Министерство образования и молодежной политики

Свердловской области

Государственное автономное профессиональное образовательное

учреждение Свердловской области «Ирбитский политехникум»

КУРСОВАЯ РАБОТА

по ПМ.01 Разработка программных модулей программного обеспечения для

компьютерных систем

МДК 01.01. Системное программирование

Тема:Термометрия, оптическая передача данных с термометра на языке Python

Выполнил:

студент группы ПКС-306

специальность 09.02.03

Программирование

в компьютерных системах

Останин Иван Васильевич

Проверил**:** преподаватель

Вишнякова Н.В.

Ирбит 2022СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc94878471)

[1. ОСОБЕННОСТИ ЯЗЫКА ПРОГРАММИРОВАНИЯ И РАЗРАБОТКА МОДЕЛИ ПРОГРАММЫ. 4](#_Toc94878472)

[1.1. Особенности языка программирования Python. 4](#_Toc94878473)

[1.2. Используемые библиотеки и инструменты. 4](#_Toc94878474)

[1.3. Принципы формирования изображения на термометре. 5](#_Toc94878475)

[1.4. Интерфейс программы. 5](#_Toc94878476)

[2. РАЗРАБОТКА И ТЕСТИРОВАНИЕ ПРОГРАММЫ. 9](#_Toc94878477)

[2.1. Разработка структуры программы. 9](#_Toc94878478)

[2.1.1 Общая структура программы. 9](#_Toc94878479)

[2.1.2 Структура класса VideoScaner. 9](#_Toc94878480)

[2.2. Проблемы, возникшие во время разработки. 13](#_Toc94878481)

[2.3. Тестирование программы. 13](#_Toc94878482)

[ЗАКЛЮЧЕНИЕ 13](#_Toc94878483)

[СПИСОК ЛИТЕРАТУРЫ 14](#_Toc94878484)

# ВВЕДЕНИЕ

На сегодняшний день язык программирования Python является одним из самых востребованных в мире. Область его применения охватывает множество сфер деятельности человека: обработка данных, искусственный интеллект, разработка приложений с интерфейсом, работа с видео и многое другое.

Также данный язык программирования обладает огромной базой пользовательских библиотек и расширений, созданных сообществом и находящихся в открытом доступе. Это расширяет возможности языка до невообразимых масштабов. Работа с сайтами, различные боты для социальных сетей, построение сложнейших приложений и кроссплатформенная разработка. Стоит отметить и то, что он обладает крайне простым синтаксисом и очень прост в освоении.

Основываясь на вышеперечисленное, было принято решение использовать Python для разработки инструмента оптической передачи данных с термометра.

Данный инструмент может пригодится в проведении различных химический и физических экспериментов для протоколирования температуры на определённом промежутке времени.

Цель курсовой работы – создание программного средства для оптической передачи данных с термометра на языке Python.

Задачи:

1. подобрать и систематизировать литературу по теме;
2. изучить принципы формирования изображения термометра (модель);
3. рассмотреть особенности разработки программного продукта на языке Python;
4. разработать интерфейс взаимодействия пользователя с программой;
5. разработать алгоритм, реализовать его на языке программирования Python и провести тестирование и отладку программного средства;
6. оформить документацию на программный продукт.

* Объект исследования: Цифровой термометр.
* Предмет исследования: Язык программирования Python.

Перейдём к рассмотрению языка программирования Python.

# ОСОБЕННОСТИ ЯЗЫКА ПРОГРАММИРОВАНИЯ И РАЗРАБОТКА МОДЕЛИ ПРОГРАММЫ.

## 1.1. Особенности языка программирования Python.

Python — высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества. Язык является полностью объектно-ориентированным в том плане, что всё является объектами.

## 1.2. Используемые библиотеки и инструменты.

PyCharm

IDE PyCharm Community Edition 2020.2 – интегрированная среда разработки для языка программирования Python. Предоставляет средства для анализа кода, графический отладчик, инструмент для запуска юнит-тестов. PyCharm разработана компанией JetBrains на основе IntelliJ IDEA.

Git

Git – распределённая система управления версиями. Она позволяет хранить всю историю изменений в проекте и удобно синхронизировать их между рабочими станциями.

OpenCV

OpenCV – библиотека для языка Python, предназначенная для работы с фото видео информацией и содержит огромное множество механизмов её обработки, включая поддержку библиотек компьютерного зрения и искусственного интеллекта.

В данной работе будут использованы только базовые механизмы этой библиотеки.

NumPy

NumPy – библиотека с открытым исходным кодом для языка Python для работы с многомерными массивами и высокоуровневыми математическими функциям, предназначенных для работы с ними.

Данная библиотека предоставляет функционал для обрезки, масштабирования и вращения изображений библиотеки OpenCV, которые и являются многомерными массивами.

MatplotLib

MatplotLib - библиотека на языке программирования Python для визуализации данных двумерной графикой, использовалась по прямому назначению.

ConfigParser

ConfigParser – встроенная библиотека python, дающая удобный доступ к конфигурационным файлам. Через эту библиотеку к программе был привязан файл конфигурации config.ini.

JSON

JSON - встроенная библиотека python для кодирования и декодирования данных JSON. Используется для экспорта данных в формат json.

Xlsxwriter

XlsxWriter – библиотека языка python предназначенная для записи файлов в формату Excel. Используется для экспорта данных в таблицу Excel.

## 1.3. Принципы формирования изображения на термометре.

В качестве объекта исследования стал цифровой погружной термометр TP300. Он обладает большим диапазоном измеряемых температур (-50°С - +300°С), погрешность измерений не превышает ±1°С на всём диапазоне, точностью в 0.1°С, а также термометр оснащён щупом из нержавеющей стали.

Показания термометра отображаются на стандартном семисегментном дисплее. Для преобразования цифрового сигнала в вид, пригодный для отображения на индикаторе, используется микросхема-дешифратор К176ИД2. Она и выполняет всю работу по формированию изображения. На рисунке 1 представлены изображения цифр отображаемых на дисплее термометра.



Рис. 1 Изображения цифр на дисплее термометра TP300.

## 1.4. Интерфейс программы.

Работа с программной разделена на 5 этапов:

* Transforming – этап предварительной подготовки видео, в котором происходит настройка масштабирования и вращения;
* Placement – этап расстановки ключевых точек;
* Naming – этап присваивания каждой точке имени, положения её сегмента в семисегментном индикаторе и распределения их между различными цифрами индикатора;
* Scanning – этап сканирования в котором на основании всех ключевых точек собирается информация о состоянии каждого элемента индикатора, эта информация дешифруется в формат целого числа и экспортируется в удобном формате;
* Fixing – этап коррекции положения ключевых точек, на случай если в процессе исследования температуры камера и термометр сместились друг относительно друга и расположение сегментов на изображении изменилось.

Последовательность этапов представлена на схеме:



Рис. 2 Последовательность этапов работы программы.

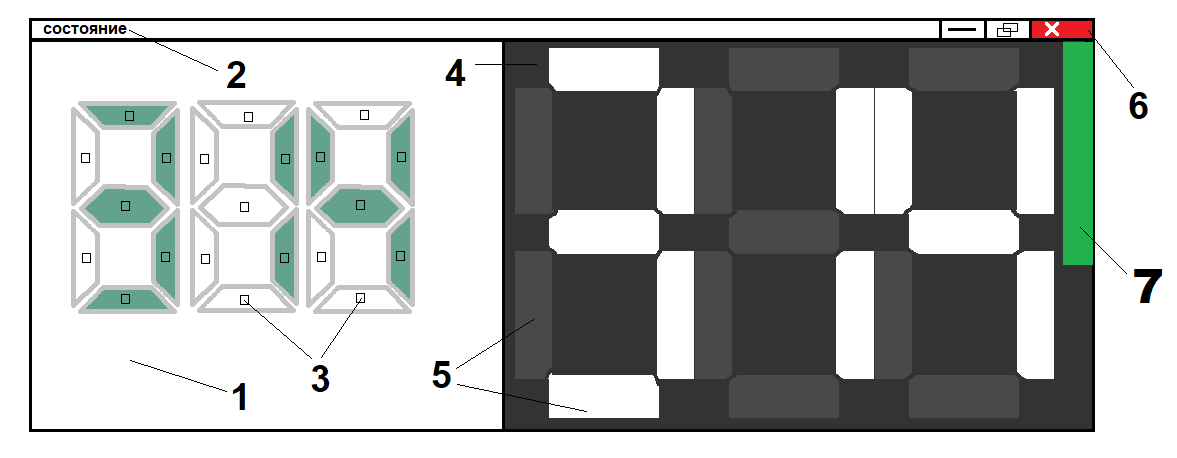


Рис. 3 Интерфейс программы.

Интерфейс программы состоит из 6 элементов:

1. Область просмотра исходного изображения;
2. Заголовок окна, в котором отображается текущий этап работы с программой;
3. Расставленные на исходном видео, ключевые точки. По цвету пиксля изображения определяется текущее значение на дисплее термометра;
4. Область предпросмотра текущей температуры, основанный на цвете ключевых точек;
5. Сегменты показателя;
6. Элементы управления окном.
7. Индикатор выполнения, представляющий собой прямоугольную область, которая «заполняется» областью зелёного цвета по мере сканирования.

В качестве механизма вывода полученных данных программа предоставляет 7 форматов экспорта:

1. RawTXT - текстовый файл, где все значения идут последовательно;
2. PythonList - текстовый файл, содержащий список всех значений, для исполнения в Python;
3. PythonDict - текстовый файл содержащий словарь всех значений с ключами, которые являются временем и пригодный для исполнения в Python;
4. JSON - текстовый файл JSON формата;
5. NumpyArray - бинарный файл массива библиотеки Numpy;
6. Excel – Еxcel таблица;
7. Graph - отображение графика на экране.

Программа настраивается через конфигурационный файл config.ini, который находится в одной директории с программой. Через него можно настроить следующие параметры:

1. videoPath – путь к видеофайлу;
2. startSec – секунда с которой начинается сканирование;
3. decimalPoint – число знаков после запятой в выходном файле;
4. exportFormat – формат экспорта выходных данных, возможные значения перечислены выше;
5. exportFileName – имя выходного файла.

Структура файла config.ini представлена на рисунке 4.



Рис. 4 Структура файла config.ini.

В данном случае сканируемое видео находится по относительному пути Experiments/E-1/video.mp4, сканирование начинается с 5 секунды, и термометр имеет один знак после запятой.

Выходные данные экспортируются в формат "Excel" в файл с названием data.xlsx.

# 2. РАЗРАБОТКА И ТЕСТИРОВАНИЕ ПРОГРАММЫ.

## 2.1. Разработка структуры программы.

### 2.1.1 Общая структура программы.

Программа разрабатывалась в методологии объектно-ориентированного программирования. Благодаря этому программу можно разделить на несколько независимых частей, каждая из которых играет определённую роль. Были созданы несколько классов. Два базовых класса:

* «VideoScaner» - Основной класс, который отвечает за обработку всей графической информации;
* «App» - класс приложения, отвечающий за работу с классом VideoScaner и экспорт полученных данных.

Также были созданы два класса, которые имеют физический аналог:

* «Segment» - объект, являющийся записью об определённом сегменте дисплея
* «Digit» - объект, являющийся записью об определённой цифре дисплея термометра и хранящий в себе массив объектов класса Segment.

Согласно руководству по написанию кода на python PEP 8, все слова в имени классов начинаются с большой буквы.

### 2.1.2 Структура класса VideoScaner.

#### 2.1.2.1 Поля класса VideoScaner.

Класс VideoScaner содержит поля, представленные в таблице:

|  |  |  |
| --- | --- | --- |
| **Имя поля** | **Тип** | **Описание** |
| config | ConfigParser | Парсер конфигурационного файла |
| path | str | Путь к видеофайлу |
| capture | cv2.VideoCapture | Объект захвата видео из видеофайла |
| fps | float | Количество кадров в секунду видео |
| cropping | ((int, int),  (int, int)) | Координаты двух точек, прямоугольник между который и является отображается в результате обрезки |
| croppingHistory | list<((int, int),  (int, int))> | История всех обрезок |
| croppingArea | list<((int, int),  (int, int))> | Необработанные данные о обрезке |
| state | SetterState | Текущий этап работы программы |
| scaleF | float | Коэффициент масштабирования |
| rotate | int | Число от 0 до 3, для которого 90° \* rotate, равен углу поворота изображения |
| digits | list<Digit> | Список всех цифр дисплея |
| noNamedSegments | list<Segment> | Список ещё неназванных сегментов |
| segmentsHistory | list<Segment> | Историй расстановки сегментов |
| nameHistory | list<Segment> | Историй наименований сегментов |
| name\_index | int | Подсчет номера сегмента |
| noNamedDigits | list<Digit> | Список ещё не названных цифр |
| error\_count | int | Количество цифр, которые при сканировании выдали не точные данные |
| selection | list<Segment> | Список выделенных сегментов на этапе коррекции |
| decimalPoint | int | Число знаков после запятой в итоговом числе, полученное из конфигурационного файла |
| totalFrameCount | int | Общее число кадров в видеофайле |
| global\_scan\_data | dict<int, float> | Словарь итоговых данных, где ключ - время видеофайле, значение - показания на дисплее |
| currentSecScan | int | Текущая секунда сканируемого файла |
| scan\_data | list<dict<  SE, bool>> | Список значений всех сегментов в своих цифрах, отправляемый на дешифровку |

Далее рассмотрим механизмы данного класса.

#### 2.1.2.2 Отображения изображения на экране.

За вывод изображения на экран отвечает метод showFrame. Его код представлен далее:

self.frame = self.source\_img.copy()  
  
self.\_cropping()  
self.\_scale()  
self.\_rotate()  
  
self.sizeY, self.sizeX, \_ = self.frame.shape  
  
self.\_drawSegments()  
self.\_drawBad()  
  
cv2.imshow('Frame', self.frame)

Сначала в поле объекта frame мы записываем копию исходного кадра, полученного из видеофайла. Далее полученное изображение обрезается методом \_cropping, масштабируется для лучшего отображения методом \_scale, вращается методом \_rotate. Ширина и высота трансформированного изображения записываются в поля sizeX и sizeY соответственно. Рисуются все сегменты и область предпросмотра, если такая необходима, методами \_drawSegments и \_drawPreview. И в конце с помощью функции imshow в окне с идентификатором “Frame” отображается полученное изображение.

Все вышеперечисленные методы не предназначены для доступа извне класса, поэтому они являются защищёнными и их идентификаторы начинаются с символа нижнего подчёркивания.

#### 2.1.2.3 Реакция программы на события мыши.

Далее представлен код метода set.

def set(self):  
 self.showFrame()  
 cv2.setMouseCallback('Frame', self.onClick)  
  
 self.transform()  
 self.placement()  
 self.naming()

В данном методе, отвечающем за вызов этапов конфигурации программы, после первого отображения кадра их видеофайла вызывается модуля cv2 setMouseCallback, которая устанавливает для окна с идентификатором “Frame” метод onClick, как функцию, вызываемую при действии мыши.

В методе onClick выполняется проверка текущего этапа работы программы из поля класс state, и, в зависимости от него, срабатывает различная логика. Их описание будет в частях описания этапов.

#### 2.1.2.4 Общие принципы работы всех этапов.

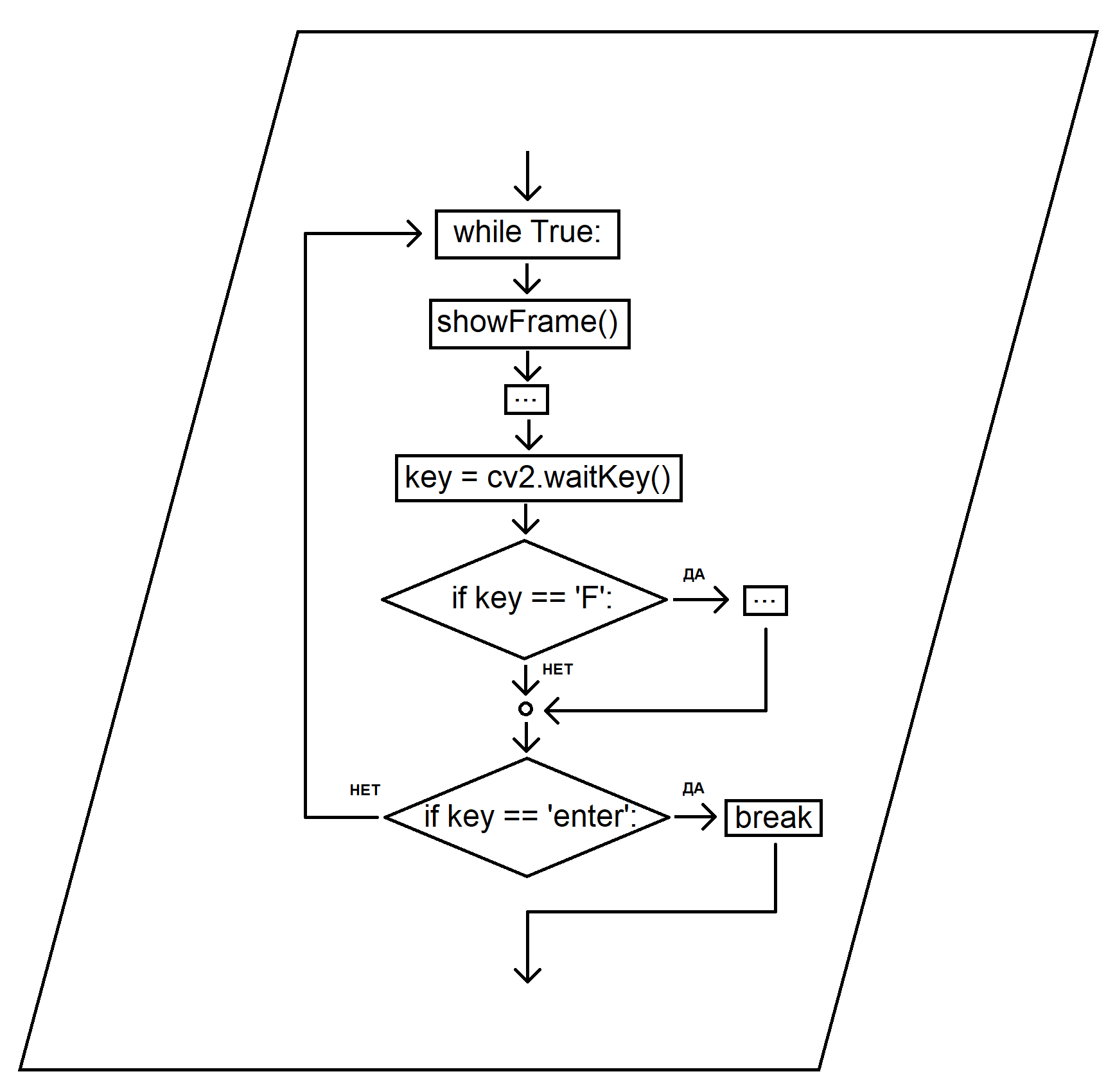


Рис. 5 Общая схема для каждого этапа.

Все этапы являются методами основного класса и реализованы как бесконечный цикл, на каждой итерации которого отрисовывается кадр методом showFrame. Программа ожидает нажатие клавиши на клавиатуре и записывает её в переменную key. Полученная клавиша сравнивается с другими, заранее подготовленными клавишами, и при совпадении вызываются какие-то функции, в примере это клавиша “F”. Также переменная key сравнивается с клавишей “ENTER” для выхода из цикла и перехода к следующему этапу.

Во время ожидания нажатия клавиши программа всё также может получать события от мыши.

#### 2.1.2.5 Этап «Transforming»

Этап «Transforming» необходим для предварительной настройки масштабирования и вращения видео. По нажатию на клавишу ‘R’ (rotate, англ.) к полю «rotate» прибавляется 1 и берётся остаток от деления на 4. После обновления, изображение на экране вращается на 90°. По нажатию на «Backspace» из историй обрезки «croppingHistory» удаляется последний элемент и используется как текущее, путём сохранения в поле cropping. После обновления, изображение на экране возвращается к предидущемму.

В функции onMouse, если текущий этап «Transforming», по нажатию на левую кнопку мыши координаты записываются в первый элемент списка в поле “croppingArea”, а при отжатии кнопки мыши координаты записываются во второй элемент того же списка. Далее этот над этим списком производятся необходимые операции, выполняется проверка на размеры полученного прямоугольника. В случае успешного её прохождения, он преобразуется в неизменяемый картеж и записывается в поле cropping. Вызывается метод showFrame, изображение на экране обрезается.

#### 2.1.2.6 Этап «Placement»

На этапе «Placement» пользователь должен расставить ключевые точки. В методе onMouse, если текущий этап «Placement», вызывается метод setSegment с передачей координат мыши в качестве аргумента. В методе создаётся экземпляр класса Segment по полученным координатам. Новый объект добавляется в список неназванных сегментов и в список истории сегментов. В случае нажатия клавиши ‘backspace’ и программа не перешла на следующий этап, через метод removeLast, удаляется последний элемент списка segmentHistory.

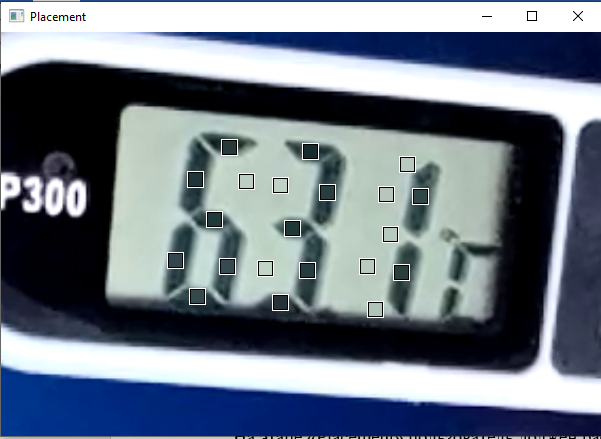


Рис. 6 Пример расстановки ключевых точек.

#### 2.1.2.7 Этап «Naming»

Этап «Naming» необходим для точного присваивания каждой точке положения её сегмента в семисегментном индикаторе и распределения их между различными цифрами индикатора.

Присваивание положения сегмента внутри цифры идёт по порядку сверху вниз, слева направо. То есть сначала пользователь нажимает на верхний сегмент, за ним на левый верхний, правый верхний, средний, нижний левый, нижний правый, нижний. Наименование цифр идёт по порядку слева направо. Названные сегменты приобретают зелёную обводку, а сегменты, входящие в заполненную цифру, синюю.

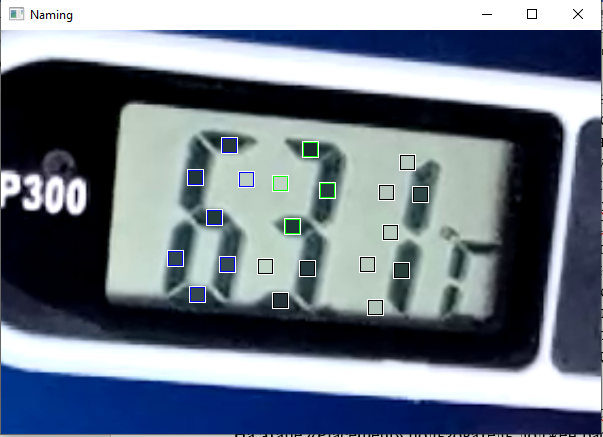


Рис. 7 Пример названных ключевых точек.

В методе onMouse, если текущий этап «Naming», относительно позиции мыши выполняется поиск ближайшего сегмента из списка неназванных и ему присваивается имя из списка всех имён по индексу “name\_index”, после к этому полю прибавляется единица и берётся остаток от деления на 7.

Выход из данного этапа возможен только, если все сегменты имеют имя и список неназванных сегментов пуст.

#### 2.1.2.8 Этап «Scanning»

Scanning – этап сканирования в котором на основании всех ключевых точек собирается информация о состоянии каждого элемента индикатора. Полученная информация интерпретируется классом Interrupt и сохраняется в словарь “global\_scan\_data”. Сегменты, входящие в цифру, которую удалось интерпретировать без проблем, имеют синюю обводку, а сегменты чья интерпретация завершилась с ошибкой жёлтую.



Рис. 8 Пример сканирования с ошибкой.

При нажатии клавиши ‘F’ программа перейдёт на этап «Fixing». По окончанию сканирования сырые данные возвращаются в класс App, который их экспортирует в нужный формат. Все события мыши на данном этапе игнорируются.

#### 2.1.2.9 Этап «Fixing»

Fixing – этап коррекции положения ключевых точек, на случай если в процессе исследования температуры камера и термометр сместились друг относительно друга и расположение сегментов на изображении изменилось.

Реализована коррекция положения ключевых точек, с помощью суммирования координат сегментов, находящихся в списке выбранных с некоторым смещением, по нажатию на клавишу на клавиатуре.

По нажатию на левую кнопку мыши список выбранных сегментов очищается и в него записывается ближайший к курсору сегмент. По нажатию на правую кнопу мыши в этот список записывается ближайший сегмент с сохранением всех уже выбранных. По нажатию на клавишу ‘F’ на клавиатуре выделяются все сегменты. По нажатию на ‘ENTER’ программа возвращается к сканированию с новыми позициями сегментов.

Сегменты, находящиеся в списке выбранных, приобретаю оранжевую обводку.



Рис. 9 Пример коррекции ошибки.

### 2.1.3 Структура класса «Digit».

Класс «Digit» является записью об определённой цифре дисплея термометра и хранящий в себе массив объектов класса Segment. Структурно данный класс необходим для управления своими сегментами, получения данных с сегментов, отправки полученных данных на интерпретацию и возвращения данных в класс «VideoScanner». Для реализации данных функция у данного класса есть следующие методы:

* setSegment – метод записи сегмента в данную цифру;
* removeLast – удалить последний записанный сегмент;
* sort – сортировка сегментов в списке данного объекта по порядку сверху вниз, слева направо;
* isFull – проверка количества сегментов у данной цифры. Возвращает истинное значение только если у данного объекта 7 сегментов;
* isNamed – проверка, названы ли все сегменты;
* scan – сканирование, интерпретация, возвращение полученных данных;
* interpret – статический метод по работе с классом «Interrupt», интерпретирующий полученный от сегментов данные.

### 2.1.4 Структура класса «Segment».

Объекты класса «Segment» являются записью об определённом сегменте дисплея. Он содержит следующие поля:

* pos – поле типа картеж двух целых чисел, координаты на изображении где расположен сегмент;
* isSelected – поле логического типа, изначально принимает значение ЛОЖЬ и принимает значение ИСТИНА только в случае если данный сегмент выбран на этапе коррекции ошибок;
* name – поле, обозначающее расположение сегмента внутри своей цифры
* videoScaner – ссылка на основной объект сканера данной программы;
* digit – ссылка на цифру в которой находится данный сегмент.

Также данный класс содержит следующие методы:

* setDigit – установка для данного сегмента родительской цифры переданной как аргумент метода;
* select и deselect – методы изменяющие поле isSelected при выборе данного сегмента на этапе коррекции ошибок;
* getColor – метод возвращающий цвет пикселя на изображении по координатам из поля pos;
* draw – отрисовка сегмента на изображении;
* scan – анализ цвета на изображении по координатам сегмента;
* move – сдвиг координат сегмента на указанное смещение.

### 2.1.5 Структура класса «Interrupt»

Класс Interrupt необходим для дешифровки данных полученных с сегментов в целое число. Класс состоит из одного поля и одного статического метода. Таблицы dataSet, хранящей состояния всех сегментов для каждой цифры по образцу с рисунка 1. Метода find который ищет совпадение переданных в него данных и одной из строк в таблице dataSet.

## 2.2. Проблемы, возникшие во время разработки.

### 2.2.1 Проблема чёткого поиска.

Суть проблемы

Так как класс Interrupt ищет точное совпадение полученных данных и одной из строк в таблице то если хотя бы в одном сегменте произойдёт ошибка при измерении состояния сегмента выходное значение будет непригодно для использования.

Решение

Реализация нечёткого поиска. Существует много алгоритмов нечёткого поиска, но в данном случае был реализован следующий. Если чёткого совпадения не было найдено создаётся список нулей длинной 10 элементов. И для каждой строки в таблице dataSet вычисляется значение какого количества сегментов не совпало с дешифруемыми значениями. Полученные вычисления сохраняются в список. После чего в данном списке находится минимальное значение, индекс которого и будет результатом нечёткого поиска.

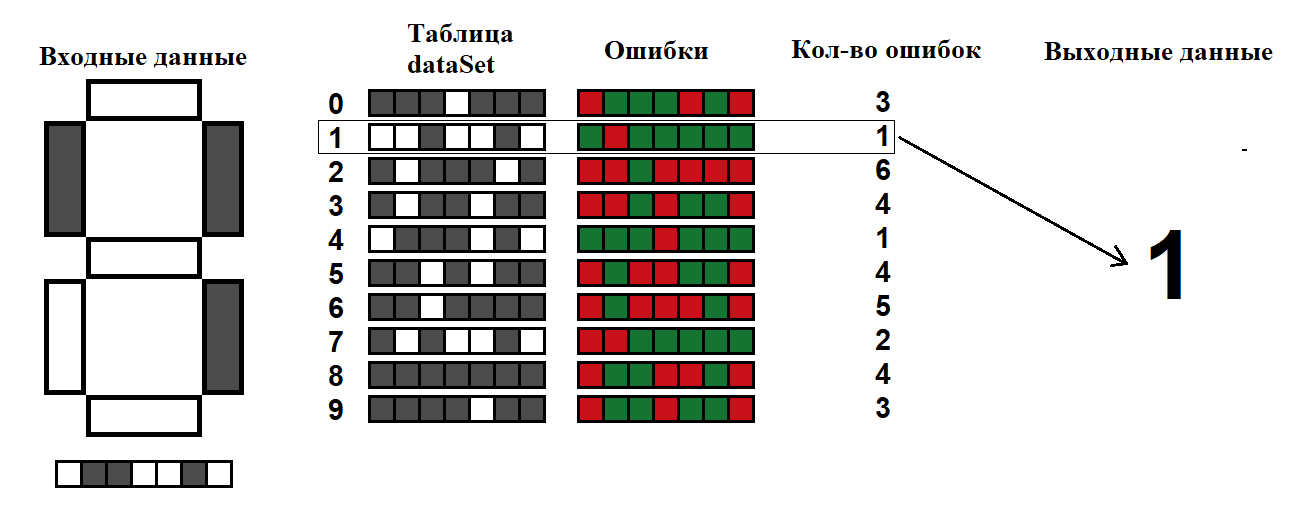


Рис. 10 Иллюстрация нечёткого поиска.

## 2.3. Тестирование программы.

### 2.3.1 Описание экспериментов.

Всего для тестирования было приготовлено 3 эксперимента:

* E-Test – чёрно-белое видео созданное с помощью компьютерной графики;
* E-ColorTest – цветное видео являющиеся копией видео эксперимента E-Test за исключением того что цвета для фона и сегментов максимально приближены к реальным, а также была добавлена ещё одна цифра повёрнутая на 90°;
* E-1 – первый реальный эксперимент.

### 2.3.2 Эксперименты E-Test и E-ColorTest.

Данные эксперименты проведены с целью проверки базовых механизмов программы. Оба видеоролика созданы с помощью компьютерной графики. Значения на экране идут последовательно от 0 до 9 четыре раза.

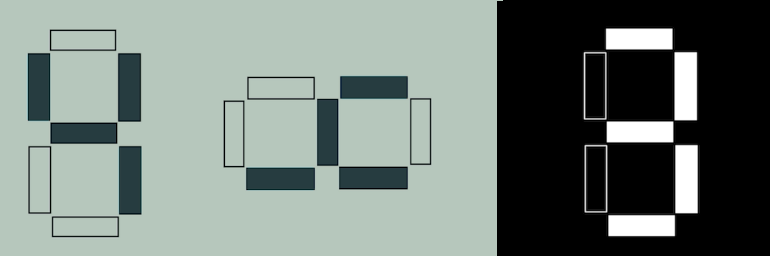


Рис. 11 E- ColorTest и E-Test.

Программа без проблем анализирует данные видеофайлы и в результате выводит следующий график.

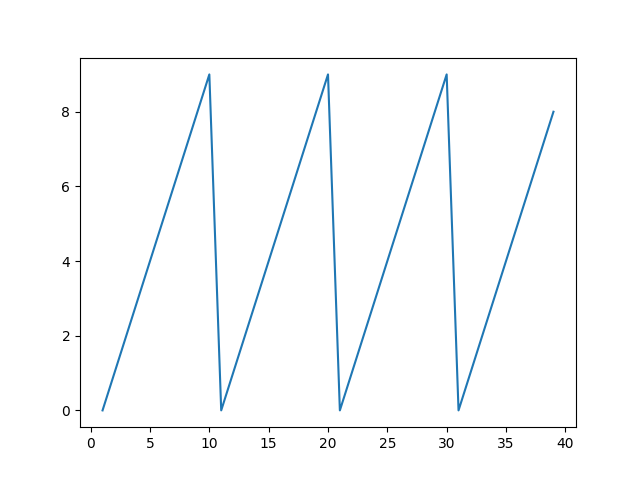


Рис 12. Результат сканирования E-ColorTest.

### 2.3.3 Эксперимент E-1.

Эксперимент E-1 проведён на реальном примере. Стакан кипячённой воды с начальной температурой 63.2°C постепенно остывает. Также на девятой и пятнадцатой минуте эксперимента в стакан было добавлено 10 мл холодной воды.

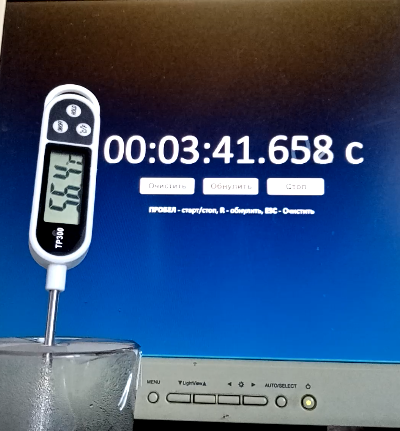


Рис. 13 Фрагмент из эксперимента E-1.

Результат сканирования:

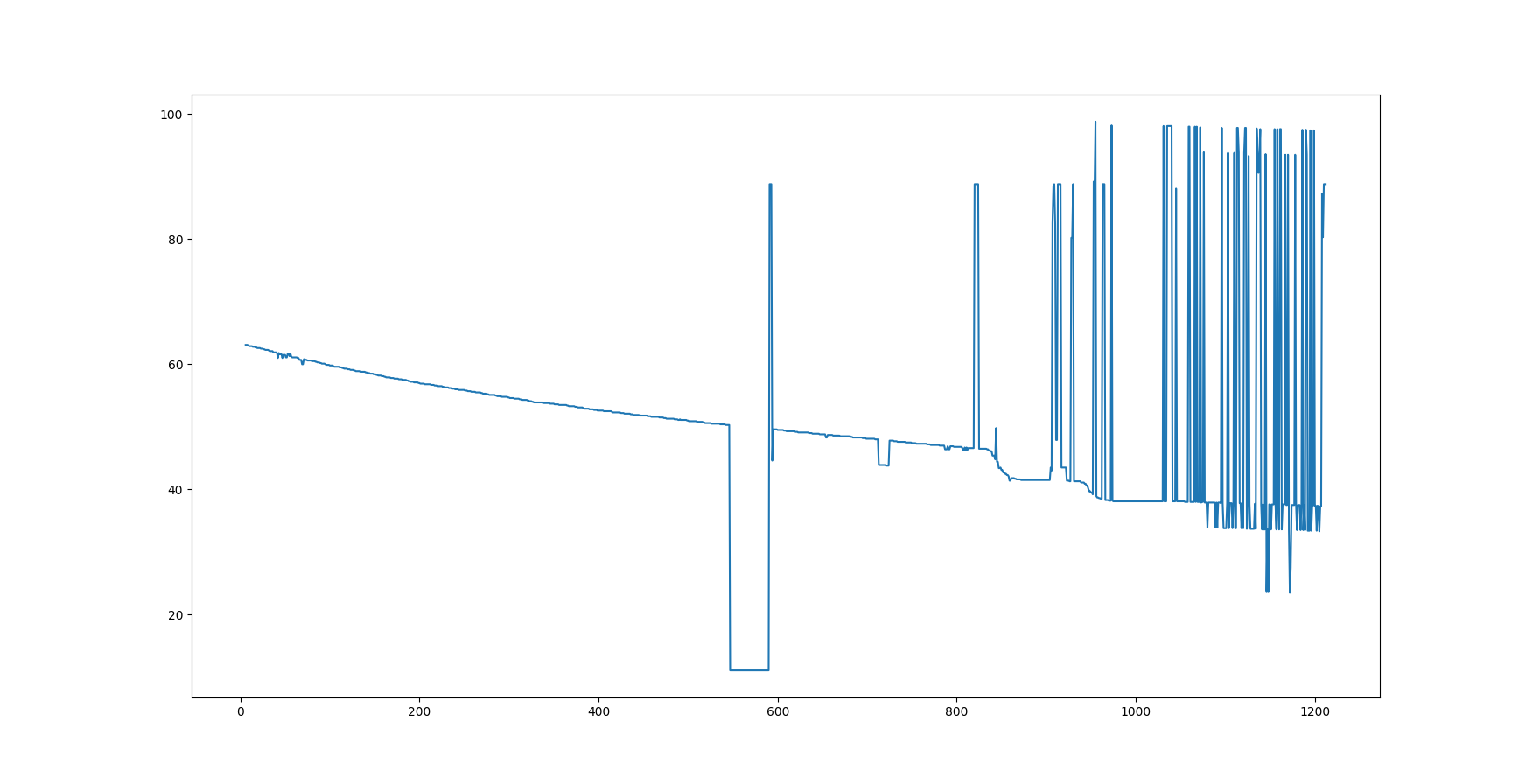


Рис. 14 Результат сканирования E-1.

Увы, но внешние обстоятельства внесли в исходное видео такие дефекты как: расфокус камеры, авто-выключение термометра на 550 секунде и изменение уровня освещения. Из-за всего вышеперечисленного удачное сканирование не было возможно изначально. Но в целом, я считаю, результат удовлетворительным.

# ЗАКЛЮЧЕНИЕ

# СПИСОК ЛИТЕРАТУРЫ

1. <https://docs.python.org/3/> - официальная документация по Python
2. <https://numpy.org/doc/> - официальная документация по NumPy
3. <https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html> - официальная документация по OpenCV для языка Python.
4. [https://docs.opencv.org/](https://docs.opencv.org/%20) - официальная документация по OpenCV для языка программирование C.
5. <http://www.catb.org/jargon/html/S/syntactic-sugar.html>
6. <https://www.python.org/dev/peps/pep-0008/> - руководство по написанию кода на Python