Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное
учреждение высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Институт открытого и дистанционного образования
Кафедра Техники, технологии и строительства

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой,
к.т.н., доцент
_____К.М. Виноградов
_____2020 г.

Разработка системы управления оранжереей

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ–09.03.01.2020.009.00.000ПЗ ВКР

Руководитель работы,
старший преподаватель кафедры
ТТС
_____С.Н. Кононов
_____2020 г.

Автор работы
студент группы ДО – 525
_____М.Н. Исмагилов
_____2020 г.

Нормоконтролер,
преподаватель
_____О.С. Микерина
_____2020 г.

Челябинск 2020

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет»
(национальный исследовательский университет)
Институт открытого и дистанционного образования
Кафедра Техника, технологии и строительство
Направление 09.03.01 «Информатика и вычислительная техника»

УТВЕРЖДАЮ
Заведующий кафедрой
_____ К.М. Виноградов
_____2020 г.

ЗАДАНИЕ

на выпускную квалификационную работу студента
Исмагилова Марата Наильевича
Группа ДО-525

1. Тема работы: Разработка системы управления оранжереей
   утверждена приказом по университету от «24» апреля 2020 г. № 627
2. Срок сдачи студентом законченной работы 11.06.2020
3. Исходные данные к работе
   3.1. Вид используемой технологии: MQTT
   3.2. Используемые языки программирования: C#, Arduino IDE
   3.3. Объект разработки: Система для управление микроклиматом оранжереи

4. Содержание расчетно-пояснительной записки (перечень подлежащих разработке вопросов)

4.1. Изучение предметной области и постановка задач

4.1.1. Предметная область

4.1.2. Обзор передовых решений

4.2. Сравнение отечественных и передовых зарубежных технологий решений

4.3. Выбор технологий функционирования разработки и среды разработки

4.3.1. Язык программирования для разработки приложения

4.3.2. Платформа для аппаратной части проекта для непосредственного управления оранжереей

4.3.3. Связь компьютера с микроконтроллером

4.4. Описания функционирование разработки

4.4.1. Общий принцип функционирования системы

4.4.2. Обзор пользовательской части программы

4.5.Листинг программ

5. Перечень графического материала

5.1.Презентация на 10–12 слайдах

5.1.1. Актуальность

5.1.2. Цель и характеристика

5.1.3. Внутреннее функционирование приложения

5.1.4. Описание элементов управления

5.1.5. Предложение по дальнейшей модернизации системы

Дата выдачи задания 27.01.2020 г.

Руководитель _____ С.Н. Кононов
(подпись)

Задание принял к исполнению _____ М.Н. Исмагилов
(подпись)

КАЛЕНДАРНЫЙ ПЛАН

Наименование этапов выпускной квалификационной работе	Срок выполнения этапов работы	Отметка руководителя о выполнении
Изучение предметной области и постановка задач	27.01.2020 – 5.02.2020	
Сравнение отечественных и передовых зарубежных технологий решений	6.02.2020 – 20.02.2020	
Выбор технологий функционирования разработки и среды разработки.	21.02.2020 – 10.03.2020	
Описание функционирования разработки.	11.03.2020 – 28.03.2020	
Листинг программ.	29.03.2020 – 15.05.2020	
Оформление пояснительной записки	16.05.2020 – 10.06.2020	
Создание презентации, подготовка доклада	16.06.2020 - 22.06.2020	

Зав. кафедрой _____ К.М. Виноградов
(подпись)

Руководитель работы _____ С.Н. Кононов
(подпись)

Студент_____ М.Н. Исмагилов
(подпись)

АННОТАЦИЯ

Исмагилов, М.Н., Разработка системы управления оранжереей - Челябинск: ФГАОУ ВО «ЮУрГУ (НИУ)», ИОДО; 2020, 61 с., 13 ил., библиографический список – 9 наименований, 3 таблицы.

В выпускной квалификационной работе представлена разработка программного обеспечения для управления микроклиматом в оранжерее.

Во введении содержатся цель, поставленная перед автором, сформулированы задачи исследования, определена актуальность темы и функции разрабатываемой системы.

В первом разделе рассматривается общая теория, происходит описание предметной области. Изучаются основные принципы функционирования системы управления микроклиматом. Формируются требования к разрабатываемой системе

Во втором разделе происходит обзор отечественных и зарубежных аналогов, а также формируются требования к системе.

В третьем разделе происходит выбор технологий, по которым будет происходить проектирование.

В четвертом разделе описывается разработанная программа.

В пятом разделе показан листинг программы.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Оранжереи или теплицы предназначены для обеспечения оптимального микроклимата для роста и развития растений. Это могут быть и большие промышленные сооружения, и небольшое место на подоконнике для выращивания любимого цветка. Но даже за самой крохотной теплицей на подоконнике нужен уход: осуществление полива, поддержание нужной температуры, уровня освещенности и тому подобное.

Основная функция умной оранжереи – оперативное получение всей необходимой информации об климатических параметрах нашей теплицы: температура и влажность воздуха, температура и увлажненность почвы, освещенность теплицы, т. е. осуществлять мониторинг климатических параметров теплицы. Система должна выводить данные мониторинга можно на дисплей, или с помощью светодиодов оповещать о критических значениях климатических параметров, или получать данные на рабочий компьютер. Также система будет производить управление микроклиматом оранжереи в автоматическом режиме по заданным пользователем параметрам.

Целью данной разработки является формирование системы по обеспечению управления микроклиматом в оранжерее.

Для достижения цели необходимо выполнить следующие задачи:

- изучение предметной области;
- формирование общих принципов управления микроклиматом;
- выбор технологий, по которым будет производиться разработка;
- разработка общего принципа функционирования системы;
- написание программы по управлению микроклиматом.

Объектом выпускной квалификационной работы является система управления микроклиматом в оранжерее.

Предмет выпускной квалификационной работы представляет из себя программа для управления микроклиматом в оранжрее.

1 ИЗУЧЕНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И ПОСТАНОВКА ЗАДАЧ

1.1 Предметная область

Автоматическая система управления технологическим процессом (АСУ ТП) оранжереи – это основа высокой урожайности и рентабельности производства тепличного комплекса. Известно, что рост и развитие растений напрямую зависят от условий внешней среды: количества света, тепла, качества воздуха, воды, питательных веществ и других факторов. Автоматизированное управление технологическими параметрами в теплице позволяет контролировать микроклимат: температуру и влажность воздуха, концентрацию углекислого газа, влажность и уровень pH почвы, освещение, а также дистанционно управлять режимами работы теплицы. Система автоматизированного контроля микроклимата может включать автоматический полив, систему подкормки углекислым газом, обогрев, вентиляцию и др. Сегодня АСУ ТП теплицы широко применяется на территориях частного земельного участка и фермерских хозяйствах для поддержания требуемых температурно-влажностных условий в теплицах, парниках и оранжереях.

Основной принцип конструкции умной теплицы: конструкция теплицы в первую очередь должна быть функциональной, то есть выполнять ту функцию, для которой она предназначена: поддерживать оптимальную для роста растений температуру и влажность воздуха. Солнце – это очень мощная бесплатная печка, которая включается утром и выключается вечером каждый день. Она нагревает теплицу за считанные минуты. Это очень удобно, если бы не один недостаток: солнце греет очень хорошо, когда на улице и так жара, и может самопроизвольно отключаться, причем именно в холодную погоду. Вот этот недостаток и должна компенсировать конструкция теплицы. Для этого необходимо:

1) Обеспечить гарантированный отвод лишнего тепла. Это значит, что должны быть большие форточки – не менее четверти площади теплицы, и лучше в верхней части крыши, так как именно там собирается самый горячий воздух. В идеале температура воздуха внутри никогда не должна превышать 40 градусов. На самый интенсивный период работы нашей солнечной печки (июнь) лучше бы еще предусмотреть возможность притеснения – штору из легкого материала которую можно накинуть поверх теплицы. Двери в это время лучше не открывать; почему – это вопрос плодородия почвы, полива и влажности воздуха.

2) Надо сберечь тепло, когда оно нужнее всего – в холодную, пасмурную погоду. Вот тут не обойтись без автомата, который будет регулировать открытие и закрытия форточек. Причем умного автомата, который при внезапном включении «печки» мог бы быстро и широко открыть все форточки, при небольшом нагреве приоткрыть их чуть-чуть, а при похолодании – закрыть. И быть при этом весьма надежным и безотказным.

Один из методов автоматизации – это использование систем управления микроклиматом.

Для того чтобы растения хорошо принимались и приносили хороший урожай необходимо вовремя проводить полив почвы, проветривание помещения, а в холодное время проводить, подогрев воздуха в оранжерее. Производить все эти манипуляции нужно ежедневно, для этого большую часть процессов в оранжерее можно автоматизировать.

Для проведения качественных исследований необходимо точное поддержание заданных параметров микроклимата. Управление температурным режимом вручную связано с определёнными трудностями. В этом случае оператор не всегда в состоянии реагировать на все изменения регулируемых факторов, и поэтому пределы колебания температуры воздуха при ручном режиме в 5... 10 раз превышают допустимые.

В настоящее время зачастую используется морально и физически устаревшее оборудование, не говоря уже о системах автоматизации. Эти системы не обеспечивают качественное, энергоэффективное управление технологическими процессами: они обладают низкой точностью поддержания параметров, особенно при быстро меняющихся внешних воздействиях. Но даже простейшая автоматизация управления температурным режимом способна обеспечить экономию 15... 18 % тепла. Особенно эффективна автоматика в периоды переменной облачности, когда ручное управление температурным режимом весьма затруднительно.

Условия температурного режима, в которых развивается растение, оказывают огромное влияние на все процессы его жизнедеятельности:

- фотосинтез;
- дыхание;
- испарение;
- корневое питание.

Всякое отклонение от благоприятного для растений температурного режима отрицательно влияет на развитие растений. При этом нужно учесть, что растению в различные фазы его жизненного цикла требуется разная температура окружающей среды. Для нормального роста, развития и плодоношения растений необходимы влага и углекислый газ, причём в определённых соотношениях в зависимости от температуры воздуха. Сама же температурная среда должна определяться уровнем освещённости.

Таким образом, на растение оказывают влияние сразу несколько факторов среды. Учесть это влияние и создать оптимальное сочетание параметров микроклимата в оранжерее возможно лишь с помощью автоматизации технологического процесса.

Для выращивания цветов широко применяют парники, оранжереи и теплицы различной конструкции. В этом случае в процессе выращивания часто возникают трудности при поддержании требуемой температуры в сооружении. Это положение часто может усугубляться отсутствием обслуживающего персонала в течение определенного времени. При этом возможно не только замерзание

растений при снижении температур в рабочих зонах таких сооружений, но и увядание их при перегреве из-за высокого уровня солнечной радиации в оранжерее или парнике в дневные часы.

Концепция автоматизированной системы управления в оранжерее аналогична промышленной, однако существует ряд ограничений и проблем, которые отличают способы и методы поддержания микроклимата в такой теплице.

1.2 Обзор передовых решений

В целом автоматизация в среде теплиц включает автоматизацию климат-контроля (контроль температуры воздуха, циркуляции воздуха, влажность воздуха), автоматизацию контроля уровня света и управления зашториванием, автоматизацию контроля двуокиси углерода (CO_2), орошения, химической обработки и управления питательными веществами.

Алгоритм управления микроклиматом оранжереи сводится к постоянному наблюдению за фактическими параметрами и сравнению их с заданными. Управляемые параметры окружающей среды задаются или изменяются с помощью предварительно определенного алгоритма управления, запрограммированных приводов, таких как вентиляторы, кулеры, система туманообразования, система освещения и распылители CO_2. Это должно быть сделано интегрированным способом, так что, например, CO_2 не вводится, когда система вентиляции включена. Аналогичным образом, охлаждение и вентиляция должны быть тщательно синхронизированы для контроля температуры и влажности. Наиболее существенными переменными в тепличной среде, которые влияют на жизнь растения, являются свет, температура (температура воздуха и корневой зоны), влажность воздуха и прикорневой зоны [3].

В тепличном состоянии воздух вокруг растений слишком жаркий и влажный.

Как естественная, так и механическая вентиляция может использоваться в тропической оранжерее как метод снижения температуры воздуха, только если температура наружного воздуха меньше, чем внутри. Вентиляция также контролирует влажность, которая имеет решающее значение для тропической оранжереи, а именно, потребления растениями питательных веществ и воды, снижения риска заболеваний и улучшения роста растений.

Естественная вентиляция происходит при нормальном движении воздуха из-за разницы давления или температуры между пространством внутри и снаружи помещения.

Механическая вентиляция – это контролируемый обмен окружающего растения воздуха с кондиционированной атмосферой в теплице. Следовательно, вентиляция изменяет температуру и содержание CO_2, которые необходимы для хорошего роста любых культур. Если относительная влажность наружного воздуха меньше, чем внутреннего, что наблюдается в тропической оранжерее, вентиляция удаляет влажный воздух изнутри и заменяет его сухим наружным воздухом. К основным типам механической вентиляции можно отнести вытяжную и вентиляцию с

рециркуляцией воздуха.

Принцип действия механизма вытяжной вентиляции заключается в следующем: в теплице устанавливается реле, и при повышении температуры оно срабатывает и запускает вытяжной вентилятор. Он будет создавать небольшое разряжение, и высасывать из сооружения нагретый воздух, а на его место будет поступать свежий, через естественные не плотности в конструкции или же естественную вентиляцию.

При использовании вытяжного способа вентиляции целесообразно также предусмотреть приточную вентиляцию. В таком случае вентилятор будет загонять в сооружение прохладный воздух и намного быстрее понижать температуру и влажность в оранжерее.

Еще одним из эффективных способов вентиляции является рециркуляция воздуха. Правильный выбор рециркуляции уменьшает случаи болезней и улучшает активность растения. Воздух, циркулирующий в оранжерее, проходит к полу, создавая постоянный оптимальный воздушный ток. В результате этого для растений создается оптимальный климат, и в то же время растения находятся на безопасном расстоянии от вентиляторов.

Среди систем, которые получили широкое применение, можно выделить два метода увлажнения воздуха: изотермический и адиабатический.

Изотермическое увлажнение – это процесс смешения водяного пара с потоком воздуха. Изотермическое увлажнение происходит при постоянной температуре ($\Delta T=0$). В воздух непосредственно поступает насыщенный пар. Фазовый переход воды из жидкого в парообразное состояние осуществляется за счет внешних источников тепла, например, выделяемого при прохождении электрического тока через воду, содержащую определенное количество растворенных минеральных солей. Вместе с тем, при увеличении абсолютного влагосодержания, а, соответственно, и относительной влажности воздуха его температура, характеризуемая явной составляющей тепла, остается неизменной.

Изотермическое увлажнение проще реализуется аппаратно, но обладает большим энергопотреблением, что связано с необходимостью компенсации скрытой теплоты испарения воды в ходе парообразования за счет внешних источников энергии. Генерация 10 кг влаги требует 7,5 кВт*ч потребляемой энергии.

Исходя из вышесказанного можно сделать вывод, что данный метод является затратным и не пригодным для использования в оранжерее.

Адиабатическое увлажнение представляет собой процесс самого обычного испарения воды в окружающую среду. Адиабатическое увлажнение является более экономичным, как минимум на 1 – 2 порядка, поскольку процесс парообразования в этом случае происходит за счет внутреннего перераспределения энергии, а внешнее энергопотребление связано с реализацией различного рода механизмов значительно менее затратных. В наиболее совершенных системах увлажнения адиабатического типа генерация 10 кг влаги требует всего 0,04 кВт*ч потребляемой

энергии [8].

Распыление воды осуществляется при помощи сети форсунок, расположенных по всему объёму оранжереи. Количество форсунок подбирается исходя из условий перекрывания общей площади помещения.

Способы адиабатического увлажнения воздуха также разделяются на три вида:

- испарительного типа;
- распылительного типа (воздушно-водяной);
- распылительного типа (водяной).

В увлажнителях испарительного типа воздух прокачивается через панели, заполненные смачиваемой водой насадкой, в результате чего за счет пленочного испарения происходит насыщение воздуха парами воды. Известны два варианта конструктивного исполнения увлажнителей данного типа:

- с рециркуляцией воды, что характеризуется опасностью размножения бактерий и последующего распространения инфекционных заболеваний различного типа;
- без рециркуляции, что характеризуется большим расходом воды, которая только в количестве 15,30 % используется по прямому назначению, т. е. испаряется и увлажняет обрабатываемый воздух.

Недостатком увлажнителей испарительного типа является так, же отсутствие возможности регулирования количества испаряемой влаги с приемлемой точностью.

Увлажнители распылительного типа (воздушно-водяные) осуществляют распыление воды через форсунки, к которым под, водятся по отдельным трубопроводам вода и сжатый воздух. Благодаря специальной конструкции форсунок вода распыляется в виде мельчайших капель (аэрозоля) диаметром 6,8 микрон, легко абсорбируемых воздухом.

К недостаткам данного типа увлажнителей относится сравнительно большая длина свободного пробега образуемых мельчайших капель воды, распространяющихся в спутном потоке сжатого воздуха (поток сжатого воздуха, совпадающий по направлению с потоком распыляемой воды). Кроме того, данный тип увлажнителей требует наличия на объекте существующей системы сжатого воздуха либо установки компрессора необходимого напора и производительности.

Увлажнители воздуха распылительного типа водяные относятся к адиабатическим увлажнителям, в которых распыление деминерализованной воды осуществляется без использования системы сжатого воздуха. В связи с отсутствием высокоскоростного спутного потока, создаваемого в случае распылительного воздушно-водяного типа за счет истечения сжатого воздуха, данный метод характеризуются малой длиной свободного пробега распыляемых капель воды, в результате чего их габаритные раз, меры в продольном направлении оказываются даже меньше, чем у увлажнителей испарительного типа.

Что касается увлажнения почвы, то важным критерием полива в тропической оранжерее является возможность осуществления нескольких видов полива. Среди

них можно выделить капельное орошение, дождевание, аэрозольное орошение, подпочвенное орошение.

Капельное орошение – метод полива, при котором вода подаётся непосредственно в прикорневую зону выращиваемых растений регулируемыми малыми порциями с помощью дозаторов-капельниц. Данный метод позволяет получить значительную экономию воды и других ресурсов (удобрений, трудовых затрат, энергии и трубопроводов).

Капельный полив заключается в том, что орошение обеспечивается за счет медленной (капля за каплей) и длительной подачей воды в корнеобитаемую зону растений и поддержания в ней оптимальной влажности в течение всего вегетационного периода.

Вода по системе трубопроводов подается в зону орошения и через специальные водовыпуски-капельницы попадает под каждое растение или ряд растений. Одновременно с водой в почву может подаваться и раствор удобрений.

Данный метод является одним из наиболее популярных методов орошения. Основные достоинства капельного орошения:

1) точная и локализованная подача воды;

2) регулирование расхода воды позволяет не только значительно экономить силы и средства, затрачиваемые на полив, но и свести к минимуму потери питательных веществ в прикорневой зоне.

3) сохранение воздушно-водяного равновесия;

4) одновременное применение воды для полива и питательных веществ для подкормки.

Еще одним эффективным методом полива является аэрозольное орошение. Его наиболее эффективно и целесообразно использовать при дефиците водных ресурсов, а также при неблагоприятных условиях зимовки двулетних и многолетних культур.

Основные преимущества таких систем:

1) значительное сокращение потребностей в поливной воде;

2) возможность массового выращивания особо ценных пищевых и лекарственных растений, размножения селекционного материала различных культур;

3) использование в качестве защиты растений и урожая от заморозков;

4) возможность использования техники для аэрозольного орошения в других целях – дождевания, внесения подкормок и удобрений, гербицидов и тому подобное.

Одним из таких источников является кабельный обогрев теплицы, который организуется по принципу системы «теплый пол». Основным достоинством данного способа является то, что для его реализации не потребуется занимать большое пространства в теплице, так вся система прокладывается под землей, что не только экономит пространство теплицы, но и способствует наилучшему прогреванию почвы, а затем воздуха. Обогрев тепличного пространства тепловым

кабелем, с точки зрения профессиональных дачников, является намного более выгодным, так как он обеспечивает, в первую очередь прогревание почвы, что позволяет регулировать температурные режимы в соответствии с этапом развития растений [9].

Подогрев почвы в теплице с помощью электрического кабеля считается одним из самых надежных способов отопления тепличного пространства, который обладает массой неоспоримых преимуществ, особое внимание среди которых необходимо уделить:

1) Значительному увеличению диапазона культивируемых растений, в том числе и субтропические растения, крайне требовательные к температурным режимам.
2) Независимость от меняющихся температурных условий внешней среды.
3) Легкость монтажных мероприятий.

Следующий возможный метод обогрева – инфракрасный обогрев теплицы. Реализация инфракрасного обогрева в теплице позволяет существенно снизить затраты на электроэнергию и, в то же время, добиться наиболее равномерного распространения воздуха внутри обогреваемого помещения. Инфракрасные источники тепла необходимо установить по вдоль внешних стен, что позволит существенно сократить тепловые потери сооружения. Такой метод обогрева способствует нагреванию окружающих предметов, которые впоследствии передают тепловую энергию воздуху.

Основным преимуществами такого обогрева является то, что подобные обогреватели легко монтируются, переустанавливаются под разные нужды, а также потребляют относительно мало электроэнергии. При этом они не занимают рабочую площадь, поскольку монтируются на потолке или стенах. Этот вид отопления является достаточно экономичным, но при большой площади теплиц необходимо обеспечить энергопотребление мощностью 0,2 КВт/$м_2$ площади теплицы.

Вывод по разделу 1.

Для круглогодичной оранжереи необходима система, позволяющая управлять системой полива, освещения, как общего, так и индивидуального, мониторинга показаний влажности и температуры.

2 СРАВНЕНИЕ ОТЕЧЕСТВЕННЫХ И ПЕРЕДОВЫХ ЗАРУБЕЖНЫХ ТЕХНОЛОГИЙ И РЕШЕНИЙ

При постройки современных оранжерей или круглогодичных теплиц используются автоматические системы управления. Это позволяет минимизировать человеческий фактор при содержании всего комплекса, а также значительно сократить затраты на поддержание высокой урожайности. Это достигается путем сокращения человеческого труда, а также созданием более благоприятного микроклимата в комплексе, путем автоматизации технологического процесса

На данный момент в мире представлено большое количество различных комплексов для автоматизации оранжерей. Лидерами в этой области являются Голландия, Израиль и Корея.

В большинстве случаев оранжереи разрабатываются большими компаниями для строго определенной местности.

В нашем же случае предлагаем выстроить гибкую систему, способную работать как в малых оранжереях и теплицах, так и в крупных комплексах. Это планируется достичь путем модульности, то есть выстроить систему управления, к которой возможно подключать различные модули. Также лидирующие страны находятся совершенно в другом климате, нежели Россия, именно поэтому на необходимо выстроить свою автоматизированную систему управления оранжереей.

Система выполняет следующие основные функции: для того что бы обеспечить определенный климат парниковой среды, температура и влажность воздуха почвы в теплице поддерживаются на постоянном уровне с помощью автоматических установок искусственного климата. Вентиляция и дополнительное освещение включаются также автоматически, обеспечивая растениям оптимальный световой режим и чистоту воздуха. К основным функциональным структурам управления можно отнести:

1) Управление и мониторинг системы проводится через веб-интерфейс.

2) Ввод данных с центрального контроллера.

3) Сбор, обработка и представление информации.

4) Создание базы данных об истории технологического процесса и представление их в удобных для анализа формах (текст, графики, гистограммы и так далее).

Применение системы обеспечивает:

1) Повышение производительности теплицы за счет жесткого автоматического поддержания требуемых параметров микроклимата.

2) Снижение энергопотребления.

3) Повышение уровня надежности и эффективности работы оборудования.

С помощью компьютера и телефона с выходом в интернет можно включать или отключать разные автоматизированные системы. Например, можно поменять климат в теплице, включить освещение, включить орошение и прочее. Вместе с развитием новых технологий развиваются и услуги широкополосной связи.

Достижения в спутниковых технологиях дает возможность в широком охвате крупных областей, что обеспечивает установления относительно недорогих соединений с помощью GSM GPRS Internet соединений. Эти технологии могут помочь в решении, связанных с управлением автоматизированных систем [3].

Вывод по разделу 2

Система будет управляться с персонального компьютера, подключенная через локальную сеть к системе. Датчики и механизмы, позволяющие управлять микроклиматом оранжереи будут подключены через Wi-Fi.

3 ВЫБОР ТЕХНОЛОГИЙ ФУНКЦИОНИРОВАНИЯ РАЗРАБОТКИ И СРЕДЫ РАЗРАБОТКИ

Исходя из изученного материала нам необходимы такие технологии, которые могут управлять определёнными механизмами при помощи компьютера.

3.1 Язык программирования для разработки приложения

Для написания программ вполне подходит язык программирования C#, язык разработки приложений для платформы Microsoft .NET Framework, так как в наше время доля операционной системы Windows установлена на 86 % компьютеров на планете.

На сегодняшний момент язык программирования C# один из самых мощных, быстро развивающихся и востребованных языков в ИТ-отрасли. В настоящий момент на нем пишутся самые различные приложения: от небольших десктопных программок до крупных веб-порталов и веб-сервисов, обслуживающих ежедневно миллионы пользователей.

C# является объектно-ориентированным и в этом плане много перенял у Java и C++. Например, C# поддерживает полиморфизм, наследование, перегрузку операторов, статическую типизацию. Объектно-ориентированный подход позволяет решить задачи по построению крупных, но в тоже время гибких, масштабируемых и расширяемых приложений. И C# продолжает активно развиваться, и с каждой новой версией появляется все больше интересных функциональностей, как, например, лямбды, динамическое связывание, асинхронные методы и т.д [4].

3.2 Платформа для аппаратной части проекта для непосредственного управления оранжереей

Для управления механизмами выбран микроконтроллер WeMos D1, который построен на платформе Arduino. Его программирование происходит в Arduino IDE.

Плата WeMos D1, которая производится в Китае, выполнена на основе WiFi модуля ESP8266 ESP-12. На модуле имеется разъем под внешнюю WiFi антенну – благодаря этому можно расширить площадь покрытия сетью. Программирование платы осуществляется с помощью стандартной среды разработки Arduino IDE. Контроллер включает в себя процессор, периферию, оперативную память и устройства ввода/вывода. Наиболее часто микроконтроллеры применяются в компьютерной технике, бытовых приборах и других электронных устройствах. WeMos отличается дешевой стоимостью и простотой подключения и программирования.

Основными областями применения контроллеров WeMos являются температурные датчики, датчики давления и другие, зарядные устройства, пульты для управления различными бытовыми приборами, системы обработки данных, робототехника. К микроконтроллеру можно подключать дополнительные компоненты: индикаторы, сенсоры, светодиоды, которые позволяют реализовывать различные проекты и расширять их возможности.

3.3 Связь компьютера с микроконтроллером

Связь компьютера с микроконтроллером будет осуществляться по протоколу MQTT.

MQTT или Message Queue Telemetry Transport – это легкий, компактный и открытый протокол обмена данными созданный для передачи данных на удалённых локациях, где требуется небольшой размер кода и есть ограничения по пропускной способности канала. Вышеперечисленные достоинства позволяют применять его в системах M2M (Машинно-Машинное взаимодействие) и IoT (Промышленный Интернет вещей) [7].

Система связи, построенная на MQTT, состоит из сервера-издателя, сервера-брокера и одного или нескольких клиентов. Издатель не требует каких-либо настроек по количеству или расположению подписчиков, получающих сообщения. Кроме того, подписчикам не требуется настройка на конкретного издателя. В системе может быть несколько брокеров, распространяющих сообщения.

MQTT предоставляет способ создания иерархии каналов связи своего рода ветвь с листьями. Всякий раз, когда у издателя есть новые данные для распространения среди клиентов, сообщение сопровождается примечанием контроля доставки. Клиенты более высокого уровня могут получать каждое сообщение, в то время как клиенты более низкого уровня могут получать сообщения, относящиеся только к одному или двум базовым каналам, «ответвляющимся» в нижней части иерархии. Это облегчает обмен информацией размером от двух байт до 256 мегабайт.

Согласно IBM, MQTT обладает следующими свойствами:

- Нейтрален к содержимому сообщения.
- Идеально подходит для распределённых коммуникаций «один ко многим» и разъединённых приложений.
- Оснащён функцией LWT (Last Will and Testament, «последняя воля и завещание») для уведомления сторон об аномальном отключении клиента.
- Полагается на TCP/IP для базовых задач связи.
- Разработан для доставки сообщений по шаблонам «максимум один раз», «минимум один раз» и «ровно один раз».

Участник системы MQTT может взять на себя роль издателя, потребителя или обе роли сразу.

Поскольку приложения IoT ныне внедряются в огромных масштабах, MQTT попал в центр внимания как открытый, простой и масштабируемый способ развёртывания распределённых вычислений и функциональности IoT для более широкой пользовательской базы – как на потребительском, так и на промышленном рынках [2].

Как указано выше, MQTT – легковесный протокол обмена сообщениями, построенный для ненадёжных сетей и устройств с ограничениями на источник питания и CPU. MQTT предоставляет различные уровни обслуживания для различных типов инфраструктуры IoT, от повторяющейся выборки данных до

управления промышленными машинами.

Данные датчиков окружающей среды: как уже упоминалось, MQTT поддерживает модель доставки сообщений «не более одного раза». В сетях с частичным покрытием территории или высокой задержкой это означает, что информация может быть потеряна или дублироваться. В областях, где удалённые датчики записывают и передают данные с заданными интервалами, это не является проблемой, так как новые показания поступают на регулярной основе. Датчики в удалённых средах – обычно маломощные устройства, что делает MQTT идеальным решением для сенсоров IoT с относительно низким приоритетом передачи данных.

Данные о работоспособности машин: для быстрого реагирования на возникающие проблемы и предотвращения простоев. Например, для ветроэлектрической установки нужна гарантированная доставка текущих показателей о работоспособности местным командам ещё до того, как эта информация попадёт в центр обработки данных. В таких ситуациях доставка сообщений «по крайней мере один раз» гарантирует, что соответствующие флаги своевременно заметят нужные специалисты, даже если они поступают как дубликаты. Это важно для межмашинной связи с более высоким приоритетом.

Биллинговые системы: есть ещё более приоритетные и точные сообщения, которые требуется правильно обрабатывать. В бизнес-ситуациях, где дублирование записей неприемлемо, в том числе в биллинговых системах, пригодится флаг QoS передачи «точно один раз». Это устраняет дублирование или потерю пакетов в системах выставления счетов или биллинга, сокращает количество аномалий и ненужных противоречий по согласованию [2].

4 ОПИСАНИЯ ФУНКЦИОНИРОВАНИЕ РАЗРАБОТКИ

Согласно задания дипломного проекта была написана программа, которая позволяет выполнять управление микроклиматом оранжереи.

4.1 Общий принцип функционирования системы

Управление микроклиматом будет с автоматизированного рабочего места, на котором установлена программа для управления.

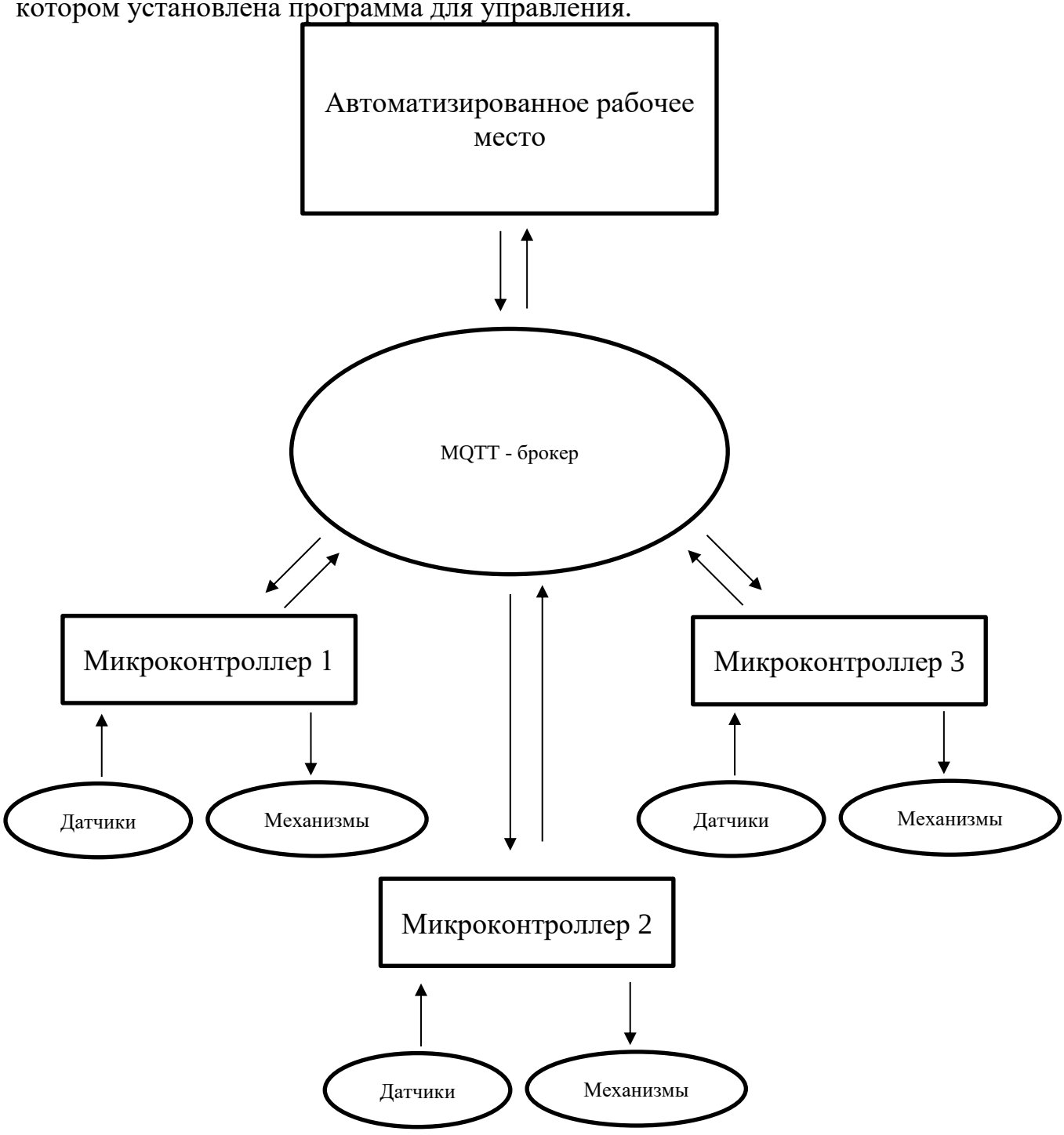

Рисунок 1 – общая схема функционирования системы

На схеме изображено (рисунок 1) взаимодействие системы с получаемыми данными. Данные формируются на датчиках, с которых микроконтроллер считывает различные параметры (температуру воздуха, влажность и так далее) Далее он эти данные формирует в пакет, который закладывает информацию о месте получения и сами параметры.

После этого микроконтроллер отправляет пакет на MQTT – брокер в определенный топик, специально предназначенный для этих данных. MQTT – брокер аккумулирует в себе пакеты приходящие от различных микроконтроллеров. Программа, установленная на автоматизированном рабочем месте, считывает информацию с MQTT – брокера и отображает в соответствующих показаниях.

Также пользователь может управлять механизмами в оранжереи для основываясь на показаниях, приходящих с датчиков.

Пользователь задает определенную команду для определенного механизма. Программа формирует пакет, в который входит информация, какому механизму предназначается команда и соответственно сама команда.

Пакет, сформированный программой, отправляется в MQTT – брокер на соответствующий брокер, где его считывает соответствующий микроконтроллер. Сам микроконтроллер, основываясь на полученном пакете, передает команду соответствующему механизму.

4.2 Обзор пользовательской части программы

Программа для управления оранжереей представляет из себя 3 окна:

- главное окно;
- окно управления грядками;
- окно управления микроклиматом.

На главном окне расположены кнопки для доступа в другие окна, такие как «Управление грядками», «Управление микроклиматом», а также кнопка закрытия программы (рисунок 2).

Также на главном окне отображаются показания температуры воздуха в помещении оранжереи, влажность воздуха и температура окружающей среды.

В правом верхнем окне отображается системная дата и время.

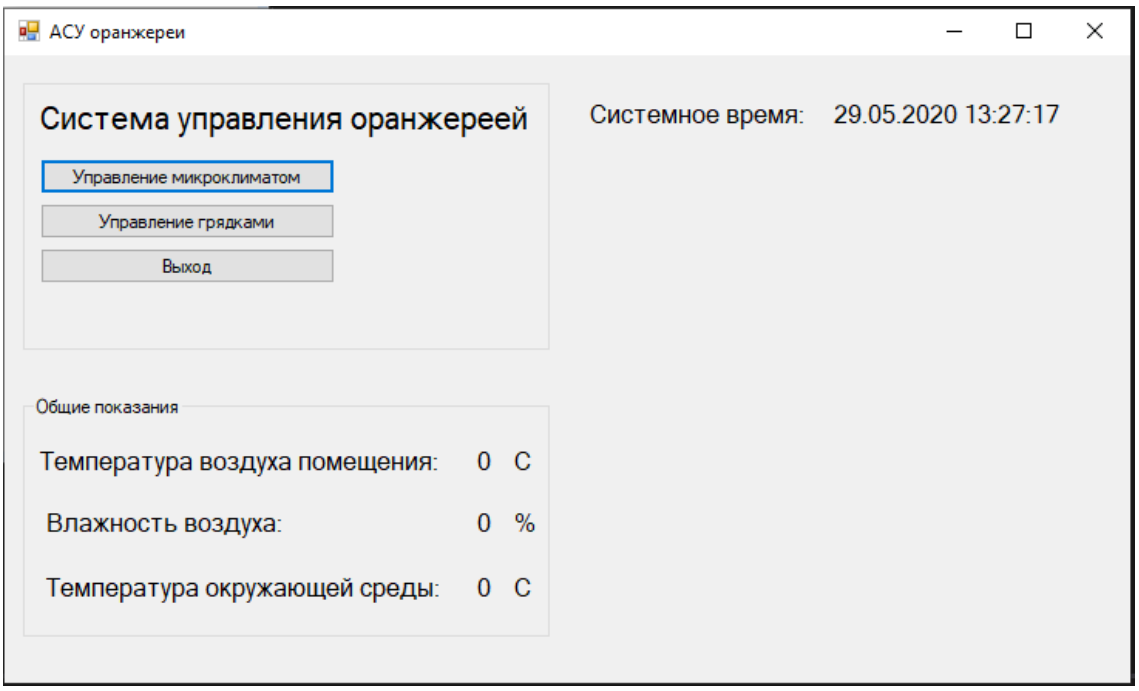

Рисунок 2 – Главное окно

Управление грядками производится через форму «Управление грядками».

В форме находятся следующие элементы:

- выпадающий список номера грядки;
- текстовые поля «Культура» и «Сорт»;
- кнопки изменения и удаления записи;
- кнопка для открытия панели «Новая запись»;
- текстовые поля «Культура» и «Сорт» для новой записи грядки;
- кнопка добавления новой записи.

Форма позволяет удалять, изменять, а также создавать грядки. Также она позволяет получить информацию о грядках (культуру, посаженную в грядку, и сорт культуры). В правом верхнем углу указана системная дата и время (рисунок 3).

Вся информация о растениях, посаженных в конкретную грядку, хранится в файле «beds.txt». При удалении грядки в файл на поля «Культура» и «Сорт» записывается слово «Пусто».

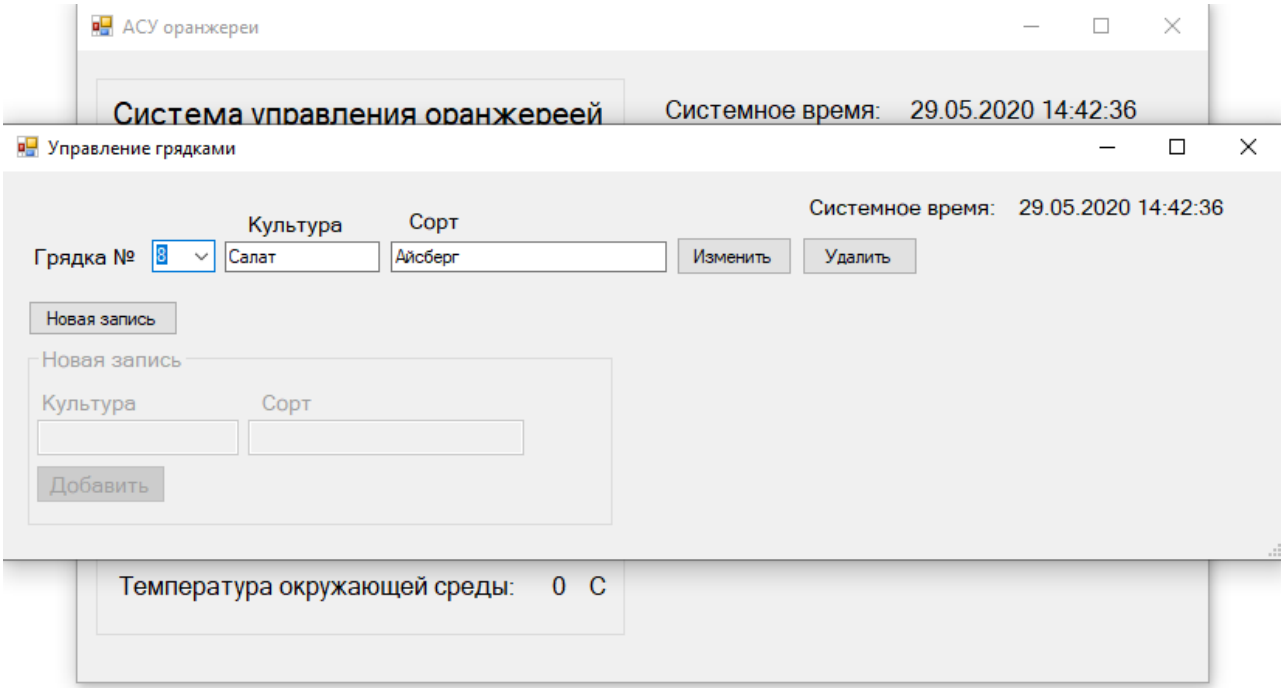

Рисунок 3 – Форма «Управление грядками»

Управление микроклиматом происходит через форму «Управление микроклиматом» (рисунок 4).

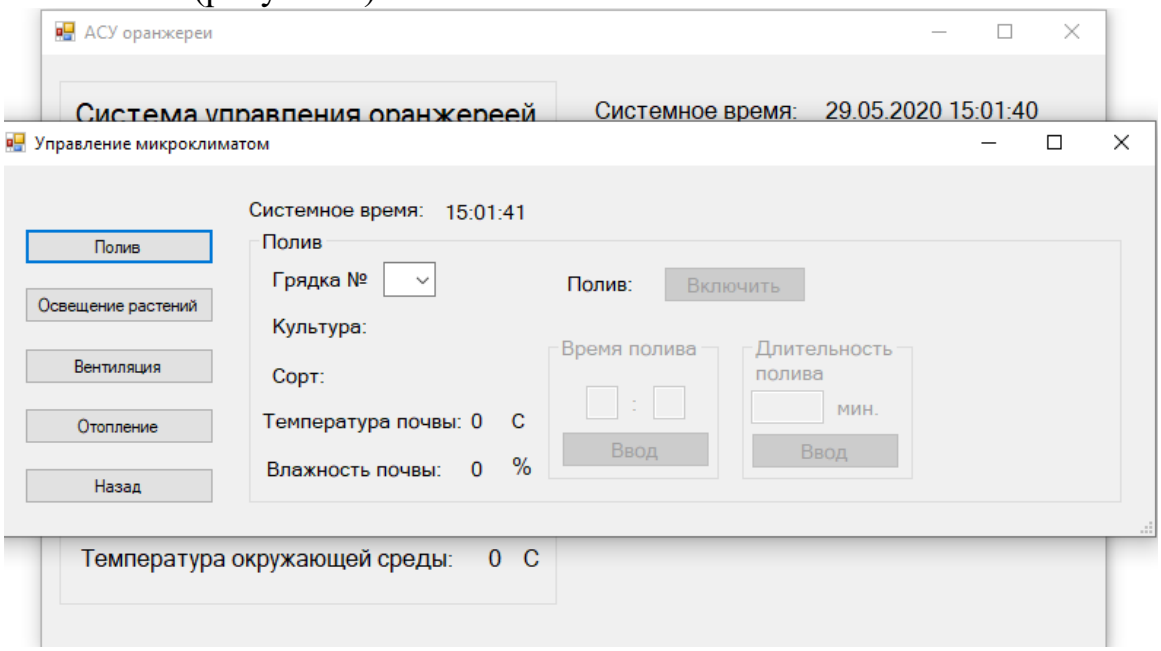

Рисунок 4 – Форма «Управление микроклиматом»

В форме находятся следующие элементы:

- кнопки переключения панелей полива, освещения, вентиляции, отопления и кнопка закрытия формы;
- системное время;
- панель «Полив»;
- панель «Освещение растений»;

- панель «Вентиляция»;
- панель «Отопление».

Каждая кнопка активирует соответствующую панель и отображает ее в форме. При открытии формы по умолчанию отображается панель «Полив». Все остальные панели находятся в форме с самого начала. Свойство Enabled каждой из них равно false, а расположение в форме находится в точке 1000x1000 пикселей, тем самым не ее не видно пользователю. При нажатии соответсвующей кнопки свойство Enabled переходит в true для конкретной панели, положение ее на форме ствановится 176x43 пикселя. Ненужные панели переходят в режим «Enabled = false». Рассмотрим панели подробнее.

Панель «Полив».

На панели «Полив» отображены выпадающий список выбора грядки. При выборе на панель выводится информация о культуре и сорте, которые посажены в соответствующую грядку.

Также при выборе грядки активируются панели времени полива, длительности полива и кнопка включения\выключения принудительного полива (рисунок 5).

В левом нижнем углу указаны следующие показания:

- температура почвы;
- влажность почвы.

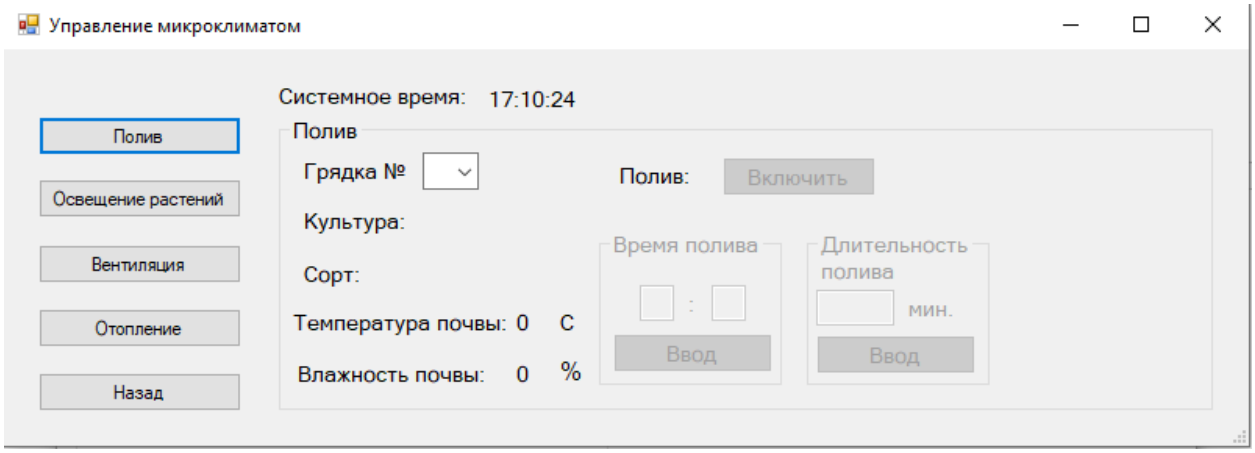

Рисунок 5 – Панель «Полив»

На этой панели можно указать время и длительность полива, а также включить полив принудительно. Время и длительность записывается в файл «timeWatering.txt».

Программа каждую секунду сканирует этот файл, и при совпадении системного времени со временем в фале включает полив. После этого отключает через указанное время.

Панель «Освещение растений».

При выборе на панель выводится информация о культуре и сорте, которые посажены в соответствующую грядку, кнопка включения\выключения общего освещения, панели времени включения и выключения индивидуального освещения грядок (рисунок 6).

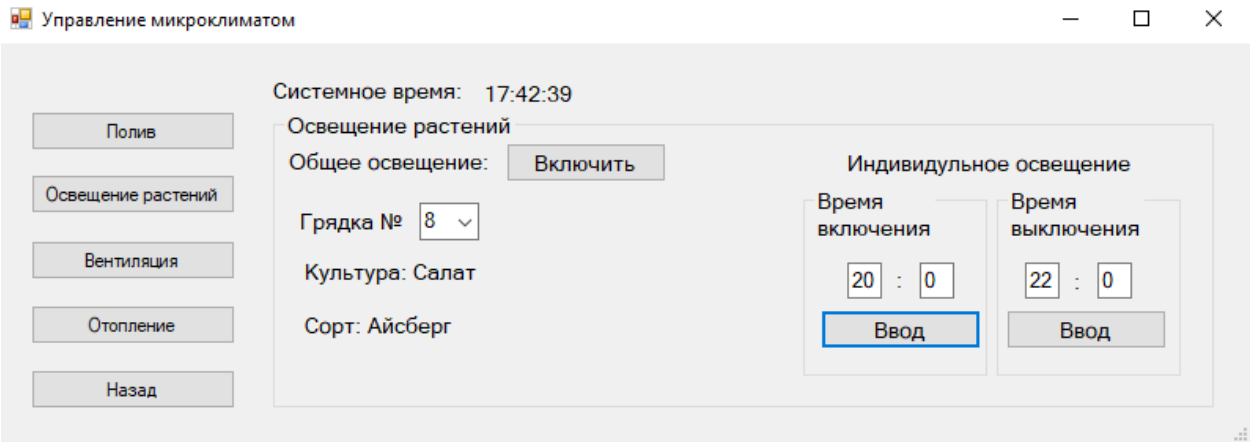

Рисунок 6 – Панель «Освещение растений»

Как только выбрана нужная грядка, то сразу выводится информация о культуре и сорте растений, посаженых в соответствующую грядку, а также активируются панели времени включения и выключения индивидуального освещения грядок, и на них сразу же отображается время включения и выключения индивидуального освещения конкретной грядки.

При изменении времени включения\выключения и подтверждения кнопкой «Ввод», изменения записываются в файл «timeLighting.txt». Программа каждую секунду сканирует файл, и при совпадении времени включения\выключения, включает либо выключает индивидуальное освещение.

Панель «Вентиляция».

На панели «Вентиляция» расположены кнопки, отвечающие за включение\выключение циркуляции воздуха внутри помещения, приточной вентиляции, вытяжной вентиляции и естественной вентиляции.

Также на панели расположены индикаторы включения\выключения вентиляции. при нажатии на кнопку цвет меняется на соответствующий (зеленый – вентиляция включена, красный – вентиляция выключена).

В нижней части панели отображаются параметры влажности воздуха в помещении (рисунок 7).

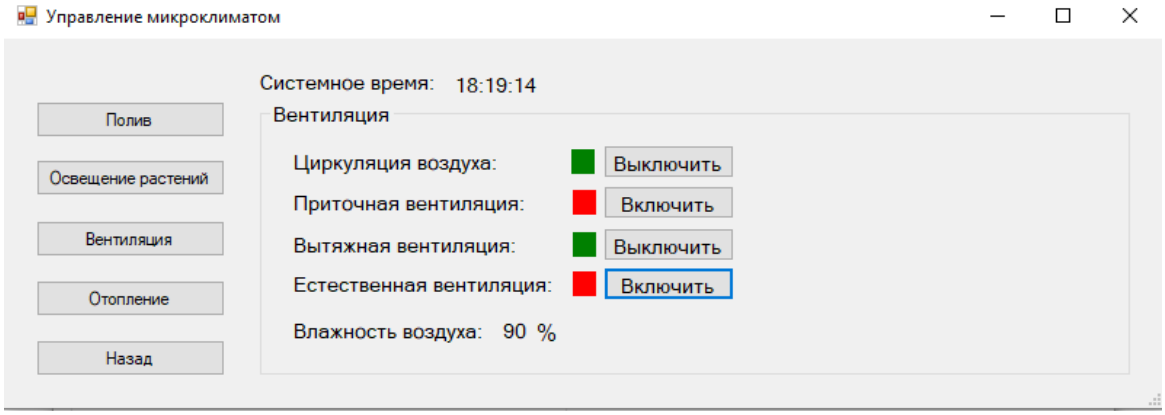

Рисунок 7 – Панель «Вентиляция»

Вся информация о включении\выключении вентиляции, а также состоянии влажности воздуха сохраняется в файл «controlAir.txt». При отключении программы файл хранит в себе текущую информацию о состоянии вентиляции. При повторном включении программа считывает файл и отображает текущее состояние вентиляции.

Также в файл «controlAir.txt» хранит в себе информацию о влажности воздуха в оранжерее. Программа сканирует этот файл каждую секунду и отображает изменение влажности.

Панель «Отопление»

На панели «Отопление» расположены два ползунка, отвечающих за настенные радиаторы и калориферы вентиляторов. При включении панели на ней отображается положение ползунков в процентном соотношении, а соответственно и процент открытия вентилей отопления на теплообменниках (рисунок 8).

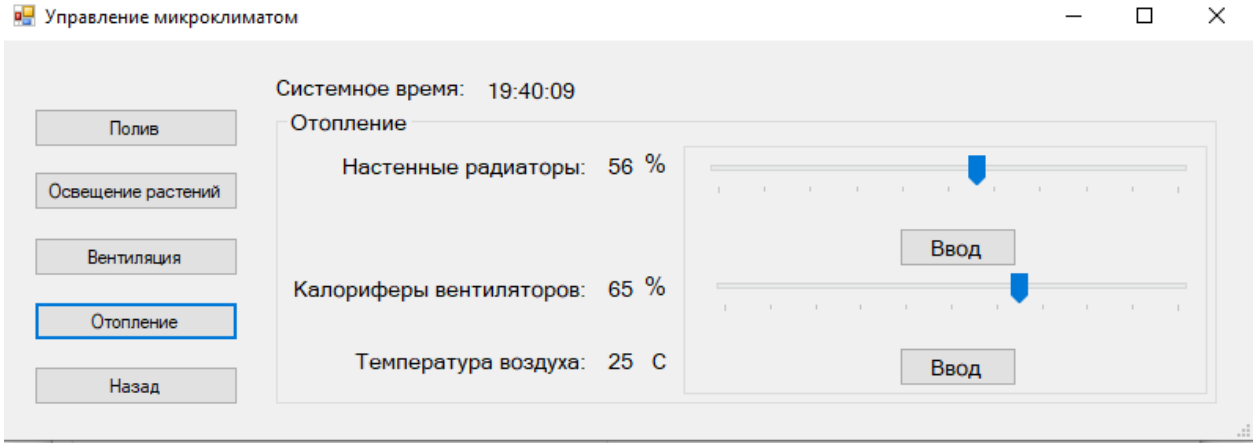

Рисунок 8 – Панель «Отопление»

Для установки нового положения сервоприводов на вентилях необходимо перетащить ползунок в нужное положение и подтвердить действие нажав кнопку «Ввод».

В левом нижнем углу также отображается информация о температуре в оранжерее.

Вся информация о текущем положении сервоприводов вентилей и температуры в оранжерее находится в файле «controlAir».

Вывод по разделу 4

Разработанная программа позволяет управлять поливом, задавать время и продолжительность автоматического полива, управлять освещением, как общим, так и индивидуальным для каждой грядки, установкой времени включения и выключения освещения, управлением всеми видами вентиляции и отопления.

ЗАКЛЮЧЕНИЕ

Повышение эффективности эксплуатации тепличных комплексов является важной проблемой для многих владельцев парников. Оптимальным решением поставленной задачей являются системы теплиц, помогающие автоматизировать определенные процессы, относящиеся к агротехническому комплексу по уходу за растениями.

При достижении цели, поставленной перед выпускной квалификационной работой, были выполнены следующие задачи:

1) Была изучена предметная область. Были определены основные параметры функционирования системы. Проведена актуализация данной разработки.
2) Сформированы общие принципы функционирования системы управления микроклиматом оранжереи. Определено то, как будет работать система, что будет выполнять пользователь для оправления микроклиматом.
3) Выбраны технологии, по которым будет троиться система управления микроклиматом. Рассмотрены их преимущества перед другими платформами.
4) Описана работа приложения. Показан интерфейс пользователя, и описаны элементы интерфейса.
5) Показан листинг написанной программы. Также даны комментарии по отдельным блокам кода.

В результате проделанной работы была построена система, позволяющая получить всю необходимую информацию об климатических параметрах. Система должна выводить данные мониторинга климатических параметров на рабочий компьютер.

БИБЛИОРГАФИЧЕСКИЙ СПИСОК

1 Ватсон, Б., C# 4.0 на примерах / Б. Ватсон. – С-Петербург: БХВ-Петербург, 2015.

2 Гастон, К., Практическое программирование MQTT / К. Гастон. UK, Birmingham: Packt Publishing Ltd, 2018.

3 Лабор, В.В., C# - Создание приложений для Windows / В.В. Лабор. – М.: Харвест, 2016.

4 Олифер, В. Безопасность компьютерных систем / В. Олифер. – С-Петербург: Телеком, 2014.

5 Петин, В.А. Проекты с использованием контроллера Arduino / В.А. Петин. М.: BHV, 2019.

6 Исследование и разработка системы поддержания жизнеобеспечения оранжереи. – http://masters.donntu.org/2018/fkita/kolomoec/library/index.htm

7 Протокол MQTT: концептуальное погружение – https://habr.com/ru/post/463669/

8 Сравнительный анализ систем адиабатического увлажнения воздуха. – https://www.c-o-k.ru/articles/sravnitel-nyy-analiz-sistem-adiabaticheskogo-uvlazhneniya-vozduha

9 Теплопотери помещений – https://studfile.net/preview/1743665/page:8/

ПРИЛОЖЕНИЕ А

Расшифровка кодов команд, приходящий в MQTT-брокер

Таблица А1 – Расшифровка кодов команд для полива и освещения грядок в топике «/грядки/*»

Отправитель	Грядка	Адресат команды	Команда
Ардуино – 1 Приложение – 0	№ грядки	Полив – 0 Индивидуальное освещение – 1	Включить – 1 Выключить – 0

Таблица А2 – Расшифровка кодов команд в топике «/общееОборудование»

	Включить	Выключить
Циркуляция воздуха	11	10
Приточная вентиляция	21	20
Вытяжная вентиляция	31	30
Естественная вентиляция	41	40
Общее освещение	51	50
Настенные радиаторы	600 + (положение серво)	
Калориферы вентиляторов	700 + (положение серво)	

Таблица А3 – Расшифровка кодов команд для топика «/сенсор»

Тип сенсора	Отправитель	Грядка	Цифровое обозначение сенсора	Показания
Влажность почвы	Ардуино – 1	№ грядки	0	Показания с датчиков
Температура почвы	Ардуино - 1	№ грядки	1	
Влажность воздуха	Ардуино - 1	0	2	
температура воздуха	Ардуино - 1	0	3	
Температура окружающей среды	Ардуино - 1	0	4	

ПРИЛОЖЕНИЕ Б

Файлы, используемые в проекте

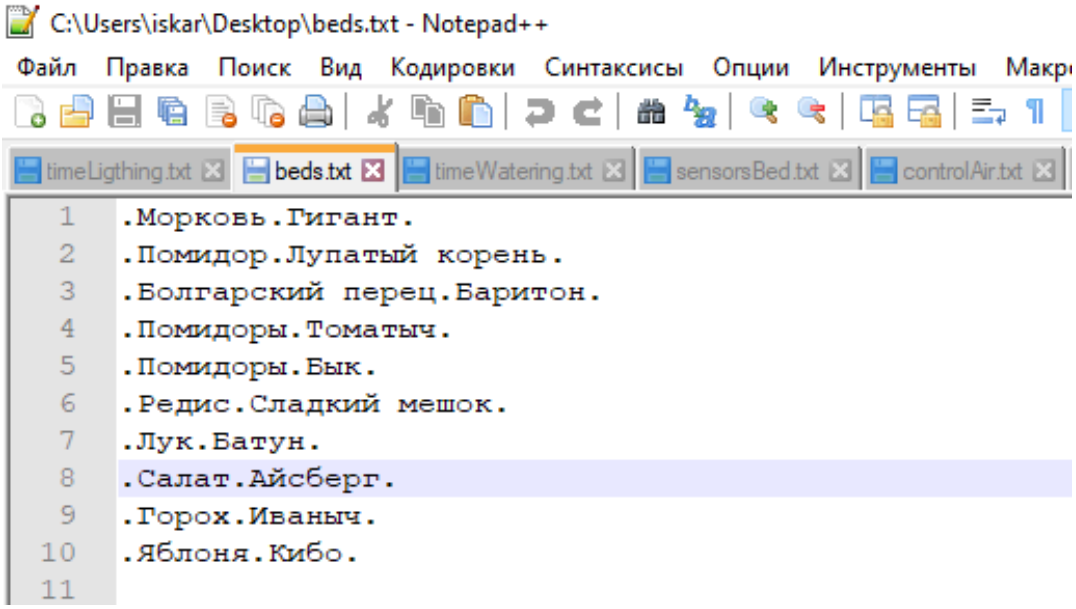

Рисунок Б1 – Файл «beds.txt»

Файл «beds.txt» используется для хранения информации о культуре и сорте растений посаженных в конкретную грядку.

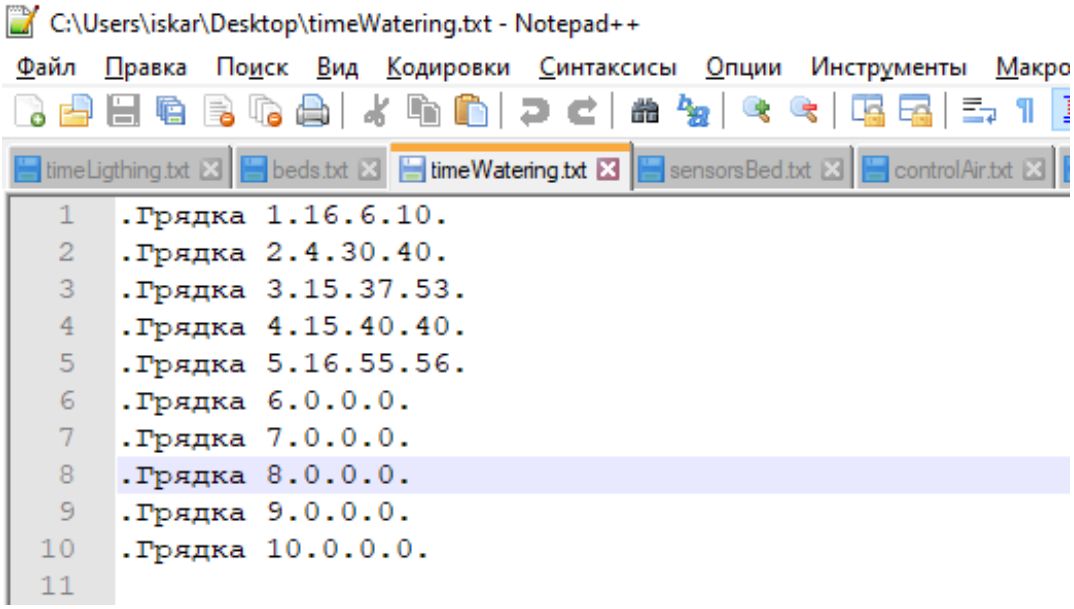

Рисунок Б2 – Файл «timeWatering.txt»

В файле «timeWatering.txt» содержится информация о времени полива. В частности данные о времени включения для конкретной грядки и продолжительности полива. Точками разграничиваются слова. Первое слово

отвечает за номер грядки. Второе и третье отвечает за время включения полива. Четвертое слово отвечает за продолжительность полива.

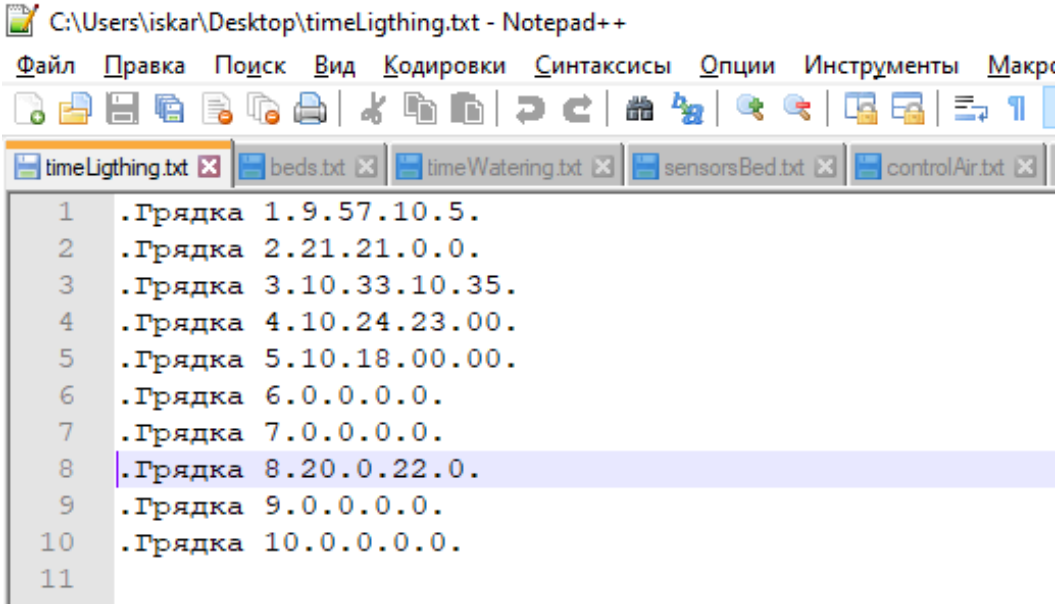

Рисунок Б3 – Файл «timeLighting.txt»

В файле «timeLighting.txt» содержится информация о времени включения и выключения индивидуального освещения. Первое слово отвечает за номер грядки. Второе и третье отвечает за время включения индивидуального освещения. Четвертое и пятое отвечает за время выключения индивидуального освещения

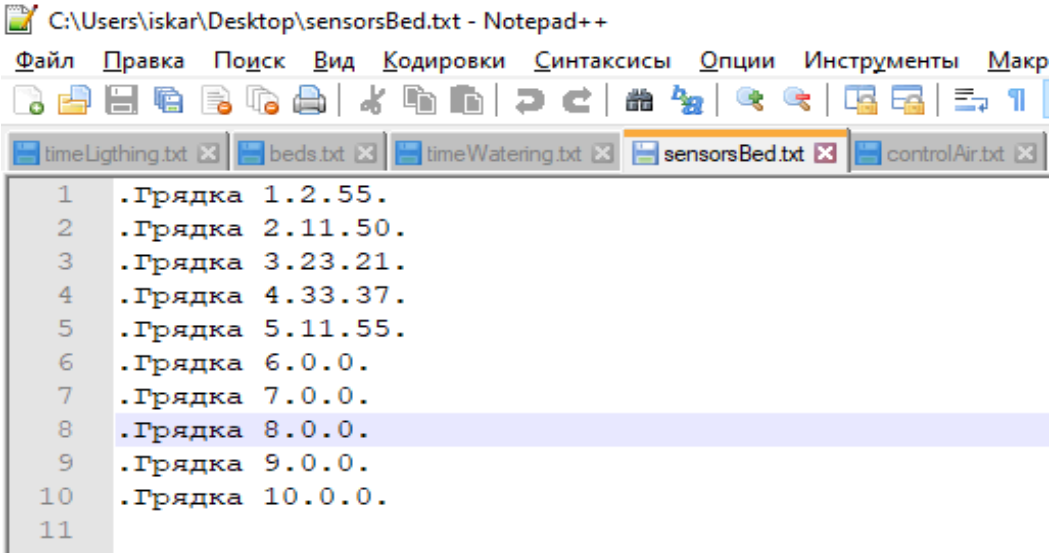

Рисунок Б4 – Файл «sensorsBed.txt»

В файле «sensorsBed.txt» содержится информация о температуре и влажности почвы грядок.

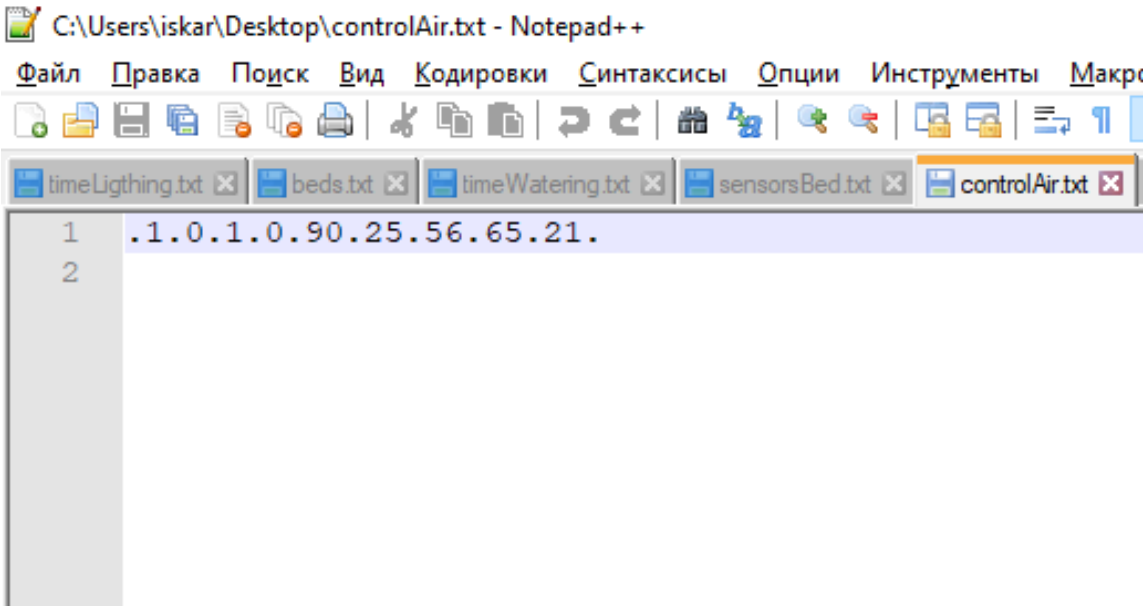

Рисунок Б5 – Файл «controlAir.txt»

В файле «controlAir.txt» находится информация о состоянии вентиляции, отопления и параметрах воздуха.

Первое число – состояние циркуляционных вентиляторов.

Второе число – состояние приточных вентиляторов.

Третье число – состояние вытяжных вентиляторов.

Четвертое число – состояние естественной вентиляции.

Пятое число – влажность воздуха.

Шестое число – температура воздуха.

Седьмое число – степень открытия отопления на радиаторы.

Восьмое число – степень открытия отопления на калориферы.

Девятое число – температура окружающей среды.

ПРИЛОЖЕНИЕ В

Листинг программ

Главная форма.

Используемые библиотеки:
Пространство имен, содержащее фундаментальные и базовые классы:
using System;

Пространство имен, содержащее типы, позволяющие осуществлять чтение и запись в файлы и потоки данных:
using System.IO;

Пространство имен, содержащее классы, которые представляют кодировки ASCII и Юникода:
using System.Text;

Пространство имен, содержащее классы для создания приложений Windows:
using System.Windows.Forms;

Пространство имен, содержащее классы для создания соединения MQTT:
using uPLibrary.Networking.M2Mqtt;

Пространство имен, содержащее классы для отправки сообщений по протоколу MQTT:
using uPLibrary.Networking.M2Mqtt.Messages;

Код программы

```
public partial class Form1 : Form
{

    string y = "00";
// путь журналу вентиляции
    string pathAir = @"C:\Users\iskar\Desktop\controlAir.txt";
    public void Client_recievedMessage(object sender, MqttMsgPublishEventArgs e)
    {
        string message = Encoding.Default.GetString(e.Message);
        y = message;
    }

    public Form1()
    {
        InitializeComponent();
        MqttClient mqttClient = new MqttClient("localhost");

        mqttClient.MqttMsgPublishReceived += Client_recievedMessage;
```

```
            string clientId = Guid.NewGuid().ToString();

            mqttClient.Connect(clientId, "KuSJIbIu", "ahohud44");

          mqttClient.Subscribe(new String[] { "/temperature" }, new byte[] {

            MqttMsgBase.QOS_LEVEL_AT_LEAST_ONCE });
        }
// кнопка закрытия программы
        private void quit_Click(object sender, EventArgs e)
        {


Close();
        }
// метод системного времени
        private void timer1_Tick(object sender, EventArgs e)
        {
            DateTime now = new DateTime();
            now = DateTime.Now;
            int hour = now.Hour;
            time.Text = now.ToString();
            time.Refresh();
        }
//метод вывода показаний сенсоров воздуха
        private void timer2_Tick(object sender, EventArgs e)
        {
            using (StreamReader stream = new StreamReader(pathAir))
            {
                string strSplit = stream.ReadLine();
                stream.Dispose();
                string[] word = strSplit.Split('.');
                humidity.Text = word[5];
                temperature.Text = word[6];
                temperatureAmbient.Text = word[9];
            }
            temperatureAmbient.Refresh();
            temperature.Refresh();
            humidity.Refresh();
        }
```

Таймер раз в секунду проверяет файл с показаниями сенсоров воздуха и

выводит их на панель

```
// кнопка открытия формы управления микроклиматом
        private void Control_Click(object sender, EventArgs e)
        {
            Form_control form_Control = new Form_control();
            form_Control.ShowDialog();
            Form1 form1 = new Form1();
            form1.Enabled = false;
        }
// кнопка управления грядками
        private void button1_Click(object sender, EventArgs e)
        {
            Control_beds control_Beds = new Control_beds();
            control_beds.ShowDialog();
        }
          }
```

Форма «Управление грядками».
Используемые библиотеки:
Пространство имен, содержащее фундаментальные и базовые классы:
using System;

Пространство имен, содержащее типы, позволяющие осуществлять чтение и запись в файлы и потоки данных:
using System.IO;

Пространство имен, содержащее классы для создания приложений Windows:
using System.Windows.Forms;

Код программы

```
public partial class Control_beds : Form
{
        // путь к файлу с данными грядок
        string path = @"C:\Users\iskar\Desktop\beds.txt";

        // путь к файлу с данными полива
        string pathWatering = @"C:\Users\iskar\Desktop\timeWatering.txt";

        // путь к файлу с данными освещения
        string pathLighting = @"C:\Users\iskar\Desktop\timeLigthing.txt";

        // путь к файлу с данными сенсоров
        string pathSensor = @"C:\Users\iskar\Desktop\sensorsBed.txt";

        string line;
      public Control_beds()
      {
          InitializeComponent();
          int i = 0;

          using (StreamReader sr = new StreamReader(path))
          {
              line = sr.ReadLine();
              while (line != null)
              {
                  if (line != "")
                  {
                      numberBeds.Items.Add(i + 1);
                      line = sr.ReadLine();
```

```
    }
                else
                {
                    line = sr.ReadLine();
                }
                i++;
            }
        }
        string[] lines = File.ReadAllLines(path);
        using (StreamWriter writer = new StreamWriter(path))
        {
            foreach (string line in lines)
                if (!string.IsNullOrWhiteSpace(line))
                    writer.WriteLine(line);
     }
      }

// системное время
        private void timer1_Tick(object sender, EventArgs e)
        {
            DateTime now = new DateTime();
            now = DateTime.Now;
            int hour = now.Hour;
            timer.Text = now.ToString();
            timer.Refresh();
        }

// активация панели новой записи
        private void newEntry_Click(object sender, EventArgs e)
        {
            if (groupNewEntry.Enabled == false) groupNewEntry.Enabled = true;
            else groupNewEntry.Enabled = false;
        }

// кнопка добавления новой записи
        private void addEntry_Click(object sender, EventArgs e)
            int i = 0;
            using (StreamWriter stream = new StreamWriter(path, true))
            {
                stream.WriteLine("." + plant_new.Text + "." + plant_variety_new.Text + ".");
            }
```

```
        using (StreamReader streamReader = new StreamReader(File.Open(path,
FileMode.Open)))
            {
                while (!streamReader.EndOfStream)
                {
                    streamReader.ReadLine();
                    i++;
                }
            }
            removeEmptyline(pathLighting, ".0.0.0.0.", i);
            removeEmptyline(pathSensor, ".0.0.", i);
            removeEmptyline(pathWatering, ".0.0.0.", i);
            Close();
        }
```

При создании новой записи о грядке также создаются новые записи о поливе этой новой грядке и индивидуальном освещении.

```
// метод добавления новых записей в дочерних файлах
        void removeEmptyline(string pathRemove, string val, int x)
        {
            string[] lines = File.ReadAllLines(pathRemove);
            using (StreamWriter writer = new StreamWriter(pathRemove))
            {
                foreach (string line in lines)
                    if (!string.IsNullOrWhiteSpace(line))
                        writer.WriteLine(line);
            }
            using (StreamWriter stream = new StreamWriter(pathRemove, true))
            {
           stream.WriteLine(".Грядка " + x + val);
            }
        }

// выбор грядки
        private void numberBeds_SelectedIndexChanged(object sender, EventArgs e)
        {
            int i = numberBeds.SelectedIndex;
            using (StreamReader stream = new StreamReader(path))
            {
                for (int ii = 0; ii < i; ii++)
                {
```

```
            stream.ReadLine();
            }
            string str = stream.ReadLine();
            stream.Dispose();
            string[] word = str.Split('.');
            plant.Text = word[1];
            plant_variety.Text = word[2];
        }
    }
```

При выборе грядки программа проверяет файл с данными о каждой из них, по индексу выбранной находит ее и выводит на панель

```
// кнопка изменения записи
    private void replace_Click(object sender, EventArgs e)
    {
        stringBeds(numberBeds.SelectedIndex, plant.Text, plant_variety.Text);
    }

// кнопка отчистки записи
    private void delete_beds_Click(object sender, EventArgs e)
    {
        stringBeds(numberBeds.SelectedIndex, "Пусто", "Пусто");
        plant.Text = "Пусто";
        plant_variety.Text = "Пусто";
    }

// управление существующими записями
    void stringBeds(int i, string plants, string variety)
    {
        string str_new, str = "", str_old = "";
        using (StreamReader streamReader = new StreamReader(File.Open(path,
FileMode.Open)))
        {
            while (!streamReader.EndOfStream)
            {
                str = streamReader.ReadToEnd();
            }
        }
        using (StreamReader streamReader = new StreamReader(File.Open(path,
FileMode.Open)))
        {
```

```
            for (int ii = 0; ii <= i; ii++)
               {
                  str_old = streamReader.ReadLine();
               }
}
            str_new = "." + plants + "." + variety + ".";
            str = str.Replace(str_old, str_new);
            using (StreamWriter streamWriter = new StreamWriter(path, false))
            {
               streamWriter.WriteLine(str);
            }
            string[] lines = File.ReadAllLines(path);
            using (StreamWriter writer = new StreamWriter(path))
            {
               foreach (string line in lines)
                  if (!string.IsNullOrWhiteSpace(line))
                     writer.WriteLine(line);
            }
         }
      }
```

Для изменения уже существующей записи грядки создан метод stringBeds, в который параметрами передается информация о индексе выбранной грядки, сорте и культуре растения.

Форма «Управление микроклиматом».

Используемые библиотеки:
Пространство имен, содержащее фундаментальные и базовые классы:
using System;

Пространство имен, содержащее типы, позволяющие осуществлять чтение и запись в файлы и потоки данных:
using System.IO;

Пространство имен, содержащее классы, которые представляют кодировки ASCII и Юникода:
using System.Text;

Пространство имен, содержащее классы для создания приложений Windows:
using System.Windows.Forms;

Пространство имен, содержащее классы для создания соединения MQTT:
using uPLibrary.Networking.M2Mqtt;

Пространство имен, содержащее классы для отправки сообщений по протоколу MQTT:
using uPLibrary.Networking.M2Mqtt.Messages;

Пространство имен, обеспечивает доступ к базовым функциональным возможностям графического интерфейса GDI+:
using System.Drawing;

Пространство имен, содержащее классы и интерфейсы, которые дают возможность программировать в многопоточном режиме:
using System.Threading;

Код программы.
Глобальные переменные:

```
// переменная получаемого сообщения
 string x = "FFFF";
// путь к файлу с данными грядок
 string path = @"C:\Users\iskar\Desktop\beds.txt";
// путь к файлу включения света
 string pathWatering = @"C:\Users\iskar\Desktop\timeWatering.txt";
// путь к файлу включения полива.
 string pathLighting = @"C:\Users\iskar\Desktop\timeLigthing.txt";
// путь к журналу сенсоров
 string pathSensor = @"C:\Users\iskar\Desktop\sensorsBed.txt";
```

```
// путь журналу вентиляции
 string pathAir = @"C:\Users\iskar\Desktop\controlAir.txt";
// переменная в вывода номера грядки в ListBox
 string line;
// установка адреса MQTT-брокера
MqttClient mqttClient = new MqttClient("localhost");
```

Отправка и получение сообщений по протоколу MQTT:

```
// метод получения сообщений
public void Client_recievedMessage(object sender, MqttMsgPublishEventArgs e)
{
    string message = Encoding.Default.GetString(e.Message);
    x = message;
}
// метод отправки сообщений
 private static void client_MqttMsgPublished(object sender,
                                                MqttMsgPublishedEventArgs e)
 {
  e.IsPublished.ToString();
 }
public Form_control()
{
   InitializeComponent();
// установка адреса MQTT-брокера
   MqttClient mqttClient = new MqttClient("localhost");
// подписка на метод получения сообщений
   mqttClient.MqttMsgPublishReceived += Client_recievedMessage;
   string clientId = Guid.NewGuid().ToString();
// установка соединения с MQTT-брокером
   mqttClient.Connect(clientId, "KuSJIbIu", "ahohud44");
// подписка на топик
     mqttClient.Subscribe(new String[] { "/сенсор" }, new byte[] {
                          MqtTMgBase.QOS_LEVEL_AT_LEAST_ONCE });

   int i = 0;
// вывод списка грядок в ComboBox
   using (StreamReader sr = new StreamReader(path))
   {
    line = sr.ReadLine();
```

```
    while (line != null)
        {
                if (line != "")
                {
                bedsLigth.Items.Add(i + 1);
                beds.Items.Add(i + 1);
                line = sr.ReadLine();
                }
                else
                {
                        line = sr.ReadLine();
                }
                 i++;
         }
     }
     string[] lines = File.ReadAllLines(pathLighting);
// удаление пустых строк в файле
     using (StreamWriter writer = new StreamWriter(pathLighting))
      {
         foreach (string line in lines)
            if (!string.IsNullOrWhiteSpace(line))
              writer.WriteLine(line);
       }
 }
 private void Contorl_form_Load(object sender, EventArgs e)
 {
//подписка на метод отправки сообщений
  mqttClient.MqttMsgPublished += client_MqttMsgPublished;
  string clientId = Guid.NewGuid().ToString();
//установка соединения с MQTT-брокером
  mqttClient.Connect(clientId, "KuSJIbIu", "ahohud44");
 }
```

Панель полива:

```
// вывод окна управления поливом
 private void watering_Click(object sender, EventArgs e)
 {
  groupWatering.Enabled = true;
  groupWatering.Visible = true;
  groupWatering.Location = new Point(176, 43);
```

```
groupLighting.Enabled = false;
  groupLighting.Visible = false;
  groupLighting.Location = new Point(1000, 1000);

  groupAir.Enabled = false;
  groupAir.Visible = false;
  groupAir.Location = new Point(1000, 1000);

  groupHeating.Enabled = false;
  groupHeating.Visible = false;
  groupHeating.Location = new Point(1000, 1000);
 }
// вывод информации о грядке
private void beds_SelectedIndexChanged(object sender, EventArgs e)
 {
  int i = beds.SelectedIndex;
  using (StreamReader stream = new StreamReader(path))
  {
        for (int ii = 0; ii < i; ii++)
        {
              stream.ReadLine();
        }
        string str = stream.ReadLine();
        stream.Dispose();
        string[] word = str.Split('.');
        namePlant2.Text = "Культура: " + word[1];
        namePlantvariety2.Text = "Сорт: " + word[2];
  }
  wateringTime.Enabled = true;
  duration.Enabled = true;
  showTimeWater();
  button_watering.Enabled = true;
 }
```

При выборе соответсввующей грядки выводится информация о культуре и сорте посаженного растения. Также активируются панели ввода времени и продолжительности полива и кнопка принудительного полива. Это сделано, чтобы предотвратить ошибку при вводе времени\продолжительности полива не выбрав грядку.

```
 // кнопка принудительного полива
```

```
private void button_watering_Click(object sender, EventArgs e)
{
  string mes;
  if (button_watering.Text == "Включить")
  {
        button_watering.Text = "Выключить";
        mes = "1";
  }
  else
  {
        button_watering.Text = "Включить";
        mes = "0";
  }
  ushort msgId = mqttClient.Publish
  ("/грядки/" +
  (beds.SelectedIndex + 1).ToString(), // топик
  Encoding.UTF8.GetBytes("000" + mes),    // тело сообщения
  MqttMsgBase.QOS_LEVEL_EXACTLY_ONCE,        // QoS уровень
  false);                                    // retained
 }

// установка времени полива
private void enter_watering_Click(object sender, EventArgs e) // установка времени
полива
 {
  int i = beds.SelectedIndex;

  string str_new, str = "", str_old = "";
  string[] word;
  int x = Convert.ToInt32(hour_watering.Text);
  int y = Convert.ToInt32(minute_watering.Text);
  if (x >= 24 || y >= 60)
  {
        hour_watering.Text = "00";
        minute_watering.Text = "00";
  }
  using (StreamReader streamReader = new StreamReader(File.Open(pathWatering,
FileMode.Open)))
  {
        while (!streamReader.EndOfStream)
        {
```

```
                str = streamReader.ReadToEnd();
        }
  }

  using (StreamReader stream = new StreamReader(pathWatering))
  {
        for (int ii = 0; ii <= i; ii++)
        {
                str_old = stream.ReadLine();
        }
  }

  using (StreamReader stream = new StreamReader(pathWatering))
  {
        for (int ii = 0; ii < i; ii++)
        {
                stream.ReadLine();
        }
        string strSplit = stream.ReadLine();
        stream.Dispose();
        word = strSplit.Split('.');
  }
  i++;
  str_new = ".Грядка " + i.ToString() + "." + hour_watering.Text + "." +
minute_watering.Text + "." + word[4] + ".";
  i--;
        if (hour_watering.Text != word[2] || minute_watering.Text != word[3])
        {
                str = str.Replace(str_old, str_new);
                using (StreamWriter streamWriter = new StreamWriter(pathWatering,
false))
                {
                        streamWriter.WriteLine(str);
                }
  }
  string[] lines = File.ReadAllLines(pathWatering);
  using (StreamWriter writer = new StreamWriter(pathWatering))
  {
        foreach (string line in lines)
                 if (!string.IsNullOrWhiteSpace(line))
        writer.WriteLine(line);
```

```
 }
 }
// ввод продолжительности полива
private void Enter_duration_Click(object sender, EventArgs e)
  {
   int i = beds.SelectedIndex;

   string str_new, str = "", str_old = "";
   string[] word;

   using (StreamReader streamReader = new StreamReader(File.Open(pathWatering,
FileMode.Open)))
   {
        while (!streamReader.EndOfStream)
        {
             str = streamReader.ReadToEnd();
        }
   }
   using (StreamReader stream = new StreamReader(pathWatering))
   {
        for (int ii = 0; ii <= i; ii++)
        {
             str_old = stream.ReadLine();
        }
   }
   using (StreamReader stream = new StreamReader(pathWatering))
   {
        for (int ii = 0; ii < i; ii++)
        {
             stream.ReadLine();
        }
        string strSplit = stream.ReadLine();
        stream.Dispose();
        word = strSplit.Split('.');
         }
   i++;
   int durations = Convert.ToInt32(word[3]) +
Convert.ToInt32(watering_duration.Text);
   if (durations > 59) durations = durations - 60;
              str_new = ".Грядка " + i.ToString() + "." + word[2] + "." + word[3] + "." +
```

```
durations.ToString() + ".";
   i--;
   if (hour_watering.Text != word[4])
   {
         str = str.Replace(str_old, str_new);
         using (StreamWriter streamWriter = new StreamWriter(pathWatering, false))
         {
               streamWriter.WriteLine(str);
         }
   }
   string[] lines = File.ReadAllLines(pathWatering);
   using (StreamWriter writer = new StreamWriter(pathWatering))
   {
         foreach (string line in lines)
               if (!string.IsNullOrWhiteSpace(line))
         writer.WriteLine(line);
   }
  }
```

Панель освещения:

```
// вывод окна управления светом
 private void lighting_Click(object sender, EventArgs e)
 {
  groupLighting.Enabled = true;
  groupLighting.Visible = true;
  groupLighting.Location = new Point(176, 43);

  groupWatering.Enabled = false;
  groupWatering.Visible = false;
  groupWatering.Location = new Point(1000, 1000);

  groupAir.Enabled = false;
  groupAir.Visible = false;
  groupAir.Location = new Point(1000, 1000);

  groupHeating.Enabled = false;
  groupHeating.Visible = false;
  groupHeating.Location = new Point(1000, 1000);
 }

// выбор грядки для освещения
```

```
private void bedsLigth_SelectedIndexChanged(object sender, EventArgs e)
 {
  int i = bedsLigth.SelectedIndex;

  using (StreamReader stream = new StreamReader(path))
  {
        for (int ii = 0; ii < i; ii++)
        {
              stream.ReadLine();
        }
        string str = stream.ReadLine();
        stream.Dispose();
        string[] word = str.Split('.');
        namePlant.Text = "Культура: " + word[1];
        name_plant_variety.Text = "Сорт: " + word[2];
  }
  groupLightsOn.Enabled = true;
  groupLightsOff.Enabled = true;
  ShowTimeLight();
 }
```

При выборе соответсввующей грядки выводится информация о культуре и сорте посаженного растения. Также активируются панели ввода времени включения и выключения индивидуального освещения. Это сделано, чтобы предотвратить ошибку при вводе времени\продолжительности полива не выбрав грядку.

```
// кнопка ввода времени включения индивидульного освещения
 private void enterLightBeds_Click(object sender, EventArgs e)
 {
  int i = bedsLigth.SelectedIndex;

  string str_new, str = "", str_old = "";
  string[] word;
  int x = Convert.ToInt32(hour_LightingOn.Text);
  int y = Convert.ToInt32(minute_LightingOn.Text);
  if (x >= 24 || y >= 60)
  {
        hour_LightingOn.Text = "00";
        minute_LightingOn.Text = "00";
  }
```

```
using (StreamReader streamReader = new StreamReader(File.Open(pathLighting,
FileMode.Open)))
 {
        while (!streamReader.EndOfStream)
        {
                str = streamReader.ReadToEnd();
        }
 }

 using (StreamReader stream = new StreamReader(pathLighting))
 {
        for (int ii = 0; ii <= i; ii++)
        {
                str_old = stream.ReadLine();
        }
 }

 using (StreamReader stream = new StreamReader(pathLighting))
 {
        for (int ii = 0; ii < i; ii++)
        {
                stream.ReadLine();
        }
        string strSplit = stream.ReadLine();
        stream.Dispose();
        word = strSplit.Split('.');
 }
```



```
 i++;
 str_new = ".Грядка " + i.ToString() + "." + hour_LightingOn.Text + "." +
minute_LightingOn.Text + "." + word[4] + "." + word[5] + ".";
 if (hour_LightingOn.Text != word[2] || minute_LightingOn.Text != word[3])
 {
        str = str.Replace(str_old, str_new);
        using (StreamWriter streamWriter = new StreamWriter(pathLighting, false))
        {
                streamWriter.WriteLine(str);
        }
 }
 string[] lines = File.ReadAllLines(pathLighting);
```

```
 using (StreamWriter writer = new StreamWriter(pathLighting))
  {
        foreach (string line in lines)
              if (!string.IsNullOrWhiteSpace(line))
                    writer.WriteLine(line);
  }
  this.Refresh();

  }
// кнопка ввода времени выключения индивидуального освещения
 private void enterLightBedsOff_Click(object sender, EventArgs e)
 {
  int i = bedsLigth.SelectedIndex;

  string str_new, str = "", str_old = "";
  string[] word;
  int x = Convert.ToInt32(hour_LightingOff.Text);
  int y = Convert.ToInt32(minute_LightingOff.Text);
  if (x >= 24 || y >= 60)
  {
        hour_LightingOff.Text = "00";
        minute_LightingOff.Text = "00";
  }
  using (StreamReader streamReader = new StreamReader(File.Open(pathLighting,
FileMode.Open)))
  {
        while (!streamReader.EndOfStream)
        {
              str = streamReader.ReadToEnd();
        }
  }

  using (StreamReader stream = new StreamReader(pathLighting))
  {
        for (int ii = 0; ii <= i; ii++)
        {
              str_old = stream.ReadLine();
        }
  }

  using (StreamReader stream = new StreamReader(pathLighting))
```

```
{
        for (int ii = 0; ii < i; ii++)
        {
                stream.ReadLine();
        }
        string strSplit = stream.ReadLine();
        stream.Dispose();
        word = strSplit.Split('.');
  }
  i++;
  str_new = ".Грядка " + i.ToString() + "." + word[2] + "." + word[3] + "." +
hour_LightingOff.Text + "." + minute_LightingOff.Text + ".";
  if (hour_LightingOn.Text != word[4] || minute_LightingOn.Text != word[5])
  {
        str = str.Replace(str_old, str_new);
        using (StreamWriter streamWriter = new StreamWriter(pathLighting, false))
        {
                streamWriter.WriteLine(str);
        }
  }
  string[] lines = File.ReadAllLines(pathLighting);
  using (StreamWriter writer = new StreamWriter(pathLighting))
  {
        foreach (string line in lines)
                if (!string.IsNullOrWhiteSpace(line))
                        writer.WriteLine(line);
  }
  this.Refresh();
 }

// включение общего освещения
 private void all_lighting_Click(object sender, EventArgs e)
 {
  if (all_lighting.Text == "Включить")
  {
         all_lighting.Text = "Выключить";
          AllEquipmentPublish("5", "1");
  }
  else
  {
        all_lighting.Text = "Включить";
```

```
        AllEquipmentPublish("5", "0");
 }
}
```

Панель вентиляции:

```
// вывод окна управления вентиляцией
 private void air_Click(object sender, EventArgs e)
 {
  groupLighting.Enabled = false;
  groupLighting.Visible = false;
  groupLighting.Location = new Point(1000, 1000);

  groupWatering.Enabled = false;
  groupWatering.Visible = false;
  groupWatering.Location = new Point(1000, 1000);

  groupAir.Enabled = true;
  groupAir.Visible = true;
  groupAir.Location = new Point(176, 43);

  groupHeating.Enabled = false;
  groupHeating.Visible = false;
  groupHeating.Location = new Point(1000, 1000);

  string[] word;
  using (StreamReader stream = new StreamReader(pathAir))
  {
        string str = stream.ReadLine();
        stream.Dispose();
        word = str.Split('.');
  }
  if (word[1] == "0")
  {
         circulation_air.BackColor = Color.Red;
         circulation_air.ForeColor = Color.Red;
         circulationAirOnOff.Text = "Включить";
  }
  else
  {
         circulation_air.BackColor = Color.Green;
         circulation_air.ForeColor = Color.Green;
```

```
            circulationAirOnOff.Text = "Выключить";
        }
        if (word[2] == "0")
        {
            inAir.BackColor = Color.Red;
            inAir.ForeColor = Color.Red;
            inAirOnOff.Text = "Включить";
        }
        else
        {
            inAir.BackColor = Color.Green;
            inAir.ForeColor = Color.Green;
            inAirOnOff.Text = "Выключить";
        }
        if (word[3] == "0")
        {
            outAir.BackColor = Color.Red;
            outAir.ForeColor = Color.Red;
            outAirOnOff.Text = "Включить";
        }
        else
        {
            outAir.BackColor = Color.Green;
            outAir.ForeColor = Color.Green;
            outAirOnOff.Text = "Выключить";
        }
        if (word[4] == "0")
        {
            usualAir.BackColor = Color.Red;
            usualAir.ForeColor = Color.Red;
            usualAirOnOff.Text = "Включить";
        }
        else
        {
            usualAir.BackColor = Color.Green;
            usualAir.ForeColor = Color.Green;
            usualAirOnOff.Text = "Выключить";
        }
    }

    // управление циркуляцией воздуха
```

```
private void circulationAirOnOff_Click(object sender, EventArgs e)
 {
 if (circulationAirOnOff.Text == "Включить")
 {
        circulation_air.BackColor = Color.Green;
        circulation_air.ForeColor = Color.Green;
        circulationAirOnOff.Text = "Выключить";
        airMemory(1, "1");
        AllEquipmentPublish("1", "1");
 }
 else
 {
        circulation_air.BackColor = Color.Red;
        circulation_air.ForeColor = Color.Red;
        circulationAirOnOff.Text = "Включить";
        airMemory(1, "0");
        AllEquipmentPublish("1", "0");

 }
 Refresh();

 }

// управление приточной вентиляцией
 private void inAirOnOff_Click(object sender, EventArgs e)
 {
 if (inAirOnOff.Text == "Включить")
 {
        inAir.BackColor = Color.Green;
        inAir.ForeColor = Color.Green;
        inAirOnOff.Text = "Выключить";
        airMemory(2, "1");
        AllEquipmentPublish("2", "1");
 }
 else
 {
        inAir.BackColor = Color.Red;
        inAir.ForeColor = Color.Red;
              inAirOnOff.Text = "Включить";
 airMemory(2, "0");
       AllEquipmentPublish("2", "0");
```

```
 }
 Refresh();
 }

// управление вытяжной вентиляцией
 private void outAirOnOff_Click(object sender, EventArgs e)
 {
 if (outAirOnOff.Text == "Включить")
 {
        outAir.BackColor = Color.Green;
        outAir.ForeColor = Color.Green;
        outAirOnOff.Text = "Выключить";
        airMemory(3, "1");
        AllEquipmentPublish("3", "1");
 }
 else
 {
        outAir.BackColor = Color.Red;
        outAir.ForeColor = Color.Red;
        outAirOnOff.Text = "Включить";
        airMemory(3, "0");
        AllEquipmentPublish("3", "0");
 }
 Refresh();
 }

// управление естественной вентиляцией
 private void usualAirOnOff_Click(object sender, EventArgs e)
 {
 if (usualAirOnOff.Text == "Включить")
 {
        usualAir.BackColor = Color.Green;
        usualAir.ForeColor = Color.Green;
        usualAirOnOff.Text = "Выключить";
        airMemory(4, "1");
        AllEquipmentPublish("4", "1");
 }
 else
 {
        usualAir.BackColor = Color.Red;
        usualAir.ForeColor = Color.Red;
```

```
            usualAirOnOff.Text = "Включить";
            airMemory(4, "0");
            AllEquipmentPublish("4", "0");
  }
  Refresh();
 }
```

Для обозначения выключния\включения вентиляции также предуствотрены красные\зеленые индикаторы. Они ориентируются на свойство Text конкретной кнопки. Если параметр равен «Включить», то индикатор горит красным и наоборот.

Панель отопления:

```
// вывод числового значения открытия серво настенного радиатора
 private void wallRadiator_Scroll(object sender, EventArgs e)
 {
  wallRadiatorValue.Text = wallRadiator.Value.ToString();
 }
// вывод числового значения открытия серво калорифера вентилятора
 private void ventilatorRadiator_Scroll(object sender, EventArgs e)
 {
  ventilatorRadiatorValue.Text = ventilatorRadiator.Value.ToString();
 }
// ввод нового значения открытия серво настенного радиатора
 private void wallRadiatorEnter_Click(object sender, EventArgs e)
 {
  airMemory(7, wallRadiator.Value.ToString());
  AllEquipmentPublish("6", wallRadiator.Value.ToString());
 }
//ввод нового значения открытия серво калорифера вентилятора
 private void ventilatorRadiatorEnter_Click(object sender, EventArgs e)
 {
  airMemory(8, ventilatorRadiator.Value.ToString());
  AllEquipmentPublish("7", ventilatorRadiator.Value.ToString());
 }
```

Вспомогательные методы:

```
// вывод информации о грядках
 void showTimeWater(){
  int i = beds.SelectedIndex;
  string str_old = "";
  string[] word;
```

```
using (StreamReader stream = new StreamReader(pathWatering))
 {
        for (int ii = 0; ii <= i; ii++)
        {
                str_old = stream.ReadLine();
        }
 }

 using (StreamReader stream = new StreamReader(pathWatering))
 {
        for (int ii = 0; ii < i; ii++)
        {
                stream.ReadLine();
        }
        string strSplit = stream.ReadLine();
        stream.Dispose();
        word = strSplit.Split('.');
 }
 int x;
 hour_watering.Text = word[2];
 minute_watering.Text = word[3];
 x = Convert.ToInt32(word[4]) - Convert.ToInt32(word[3]);
 if (x < 0) x = x + 60;
 watering_duration.Text = x.ToString();

 using (StreamReader stream = new StreamReader(pathSensor))
 {
        for (int ii = 0; ii <= i; ii++)
        {
                str_old = stream.ReadLine();
        }
 }

 using (StreamReader stream = new StreamReader(pathSensor))
 {
        for (int ii = 0; ii < i; ii++)
        {
                stream.ReadLine();
        }
        string strSplit = stream.ReadLine();
        stream.Dispose();
```

```
            word = strSplit.Split('.');
  }

  temperature_soil.Text = word[2];
  moisture_soil.Text = word[3];

  this.Refresh();
 }
// системное время
 public void timer2_Tick(object sender, EventArgs e)
 {
  DateTime now = new DateTime();
  now = DateTime.Now;
  time.Text = now.TimeOfDay.ToString().Substring(0, 8);
  time.Refresh();
 }
// закрыть форму
 private void back_Form_Click(object sender, EventArgs e)
 {
  Close();
 }

//показания температуры и влажности
 private void timer1_Tick(object sender, EventArgs e)
 {

  if (x.Substring(2, 1) == "0")
  {
          adruinoSensorSoil(0);
  }
  else if (x.Substring(2, 1) == "1")
  {
          adruinoSensorSoil(1);
  } else if (x.Substring(2,1) == "2")
  {
          airMemory(5, x.Substring(3));
  } else if (x.Substring(2, 1) == "3")
  {
          airMemory(6, x.Substring(3));
```

```
 }
  else if (x.Substring(2, 1) == "4")
  {
        airMemory(9, x.Substring(3));
  }
 }
```

Метод каждую секунду проверяет сообщения, пришедшие на MQTT-брокер. И в зависимости от 3й цифры в ней записывает в файл «sensorsBed.txt».

```
// метод записи показаний в журнал сенсоров грядок
 void adruinoSensorSoil(int puc)
 {
  int i = Convert.ToInt32(x.Substring(1, 1)) - 1;

  string str_new = ".Грядка " + i + ".0.0.", str = "", str_old = "";
  string[] word;

  using (StreamReader streamReader = new StreamReader(File.Open(pathSensor,
FileMode.Open)))
  {
        while (!streamReader.EndOfStream)
        {
              str = streamReader.ReadToEnd();
        }
  }
  using (StreamReader stream = new StreamReader(pathSensor))
  {
        for (int ii = 0; ii <= i; ii++)
        {
              str_old = stream.ReadLine();
        }
  }
  using (StreamReader stream = new StreamReader(pathSensor))
  {
        for (int ii = 0; ii < i; ii++)
        {
              stream.ReadLine();
        }
        string strSplit = stream.ReadLine();
        stream.Dispose();
        word = strSplit.Split('.');
```

```
}
 i++;
 if (puc == 0)
 {
        str_new = ".Грядка " + i.ToString() + "." + x.Substring(3) + "." + word[3] + ".";
 }
 else if (puc == 1)
 {
        str_new = ".Грядка " + i.ToString() + "." + word[2] + "." + x.Substring(3) + ".";
 }
 i--;
 str = str.Replace(str_old, str_new);
 using (StreamWriter streamWriter = new StreamWriter(pathSensor, false))
 {
        streamWriter.WriteLine(str);
 }

 string[] lines = File.ReadAllLines(pathSensor);
 using (StreamWriter writer = new StreamWriter(pathSensor))
 {
        foreach (string line in lines)
                if (!string.IsNullOrWhiteSpace(line))
                        writer.WriteLine(line);
 }
 moisture_soil.Refresh();
 temperature_soil.Refresh();
}
// вывод информации о освещении грядок
 void ShowTimeLight()
 {
 int i = bedsLigth.SelectedIndex;

 string str_old = "";
 string[] word;

 using (StreamReader stream = new StreamReader(pathLighting))
 {
        for (int ii = 0; ii <= i; ii++)
        {
                str_old = stream.ReadLine();
```

```
        }
    }

    using (StreamReader stream = new StreamReader(pathLighting))
    {
        for (int ii = 0; ii < i; ii++)
        {
            stream.ReadLine();
        }
        string strSplit = stream.ReadLine();
        stream.Dispose();
        word = strSplit.Split('.');
    }

    hour_LightingOn.Text = word[2];
    minute_LightingOn.Text = word[3];
    hour_LightingOff.Text = word[4];
    minute_LightingOff.Text = word[5];

    this.Refresh();
   }
   bool flag = false;
  // проверка времени выполнения операций
   private void time_watering_Tick_1(object sender, EventArgs e)
   {
    int i = 0;
    int y = 0;
  // подсчет строк файла
    using (StreamReader stream = new StreamReader(pathLighting))
    {
        while (!stream.EndOfStream)
        {
            stream.ReadLine();
            i++;
        }
    }
  // управление освещением
    using (StreamReader stream = new StreamReader(pathLighting))
    {
        DateTime now = new DateTime();
```

```
            now = DateTime.Now;
             int numberBeds = 0;
             for (int ii = 0; ii < i; ii++)
             {
              numberBeds++;
              string[] word = stream.ReadLine().Split('.');
              if (word[2] == now.Hour.ToString() && word[3] == now.Minute.ToString())
              {
                    flag = true;
                    messageID("1", numberBeds.ToString(), "1");
                    this.Refresh();
              }
              else if (word[4] == now.Hour.ToString() && word[5] ==
now.Minute.ToString())
              {
                    flag = true;
                    messageID("0", numberBeds.ToString(), "1");
                    this.Refresh();
              }
  }

 }
// подсчет строк файла
  using (StreamReader stream = new StreamReader(pathWatering))
  {
             while (!stream.EndOfStream)
             {
                    stream.ReadLine();
                    y++;
             }
  }
// управление поливом
  using (StreamReader stream = new StreamReader(pathWatering))
  {
            DateTime now = new DateTime();
            now = DateTime.Now;
            int numberBeds = 0;
            int x = 0;
            for (int ii = 0; ii < y; ii++)
            {
                    numberBeds++;
```

```
                string[] word = stream.ReadLine().Split('.');
                if (word[2] == now.Hour.ToString() && word[3] ==
now.Minute.ToString())
                {
                        flag = true;
                        x = numberBeds;
                        messageID("1", numberBeds.ToString(), "0");
                        this.Refresh();
                 }
        else if (word[4] == now.Minute.ToString() && word[2] ==
now.Hour.ToString())
        {
                         flag = true;
                         messageID("0", numberBeds.ToString(), "0");
                         this.Refresh();
         }
  }
 }
}

// метод отправки сообщений
 private void messageID(string mes, string top, string taskBed)
 {
  if (flag == true)
  {
        ushort msgId = mqttClient.Publish("/грядки/" + top, // топик
        Encoding.UTF8.GetBytes("0" + top + taskBed + mes), // тело сообщения
        MqttMsgBase.QOS_LEVEL_EXACTLY_ONCE, // QoS уровень
        false);
        flag = false;
        Thread.Sleep(60000);
  }

 }

// память состояния вентиляции

 public void airMemory(int i, string y)
 {
  string[] word;
  string strSplit, str_new = "";
```

```
using (StreamReader stream = new StreamReader(pathAir))
  {
        strSplit = stream.ReadLine();
        stream.Dispose();
        word = strSplit.Split('.');
  }

  if (i == 1)str_new = "." + y + "." + word[2] + "." + word[3] + "." + word[4] + "." +
word[5] + "." + word[6] + "." + word[7] + "." + word[8] + "." + word[9] + ".";

  else if (i == 2)str_new = "." + word[1] + "." + y + "." + word[3] + "." + word[4] + "." +
word[5] + "." + word[6] + "." + word[7] + "." + word[8] + "." + word[9] + ".";

  else if (i == 3)str_new = "." + word[1] + "." + word[2] + "." + y + "." + word[4] + "." +
word[5] + "." + word[6] + "." + word[7] + "." + word[8] + "." + word[9] + ".";

  else if (i == 4)str_new = "." + word[1] + "." + word[2] + "." + word[3] + "." + y + "." +
word[5] + "." + word[6] + "." + word[7] + "." + word[8] + "." + word[9] + ".";

  else if (i == 5)str_new = "." + word[1] + "." + word[2] + "." + word[3] + "." + word[4]
+ "." + y + "." + word[6] + "." + word[7] + "." + word[8] + "." + word[9] + ".";

  else if (i == 6)str_new = "." + word[1] + "." + word[2] + "." + word[3] + "." + word[4]
+ "." + word[5] + "." + y + "." + word[7] + "." + word[8] + "." + word[9] + ".";

  else if (i == 7)str_new = "." + word[1] + "." + word[2] + "." + word[3] + "." + word[4]
+ "." + word[5] + "." + word[6] + "." + y + "." + word[8] + "." + word[9] + ".";

  else if (i == 8)str_new = "." + word[1] + "." + word[2] + "." + word[3] + "." + word[4]
+ "." + word[5] + "." + word[6] + "." + word[7] + "." + y + "." + word[9] + ".";

  else if (i == 9)str_new = "." + word[1] + "." + word[2] + "." + word[3] + "." + word[4]
+ "." + word[5] + "." + word[6] + "." + word[7] + "." + word[8] + "." + y + ".";

  using (StreamWriter streamWriter = new StreamWriter(pathAir, false))
  {
        streamWriter.WriteLine(str_new);
        streamWriter.Dispose();
  }
}
```

Метод полностью перезаписывает файл «controlAir.txt». Но записывает строку с определенным изменением определенного параметра.

```
// метод отправки сообщений общего оборудования
void AllEquipmentPublish(string numberAir, string OnOffMessage)
 {
        ushort msgId = mqttClient.Publish("/общееОборудование",    // топик
        Encoding.UTF8.GetBytes(numberAir + OnOffMessage),                  // тело
сообщения
        MqttMsgBase.QOS_LEVEL_EXACTLY_ONCE,                                  //
QoS уровень
        false);
 }
// вывод информации сенсоров влажности и температуры воздуха
 private void timerAir_Tick(object sender, EventArgs e)
 {
  using (StreamReader stream = new StreamReader(pathAir))
  {
        string strSplit = stream.ReadLine();
        stream.Dispose();
        string[] word = strSplit.Split('.');
        moistureAir.Text = word[5];
        temperatureAir.Text = word[6];
  }
 }
```

Листинг Arduino.

Скетч для управления микроклиматом на грядках.

```
 #include <ESP8266WiFi.h>
 #include <PubSubClient.h>
 #include <OneWire.h>
 #include <DallasTemperature.h>

 const int ID_beds = 1; // идентификатор грядки
 String topics = "грядки/" + ID_beds; // топик для подписки и отправки данных
```

```
  String publishMessage = ":1:" + ID_beds + ":";
  const char *ssid =  "AIRPORT";  // Имя вайфай точки доступа
  const char *pass =  "PASSWORD"; // Пароль от точки доступа

  const char *mqtt_server = "server"; // Имя сервера MQTT
  const int mqtt_port = 11140; // Порт для подключения к серверу MQTT
  const char *mqtt_user = "Login"; // Логи от сервер
  const char *mqtt_pass = "Pass"; // Пароль от сервера


  // Функция получения данных от сервера
  void callback(const MQTT::Publish& pub)
  {
    String payload = pub.payload_string();

    if(String(pub.topic()) == topics) // проверяем из нужного ли нам топика пришли данные
    {
    int stled = payload.toInt(); // преобразуем полученные данные в тип integer
    digitalWrite(5,stled);  //  включаем или выключаем светодиод в зависимоти от полученных значений данных
    }
  }

  WiFiClient wclient;
  PubSubClient client(wclient, mqtt_server, mqtt_port);

  void setup() {

    sensors.begin();
    Serial.begin(115200);
    delay(10);
    pinMode(5, OUTPUT);
  }

  void loop() {
    // подключаемся к wi-fi
    if (WiFi.status() != WL_CONNECTED) {
      WiFi.begin(ssid, pass);

      if (WiFi.waitForConnectResult() != WL_CONNECTED)
```

```
      return;
    }

    // подключаемся к MQTT серверу
    if (WiFi.status() == WL_CONNECTED) {
      if (!client.connected()) {
        if (client.connect(MQTT::Connect("arduinoClient2")
                           .set_auth(mqtt_user, mqtt_pass))) {
          client.set_callback(callback);
          client.subscribe(topics); // подписываемся по топик с данными для
светодиода
        } else {
        }
      }
      if (client.connected()){
        client.loop();
        TempSend();
            wetSensor();
    }

  }
  } // конец основного цикла

  // Функция отправки показаний с термодатчика
  void TempSend(){
    if (tm==0)
    {
    sensors.requestTemperatures();   // от датчика получаем значение температуры
    float temp = sensors.getTempCByIndex(0);
```



```
  client.publish(topics,String(temp)); // отправляем в топик для термодатчика
значение температуры
    tm = 300;  // пауза меду отправками значений температуры  коло 3 секунд
    }
    tm--;
    delay(10);
  }

  void wetSensor(){
```

```
        // контакт подключения аналогового выхода датчика
        int aPin=A0;
        // переменная для сохранения значения датчика
        int avalue=0;
        // получение значения с аналогового вывода датчика
        avalue=analogRead(aPin);
        if(tm == 0)
        {
        // отправление данных в топик
                client.publish(topics, publishMessage + String(avalue) + ":");
        }
        tm--;
        // пауза перед следующим получением значения 1000 мс
        delay(1000);
}
```

Скетч для управления вентиляцией

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include "DHT.h"
#include <Servo.h>

#define DHTPIN 2 // номер пина для датчика температуры и влажности воздуха

DHT dht(DHTPIN, DHT22); //Инициация датчика

String topic1 = "/общееОбородувание"; // топик для получения команд
String topic2 = "/сенсор" // топик для отправки данных
String publishMessagess = "10"; // адрес доставки сообщения
const char *ssid =  "AIRPORT";  // Имя вайфай точки доступа
const char *pass =  "PASSWORD"; // Пароль от точки доступа

const char *mqtt_server = "192.168.0.15"; // Имя сервера MQTT
const int mqtt_port = 1883; // Порт для подключения к серверу MQTT
const char *mqtt_user = "Login"; // Логи от сервер
const char *mqtt_pass = "Pass"; // Пароль от сервера

int PIN_circAir = 5; // пин для управления вентиляторами циркуляции
int PIN_inAir = 6; // пин для управления приточными вентиляторами
int PIN_outAir = 7; // пин для управления вытяжными вентиляторами
```

```
Servo servo; // переменная для управления естественной вентиляцией
servo.attach(8); // пин servo


void callback(const MQTT::Publish& pub)
{
  String payload = pub.payload_string();

 if (String(pub.topic()) == topic1) // проверяем из нужного ли нам топика пришли данные
  {
   if (pub.payload_string().substring(0, 1) == "1")
   {
    if (pub.payload_string().substring(1) == "1")
    {
     digitalWrite(PIN_circAir, HIGH); // включить вентиляцию циркуляции
    } else if (pub.payload_string().substring(1) == "0")
    {
     digitalWrite(PIN_circAir, LOW); // выключить вентиляцию циркуляции
    }
   } else if (pub.payload_string().substring(0, 1) == "2")
   {
    if (pub.payload_string().substring(1) == "1")
    {
     digitalWrite(PIN_inAir, HIGH); // включить приточную вентиляцию
    } else if (pub.payload_string().substring(1) == "0")
    {
            digitalWrite(PIN_inAir, LOW); // выключить приточную вентиляцию
    }
     } else if (pub.payload_string().substring(0, 1) == "3")
   {
    if (pub.payload_string().substring(1) == "1")
    {
     digitalWrite(PIN_outAir, HIGH); // включить вытяжную вентиляцию
    } else if (pub.payload_string().substring(1) == "0")
    {
            digitalWrite(PIN_outAir, LOW); // выключить вытяжную вентиляцию
    }
   }else if (pub.payload_string().substring(0, 1) == "4")
   {
    if (pub.payload_string().substring(1) == "1")
```

```
    {
       servo(90); // включить ествественную вентиляцию
     } else if (pub.payload_string().substring(1) == "0")
     {
              servo(-90); // выключить ествественную вентиляцию
     }
    }
  }
}

int tm = 600;
WiFiClient wclient;
PubSubClient client(wclient, mqtt_server, mqtt_port);

void setup() {
  Serial.begin(9600);
  delay(500);
  dht.begin();
}
void loop() {
  // подключаемся к wi-fi
  if (WiFi.status() != WL_CONNECTED) {
    WiFi.begin(ssid, pass);
    if (WiFi.waitForConnectResult() != WL_CONNECTED)
      return;
  }
  // подключаемся к MQTT серверу
  if (WiFi.status() == WL_CONNECTED) {
    if (!client.connected()) {
      if (client.connect(MQTT::Connect("arduinoClient2").set_auth(mqtt_user,
mqtt_pass)))
          {
        client.set_callback(callback);
        client.subscribe(topic1); // подписываемся по топик с общего оборудования
      } else {

      }
    }
    if (client.connected()) {
      client.loop();
      tempSensor();
```

```
      wetSensor();
    }
   }
  } // конец основного цикла
  void WetTempSensor(){
        float h = dht.readHumidity(); //Измеряем влажность
        float t = dht.readTemperature(); //Измеряем температуру
        if (tm == 300){
              client.publish(topics, publishMessagess + "2" + String(h)); // отправляем в
топик для термодатчика значение влажности
        } else if (tm == 0){
              client.publish(topic2, publishMessagess + "3" + String(t)); // отправляем в
топик для термодатчика значение температуры
        }
  }
```

Скетч для управления отоплением

```
  #include <ESP8266WiFi.h>
  #include <PubSubClient.h>
  #include <Servo.h>

  String topic1 = "/общееОбородувание"; // топик для получения команд

  const char *ssid =  "AIRPORT";  // Имя вайфай точки доступа
  const char *pass =  "PASSWORD"; // Пароль от точки доступа

  const char *mqtt_server = "192.168.0.15"; // Имя сервера MQTT
  const int mqtt_port = 1883; // Порт для подключения к серверу MQTT
  const char *mqtt_user = "Login"; // Логи от сервер
  const char *mqtt_pass = "Pass"; // Пароль от сервера

  Servo servo1; // переменная для управления отоплением
  servo1.attach(8); // пин servo
  Servo servo2; // переменная для управления отоплением
  servo2.attach(8); // пин servo

  void callback(const MQTT::Publish& pub)
  {
    String payload = pub.payload_string();
    int grad = 0;
```

```
  grad = atoi(pub.playload_string());

   if (String(pub.topic()) == topic1) // проверяем из нужного ли нам топика пришли данные
   {
       if (pub.payload_string().substring(0, 1) == "6")
    {
     if (pub.payload_string().substring(1) == "1")
     {
      servo1(grad); // включить отопление на радиатор
     } else if (pub.payload_string().substring(1) == "0")
     {
              servo1(-grad); // выключить отопление на радиатор
     }
    } else     if (pub.payload_string().substring(0, 1) == "7")
    {
     if (pub.payload_string().substring(1) == "1")
     {
      servo2(grad); // включить отопление на калорифер вентилятора
     } else if (pub.payload_string().substring(1) == "0")
     {
              servo2(-grad); // выключить отопление на калорифер вентилятора
     }
    }
   }
 }
 WiFiClient wclient;
 PubSubClient client(wclient, mqtt_server, mqtt_port);

 void setup() {
       Serial.begin(115200);
       delay(10);
 }

 void loop() {
  // подключаемся к wi-fi
  if (WiFi.status() != WL_CONNECTED) {
   WiFi.begin(ssid, pass);
   if (WiFi.waitForConnectResult() != WL_CONNECTED)
    return;
  }
```

```
    // подключаемся к MQTT серверу
    if (WiFi.status() == WL_CONNECTED) {
      if (!client.connected()) {
        if (client.connect(MQTT::Connect("arduinoClient2").set_auth(mqtt_user,
mqtt_pass)))
            {
          client.set_callback(callback);
        } else {
        }
          }
    }
  }
```