

PLY

Python Lex-Yacc

Horácio Paganini, Gustavo Pimenta, João Victor Marchiori, Vitor yukio e Marcelly Costa

PLY (Python Lex-Yacc) – O que é?

- PLY (Python Lex-Yacc) é um conjunto de ferramentas em Python para escrever analisadores léxicos e sintáticos (lexers e parsers).
- O PLY é baseado no lex e yacc, que são ferramentas clássicas para criar analisadores léxicos e sintáticos em linguagens como C.
- O PLY permite que você defina gramáticas complexas em Python e gere analisadores automáticos para processar essas gramáticas.



PLY (Python Lex-Yacc) – O que é?

- **PLY Lex (analisador léxico)**

Uma ferramenta para criar analisadores léxicos (lexers) em Python. Ele ajuda a dividir o texto de entrada em tokens (símbolos léxicos), como números, identificadores e palavras-chave.

- **PLY Yacc (analisador sintático)**

Uma ferramenta para criar analisadores sintáticos (parsers) em Python. Ele ajuda a definir a estrutura gramatical de uma linguagem e a construir um analisador que pode reconhecer e interpretar essa estrutura.

PLY (Python Lex-Yacc) – O que é?

- Em resumo, as duas ferramentas devem funcionar juntas. Assim, o `lex.py` fornece uma interface para produzir tokens. `yacc.py` usa isso para recuperar tokens e invocar regras gramaticais.

-

PLY (Python Lex-Yacc) – Como funciona?

- **PLY Lex (Analizador Léxico):**

Define-se os tokens da linguagem. Tokens são unidades básicas de uma linguagem, como números, operadores e parênteses. No PLY, isso é feito usando expressões regulares.

No exemplo ao lado são definidos os tokens para números, operadores e parênteses. Também definimos expressões regulares para cada token.



```
import ply.lex as lex

tokens = (
    'NUMBER',
    'PLUS',
    'MINUS',
    'TIMES',
    'DIVIDE',
    'LPAREN',
    'RPAREN'
)

t_PLUS = r'\+'
t_MINUS = r'\-'
t_TIMES = r'\*'
t_DIVIDE = r'\/'
t_LPAREN = r'\('
t_RPAREN = r'\)'

def t_NUMBER(t):
    r'\d+'
    t.value = int(t.value)
    return t

t_ignore = ' \t'

lexer = lex.lex()
```


PLY (Python Lex-Yacc) – Como funciona?

- **PLY Yacc (Analizador Sintático):**

Define-se a gramática da linguagem usando o PLY Yacc. Isso envolve definir regras de produção que descrevem como os diferentes tokens podem ser combinados para formar expressões matemáticas válidas.

No exemplo ao lado define-se as regras de produção para expressões, termos e fatores. Cada regra descreve como os diferentes elementos da linguagem podem ser combinados



```
import ply.yacc as yacc

def p_expression(p):
    """
    expression : expression PLUS term
               | expression MINUS term
    """
    if p[2] == '+':
        p[0] = p[1] + p[3]
    elif p[2] == '-':
        p[0] = p[1] - p[3]

def p_expression_term(p):
    """
    expression : term
    """
    p[0] = p[1]

def p_term(p):
    """
    term : term TIMES factor
         | term DIVIDE factor
    """
    if p[2] == '*':
        p[0] = p[1] * p[3]
    elif p[2] == '/':
        p[0] = p[1] / p[3]

def p_term_factor(p):
    """
    term : factor
    """
    p[0] = p[1]

def p_factor(p):
    """
    factor : NUMBER
           | LPAREN expression RPAREN
    """
    if len(p) == 2:
        p[0] = p[1]
    else:
        p[0] = p[2]

parser = yacc.yacc()
```