Universidade Federal de Santa Catarina

Departamento de Informática e Estatística

INE 5426 - Construção de Compiladores

Relatório

Nome: Fabio Oliveira de Abreu (18100529)
Nome: Bruno Duarte Barreto Borges (18100519)
Nome: Erik Kazuo Sugawara (18100528)

Professor: Alvaro Junio Pereira Franco

1. Identificação dos tokens.

Resposta:

A identificação inicial dos tokens foi retirada da grámatica CC-2021-2, onde, para facilitar a organização dentro do código, separamos cada token em cinco grupos: palavras reservadas, operadores, símbolos especiais, constantes e identificadores. Seguem alguns exemplos de tokens com a sua expressão regular.

```
t_ASSIGN = r'\='
t_GT = r'\>'
t_LT = r'\<'
t_EQ = r'\=='
t_LE = r'\<='
t_GE = r'\>='
t_NEQ = r'\!='
t_PLUS = r'\+'
t_MULTIPLY = r'\*'
t_DIVIDE = r'\/'
t_REM = r'\%'
```

2. Produção das definições regulares para cada token.

Resposta:

Para produção das definições regulares, foram utilizadas expressões regulares. Segue um exemplo, da definição regular do token "FLOAT" utilizando a ferramenta PLY.

```
def t_float_constant(t):
    r'[+-]?\d+\.\d+'
    t.value = float(t.value)
    return t
```

3. Construção dos diagramas de transição para cada token

Resposta:

Os diagramas de transição dos tokens foram feitos em uma tabela xls, que se localiza na pasta xxxx.

4. Descrição de uma tabela de símbolos (como foi implementada), quais são os símbolos armazenados na tabela e quais são os atributos dos símbolos escolhidos para armazenar na tabela

Resposta:

A descrição da tabela de símbolos foi feita através dos tokens que eram retornados pelo analisador léxico, onde, dado uma entrada, a ferramenta identificava seu respectivo token. Com todos tokens identificados, é utilizado o método "print_table" que imprime a tabela de forma elegante no próprio terminal. Os atributos que foram escolhidos para serem mostrados na tabela foram: Token, Valor, Linha e Coluna.

```
def print_table(lexer):
    pattern = "{:^25} | {:^20} | {:^5} | {:^5}"
    print("\033[4m" + pattern.format("TOKEN", "VALUE", "L", "C") + "\033[0m")
    while True:
        tok = lexer.token()
        if not tok:
            break
        print(pattern.format(tok.type, tok.value, tok.lineno, find_column(tok)))

    for e in errors:
        print(e)
```

5. Se não usou ferramenta, uma descrição da implementação do analisador léxico (Usou diagramas de transição? Quais? Quantos? Se não usou diagramas de transição, então o que foi usado?)

Resposta:

Foi utilizado a ferramenta PLY (Python Lex-Yacc).

6. Se usou ferramenta, uma descrição detalhada da entrada exigida pela ferramenta e da saída dada por ela. É necessário haver exemplos pequenos da entrada e da saída gerada pela ferramenta com essa entrada.

Resposta:

Para a criação do analisador léxico, foi utilizada o PLY (Python Lex-Yacc), uma implementação das ferramentas lex e yacc para python. Nela, podemos criar uma lista de tokens a serem identificados pelo analisador, onde devem ser armazenados em uma lista chamada "tokens". As váriaveis e funções que se iniciam pelos caracteres "_t" são identificadas como um token pelo interpretador, onde podemos definir suas respectivas expressões regulares.