Белорусский государственный университет информатики и радиоэлектроники

Кафедра информатики

Лабораторная работа № 3

**Семантический анализатор**

Выполнила студентка. гр. 453503: Долматович А.С.

Проверил ассистент КИ: Шиманский В.В.

Минск, 2017

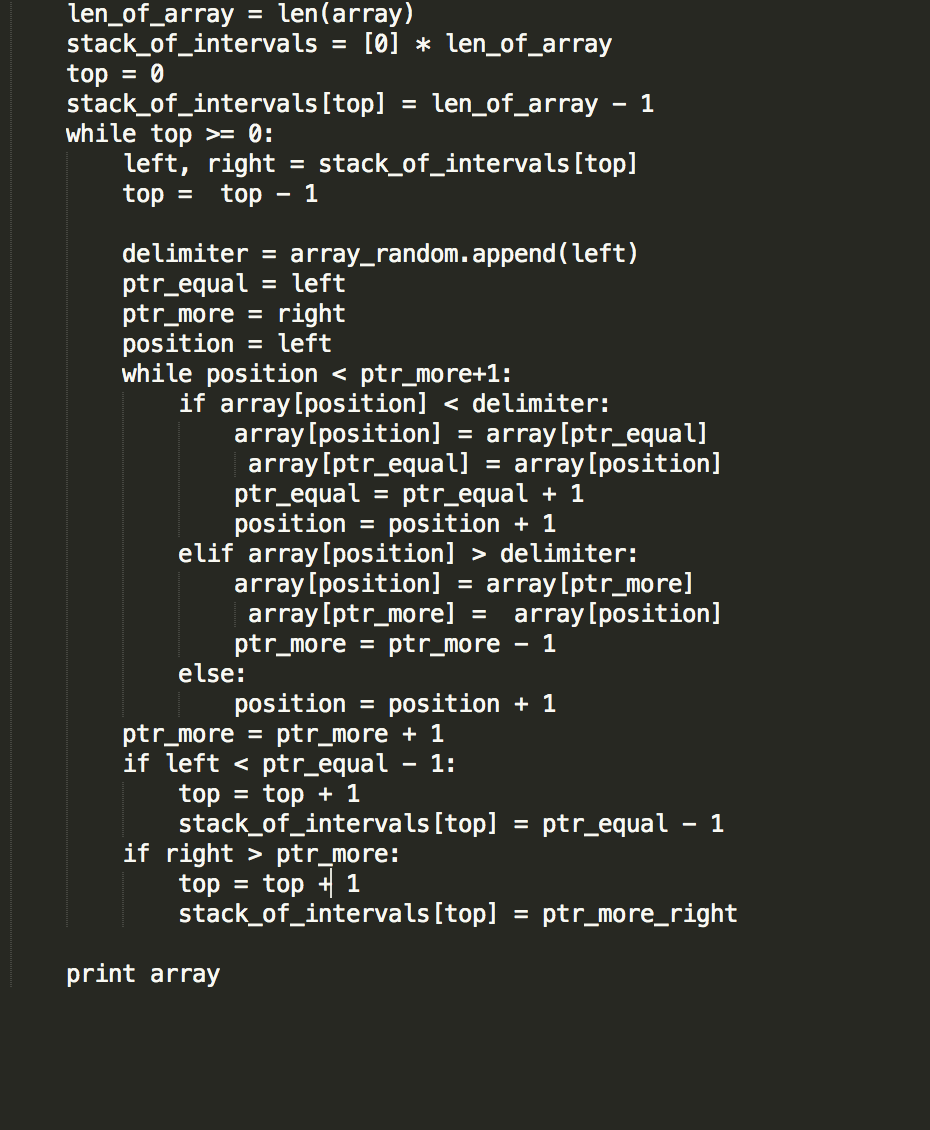
**1.Постановка задачи**

В данной работе ставится задача исследовать область семантических анализаторов, рассмотреть существующие аналоги и написать свой собственный семантический анализатор, способный распознавать семантические ошибки подмножества языка программирования.

В качестве анализируемого подмножества языка программирования будет использован язык программирования Python.

Для написания анализатора использован язык программирования Python.

На рис. 1.1 показан код программы на языке Python:



**Рис. 1.1. Код программы на языке Python**

**2.Теория**

[**Семантический анализ**](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D0%BC%D0%B0%D0%BD%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7). Дерево разбора обрабатывается с целью установления его семантики (смысла) — например, привязка идентификаторов к их декларациям, типам, проверка совместимости, определение типов выражений и т. д. Результат обычно называется «промежуточным представлением/кодом», и может быть дополненным деревом разбора, новым деревом, абстрактным набором команд или чем-то ещё, удобным для дальнейшей обработки.

**Семантика программы** – внутренняя модель (база данных) множества именованных объектов, с которыми работает программа, с описанием их свойств,  характеристик и связей.

Семантика программы тоже не обладает структурной целостностью и представлена фрагментарно, но при этом связана с синтаксисом следующим образом:

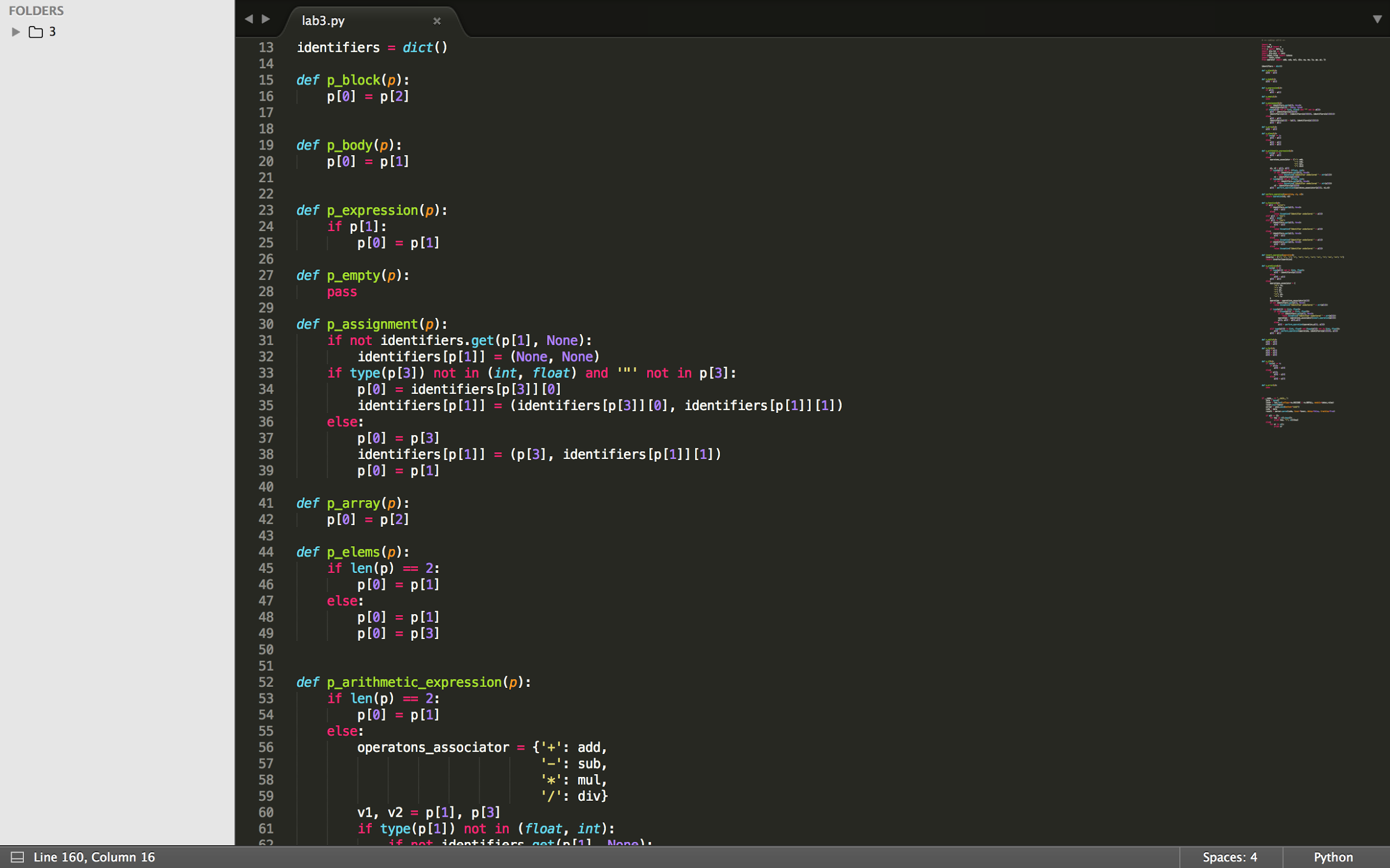
* один и тот же семантический объект (например, переменная) может встречаться в различных, синтаксически несвязанных частях программы;
* синтаксические конструкции *описаний, определений и объявлений* являются **источником семантики**объектов программы, они «заявляют» о существовании объектов и задают их свойства;
* синтаксические конструкции, связанные с *действиями, выполняемыми над объектами*, являются **потребителями семантики,**их интерпретация, корректность, «смысл» зависят от семантических свойств объекта;
* первичным источником семантики является лексический анализ. Некоторые из лексем (например, идентификаторы и константы) наряду с **классом лексемы**(обозначение выходной единицы лексического анализа), т.е. **символом**(обозначение той же единицы на входе синтаксического анализатора) имеют **значение.**

Задача семантического анализа, т.е. «описания смысла» фразы относится скорее к области искусственного интеллекта. Ее неформализуемость означает, что она  **не имеет формальных средств описания,** например, языков**.**Следовательно, семантическая модель языка разрабатывается в каждом случае уникально, здесь отсутствует общий подход, а имеет место набор частных решений и рекомендаций. Отсюда и уникальность семантики языка.

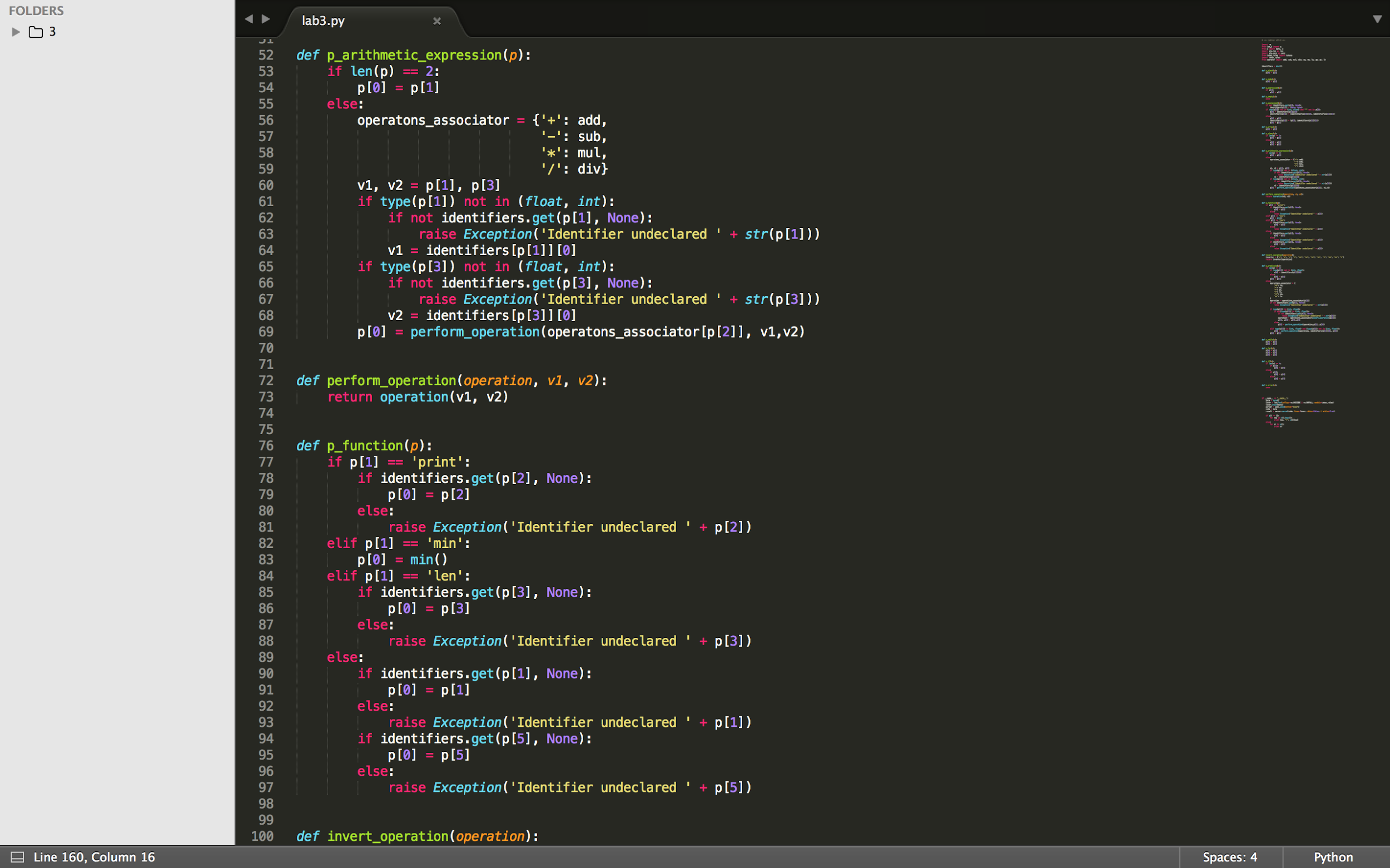
**3. Программа и комментарии**

В результате выполнения работы использовали уже написанный лексический и синтаксический анализатор. Из выражений, уже выделенных, составлялись синтаксические конструкции по некоторым правилам. В работе анализировался код со стороны правильной очередности присваиваний и правильности арифметических операций (деление на ноль).

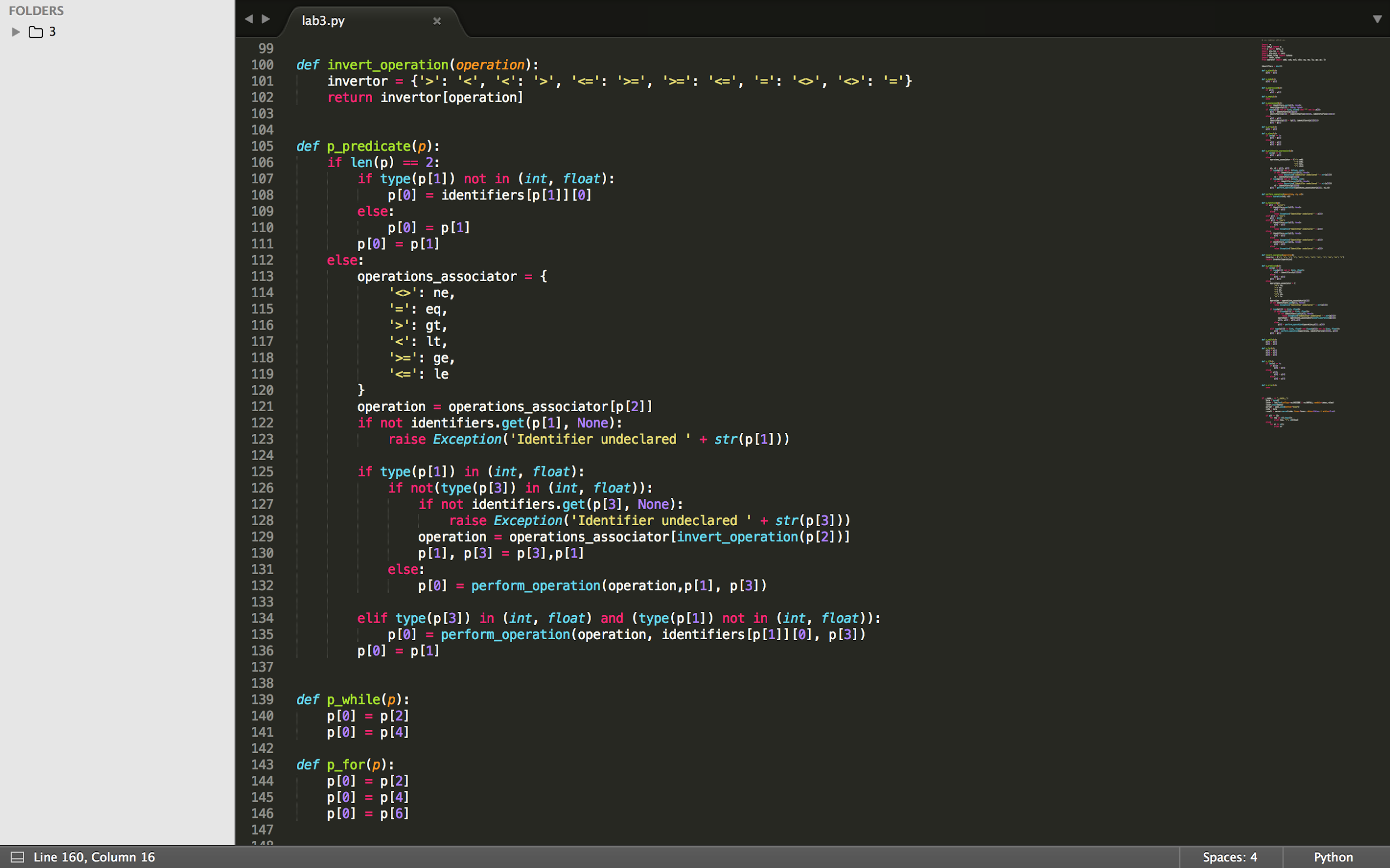
Для построения семантического анализаторы был использован следующий код (Рис. 3.1 - 3.4.):



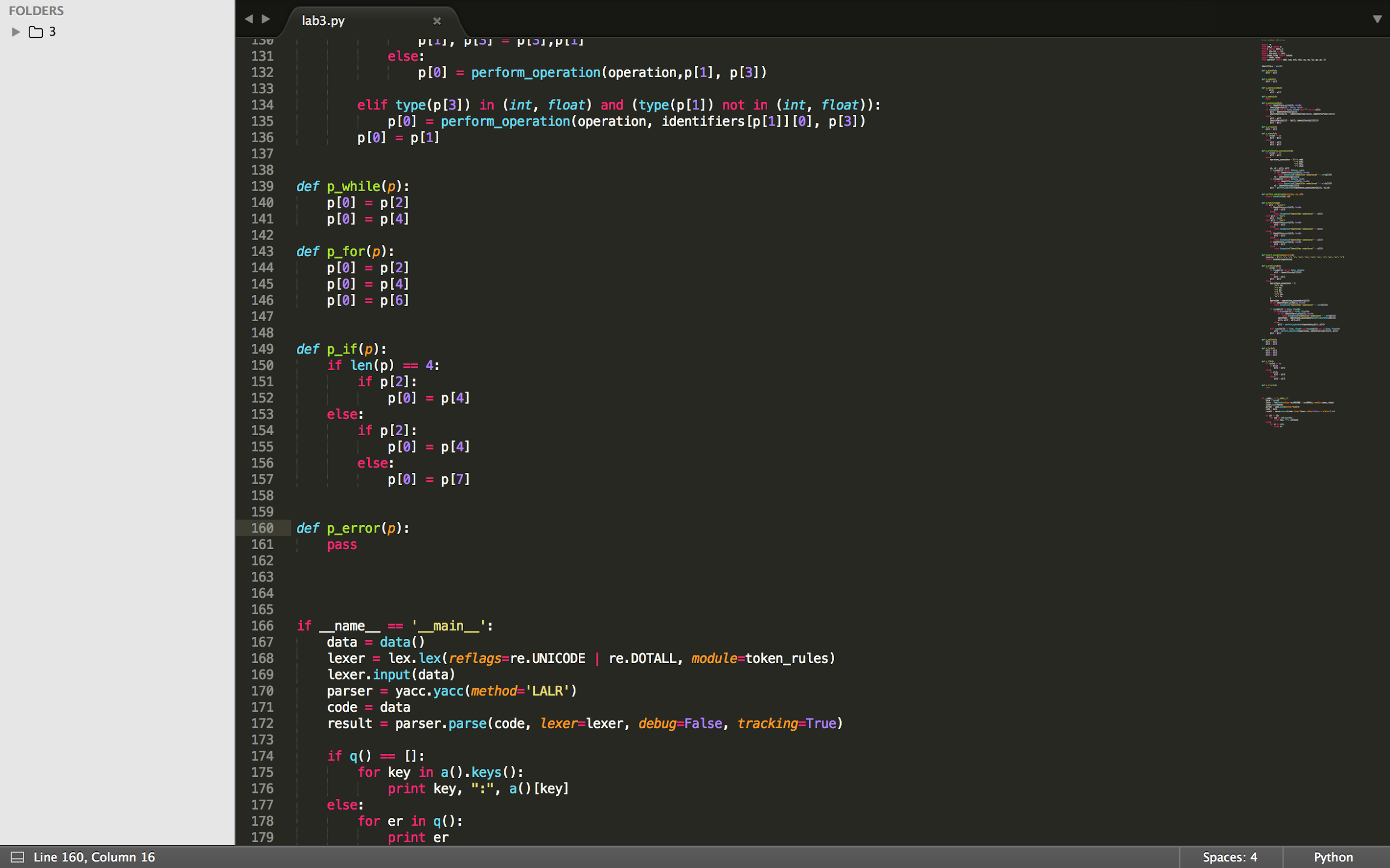
**Рис. 3.1. Код программы семантического анализатора**



**Рис. 3.2. Код программы семантического анализатора (продолжение)**



**Рис. 3.3. Код программы семантического анализатора (продолжение)**



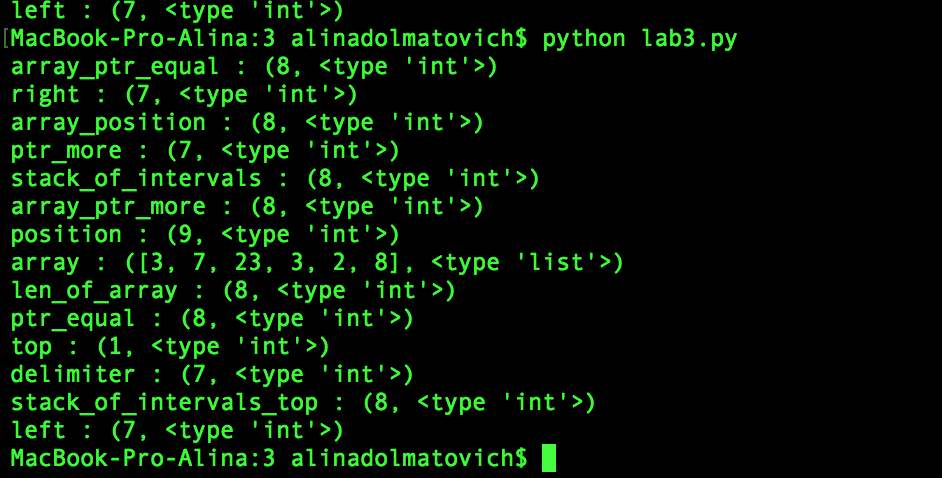
**Рис. 3.4. Код программы семантического анализатора (продолжение)**

**4. Результаты выполнения**

**4.1 Выполнение без ошибок**

Если в исходном тексте ошибки не обнаруженный, тогда выводится список распознанных идентификаторов в исходном коде программы.

Результат (Рис. 4.1.):

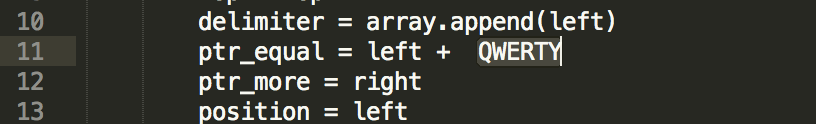


**Рис. 4.1. Результаты выполнения программы**

**4.2 Выполнение программы с допущением ошибок**

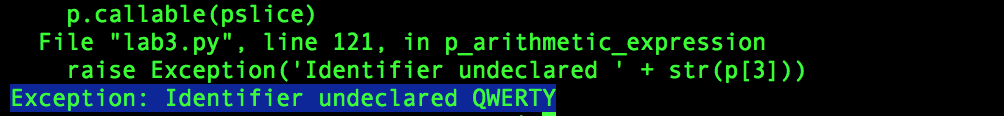
Для обнаружения ошибок необъявленных переменных сначала создаётся словарь вида: «имя переменной»: «значение». Затем происходит проход по дереву в порядке выполнения. При определении текущей конструкции как «присваивание» производится добавление имени переменной с неопределенным значением в словарь. При вычислении арифметического выражения (присваивание, арифметические операции, операции сравнения) при наличии идентификатора в выражении производится поиск в словаре переменных. Если данная переменная не объявлена или не определена, то выбрасывается исключение.

Пример кода с необъявленной переменной (Рис. 4.2.):



**Рис. 4.2. Код программы с необъявленной переменной**

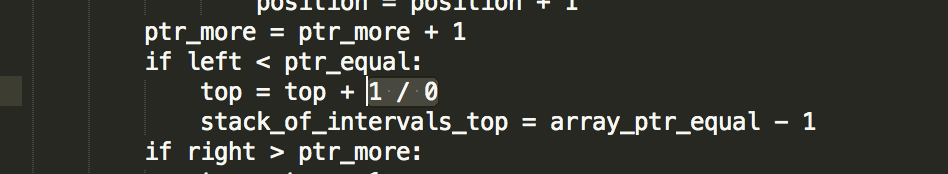
Результат:



**Рис. 4.3. Результат работы**

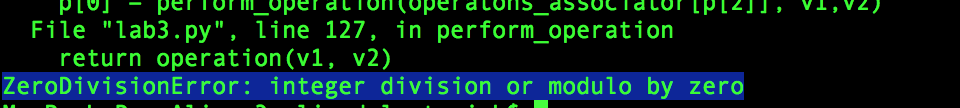
Для обнаружения ошибки «деление на нуль» используем тот же словарь вида («имя переменной»: «значение»). Затем совершаем проход по дереву в порядке выполнения. При определении текущей конструкции как «арифметическое выражение» производится поиск идентификатора в выражении (при наличии). Вместо идентификатора подставляется значение (числовое, в данном случае) и выражение продолжает вычисление. При возникновении деления на нуль выбрасывается ошибка.

Пример кода с делением на нуль:



**Рис. 4.3. Код программы с делением на ноль**

Результат:



**Рис. 4.4. Результат работы**

**5. Выводы**

В результате выполнения работы были получены знания о семантических анализаторах и их предназначении.

В итоге работы был построен свой собственный простой семантический анализатор на основе уже имеющегося лексического и синтаксического для анализируемого подмножества языка программирования Python. Сам анализатор написан на языке программирования Python.

Данный анализатор способен обнаружить семантические ошибки в коде, уведомлять о них, генерируя исключения. Анализатор способен построить таблицу встречающихся лексем и вывести результат. Если среди лексем есть ошибки, тогда на консоль выводятся строки, в которых встречаются лексические ошибки.

Было замечено, что для применения семантического анализа нужно непрерывно следить за ходом выполнения программы.