



منتدی فیجوال بیسک

www.vb4arab.com



Arabic Institute For Programming & Technology
AIPT

قسم الأبحاث والتطوير

ورقة بحثية بعنوان:

لغة فجول بيسك

Excel & Word **والتعامل مع برنامجي**

إعداد:

أبو بكر شرف الدين سويدان

2004

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

وَالسُّوفِ بِعَمَلِكِ رَبِّكَ فَتَرْتَمِدُ

صِفَاتُ اللَّهِ الْعَظِيمِ

المحتوى

الصفحة	الموضوع	ر.م
1	مقدمة	1
أولاً: برنامج وورد		
2	إدراج مكتبات برنامج وورد	2
2	تعريف متغير يمثل برنامج وورد	3
3	الكائن APPLICATION	4
3	خصائص ووظائف وأحداث الكائن APPLICATION	5
4	الكائن DOCUMENT	6
4	تجمع الوثائق	7
5	بعض العمليات التي تجرى على الوثيقة	8
8	الكائن RECENTFILE	9
9	التعامل مع النصوص في الوثيقة	10
11	التعامل مع الجداول	11
14	الكائن PAGESETUP	12
15	أمثلة متنوعة	13
ثانياً : برنامج أكسل		
18	إدراج مكتبات برنامج أكسل	14
19	مصطلحات أساسية	15
20	ملف أكسل WORKBOOK	16
20	خصائص ووظائف الـ WORKBOOK	17
21	العمليات على ملفات أكسل وأوراق عملها	18
23	الخلايا والنطاقات	19
31	استخدام بعض الدوال التابعة لأكسل في فجول بيسك	20
35	الخاتمة	21

مقدمة:

قامت شركة مايكروسوفت بتصميم مجموعة البرامج التي أطلقت عليها اسم (مجموعة برامج المكتب Office Programs Group) لتلبي حاجات المستخدم من الحاسوب ، ومن أبرز هذه البرامج برنامج تحرير النصوص ومعالجتها (Microsoft Word) ، وبرنامج معالجة الجداول الإلكترونية (Microsoft Excel).

ولإفساح المجال أمام المطورين ليدلوا بدلوهم في هذه المجموعة ، أتاحت مايكروسوفت البرمجة بلغة فجول بيسك ضمن مجموعة برامج المكتب ، بحيث ضمنها بكل برنامج ، ووفرت الأوامر المتعلقة بإنجاز كل وظيفة لتلائم لغة فجول بيسك ، وهو ما يسمى بالتوافق بين البرامج.

وفي هذه المذكرة ، قمت بتسليط شيء من الضوء على بعض من الأوامر التي يمكن استخدامها في فجول بيسك للتعامل مع الوثيقة (الملف) المنشأ من قبل وورد ، بحيث نتحكم في خصائص الوثيقة وكافة محتوياتها من خلال برنامج في فجول بيسك دون تدخل برنامج وورد ، وكذلك التعامل مع برنامج أكسل بحيث نتحكم في الملف وأوراق عمله وخلاياه ونطاقاته ومخططاته.

وهذه المذكرة موجهة بالأساس إلى المحترفين في البرمجة بلغة فجول بيسك ، فهم وحدهم القادرون على فهم ما أقصده من عبارات وجمل قد تغيب عن المبتدئ والمتوسط.

والله أسأل أن تؤتي المذكرة أكلها ، وأن تنفع بها كل من بحث في دهاليز المنتديات بكلمة (وورد أو أكسل) ، إنه نعم المجيب.

أولاً: برنامج وورد Microsoft Word:

ببساطة هو برنامج معالجة النصوص والرسوم والجداول وغيرها من مكونات الوثائق والرسائل والكتب والصحف والمجلات وغيرها. وللتعامل معه برمجياً من خلال فصول بيسك ، يجب تعريفه للغة أولاً ، وذلك بإدراج مكتبته في اللغة ثم تعريف متغير على أنه يمثل برنامج وورد بكامل خصائصه ووظائفه.

إدراج مكتبات برنامج وورد:

لإدراج مكتبات البرنامج وورد في مشروع في فصول بيسك نتبع الخطوات التالية:

1. نفتح مشروعاً جديداً.
2. نفتح القائمة (Project) ثم نختار الأمر (References).
3. يظهر مربع الحوار الخاص بالمكتبات ، نحدد الخيار (Microsoft Word 10.0 Object Library) بوضع علامة (صح) أمامه.
4. ننقر فوق الزر (Ok).

تعريف متغير على أنه يمثل برنامج وورد:

نكتب الكود التالي في المكان المناسب:

```
كود برمجي  
Dim WordPro As Word.Application
```

وفي هذا الكود ، تم تعريف المتغير Wrd_Pro على أنه برنامج وورد بالكامل ، ويمكننا إجراء ما يمكن إجراؤه في برنامج وورد على هذا المتغير ، فالبرنامج يتكون من مجموعة من الأوامر ، منها ما يتعلق به كبرنامج ، ومنها ما يتعلق بالوثائق والملفات المفتوحة منها والمغلقة ، ومنها ما يتعلق بمكونات الوثيقة من جداول وفقرات وكلمات وحدود وتظليل ورأس الصفحة وتذييلها وغير ذلك.

وتسمى الوثيقة برمجياً بالكائن Document Object ، وكل مكون من مكوناتها يعتبر كائناً بذاته ، وينتمي لتجمع لنفس النوع من الكائنات بمعنى أن مجموعة الوثائق تنتمي لكائن أوسع وأشمل يسمى (تجمع الوثائق Documents Collection) ، والجداول تنتمي لكائن أوسع وهو (تجمع الجداول Tables Collection) وهكذا.

الكائن Application:

ويمثل البرنامج وورد بكامل فعالياته ووظائفه وخصائصه.

بعض أهم خصائص الكائن Application:

1. **ActiveDocument**: وتمثل الوثيقة النشطة المفتوحة حالياً.
2. **ActivePrinter**: وتمثل الطابعة الافتراضية النشطة.
3. **DisplayRecentFiles**: قيمتها True إذا كانت هناك أسماء وثائق مفتوحة مسبقاً في قائمة (ملف File).
4. **Selection**: وتمثل النص المحدد في الوثيقة.

بعض أهم وظائف وطرق الكائن Application:

1. **Quit**: وتستخدم عند إغلاق برنامج وورد.
2. **SendFax**: وتستخدم لبدء تشغيل معالج الفاكس.
3. **ToggleKeyboard**: وتستخدم لتغيير اللغة من عربية إلى لاتينية والعكس.

بعض أهم أحداث الكائن Application:

1. **DocumentBeforeClose**: وينطلق قبل إغلاق الوثيقة تماماً.
2. **DocumentBeforePrint**: وينطلق قبل البدء في طباعة الوثيقة.
3. **DocumentBeforeSave**: وينطلق قبل حفظ التغييرات في الوثيقة.
4. **DocumentChange**: ينطلق عند حدوث تغيرات في محتويات الوثيقة.
5. **DocumentOpen**: ينطلق عند فتح الوثيقة.
6. **NewDocument**: ينطلق عند إنشاء وثيقة جديدة.
7. **Quit**: ينطلق عندما يغلق المستخدم برنامج وورد.
8. **WindowActive**: ينطلق عندما تصير نافذة الوثيقة نشطة.
9. **WindowDeactivate**: وينطلق عندما يغيب التركيز عن نافذة الوثيقة النشطة.

الكائن Document (الوثيقة):

ويمثل وثيقة في برنامج وورد ، وعند فتح وثيقة جديدة أو موجودة مسبقاً يتم إضافتها إلى ما يسمى (تجمع الوثائق Document Collection).

تجمع الوثائق Document Collection:

وهو عبارة عن كائن يضم مجموعة الوثائق المفتوحة حالياً. وتعرف كل وثيقة بدلالة اسمها أو رقم فهرسها.

أهم خصائص تجمع الوثائق:

1. الخاصية **Application**: ويمثل البرنامج Word.
2. الخاصية **Count**: ويمثل عدد الوثائق المفتوحة حالياً.
3. الخاصية **Creator**: وتحدد هل البرنامج الذي أنشأ الوثيقة هو Microsoft Word أم لا.

أهم وظائف تجمع الوثائق:

1. الوظيفة **Add**: وتستخدم في إنشاء وثيقة جديدة وضمها إلى التجمع.
2. الوظيفة **Open**: وتستخدم لفتح وثيقة موجودة مسبقاً وضمها للتجمع.
3. الوظيفة **Save**: وتستخدم لحفظ التغييرات الحاصلة على محتويات الوثيقة.
4. الوظيفة **Close**: وتستخدم لإغلاق وثيقة وإزالتها من التجمع.

أهم الأحداث التي تتعرض لها الوثيقة:

1. الحدث **New**: وينطلق عند إنشاء وثيقة جديدة.
2. الحدث **Open**: وينطلق عند فتح وثيقة موجودة مسبقاً.
3. الحدث **Close**: وينطلق عند إغلاق وثيقة.

وقبل إجراء أي عملية على الوثيقة أو تجمع الوثائق يجب أن نعلم طريقة تعريفهما في المشروع.
انظر للمثال التالي:

```
كود برمجي
Dim Doc As Word.Document
Dim Docs As Word.Documents
```

تم في السطر الأول تعريف المتغير Doc بأنه يمثل وثيقة ، والمتغير Docs بأنه تجمع للوثائق في برنامج وورد.

بعض العمليات التي تجرى على الوثيقة:

1. إنشاء وثيقة جديدة برمجياً:

يضم تجمع الوثائق Documents Collection كافة الوثائق المفتوحة. ولإنشاء وثيقة جديدة (Document Object) ، نستخدم الطريقة Add التابعة لهذا التجمع. والمثال التالي يوضح كيفية إنشاء وثيقة جديدة:

```
كود برمجي
Documents.Add
```

والطريقة الأفضل لإنشاء وثيقة جديدة هي إسناد القيمة الراجعة إلى متغير كائن. فالطريقة Add أرجعت الكائن Document الذي يشير إلى الوثيقة الجديدة.

وفي المثال التالي يتم إسناد الكائن Document الراجع عن طريق استخدام Add إلى متغير كائن. وبعد ذلك يتم ضبط العديد من خصائص ووظائف هذا الكائن.

```
كود برمجي
Sub NewSampleDoc ()
    Dim docNew As Document
    Set docNew = Documents.Add
    With docNew
        .Content.Font.Name = "Tahoma"
        .SaveAs FileName:="Sample.doc"
    End With
End Sub
```

2. فتح وثيقة:

لفتح وثيقة نستخدم الطريقة Open التابعة للتجمع Documents. التعليمات التالية تفتح وثيقة باسم (Sample.doc) موجودة في المجلد (MyFolder).

كود برمجي

```
Sub OpenDocument ()
    Documents.Open FileName:="C:\MyFolder\Sample.doc"
End Sub
```

3. حفظ وثيقة:

لحفظ وثيقة نستخدم الطريقة Save التابعة للتجمع Documents. وفي المثال التالي يتم حفظ الوثيقة باسم (Sales.doc).

كود برمجي

```
Sub SaveDocument ()
    Documents ("Sales.doc").Save
End Sub
```

وبإمكاننا حفظ كافة الوثائق المفتوحة وذلك باستخدام الطريقة Save التابعة للتجمع Documents بالصورة التالية:

كود برمجي

```
Sub SaveAllOpenDocuments ()
    Documents.Save
End Sub
```

4. حفظ نسخة أخرى من الوثيقة:

نستخدم لذلك الطريقة SaveAs التابعة للتجمع Documents. فالمثال التالي يحفظ الوثيقة النشطة باسم (Temp.doc) في المجلد الحالي.

كود برمجي

```
Sub SaveNewDocument ()
    ActiveDocument.SaveAs FileName:="Temp.doc"
End Sub
```

5. إغلاق وثيقة:

نستخدم لذلك الطريقة Close التابعة لنفس التجمع ، والمثال التالي يحفظ ويغلق الوثيقة المسماة (Sales.doc):

كود برمجي

```
Sub CloseDocument ()
    Documents ("Sales.doc").Close SaveChanges:=wdSaveChanges
End Sub
```

ويمكننا إغلاق كافة الوثائق باستخدام نفس الطريقة ، والمثال التالي يغلق الوثائق دون حفظ

التغييرات:

كود برمجي

```
Sub CloseAllDocuments ()
    Documents.Close SaveChanges:=wdDoNotSaveChanges
End Sub
```

والمثال التالي يستشير المستخدم في حفظ التغييرات قبل الإغلاق:

كود برمجي

```
Sub PromptToSaveAndClose ()
    Dim doc As Document
    For Each doc In Documents
        doc.Close SaveChanges:=wdPromptToSaveChanges
    Next
End Sub
```

6. تنشيط وثيقة:

لتنشيط وثيقة ما ، نستخدم الطريقة Activate التابعة للكائن Document. والمثال التالي

ينشط الوثيقة المسماة (Sales.doc):

كود برمجي

```
Sub ActivateDocument ()
    Documents ("Sales.doc").Activate
End Sub
```

7. هل هناك وثيقة مفتوحة؟

لاختبار ما إذا كانت هناك وثيقة مفتوحة ، نقوم بالتجول برمجياً داخل التجمع Document

عن طريق استعمال العبارة For Each... Next. والمثال التالي ينشط الوثيقة المسماة

Sample.doc إذا كانت الوثيقة مفتوحة ، أو يفتحها إن كانت غير مفتوحة حالياً:

كود برمجي

```
Sub ActivateOrOpenDocument ()
    Dim doc As Document
    Dim docFound As Boolean

    For Each doc In Documents
        If InStr(1, doc.Name, "sample.doc", 1) Then
            doc.Activate
            docFound = True
            Exit For
        Else
            docFound = False
        End If
    Next
End Sub
```

```
Next doc

If docFound = False Then
    Documents.Open FileName:="Sample.doc"
End If
End Sub
```

8. الإشارة إلى وثيقة:

عوضاً عن الإشارة إلى وثيقة ما عن طريق اسمها أو رقم فهرسها ، فإننا نستخدم الخاصية ActiveDocument التي ترجع كائناً من النوع Document الذي يشير بدوره إلى الوثيقة النشطة حالياً. المثال التالي يعرض اسم الوثيقة النشطة ، وإن لم تكن هناك وثيقة مفتوحة يعرض رسالة:

كود برمجي

```
If Documents.Count >= 1 Then
    MsgBox ActiveDocument.Name
Else
    MsgBox "No documents are open"
End If
```

الكائن RecentFile:

ويمثل آخر الوثائق التي تم فتحها في برنامج وورد. فكلما تم فتح وثيقة يتم تسجيلها تلقائياً ضمن تجمع الـ RecentFiles.

فلفتح وعرض آخر وثائق تم فتحها وتعديلها نستخدم كائناً يسمى (RecentFile) كما بالمثال

التالي:

كود برمجي

```
Private Sub cmdOpenAllDocs_Click()
Dim rFile As RecentFile
    For Each rFile In RecentFiles
        rFile.Open
    Next rFile
End Sub
```

المثال التالي يحدد عدد الوثائق في التجمع:

كود برمجي

```
RecentFiles.Maximum = 5
```

في حين أن المثال التالي يبطل عمل قائمة آخر الوثائق:

كود برمجي

```
RecentFiles.Maximum = 0
```

والمثال التالي يتأكد من حفظ الوثيقة الحالية ، فإن حفظت يقوم بتسجيلها في قائمة الـRecentFiles:

كود برمجي

```
If ActiveDocument.Saved = True Then  
    RecentFiles.Add Document:=ActiveDocument.FullName, _  
        ReadOnly:=True  
End If
```

والمثال التالي يفتح أول وثيقة في الـRecentFiles:

كود برمجي

```
If RecentFiles.Count >= 1 Then RecentFiles(1).Open
```

والمثال التالي يزيد عدد الوثائق في الـRecentFiles:

كود برمجي

```
num = RecentFiles.Maximum  
If num <> 9 Then RecentFiles.Maximum = num + 1
```

التعامل مع النصوص في الوثيقة:

إدراج نص في وثيقة:

لإدراج نص معين في الوثيقة نستخدم إما الطريقة InsertAfter أو الطريقة

.InsertBefore

والمثال التالي يدرج نصاً في نهاية الوثيقة النشطة:

كود برمجي

```
Sub InsertTextAtEndOfDocument()  
    ActiveDocument.Content.InsertAfter Text:=" The end."  
End Sub
```

والمثال التالي يدرج نصاً قبل التحديد:

كود برمجي

```
Sub AddTextBeforeSelection()  
    Selection.InsertBefore Text:="new text "  
End Sub
```

تحديد نص في وثيقة:

نستخدم الطريقة Select لتحديد عنصر ما في الوثيقة. وهناك العديد من الكائنات التي تدعم هذه الطريقة مثل: الكائن BookMark ، والكائن Field ، والكائن Range ، والكائن Table. والمثال التالي يحدد أول جدول في الوثيقة النشطة:

كود برمجي

```
Sub SelectTable()  
    ActiveDocument.Tables(1).Select  
End Sub
```

والمثال التالي يحدد أول حقل في الوثيقة النشطة:

كود برمجي

```
Sub SelectField()  
    ActiveDocument.Fields(1).Select  
End Sub
```

المثال التالي يحدد أول أربع فقرات في الوثيقة النشطة. تم استخدام الطريقة Range لإنشاء الكائن Range الذي يضم الفقرات الأربع.

كود برمجي

```
Sub SelectRange()  
    Dim rngParagraphs As Range  
    Set rngParagraphs = ActiveDocument.Range(  
        Start:=ActiveDocument.Paragraphs(1).Range.Start, _  
        End:=ActiveDocument.Paragraphs(4).Range.End)  
    rngParagraphs.Select  
End Sub
```

هل هناك نص محدد أم لا؟:

تقوم الخاصية Type التابعة للكائن Selection بإرجاع معلومات عن نوع التحديد. والمثال التالي يعرض رسالة إذا كان التحديد هو نقطة إدراج Insertion Point.

كود برمجي

```
Sub IsTextSelected()  
    If Selection.Type = wdSelectionIP Then  
        MsgBox "Nothing is selected"  
    End if  
End Sub
```

استرجاع نص من وثيقة:

نستخدم الخاصية Text التابعة للكائن Selection والتي تمثل النص المحدد لاسترجاع

النصوص كما بالمثال:

كود برمجي

```
Sub ShowSelection()  
    Dim strText As String  
    strText = Selection.Text  
    MsgBox strText  
End Sub
```

والمثال التالي يقوم باسترجاع الكلمة الأولى في الوثيقة النشطة:

كود برمجي

```
Sub ShowFirstWord()  
    Dim strFirstWord As String  
    strFirstWord = ActiveDocument.Words(1).Text  
    MsgBox strFirstWord  
End Sub
```

التعامل مع الجداول Tables:

يمثل الكائن Table أي جدول في وثيقة وورد ، وبمجرد إنشاء جدول أو أكثر يتم إنشاء

تجمع الجداول Tables Collection يضم كافة الجداول.

إنشاء جدول:

المثال التالي يقوم بإنشاء جدول في وثيقة:

كود برمجي

```
Sub CreateNewTable()  
    Dim docActive As Document  
    Dim tblNew As Table  
    Dim celTable As Cell  
    Dim intCount As Integer  
  
    Set docActive = ActiveDocument  
    Set tblNew = docActive.Tables.Add( _  
        Range:=docActive.Range(Start:=0, End:=0), _  
        NumRows:=3, NumColumns:=4)  
    intCount = 1  
  
    For Each celTable In tblNew.Range.Cells  
        celTable.Range.InsertAfter "Cell " & intCount  
        intCount = intCount + 1  
    Next celTable
```

```
tblNew.AutoFormat Format:=wdTableFormatColorful2, _  
    ApplyBorders:=True, ApplyFont:=True, ApplyColor:=True  
End Sub
```

إدراج نص في خلية بجدول:

المثال التالي يقوم بإدراج نص في أول خلية في أول جدول بالوثيقة النشطة. وتم استعمال الوظيفة Delete لحذف النص الموجود بها ، والوظيفة InsertAfter لإدراج النص "Cell 1,1":

كود برمجي

```
Sub InsertTextInCell()  
    If ActiveDocument.Tables.Count >= 1 Then  
        With ActiveDocument.Tables(1).Cell(Row:=1,  
Column:=1).Range  
            .Delete  
            .InsertAfter Text:="Cell 1,1"  
        End With  
    End If  
End Sub
```

استرجاع نص من خلية بجدول:

المثال التالي يقوم باسترجاع وعرض محتويات كل خلية في الصف الأول من أول جدول بالوثيقة النشطة:

كود برمجي

```
Sub ReturnTableText()  
    Dim tblOne As Table  
    Dim celTable As Cell  
    Dim rngTable As Range  
  
    Set tblOne = ActiveDocument.Tables(1)  
    For Each celTable In tblOne.Rows(1).Cells  
        Set rngTable =  
ActiveDocument.Range(Start:=celTable.Range.Start, _  
            End:=celTable.Range.End - 1)  
        MsgBox rngTable.Text  
    Next celTable  
End Sub  
Sub ReturnCellText()  
    Dim tblOne As Table  
    Dim celTable As Cell  
    Dim rngTable As Range  
  
    Set tblOne = ActiveDocument.Tables(1)  
    For Each celTable In tblOne.Rows(1).Cells  
        Set rngTable = celTable.Range  
        rngTable.MoveEnd Unit:=wdCharacter, Count:=-1  
        MsgBox rngTable.Text  
    Next celTable  
End Sub
```


تحويل نص إلى جدول:

المثال التالي يقوم بإدراج نصوص تفصل بينها علامة الجدولة (Tab) ومن ثم يحولها إلى

جدول في الوثيقة:

كود برمجي

```
Sub ConvertExistingText()  
    With Documents.Add.Content  
        .InsertBefore "one" & vbTab & "two" & vbTab & "three"  
    & vbCr  
        .ConvertToTable Separator:=Chr(9), NumRows:=1,  
NumColumns:=3  
    End With  
End Sub
```

نسخ كافة الجداول بالوثيقة ولصقها في وثيقة جديدة:

المثال التالي يقوم بنسخ جميع الجداول في الوثيقة النشطة وفتح وثيقة جديدة ولصق الجداول

المختارة فيها:

كود برمجي

```
Sub CopyTablesToNewDoc()  
    Dim docOld As Document  
    Dim rngDoc As Range  
    Dim tblDoc As Table  
  
    If ActiveDocument.Tables.Count >= 1 Then  
        Set docOld = ActiveDocument  
        Set rngDoc = Documents.Add.Range(Start:=0, End:=0)  
        For Each tblDoc In docOld.Tables  
            tblDoc.Range.Copy  
            With rngDoc  
                .Paste  
                .Collapse Direction:=wdCollapseEnd  
                .InsertParagraphAfter  
                .Collapse Direction:=wdCollapseEnd  
            End With  
        Next  
    End If  
End Sub
```

الكائن PageSetup:

ونستفيد منه في إعداد صفحات الوثيقة. فالأمثلة التالية تعدل خصائص صفحات الوثيقة النشطة من حيث هوامشها ، وطريقة عرضها (طولياً - عرضياً) وغير ذلك من الخصائص الأخرى:
1. عرض الصفحة طولياً:

كود برمجي

```
ActiveDocument.PageSetup.Orientation = wdOrientPortrait
```

والخاصية OrientPortrait تحتمل القيمتين:

- ❖ wdOrientPortrait وتعني عرض الصفحة طولياً.
- ❖ wdOrientLandscape وتعني عرض الصفحة عرضياً.

2. عرض الصفحة عرضياً:

كود برمجي

```
ActiveDocument.PageSetup.Orientation = wdOrientLandscape
```

3. تعديل هوامش الصفحة:

كود برمجي

```
With ActiveDocument.PageSetup  
    .LeftMargin = InchesToPoints(0.75)  
    .RightMargin = InchesToPoints(0.75)  
    .TopMargin = InchesToPoints(1.5)  
    .BottomMargin = InchesToPoints(1)  
End With
```

4. تغيير شكل وأبعاد الصفحة:

كود برمجي

```
Private Sub cmdChange_Click()  
Documents(1).PageSetup.PaperSize = wdPaperEnvelope10  
End Sub
```

حيث أن الخاصية PaperSize تحتمل إحدى القيم التالية:

- wdPaperA4 - wdPaperA3 - dPaper11x17 - wdPaper10x14
- wdPaperB4 - wdPaperA5 - wdPaperA4Small

أمثلة متنوعة:

1. المثال التالي يوضح كيفية صنع (سمة) لخلفية الوثيقة:

كود برمجي

```
Sub CheckTheme ()
    ActiveDocument.ApplyTheme "artsy 100"
    MsgBox ActiveDocument.ActiveTheme
End Sub
```

2. المثال التالي يعرض عنوان الوثيقة النشطة حالياً:

كود برمجي

```
Sub WindowCaption()
    MsgBox ActiveDocument.ActiveWindow.Caption
End Sub
```

طريقة أخرى:

كود برمجي

```
If Application.Documents.Count >= 1 Then
    MsgBox ActiveDocument.Name
Else
    MsgBox "No documents are open"
End If
```

3. المثال التالي يضبط لون خلفية الوثيقة باللون المطلوب ، شرط أن يكون العرض بتخطيط ويب:

كود برمجي

```
ActiveDocument.ActiveWindow.View.Type = wdWebView
With ActiveDocument.Background.Fill
    .Visible = True
    .ForeColor.RGB = RGB(192, 192, 192)
End With
```

4. المثال التالي يحدد صورة لتكون خلفية للوثيقة (في وضع العرض بتخطيط ويب):

كود برمجي

```
ActiveDocument.ActiveWindow.View.Type = wdWebView
ActiveDocument.Background.Fill.UserPicture _
    PictureFile:="C:\Windows\Bubbles.bmp"
```

5. المثال التالي يضبط نوع وحجم خط محتويات الوثيقة النشطة:

كود برمجي

```
Set myRange = ActiveDocument.Content
With myRange.Font
    .Name = "Arial"
    .Size = 10
End With
```

6. المثال التالي يفتح وثيقة ويضع لها كلمة سر ثم يغلقها:

كود برمجي

```
Set myDoc = Documents.Open(FileName:="C:\computer.doc")
myDoc.Password = "why"
myDoc.Close
```

7. المثال التالي يطبع محتويات وثيقة إلى الطباعة:

كود برمجي

```
Private Sub mnuprint40_Click()
Dim b As Word.Document

Set b = GetObject("c:\acc\new\reports\40.doc")
b.Activate
b.PrintOut FileName = "c:\acc\new\reports\40.doc"
End Sub
```

8. المثال التالي يقوم:

أ. بتلوين النص المحدد باللون الأزرق.

ب. بجعل الخط أسوداً عريضاً.

ج. طباعة الجملة This is some text بعد التحديد مباشرة.

كود برمجي

```
Private Sub Command1_Click()
With Selection
    .Font.Color = wdColorBlue
    .Font.Bold = True
End With
Selection.InsertAfter "This is some text."
End Sub
```

9. المثال التالي يكتب نصاً في بداية الوثيقة ومن ثم يطبعها:

كود برمجي

```
Dim rngTemp As Range

Set rngTemp = ActiveDocument.Range(Start:=0, End:=0)
With rngTemp
    .InsertBefore "Company Report"
    .Font.Name = "Arial"
    .Font.Size = 24
    .InsertParagraphAfter
End With
ActiveDocument.PrintOut
```

10. المثال التالي يعرض اسم الطابعة الافتراضية النشطة:

كود برمجي

```
MsgBox " & ActivePrinter اسم الطابعة النشطة لديك هو:"
```

11. المثال التالي يعين نوعاً معيناً من الطابعات كطابعة نشطة على المنفذ LPT1:

كود برمجي

```
Application.ActivePrinter = "HP LaserJet 4 local on LPT1:"
```

12. المثال التالي يطلب من المستخدم حفظ كافة الوثائق لأن برنامج وورد في طريقه للإغلاق:

كود برمجي

```
Dim intResponse As Integer

intResponse = MsgBox("هل تريد حفظ كافة الوثائق?", vbYesNo)
If intResponse = vbYes Then Application.Quit _
    SaveChanges:=wdSaveChanges, OriginalFormat:=wdWordDocument
```

13. المثال التالي يبدل اللغة في الوثيقة:

كود برمجي

```
Private Sub cmdChangeLanguage_Click()
Dim x As Integer

x = MsgBox("Switch the keyboard language setting?", vbYesNo)
If x = vbYes Then wrddoc.Application.ToggleKeyboard
End Sub
```

ثانياً: برنامج أكسل Microsoft Excel:

ببساطة هو برنامج لمعالجة الجداول الإلكترونية ، ويستخدم بكثرة في إنجاز العمليات الحسابية والإحصائية خصوصاً تلك التي تحتاج إلى تنسيق البيانات في جداول. وللتعامل معه برمجياً من خلال فجول بيسك ، يجب تعريفه للغة أولاً ، وذلك بإدراج مكتبته في اللغة ثم تعريف متغير على أنه يمثل برنامج أكسل بكامل خصائصه ووظائفه.

إدراج مكتبات برنامج أكسل:

- لإدراج مكتبات البرنامج أكسل في مشروع في فجول بيسك نتبع الخطوات التالية:
5. نفتح مشروعاً جديداً.
 6. نفتح القائمة (Project) ثم نختار الأمر (References).
 7. يظهر مربع الحوار الخاص بالمكتبات ، نحدد الخيار (Microsoft Excel 10.0 Object Library) بوضع علامة (صح) أمامه.
 8. ننقر فوق الزر (Ok).

تعريف متغير على أنه يمثل برنامج أكسل:

نكتب الكود التالي في المكان المناسب:

```
كود برمجي  
Dim Excel_Pro As Excel.Application
```

وفي هذا الكود ، تم تعريف المتغير Excel_Pro على أنه برنامج أكسل بالكامل ، ويمكننا إجراء ما يمكن إجراؤه في برنامج أكسل على هذا المتغير ، فالبرنامج يتكون من مجموعة من الأوامر ، منها ما يتعلق به كبرنامج ، ومنها ما يتعلق بالصفحات وأوراق العمل ، ومنها ما يتعلق بمكونات الورقة من خلايا ونطاقات ومخططات ورسومات وغيرها. وكل ما ذكرته يسمى كائناً ، فالملف في أكسل كائن ، والورقة كائن ، والخلية والنطاق والمخططات كلها كائنات.

مصطلحات أساسية:

1. **WorkBook**: ويعني ملف يحتوي على ورقة عمل أو أكثر ، ويخزن في القرص باللاحقة (xls). وسنطلق عليه اسم (وورك بوك) في هذه المذكرة.
2. **WorkSheet**: ورقة العمل ، وهي عبارة عن مجموعة كبيرة من الصفوف والأعمدة ، تسمى نقطة تلاقي الصف بالعمود (بالخلية). وهناك عدة أنواع من أوراق العمل منها: worksheets, charts, modules, and dialog sheets.
3. **Cell**: الخلية ، وهي نقطة التقاء الصف بالعمود ، وتحدد الخلية من خلال رقم الصف والحرف الدال على العمود.
4. **Chart**: المخطط ، وهو رسم بياني يمثل قيم مجموعة من الخلايا.

ملف أكسل Excel WorkBook :

وهو ملف يمثل مجموعة من أوراق العمل Worksheets التي تتكون من عدد كبير جداً من الصفوف Rows والأعمدة Columns.

بعض أهم خصائص WorkBook :

1. **ActiveSheet**: وتمثل الورقة النشطة حالياً.
2. **Password**: ويستفاد منها في وضع كلمة مرور للملف.
3. **Saved**: وتختبر هل الملف محفوظ أم لا؟
4. **Sheets**: وتمثل كافة الأوراق المفتوحة في الـ WorkBook الحالي.

بعض أهم وظائف الـ WorkBook :

1. **Activate**: وتستخدم لتنشيط الملف أو ورقة العمل أو خلية أو نطاق من الخلايا...الخ.
2. **Close**: لإغلاق الـ WorkBook.
3. **Save**: لحفظ الـ WorkBook.
4. **SaveAs**: لحفظ الـ WorkBook باسم آخر.

العمليات على ملفات أكسل وأوراق عملها:

إنشاء Workbook جديد:

لإنشاء ملف أكسل جديد برمجياً نستخدم الطريقة Add التابعة لتجمع الملفات WorkBooks، والمثال التالي يوضح ذلك:

كود برمجي

```
Sub AddOne ()  
    Workbooks.Add  
End Sub
```

طريقة أخرى:

كود برمجي

```
Sub AddNew()  
Set NewBook = Workbooks.Add  
    With NewBook  
        .Title = "All Sales"  
        .Subject = "Sales"  
        .SaveAs Filename:="Allsales.xls"  
    End With  
End Sub
```

وفي هذا المثال ، تم إنشاء كائن جديد ، وضبطت قيم خصائصه ، وخرن باسم (Allsales.xls).

فتح ملف أكسل برمجياً:

نستخدم الوظيفة Open التابعة لتجمع ملفات أكسل بالصورة التالية:

كود برمجي

```
Sub OpenUp()  
    Workbooks.Open("C:\MyFolder\MyBook.xls")  
End Sub
```

إغلاق ملف أكسل:

نستخدم الوظيفة Close لإغلاق الملف بالصورة التالية:

كود برمجي

```
Workbooks("BOOK1.XLS").Close SaveChanges:=False
```

في المثال السابق تم إغلاق الملف المشار إليه دون تخزين التغييرات.

الإشارة إلى ورقة عمل داخل الـ (Workbook) عن طريق رقم الفهرس:
ويتم بالصورة التالية:

```
كود برمجي
Sub FirstOne ()
    Worksheets(1).Activate
End Sub
```

أما إذا أردنا الإشارة إلى أي نوع من أوراق العمل فنكتب:

```
كود برمجي
Sub FourthOne ()
    Sheets(4).Activate
End Sub
```

ملاحظة / يتغير ترتيب رقم الفهرس إذا تم إضافة أو حذف ورقة عمل أو أكثر.

الإشارة إلى ورقة عمل داخل الـ (Workbook) عن طريق اسمها:
ويتم بالصورة التالية:

```
كود برمجي
Worksheets("Sheet1").Activate
Charts("Chart1").Activate
DialogSheets("Dialog1").Activate
```

أو بالصورة التالية:

```
كود برمجي
Sub ActivateChart ()
    Sheets("Chart1").Activate
End Sub
```

حفظ العمل كصفحة ويب:

نستخدم الخاصية SaveAs لحفظ ملف الأكسل بصيغة HTML الخاصة بصفحات ويب كما

بالصورة التالية:

```
كود برمجي
Sub ActivateChart ()
    Sheets("Chart1").Activate
End Sub
```

الخلايا والنطاقات Cells & ranges

1. كيفية الإشارة إلى الخلايا أو النطاقات:

إن الطريقة الشائعة لتحديد خلية أو نطاق خلايا ومن ثم إجراء بعض العمليات عليها كإدراج صيغة رياضية ما أو تنسيقها تنسيقاً معيناً ، هو استعمال عبارة برمجية تحدد النطاق ، وتغير قيمة خاصية معينة أو تنفذ طريقة ما.

يمثل الكائن Range في لغة فجول بيسك خلية واحدة أو مجموعة من الخلايا. والمواضيع التالية ستوضح الطرق الشائعة في التعامل مع النطاقات Range.

أ. الإشارة إلى الخلايا والنطاقات بدلالة العمود A1:

يمكننا الإشارة إلى خلية ما أو مجموعة من الخلايا ضمن العمود A1 عن طريق الخاصية Range. المثال التالي يغمق لون كتابة خلايا النطاق (A1:D5):

```

كود برمجي
Sub FormatRange ()
    Workbooks ("Book1").Sheets ("Sheet1").Range ("A1:D5") _
        .Font.Bold = True
End Sub
    
```

والجدول التالي يوضح بعض الإشارات عن طريق A1 وباستخدام الخاصية Range:

العبارة	المعنى
Range("A1")	الخلية A1
Range("A1:B5")	النطاق من A1 وحتى B5
Range ("C5:D9,G9:H16")	نطاقين مختلفين
Range ("A:A")	العمود A بكامله
Range ("1:1")	الصف 1 بكامله
Range ("A:C")	العمود A حتى العمود C بكاملهما
Range ("1:5")	الصف 1 وحتى 5 بكاملهما
Range ("1:1, 3:3, 8:8")	الصفوف 1 ، 3 ، و 8
Range ("A:A, C:C, F:F")	الأعمدة A ، C ، F

ب. الإشارة إلى الخلايا عن طريق رقم فهرسها:

نستخدم الخاصية Cells للإشارة إلى خلية ما عن طريق رقم فهرسها (عمود : صف). هذه الخاصية ترجع الكائن Range الذي يمثل تلك الخلية. في المثال التالي الصيغة Cells(6, 1) ترجع الخلية A6 وتضبط قيمتها بـ10:

كود برمجي

```
Sub EnterValue()  
    Worksheets("Sheet1").Cells(6, 1).Value = 10  
End Sub
```

تعمل الخاصية Cells بكفاءة عند التجول في نطاق معين للخلايا ، لأننا نستطيع تعيين متغيرات تمثل العدادات (عدادات الصفوف والأعمدة) كما هو موضح بالمثال التالي:

كود برمجي

```
Sub CycleThrough()  
    Dim Counter As Integer  
    For Counter = 1 To 20  
        Worksheets("Sheet1").Cells(Counter, 3).Value = Counter  
    Next Counter  
End Sub
```

ج. الإشارة إلى الصفوف والأعمدة:

نستخدم الخاصية Rows التي تمثل الصفوف و/أو الخاصية Columns التي تمثل الأعمدة للتعامل مع الصفوف أو الأعمدة المحددة. هاتان الخاصيتان ترجعان الكائن Range الذي يمثل النطاق المحدد.

وفي المثال التالي ، ترجع العبارة Rows(1) الصف رقم (1) في الورقة النشطة ، بعد ذلك يتم تغميق لون كتابتها:

كود برمجي

```
Sub RowBold()  
    Worksheets("Sheet1").Rows(1).Font.Bold = True  
End Sub
```

والجدول التالي يبين طرق الإشارة إلى الصفوف والأعمدة برمجياً:

العبارة	المعنى
Rows (1)	الصف الأول
Rows	كافة صفوف الورقة
Columns (1)	العمود الأول
Columns ("A")	العمود الأول
Columns	كافة الأعمدة في الورقة

وللتعامل مع عدة صفوف وأعمدة في نفس الوقت ، نستخدم الطريقة Union التي تضم العديد من الخلايا لإحدى الخاصيتين Rows أو Columns. والمثال التالي يغمق لون كتابة الصفوف 1 ، 3 ، 5 من الورقة النشطة:

```
كود برمجي
Sub SeveralRows ()
  Worksheets ("Sheet1").Activate
  Dim myUnion As Range
  Set myUnion = Union (Rows (1), Rows (3), Rows (5))
  myUnion.Font.Bold = True
End Sub
```

د. الإشارة إلى الخلايا المرتبطة بغيرها من الخلايا الأخرى:

إن الطريقة الشائعة للتعامل مع خلية مرتبطة بغيرها هي استعمال الخاصية Offset. ففي المثال التالي فإن محتويات الخلية التي احداثيها صفراً واحداً لأسفل ، وثلاثة أعمد لأعلى على الورقة النشطة مسطرة بخطين:

```
كود برمجي
Sub Underline ()
  ActiveCell.Offset (1, 3).Font.Underline = xlDouble
End Sub
```

هـ. الإشارة إلى الخلايا عن طريق الكائن Range:

إذا تم إسناد متغير إلى الكائن Range ، فنه يمكننا وبسهولة التعامل مع النطاق باستخدام اسم ذلك المتغير.

فالإجراء التالي ينشئ كائناً متغيراً (myRange) ثم يعطيه النطاق (A1:D5) على الـ Sheet1 في الورقة النشطة ن ويقوم ببعض المهام الأخرى:

كود برمجي

```
Sub Random()  
Dim myRange As Range  
Set myRange = Worksheets("Sheet1").Range("A1:D5")  
myRange.Formula = "=RAND()"  
myRange.Font.Bold = True  
End Sub
```

و. الإشارة إلى كافة الخلايا على الـ **Worksheet**:

عندما نستخدم الخاصية Cells دون أية بارامترات ، فهذا يعني الإشارة إلى كافة الخلايا في الورقة النشطة.

والإجراء التالي يسمح ببيانات كافة الخلايا:

كود برمجي

```
Sub ClearSheet()  
Worksheets("Sheet1").Cells.ClearContents  
End Sub
```

ز. الإشارة إلى نطاقات متعددة:

نستخدم الطريقتين Range و Union للإشارة إلى مجموعة من النطاقات ، ثم نستخدم الخاصية Areas للإشارة إلى مجموعة النطاقات المختارة في الورقة النشطة.

* استخدام الخاصية **Range**:

نستطيع الإشارة إلى عدة نطاقات باستخدام الخاصية Range وذلك بوضع فواصل بين كل إشارتين أو أكثر. والمثال التالي يسمح محتويات ثلاث نطاقات على الـ Sheet1:

كود برمجي

```
Sub ClearRanges()  
Worksheets("Sheet1").Range("C5:D9,G9:H16,B14:D18"). _  
ClearContents  
End Sub
```

* استعمال الطريقة **Union**:

نستطيع ربط مجموعة من النطاقات بكائن Range واحد باستخدام الطريقة Union. والمثال التالي ينشئ الكائن Range ويسميه myMultipleRange ، ويعرفه على أنه النطاق (A1:B2) و (C3:D4) في نفس الوقت ويغلق محتوياتهما:

كود برمجي

```
Sub MultipleRange()  
    Dim r1, r2, myMultipleRange As Range  
    Set r1 = Sheets("Sheet1").Range("A1:B2")  
    Set r2 = Sheets("Sheet1").Range("C3:D4")  
    Set myMultipleRange = Union(r1, r2)  
    myMultipleRange.Font.Bold = True  
End Sub
```

* استعمال الخاصية Areas:

يمكننا استعمال الخاصية Areas للإشارة إلى النطاق المختار أو تجمع النطاقات في (اختيار متعدد المناطق). الإجراء الحالي يعد المناطق المختارة ، فإن كانت أكثر من منطقة واحدة تظهر رسالة تحذير:

كود برمجي

```
Sub FindMultiple()  
    If Selection.Areas.Count > 1 Then  
        MsgBox "Cannot do this to a multiple selection."  
    End If  
End Sub
```

2. التجول داخل الخلايا والنطاقات برمجياً:

عند استعمال لغة فجول بيسك ، فإننا نرغب دائماً في تنفيذ نفس الكود على خلية أو نطاق خلايا. ولعمل ذلك ، فإننا نقوم بالتجول Looping داخل هذا النطاق بحيث ننفذ كافة العمليات المطلوبة على خلايا النطاق المحدد الواحدة تلو الأخرى. وهناك طريقة للتجول داخل نطاق خلايا معين باستعمال العبارة For... Next ، بحيث نستفيد من الخاصية cells.

وباستعمال الخاصية Cells ، نتجول بدلالة موقع الخلية ، أو بدلالة متغيرات نستخدمها لهذا الشأن. وفي المثال التالي ، يتم تعريف متغير كعداد ، يمثل أرقام الصفوف ، ومن ثم يتجول في النطاق (C1:C20) ، ويضبط قسمة الخلية بـ 0 إذا كانت قيمتها المطلقة أقل من 0.01:

كود برمجي

```
Sub RoundToZero1()  
    For Counter = 1 To 20  
        Set curCell = Worksheets("Sheet1").Cells(Counter, 3)  
        If Abs(curCell.Value) < 0.01 Then curCell.Value = 0  
    Next Counter  
End Sub
```

وهناك طريقة أخرى للتجول داخل النطاق باستخدام العبارة For Each... Next بحيث تتعامل مع تجمع الخلايا المحددة في الخاصية range. يقوم فجول بيسك تلقائياً بإسناد متغير كائن يمثل الخلية التالية كلما نُفذ التجول (التكرار).

والإجراء التالي يتجول في النطاق (A1:D10) ويضع 0 في كل خلية تقل قيمتها عن 0.01:

كود برمجي

```
Sub RoundToZero2 ()
  For Each c In Worksheets("Sheet1").Range("A1:D10").Cells
    If Abs(c.Value) < 0.01 Then c.Value = 0
  Next
End Sub
```

ملاحظة مهمة:

إذا لم تعرف حدود النطاق الذي تريد التجول في خلاياه ، يمكنك استخدام الخاصية CurrentRegion التي ترجع لك النطاق الذي توجد به الخلية النشطة. فالإجراء التالي على سبيل المثال ، يتجول في النطاق الذي توجد به الخلية النشطة ، ويقوم بضبط قيم الخلايا التي قيمها أقل من 0.01 بقيمة الصفر:

كود برمجي

```
Sub RoundToZero3 ()
  For Each c In ActiveCell.CurrentRegion.Cells
    If Abs(c.Value) < 0.01 Then c.Value = 0
  Next
End Sub
```

3. اختيار ونشيط الخلايا:

عندما نعمل في برنامج أكسل ، نقوم عادة باختيار خلية أو أكثر ثم نجري عليها ما نريد من عمليات. أما في حالة التعامل مع الخلايا عن طريق فجول بيسك فلا يجب تحديد الخلية قبل إجراء أي عملية عليها.

فعلى سبيل المثال ، إذا أردنا إدراج صيغة ما على الخلية D6 فلا يجب أن نحدد تلك الخلية. بل نحتاج فقط إلى إرجاع كائن Range يمثل الخلية المطلوبة ثم نعدل قيمة الخاصية Formula بحيث تحمل الصيغة المطلوب تنفيذها على محتوى تلك الخلية:

كود برمجي

```
Sub EnterFormula ()
  Worksheets("Sheet1").Range("D6").Formula = "=SUM(D2:D5) "
End Sub
```


تحديد الخلايا على الورقة النشطة:

عند استخدام الطريقة Select لتحديد الخلايا ، يجب التنبه إلى أن التحديد يسري فقط على الورقة النشطة. فإن نفذنا الكود من الوحدة النمطية Module فستفشل الطريقة Select إلا إذا نشطنا الورقة برمجياً قبل استخدام الطريقة Select على نطاق الخلايا. فعلى سبيل المثال ، فإن الإجراء التالي ينسخ صفاً من الـ Sheet1 إلى الـ Sheet2 في ملف أكسل النشط:

كود برمجي

```
Sub CopyRow()  
Worksheets("Sheet1").Rows(1).Copy  
Worksheets("Sheet2").Select  
Worksheets("Sheet2").Rows(1).Select  
Worksheets("Sheet2").Paste  
End Sub
```

تنشيط خلية ضمن نطاق من الخلايا المحددة:

نستخدم الخاصية Activate لتنشيط خلية موجودة ضمن نطاق محدد من الخلايا. فهناك خلية واحدة فقط يمكن تنشيطها حتى وإن تم اختيار نطاق كامل من الخلايا. والإجراء التالي يحدد نطاقاً من الخلايا ثم ينشط خلية موجودة ضمنه دون تغيير التحديد:

كود برمجي

```
Sub MakeActive()  
Worksheets("Sheet1").Activate  
Range("A1:D4").Select  
Range("B2").Activate  
End Sub
```

4. التعامل مع نطاقات ثلاثية الأبعاد:

إذا كنا نعمل مع نفس النطاق في أكثر من ورقة واحدة نستخدم الدالة Array لتحديد ورقتين أو أكثر. ادرس المثال التالي حيث يحدد نطاق ثلاثي من الخلايا:

كود برمجي

```
Sub FormatSheets()  
Sheets(Array("Sheet2", "Sheet3", "Sheet5")).Select  
Range("A1:H1").Select  
Selection.Borders(xlBottom).LineStyle = xlDouble  
End Sub
```

5. التعامل مع الخلية النشطة:

تقوم الخاصية `ActiveCell` بإرجاع كائن `Range` يحدد الخلية النشطة في الورقة النشطة. وبالتالي يمكن تعديل خصائص هذا الكائن الذي يمثل الخلية المطلوبة كما هو مبين بالمثال التالي:

كود برمجي

```
Sub SetValue()  
    Worksheets("Sheet1").Activate  
    ActiveCell.Value = 35  
End Sub
```

تغيير الخلية النشطة:

نستطيع باستخدام الطريقة `Activate` أن نغير الخلية النشطة. ادرس المثال التالي:

كود برمجي

```
Sub SetActive()  
    Worksheets("Sheet1").Activate  
    Worksheets("Sheet1").Range("B5").Activate  
    ActiveCell.Font.Bold = True  
End Sub
```

استخدام بعض الدوال التابعة لـ Sheets في جدول بيسك:

يمكننا استعمال معظم الدوال التي يمكن استعمالها في أوراق عمل أكسل بسهولة في فـجول بيسك ، وفيما يلي قائمة بأهم تلك الدوال:

وظيفة	الدالة
إرجاع قوس جيب التمام، أو معكوس جيب التمام لزاوية. قوس جيب التمام هو زاوية جيب تمامها عبارة عن رقم. وتكون قيمة الزاوية التي يتم إرجاعها بالتقدير الدائري في النطاق 0 (صفر) إلى pi.	ACOS
إرجاع معكوس جيب التمام لقطع زائد لرقم. يجب أن يكون الرقم أكبر من أو يساوي 1.	ACOSH
إرجاع TRUE إذا كانت كافة وسائطها TRUE ؛ وإرجاع FALSE إذا كانت هناك وسيطة واحدة أو أكثر FALSE.	AND
إرجاع قوس الجيب، أو معكوس الجيب لزاوية. قوس الجيب هو الزاوية التي يكون جيب زاويتها عبارة عن رقم. ويتم إرجاع الزاوية بالتقويم الدائري في النطاق من $-\pi/2$ إلى $\pi/2$.	ASIN
إرجاع معكوس الجيب لقطع زائد لرقم.	ASINH
إرجاع قوس الظل أو معكوس الظل للإحداثيين س و ص المحددين. قوس الظل هو الزاوية من المحور س لخط يحتوي على نقطة الأصل (0, 0) ونقطة ذات الإحداثيات (س و ص). يتم تقديم الزاوية بالتقدير الدائري بين $-\pi$ و π ، ولا تشمل $-\pi$.	ATAN2
إرجاع معكوس ظل الزاوية لقطع زائد. يجب أن يكون رقم الزاوية بين -1 و 1 (ليس فيه -1 و 1).	ATANH
إرجاع متوسط الانحرافات لنقاط البيانات عن وسطها. AVEDEV عبارة عن مقياس التباين في مجموعة البيانات.	AVEDEV
إرجاع متوسط الوسائط (الوسط الحسابي).	AVERAGE
إرجاع دالة كثافة احتمالات بيتا التراكمية. يتم استخدام دالة كثافة احتمالات بيتا التراكمية لدراسة التباين النسب المئوية لشيء ما في مجموعة العينات، مثل جزء اليوم الذي يقضيه الأشخاص في مشاهدة التلفزيون.	BETADIST
إرجاع معكوس دالة كثافة احتمالات بيتا التراكمية.	BETAINV
إرجاع الحدود الجبرية الفردية لاحتمالية التوزيع ذات الحدين.	BINOMDIST

يتبع / استخدام بعض الدوال التابعة لـ Sheets في فجول بيسك:

الدالة	وظيفتها
CHIINV	إرجاع معكوس الاحتمال وحيد الطرف لتوزيع كاي تربيع.
CHITEST	إرجاع اختبار الاستقلالية.
CHOOSE	لتحديد واحدة من القيم التي يصل عددها 29 كحد أقصى استناداً إلى رقم الفهرس. فعلي سبيل المثال، إذا كانت تمثل القيمة 1 إلى القيمة 7 أيام الأسبوع، فترجع CHOOSE أحد الأيام عند استخدام رقم بين 1 و 7 كرقم فهرس (index_num).
CLEAN	إزالة كافة الأحرف غير القابلة للطباعة من النص. استخدم CLEAN على النص الذي تم استيراده من تطبيقات أخرى تحتوي على أحرف التي قد لا يتم طباعتها على نظام التشغيل لديك. فعلي سبيل المثال، يمكنك استخدام CLEAN لإزالة بعض تعليمات الكمبيوتر البرمجية ذات المستوى المنخفض التي تتكرر باستمرار في بداية ونهاية ملفات البيانات ولا يمكن طباعتها.
CONFIDENCE	إرجاع فترة الثقة لوسط مجموعة بيانات.
COUNT	حساب عدد الخلايا التي تحتوي على أرقام وأيضاً أرقام داخل قائمة الوسائط، استخدم COUNT للحصول على عدد الإدخالات في حقل أرقام بنطاق أو مصفوفة الأرقام.
COUNTIF	حساب عدد الخلايا داخل نطاق يفي بالمعايير المعطاة.
DAYS360	إرجاع عدد الأيام بين تاريخين استناداً إلى سنة تتكون من 360 يوماً (اثنا عشر شهراً يتكون كل منها من 30 يوم)، الذي يتم استخدامه في بعض العمليات المحاسبية. استخدم هذه الدالة في حساب دفعات السداد إذا كان نظام المحاسبة الذي تستخدمه مستنداً إلى اثني عشر شهراً يتكون كل منها من 30 يوم.
DEGREES	تحويل التقويم الدائري إلى درجات.
DMAX	إرجاع أكبر رقم في أحد أعمدة قائمة أو قاعدة بيانات تطابق الشروط التي تحددها.
DMIN	إرجاع أصغر رقم في أحد أعمدة قائمة أو قاعدة بيانات تطابق الشروط التي تضعها.
DSUM	لجمع الأرقام التي تطابق الشروط التي تحددها في عمود ضمن قائمة أو قاعدة بيانات .

يتبع / استخدام بعض الدوال التابعة لـ Sheets في فجول بيسك:

وظائفها	الدالة
إرجاع رقم بعد تقريبه لأقرب رقم صحيح زوجي.	EVEN
إرجاع مضروب أحد الأرقام. مضروب الرقم يساوي 1*2*3*...*رقم.	FACT
FIND تقوم بالبحث عن سلسلة نصية (find_text) ضمن سلسلة نصية أخرى (within_text)، وإرجاع رقم موضع بداية لسلسلة المراد البحث عنها (find_text)، من الحرف الأول للسلسلة المراد البحث فيها (within_text). يمكنك أيضاً استخدام SEARCH للعثور على سلسلة نصية واحدة داخل سلسلة أخرى، ولكن بخلاف SEARCH، فإن FIND تكون حساسة للأحرف ولا تسمح بأحرف البديل.	FIND
لحساب أو التنبؤ بقيمة مستقبلية باستخدام قيم موجودة. تكون القيمة المتوقعة عبارة عن قيمة حل لقيمة س المعطاة. القيم المعطاة هي قيم س وقيم ص الموجودة، وقيم التنبؤ بالقيمة الجديدة باستخدام الانحدار الخطي. يمكنك استخدام هذه الدالة للتنبؤ بالمبيعات، ومتطلبات المخزون، واتجاهات السوق المستقبلية.	FORECAST
إرجاع الوسط التوافقي لمجموعة بيانات. الوسط التوافقي هو معكوس الوسط الحسابي لمقلوب الأرقام.	HARMEAN
تشير إلى أي عنصر ليس نصاً. (لاحظ أن تلك الدالة تقوم بإرجاع TRUE إذا كانت value تشير إلى خلية فارغة.	ISNONTEXT
تشير إلى رقم.	ISNUMBER
تشير إلى أي قيمة منطقية.	ISLOGICAL
تشير إلى نص.	ISTEXT
إرجاع اللوغاريتم الطبيعي لرقم. تستند اللوغاريتمات الطبيعية إلى ثابت e (2.718281845904).	LN
إرجاع لوغاريتم رقم بالأساس الذي تحدده.	LOG
إرجاع أكبر قيمة في مجموعة قيم.	MAX
إرجاع أصغر رقم في مجموعة من القيم.	MIN
إرجاع رقم يتم تقريبه إلى أقرب أعلى عدد صحيح فردي.	ODD
إرجاع TRUE إذا كانت أي من الوسائط تساوي TRUE؛ وإرجاع FALSE إذا كانت كافة الوسائط تساوي FALSE.	OR

يتبع / استخدام بعض الدوال التابعة لـ Sheets في فجول بيسك:

وظائفها	الدالة
إرجاع الرقم 3.14159265358979، الثابت الرياضي pi، بدقة تصل إلى 15 رقماً.	PI
إرجاع النتيجة لرقم مرفوع إلى أس.	POWER
لتحويل الدرجات إلى تقدير دائري .	RADIANS
تقوم REPLACE باستبدال جزء من السلسلة النصية، بالاستناد إلى عدد الأحرف التي تحدها، بسلسلة نصية أخرى.	REPLACE
لتقريب رقم بعدد محدد من الخانات.	ROUND
تُرجع SEARCH عدد الأحرف التي عثر عندها على حرف معين أو سلسلة نصية، بداية بـ start_num (رقم البدء). استخدم SEARCH لتحديد موقع الحرف أو السلسلة النصية داخل سلاسل نصية أخرى حيث يمكنك استخدام الدالات MID أو REPLACE لتغيير النص.	SEARCH
إرجاع ترتيب أصغر قيمة k في مجموعة بيانات. استخدم هذه الدالة لإرجاع القيم حسب ثابت نسبي محدد في مجموعة بيانات.	SMALL
جمع كافة الأرقام الموجودة في نطاق من الخلايا.	SUM
جمع الخلايا المحددة حسب معايير معطاة.	SUMIF
تحويل قيمة إلى نص في تنسيق رقمي محدد.	TEXT

الختاتمة

بعد هذا الجهد المضني ، والعمل الدؤوب ، وضعنا أقدامنا على أول خطوة في طريق تطوير برامج تستفيد من خاصية التكامل التي أوجدتها مايكروسوفت بين برامجها ، بحيث يمكننا الاستفادة من إمكانيات أي برنامج من تصميم مايكروسوفت في فجول بيسك.

فقد سلطت الضوء على كيفية تعريف البرنامجين (وورد وأكسل) في لغة فجول بيسك ، وكيف يمكننا استعمال أهم خصائصهما ووظائفهما وأحداثهما في التعامل مع الوثائق وأوراق العمل.

وبذا ، أرجو أن أكون قد وفقت في إيصال الرسالة التي مفادها: لا صعب مع اجتهاد!!.

السلام عليكم

أبو بكر شرف الدين سويدان