Assignment 4

Exercise 1

Let's consider the following context-free grammar. We are creating a calculator handling identifiers.

```
S -> I; S | ε
I -> ident := E | E | ε
E -> E + T | E - T | T
T -> T * F | T / F | F
F -> ident | const | (E)
```

Question 1

Give a simple interface (API), for a symbol table adapted to this calculator.

1. Define a symbol, keep its value;
2. Test if a symbol is defined.

Question 2

With your favorite parser, create semantic rules associated to this grammar.

Implemented in `p1.py`, with lib files in dir `ply`.

```
$ echo "a := 5; a * a + 2; a * (a - 3);" | python p1.py
[None, 27, 10]
```

```
$ echo "b := 5; a * a + 2; a * (a - 3);" | python p1.py
ValueError: symbol a is not defined.
```

Question 3

As illustrated in previous section.

Exercise 2

Question 1

Implemented in `p2.py`.

The output of the program is a list of nested tuple.

```
$ echo "int foo(); int bar(string a, int b);" | python p2.py
[
    ('int', 'foo', []),
    ('int', 'bar', [('string', 'a'), ('int', 'b')])
    ]
```

Question 2

This feature is added. For example, following declarations are duplicated.

```
$ echo "int foo(int a); int foo(int b);" | python p2.py
Traceback (most recent call last):
       ...
       ValueError: func define conflit "foo$int".
```

While this is working.

```
$ echo "int foo(int a); int foo(int b, int c);" | python p2.py
[
    ('int', 'foo', [('int', 'a')]),
    ('int', 'foo', [('int', 'b'), ('int', 'c')])
    ]
```