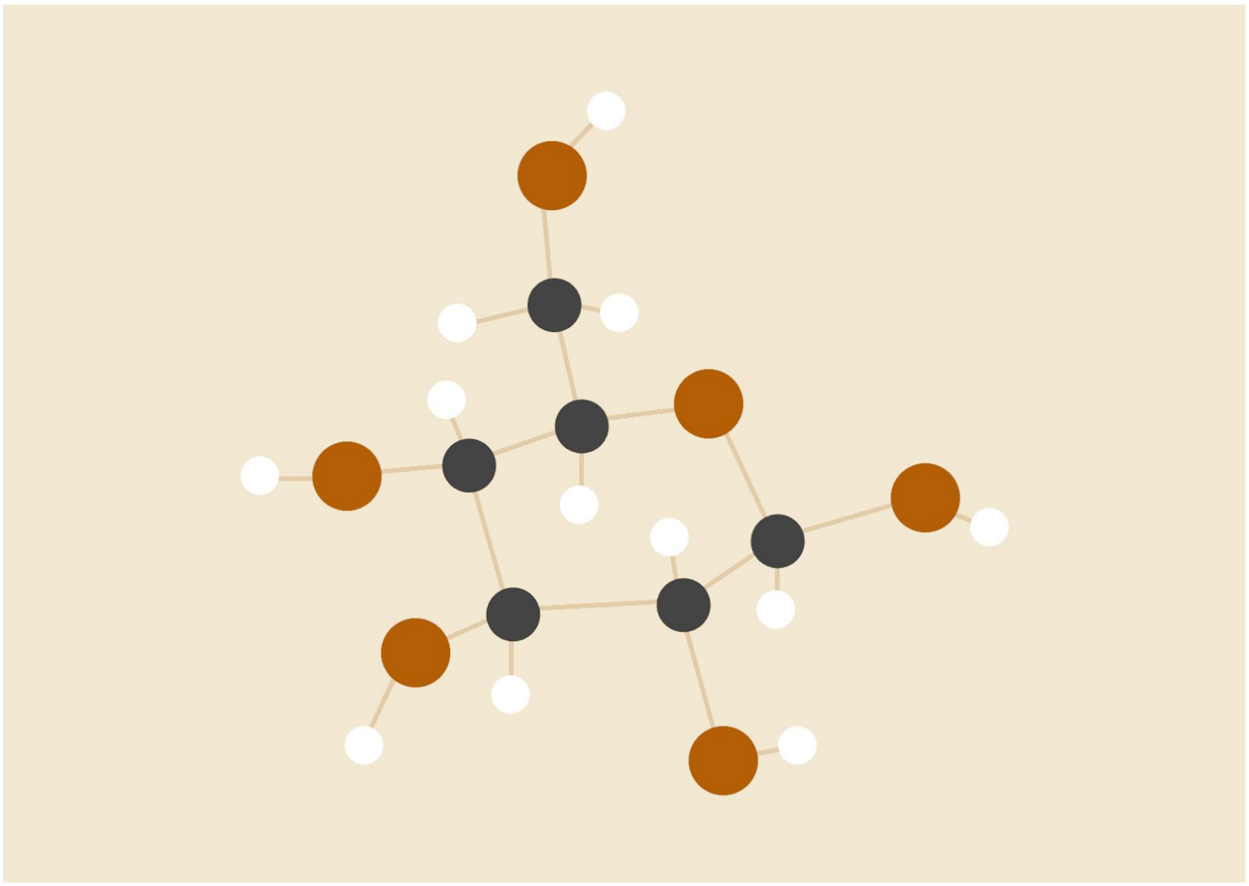
COMPILERS REPORT



Σπύρος Μπαξεβανάκης

23.05.2019

Κανόνες Γραμματικής

Stmt_list	->	Stmt Stmt_list | ε
Stmt	->	id = Expr | print Expr
Expr	->	Term Term_tail
Term_tail	->	Xor Term Term_tail | ε
Term	->	Factor Factor_tail
Factor_tail	->	Or Factor Factor_tail | ε
Factor	->	Atom Atom_tail.
Atom_tail	->	And Atom Atom_tail | ε
Atom	->	(Expr) | id | number
Xor	->	^
Or	->	|
And	->	&

Αποτελέσματα Ελέγχου για LL(1)

Σύνδεσμος στα αποτελέσματα.

- All nonterminals are reachable and realizable.
- The nullable nonterminals are: Stmt_list Term_tail Factor_tail Atom_tail.
- The endable nonterminals are: Atom_tail Atom Factor_tail Factor Term_tail Term Expr Stmt_list Stmt.
- No cycles.

The grammar is LL(1).

```
Stmt_list      ->     Stmt Stmt_list | .

Stmt           ->     id equal Expr | print Expr.
```

Expr -> Term Term_tail.

Term_tail -> Xor Term Term_tail | .

Term -> Factor Factor_tail.

Factor_tail -> Or Factor Factor_tail | .

Factor -> Atom Atom_tail.

Atom_tail -> And Atom Atom_tail | .

Atom -> lP Expr rP | id | number.

Xor -> carrot.

Or -> slash.

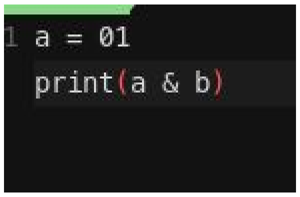
And -> amper.

FIRST & FOLLOW sets για τα μη τερματικά σύμβολα

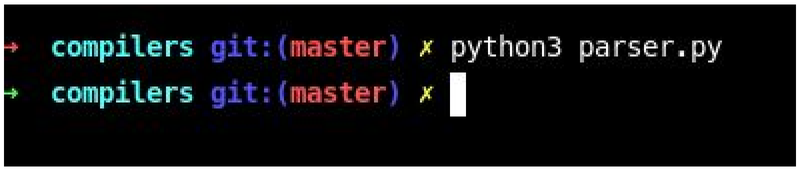
Xor	carrot	lP id number	no	no
Or	slash	lP id number	no	no
And	amper	lP id number	no	no

Αποτελέσματα εξόδου για έγκυρες και άκυρες μορφές εισόδου.

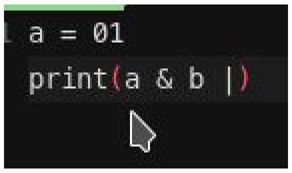
Parser:



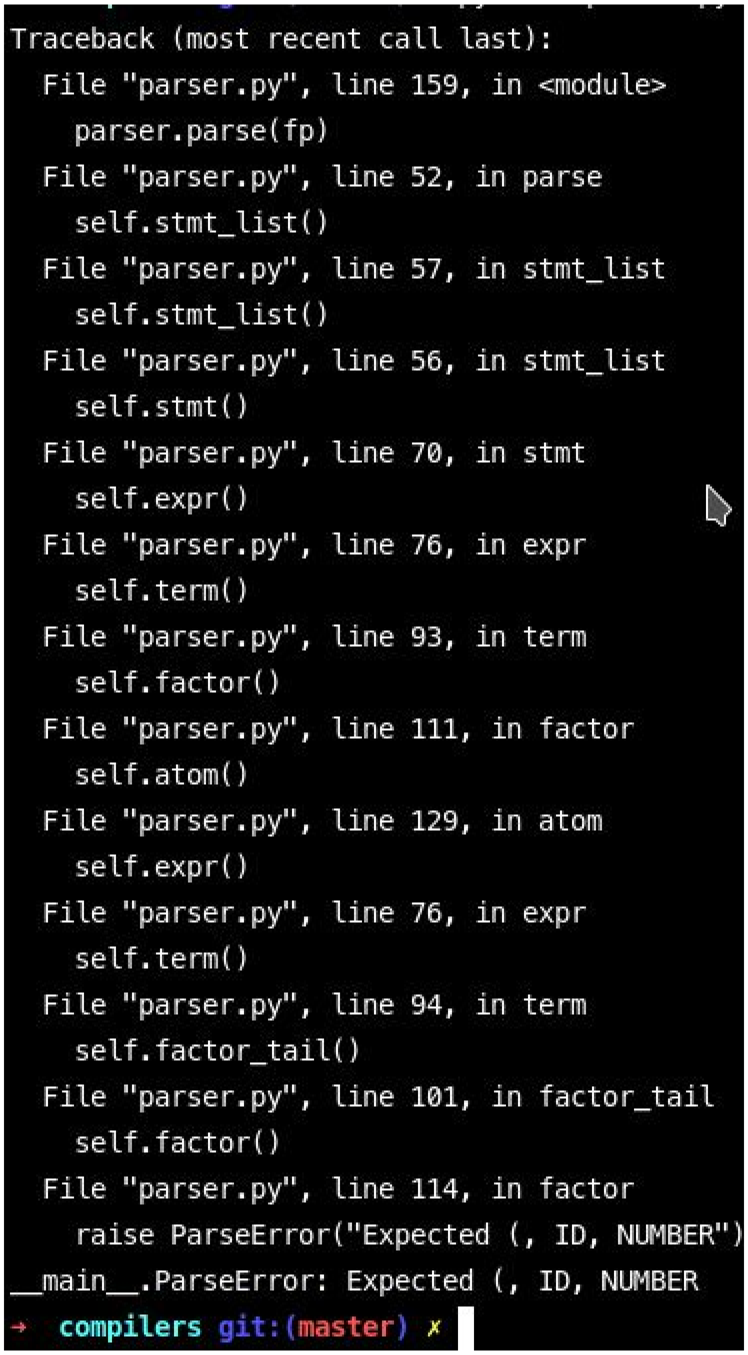
```
a = 01
print(a & b)
```



```
→  compilers git:(master) ✗ python3 parser.py
→  compilers git:(master) ✗
```

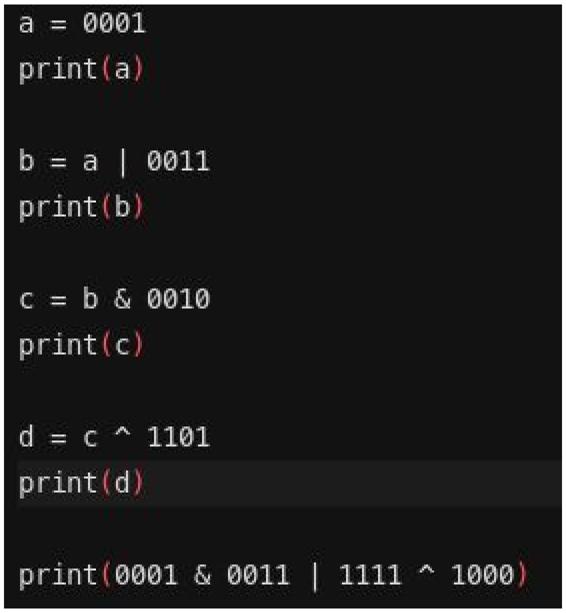


```
a = 01
print(a & b |)
```

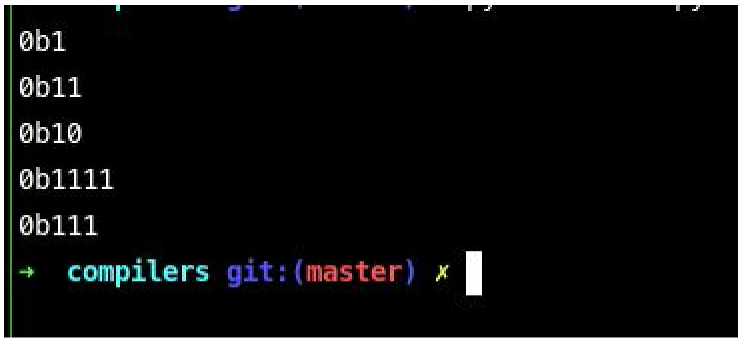


```
Traceback (most recent call last):
  File "parser.py", line 159, in <module>
    parser.parse(fp)
  File "parser.py", line 52, in parse
    self.stmt_list()
  File "parser.py", line 57, in stmt_list
    self.stmt_list()
  File "parser.py", line 56, in stmt_list
    self.stmt()
  File "parser.py", line 70, in stmt
    self.expr()
  File "parser.py", line 76, in expr
    self.term()
  File "parser.py", line 93, in term
    self.factor()
  File "parser.py", line 111, in factor
    self.atom()
  File "parser.py", line 129, in atom
    self.expr()
  File "parser.py", line 76, in expr
    self.term()
  File "parser.py", line 94, in term
    self.factor_tail()
  File "parser.py", line 101, in factor_tail
    self.factor()
  File "parser.py", line 114, in factor
    raise ParseError("Expected (, ID, NUMBER")
__main__.ParseError: Expected (, ID, NUMBER
→  compilers git:(master) ✗
```

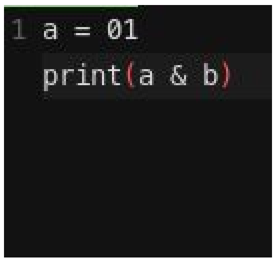
Runner:



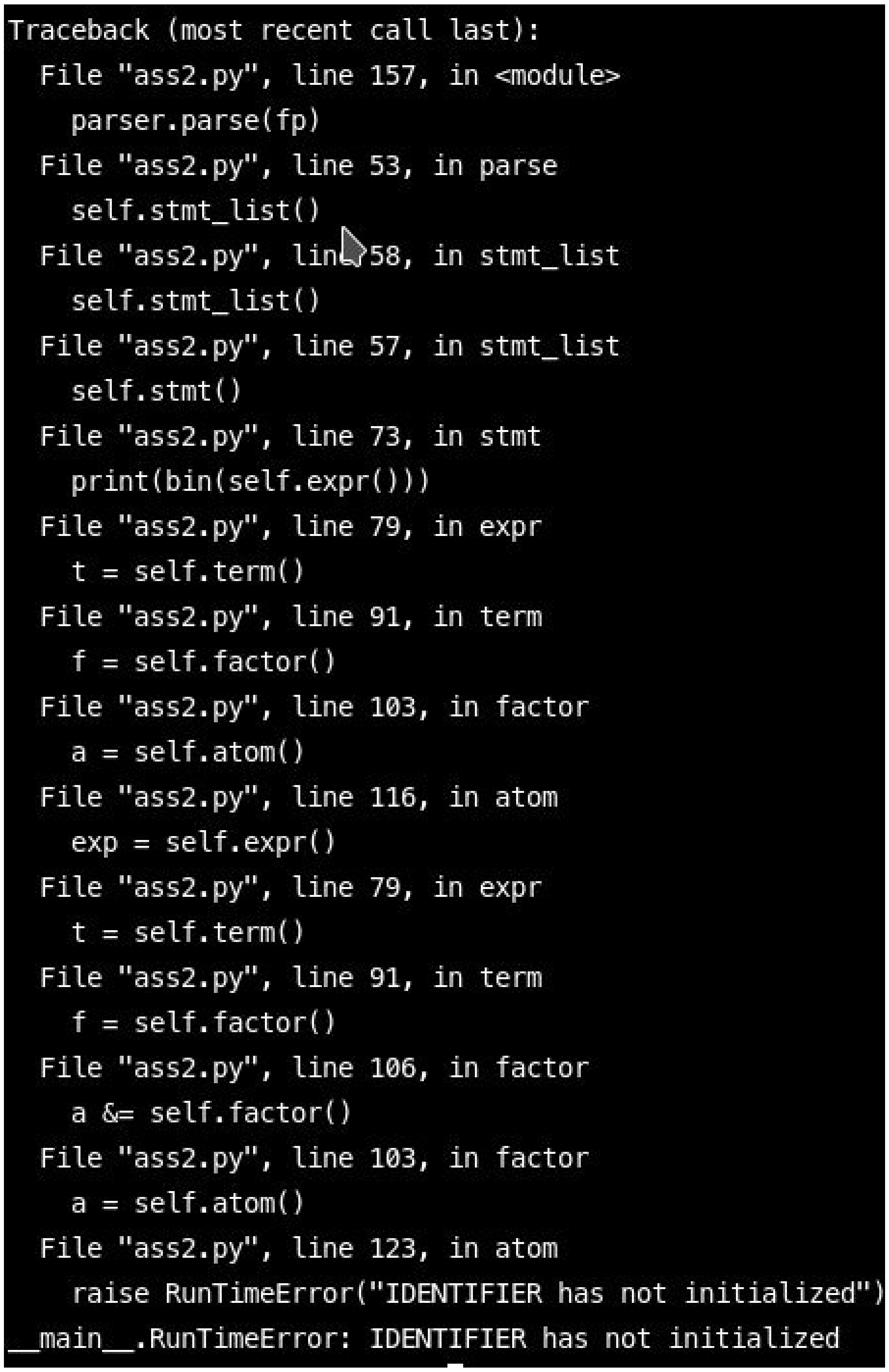
```
a = 0001
print(a)

b = a | 0011
print(b)

c = b & 0010
print(c)

d = c ^ 1101
print(d)

print(0001 & 0011 | 1111 ^ 1000)
```



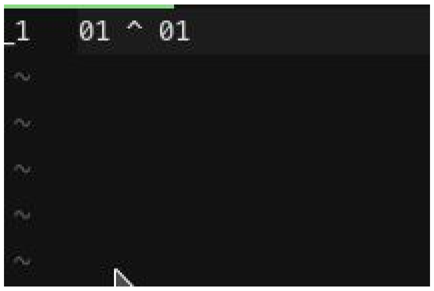
```
0b1
0b11
0b10
0b1111
0b111
→  compilers git:(master) ✗
```



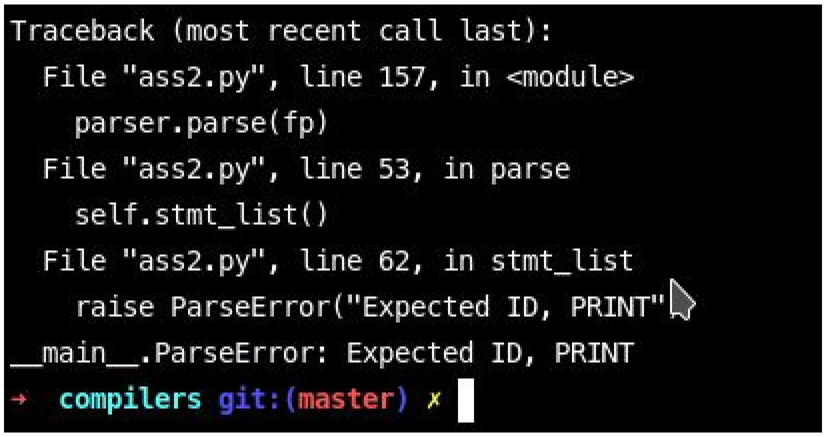
```
a = 01
print(a & b)
```



```
Traceback (most recent call last):
  File "ass2.py", line 157, in <module>
    parser.parse(fp)
  File "ass2.py", line 53, in parse
    self.stmt_list()
  File "ass2.py", line 58, in stmt_list
    self.stmt_list()
  File "ass2.py", line 57, in stmt_list
    self.stmt()
  File "ass2.py", line 73, in stmt
    print(bin(self.expr()))
  File "ass2.py", line 79, in expr
    t = self.term()
  File "ass2.py", line 91, in term
    f = self.factor()
  File "ass2.py", line 103, in factor
    a = self.atom()
  File "ass2.py", line 116, in atom
    exp = self.expr()
  File "ass2.py", line 79, in expr
    t = self.term()
  File "ass2.py", line 91, in term
    f = self.factor()
  File "ass2.py", line 106, in factor
    a &= self.factor()
  File "ass2.py", line 103, in factor
    a = self.atom()
  File "ass2.py", line 123, in atom
    raise RunTimeError("IDENTIFIER has not initialized")
__main__.RunTimeError: IDENTIFIER has not initialized
```



```
_1   01 ^ 01
~
~
~
~
~
```



```
Traceback (most recent call last):
  File "ass2.py", line 157, in <module>
    parser.parse(fp)
  File "ass2.py", line 53, in parse
    self.stmt_list()
  File "ass2.py", line 62, in stmt_list
    raise ParseError("Expected ID, PRINT"
__main__.ParseError: Expected ID, PRINT
→  compilers git:(master) ✗
```