الجلسة الرابعة: إعراب جملة if-else

الهدف من الجلسة

كتابة قواعد تعليمة if-else

بناء برنامج لإعراب العبارة الشرطية.

خطوات بناء مترجم للشرط:

1. بناء ملف الماسح الذي يحتوي على جميع الـtokens التي نحتاجها إضافة إلى التعابير النظامية التي تعبر عن هذه الـ tokens.
2. كتابة قواعد الإعراب التي تعبر عن الشرط في كل الحالات:
   - وجود if لوحدها
   - وجود if-else
3. تحويل القواعد إلى كود بايثون، حيث كل قاعدة يكون لها تابع خاص لتحقيقها مع مراعاة أولوية العمليات.

ملف الماسح:

- إضافة الرموز التي نحتاجها:

```
# Token list
tokens = (
    'IF', 'THEN', 'ELSE',
    'ID', 'NUM',
    'LPAR', 'RPAR', 'SEMI',
    'LT', 'GT', 'LE', 'GE', 'EQ', 'NE',
    'OR', 'AND',
    'PLUS', 'MINUS', 'MULT', 'DIVS',
    'EQUAL'
)
```

رسم توضيحي 1 الرموز المطلوبة

- إضافة التعابير النظامية استجابة للرموز:

<u>التعابير البسيطة:</u>

```
# Regular expression rules for simple tokens
t_LPAR = r'\('
t_RPAR = r'\)'
t_SEMI = r';'
t_LT = r'<'
t_GT = r'>'
t_LE = r'<='
t_GE = r'>='
t_EQ = r'=='
t_NE = r'!='
t_OR = r'\|\|'
t_AND = r'&&'
t_PLUS = r'\+'
t_MINUS = r'-'
t_MULT = r'\*'
t_DIVS = r'/'
t_EQUAL = r'='
```

رسم توضيحي 2 الاستجابة للتعابير البسيطة

التعابير المركبة:

```
# Regular expression rule for numbers
def t_NUM(t):
    r'\d+'
    t.value = int(t.value)
    return t

# Regular expression rule for identifiers and keywords
def t_ID(t):
    r'[a-zA-Z_][a-zA-Z_0-9]*'
    t.type = reserved.get(t.value, 'ID')  # Check for reserved words
    return t

# Track line numbers
def t_newline(t):
    r'\n+'
    t.lexer.lineno += len(t.value)
```

رسم توضيحي 3 الاستجابة للتعابير الأكثر تعقيداً

حالات الشرط if:

حالة if لوحدها:

ليكن لدينا الكود التالي:

If(5 > 4) then a=10;

يمثل هذا الكود شرط اختبار كون عدد أكبر من الآخر.

القاعدة التي تعبر عن هذا الكود:

IF LPAR E RPAR THEN ST1 SEMI

يمكن لـ E أن تحتوي عمليات حسابية أومنطقية إضافة لإمكانية وجود عدد وحده.

تابع البايثون الوافق للقاعدة السابقة:

```
def p_ST_if_then(p):
    'ST : IF LPAR E2 RPAR THEN ST1 SEMI'
    p[0] = ('if_then', p[3], p[6])
```

القاعدة

الفعل الموافق للقاعدة

تمثل الجزء E2 من القاعدة

تمثل الجزء ST1 من القاعدة

حالة if-else:

ليكن لدينا الكود التالي:

If(5 > 4) then a=10;

else a=4;

القاعدة التي تعبر عن هذا الكود:

IF LPAR E2 RPAR THEN ST1 SEMI ELSE ST1 SEMI

تابع البايثون الذي يعبر عن هذه القاعدة:

```
def p_ST_if_then_else(p):
    'ST : IF LPAR E2 RPAR THEN ST1 SEMI ELSE ST1 SEMI'
    p[0] = ('if_then_else', p[3], p[6], p[9])
```

القواعد التي تعبر عن الحالات الشرطية السابقة:

```
Rule 0     S' -> S
Rule 1     S -> ST
Rule 2     ST -> IF LPAR E2 RPAR THEN ST1 SEMI ELSE ST1 SEMI
Rule 3     ST -> IF LPAR E2 RPAR THEN ST1 SEMI
Rule 4     ST1 -> ST
Rule 5     ST1 -> E
Rule 6     E -> ID EQUAL E
Rule 7     E -> E PLUS E
Rule 8     E -> E MINUS E
Rule 9     E -> E MULT E
Rule 10    E -> E DIVS E
Rule 11    E -> E LT E
Rule 12    E -> E GT E
Rule 13    E -> E LE E
Rule 14    E -> E GE E
Rule 15    E -> E EQ E
Rule 16    E -> E NE E
Rule 17    E -> E OR E
Rule 18    E -> E AND E
Rule 19    E -> MINUS E
Rule 20    E -> ID
Rule 21    E -> NUM
Rule 22    E2 -> E LT E
Rule 23    E2 -> E GT E
Rule 24    E2 -> E LE E
Rule 25    E2 -> E GE E
Rule 26    E2 -> E EQ E
Rule 27    E2 -> E NE E
Rule 28    E2 -> E OR E
Rule 29    E2 -> E AND E
Rule 30    E2 -> ID
Rule 31    E2 -> NUM
```

رسم توضيحي 4 القواعد التي تعبر عن شرط if-else

ملف المعرب

```
# Precedence rules (lowest to highest)
precedence = (
    ('right', 'EQUAL'),
    ('left', 'OR'),
    ('left', 'AND'),
    ('left', 'LT', 'GT', 'LE', 'GE', 'EQ', 'NE'),
    ('left', 'PLUS', 'MINUS'),
    ('left', 'MULT', 'DIVS'),
)

# Grammar rules
def p_S(p):
    '''S : ST'''
    print("Input accepted.")
    p[0] = p[1]

def p_ST_if_then_else(p):
    'ST : IF LPAR E2 RPAR THEN ST1 SEMI ELSE ST1 SEMI'
    p[0] = ('if_then_else', p[3], p[6], p[9])


def p_ST_if_then(p):
    'ST : IF LPAR E2 RPAR THEN ST1 SEMI'
    print('p[3]is:',p[3])
    p[0] = ('if_then', p[3], p[6])


def p_ST1_statement(p):
    'ST1 : ST'
    p[0] = p[1]

def p_ST1_expression(p):
    'ST1 : E'
    p[0] = p[1]

# Expression rules for E (assignment and arithmetic)
def p_E_assignment(p):
    'E : ID EQUAL E'
    p[0] = ('assign', p[1], p[3]) #semantic action

def p_E_binop(p):
    '''E : E PLUS E
         | E MINUS E
         | E MULT E
         | E DIVS E
         | E LT E
         | E GT E
         | E LE E
         | E GE E
         | E EQ E
         | E NE E
         | E OR E
         | E AND E'''
    p[0] = ('binop', p[2], p[1], p[3]) #binop is a binary operation
```

رسم توضيحي 5.1 إعراب قواعد الشرط

```python
def p_E_negative(p):
    'E : MINUS E'
    p[0] = ('neg', p[2])

def p_E_id(p):
    'E : ID'
    p[0] = ('id', p[1])

def p_E_num(p):
    'E : NUM'
    p[0] = ('num', p[1])

# Expression rules for E2 (conditional expressions only)
def p_E2_binop(p):
    '''E2 : E LT E
          | E GT E
          | E LE E
          | E GE E
          | E EQ E
          | E NE E
          | E OR E
          | E AND E'''
    p[0] = ('binop', p[2], p[1], p[3])

def p_E2_id(p):
    'E2 : ID'
    p[0] = ('id', p[1])

def p_E2_num(p):
    'E2 : NUM'
    p[0] = ('num', p[1])

def p_error(p):
    if p:
        print(f"Syntax error at '{p.value}'")
    else:
        print("Syntax error at EOF")

# Build the parser
parser = yacc.yacc()
```

رسم توضيحي 5.2 إعراب قواعد الشرط

تمارين:

- عدل الكود السابق ليقبل الشرط بالشكل التالي:

If(2+3) then x=1

انتهت الجلسة

إعداد:م. ريم جبيلي