

Министерство образования Республики Беларусь
Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей
Кафедра информатики

Дисциплина: Методы трансляции

Отчет по лабораторной работе №4
Семантический анализатор.

Выполнил:
студент гр. 953504 Басенко К. А.

Проверил:
Шиманский В.В.

Минск 2022

Содержание

1. Постановка задачи. 3
2. Теория. 5
3. Результат работы анализатора. 7
4. Выводы. 8

1. Постановка задачи

В данной работе ставится задача исследования области семантических ошибок, подробному изучению теории семантики: приведение типов, операции с различными типами. Основной целью работы является семантика, то есть возможность интерпретатора распознавать типы. Это является следующим шагом анализа текста программы – семантический, существенно отличающийся от двух предыдущих – лексического и синтаксического. Таким образом, программа выполнит 3 фазу – выполнение приведения типов и затронет некоторый функционал, связанный с работой интерпретатора

Исследуемый код языка представлен ниже:

```
1  _ = "string string";
2
3  _ =          5 + 5;
4
5  _ = 1 + 2;
6  _ = 0 -          3;
7  _ = 9 / 5;
8  _ = 4 * 3          ;
9
10 _ = (1 +2) * 3;
11 _ = (9 - 4)/7;
12 _ = 5 + 3 - 2;
13
14 float prevar;
15 int variable = 2;
16
17 const int _1 =1; const int _2= 2;const int _3=3;
18
19 var a = 3d;
20 double b = a + 5;
21 var c=(1.5435e6+b)*0.543f;
22
23
24 (_, _) = (2f, 2d);
25 (_, _) = (2m, 2u);
26 (_, _) = (2l, 2L);
27 (_, _) = (2ul, 2UL);
28
29 a += c - b;
30 b -= (a + c) * 0.11e2D;
31 c *= a * b;
32
33 _ = a < b;
34 _ = b >= c;
35
```

```

36 int func(int a, int b, int c)
37 {
38     for (int i = 0; i < b; i++)
39     {
40         a--;
41     }
42
43     while (c > 0)
44     {
45         a *= a;
46         c--;
47     }
48
49     return a;
50 }
51
52 _ = func(4, 5, 2);
53
54
55 string _se1 = 5;
56 int _se2 = "gerge";
57 _se1 = func(1, 2, 3);
58 _se2 = true;
59 .

```

2. Теория

Следующий шаг анализа текста программы – семантический, существенно отличается от двух предыдущих – лексического и синтаксического. И дело не столько в том, что фаза семантического анализа реализуется не формальными, а содержательными методами (т.е. на данный момент нет универсальных математических моделей и формальных средств описания «смысла» программы). Лексический и синтаксический анализ имеют дело со структурными, т.е. внешними, текстовыми конструкциями языка. Семантика же, ориентированная на содержательную интерпретацию, имеет дело с внутренним представлением «смысла» объектов, описанных в программе. Для любого, имеющего опыт практического программирования, ясно, что формальные конструкции языка дают описание свойств и действий над внутренними объектами, с которыми имеет дело программа. Для начала перечислим все, что их касается и лежит на поверхности:

- большинство объектов являются именованными. Имя объекта позволяет его идентифицировать, существуют различные области действия имен, соглашения об именах, различные умолчания и т.п. Все это относится к семантике;
- виды, сложность и набор характеристик объектов различаются в разных языках программирования и сильно зависят от области приложения языка (в этом смысле семантика языков программирования более разнообразна, нежели синтаксис и лексика).
- объекты связаны между собой (ссылаются друг на друга).

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

Теперь, когда у нас есть представление о синтаксической фазе, можно оценить ее центральную роль в организации процесса трансляции.

Лексические единицы независимы друг от друга и являются терминальными символами синтаксиса. Семантика программы тоже не обладает структурной целостностью и представлена фрагментарно, но при этом связана с синтаксисом следующим образом:

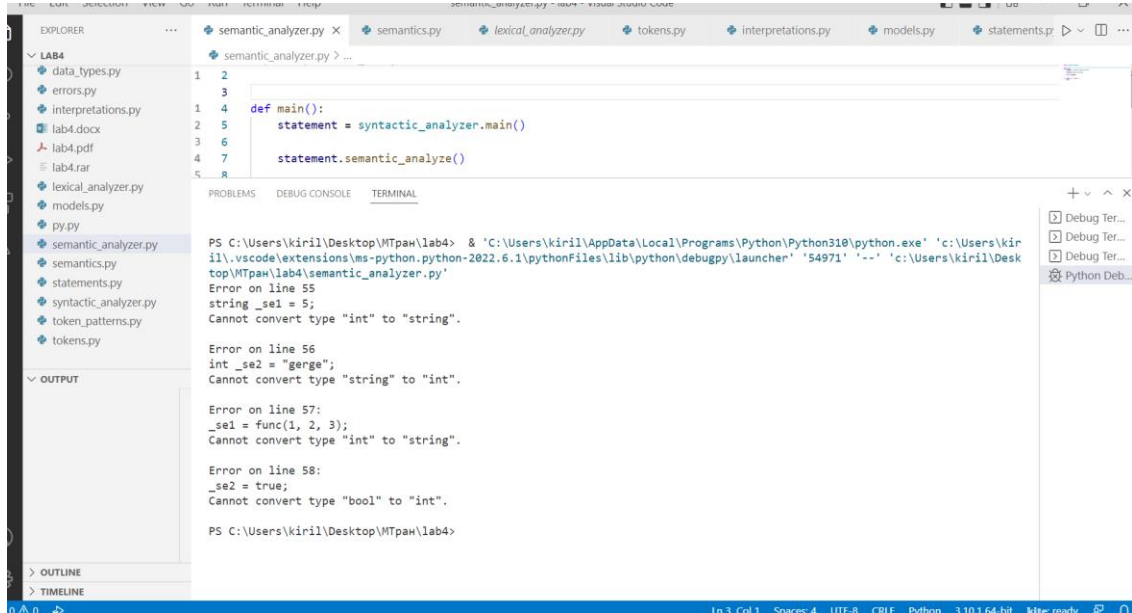
- один и тот же семантический объект (например, переменная) может встречаться в различных, синтаксически несвязанных частях программы;

- 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. Забегая вперед, можно заметить, что заключительная фаза трансляции (генерация кода, интерпретация) может рассматриваться как особые семантические действия, производимые над объектами;
- первичным источником семантики является лексический анализ. Значением лексемы является сама распознанная цепочка литер, она и представляет семантическую составляющую лексемы, которая и обрабатывается;
- лексемы, или то же самое, что терминальные символы входной строки (в терминах синтаксического анализа), ссылаются в семантические таблицы на свою семантику. Формирование семантической составляющей связано с движением снизу-вверх по синтаксическому дереву, от вершин – потомков к предкам.

Таким образом, семантическая составляющая транслятора тоже является фрагментарной (набор семантических процедур, соответствующих правилам грамматики) и объединяется в единое целое только в рамках синтаксического дерева.

3. Результат работы анализатора

В ходе работы анализатора проверяются типы операндов, и, при несовместности типов, выводится ошибка:



The screenshot shows a Visual Studio Code editor with a project named 'LAB4'. The file explorer on the left lists several Python files: data_types.py, errors.py, interpretations.py, lab4.docx, lab4.pdf, lab4.rar, lexical_analyzer.py, models.py, py.py, semantic_analyzer.py, semantics.py, statements.py, syntactic_analyzer.py, token_patterns.py, and tokens.py. The 'semantic_analyzer.py' file is open in the editor, showing a main function that calls syntactic_analyzer.main() and then semantic_analyze(). The terminal at the bottom displays the command to run the script and the resulting errors:

```
PS C:\Users\kiril\Desktop\МТр\lab4> & 'C:\Users\kiril\AppData\Local\Programs\Python\Python310\python.exe' 'c:\Users\kiril\.vscode\extensions\ms-python.python-2022.6.1\pythonFiles\lib\python\debugpy\launcher' '54971' '--' 'c:\Users\kiril\Desktop\МТр\lab4\semantic_analyzer.py'
Error on line 55:
string _se1 = 5;
Cannot convert type "int" to "string".

Error on line 56:
int _se2 = "gerge";
Cannot convert type "string" to "int".

Error on line 57:
_se1 = func(1, 2, 3);
Cannot convert type "int" to "string".

Error on line 58:
_se2 = true;
Cannot convert type "bool" to "int".

PS C:\Users\kiril\Desktop\МТр\lab4>
```

4. Выводы

Я провёл семантический анализ выбранного языка, сформировано его подмножество, придуман способ, который и способен отлавливать ошибки между разными типами и показаны 4 отловленные ошибки в коде программы на языке Lisp, успешно отлавливаемые синтаксическим анализатором.

Сложность заключалась в создании отдельного словаря, куда и записываются все переменные с их значениями, которые язык программирования Python, благодаря своей динамической типизации, способен отловить ошибку в преобразованиях, а программа только выводит её. Язык Lisp также динамический язык программирования как и Python, поэтому не было проблем в данной лабораторной работе.

Семантический анализатор представляет собой третью фазу компилятора, основная задача которого состоит в создании блока программы, решающей проблему с типами и работой их в коде. В дальнейшем это станет необходимым для создания виртуальной машины или, другими словами, интерпретатора.

Язык разбора Python — общего назначения, ориентированный на повышение производительности разработчика и читаемости кода, в то же время включает большой объём полезных функций. Основные архитектурные черты — , , полная , механизм , поддержка и удобные высокоуровневые . Код в Python организовывается в функции и , которые могут объединяться в (они в свою очередь могут быть объединены в пакеты).

Популярной реализацией Python является интерпретатор , поддерживающий большинство активно используемых платформ. Он распространяется под Python Software Foundation License, позволяющей использовать его без ограничений в любых приложениях. Есть реализации интерпретаторов для , , и других