ПРАКТИЧЕСКАЯ РАБОТА№1

«Условные конструкции и операторы выбора»

Цель работы: Научиться работать с условными конструкциями и операторами выбора, решить задачи для закрепления.

Программа №1

Дано целое число. Если оно является положительным, то прибавить к нему 1; в противном случае не изменять его. Вывести полученное число.

Реализуем пользовательский ввод числа и проверяем на положительность. Если число больше нуля, то прибавляем к нему единицу, если же нет, то ничего с ним не делаем.

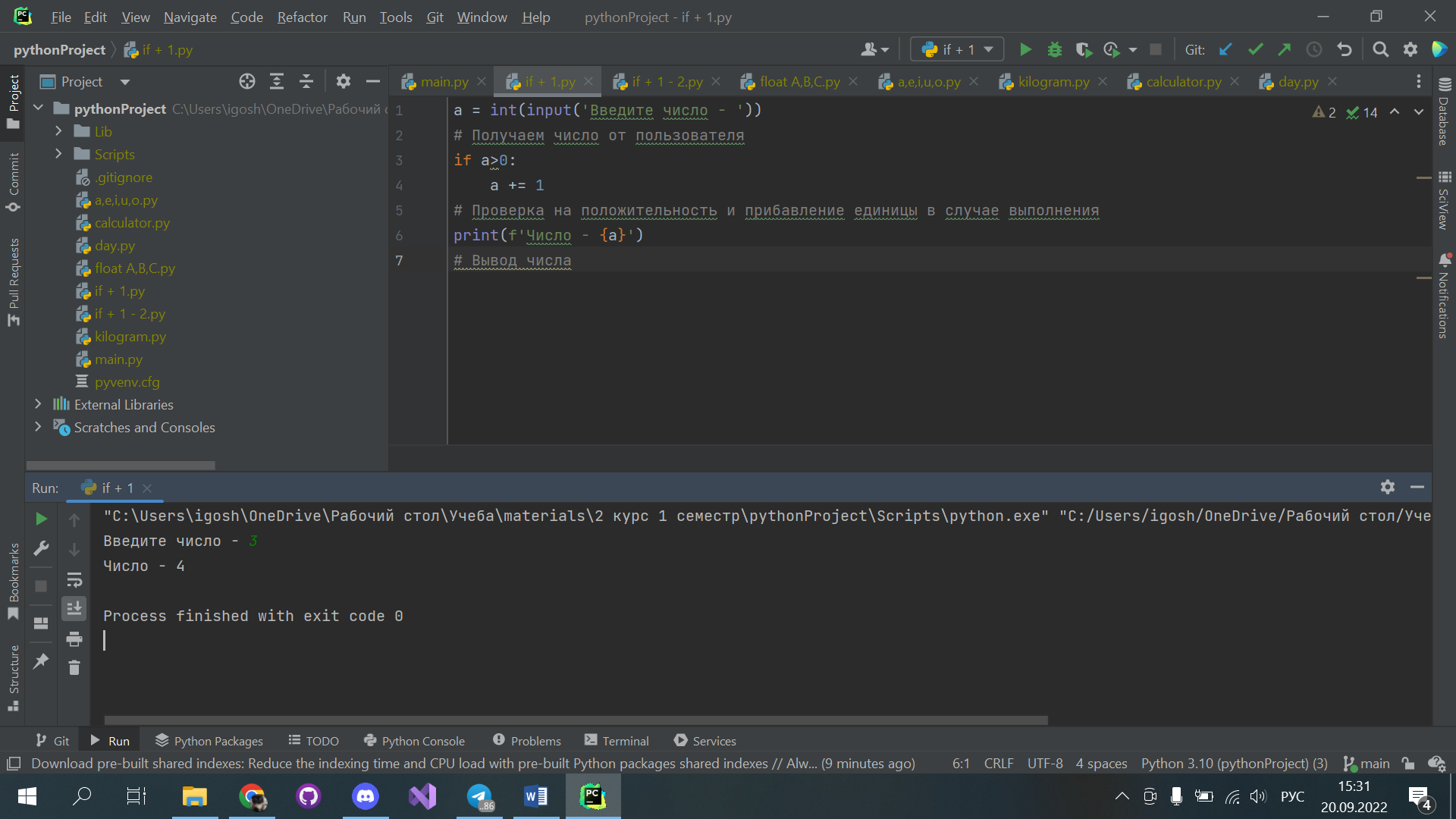


Рисунок – Пользовательский ввод и условная конструкция

Результат работы:

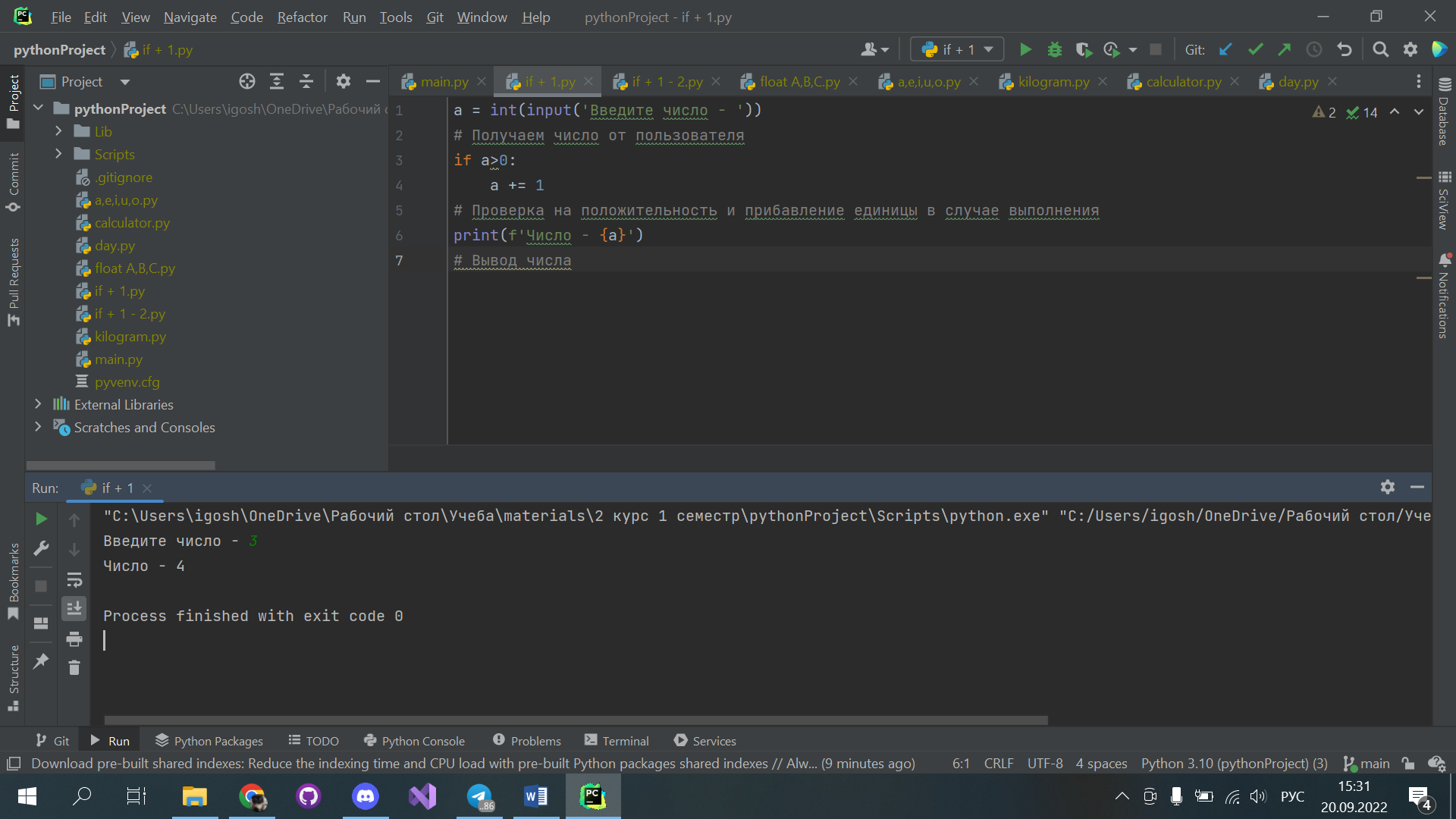


Рисунок – Вывод полученного числа

Программа №2

Дано целое число. Если оно является положительным, то прибавить к нему 1; если отрицательным, то вычесть из него 2; если нулевым, то заменить его на 10. Вывести полученное число.

Реализуем пользовательский ввод числа и проверяем на положительность. Если число больше нуля, то прибавляем к нему единицу, если же нет, то проверяем на отрицательность, и вычитаем из него двойку, иначе приравниваем к десяти.

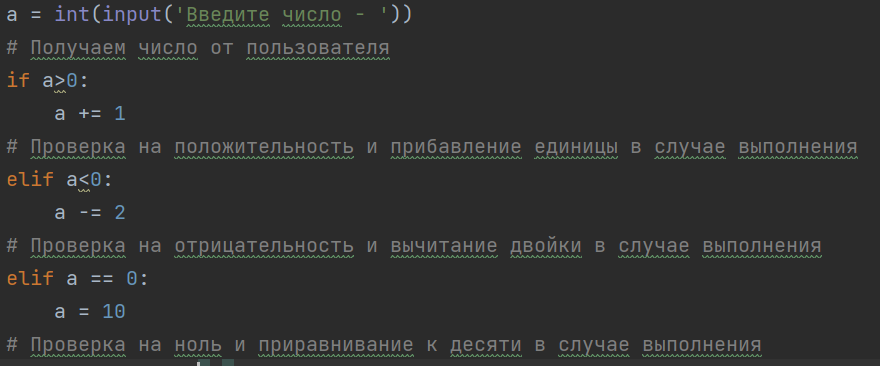


Рисунок – Пользовательский ввод, проверка на положительность, отрицательность и ноль

Результат работы:

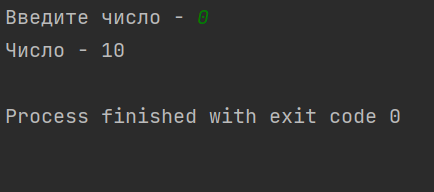


Рисунок – Вывод полученного числа

Программа №3

Даны три переменные вещественного типа: A, B, C. Если их значения упорядочены по возрастанию, то удвоить их; в противном случае заменить значение каждой переменной на противоположное. Вывести новые значения переменных A, B, C.

Реализуем пользовательский ввод трех вещественных чисел и проверяем на положительность. Если число больше нуля, то прибавляем к нему единицу, если же нет, то проверяем на отрицательность, и вычитаем из него двойку, иначе приравниваем к десяти.

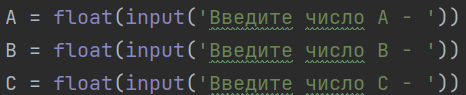


Рисунок – Пользовательский ввод

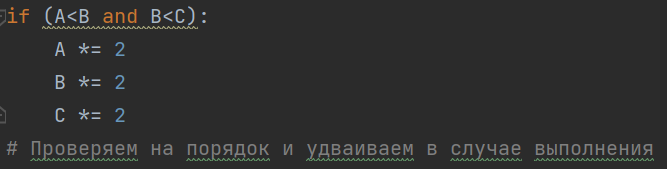


Рисунок – Проверка на возрастание

Результат работы:

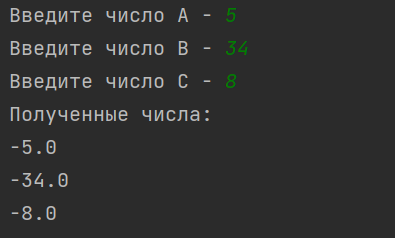


Рисунок – Вывод полученных чисел

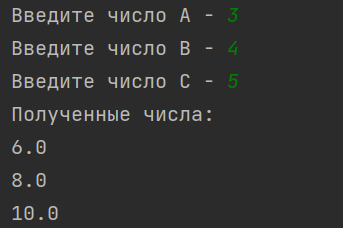


Рисунок – Вывод полученных чисел

Программа №4

Разработайте программу, запрашивающую у пользователя букву латинского алфавита. Если введенная буква входит в следующий список (a, e, i, o или u), необходимо вывести сообщение о том, что эта буква гласная. Если была введена буква y, программа должна написать, что эта буква может быть как гласной, так и согласной. Во всех других случаях должно выводиться сообщение о том, что введена согласная буква.

Реализуем пользовательский ввод латинской буквы, проверяем на гласную, согласную, либо на гласную и согласную одновременно и выводим в зависимости от этого сообщение.

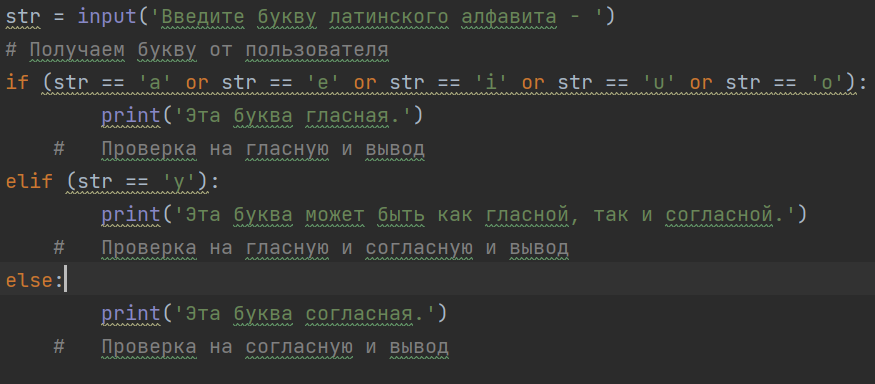


Рисунок – Пользовательский ввод, проверка на гласную, согласную, гласную и согласную одновременно и вывод сообщения

Результат работы:

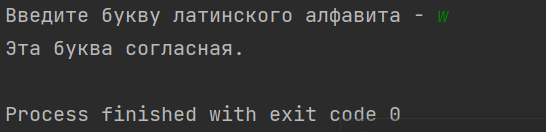


Рисунок – Вывод сообщения о гласности либо согласности буквы

Программа №5

Единицы массы пронумерованы следующим образом: 1 — килограмм, 2 — миллиграмм, 3 — грамм, 4 — тонна, 5 — центнер. Дан номер единицы массы (целое число в диапазоне 1–5) и масса тела в этих единицах (вещественное число). Найти массу тела в килограммах.

Реализуем пользовательский ввод номера единицы массы и массу самого тела.

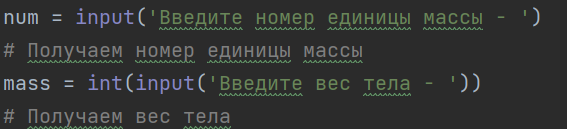


Рисунок – Пользовательский ввод номера единицы массы и веса тела

Проверяем номер единицы массы и выполняем преобразование в килограммы, после чего выводим результат.

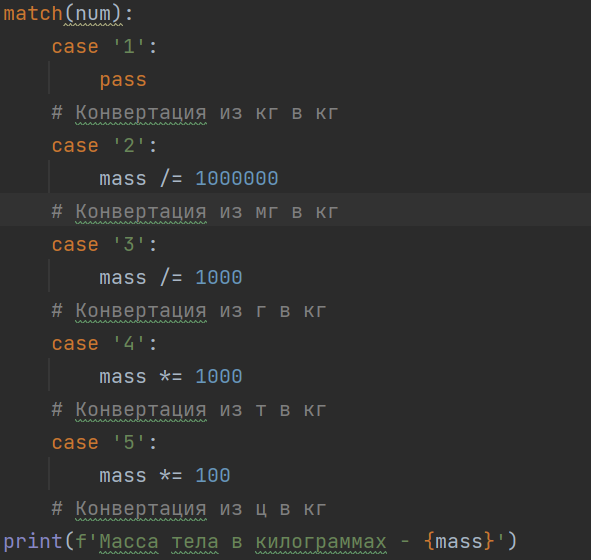


Рисунок – Проверка единицы массы, перевод в килограммы, вывод массы

Результат работы:

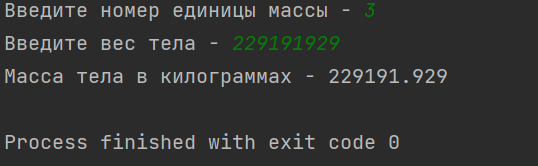


Рисунок – Вывод полученной массы

Программа №6

Необходимо реализовать консольный калькулятор со всеми основными арифметическими операциями. Пользователь вводит 2 числа с клавиатуры и выбирает операцию из списка. Если пользователь выбрал не существующую операцию необходимо вывести соответствующее сообщение.

Реализуем пользовательский ввод двух чисел и арифметической операции.

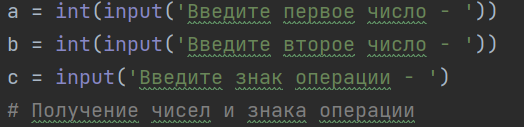


Рисунок – Получение чисел и знака операции

Проверяем знак операции и выполняем вычисления в зависимости от него, выводя результат на экран. Если такой знак отсутствует, выводим соответствующее сообщение на экран.

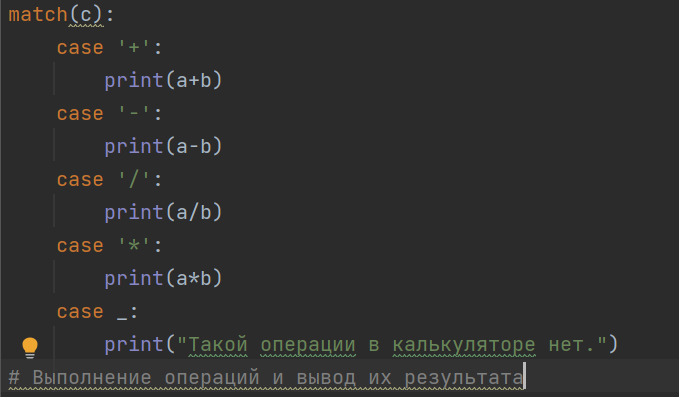


Рисунок – Проверка знака и выполнение операции в зависимости от него

Результат работы:

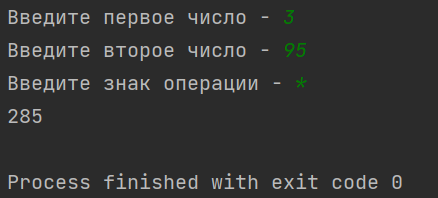


Рисунок – Вывод полученного в ходе вычисления числа

Программа №7

Дано целое число в диапазоне 1–7. Вывести строку — название дня недели, соответствующее данному числу (1 — «понедельник», 2 — «вторник» и т. д.).

Реализуем пользовательский ввод целого числа.



Рисунок – Пользовательский ввод числа

Проверяем полученное от пользователя число и выводим название дня недели в зависимости от введенного числа.

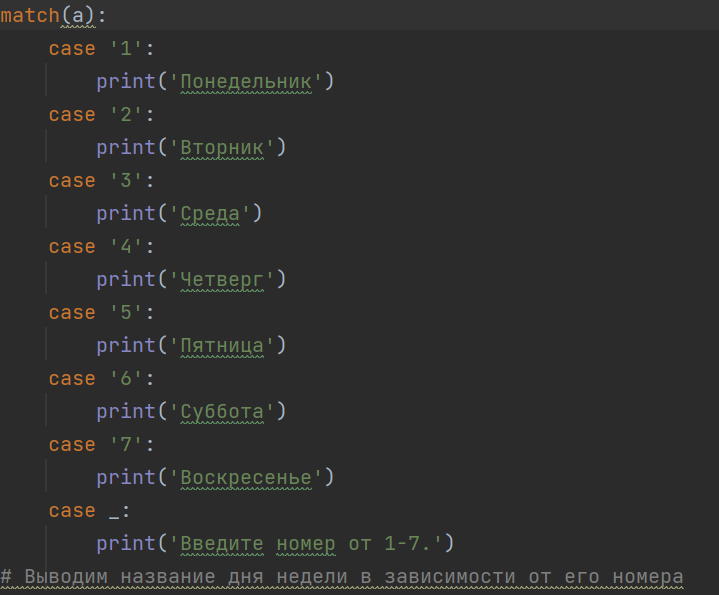


Рисунок – Проверка номера дня и вывод его названия

Результат работы:

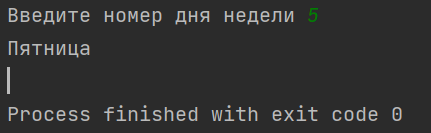


Рисунок – Вывод дня недели в зависимости от номера

Вывод: Научились работать с условными конструкциями и операторами вывода, выполняя задачи для развития заданных навыков работы с определенными условиями.

ПРАКТИЧЕСКАЯ РАБОТА №2

«Циклы»

Цель работы: научиться использовать циклы, понять их принцип работы, решить задачи.

Программа №1

Необходимо написать программу для подсчета среднего значения всех введенных пользователем чисел. Индикатором окончания ввода будет служить ноль. При этом программа должна выдавать соответствующее сообщение об ошибке, если первым же введенным пользователем значением будет ноль.

Необходимо реализовать пользовательский ввод через цикл, создав для этого необходимые переменные для счета количества произведенных циклов и для суммы. В цикл добавляем условия работы с нулем.

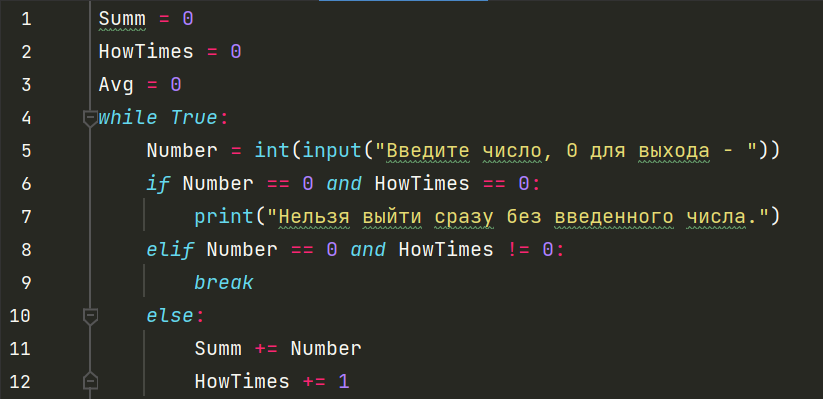


Рисунок – Создание переменных, цикл с пользовательским вводом, условия с нулём

После выхода из цикла находим среднее арифметическое, деля сумму на счетчик.

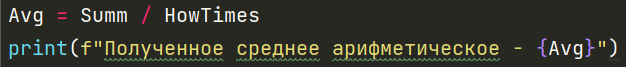


Рисунок – Расчет и вывод среднего арифметического

Результат работы:

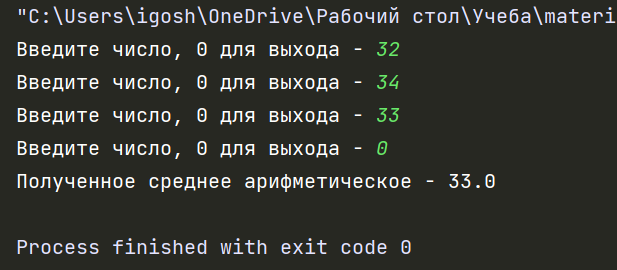


Рисунок – Результат работы

Программа №2

В зоопарке цена входного билета зависит от возраста посетителя. Дети до двух лет допускаются бесплатно. Дети в возрасте от трех до 12 лет могут посещать зоопарк за $14,00. Пенсионерам старше 65 лет вход обойдется в $18,00, а обычный взрослый билет стоит $23,00. Напишите программу, которая будет запрашивать возраст всех посетителей в группе (по одному за раз) и выводить общую цену билетов для посещения зоопарка этой группой. Общую цену билетов стоит выводить в формате с двумя знаками после запятой.

Необходимо реализовать пользовательский ввод кол-ва посетителей зоопарка и создать переменные необходимые для работы цикла.

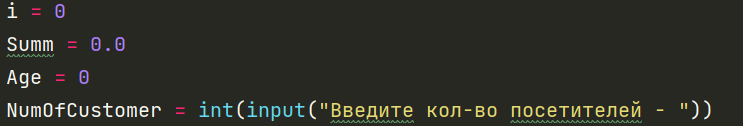


Рисунок – Создание переменных, пользовательский ввод

Далее нужно создать цикл, зависящий от количества посетителей. Он будет просить по очереди вводить возраст посетителей и просчитывать цену за билет, записывая их в сумму чека, и вывести ее после окончания цикла.

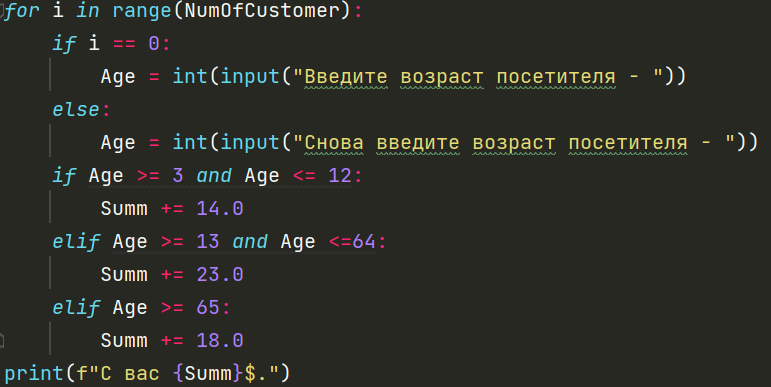


Рисунок – Цикл, Пользовательский ввод с условиями, добавление цены билета, вывод суммы

Результат работы:

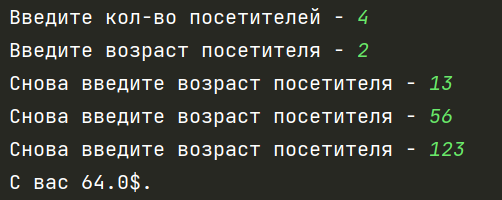


Рисунок – Результат работы

Программа №3

Строка называется палиндромом, если она пишется одинаково в обоих направлениях. Например, палиндромами в английском языке являются слова «anna», «civic», «level», «hannah». Напишите программу, запрашивающую у пользователя строку и при помощи цикла определяющую, является ли она палиндромом.

Необходимо создать пользовательский ввод слова, необходимого для проверки, а также создать переменные, необходимые для работы цикла.

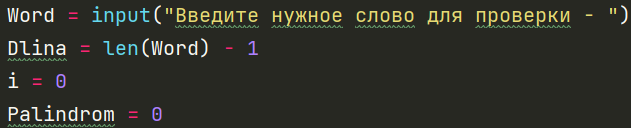


Рисунок – Пользовательский ввод, создание переменных

Далее создаем цикл, последовательно сравнивающий две части слова. Если они сходятся, то прибавляет 1 к счетчику, если же нет, то придает переменной, отвечающей за палиндром единицу, и прекратит выполнение цикла.

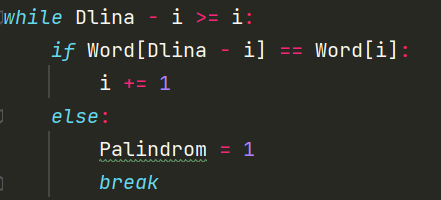


Рисунок – Цикл, проверяющий слово

После выводим сообщение о том, является ли слово палиндромом или нет.

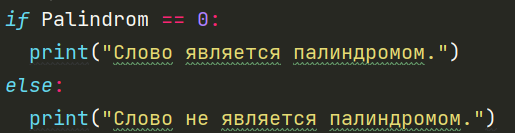


Рисунок – Вывод сообщения

Результат работы:

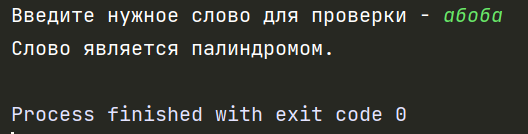


Рисунок – Результат работы

Программа №4

В магазине была объявлена скидка размером 60 % на ряд товаров, и для того чтобы покупатели лучше ориентировались, владелец торговой точки решил вывесить отдельную таблицу со скидками с указанием уцененных товаров и их оригинальных цен. Используйте цикл для создания подобной таблицы, в которой будут исходные цены, суммы скидок и новые цены для покупок на сумму $4,95, $9,95, $14,95, $19,95 и $24,95. Убедитесь в том, что суммы скидки и новые цены отображаются с двумя знаками после запятой.

Необходимо создать списки с ценами без скидки, суммы скидок, со скидкой и создать вспомогательные переменные для цикла.

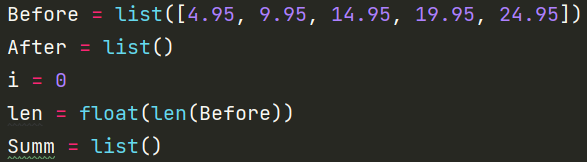


Рисунок – Создание списков и переменных

Далее создаем цикл для заполнения массива новыми ценами, применяя скидку к прошлым ценам, округляя результат до двух знаков после запятой, вычисляем разницу и записываем в сумму.

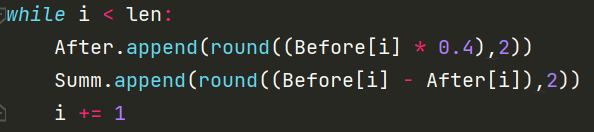


Рисунок – Цикл, заполнение двух листов

После выводим полученные списки.

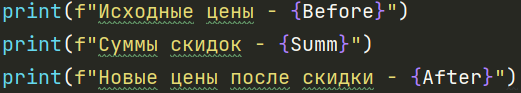


Рисунок – Вывод полученных списков

Результат работы:

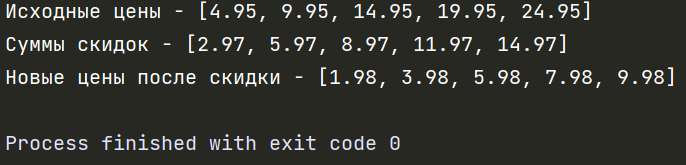


Рисунок – Результат работы

Программа №5

Необходимо от пользователя принять строку. Если элемент строки является четным то перевести символ в нижний регистр. Если элемент строки является не четным перевести символ в верхний регистр.

Реализуем пользовательский ввод и создаем счетчик переменной.



Рисунок – Пользовательский ввод и счетчик

Переводим в различные регистры поочередно буквы, используя цикл и выводим полученное слово.

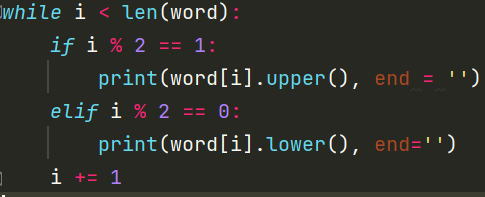


Рисунок – Цикл и вывод слова

Результат работы:

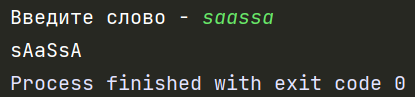


Рисунок – Результат работы

Вывод: научились работать с циклами, поняли их принцип работы, решили несколько задач, связанных с ними.

ПРАКТИЧЕСКАЯ РАБОТА №3

Цель работы: научиться работать со списками, словарями и кортежами, использовать функции для их форматирования, решая задачи.

Программа №1

Дан массив, состоящий из n кортежей. Найдите максимальное число повторений элементов в кортежах.

Необходимо создать массив, состоящий из кортежей, и заполнить кортежи, состоящие в нем, а также создать переменные, необходимые для корректной работы цикла.

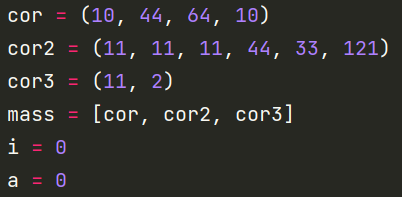


Рисунок – Создание кортежей и переменных

Далее создаем цикл, высчитывающий максимальное число повторений элементов в массиве и выводим результат.

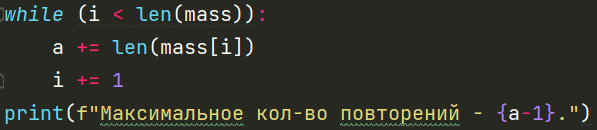


Рисунок – Цикл и вывод

Результат работы:

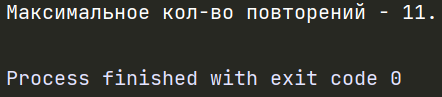


Рисунок – Результат работы

Программа №2

Дан словарь данных шаблона "фамилия: {дата рождения, email}", а также дана дата проверки. вывести все фамилии пользователей, которые родились позже этой даты.

Необходимо реализовать пользовательский ввод даты и создать массив, состоящий из словарей.

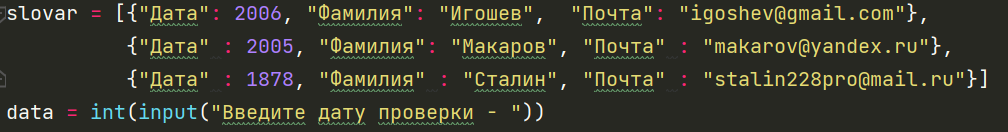


Рисунок – Пользовательский ввод, массив, словари

Далее создаем цикл проверки массива и сравниваем дату пользователя и словаря, в случае выполнения выводим фамилию словаря.

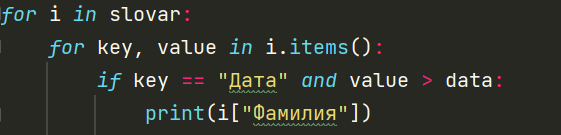


Рисунок – Цикл, вывод

Результат работы:

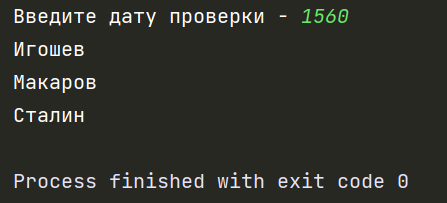


Рисунок – Результат работы

Программа №3

Дана строка, состоящая из некоторого количества слов. вывести слова из этой строки без их повторений.

Необходимо реализовать пользовательский ввод строки и вывод результата ее форматирования.



Рисунок – Вывод отформатированного предложения

Результат работы:

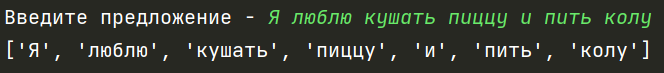


Рисунок – Результат работы

Программа №4

Напишите программу, которая будет запрашивать у пользователя целочисленные значения и сохранять их в виде списка. Индикатором окончания ввода значений должен служить ноль. Затем программа должна вы- вести на экран все введенные пользователем числа (кроме нуля) в порядке возрастания – по одному значению в строке. Используйте для сортировки либо метод sort, либо функцию sorted.

Для начала следует создать список и цикл, заполняющийся с помощью пользователя целыми числами, где ввод нуля является выходом из цикла.

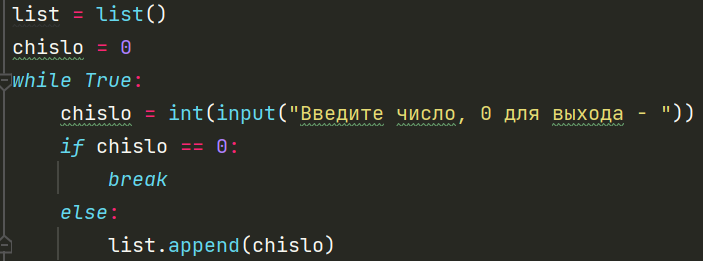


Рисунок – Создание списка, цикл с пользовательским вводом

Далее следует сортировка списка и его последовательный вывод.

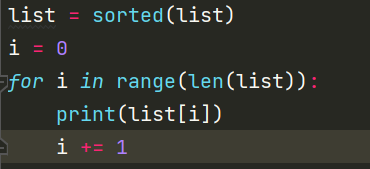


Рисунок – Сортировка, цикл вывода

Результат работы:

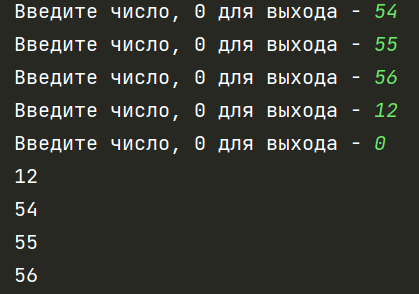


Рисунок – Результат работы

Программа №5

Напишите программу, которая, как и в предыдущем случае, будет запрашивать у пользователя целые числа и сохранять их в виде списка. Индикатором окончания ввода значений также должен служить ноль. На этот раз необходимо вывести на экран введенные значения в порядке убывания.

Следует, как и в прошлом задании создать список и цикл для его заполнения.

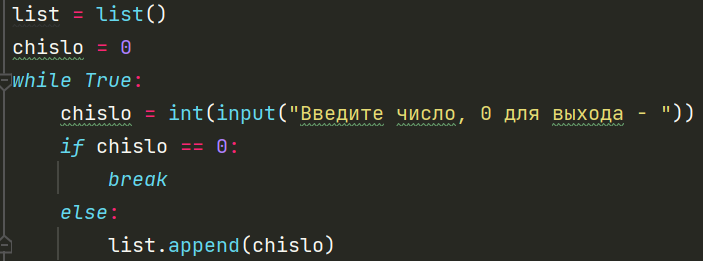


Рисунок – Список, цикл с пользовательским вводом

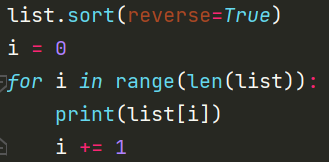


Рисунок – Сортировка и последовательный вывод

Результат работы:



Рисунок – Результат работы

Программа №6

Напишите программу, запрашивающую у пользователя целые числа, пока он не оставит строку ввода пустой. После окончания ввода на экран должны быть выведены сначала все отрицательные числа, которые были введены, затем нулевые и только после этого положительные. Внутри каждой группы числа должны отображаться в той последовательности, в которой были введены пользователем. Например, если он ввел следующие числа: 3, -4, 1, 0, -1, 0 и -2, вывод должен оказаться таким: -4, -1, -2, 0, 0, 3 и 1. Каждое значение должно отображаться на новой строке.

Для начала следует создать списки и заполнить их с помощью цикла, реализовав выход из него с помощью ввода пустой строки.

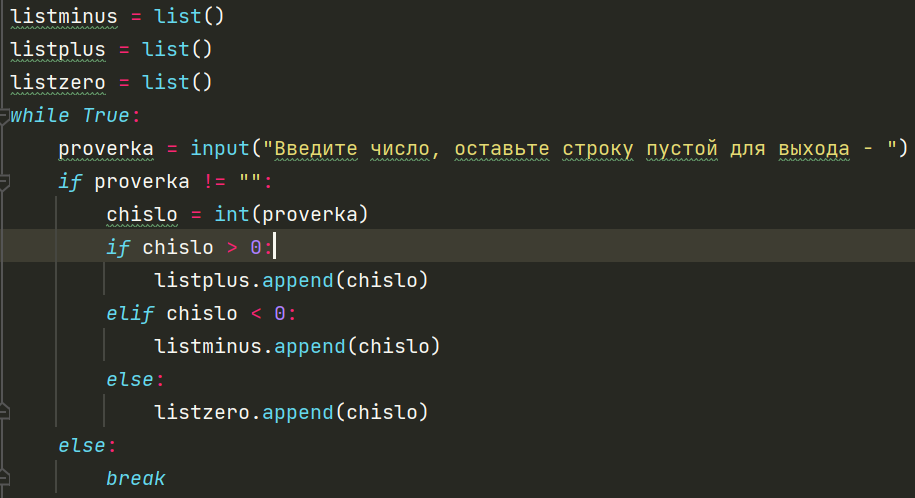


Рисунок – Создание списков, их заполнение через цикл

Далее делаем последовательный вывод списков через цикл с каждой новой строки.

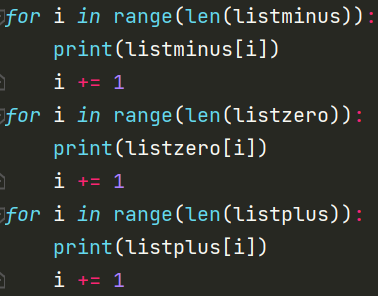


Рисунок – Вывод списков

Результат работы:

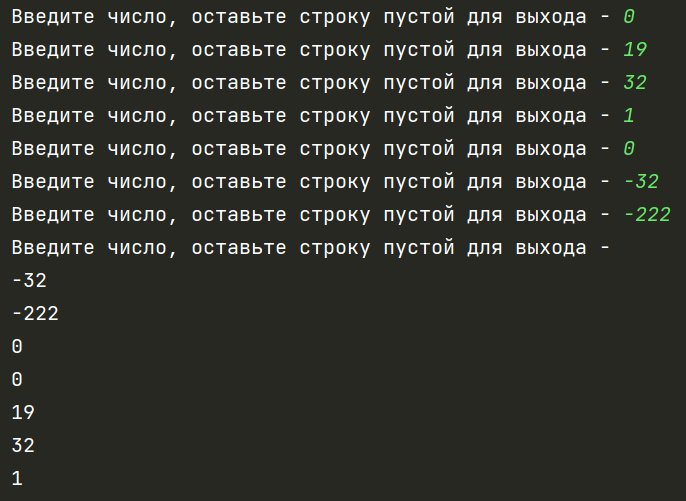


Рисунок – Результат работы

Программа №7

Напишите функцию с названием reverseLookup, которая будет осуществлять поиск всех ключей в словаре по заданному значению. Функция должна принимать в качестве параметров словарь и значение для поиска и возвращать список ключей (он может быть пустым) из этого словаря, соответствующих переданному значению. В основной программе продемонстрируйте работу функции путем создания словаря и поиска в нем всех ключей по заданному значению. Убедитесь, что функция работает корректно при наличии нескольких ключей для искомого значения, одного ключа и их отсутствии.

Необходимо реализовать пользовательский ввод ключа а также создание массива и переменных.

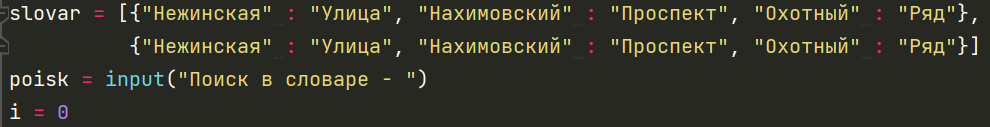


Рисунок – Пользовательский ввод, массив

После следует создать цикл, выводящий найденные ключи из списка.

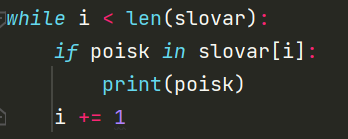


Рисунок – Цикл, выводящий ключи

Результат работы:

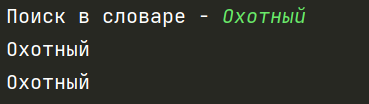


Рисунок – Результат работы

Программа №8

Напишите программу, определяющую и выводящую на экран количество уникальных символов во введенной пользователем строке. Например, в строке Hello, World! содержится десять уникальных символов, а в строке zzz – один. Используйте словарь или набор для решения этой задачи.

Следует организовать пользовательский ввод строки с последующим ее анализом.



Рисунок – Вывод уникальных символов

Результат работы:

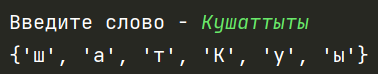


Рисунок - Результат работы

Вывод: научились работать со списками, словарями и кортежами, использовали функции для их форматирования, решая задачи.

ПРАКТИЧЕСКАЯ РАБОТА №4

Функции

Цель работы: научиться работать с функциями в Python, решая задачи и применяя их на практике.

Программа №1

Представьте, что сумма за пользование услугами такси складывается из базового тарифа в размере $4,00 плюс $0,25 за каждые 140 м поездки. Напишите функцию, принимающую в качестве единственного параметра расстояние поездки в километрах и возвращающую итоговую сумму оплаты такси. В основной программе должен демонстрироваться результат вызова функции.

Вначале следует запросить у пользователя расстояние поездки, вызвать функцию и вывести ее значение. Действия функции описаны в комментарии.

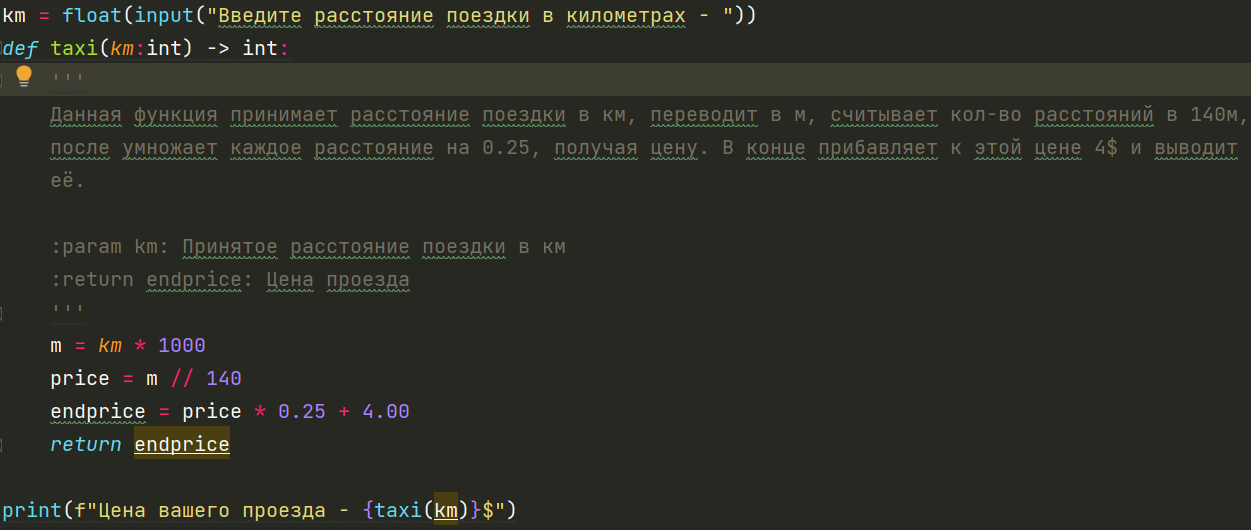


Рисунок – Пользовательский ввод, функция, вывод

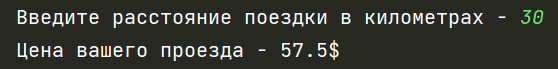


Рисунок – Результат работы

Программа №2

Интернет-магазин предоставляет услугу экспресс-доставки для части своих товаров по цене $10,95 за первый товар в заказе и $2,95 – за все последующие. Напишите функцию, принимающую в качестве единственного параметра количество товаров в заказе и возвращающую общую сумму доставки. В основной программе должны производиться запрос количества позиций в заказе у пользователя и отображаться на экране сумма доставки.

Вначале следует запросить у пользователя количество товаров, отправленных по доставке. После, с помощью функции мы рассчитываем стоимость доставки и выводим ее. Подробное описание функции написано в комментарии.

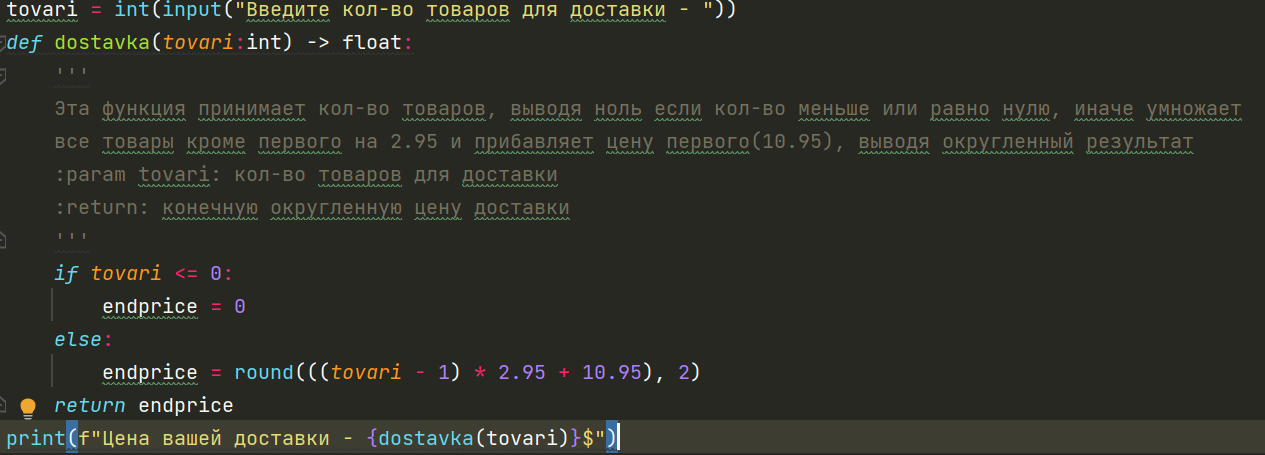


Рисунок – Пользовательский ввод, функция, вывод

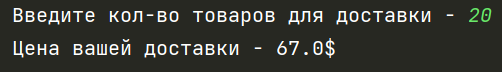


Рисунок – Результат работы

Программа №3

Простое число представляет собой число, большее единицы, которое без остатка делится лишь на само себя и единицу. Напишите функцию для определения того, является ли введенное число простым. Возвращаемое значение должно быть либо True, либо False. В основной программе, как и ожидается, пользователь должен ввести целое число и получить ответ о том, является ли оно простым. Убедитесь, что основная программа не будет запускаться, если файл импортирован в другой файл в качестве модуля.

Для начала следует создать функцию Main, пользовательский ввод, функцию, в которой будет проверяться простота, и вывод. Подробное описание функции находится в комментарии.

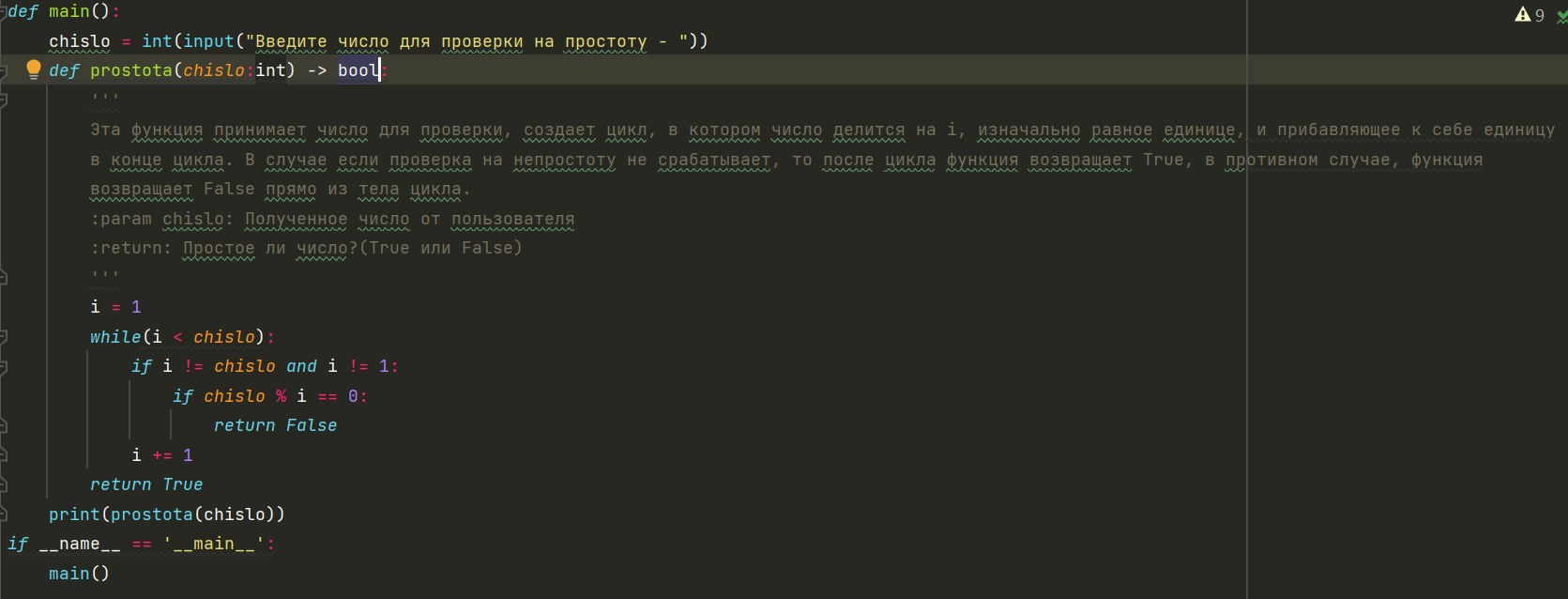


Рисунок – Пользовательский ввод, функция, вывод

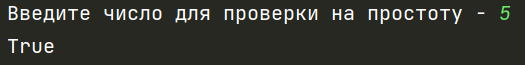


Рисунок – Результат работы

Программа №4

Напишите функцию, принимающую на вход два целочисленных параметра, представляющих числитель и знаменатель дроби. В теле функции должно выполняться максимально возможное сокращение дроби, а полученные в итоге числитель и знаменатель должны быть возвращены исходной программе. Например, если на вход функции передать числа 6 и 63, числитель и знаменатель итоговой дроби должны быть 2 и 21. В основной программе нужно запросить у пользователя числитель и знаменатель исходной дроби, передать их в функцию и вывести на экран результат.

Сначала следует создать пользовательский ввод, функцию, расчитывающую возможные сокращения, вывод. Подробные комментарии написаны в комментарии.

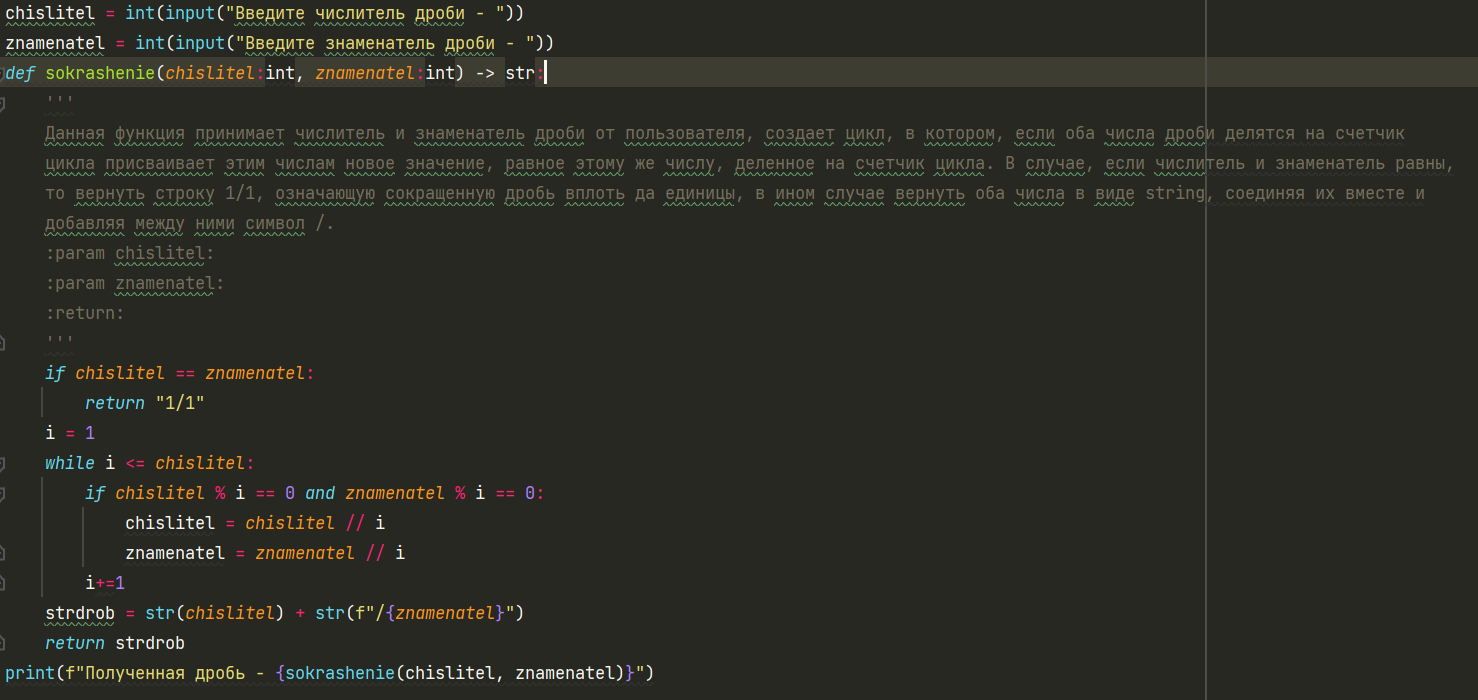


Рисунок – Пользовательский ввод, функция, вывод

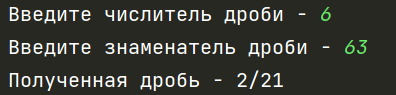


Рисунок – Результат работы

Программа №5

Представьте, что в вашем регионе устаревшим является формат номер- ных автомобильных знаков из трех букв, следом за которыми идут три цифры. Когда все номера такого шаблона закончились, было решено обновить формат, поставив в начало четыре цифры, а за ними три буквы. Напишите функцию, которая будет генерировать случайный номерной знак. При этом номера в старом и новом форматах должны создаваться примерно с одинаковой вероятностью. В основной программе нужно сгенерировать и вывести на экран случайный номерной знак.

Для начала следует добавить библиотеки рандома и стринга, функцию, создающую рандомный номер и вывод. Подробное описание функции лежит в комментарии.

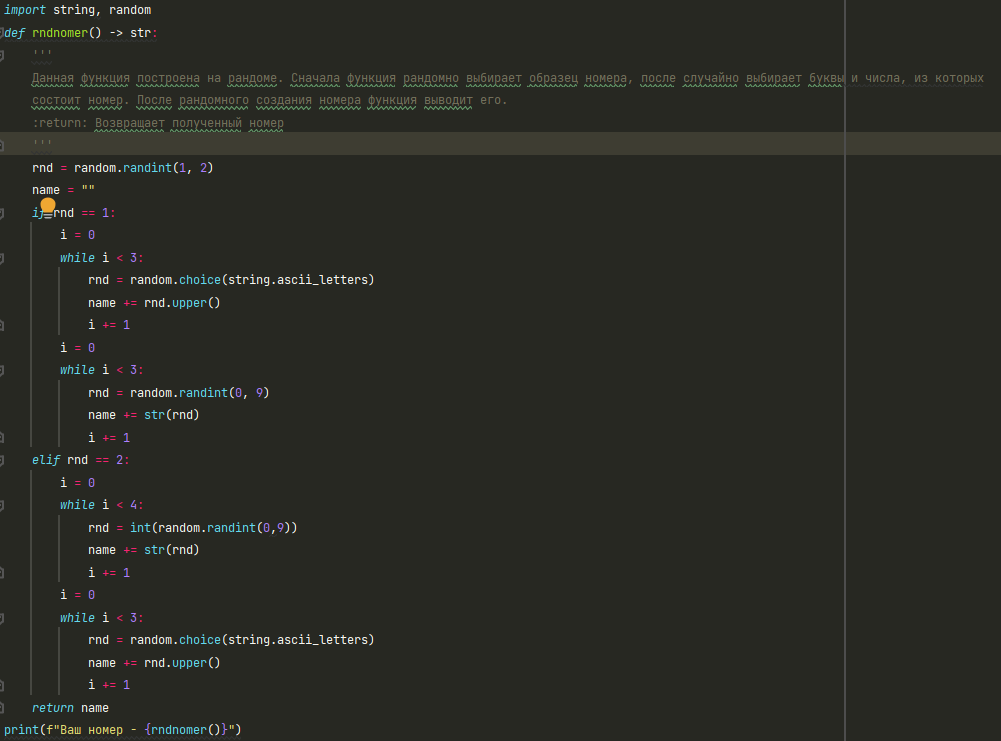


Рисунок – Библиотеки, функция, вывод



Рисунок – Результат работы

Вывод: научились работать с функциями в Python, решая задачи и применяя их на практике.

ПРАКТИЧЕСКАЯ РАБОТА №5

«Обработка ошибок и исключений»

Цель работы: исследовать возможные ошибки и исключения на языке программирования Python, рассмотреть их на практике, осознать, понять, запомнить.

Ошибки и исключения возникают в процессе написания программы, что может привести к нежелательному результату.

Синтаксические ошибки - они возникают при неправильном написании методов, функций, названий переменных, отсутствие необходимых операторов, т.е. все что связано с написание кода. Например, вместо написания for написать rof, или wile вместо while.

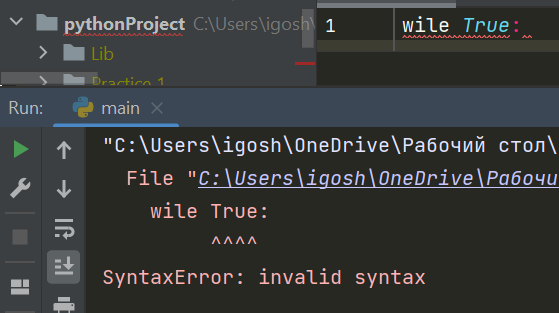


Рисунок – Синтаксическая ошибка

Логические ошибки – это ошибки в логике программы. Они возникают вследствие ошибок при проектировании и написании программы. Например, счетчик может увеличиваться не на 1, после прогонки цикла, а на 2. Синтаксической ошибки здесь нет, но при этом это ломает логику программы.

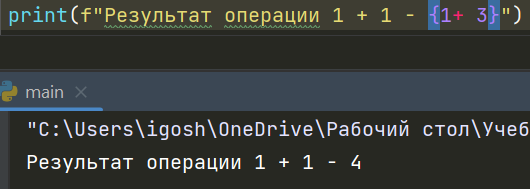


Рисунок – Логическая ошибка

Кроме того, существуют ошибки во время выполнения программы – по причине отсутствия прав администратора, или занятой другим процессом ОЗУ.

Некоторые операции могут быть невозможными, из-за чего наша программа рискует зависнуть, или вовсе каким-либо образом навредить нашему ПК.

Для того, чтобы узнать список всех существующих исключений, следует ввести print(dir(locals()["\_\_builtins\_\_"])) в компилятор Python, что выведет список всех исключений.

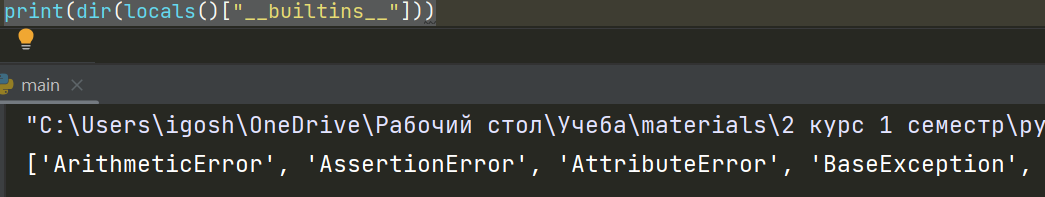


Рисунок – Демонстрация вывода списка

Ниже будут находится наименования и краткие описания всех исключений в Python:

BaseException - базовое исключение, от которого берут начало все остальные.

SystemExit - исключение, порождаемое функцией sys.exit при выходе из программы.

KeyboardInterrupt - порождается при прерывании программы пользователем (обычно сочетанием клавиш Ctrl+C).

GeneratorExit - порождается при вызове метода close объекта generator.

Exception - тут уже заканчиваются полностью системные исключения (которые лучше не трогать) и начинаются обыкновенные, с которыми можно работать.

StopIteration - порождается встроенной функцией next, если в итераторе больше нет элементов.

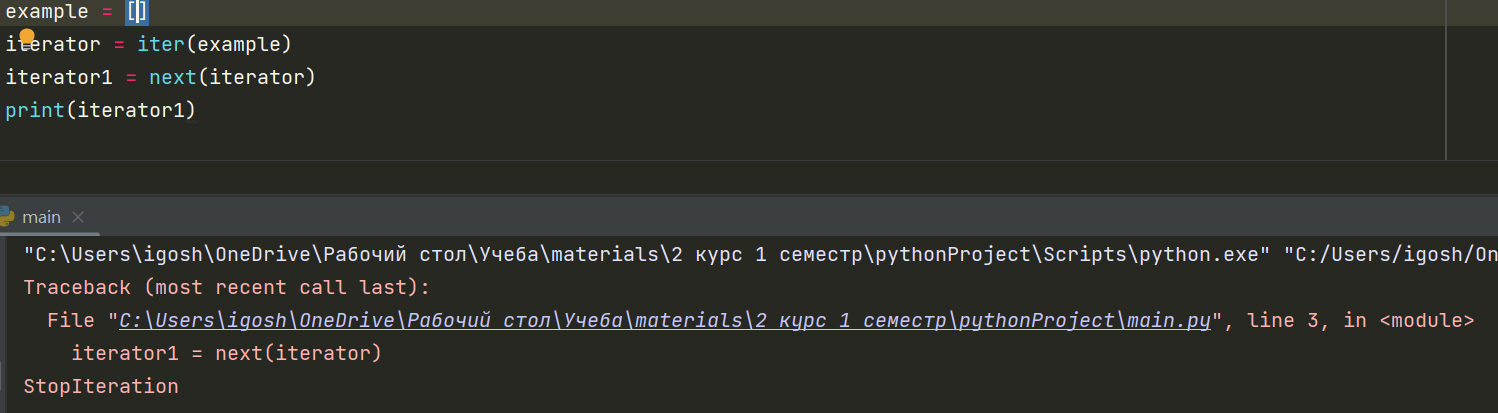


Рисунок 72 - StopIteration

ArithmeticError - арифметическая ошибка.

* FloatingPointError - порождается при неудачном выполнении операции с плавающей запятой. На практике встречается нечасто.
* OverflowError - возникает, когда результат арифметической операции слишком велик для представления. Не появляется при обычной работе с целыми числами (так как python поддерживает длинные числа), но может возникать в некоторых других случаях.
* ZeroDivisionError - деление на ноль.

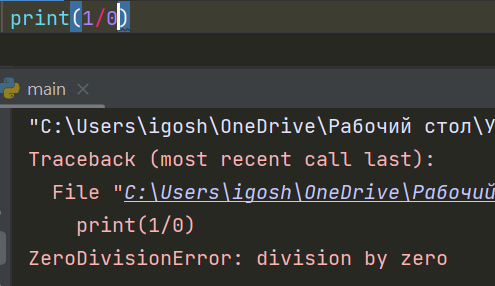


Рисунок 73 - ZeroDivisionError

AssertionError - выражение в функции assert ложно.

AttributeError - объект не имеет данного атрибута (значения или метода).

BufferError - операция, связанная с буфером, не может быть выполнена.

EOFError - функция наткнулась на конец файла и не смогла прочитать то, что хотела.

ImportError - не удалось импортирование модуля или его атрибута.

LookupError - некорректный индекс или ключ.

* IndexError - индекс не входит в диапазон элементов.

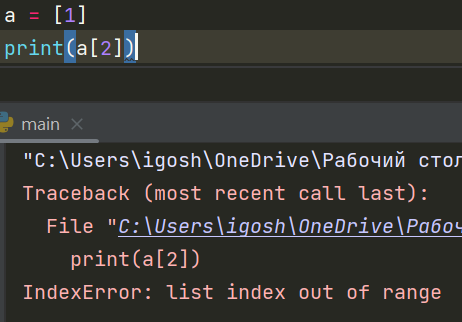


Рисунок - IndexError

* KeyError - несуществующий ключ (в словаре, множестве или другом объекте).

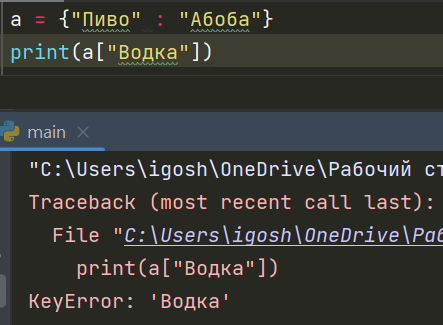


Рисунок - KeyError

MemoryError - недостаточно памяти.

NameError - не найдено переменной с таким именем.

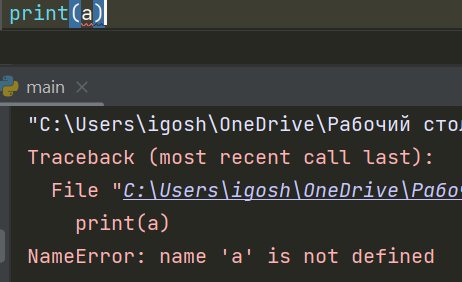


Рисунок - NameError

* UnboundLocalError - сделана ссылка на локальную переменную в функции, но переменная не определена ранее.

OSError - ошибка, связанная с системой.

* BlockingIOError
* ChildProcessError - неудача при операции с дочерним процессом.
* ConnectionError - базовый класс для исключений, связанных с подключениями.
* BrokenPipeError
* ConnectionAbortedError
* ConnectionRefusedError
* ConnectionResetError
* FileExistsError - попытка создания файла или директории, которая уже существует.
* FileNotFoundError - файл или директория не существует.
* InterruptedError - системный вызов прерван входящим сигналом.
* IsADirectoryError - ожидался файл, но это директория.
* NotADirectoryError - ожидалась директория, но это файл.
* PermissionError - не хватает прав доступа.
* ProcessLookupError - указанного процесса не существует.
* TimeoutError - закончилось время ожидания.

ReferenceError - попытка доступа к атрибуту со слабой ссылкой.

RuntimeError - возникает, когда исключение не попадает ни под одну из других категорий.

NotImplementedError - возникает, когда абстрактные методы класса требуют переопределения в дочерних классах.

SyntaxError - синтаксическая ошибка.

* IndentationError - неправильные отступы.
* TabError - смешивание в отступах табуляции и пробелов.

SystemError - внутренняя ошибка.

TypeError - операция применена к объекту несоответствующего типа.



Рисунок - TypeError

ValueError - функция получает аргумент правильного типа, но некорректного значения.



Рисунок - ValueError

UnicodeError - ошибка, связанная с кодированием / раскодированием unicode в строках.

* UnicodeEncodeError - исключение, связанное с кодированием unicode.
* UnicodeDecodeError - исключение, связанное с декодированием unicode.
* UnicodeTranslateError - исключение, связанное с переводом unicode.

Warning - предупреждение.

Вывод: исследовали возможные ошибки и исключения на языке программирования Python, рассмотрели их на практике, осознали, поняли, запомнили.

ПРАКТИЧЕСКАЯ РАБОТА №6

«Работа с файлами»

Цель работы: научиться работать с файлами в Python, использовать библиотеку tabulate, решить задачи.

Программа №1(23)

В автохозяйстве имеется N автомашин (N –заданное число). Для каждого автомобиля заданы три характеристики: номер, марка машины. Тип неисправности (или ее отсутствие). Необходимо составить сводные (по типам неисправностей) заявки на ремонт машин с указанием их номеров и марок. Отсортируйте список по полю «марка» и по полю «тип неисправности».

Для начала следует создать функцию main, которая будет выполнять административные функции и из которой будут вызываться другие функции. Кроме того, эта функция реализует консольное меню и выход из программы.

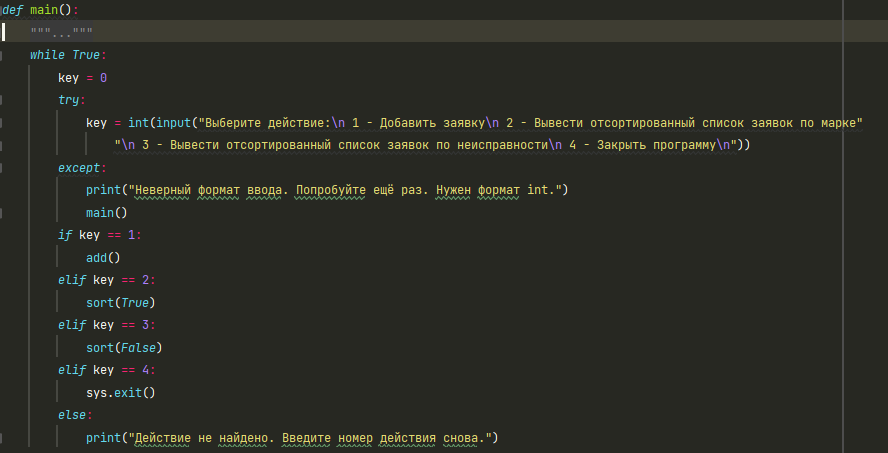


Рисунок – Main

А также необходимо создать точку, из которой будет запускаться эта функция.

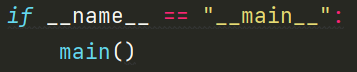


Рисунок – Вызов Main

После этого следует создать функцию заполнения файла пользователем. В данной программе это реализуют две функции: add, получающая количество машин у пользователя, и adding, непосредственно заполняющая файл и принимающая от add количество машин.

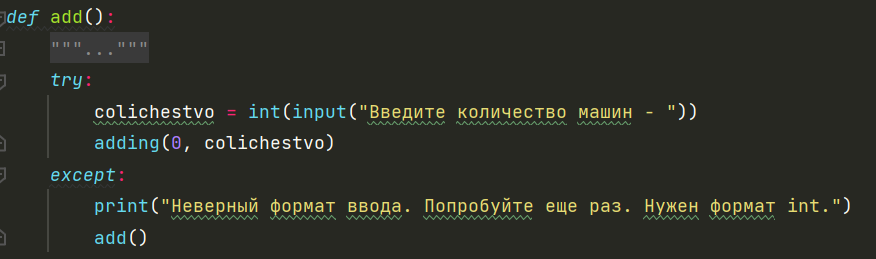


Рисунок - Add

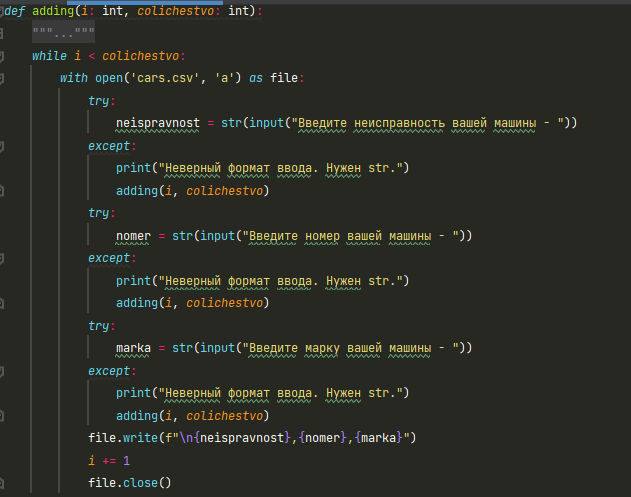


Рисунок – Adding

После заполнения файла необходимо создать функцию с сортировкой данных из файла по полю марка или по неисправности машины.

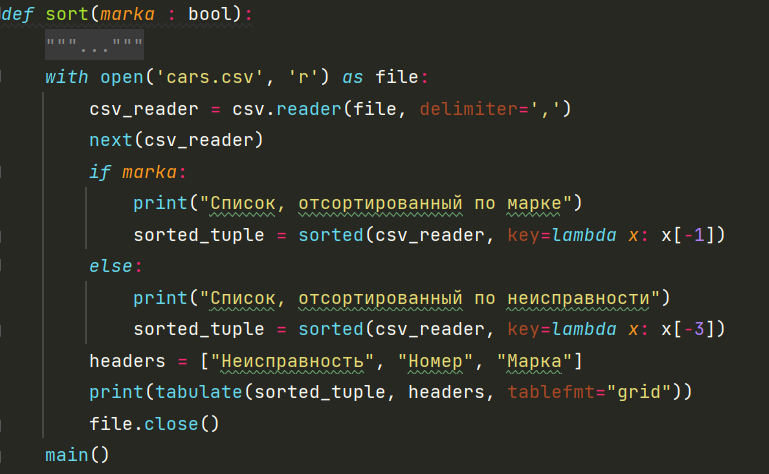


Рисунок – Sort

Результат работы:

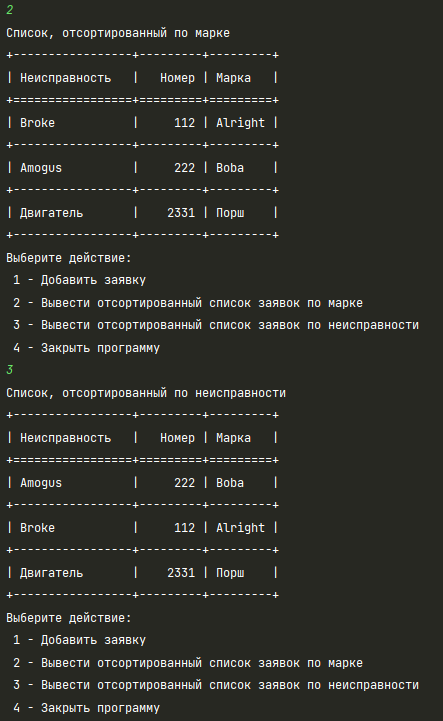


Рисунок – Результат работы

Программа №2(24)

Программа содержит список пассажиров авиарейса. Список имеет следующую структуру: фамилия - номер удостоверения личности - место - вес багажа. Программа должна заполнить список, рассчитать средний вес багажа всех пассажиров, а затем вывести три списка - пассажиров, не имеющих багажа, пассажиров, у которых вес багажа, ниже среднего, и пассажиров, у которых вес багажа выше среднего. Отсортируйте список по полю «вес багажа».

Для начала следует создать функцию main, которая будет выполнять административные функции и из которой будут вызываться другие функции. Кроме того, эта функция реализует консольное меню и выход из программы.

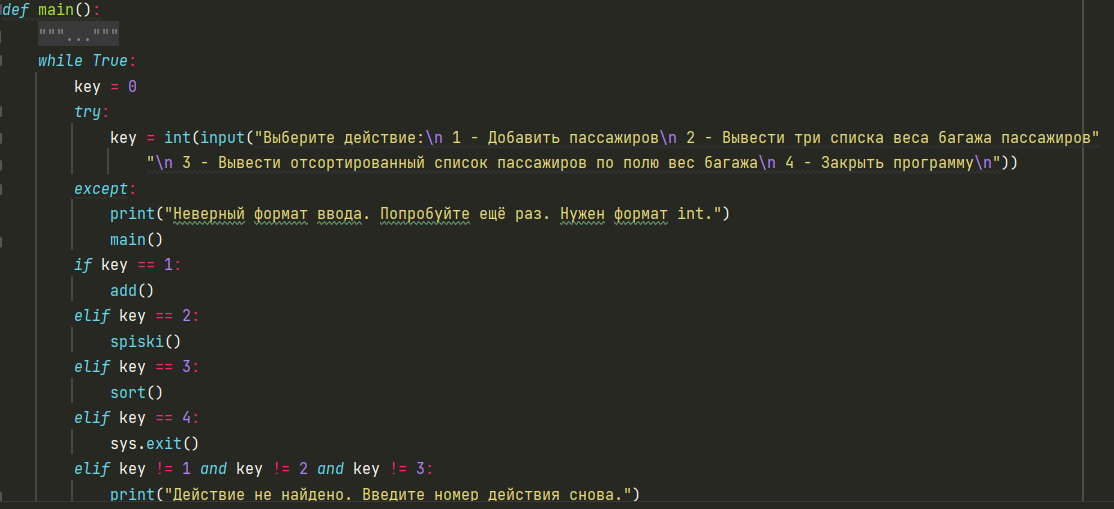


Рисунок – Main

А также необходимо создать точку, из которой будет запускаться эта функция.



Рисунок – Вызов Main

После этого следует создать функцию заполнения файла пользователем. В данной программе это реализуют две функции: add, получающая количество пассажиров у пользователя, и adding, непосредственно заполняющая файл и принимающая от add количество пассажиров.



Рисунок – Add

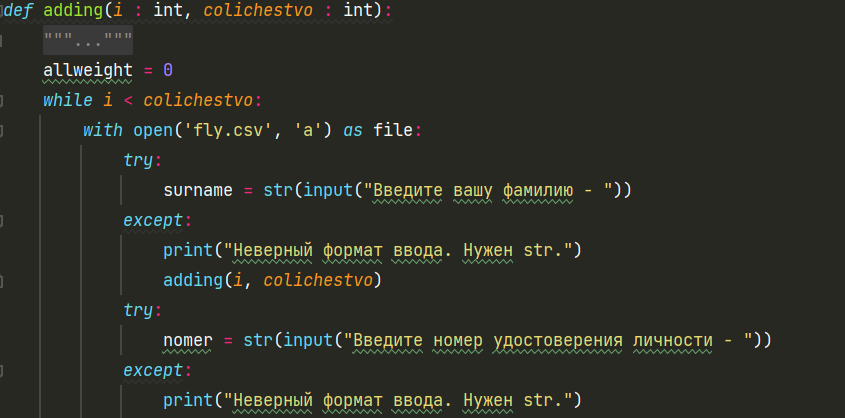


Рисунок – Adding

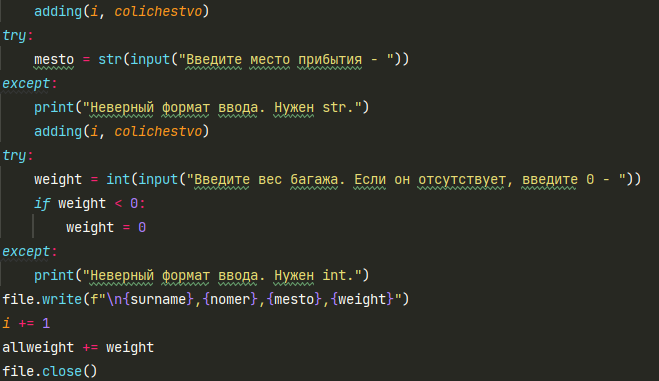


Рисунок – Adding

После этого, необходимо создать функцию, которая будет сортировать заполненный пользователем файл по полю вес багажа и выводить его на экран в виде таблицы.

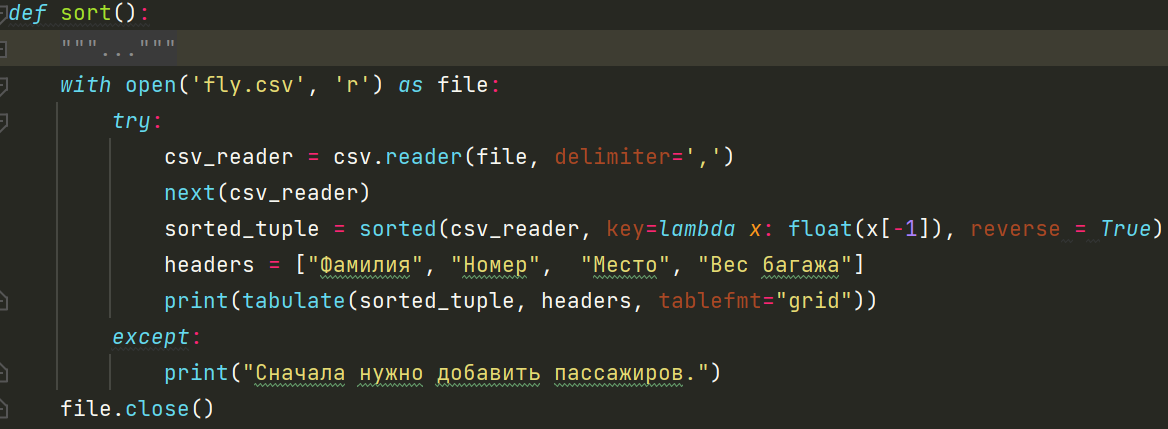


Рисунок – Sort

После этого создаем функцию srarifmet, которая будет вычислять среднее арифметическое значение веса багажа пассажиров.

Далее требуется создать функцию spiski, которая будет выводить три списка пассажиров, которые: без багажа, с багажом тяжелее среднестатистического, с багажом ниже среднестатистического.



Рисунок – Spiski

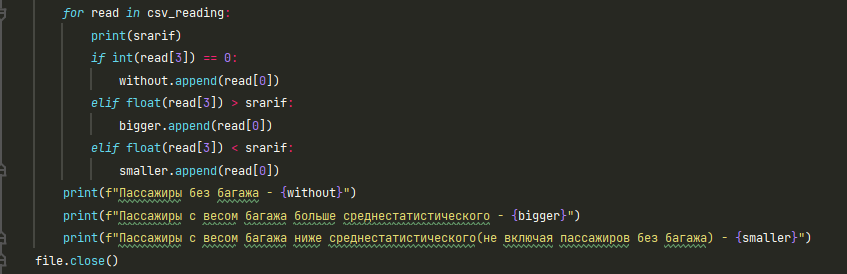


Рисунок – Spiski

Результат работы:

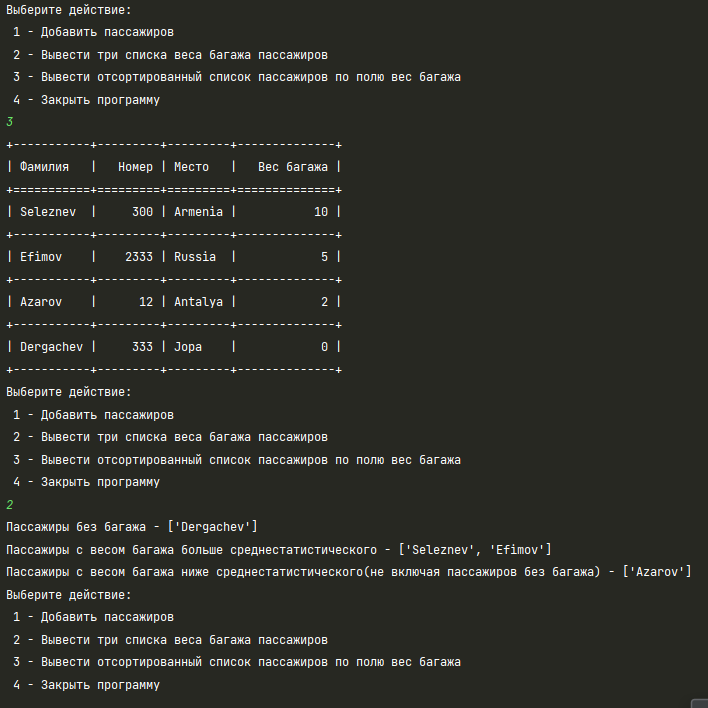


Рисунок – Результат работы

Вывод: научились работать с файлами в Python, использовали библиотеку tabulate, решили задачи.

# ПРАКТИЧЕСКАЯ РАБОТА №7

Итоговый проект

Цель работы: разработать консольный проект по работе с товарами, закрепить знания, полученные по ходу семестра, научиться создавать объемные проекты.

Условия:

Все данные должны хранится в файлах. Всего должно быть 2 файла: 1 – информация о пользователях (логин, пароль, роль), информация о товарах (наименование, описание, поставщик, цена, доступное количество). При запуске программы данные из файлов должны считываться и помещаться в список/словарь. При запуске программы должно выводиться консольное меню:

1 – авторизироваться: При авторизации пользователь вводит логин и пароль. Эти данные проверяются со значением в списке/словаре. Если у пользователя роль обычного пользователя то он попадает в личный кабинет пользователя(Вывод всех товаров в библиотеку tabulate). Если у пользователя роль администратора то он попадает в личный кабинет администратора(вывод всех товаров в библиотеку tabulate, редактирование товара, добавление товара, удаление товара)

2 – зарегистрироваться: Пользователь должен ввести логин, пароль, повторный ввод пароля. Если первый ввод пароля и второй ввод пароля не совпадают необходимо выводить соответствующее сообщение и вызывать функцию авторизации. Если логин уже занят необходимо выводить соответствующее сообщение и вызывать функцию авторизации. По умолчанию все пользователи регистрируются как обычные пользователи. В файле изначально должен быть 1 администратор. В данном приложение необходимо использовать функции, документирующие комментарии к функциям, type hints, файлы. По возможности необходимо использовать классы и объекты классов (преобразование товаров и пользователей в объекты классов).

Результаты работы программы:

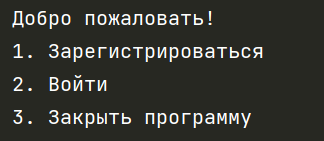


Рисунок – Консольное меню

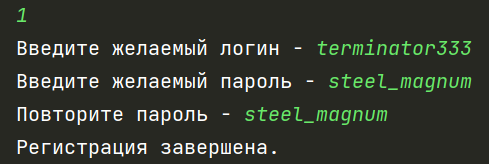


Рисунок - Регистрация

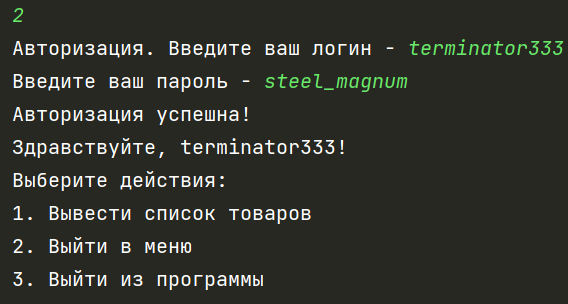


Рисунок - Авторизация

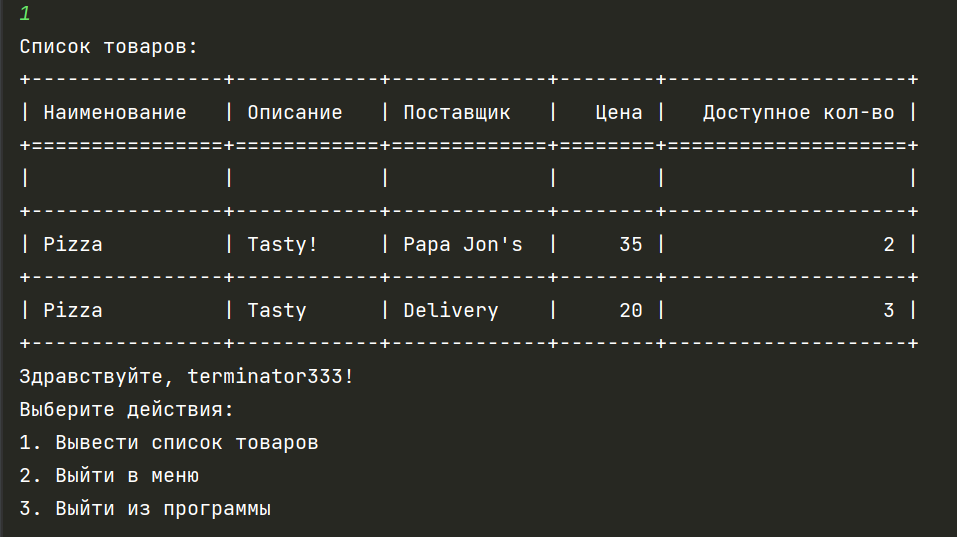


Рисунок – Авторизация с ролью user, меню и вывод списка товаров

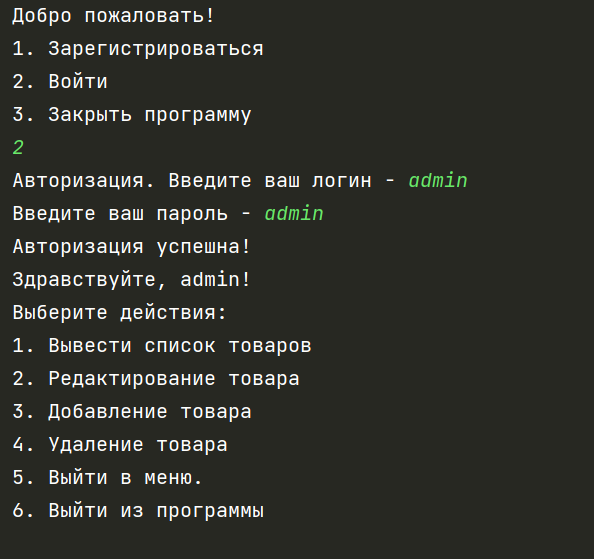


Рисунок – Авторизация с ролью admin, консольное меню

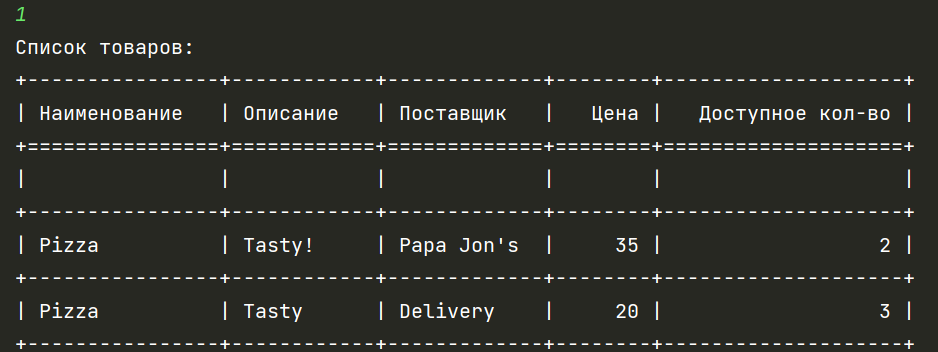


Рисунок – Вывод списка товаров



Рисунок – Редактирование товара

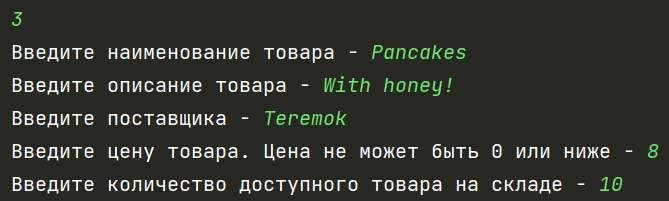


Рисунок – Добавление товара

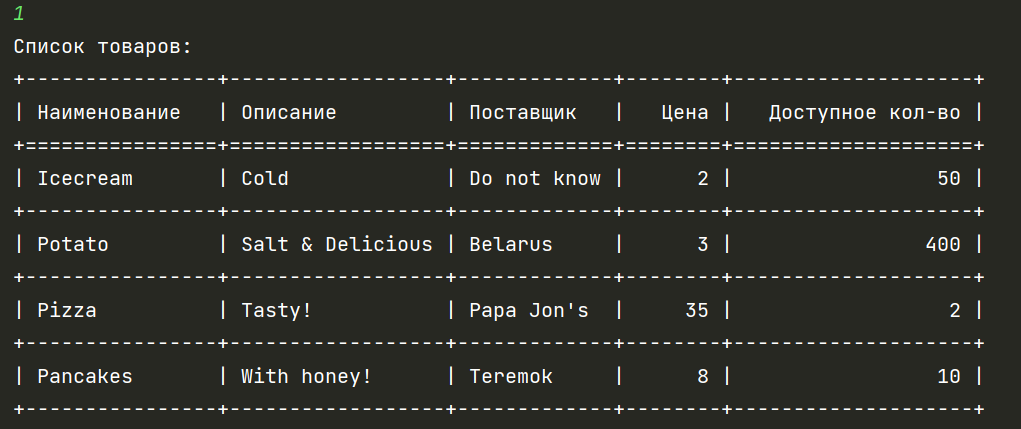


Рисунок – Демонстрация добавки

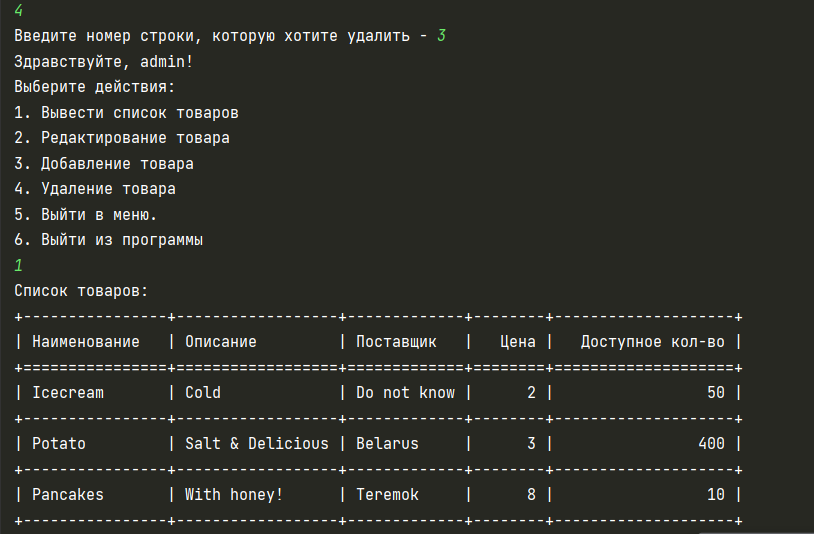


Рисунок – Удаление товара

Вывод: разработали консольный проект по работе с товарами, закрепили знания, полученные по ходу семестра, научились создавать объемные проекты.

Далее идут практические работы второго семестра:

ПРАКТИЧЕСКАЯ РАБОТА №1

Тема: «Работа с хэш-функциями»

Цель работы: научиться работать с хэш функциями, рассмотреть их возможности, особенности, разработать программу, использующую хэш-функции для хэширования паролей, используя соль.

Для подробного описания функции можно посмотреть на комментарий ниже её названия.

Для начала, создадим функцию main с консольным меню.

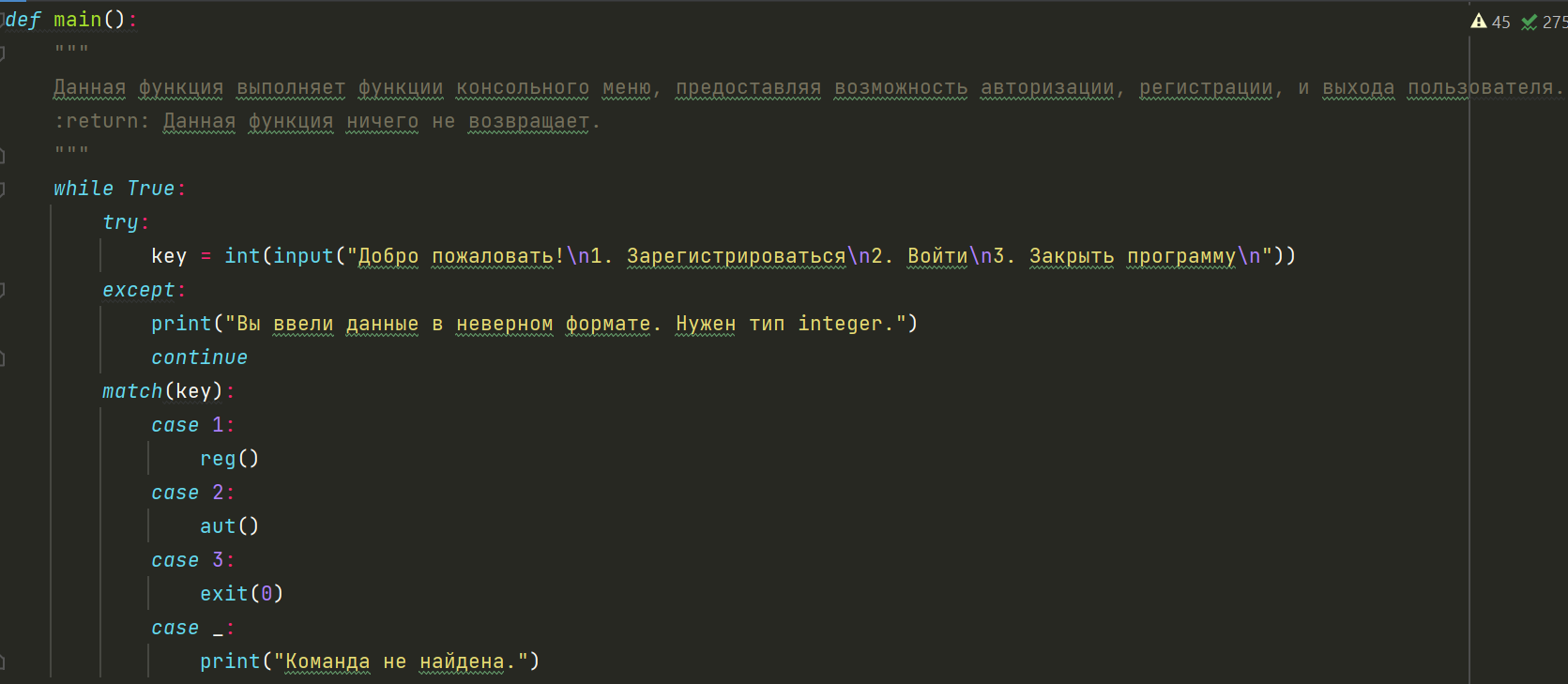


Рисунок – Функция main

Теперь, создадим её вызов.

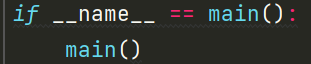


Рисунок – Вызов функции main

На очереди – функция регистрации пользователей.



Рисунок – Функция reg

Здесь – функция авторизации пользователей.

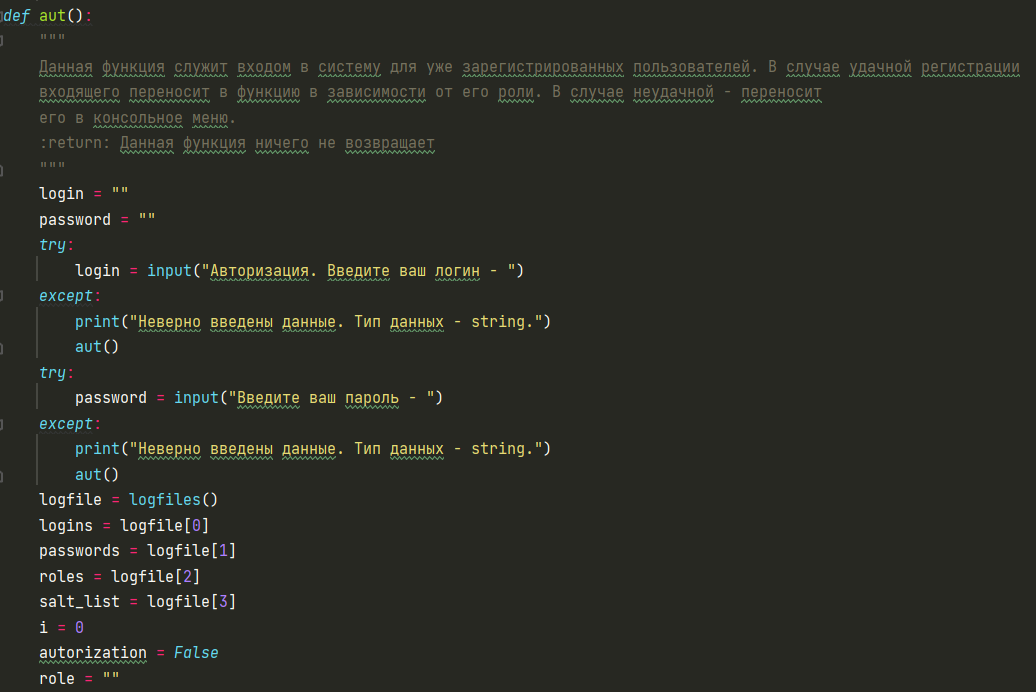


Рисунок – Функция aut

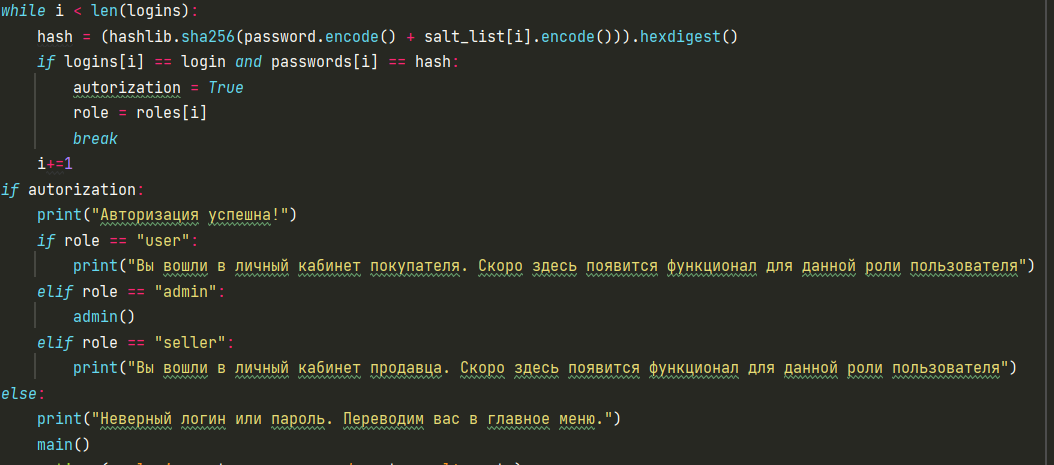


Рисунок – Функция aut

Теперь, функция продолжения регистрации и записи в файл.

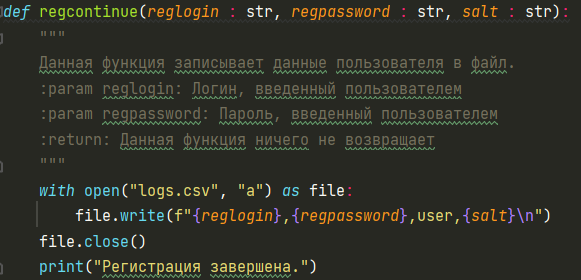


Рисунок – Функция regcontinue

На очереди – очень полезная функция, считывающая файл и разбивающая его на массивы.

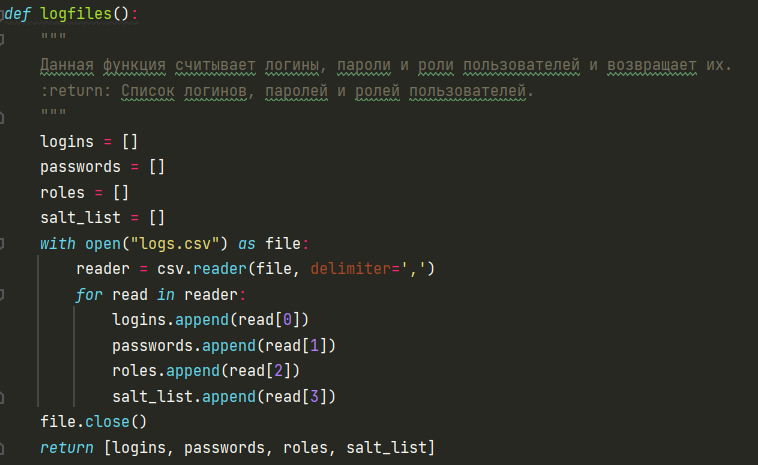


Рисунок – Функция logfiles

Здесь – функция личного кабинета администратора.

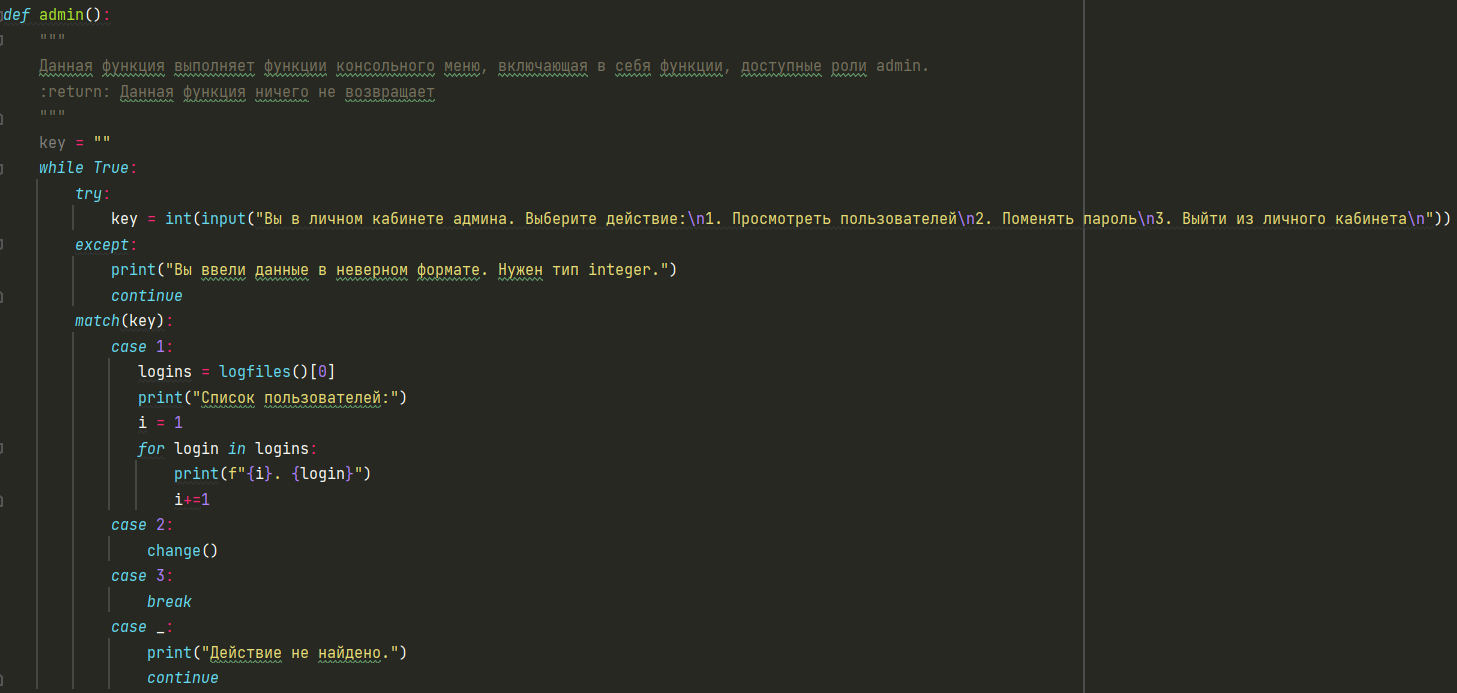


Рисунок – Функция admin

Здесь – функция администратора с возможностью смены пароля пользователям.



Рисунок – Функция change

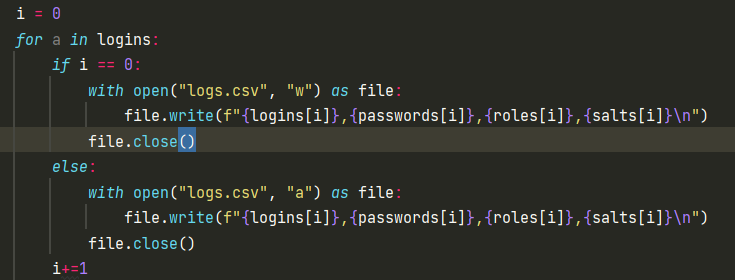


Рисунок – Функция change

Теперь, приступим к результатам работы:

Для начала, зарегистрируемся и попробуем войти.

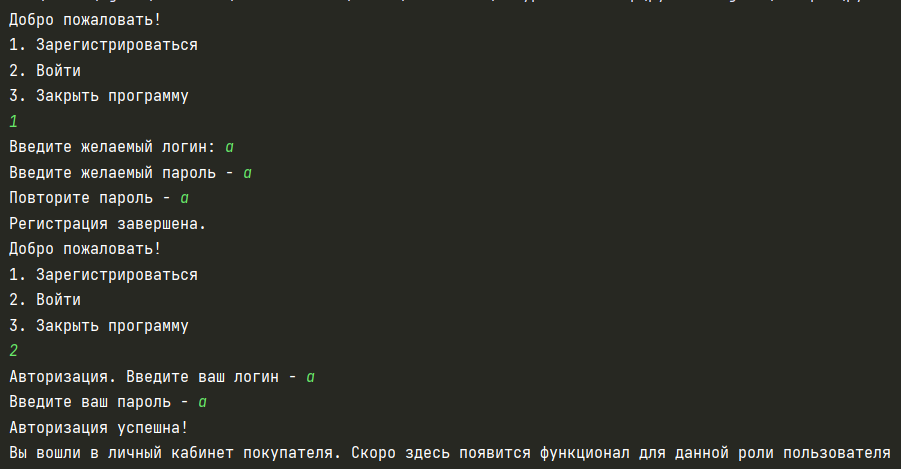


Рисунок – Результат работы

Теперь, войдем в аккаунт с ролью продавца.

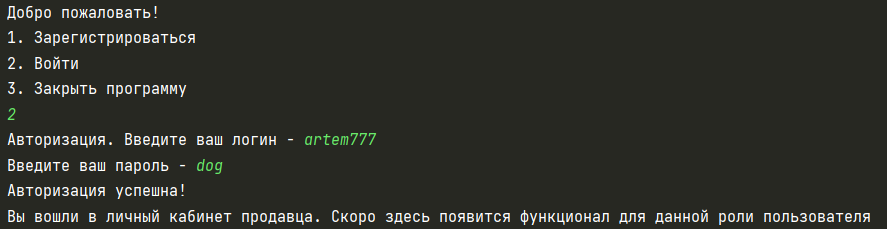


Рисунок – Результат работы

Теперь, зайдем в профиль роли администратора.

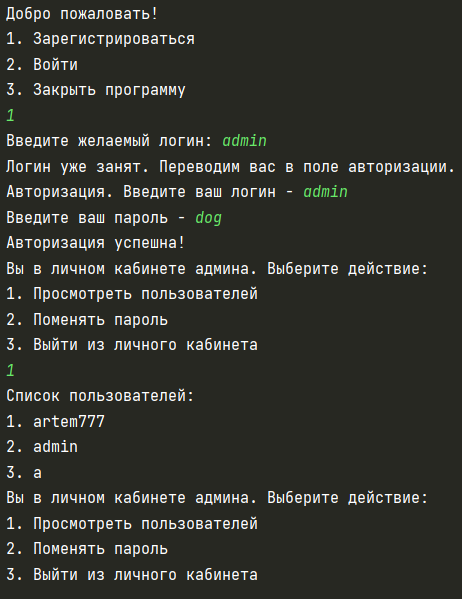


Рисунок – Результат работы

Попробуем поменять пароль нашему продавцу.

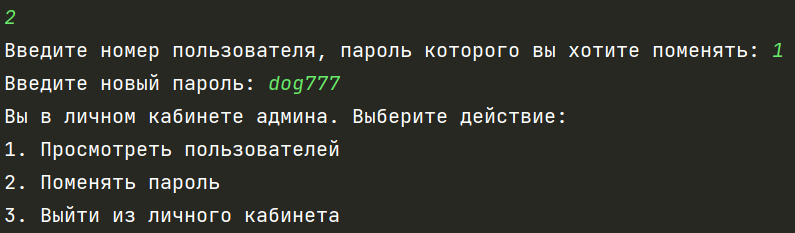


Рисунок – Результат работы

Как видим старый пароль dog не подходит, зато новый dog777 отлично работает.

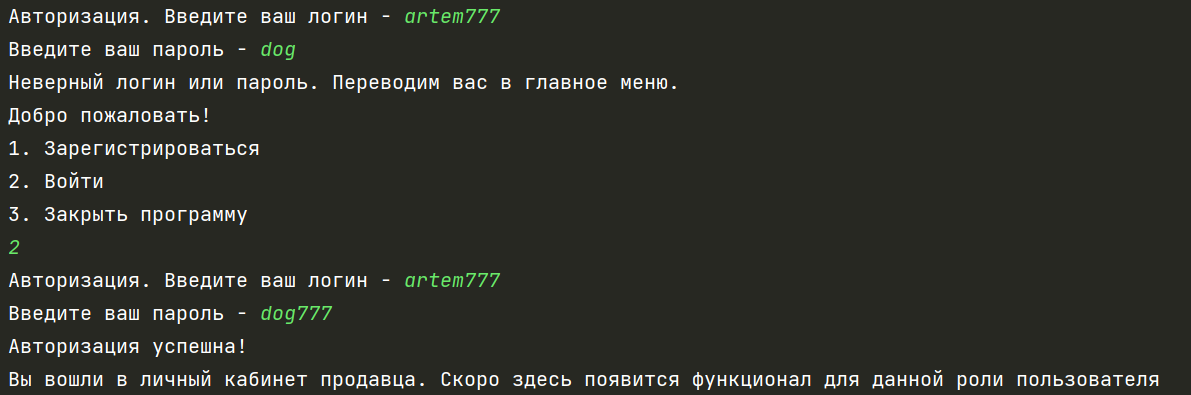


Рисунок – Результат работы

Вывод: научились работать с хэш функциями, рассмотрели их возможности, особенности, разработали программу, использующую хэш-функции для хэширования паролей, используя соль.

ПРАКТИЧЕСКАЯ РАБОТА №2

Тема: «Разработка смарт-контракта агентства недвижимости»

Цель работы: научиться работать со смарт-контрактами, создав контракт агентства недвижимости с некоторыми функциями покупки и продажи недвижимости.

Для начала, создадим структуру для недвижимости с необходимыми для этого полями.

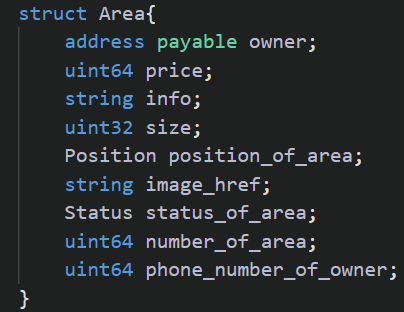


Рисунок – Структура

Теперь, реализуем функцию добавления объекта структуры.

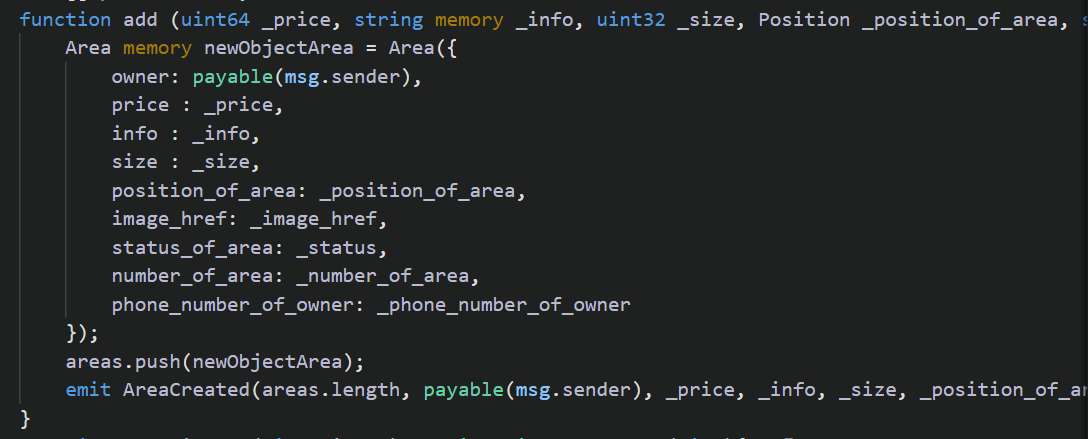


Рисунок – Функция добавления недвижимости

Теперь же, для удобства покупки, создадим функцию, возвращающую цену по индексу.

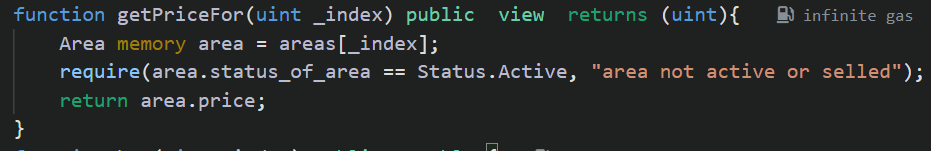


Рисунок – Функция вывода цены недвижимости

Ну и сама функция покупки недвижимости:

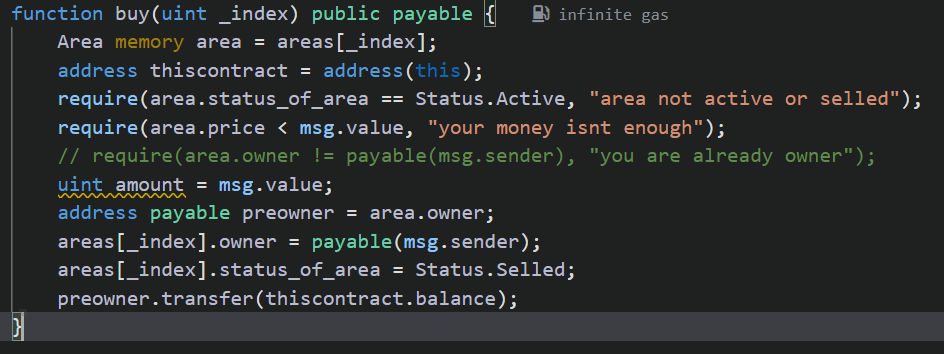


Рисунок – Функция покупки недвижимости

Также, есть снятие денег с контракта владельцем.

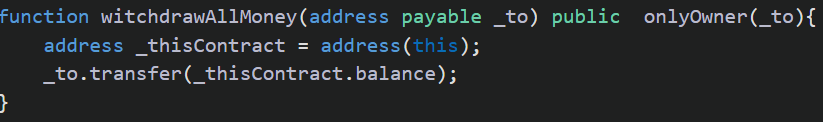


Рисунок – Функция снятия денег владельцем контракта

Также, в программе использовались и некоторые другие модификаторы, переменные, которые видно в коде ниже:

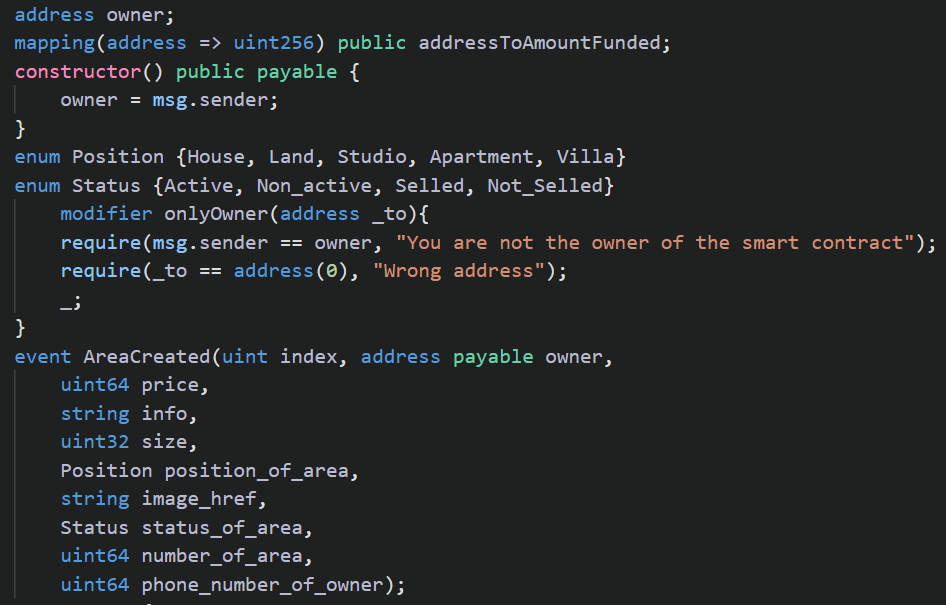


Рисунок – Прочие модификаторы и события контракта

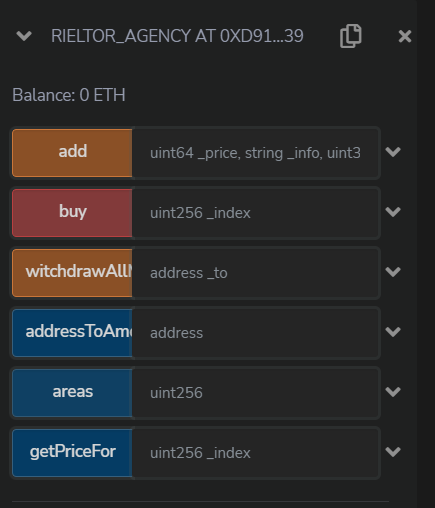


Рисунок – Результат работы

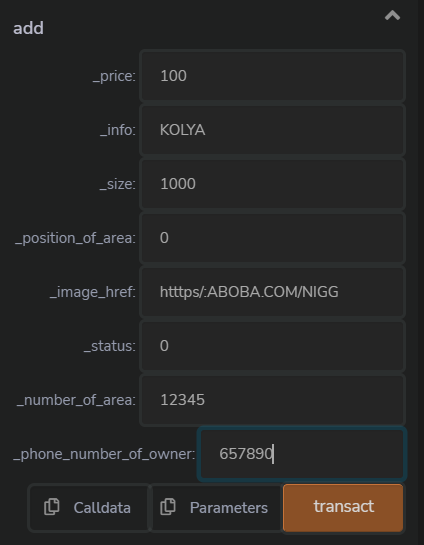


Рисунок - Добавление

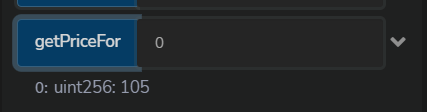


Рисунок – Запрос цены



Рисунок – Запрос информации

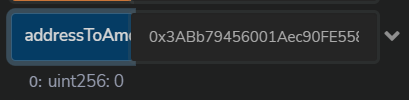


Рисунок – Запрос кол-ва покупок

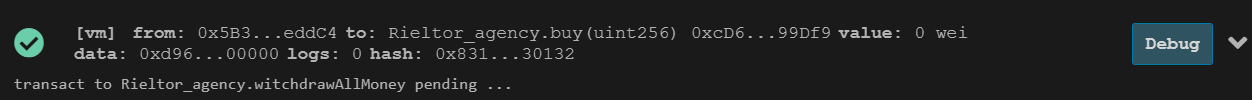


Рисунок – Покупка

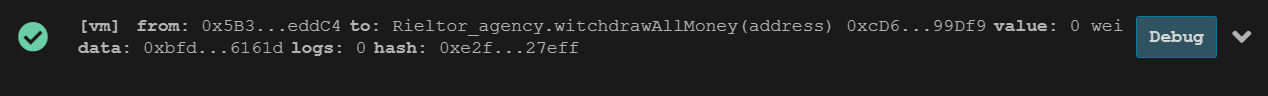


Рисунок – Снятие денег

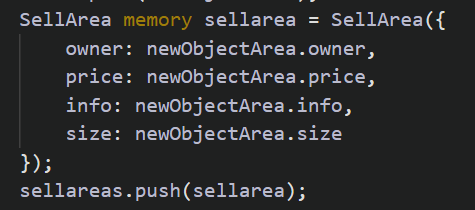


Рисунок – Создание объекта

Вывод: научились работать со смарт-контрактами, создав контракт агентства недвижимости с некоторыми функциями покупки и продажи недвижимости.

ПРАКТИЧЕСКАЯ РАБОТА №3

Тема: «Разработка смарт-контракта для финального проекта»

Цель работы: разработать смарт-контракт для финального проекта, закрепив полученные знания на языке смарт-контрактов Solidity, полученные в течение выполнения предыдущих практических работ.

Начнём с разработки контракта, в котором будут содержаться различные константы и структуры, необходимые нам в разработке ролевой модели и отзывов с ответами.

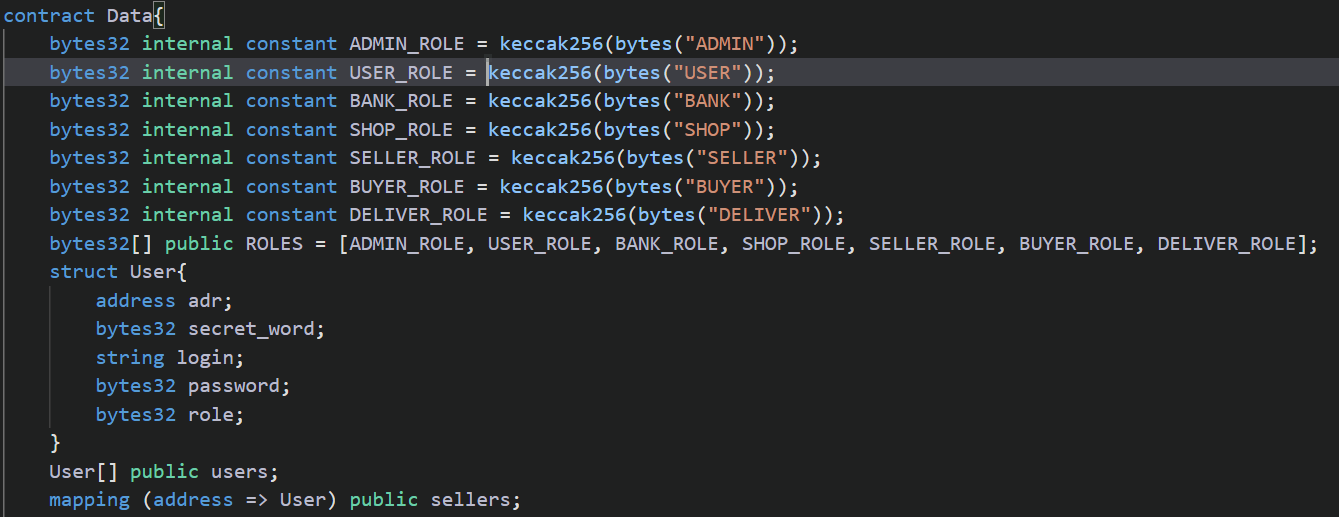


Рисунок – Контракт Data

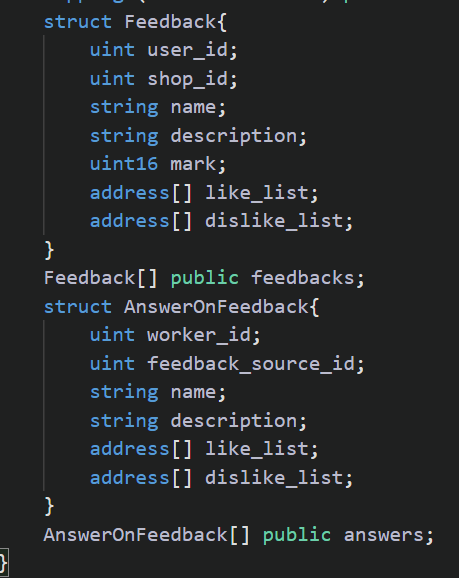


Рисунок – Контракт Data

Теперь, создадим контракт логики, где будут содержаться все наши основные функции, который в свою очередь будет наследовать контракт Data.



Рисунок – Контракт Logic наследующий Data

Реализуем функцию регистрации пользователя:



Рисунок – Функция регистрации



Рисунок – Функция регистрации

Теперь, функцию смены роли:



Рисунок – Функция смены роли

Можно теперь перейти и к отзывам. Создадим функцию для их добавления.

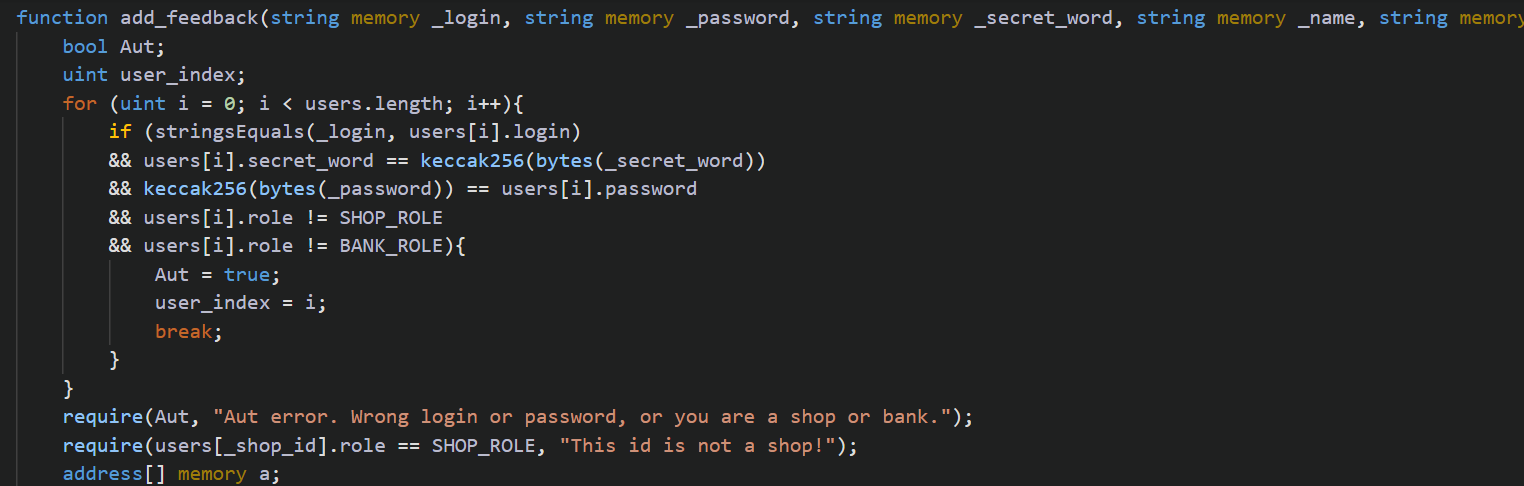


Рисунок – Функция добавления отзыва

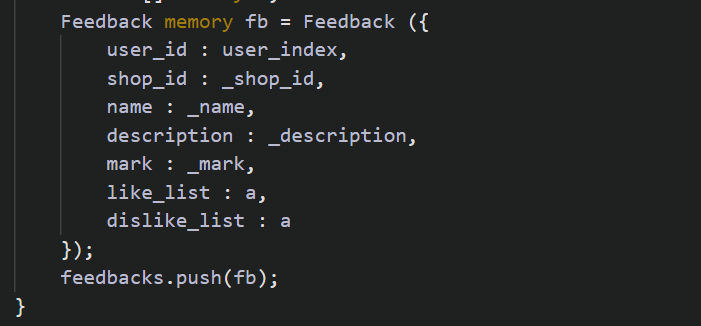


Рисунок – Функция добавления отзыва

Добавим теперь функцию добавления ответа на отзыв от продавца магазина.

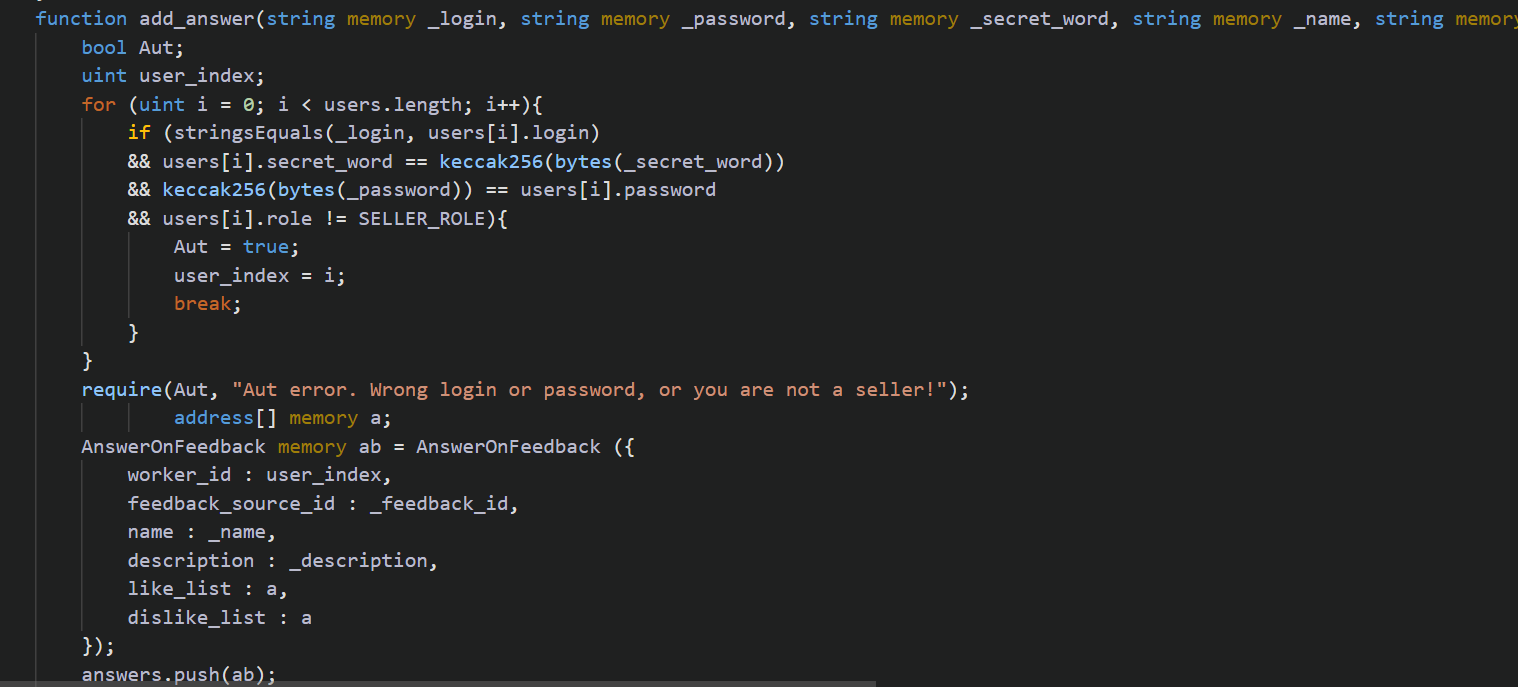


Рисунок – Функция добавления ответа на отзыв

Теперь добавим функции по работе с лайками и дизлайками для наших отзывов и ответов на них.

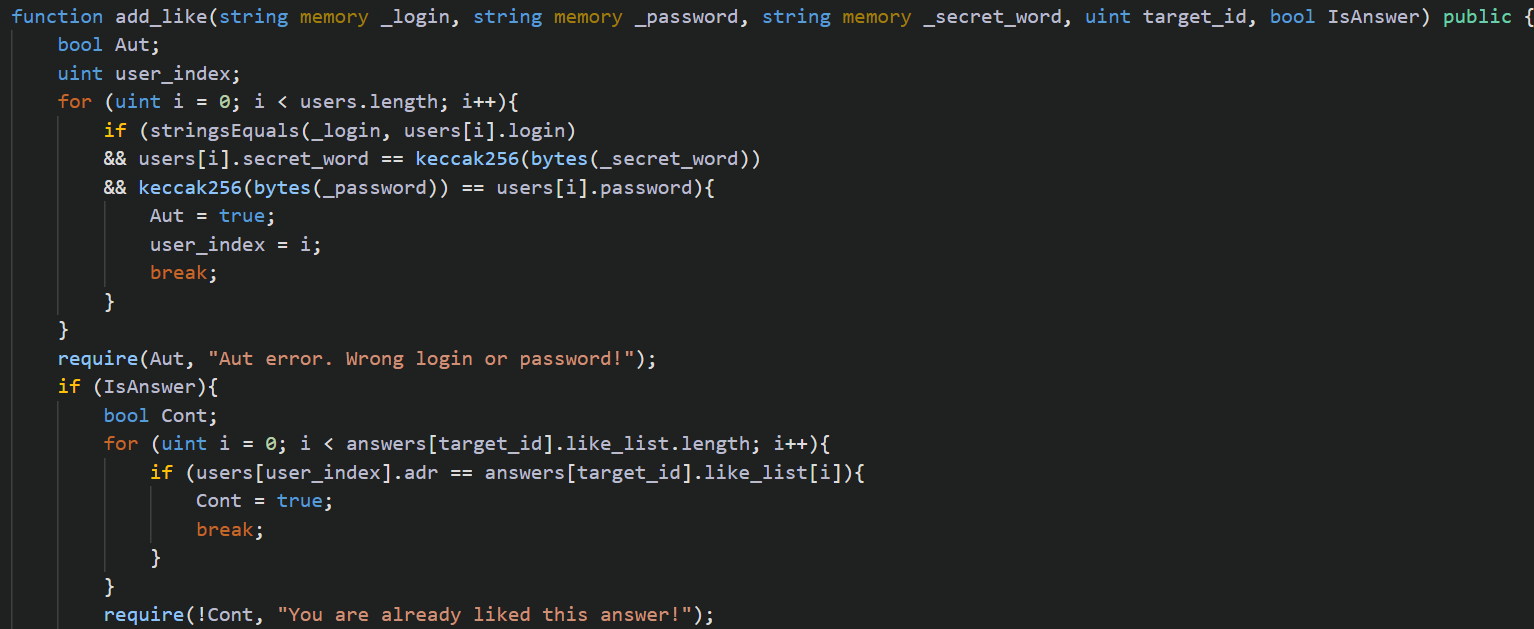


Рисунок – Функция добавления лайка

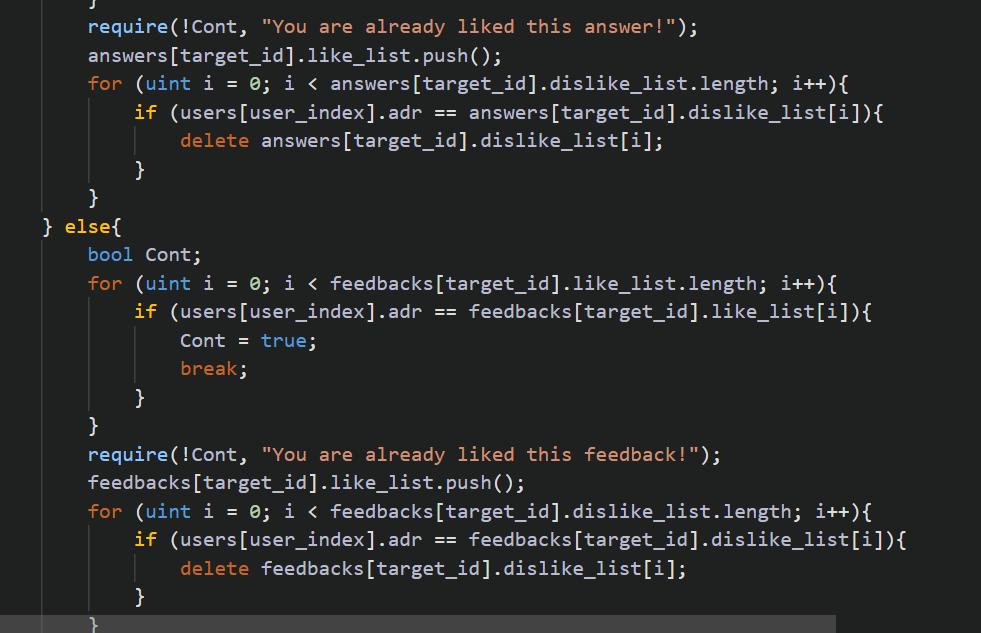


Рисунок – Функция добавления лайка

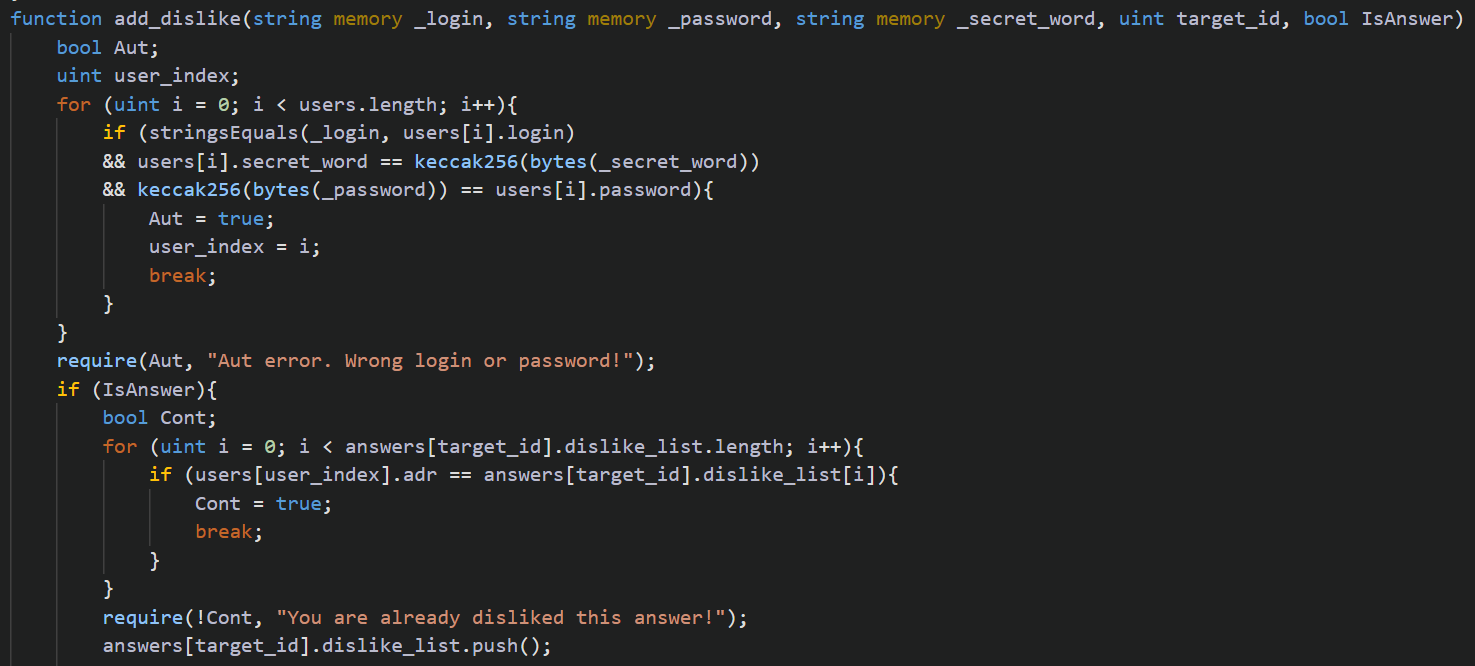


Рисунок – Функция добавления дизлайка

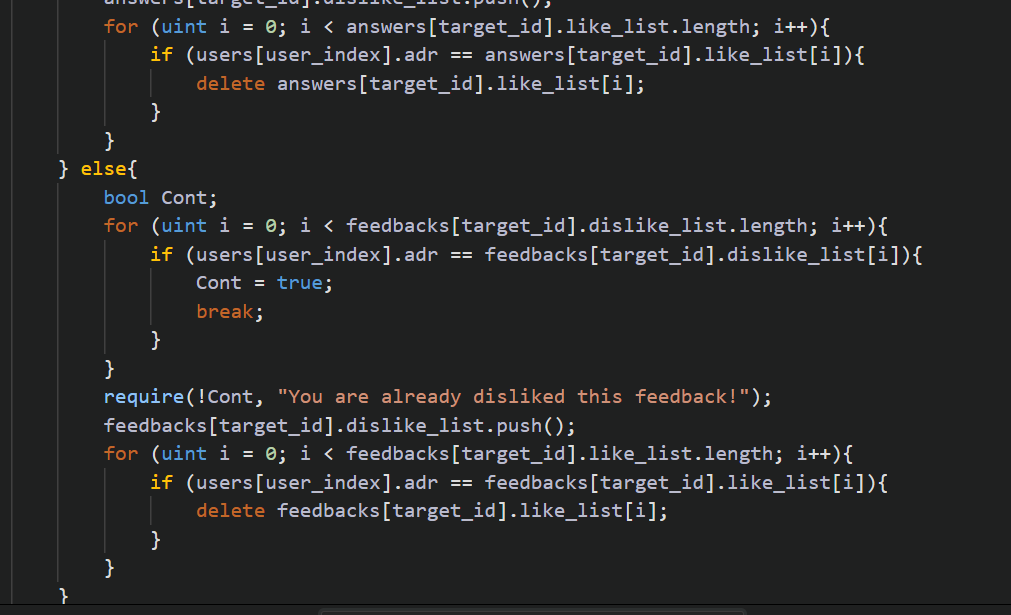


Рисунок – Функция добавления дизлайка

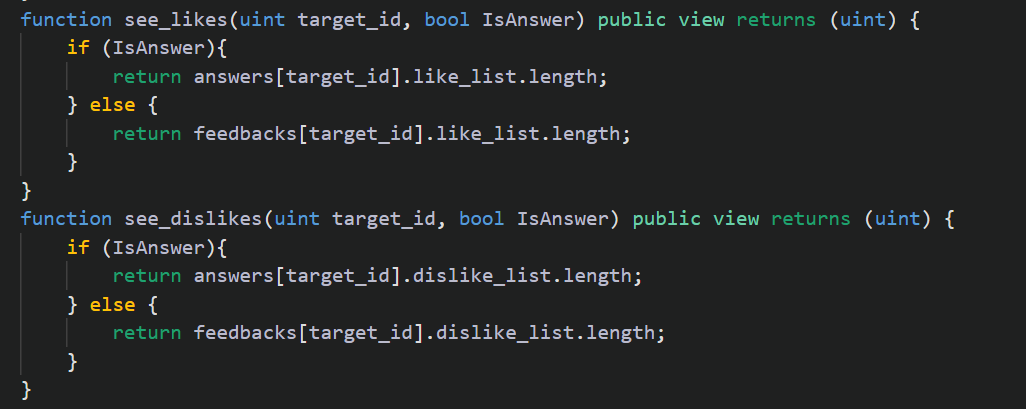


Рисунок – Просмотреть лайки и дизлайки

Кроме того, некоторые функции по приведению типов были реализованы в контракте, но нужны они были лишь для внутреннего взаимодействия логики и для пользователя были недоступны.

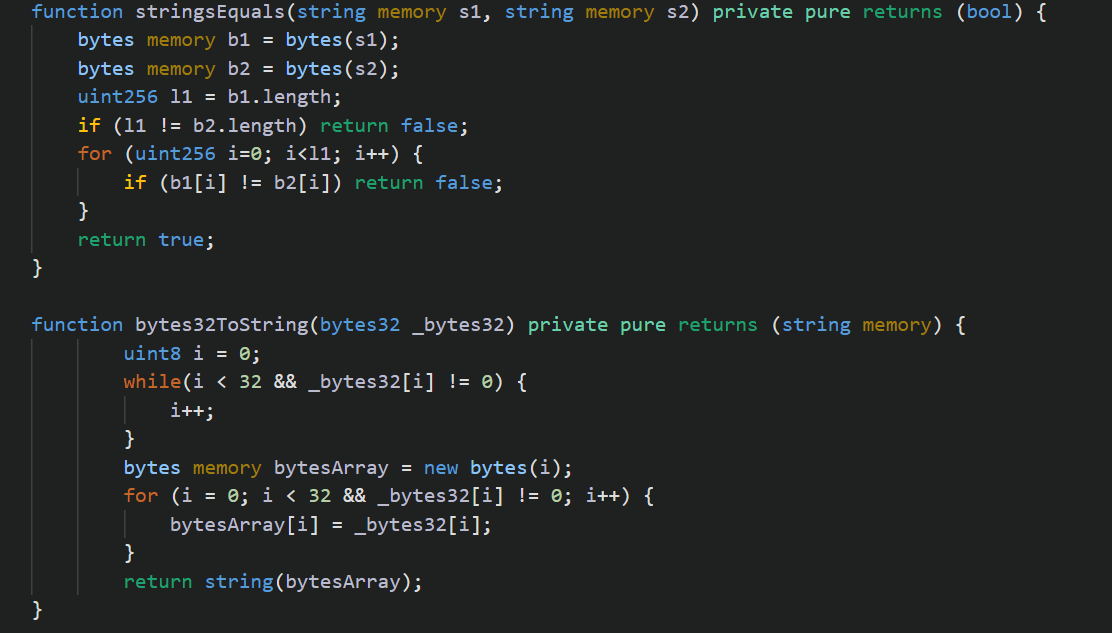


Рисунок – Функции, необходимые при работе со строками и хэшами

Результаты работы:

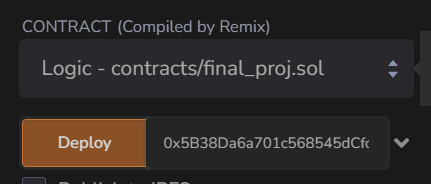


Рисунок – Вызов контракта

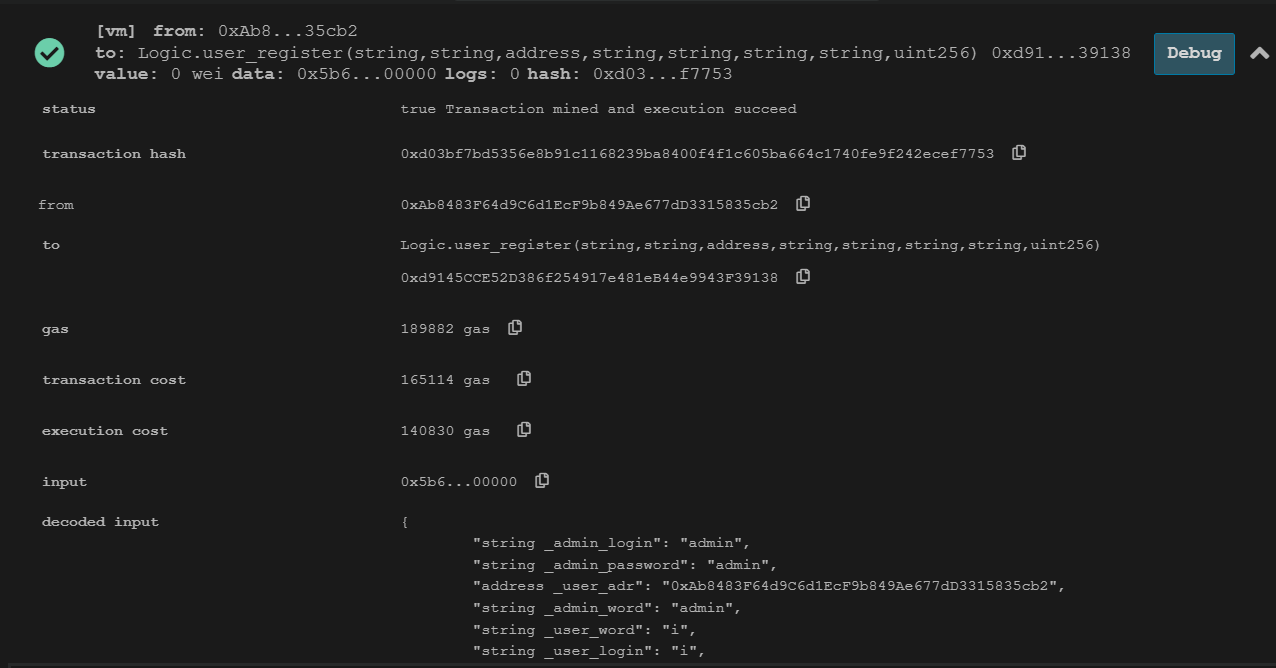


Рисунок – Вызов функции регистрации

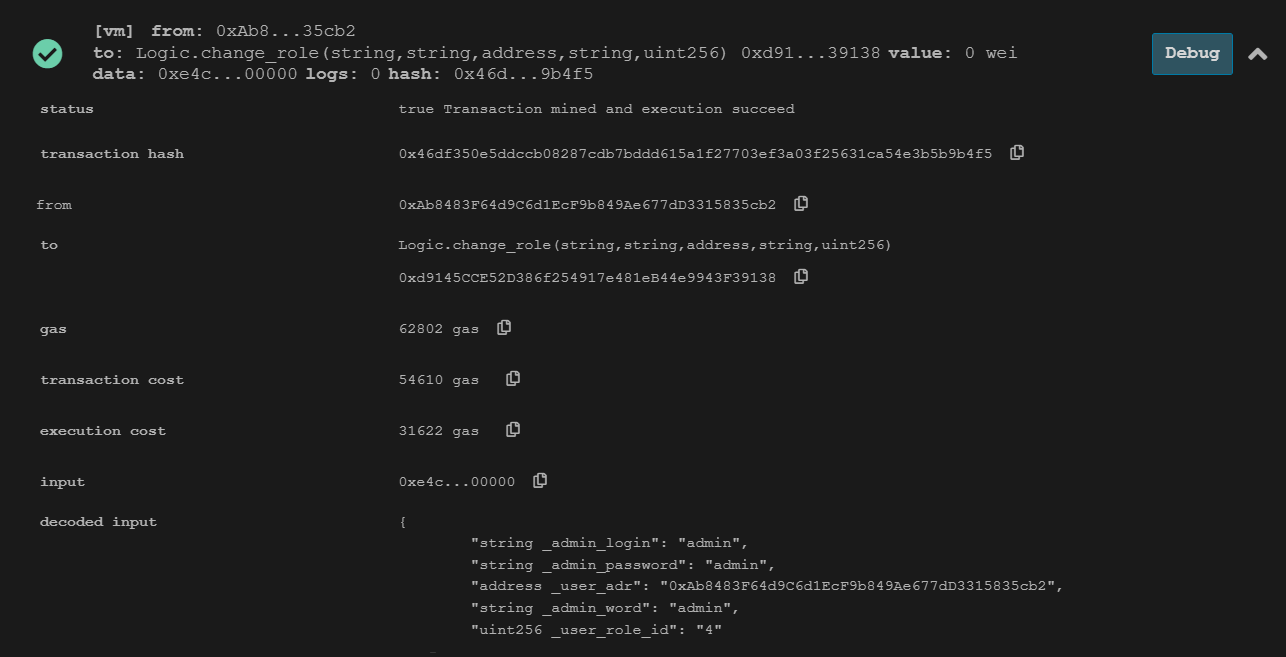


Рисунок – Вызов функции смены роли



Рисунок – Функция добавления отзыва



Рисунок – Функция добавления ответа на отзыв



Рисунок – Вызов функции добавления лайка



Рисунок – Вызов функции добавления дизлайка

Вывод: разработали смарт-контракт для финального проекта, закрепив полученные знания на языке смарт-контрактов Solidity, полученные в течение выполнения предыдущих практических работ.