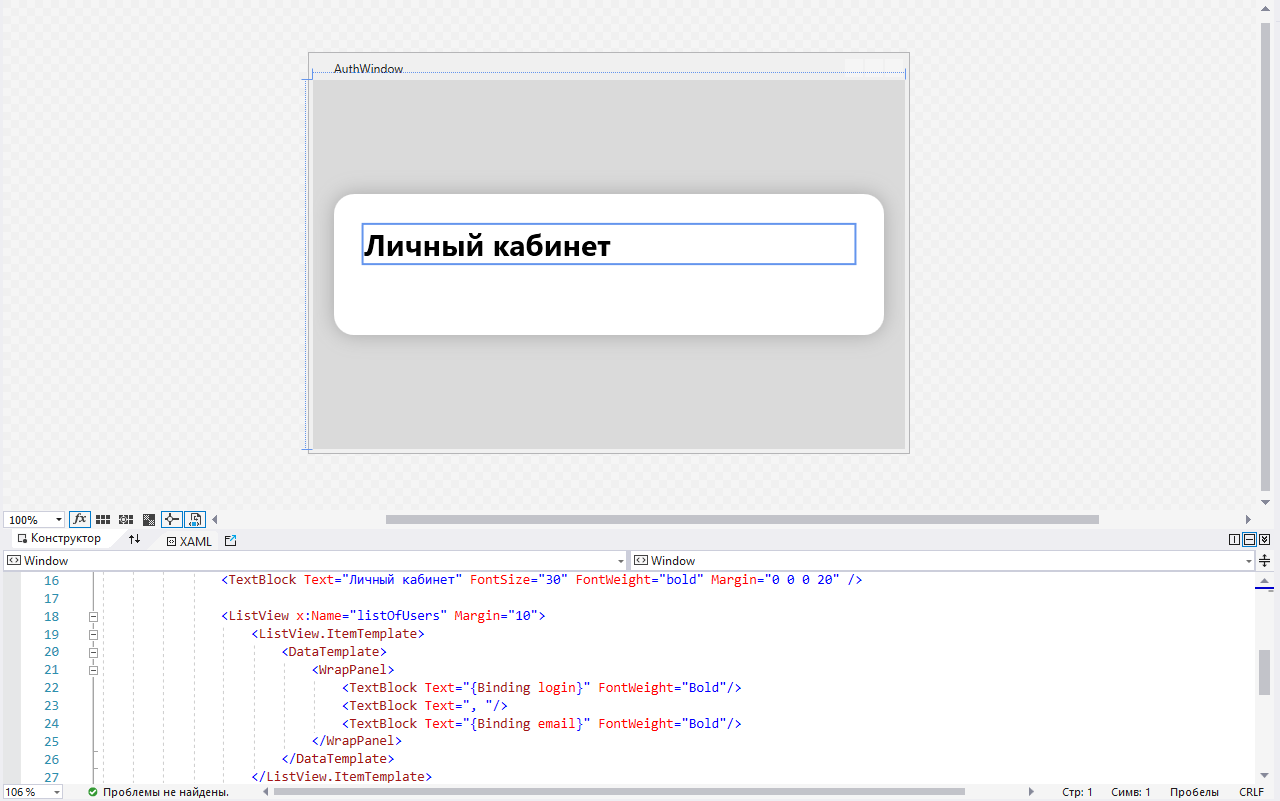


Рисунок 13. Окно «Личный кабинет»

* Добавлен шаблон для TextBox, чтобы результаты выводились в созданное окно

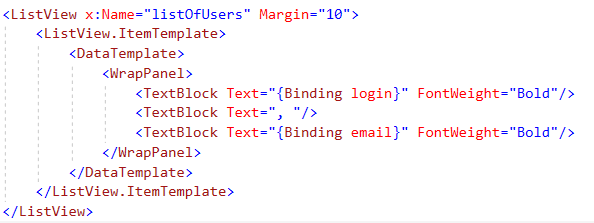


Рисунок 14. Код шаблона

* Результат приведен ниже

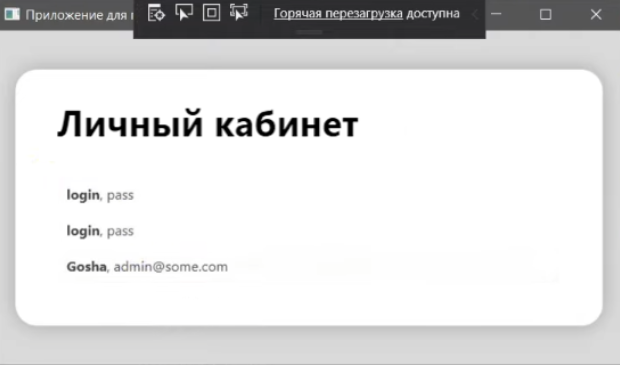


Рисунок 15. Окно с данными из бд

# Выводы

По окончании учебной практики были решены следующие задачи:

* Изучена предметную область.
* Освоен курс “C# Мини-приложения”.
* Освоено проектирование и создание баз данных.
* Изучено создание запросов для взаимодействия с БД.
* Разработаны приложение на платформе WPF.
* Знакомство с литературой.
* Изучена библиотека OpenCV.

В ходе прохождения учебной практики были сформированы следующие профессиональные компетенции:

ПК 1.1 Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.

ПК 1.2 Разрабатывать программные модули в соответствии с техническим заданием.

ПК 1.3 Выполнять отладку программных модулей с использованием специализированных программных средств

ПК 1.4 Выполнять тестирование программных модулей

ПК 1.5 Осуществлять рефакторинг и оптимизацию программного кода

ПК 1.6 Разрабатывать модули программного обеспечения для мобильных платформ.

ПК 2.1. Разрабатывать требования к программным модулям на основе анализа проектной и технической документации на предмет взаимодействия компонент.

ПК 2.2. Выполнять интеграцию модулей в программное обеспечение.

ПК 2.3. Выполнять отладку программного модуля с использованием специализированных программных средств.

ПК 2.4. Осуществлять разработку тестовых наборов и тестовых сценариев для программного обеспечения.

ПК 2.5. Производить инспектирование компонент программного обеспечения на предмет соответствия стандартам кодирования.

1. **Список литературы**
2. Орлов С.А. Программная инженерия. Технологии разработки программного обеспечения: учебник / С. А. Орлов. – 5-е изд., обновл. и доп. – Москва; Санкт- Петербург:Питер, 2016. – 640 с.
3. Джепикс Филипп, Троелсен Эндрю. Язык программирования C# 7 и платформы .NET и .NET Core / Джепикс Филипп, Троелсен Эндрю – М.; Вильямс, 2018. – 1328. «Наука», 2017 – 87с.
4. Павловская Т.А. С/С++. Программирование на языке высокого уровня. СПб.: Питер, 2017. – 461 с.
5. Семакин И. Г. Основы алгоритмизации и программирования: учебник для студ. учреждений сред. проф. образования / И. Г. Семакин, А. П. Шестаков. — Издательский центр «Академия», 2017. — 304 с
6. Хабибулин И.Ш. Программирование на языке высокого уровня. C/C++. – СПб.: БХВ– Петербург, 2017. – 512 с.
7. Прайс Марк. C# 7 и .NET Core. Кроссплатформенная разработка для профессионалов / Прайс Марк – Москва; Вильямс, 2018. – 604.
8. Скит Джон. C# для профессионалов. Тонкости программирования / Скит Джон – Москва; Вильямс, 2019. – 608.
9. Мюллер Д. П. C# для чайников / Мюллер Д. П. – Москва; Москва, 2019, – 608 с.
10. https://sql-academy.org/ru
11. https://metanit.com/sharp/
12. *Кэлер А., Брэдски Г.* Изучаем OpenCV 3 = Learning OpenCV 3. — М.: ДМК-Пресс, 2017. — 826 с. — [ISBN 978-5-97060-471-7](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/9785970604717).
13. *Буэно, Суарес, Эспиноса.* Обработка изображений с помощью OpenCV = Learning Image Processing with OpenCV. — М.: ДМК-Пресс, 2016. — 210 с. — [ISBN 978-5-97060-387-1](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/9785970603871).
14. *Прохоренок Н.* OpenCV и Java. Обработка изображений и компьютерное зрение. — СПб.: [БХВ-Петербург](https://ru.wikipedia.org/w/index.php?title=%D0%91%D0%A5%D0%92-%D0%9F%D0%B5%D1%82%D0%B5%D1%80%D0%B1%D1%83%D1%80%D0%B3&action=edit&redlink=1), 2018. — 320 с. — [ISBN 978-5-9775-3955-5](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/9785977539555).
15. SQL [Электронный ресурс]: база данных. <https://sql-academy.org/ru>
16. <https://metanit.com/sharp/>
17. OpenCV шаг за шагом - <https://robocraft.ru/blog/computervision/319.html>
18. OpenCV [Электронный ресурс]: база данных. – <https://opencv.org/>