

**ОТЧЕТ**

по производственной практике ПП.01.01 по модулю ПМ.01  
«Разработка программных модулей программного обеспечения   
для компьютерных систем»

по специальности 09.02.03 «Программирование в компьютерных системах»

Выполнил студент гр. П1-18

Ларченко М. А.

\_\_\_\_\_\_\_\_\_\_ (подпись)

Принял преподаватель

Гусятинер Л. Б.

\_\_\_\_\_\_\_\_\_\_(подпись)

\_\_\_\_\_\_\_\_\_\_\_(оценка)

Королев, 2021

**Оглавление**

[1 Общие сведения об организации 4](#__RefHeading___Toc1014_903855424)

[1.1 Структура организации 4](#__RefHeading___Toc1016_903855424)

[1.2 Структура отдела 4](#__RefHeading___Toc1018_903855424)

[1.3 Основные функции отдела 5](#__RefHeading___Toc1020_903855424)

[1.4 Нормативные документы, которые регламентируют профессиональную деятельность 5](#__RefHeading___Toc1022_903855424)

[1.5 Информационные технологий предприятия 6](#__RefHeading___Toc1024_903855424)

[1.6 Программное обеспечение предприятия 6](#__RefHeading___Toc1026_903855424)

[1.7 Задачи, подлежащие автоматизации 6](#__RefHeading___Toc1028_903855424)

[2 Содержание выполняемых видов работ 7](#__RefHeading___Toc1030_903855424)

[2.1 Разработка спецификаций отдельных компонент 7](#__RefHeading___Toc1032_903855424)

[2.2 Разработка кода программного продукта на основе готовых спецификаций на уровне модуля 8](#__RefHeading___Toc1034_903855424)

[2.3 Отладка программного модуля с использованием специализированных программных средств 9](#__RefHeading___Toc1036_903855424)

[2.4 Тестирование программного модуля 11](#__RefHeading___Toc1038_903855424)

[2.5 Оптимизация программного кода модуля 13](#__RefHeading___Toc1040_903855424)

[2.6 Разработка компонентов проектной и технической документации с использованием графических языков спецификаций 13](#__RefHeading___Toc1042_903855424)

[3 Выводы 15](#__RefHeading___Toc1044_903855424)

[4 Дневник практики 16](#__RefHeading___Toc1046_903855424)

[5 Список использованной литературы 17](#__RefHeading___Toc1048_903855424)

[3 Приложения 18](#__RefHeading___Toc1050_903855424)

Введение

На 3 курсе обучения в ККМТ, студентом группы П1-18 Ларченко Михаилом была пройдена производственная практика по модулю ПМ.01 «Разработка программных модулей программного обеспечения для компьютерных систем» на предприятии ЗАО «НВП Болид». Студент получил задание разработать программный модуль (ПМ) и инструкцию оператору для визуализации исходных данных, текущих и выходных результатов функционирования ПМ, разрабатываемых по индивидуальным заданиям № 1-3, а также изучить структуру предприятия, на котором он проходит практику.

# **О****бщие сведения об организации**

## Структура организации

Организационная структура организации представлена на Рис. 6.1 в Приложении 1.

Информационная структура организации представлена на Рис. 6.2 в Приложении 2.

## Структура отдела

Во главе стоит руководитель отдела, в его подчинении находятся заместитель руководителя и инженер-программист, техники-программисты выполняют поручения инженера-программиста.

Организационная структура отдела представлена на Рис. 1.2.1.

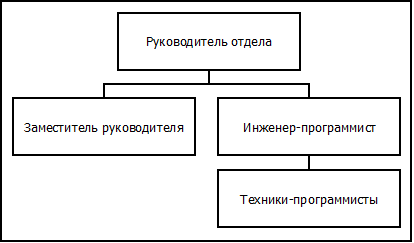


Рис. 1.2.1. Организационная структура отдела.

Информационная структура отдела представлена на Рис. 1.2.2.

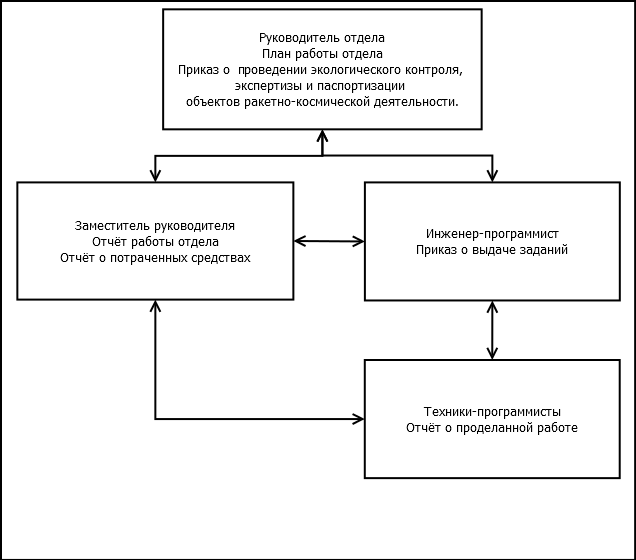


Рис. 1.2.2. Информационная структура отдела

## Основные функции отдела

* обеспечение создания и развития информационно-телекоммуникационных систем;
* обеспечение обслуживания и технической поддержки информационно-телекоммуникационных систем.

## **Нормативные документы, которые регламентируют профессиональную деятельность**

* Политика информационной безопасности организации;
* Положение по обеспечению информационной безопасности организации;
* Положение о порядке доступа к информационным ресурсам;
* Положение о коммерческой или служебной тайне;
* Порядок обращения с информацией, подлежащей защите;
* Порядок защиты от несанкционированного доступа к информации и незаконного вмешательства в процесс функционирования информационной системы;
* Положение об отделе информационной безопасности;
* Классификация и перечень основных видов угроз информационной безопасности предприятия;
* Требования и рекомендации по обеспечению информационной безопасности предприятия;
* Положение об отделе технической защиты информации.

## **Информационные технологий предприятия**

* GitLab

## **Программное обеспечение предприятия**

* Qt Creator 6.1.1
* Windows 10

## **Задачи, подлежащие автоматизации**

* Создание систем обеспечения безопасности предприятий.
* Разработка систем управления системами безопасности предприятия.
* Оценка рисков проектов создания сложных технических систем.

# **Содержание выполняемых видов работ**

## **Разработка спецификаций отдельных компонент**

Всего было дано 6 заданий на разработку программных модулей:

1. ПМ1. Программный модуль (ПМ) на основе выданного описания алгоритма построчного слияния двух или более текстовых файлов в один с использованием многопоточности.

Входные данные модуля: текстовые файлы.

Выходные данные: текстовый файл.

1. ПМ2. Программный модуль (ПМ) на основе выданного описания алгоритма многопоточного умножения двух матриц.

Входные данные модуля: текстовый файл с матрицей 1, текстовый файл с матрицей 2.

Выходные данные: текстовый файл с полученной матрицей

1. ПМ3. Программный модуль (ПМ) на основе выданного описания алгоритма обработки базы данных полученной из системы «Орион».

Входные данные модуля: база данных системы «Орион».

Выводимые данные: отфильтрованные по заданным правилам записи из БД.

1. ПМ4. Программный модуль (ПМ) на основе выданного описания алгоритма вычисления контраста и освещенности входных черно-белых изображений.

Входные данные модуля: черно-белые изображения.

Выводимые данные: оценка освещенности, оценка контраста.

1. ПМ5. Программный модуль (ПМ) для получения полной версии ПО сетевой камеры.

Входные данные модуля: IP-адрес камеры, порт, логин, пароль.

Выводимые данные: полная версия ПО.

1. ПМ6. Программный модуль (ПМ) c точной реализацией на Qt части демонстрационной программы написанной с помощью MFC.

Входные данные модуля: IP-адрес камеры, порт, логин, пароль.

## **Разработка кода программного продукта на основе готовых спецификаций на уровне модуля**

Функция получения контраста изображения приведёна в Листинге 1. Данная функция получает объект изображения и среднее арифметическое яркости всех пикселей изображения и вычисляет root mean square (RMS) контраст от 0 до 0.5.

**Листинг 1. Модуль отображения всех объектов.**

// Вычисление контсраста изображения

double calculateContrast(const QImage& image, int mean)

{

double sum\_c = 0;

for (int x=0; x < image.width(); x++)

for (int y=0; y < image.height(); y++)

sum\_c += pow(((double)image.pixelColor(x, y).red()/256.0)-((double)mean/256.0), 2);

return sqrt(sum\_c/(double)(image.width()\*image.height()));

}

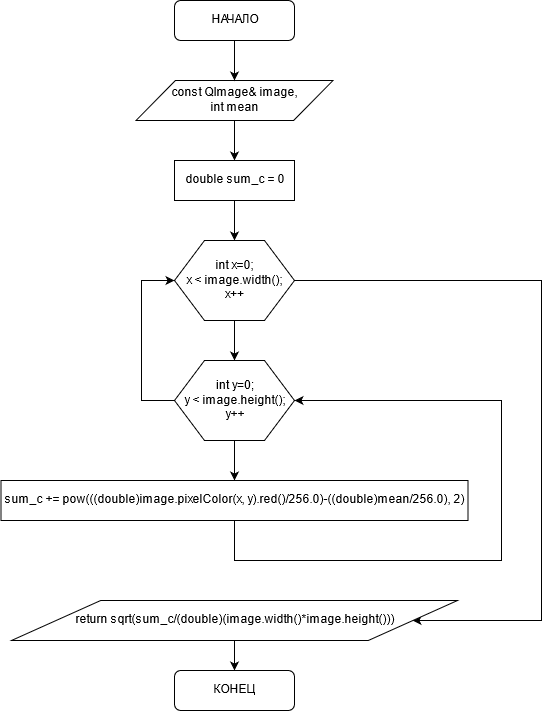


Рис. 2.2.1. Блок-схема листинга 1

## **Отладка программного модуля с использованием специализированных программных средств**

Отладка проводилась с помощью функции qDebug(), а также окон “Локальные переменные” и “Выражения”

qDebug - это функция класса QDebug, позволяющая выводить сообщения и данные в консоль.

С её помощью я проводил отладку функции on\_btnSelectFile1\_clicked

класса MainWindow (Листинг 3.1 Приложение 1), проводилась проверка правильности внутренних переменных фукнкции. Пример приведён на Рис.2.3.1.

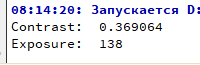


Рис.2.3.1. Пример отладки qDebug

## Тестирование программного модуля

Тестирование программы с использованием входного изображения (Рис.2.4.1).



Рис. 2.4.1. Входные данные

Выходными данными программы являются две характеристики (яркость и контрастность) на форме (Рис. 2.4.3.) и и гистограмма на форме. (Рис. 2.4.4)

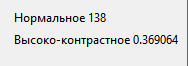


Рис. 2.4.3. Выходной журнал работы программы

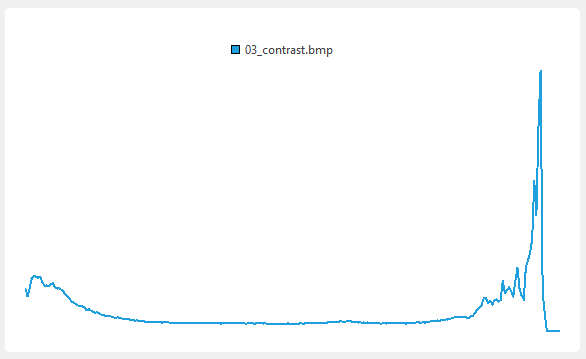


Рис. 2.4.4. Файл минимального времени приёма

## **Оптимизация программного кода модуля**

Была оптимизирована функция вычисления контраста в Листинге 2.5.1. Для оптимизации был удален лишний код так-как он моджет быть заменен вызовом функции calculateExposure и передачи параметра в оптимизируемую функцию. Таким образом код был уменьшен, улучшена модульность программы, а также убраны лишние вычисления которые могут быть закэшированы.

Листинг 2.5.1. Функция ычисления контраста.

// Вычисление контсраста изображения

double **calculateContrast**(const QImage& image)

{

int64\_t sum=0;

int64\_t count=0;

for (int x=0; x < image.width(); x++)

for (int y=0; y < image.height(); y++)

{

sum += image.pixelColor(x, y).red();

count++;

}

int mean = sum/count;

double sum\_c = 0;

for (int x=0; x < image.width(); x++)

for (int y=0; y < image.height(); y++)

sum\_c += pow(((double)image.pixelColor(x, y).red()/256.0)-((double)mean/256.0), 2);

return sqrt(sum\_c/(double)(image.width()\*image.height()));

}

Листинг 2.5.2. Функция определения цвета шкалы после оптимизации.

// Вычисление контсраста изображения

double **calculateContrast**(const QImage& image, int mean)

{

double sum\_c = 0;

for (int x=0; x < image.width(); x++)

for (int y=0; y < image.height(); y++)

sum\_c += pow(((double)image.pixelColor(x, y).red()/256.0)-((double)mean/256.0), 2);

return sqrt(sum\_c/(double)(image.width()\*image.height()));

}

## **Разработка компонентов проектной и технической документации с использованием графических языков спецификаций**

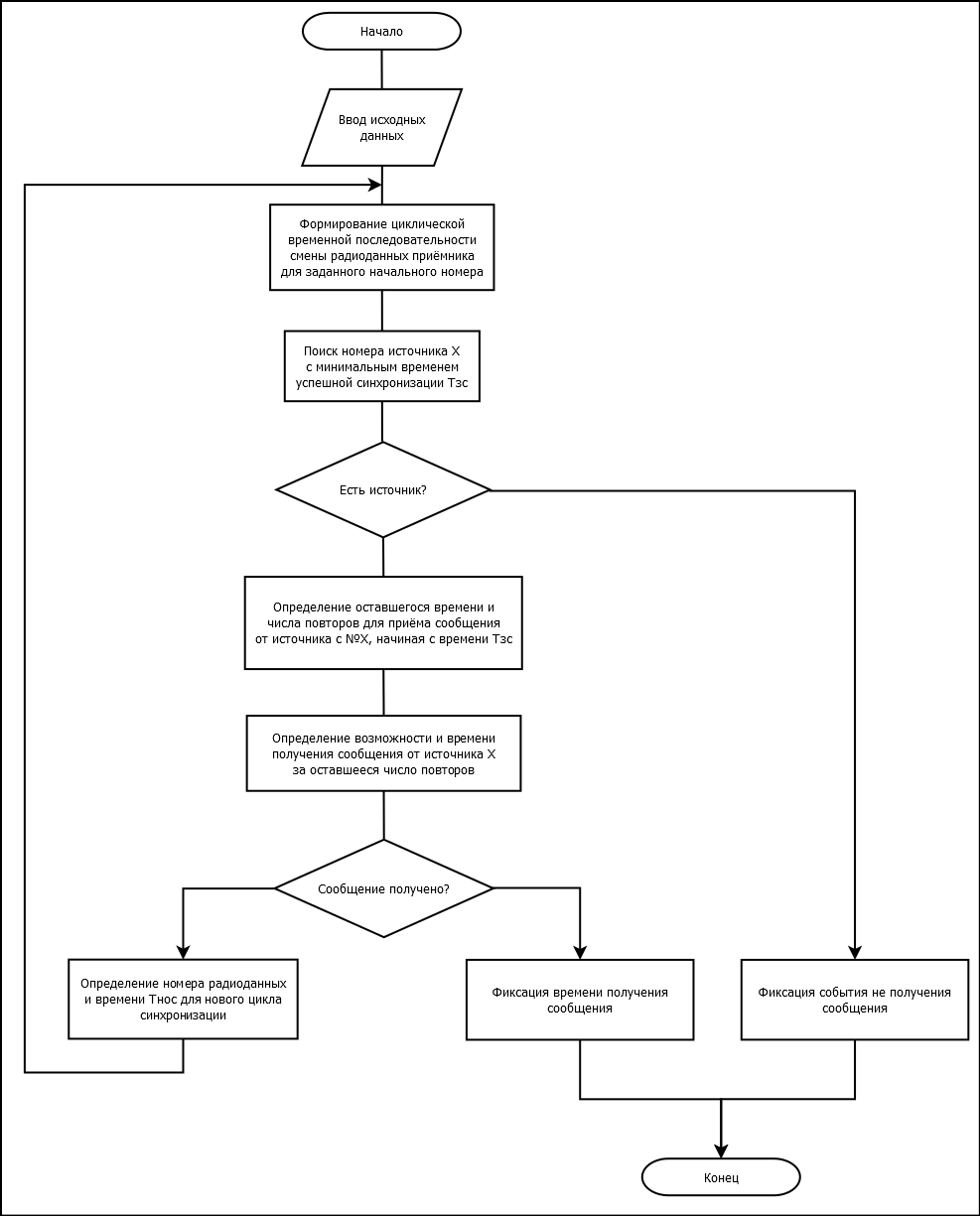


Рис. 2.6.1 Общая блок-схема модуля

# **Выводы**

Полученные навыки:

* Работа со структурами в Qt
* Работа с базами данных в Qt
* Работа с изображениями в Qt
* Работа с неизвестными SDK в C++
* Переписывание программ с MFC на Qt
* Компиляция программы в Qt
* Создание баз данных в SQLite Studio

Полученные умения:

* Разработка программ в Qt Creator
* Работа с файловой системой в Qt Creator
* Работа с файлами qml
* Разработка программ на qml

# **Дневник практики**

Таблица 1. Дневник практики.

|  |  |  |
| --- | --- | --- |
| Дата | Содержание работ | Отметка о выполнении |
| 15.03 | Изучение предметной области |  |
| 17.03-22.03 | Освоение QObject и создание программы FirstApp |  |
| 22.03-29.03 | Освоение многопоточности в Qt и создание программы SecondApp |  |
| 29.03-05.04 | Изучение встраивания SQLite в программу на Qt, а также разработка ThirdApp |  |
| 05.04-12.04 | Изучение программного определения контрастности и освещенности изображения, разработка программы BmpAnalyzer |  |
| 19.04-26.04 | Программа GetVerlyzer написанная с помощью NetSDK |  |
| 26.04-03.05 | Изучение примера программы написанной на MFC |  |
| 03.05-07.05 | Переписывание изученной программы с MFC на Qt, разработка программы Encode |  |
| 31.05-05.06 | Завершение разработки программы Encode |  |

# **Список использованной литературы**

1. Документация по Qt. <https://doc.qt.io/>
2. Википедия. Статья «Контраст». <https://en.wikipedia.org/wiki/Contrast_(vision)#RMS_contrast>

# **Приложения**

**Приложение 3**

Листинг 3.1 файл mainwindow.cpp.

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include <QPixmap>

#include <QFileDialog>

#include <QStandardPaths>

#include <QMessageBox>

#include <QtDebug>

#include <QPainter>

#include <QLineSeries>

#include <QMap>

#include <QPair>

#include <array>

#include <cmath>

#include <imageprocessing.h>

QMap<QPair<double, double>, QString> brightnessLevels {

{QPair<double, double>{0, 85}, "Слишком темное"},

{QPair<double, double>{85, 170}, "Нормальное"},

{QPair<double, double>{170, 256}, "Засвеченное"}

};

QMap<QPair<double, double>, QString> contrastLevels {

{QPair<double, double>{0, 0.16}, "Мало-контрастное"},

{QPair<double, double>{0.16, 0.32}, "Нормальное"},

{QPair<double, double>{0.32, 0.5}, "Высоко-контрастное"}

};

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(*parent*)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~*MainWindow*()

{

delete ui;

}

QString rangeSearch(QMap<QPair<double, double>, QString> strings, double value)

{

for (QPair<double, double> range : strings.keys())

if ((range.first <= value) && (range.second > value))

return strings[range];

}

void MainWindow::on\_btnSelectFile1\_clicked()

{

QString inputFilename1 = QFileDialog::getOpenFileName(this,

tr("Open File"), QStandardPaths::standardLocations(QStandardPaths::DocumentsLocation)[0], tr("Image Files (\*.bmp)"));

if (inputFilename1.isEmpty())

return;

QImage image(inputFilename1);

Histogram hist = getImageHistogram(image);

QLineSeries\* series = new QLineSeries();

for (int i=0; i<255; i++)

series->append(i, hist[i]);

series->setName(inputFilename1.split('/').last());

ui->chart->chart()->addSeries(*series*);

ImageCharacteristics chars = processImage(image);

qDebug() << "Contrast: " <<chars.contrast;

qDebug() << "Exposure: " << chars.exposure;

ui->labelBrightness->setText(rangeSearch(brightnessLevels, chars.exposure)+' '+QString::number(chars.exposure));

ui->labelContrast->setText(rangeSearch(contrastLevels, chars.contrast)+' '+QString::number(chars.contrast));

}

**Приложение 4**

Листинг 3.2 файл imageprocessing.cpp.

#include "imageprocessing.h"

#include <cmath>

// Вычисление средней освещенности изображения

int calculateExposure(const QImage& image)

{

int64\_t sum=0;

int64\_t count=0;

for (int x=0; x < image.width(); x++)

for (int y=0; y < image.height(); y++)

{

sum += image.pixelColor(x, y).red();

count++;

}

return sum/count;

}

// Вычисление контсраста изображения

double calculateContrast(const QImage& image, int mean)

{

double sum\_c = 0;

for (int x=0; x < image.width(); x++)

for (int y=0; y < image.height(); y++)

sum\_c += pow(((double)image.pixelColor(x, y).red()/256.0)-((double)mean/256.0), 2);

return sqrt(sum\_c/(double)(image.width()\*image.height()));

}

ImageCharacteristics processImage(const QImage& image)

{

ImageCharacteristics imgChar;

imgChar.exposure = calculateExposure(image);

imgChar.contrast = calculateContrast(image, imgChar.exposure);

return imgChar;

}

Histogram getImageHistogram(const QImage& image)

{

Histogram hist{0};

for (int x=0; x < image.width(); x++)

for (int y=0; y < image.height(); y++)

hist[image.pixelColor(x, y).red()]++;

return hist;

}