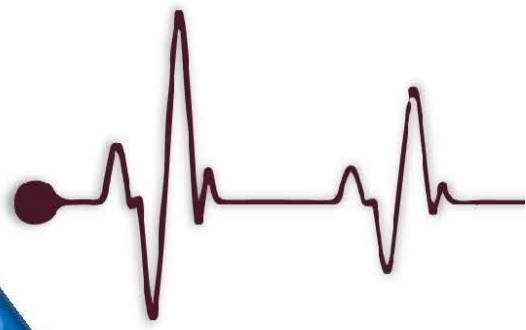
جامعة البعث
كلية الهندسة المعلوماتية
السنة الرابعة

لغات برمجة

القسم العملي

المحاضرة 4



ITP Team

2024/2023

Al-Baath University

IT Faculty

4th Year

Programming Languages

Practical Labs

Lab -4- Dealing with Ranges

Introduced By:
Eng. Mohammed Haj Hasan

Supervised By:
Dr. Osama Naser

Table of Contents

مقدمة:

إن المجالات Ranges في إكسل تشير إلى مجموعة من الخلايا المتجاورة ضمن ورقة العمل ذاتها Spreadsheet .

يتم تحديد المجال عن طريق تحديد عنوان الخلية التي تمثل بداية المجال مبتوعاً بعلامة النقطتين (:) من ثم عنوان الخلية التي تمثل نهاية المجال.

على سبيل المثال، "A1:B5" يمثل مجالاً من الخلايا يشمل جميع الخلايا من الخلية A1 إلى الخلية B5.

يمكن استخدام المجالات في مختلف توابع وصيغ إكسل وتطبيق التنسيقات أو إنشاء الرسوم البيانيةمما يتيح لنا العمل مع عدة خلايا في نفس الوقت وأداء العمليات على مجموعة من الخلايا دفعة واحدة.

في هذه الجلسة سوف نقوم ببناء مفسر Interpreter باستخدام لغة Python للتعامل مع المجالات في برنامج إكسل.

= (A1 : A3) * (B1 : B3)

خطوات بناء المفسر:

Lexical Analyzer:

- في هذه المرحلة سيتم بناء المحلل اللفظي Lexical Analyzer
- سنقوم بتعريف تابع tokenize الذي يأخذ كدخل المجال Range ويعيد كخرج مجموعة الرموز tokens.

```
# Lexical Analyzer
def tokenize(expression):
    tokens = re.findall(r'=|[A-Za-z]\d+|\d+|\+|-|\*|/|\(|\)|\:', expression)
    return tokens
```

- إن التابع السابق يستخدم التعابير المنتظمة من أجل استخراج مجموعة الرموز Tokens.

- = : من أجل استخراج الرمز الممثل لاشارة =
- [A-Za-z] : لاستخراج الرمز الممثل للمحارف في اسم الخلية.
- \d+ : لاستخراج الرمز الممثل للرقم.

Syntax Analyzer:

- في هذه المرحلة سيتم بناء المحلل القواعدي Syntax Analyzer
- تم تعريف التابع parser الذي يأخذ كدخل مجموعة الرموز و يعيد كخرج ناتج العملية الحسابية المطبقة بين المجالين المدخلين

```
# Syntax Analyzer
def parser(tokens):
    equal_token = tokens.pop(0)
    if equal_token not in ['='] :
        raise ValueError('Syntax Error : Excel Expressions must Starts with =')
    else:
        cells_range,operation = deal_with_range(tokens)
        first_range = cells_range[0]
        second_range = cells_range[1]
        first_range_values = []
        second_range_values = []
        for cell in first_range:
            temp = read_excel_cell(file_path,sheet_name,cell)
            first_range_values.append(temp)

        for cell in second_range:
            temp = read_excel_cell(file_path,sheet_name,cell)
            second_range_values.append(temp)

        if operation in ['*']:
            ranges_times(first_range,first_range_values,second_range,second_range_values)
        elif operation in ['/']:
            ranges_divide(first_range,first_range_values,second_range,second_range_values)
        elif operation in ['+']:
            ranges_add(first_range,first_range_values,second_range,second_range_values)
        else :
            ranges_minus(first_range,first_range_values,second_range,second_range_values)
```

:deal with range Function

```
# Dealing with Ranges
# Input  : List of Tokenes
# Output : Ranges of Cells, Operation between Ranges
def deal_with_range(tokens):
    left_range = []
    right_range = []
    all_ranges = []
    all_ranges_cells_name = []

    # get the Operation Index
    token_string = str(tokens)
    operation_token = re.findall(r'\+|-|\*|/', token_string)
    operation_index = tokens.index(operation_token[0])

    # get the Left Range
    for i in range(0,operation_index):
        left_range.append(tokens[i])

    # get the Right Range
    for i in range(operation_index+1,len(tokens)):
        right_range.append(tokens[i])

    # Combine Left and Right Ranges in one List
    all_ranges.append(left_range)
    all_ranges.append(right_range)

    # if Number of ( != Number of ) then Raise Syntax Error
    for my_range in all_ranges:
        total_left_bracket,total_right_brackets = get_brackets_number(my_range)

        if total_left_bracket > total_right_brackets:
            raise ValueError('Syntax Error : Missing Closing Bracket')
            return
```

✓ نلاحظ أن هذا التابع يأخذ كدخل مجموعة الرموز Tokens ويعيد كخرج العملية الحسابية المطبقة و المجالات

```
        elif total_right_brackets > total_left_bracket:
            raise ValueError('Syntax Error : Missing Opening Bracket')
            return

        # range syntax correct
        else:
            # Search for Colon, Previous Operand and the Next Operand to Defin Range Start and End
            # Colon Token index
            colon_token_index = my_range.index(':')

            # Starting and Ending of the Range
            range_start = my_range[colon_token_index-1]
            range_end = my_range[colon_token_index+1]

            # Get the Starting, Ending and Name of Range Cells
            range_end_ref_num = re.findall(r'\d+',range_end)
            range_start_ref_num = re.findall(r'\d+',range_start)
            range_ref_name = re.findall(r'[A-Za-z]',range_end)

            # Get the Cells between Start and End of Range ex: a1:a3 => a1 a2 a3
            range_cells_names = []
            for i in range(int(range_start_ref_num[0]),int(range_end_ref_num[0])+1):
                range_cells_names.append(str(range_ref_name[0])+str(i))

            all_ranges_cells_name.append(range_cells_names)

    return all_ranges_cells_name,operation_token[0]
```

:get brackets number Function

```python
# to get the Number of Brackets in a Single Range
def get_brackets_number(range_token):
    total_left_bracket = range_token.count('(')
    total_right_brackets = range_token.count(')')
    return total_left_bracket,total_right_brackets
```

:Testing the Code