Изображение выглядит как графическая вставка, Графика, мультфильм

Автоматически созданное описание

**КУРСОВОЙ ПРОЕКТ**

**По дисциплине: МДК 01.01 Разработка программных модулей**

**Тема: Разработка программного модуля**

**информационной системы «Ресторан»**

**Специальность 09.02.07 «Информационные системы и программирование»**

|  |  |  |
| --- | --- | --- |
| **Выполнил студент(ка) группы 31ИС-23** | **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | **И.А. Гуров** |
| **Руководитель** | **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | **Л.Б. Гусятинер** |

**Москва 2024**

Изображение выглядит как графическая вставка, Графика, мультфильм

Автоматически созданное описание

**УТВЕРЖДАЮ**

**Зам. директора КМПО**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_С.Ф. Гасанов**

**«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.**

**ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ**

**По дисциплине: МДК 01.01 Разработка программных модулей**

**Специальность 09.02.07 «Информационные системы и программирование»**

**Студент группы 31ИС-23 Иван Иванов**

**ТЕМА: «Разработка программного модуля информационной**

**системы «Ресторан**

Дата выдачи задания «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_ 2024 г.

Срок сдачи проекта «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2024 г.

**Москва 2024**

Оглавление

[Введение 4](#_Toc184274685)

[Глава 1. Теоретическая часть 5](#_Toc184274686)

[1.1 Описание предметной области 5](#_Toc184274687)

[1.1.1 Интернет вещей 5](#_Toc184274688)

[1.1.2 Основные компоненты 5](#_Toc184274689)

[1.1.3 Применение Интернет вещей 6](#_Toc184274690)

[1.2 Описание существующих разработок 7](#_Toc184274691)

[Глава 2. Проектная часть 8](#_Toc184274692)

[2.1 Диаграмма прецедентов 8](#_Toc184274693)

[2.2 Выбор инструментов 9](#_Toc184274694)

[2.3 Проектирование сценария 10](#_Toc184274695)

[2.6 Описание модулей 20](#_Toc184274696)

[2.7 Описание тестовых наборов модулей 21](#_Toc184274697)

[Глава 3. Эксплуатационная часть 30](#_Toc184274698)

[3.1 Руководство оператора 30](#_Toc184274699)

[1 Назначение программы 30](#_Toc184274700)

[2 Условия выполнения программы 30](#_Toc184274701)

[Заключение 32](#_Toc184274702)

[Список литературы и интернет-источников 33](#_Toc184274703)

[Приложение 1 main.c 35](#_Toc184274704)

[Приложение 2 hash.c 40](#_Toc184274705)

[Приложение 3 hash.h 41](#_Toc184274706)

[Приложение 4 login.c 42](#_Toc184274707)

[Приложение 5 login.h 43](#_Toc184274708)

[Приложение 6 sketch\_nov30a.ino 44](#_Toc184274709)

# Введение

Целью данного курсового проекта является написание системы умного дома, которая обеспечит автоматизацию повседневных обязанностей в повседневной жизни основанная на базе Arduino ESP8266 с Wi-Fi модулем. Система умного дома в современном мире появилась относительно недавно и с каждым днем набирает все больше популярности среди современного населения по всему земному шару. Возможности умного дома ограничиваются только идеями самого пользователя например: удалённое управление освещением, просмотр данных всех различных датчиков, открытие и закрытие окон и дверей, просмотр камер видео наблюдения и многое другое.

В первой части будут рассмотрены предметная область данной темы и несколько готовых решений.

Во второй части будут рассмотрены разработанные модули и инструменты, а также листинги некоторых частей программы.

В третьей части будет рассмотрено руководство оператора.

# Глава 1. Теоретическая часть

## 1.1 Описание предметной области

Умный дом или известный еще как домашняя автоматизация - кластер устройств способных выполнять и анализировать множество команд и задач без прямого участия человека. Наиболее распространение сценарии использования систем умного дома:

1. автоматическое включение и выключение освещения
2. управление отопительными и сплит системами
3. системы уведомления разных датчиков

Домашняя автоматизация это одна из тем интернет вещей.

### 1.1.1 Интернет вещей

Интернет вещей (IoT) — это концепция, которая описывает сеть объектов реального и виртуального мира, подключённых к интернету и способных обмениваться данными. Это могут быть как физические устройства, так и виртуальные системы, которые взаимодействуют друг с другом, собирая и анализируя информацию для улучшения различных процессов и услуг.

### 1.1.2 Основные компоненты

Устройства и сенсоры: это физические объекты, такие как умные термостаты, фитнес-трекеры, автомобили и бытовая техника, которые могут собирать данные и отправлять их в облако или на другие устройства.

Сеть: Устройства подключаются к интернету через различные протоколы связи такие как Wi-Fi, Bluetooth, Zigbee, Meter-Bus и другие. Это позволяет им обмениваться данными в реальном времени.

Обработка данных:

Собранные данные анализируются с помощью облачных технологий или локальных серверов, что позволяет извлекать полезную информацию и принимать решения на основе анализа.

Интерфейсы: Пользователи могут взаимодействовать с устройствами через приложения, веб-интерфейсы или голосовые команды, что делает управление устройствами более удобным и доступным.

### 1.1.3 Применение Интернет вещей

Интернет вещей находит применение в различных сферах:

Умные дома:

Устройства, такие как умные лампочки, термостаты и системы безопасности, позволяют пользователям управлять своим домом удалённо и оптимизировать потребление ресурсов.

Здравоохранение:

Умные медицинские устройства, такие как носимые трекеры, помогают следить за состоянием здоровья пациентов и передавать данные врачам в реальном времени.

Промышленность:

IoT используется для мониторинга производственных процессов, управления оборудованием и предсказания поломок, что повышает эффективность и снижает затраты.

Транспорт: Умные транспортные системы, включая автомобили с автопилотом и системы управления трафиком, помогают улучшить безопасность и снизить заторы на дорогах.

Будущее Интернет вещей

По прогнозам, рынок Интернет вещей будет продолжать расти. В 2021 году ожидается, что инвестиции в эту область достигнут порядка 1,4 трлн долларов. Это связано с увеличением числа подключённых устройств и развитием технологий, таких как 5G, которые обеспечивают более быструю и надёжную связь.

Таким образом, Интернет вещей представляет собой важный шаг в развитии технологий, который открывает новые возможности для бизнеса и повседневной жизни, улучшая эффективность и удобство различных процессов.

## 1.2 Описание существующих разработок

На рынке программного обеспечения (дальше именуемым ПО) управления системами умного дома существует множество вариантов, безусловными лидерами данной области на российском рынке являются:

1. Яндекс с системой управлением умным домом именуемый “Алиса”
2. Сбер с системой управлением умным домом именуемый “Салют”

Обе компании предлагают мобильное приложение, которое позволяет управлять умными устройствами не только тех, которые производит сама компания, но и сторонних производителей.

Так же к системе умного дома они предлагают устройство для голосового управления системой умного дома, а именно колонку с голосовым ассистентом.

# Глава 2. Проектная часть

## 2.1 Диаграмма прецедентов

Для определения вариантов использования к проекту была построена диаграмма прецедентов

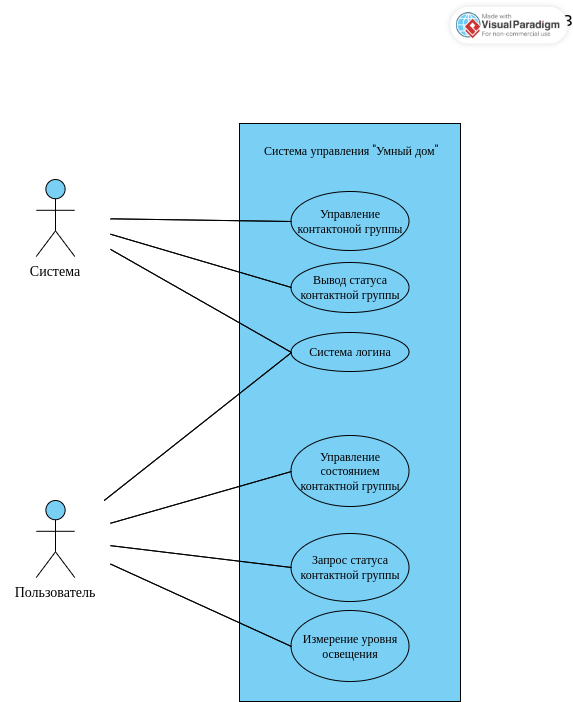


Рисунок 1. Диаграмма прецедентов для проекта

## 2.2 Выбор инструментов

При выборе инструмента и языка программирования было проведено сравнение по критериям, которые представлены в таблицах 1 и 2.

Степень важности будет оцениваться по 5 системе, где 5 — это наибольшая важность, а 1 — это необязательный фактор.

Таблица 1. Критерии выбора инструмента.

|  |  |
| --- | --- |
| Критерий | Важность критерия |
| Функционал | 3 |
| Удобство использования | 2 |
| Скорость разработки | 3 |

Таблица 2. Оценка языков программирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерий  Язык  программирования | C++ | Python | C | bash |
| Функционал | 4 | 5 | 4 | 3 |
| Удобство использования | 3 | 2 | 5 | 2 |
| Скорость разработки | 3 | 5 | 4 | 3 |
| Итого | 10 | 12 | 13 | 8 |

По итогам проделанной работы я выбрал для себя язык программирования C.

## 2.3 Проектирование сценария

В данном разделе приведен пример использования программы пользователем.

После запуска программы пользователь должен войти в учетную запись. После пользователь для изучения функционала может написать команду help которая выведет перечень команд. Подробнее можно узнать в Главе 3.

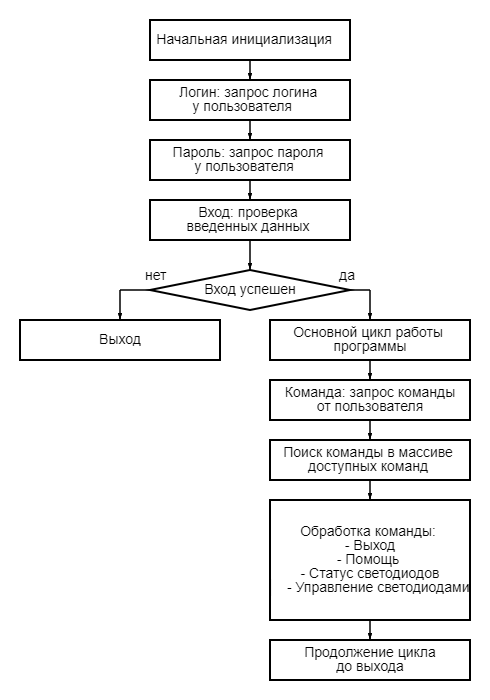


Рисунок 2. Сценарий использования2.4 Описание главного модуля

Главный модуль представляет собой файл main.c. К главному модулю подключаются остальные модули, содержащие в себе классы и функции.

Листинг 1. Код главного модуля

#include <stdio.h>

#include <string.h>

#include <stdbool.h>

#include "hash.h"

#include "login.h"

// Функция для поиска команды в массиве

int find(char \*word, char \*mass[]) {

for (int i = 0; mass[i] != NULL; i++) {

if (strcmp(mass[i], word) == 0) {

return i;

}

}

return -1;

}

'''

Вызов приведущих функций через стрелки

clear

'''

int main() {

int led\_1\_status = 0, led\_2\_status = 0, led\_3\_status = 0, led\_4\_status = 0;

char login[MAX\_LEN\_STR];

char password[MAX\_LEN\_STR];

static char \*commands[] = {"exit", "help", "status", "led\_1 0", "led\_1 1", "led\_2 0", "led\_2 1", "led\_3 0", "led\_3 1","led\_4 0", "led\_4 1", NULL};

static char \*yes[] = {"yes", "y", "yea", "yup", "yep"};

bool a = true;

// Запрос логина

printf("login: ");

scanf("%s", login);

// Запрос пароля

printf("password for %s: ", login);

scanf("%s", password);

// Попытка входа

bool access = log\_in(login, password);

getchar();

// Цикл команд

while (access) {

char input[MAX\_LEN\_STR];

if(a)

printf("For a short summary of all commands, run 'help'\n");

a = false;

// Запрос команды

printf("[%s] ", login);

fgets(input, MAX\_LEN\_STR, stdin);

int length = strlen(input);

if (input[length - 1] == '\n') {

input[length - 1] = '\0'; // Убираем символ новой строки в конце строки

length--; // Уменьшаем длину на 1, если убрали символ новой строки

}

// Поиск команды

int command\_index = find(input, commands);

// Обработка команд

printf("[JARVIS] ");

switch (command\_index) {

case 0: // "exit"

char temp[MAX\_LEN\_STR];

printf("you are sure? (yes or no):");

scanf("%s", temp);

if(find(temp ,yes) != -1){

printf("exiting... \n");

access = false;

}

else{

getchar();

}

break;

case 1:

printf("\n\

exit for exit from terminal\n\

status for print status pin\n\

led\_<num of pin> <status\_pin> for control pin\n");

break;

case 2:

printf("status all pins");

if(led\_1\_status == 0){

printf("\nled\_1 status" COLOR\_RED " %d" COLOR\_RESET"\n",led\_1\_status);

}

else{

printf("\nled\_1 status" COLOR\_GREEN " %d" COLOR\_RESET"\n",led\_1\_status);

}

if(led\_2\_status == 0){

printf("\nled\_2 status" COLOR\_RED " %d" COLOR\_RESET"\n",led\_2\_status);

}

else{

printf("\nled\_2 status" COLOR\_GREEN " %d" COLOR\_RESET"\n",led\_2\_status);

}

if(led\_3\_status == 0){

printf("\nled\_3 status" COLOR\_RED " %d" COLOR\_RESET"\n",led\_3\_status);

}

else{

printf("\nled\_3 status" COLOR\_GREEN " %d" COLOR\_RESET"\n",led\_3\_status);

}

if(led\_4\_status == 0){

printf("\nled\_4 status" COLOR\_RED " %d" COLOR\_RESET"\n",led\_4\_status);

}

else{

printf("\nled\_4 status" COLOR\_GREEN " %d" COLOR\_RESET"\n",led\_4\_status);

}

break;

case 3:

led\_1\_status = 0;

printf("led 1" COLOR\_RED " off" COLOR\_RESET"\n");

break;

case 4:

led\_1\_status = 1;

printf("led 1" COLOR\_GREEN " on" COLOR\_RESET"\n");

break;

case 5:

led\_2\_status = 0;

printf("led 2" COLOR\_RED " off" COLOR\_RESET"\n");

break;

case 6:

led\_2\_status = 1;

printf("led 2" COLOR\_GREEN " on" COLOR\_RESET"\n");

break;

case 7:

led\_3\_status = 0;

printf("led 3" COLOR\_RED " off" COLOR\_RESET"\n");

break;

case 8:

led\_3\_status = 1;

printf("led 3" COLOR\_GREEN " on" COLOR\_RESET"\n");

break;

case 9:

led\_4\_status = 0;

printf("led 4" COLOR\_RED " off" COLOR\_RESET"\n");

break;

case 10:

led\_4\_status = 1;

printf("led 4" COLOR\_GREEN " on" COLOR\_RESET"\n");

break;

default: // Неизвестная команда

printf("unknown command\n");

}

}

return 0;

}

Изображение выглядит как текст, диаграмма, Параллельный, линия

Автоматически созданное описаниеНиже представлена схема главного модуля программы

Рисунок 3. Блок-схема программы.

2.5 Спецификация к модулям

Спецификация к модулю hash (листинг 2).

Листинг 2 Код модуля hash.

#include "hash.h"

int hash(const char \*str) {

unsigned long hash = 5381; // Стандартное начальное значение для DJB2

int c;

while ((c = \*str++)) {

hash = ((hash << 5) + hash) + c; // hash \* 33 + c

}

return hash;

}

Спецификация к модулю login (листинг 3).

Листинг 3 Код модуля login.

#include <stdio.h>

#include <string.h>

#include "hash.h"

#include "login.h"

int log\_in(char \*login\_i, char \*password\_i) {

unsigned long login\_h = hash(login\_i);

unsigned long password\_h = hash(password\_i);

unsigned long long password = 252832602ULL; // Используем ULL для больших чисел

unsigned long long login = 404414076; // Используем ULL для больших чисел

if (login != login\_h) {

printf(COLOR\_RED "invalid login" COLOR\_RESET "\n");

return false;

}

if (password != password\_h) {

printf(COLOR\_RED "access denied" COLOR\_RESET "\n");

return false;

} else {

printf(COLOR\_GREEN "access successful" COLOR\_RESET "\n");

return true;

}

}

## 2.6 Описание модулей

В данной главе описаны используемые модули.

1. Главный модуль

Главный модуль представляет собой файл main.c. К главному модулю подключаются остальные модули, содержащие в себе функции.

1. Модуль функций

В модуле функций находятся функции, используемые в проекте:

* strcmp - сравнивает строки.
* strlen – возвращает длину строки
* printf – вывод сообщения в терминал.
* scanf - осуществляет форматированный ввод данных.
* getchar - предназначена для чтения одного байта.
* fgets - считывает строку с клавиатуры.
* hash – функция которая хеширует передаваемую в нее строку.
* find – функция, которая получает строку и массив строк и выводит номер строки в массиве.
* log\_in – функция которая получает на вход логин и пароль и сравнивает с сохранённым хешированным логином и паролем и возвращает true если они совпали.

## 2.7 Описание тестовых наборов модулей

В этом разделе будут продемонстрированы результаты тестирования “черного ящика”.

Тест 1. Вход в профиль.

Действия: запуск программы, ввод правильного логина и пароля.

Ожидаемый результат: “access successful”

Результат теста:

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 4. Вход в профиль.

Тест 2. Неверный пароль.

Действия: запуск программы, ввод правильного логина и не правильного пароля.

Ожидаемый результат: “access denied”

Результат теста:

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 5. Неверный пароль.

Тест 3. Неверный Логин.

Действия: запуск программы, ввод не правильного логина и пароля.

Ожидаемый результат: “invalid login”

Результат теста:

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рисунок 6. Неверный логин.

Тест 4. Команда помощи.

Действия: ввод команды помощи “help”.

Ожидаемый результат: вывод справочной информации

Результат теста:

Изображение выглядит как текст, снимок экрана, Шрифт, черный

Автоматически созданное описание

Рисунок 7. Команда помощи.

Тест 5. Команда статуса.

Действия: ввод команды помощи “status”.

Ожидаемый результат: вывод статуса устройств

Результат теста:

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 8. Команда статуса.

Тест 6. Команда изменения статуса устройства.

Действия: ввод команды помощи “led\_1 1”.

Ожидаемый результат: led 1 on

Результат теста:

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Рисунок 9. Команда статуса.

Тест 7. Команда изменения статуса устройства в меню статуса устройств.

Действия: ввод команды помощи “status”.

Ожидаемый результат: вывод статуса устройств

Результат теста:

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 9. Команда изменения статуса устройства в меню статуса устройств.

Тест 8. Неизвестная команда.

Действия: ввод неизвестной команды.

Ожидаемый результат: unknown command

Результат теста:

Изображение выглядит как текст, Шрифт, снимок экрана, дизайн

Автоматически созданное описание

Рисунок 10. Неизвестная команда.

Тест 9. Выход из программы.

Действия: ввод “exit” и подтверждение выхода.

Ожидаемый результат: exiting…

Результат теста:

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рисунок 11. Выход из программы.

# Глава 3. Эксплуатационная часть

## 3.1 Руководство оператора

### 1 Назначение программы

**Функциональное назначение программы**

Основной функцией проекта "smart-home" является автоматизация процесса управления устройствами умного дома.

**Эксплуатационное назначение программы**

Программное обеспечение "smart-home" может быть использовано в любом коммерческом и не коммерческом здании для управления устройствами умного дома.

**Состав функций**

Функции оператора:

1. Включение\выключение освещения
2. Просмотр статус устройства
3. Авторизация

### 2 Условия выполнения программы

Минимальный состав аппаратных средств

Минимальный состав используемых технических (аппаратных) средств:

* OC Windows 7/8.1/10/11 , GNU Linux
* Процессор с тактовой частотой не ниже 500 МГц
* ОЗУ 128КБ
* Место на жестком диске 7КБ
* Доступ в интернет

**Минимальный состав программных средств**

Дополнительные программы не требуются

**Требования к персоналу (оператору)**

Элементарные умения работы с терминалом.

**Загрузка и запуск программы**

Для запуска необходимо запустить скомпилированную программу под вашу ОС.

При запуске программы должно открыться следующее окно:

Изображение выглядит как снимок экрана, мультфильм, Анимация

Автоматически созданное описание

Рисунок 12. Начало работы с программой.

Далее требуется ввести логин и пароль если пароль логин и пароль введен не верен, то программа завершиться (см. рисунок 4–6). После успешной авторизации пользователь может узнать какие функции поддерживает программа, введя команду “help”(см. рисунок 7).

# Заключение

В результате выполнения курсового проекта была написана программа "smart-home" для автоматизации процесса управления умным домом.

В ходе работы были проанализированы предметная область, существующие разработки, посвященные данному направлению, получены практические навыки.

Так же планируется продолжить работу над данным проектом с целью расширения возможностей и удобства для пользователей. Планы по доработке представлены ниже.

1. Улучшение системы управления
2. Расширение функционала программы
3. Реализация голосового помощника .

# Список литературы и интернет-источников

Ли П. Архитектура интернета вещей / пер. с анг. М. А. Райтмана. – М.: ДМК Пресс, 2019. – 454 с.: ил

Умный дом на базе Arduino. Руководство пользователя

Умный дом своими руками. Тесля Е.В.

Кокунин П.А. Введение в Интернет вещей. Казань: Издательство Казанского университета, 2022. – 147 с. – Систем. требования: Adobe Acrobat Reader. – URL: https://kpfu.ru/portal/docs/F\_378200975/IOT.pdf. –

ISBN 978-5-00130-649-8

Глушак, Елена Владимировна. Введение в Интернет вещей: учебное пособие – Самара: Издательство Самарского университета, 2023. – 104 с.: ил.

ISBN 978-5-7883-2010-6

ИНТЕРНЕТ ВЕЩЕЙ, УМНЫЙ ДОМ И УМНЫЕ ГОРОДА. Кузяшев Азат Нургалеевич, Смолин Артем Евгеньевич

Папуловская, Наталья Владимировна. Основы интернета вещей : учебно-методическое пособие; М-во науки и высшего образования РФ.— Екатеринбург : Изд-во Урал. ун-та, 2022.— 104 с

То ли дом, то ли лаборатория: как устроен умный дом профессионала

https://habr.com/ru/companies/wirenboard/articles/821347/

Секреты умного дома от профессионала: что скрывает Андрей в своем особняке. https://habr.com/ru/companies/wirenboard/articles/850612/

Как работает умный дом и чем он может быть полезен

https://ichip.ru/tekhnologii/kak-rabotaet-umnyj-dom-854522

Основы построения системы "умный дом"

https://intuit.ru/studies/courses/644/500/info

Устелемова Основы построения системы "умный дом"

Что такое Умный Дом, общая структура Умного Дома и роль MajorDoMo при построении Умного Дома. https://rutube.ru/video/c0fafdeae14c3bb02f90232223209ad8/

Средства домашней автоматизации. Теория и практика «Умных домов». Часть первая. https://habr.com/ru/articles/381647/

Средства домашней автоматизации. Теория и практика «Умных домов». Часть вторая. https://habr.com/ru/articles/381649/

Система «Умный дом». Что это такое и почему он нужен каждому?

https://blog.eldorado.ru/publications/sistema-umnyy-dom-chto-eto-takoe-i-pochemu-on-nuzhen-kazhdomu-37927

Принципы построения современной системы «Умный дом»

https://www.elec.ru/publications/tsifrovye-tekhnologii-svjaz-izmerenija/1885/

Умный дом с нуля. https://stepik.org/course/125483/syllabus

Управление домом при помощи KNX: освещение

https://habr.com/ru/articles/427669/

С.В. Богданов - Умный Дом

Элсенпитер, Велт - Умный Дом строим сами

В.А. Петин - Создание умного дома на базе Arduino

А.П. Кашкаров - Электронные схемы для "умного дома"

# Приложение 1 main.c

#include <stdio.h>

#include <string.h>

#include <stdbool.h>

#include "hash.h"

#include "login.h"

// Функция для поиска команды в массиве

int find(char \*word, char \*mass[]) {

for (int i = 0; mass[i] != NULL; i++) {

if (strcmp(mass[i], word) == 0) {

return i;

}

}

return -1;

}

'''

Вызов приведущих функций через стрелки

clear

'''

int main() {

int led\_1\_status = 0, led\_2\_status = 0, led\_3\_status = 0, led\_4\_status = 0;

char login[MAX\_LEN\_STR];

char password[MAX\_LEN\_STR];

static char \*commands[] = {"exit", "help", "status", "led\_1 0", "led\_1 1", "led\_2 0", "led\_2 1", "led\_3 0", "led\_3 1","led\_4 0", "led\_4 1", NULL};

static char \*yes[] = {"yes", "y", "yea", "yup", "yep"};

bool a = true;

// Запрос логина

printf("login: ");

scanf("%s", login);

// Запрос пароля

printf("password for %s: ", login);

scanf("%s", password);

// Попытка входа

bool access = log\_in(login, password);

getchar();

// Цикл команд

while (access) {

char input[MAX\_LEN\_STR];

if(a)

printf("For a short summary of all commands, run 'help'\n");

a = false;

// Запрос команды

printf("[%s] ", login);

fgets(input, MAX\_LEN\_STR, stdin);

int length = strlen(input);

if (input[length - 1] == '\n') {

input[length - 1] = '\0'; // Убираем символ новой строки в конце строки

length--; // Уменьшаем длину на 1, если убрали символ новой строки

}

// Поиск команды

int command\_index = find(input, commands);

// Обработка команд

printf("[JARVIS] ");

switch (command\_index) {

case 0: // "exit"

char temp[MAX\_LEN\_STR];

printf("you are sure? (yes or no):");

scanf("%s", temp);

if(find(temp ,yes) != -1){

printf("exiting... \n");

access = false;

}

else{

getchar();

}

break;

case 1:

printf("\n\

exit for exit from terminal\n\

status for print status pin\n\

led\_<num of pin> <status\_pin> for control pin\n");

break;

case 2:

printf("status all pins");

if(led\_1\_status == 0){

printf("\nled\_1 status" COLOR\_RED " %d" COLOR\_RESET"\n",led\_1\_status);

}

else{

printf("\nled\_1 status" COLOR\_GREEN " %d" COLOR\_RESET"\n",led\_1\_status);

}

if(led\_2\_status == 0){

printf("\nled\_2 status" COLOR\_RED " %d" COLOR\_RESET"\n",led\_2\_status);

}

else{

printf("\nled\_2 status" COLOR\_GREEN " %d" COLOR\_RESET"\n",led\_2\_status);

}

if(led\_3\_status == 0){

printf("\nled\_3 status" COLOR\_RED " %d" COLOR\_RESET"\n",led\_3\_status);

}

else{

printf("\nled\_3 status" COLOR\_GREEN " %d" COLOR\_RESET"\n",led\_3\_status);

}

if(led\_4\_status == 0){

printf("\nled\_4 status" COLOR\_RED " %d" COLOR\_RESET"\n",led\_4\_status);

}

else{

printf("\nled\_4 status" COLOR\_GREEN " %d" COLOR\_RESET"\n",led\_4\_status);

}

break;

case 3:

led\_1\_status = 0;

printf("led 1" COLOR\_RED " off" COLOR\_RESET"\n");

break;

case 4:

led\_1\_status = 1;

printf("led 1" COLOR\_GREEN " on" COLOR\_RESET"\n");

break;

case 5:

led\_2\_status = 0;

printf("led 2" COLOR\_RED " off" COLOR\_RESET"\n");

break;

case 6:

led\_2\_status = 1;

printf("led 2" COLOR\_GREEN " on" COLOR\_RESET"\n");

break;

case 7:

led\_3\_status = 0;

printf("led 3" COLOR\_RED " off" COLOR\_RESET"\n");

break;

case 8:

led\_3\_status = 1;

printf("led 3" COLOR\_GREEN " on" COLOR\_RESET"\n");

break;

case 9:

led\_4\_status = 0;

printf("led 4" COLOR\_RED " off" COLOR\_RESET"\n");

break;

case 10:

led\_4\_status = 1;

printf("led 4" COLOR\_GREEN " on" COLOR\_RESET"\n");

break;

default: // Неизвестная команда

printf("unknown command\n");

}

}

return 0;

}

# Приложение 2 hash.c

#include "hash.h"

int hash(const char \*str) {

unsigned long hash = 5381; // Стандартное начальное значение для DJB2

int c;

while ((c = \*str++)) {

hash = ((hash << 5) + hash) + c; // hash \* 33 + c

}

return hash;

}

# Приложение 3 hash.h

#ifndef HASH\_H

#define HASH\_H

// Функция для вычисления хеша строки

int hash(const char \*str);

#endif // HASH\_H

# Приложение 4 login.c

#include <stdio.h>

#include <string.h>

#include "hash.h"

#include "login.h"

int log\_in(char \*login\_i, char \*password\_i) {

unsigned long login\_h = hash(login\_i);

unsigned long password\_h = hash(password\_i);

unsigned long long password = 252832602ULL; // Используем ULL для больших чисел

unsigned long long login = 404414076; // Используем ULL для больших чисел

if (login != login\_h) {

printf(COLOR\_RED "invalid login" COLOR\_RESET "\n");

return false;

}

if (password != password\_h) {

printf(COLOR\_RED "access denied" COLOR\_RESET "\n");

return false;

} else {

printf(COLOR\_GREEN "access successful" COLOR\_RESET "\n");

return true;

}

}

# Приложение 5 login.h

#ifndef LOGIN\_H

#define LOGIN\_H

#include <stdbool.h>

// Цвета для вывода в терминал

#define COLOR\_RED "\x1b[31m"

#define COLOR\_GREEN "\x1b[32m"

#define COLOR\_YELLOW "\x1b[33m"

#define COLOR\_BLUE "\x1b[34m"

#define COLOR\_MAGENTA "\x1b[35m"

#define COLOR\_CYAN "\x1b[36m"

#define COLOR\_RESET "\x1b[0m"

// Константы

#define MAX\_LEN\_STR 255

// Функция для проверки логина и пароля

int log\_in(char \*login\_i, char \*password\_i);

#endif // LOGIN\_H

# Приложение 6 sketch\_nov30a.ino

#include <ESP8266WiFi.h>

const char\* \_ssid = "Paranoia";

const char\* \_password = "Paranoia\_Sarmat" ;

// Определение пинов кнопок и светодиодов

const int btn\_pin1 = 3;

const int btn\_pin2 = 5;

const int btn\_pin3 = 4;

const int btn\_pin4 = 2;

const int led\_pin1 = 14;

const int led\_pin2 = 12;

const int led\_pin3 = 13;

const int led\_pin4 = 15;

bool btn\_state1 = false;

bool btn\_state2 = false;

bool btn\_state3 = false;

bool btn\_state4 = false;

void setup() {

WIFIinit();

Serial.begin(9600); // Инициализация последовательного порта

pinMode(btn\_pin1, INPUT\_PULLUP);

pinMode(btn\_pin2, INPUT\_PULLUP);

pinMode(btn\_pin3, INPUT\_PULLUP);

pinMode(btn\_pin4, INPUT\_PULLUP);

pinMode(led\_pin1, OUTPUT);

pinMode(led\_pin2, OUTPUT);

pinMode(led\_pin3, OUTPUT);

pinMode(led\_pin4, OUTPUT);

}

void loop() {

// Проверка состояния каждой кнопки

checkButton(btn\_pin1, &btn\_state1, led\_pin1);

checkButton(btn\_pin2, &btn\_state2, led\_pin2);

checkButton(btn\_pin3, &btn\_state3, led\_pin3);

checkButton(btn\_pin4, &btn\_state4, led\_pin4);

}

void checkButton(int btn\_pin, bool \*btn\_state, int led\_pin) {

int currentBtnState = digitalRead(btn\_pin);

if (currentBtnState == LOW && \*btn\_state == true) {

delay(50);

\*btn\_state = false;

digitalWrite(led\_pin, !digitalRead(led\_pin));

} else if (currentBtnState == HIGH && \*btn\_state == false) {

\*btn\_state = true;

}

}