**Содержание**

1. По программе

* Как работает программа, её возможности и краткое описание.
* Требования к вводимым данным.

1. Описание кода

* Файлы и их назначения
* Доп. библиотеки
* Разбор main.py
* Разбор Interface\_tkinter.py
* Разбор Interface\_Functions.py
* Разбор For\_basedata.py
* Разбор Scrapping\_weather.py

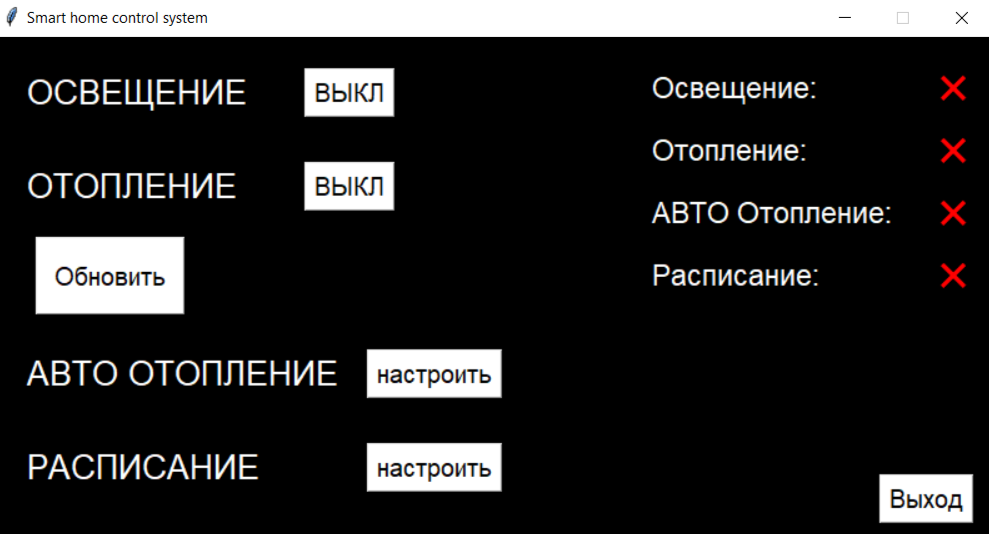
По программе

Краткое описание.

Программа системы умного дома включает в себя возможности включения и настройки освещения и отопления. Имеет централизованный интерфейс, с сохранением состояния каждого устройства в базу данных. Так же есть возможность отслеживания настроек каждого устройства – пользователь сможет легко понимать, какие настройки он ставил ранее, чтобы удобнее пользоваться функционалом приложения.

Инструкция.

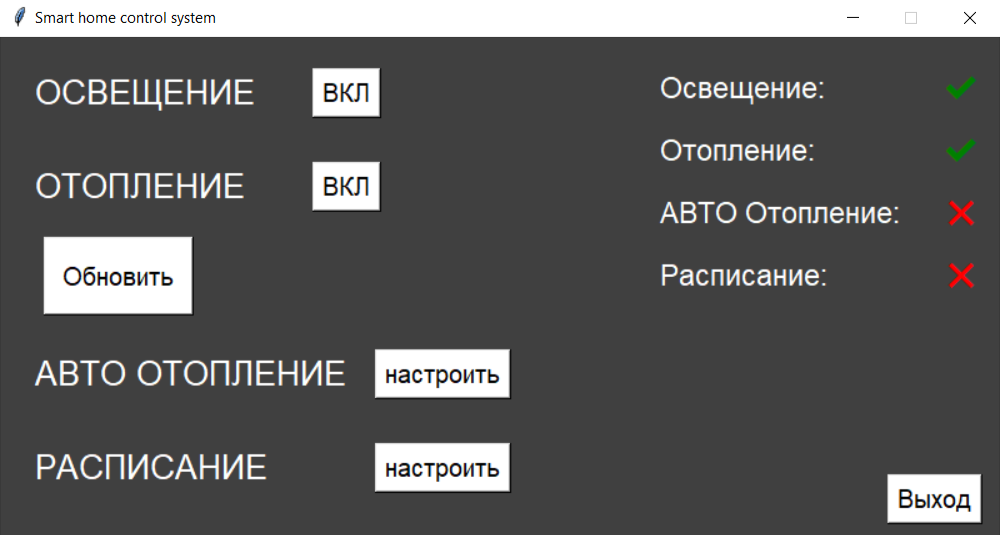
Запуская программу, у вас выйдет такое окно:



Это главное меню приложения. В нём вы имеете возможность включить *освещение*, *отопление*, настроить *авто* *отопление* и *расписание* включения освещения и отопления, а также кнопки *Выход* и *Обновить*.

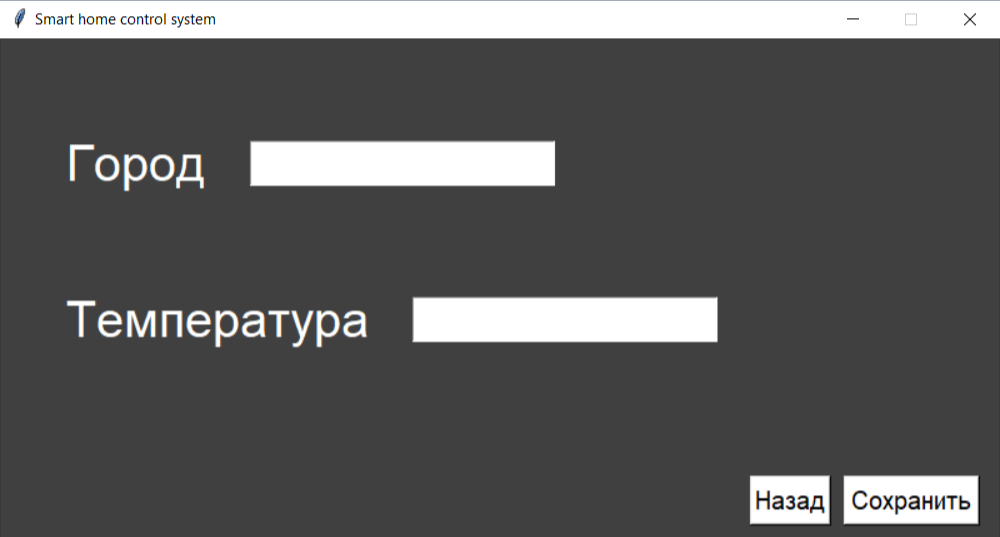
В правом верхнем углу у программы отображаются состояния устройств системы умного дома.

Включая освещение и отопление, изменятся их состояния в части мониторинга (правый верхний угол) и изменится цвет всего интерфейса на более яркий.



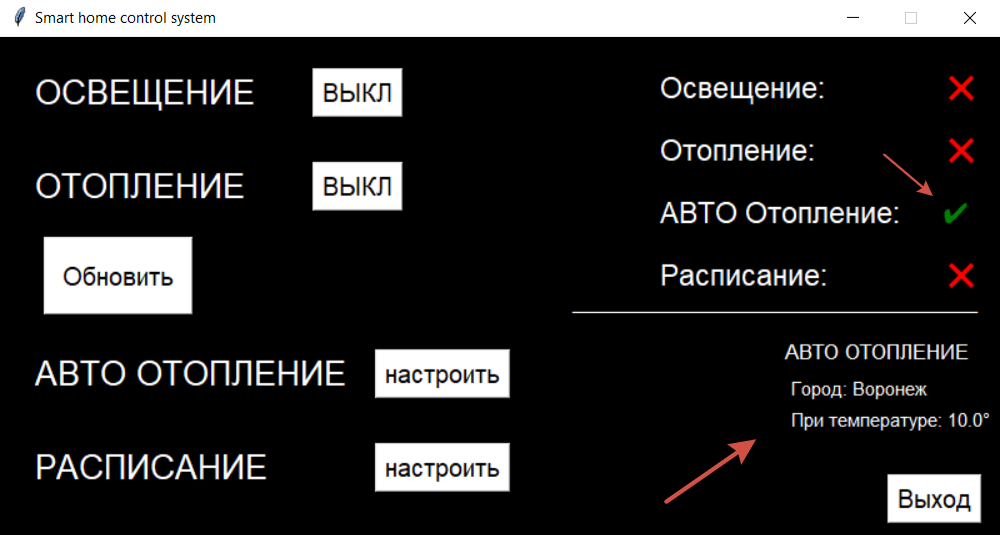
Нажимая кнопку Обновить, обновятся все параметры системы. На экране отобразятся самые актуальные состояния всех устройств.

Нажимая кнопку настроить у авто отопления, появится новый интерфейс настройки авто отопления.



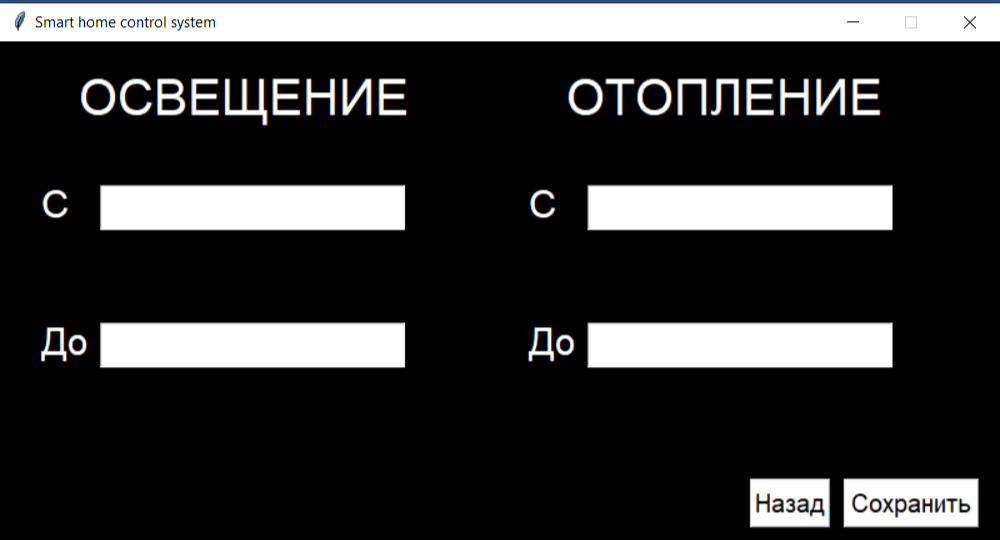
Здесь вам будет предложено, ввести город, температура которого будет браться за основу сравнения, и температуру, ниже которой нужно будет включать отопление. Можно вернуться обратно в главное меню, нажав кнопку Назад, или сохранить изменения, нажав кнопку Сохранить.

Если вы включили авто отопление, то поменяется его значок состояния, а так же появится его отдельный экран мониторинга:



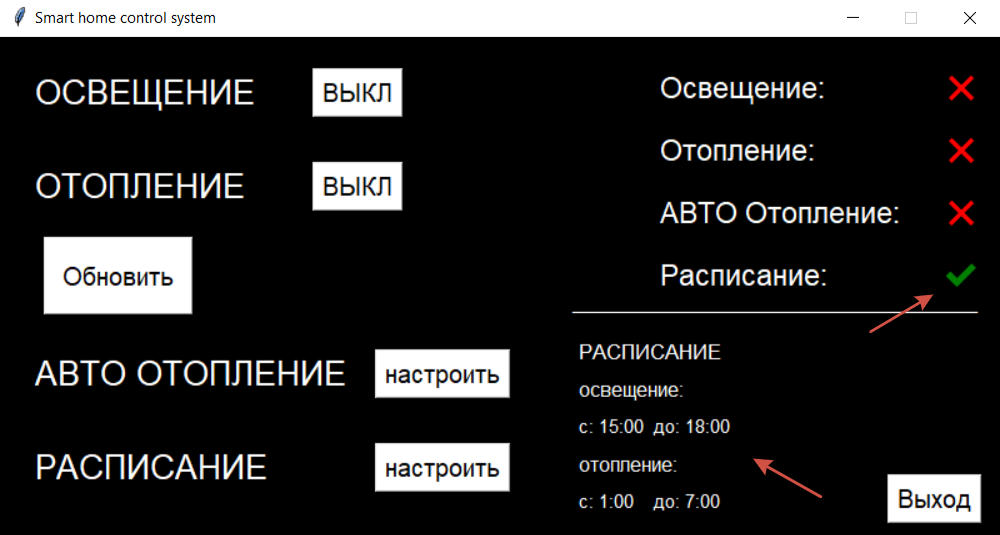
Если температура в городе будет ниже указанной, то будет автоматически включаться отопление.

Нажав на кнопку ‘настроить’ возле строки расписание, ваш интерфейс изменится и будет предложено настроить расписание освещения и отопления:



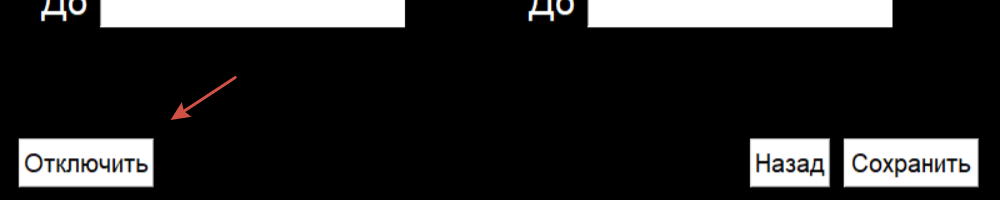
Здесь нужно ввести, с которого и до какого времени нужно включать освещение и отопление. Также, чтобы сохранить, просто нажимаем сохранить.

После включения расписания появится его отдельный экран мониторинга и изменится состояние.



Освещение и отопление будет автоматически включаться в определенное время, если хотите это увидеть на мониторах, то нажмите кнопку Обновить.

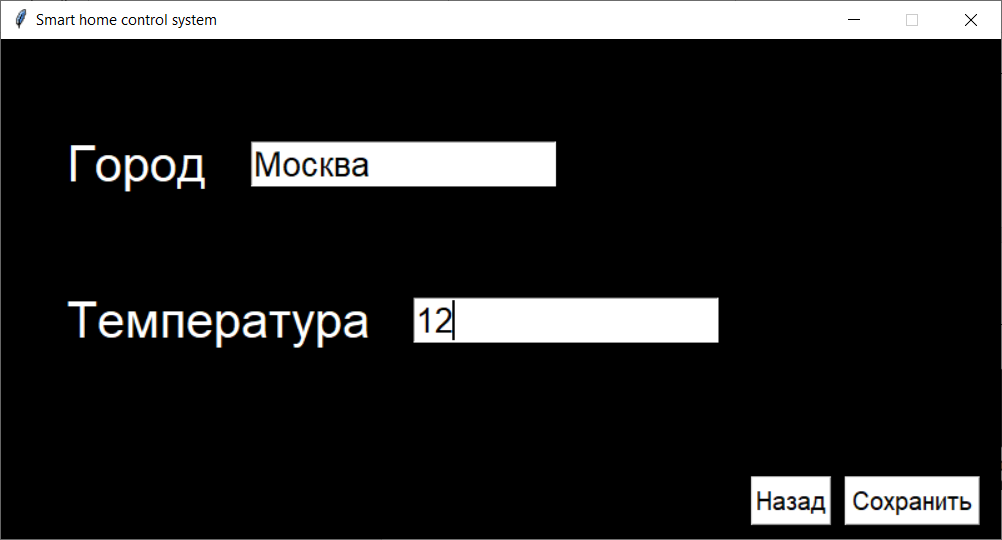
Чтобы отключить авто отопление или расписание, достаточно перейти по нужному параметру, нажав настроить, и в левом нижнем углу появится кнопки Отключить.



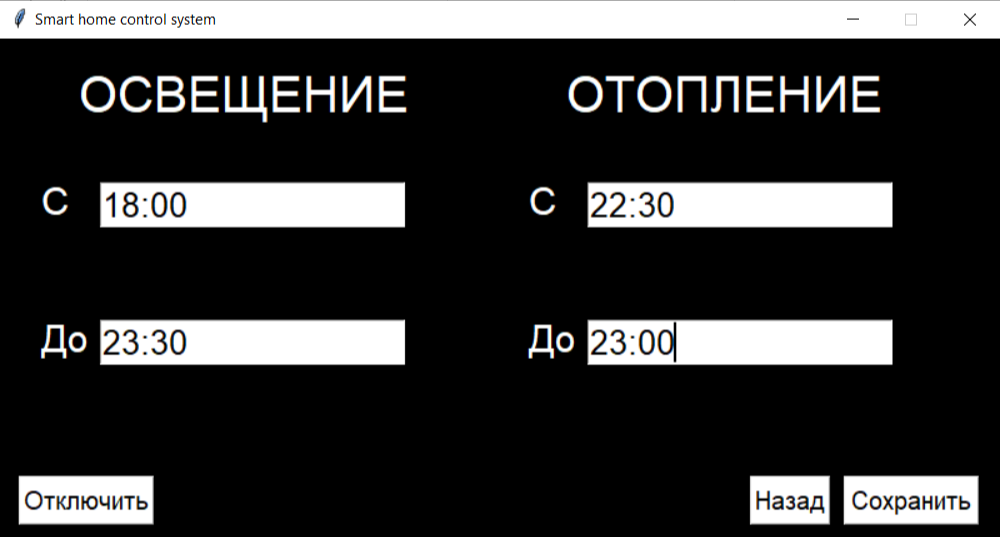
Ввод данных

Чтобы включить настройки авто отопление, нужно ввести город в формате: Название,

и желаемую температуру: 10. Температуру можно ввести как вещественное число (через точку), но оно будет округлено до десятых. Пример:



Чтобы включить расписания, нужно ввести время С которого будет включаться освещение и отопление, и До которого они будут работать в формате: *12:00*

**

Описание кода

Код содержится в файлах: main.py, Interface\_tkinter.py, Interface\_Functions.py, For\_basedata.py, Scrapping\_weather.py.

Используемые библиотеки: tkinter, datatime, sqlite3, bs4, requests, googletrans==3.1.0a0. Так же необходимо установить lxml.

**main.py**

В этом файле запускается код, создается главное окно, все нужные таблицы в базе данных и вызывается интерфейс главного меню.

**Interface\_tkinter.py**

В этом файле главные действия по созданию всех интерфейсов приложения. А именно имеются 3 класса:

* Main\_Interface – Создаётся интерфейс главного меню и функционал каждой кнопки. В него передаётся главное окно.
* Settings\_Schedule – Создается интерфейс настроек расписания и функционал каждой кнопки. В него передаётся главное окно и цвет фона.
* Settings\_Auto\_Heating – Создается интерфейс настроек авто отопления и функционал каждой кнопки. В него передаётся главное окно и цвет фона.

А также одна функция:

* start\_pend – обновляет систему. Сначала берёт нынешнее время и сравнивает его со временем, введённым пользователем. Затем очищает старый интерфейс и создает новый более актуальный.

**Interface\_Functions.py**

В этом файле содержатся вспомогательные функции для файла Interface\_tkinter.py, а именно 3 функции:

* delete\_vidgets – Удаляет все виджеты текущего окна. В неё передаётся это самое окно.
* set\_status\_monitoring – создает статусы устройств. В неё передаются главное окно и цвет фона интерфейса. Берет из базы данных состояния устройств и на их основе создает нужные виджеты для монитора.

✔️ / ❌

* destroy\_status\_vidgets – удаляет виджеты состояний устройств. Передаются виджеты для удаления.

**For\_basedata.py**

Файл нужен для взаимодействия кода с базой данных; состоит из 1 класса:

* State\_functions – Записывает состояния устройств в базу данных. При вызове определенного метода этого класса, записывается 1 или 0 (вкл/выкл) в базу под определенным устройством.

И 9 функций:

* create\_tables – Создает все нужные таблицы.
* set\_interface – Записывает какой последний интерфейс был пользователем. Это нужно для того, чтобы программа могла понимать в каком окне был пользователь если произойдет какое-то действие в приложении. (вспомогательная функция)
* monitoring – Получает актуальные данные из базы данных для пользователя.
* get\_data - Получает данные из базы данных о состояниях устройств для кода. (Техническая функция)
* set\_schedule – Записывает данные в таблицу настроек расписания.
* set\_auto\_heating - Записывает данные в таблицу настроек авто отопления. Принимает аргументы: город, нужная температура.
* get\_data\_auto\_heating – Получает данные из бд таблицы авто отопления. (Техническая функция).
* get\_data\_schedule - Получает данные из бд таблицы Расписания. (Техническая функция).
* get\_interface\_data – Возвращает интерфейс, который был открыт.

**Scrapping\_weather.py**

Файл нужен для обработки и парсинга температуры по введенному городу. Данные поступают с сайта: <https://www.ventusky.com>

Состоит из 2 функций:

* tranclate\_city – Получает и переводит город (язык).
* scrapping\_weather – Собирает температуру с сайта. В качестве аргумента принимает переведенный город.

Весь код работы хорошо закомментирован, что поможет разобраться при его чтении и анализе.