

ДЕПАРТАМЕНТ ОБРАЗОВАНИЯ ГОРОДА МОСКВЫ  
Государственное бюджетное общеобразовательное учреждение города Москвы  
«Школа № 1387»  
(ГБОУ школа № 1387)

## **Кейс №5: Сбор и обработка данных температуры**

Авторы работы:  
Марков Александр  
Прошина Евгения  
Аваков Артём Артурович  
11 И класс  
ГБОУ школа № 1387  
Научный руководитель:

Москва, 2020

## Оглавление

<b>Анализ технических требований</b> .....	2
<b>Задание</b> .....	2
<b>Условия выполнения</b> .....	2
<b>Пользовательский интерфейс</b> .....	3
<b>Язык программирования и программные средства</b> .....	3
<b>Языки программирования</b> .....	3
Сравнение языков программирования .....	3
Модули стандартной библиотеки Python .....	4
Библиотеки Python .....	4
<b>Программные средства</b> .....	4
<b>Описание основных этапов разработки</b> .....	4
<b>Первый этап</b> .....	4
<b>Второй этап</b> .....	4
<b>Третий этап</b> .....	5
<b>Четвёртый этап</b> .....	5
<b>Пятый этап</b> .....	5
<b>Структурная и функциональная схема</b> .....	5
Функциональные схемы: Parser .....	6
<b>Алгоритм работы программного продукта</b> .....	7
<b>Результаты разработки</b> .....	9
<b>Программный код</b> .....	9

## Анализ технических требований

### Задание

Реализовать программный модуль для сбора, хранения и обработки данных с удалённых температур данных

### Условия выполнения

Посредством специализированного сервиса, расположенного по адресу [http://dt.miet.ru/ppo\\_it](http://dt.miet.ru/ppo_it), осуществить сбор данных об уличной температуре в 16 городах. Необходимо использовать показатели датчиков, находящихся в 10 квартирах не менее, чем пяти районов города. Время осуществления – 48 часов реального времени.

Обращение к сервису происходит не реже, чем один раз в 10 минут.

Полученные данные хранятся с помощью реляционной СУБД, реализованной на основе ER-модели.

Взаимодействие с данными осуществляется через пользовательский интерфейс, который по запросу пользователя отображает информацию о температуре в квартирах и на улице в виде графика/диаграммы (см. пункт [пользовательский интерфейс](#))

Пользовательский интерфейс является кроссплатформенным, а также организован в соответствии со стандартами построения UI

Для облегчения работы с вносимыми в программный код изменениями используется система управления версиями git

### Пользовательский интерфейс

Согласно регламенту испытаний функционал UI подразумевает выведение следующих данных в виде графика/диаграммы/величины:

- Данные температуры в реальном времени в определённой квартире
- График изменения уличной температуре на протяжении суток реального времени в одном из городов
- График изменения средней температуры в квартирах в одном из городов на протяжении суток реального времени
- График изменения температуры в одной квартире в каждом из городов
- Диаграмма максимальных температур в квартирах в каждом из районов (не менее пяти) одного города

## Язык программирования и программные средства

### Языки программирования

Учитывая изложенные выше особенности технического задания, было решено, что в качестве основного языка программирования будет использоваться Python.

При выборе важную роль сыграли такие его качества, как минималистичный синтаксис ядра и богатая стандартная библиотека, позволяющая работать с высокоуровневыми структурами данных и взаимодействовать со многими сетевыми протоколами, в частности HTTP.

Управление БД осуществляется посредством СУБД SQLite ввиду того, что его легко использовать при кроссплатформенном переносе, а также он очень надёжен с точки зрения программного кода.

### Сравнение языков программирования

	Python	C++	Java
Опыт работы	+	-	-

Простота синтаксиса и удобство	+	-	-
Производительность	-	+	+
Работа с БД	+	-	-
Динамическая типизация	+	-	-

*Таблица 1*

Производительность языка не играла важную роль при выборе языка, так как программный продукт используется малым числом лиц и объёмы обрабатываемых данных относительно малы

### Модули стандартной библиотеки Python

- модуль requests – инструмент составления HTTP и GET-запросов для взаимодействия с сервером
- модуль time – для работы с реальным/серверным временем
- модуль sqlite3 – кроссплатформенное средство работы с БД

### Библиотеки Python

- Matplotlib – осуществляет визуализацию информации из БД в UI
- IPyWidgets – интерактивные HTML виджеты для Jupyter Notebook

### Программные средства

Дистрибутив Anaconda предуснавливает инструмент интерактивной разработки Jupyter Notebook, удобный с точки зрения разработки и использования интерфейса, в том числе он поддерживает создание графического интерфейса для пользователя. В качестве IDE используется Spyder из дистрибутива Anaconda. Одна из важных особенностей этой среды разработки – это интеграция с научными библиотеками Python, к примеру Matplotlib

## Описание основных этапов разработки

### Первый этап

С помощью интернет-ресурсов мы изучили синтаксис и принципы работы описанных выше модулей и библиотек языка Python, СУБД SQLite, а также ознакомились с основами работы в программных средствах Jupyter Notebook и Spyder

### Второй этап

Для реализации программного кода выбраны методы структурного программирования. Составлены концепции функций, образующих модуль взаимодействия с сервером и обработкой полученной информации и модуль

пользовательского интерфейса и визуализации данных (далее «модуль 1» и «модуль 2» соответственно; см. раздел [Структурная и функциональная схема](#)).

### Третий этап

На сайте <https://github.com/> создана система контроля версий продукта, доступная по следующей ссылке: <https://github.com/CangCiwei/PredProf>. Составлен план выполнения технического задания согласно функциональности продукта.

### Четвёртый этап

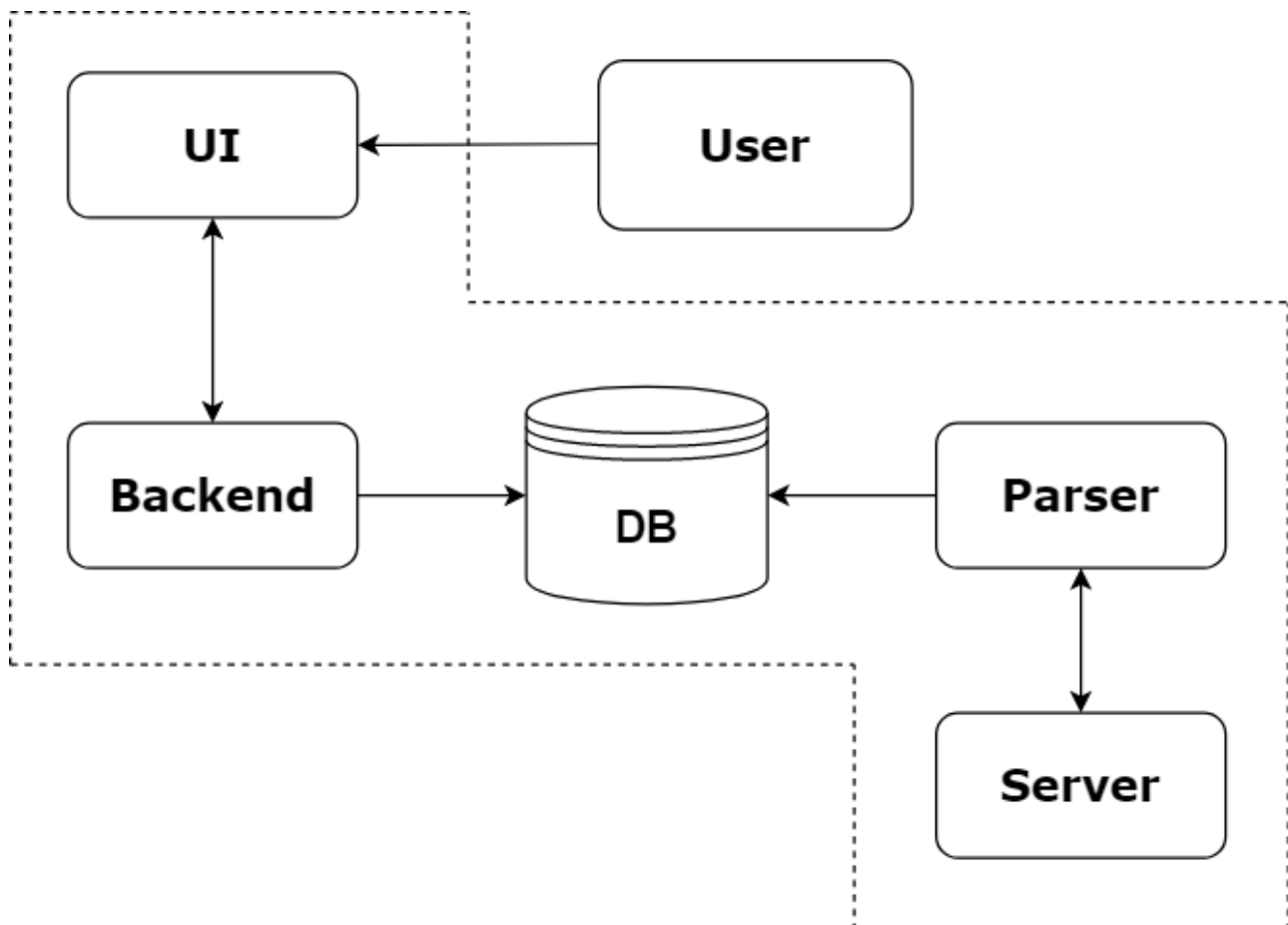
Проектируется UI. Осуществляется написание программного кода для модуля 1 и модуля 2. Код комментируется. Каждая составная функцию тестируется на корректность возвращаемых данных.

### Пятый этап

Тестируется функциональность программы. Проводятся необходимые корректировки программного кода (см. [Третий этап](#)) и повторные тестирования.

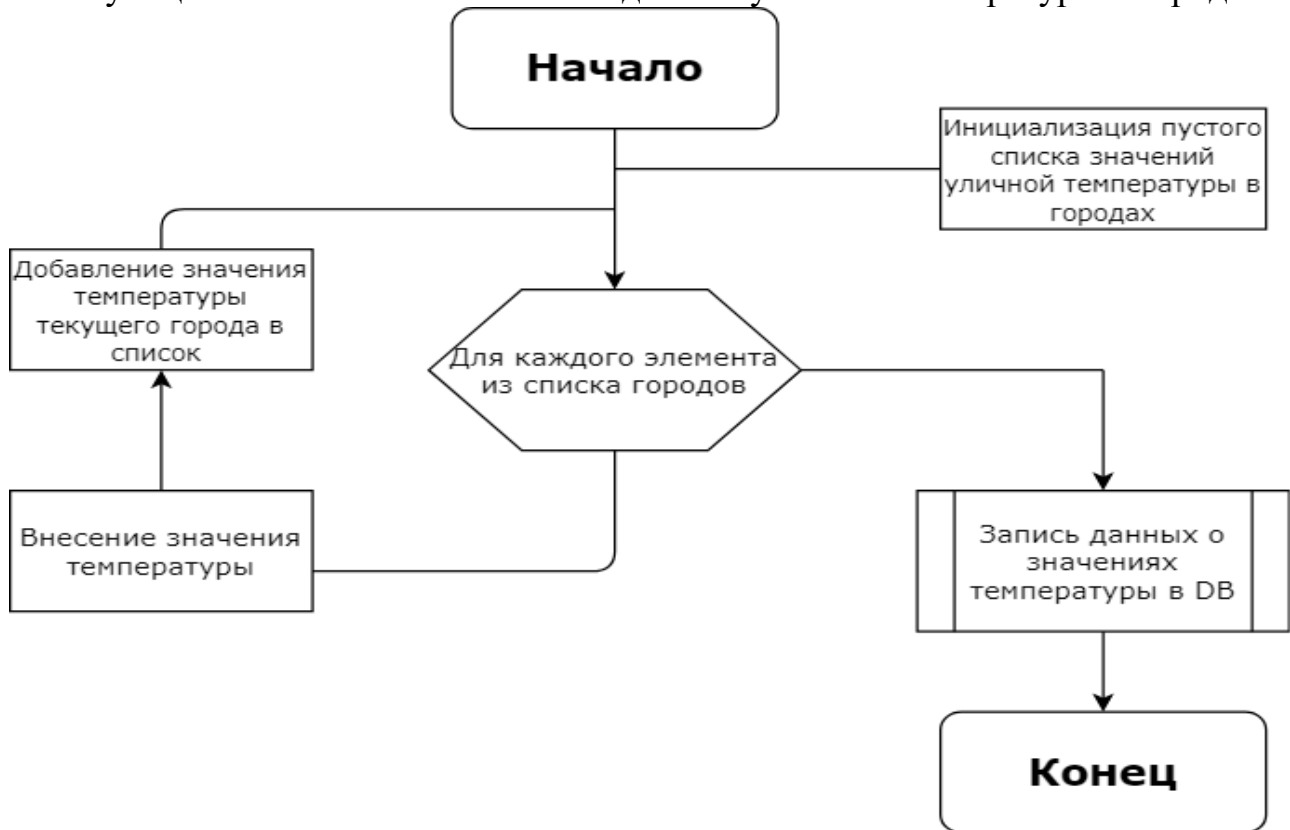
## Структурная и функциональная схема

### 1. Структурная схема работы алгоритма

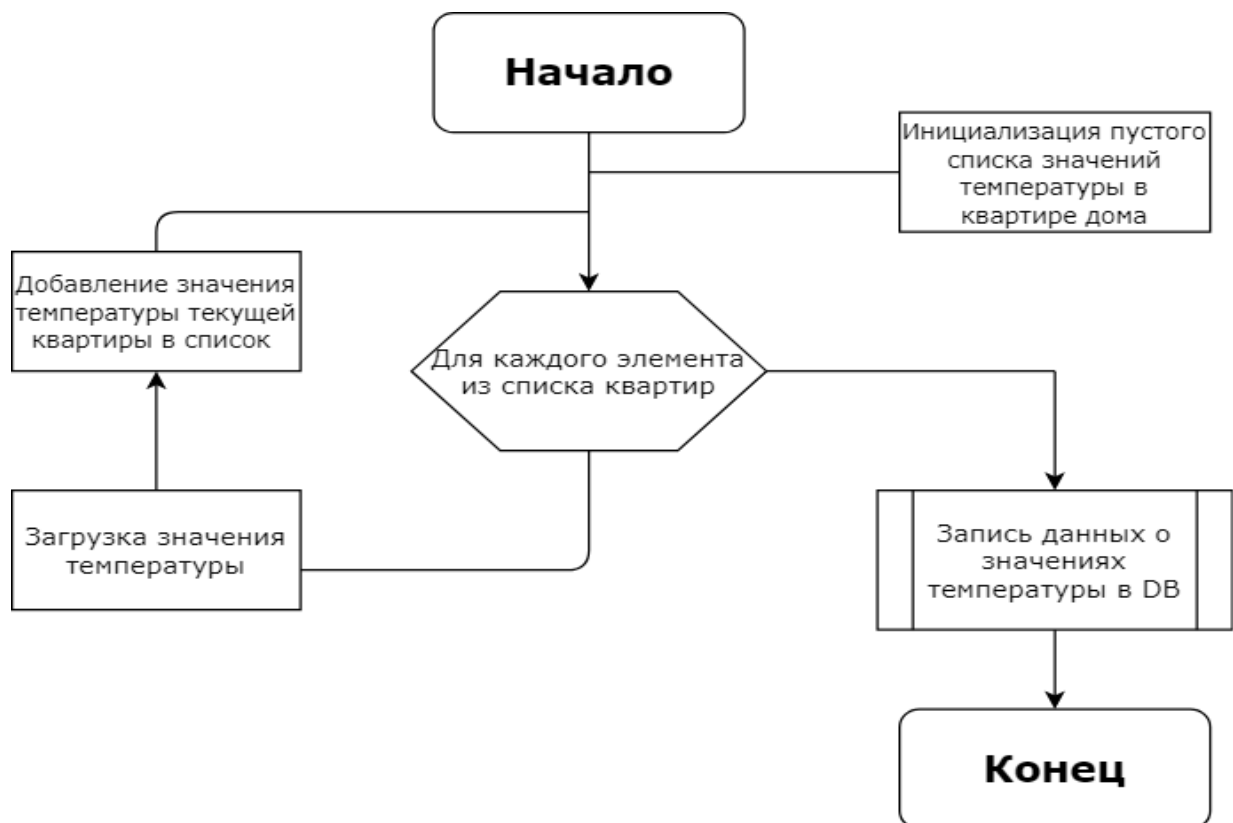


## Функциональные схемы: Parser

### 2. Функциональная схема: внесения данных уличной температуры в городах



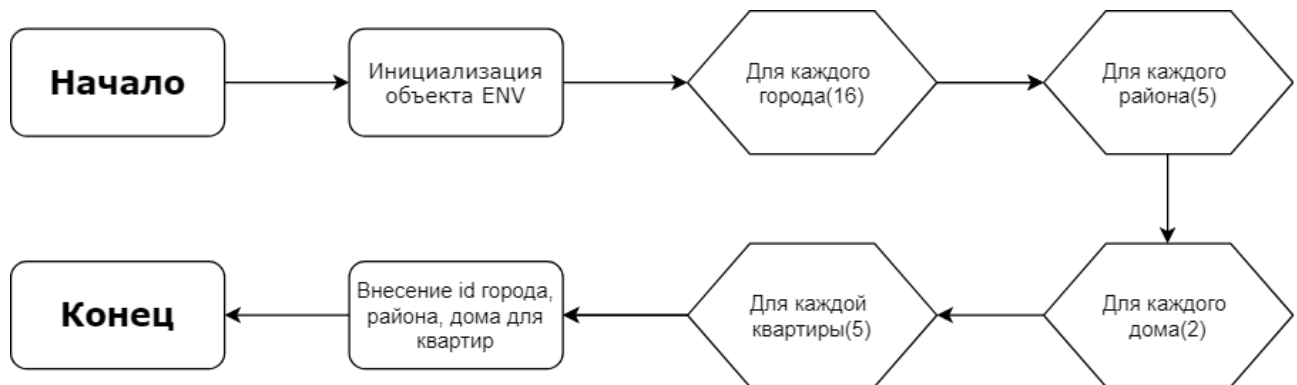
### 3. Функциональная схема: внесение значений температуры в квартире



## Алгоритм работы программного продукта

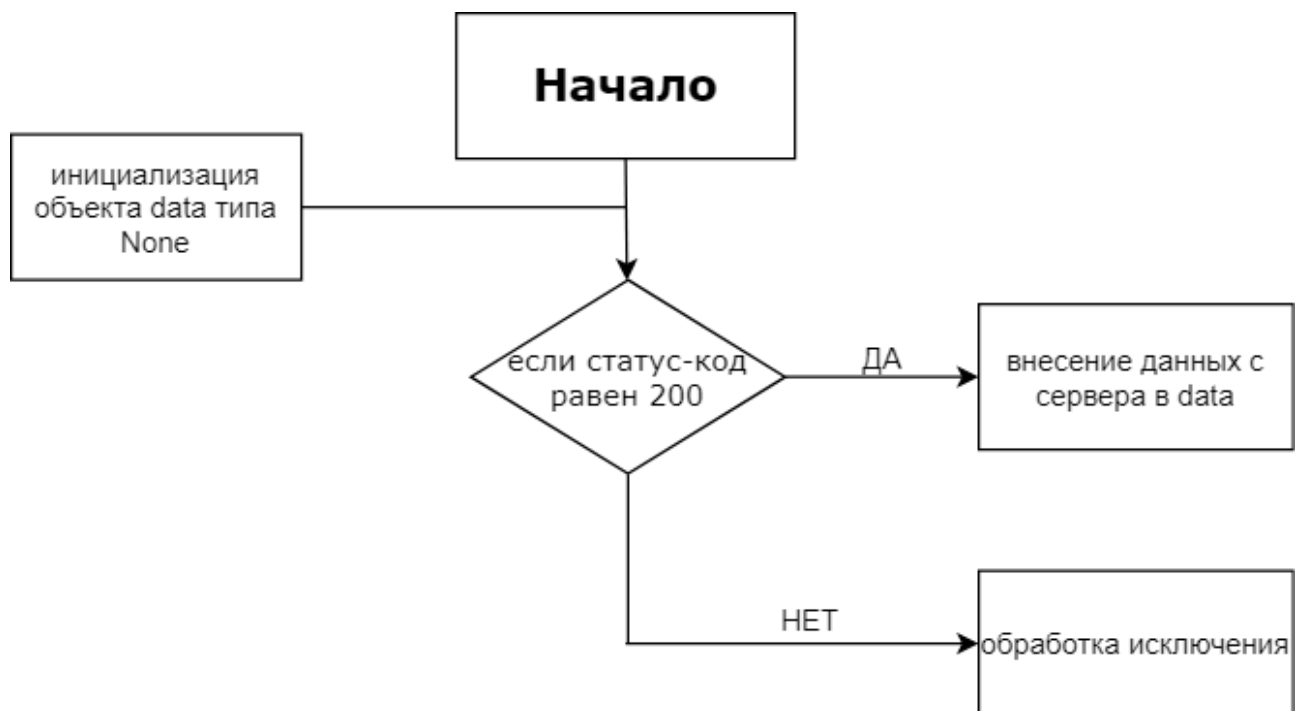
/// Здесь должно быть общее описание работы алгоритма ///

1. Загрузка «целей» - адресов квартир, из которых будут считываться данные температуры



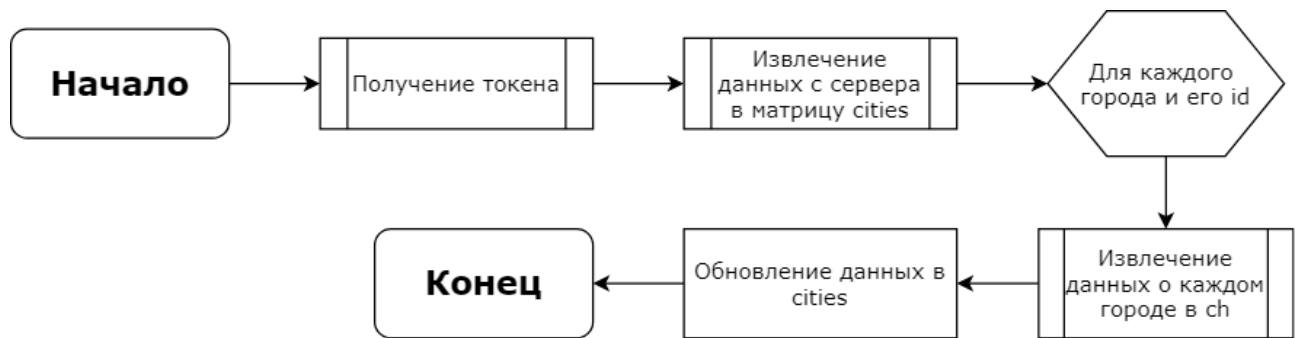
ENV – это объект типа dict, содержащий список целей targets, url-адрес сервиса, token и время работы программы.

2. Извлечение данных с сервера



Алгоритм возвращает объект data

3. Извлечение подробных данных о каждом городе



Дополнительно, перед использованием объектов cities и ch они проверяются на корректность (не тип None). Алгоритм возвращает cities в удобном для обработки виде

#### 4. Сбор данных о каждой цели



Алгоритм возвращает список показаний температуры в квартирах

#### 5. Основной алгоритм (нет в github)



## Результаты разработки

Реализован программный продукт, позволяющий выводить данные температуры в различных квартирах и регионах (см. пункт [Пользовательский интерфейс](#)). Взаимодействие с пользователем происходит через специализированный UI. Интерфейс является самоадаптирующимся для различных ОС/имеет единый стиль, общий для всех ОС/гибридное решение. Выполнены все пункты технического задания, описанные в начале документации.

## Программный код