VBA में लूपिंग स्टेटमेंट्स (Looping statements in VBA)

उद्देश्य : इस पाठ के अन्त में आप निम्नलिखित कार्य करने योग्य होंगे :

- **VBA में "for" लूप का वर्णन करना**
- **VBA में "do" लूप का वर्णन करना**
- **VBA लूप्स में "exit" कथन के उपयोग की व्याख्या करना**
- **दोहराये जाने वाले कार्यों को करने के लिए उपयुक्त कोड लिखना।**

परिचय (Introduction)

ऐसी कई स्थितियाँ हो सकती हैं जहाँ आपको बार-बार कार्य करने की आवश्यकता होती है / कुछ बार। ऐसे मामलों में कार्य के लिए कोड लूप के अंदर रखा जाता है और प्रोग्राम लूप के माध्यम से एक निश्चित संख्या को पूरा करता है या फिर एक निश्चित स्थिति पूरी होने तक दोहराया जाता है। इस तरह के दोहराव वाले कार्यों के कुछ उदाहरण हैं :

a एक पाठ या संख्या 'n' संख्या को कई बार प्रिंट करना।

b अनुक्रम या संख्याओं की श्रृंखला उत्पन्न करना।

c कुछ गणनाओं की एक टेबल उत्पन्न करना।

d संख्याओं का एक सेट खोज / पुनर्व्यवस्थित करना आदि।

VBA लूपिंग आवश्यकताओं को संभालने के लिए निम्न प्रकार के लूप प्रदान करता है। (टेबल 1 देखें)

टेबल 1

लूप के प्रकार	विवरण
for.....next लूप	कई बार बयान के अनुक्रम का निष्पादन करता है और लूप का प्रबंधन करने वाले कोड को संक्षेप में करता है।
do....until लूप	एक शर्त पूरी होने तक बयान या कथन के समूह को दोहराता है।
do….while लूप	जब तक शर्त सही होती है तब तक एक बयान या कथन के समूह को दोराताहै।

फॉर लूप (The For Loop)

'फॉर-नेक्स्ट' लूप वेरिएबल को मानों के निर्दिष्ट सेट पर सेट करता है और प्रत्येक मान के लिए, लूप के अंदर VBA कोड चलाता है। उदाहरण के लिए

```
For n = 1 To 10
debug.print  n
Next n
```

इस उदाहरण में, 'n' का प्रारंभिक मान 1 पर सेट किया गया है, और लूप कोड। यानी, 'n' मूल्य को प्रिंट किया जाता है। 'N' का मान अगले मान पर सेट किया गया है जो डिफ़ॉल्ट रूप से 1 की वृद्धि है। इस प्रकार यह लूप10 बार निष्पादित किया जाता है और संख्या 1 से 10 को प्रिंट करेगा। उपरोक्त कोड में 'For' कथन जैसे ही है "For n = 1 To 10 Step 1" क्योंकि डिफ़ॉल्ट वृद्धि 1 है।

यह कोड नीचे दिखाए गए अनुसार ऋणात्मक मान में बदल दिया गया है तो वही कोड संख्या 10 से 1 तक प्रिंट करेगा।

```
For n = 10 To 1 Step -1
debug. print  n
Next n
```

इस प्रकार, निम्नलिखित उदाहरण सभी संख्याओं को 1 से 10 तक जोड़ देगा और योग मुद्रित करेगा।

```
Dim n, sum as integer
Sum=0
For n = 1 To 10
sum=sum + n
debug. print  sum
Next n
```

For Each लूप (The For Each Loop)

For Each लूप फोर-नेक्स्ट लूप के समान है लेकिन एक चर के लिए मानों के सेट के माध्यम से लूप करने की बजाय, यह ऑब्जेक्ट्स के सेट के भीतर प्रत्येक ऑब्जेक्ट के माध्यम से लूप करता है। निम्नलिखित उदाहरण सभी वर्कशीट के नाम प्रिंट करेगा।

```
Dim ws As Worksheet
For each ws in Worksheets
debug. print ws.name
Next ws
```

Exit For स्टेटमेंट (The Exit For Statement)

यदि आपको अंतिम स्थिति तक पहुँचने या मिलने से पहले **For** लूप को समाप्त करने की आवश्यकता है, तो IF कथन के साथ संयोजन में END FOR का उपयोग करें। नीचे दिए गए उदाहरण में, हम लूप समयपूर्वता से बाहर निकलें और अंत स्थिति से पहले **For** का उदाहरण नीचे दिया हुआ है। लूप तब तक चलेगा जब तक वह 5 तक नहीं पहुँच जाता।

```
For n = 0 To 10
debug.print n
If n=5 Then Exit For
Next n
```

Do ….Until लूप एक कथन या बयान के समूह हो तब तक दोहराता है जब तक कोई शर्त पूरी नहीं हो जाती।

Excel VBA मैक्रो कोड में Do Until का उपयोग करने के 2 तरीके हैं।

a लूप में कोड निष्पादित करने से पहले स्थिति का परीक्षण करें।

b लूप में कोड निष्पादित करने से पहले स्थिति का परीक्षण करें।

Do Until….. लूप (Do Until….. Loop)

इस उदाहरण में, लूप में जाने से पहले n का मान परीक्षण किया जाता है।

यदि स्थिति n=10 शुरूआत में ही सही नहीं है। लूप के अंदर कोड बिल्कुल निष्पादित नहीं किया गया है। नियंत्रण तब लूप स्टेटमेंट के बाद दिखाई देने वाले बयान पर जम्प करता है।

Do Until n=10

Debug.print n

n=n+1

Loop

ा (Do ….. Loop Until)

इस उदाहरण में, लूप में कोड कम से कम एक बार स्थिति का परीक्षण करने से पहले निष्पादित किया जाता है। अगर स्थिति सच है। लूपिंग बंद हो जाती है, अन्यथा लूप फिर से निष्पादित किया जाता है।

Do

Debug. print n

n=n+1

Loop Until n=10

Do While … Loop एक कथन या कथन के समूह को तब तक दोहराता है जब तक कि स्थिति सत्य न हो।

दो लूप लूप की तरह, दो तरीकों से एक Do until लूप का भी उपयोग किया जा सकता है।

a लूप में कोड को रद्द करने से पहले स्थिति का परीक्षण करें।

b लूप में कंडीशन का निष्पादन करता है फिर लूप का कोड कन करता है।

(Do While ….Loop)

इस उदाहरण में, कंडीशन num<10 लूप में इंटर होने से पहले जांची जाती है। जब कंडीशन सच होती हैं तभी लूप के अंदर का कोड निष्पादित होता है अन्यथा पूरी तरह स्किप हो जाता है। यह उदाहरण टेबल प्रिंट करेगा जैसा Fig 1 में दिखाया गया है।

Fig 1

```
number spc(3)
   1          1
   2          4
   3          9
   4          16
   5          25
   6          36
   7          49
   8          64
   9          81
   10          100
```

Dim num As Integer

Debug.Print "number"; Spc(2); "square"

Do While num < 10

num = num + 1

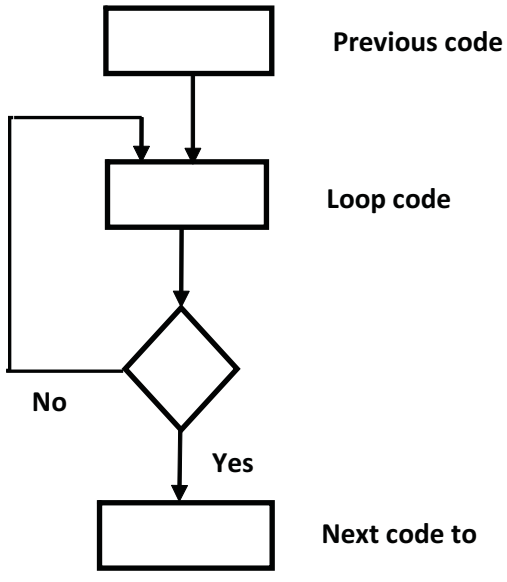
Debug.Print num; Spc(5); num * num

Loop

(Do…. Loop While)

Do…. Loop While , लूप में स्टेटमेंट्स का समूह एक बार ही निष्पादित होता है, उसके बाद कंडीशन चेक की जाती हैं। लूप के अंदर का कोड तभी निष्पादित होता है जब कंडीशन मिलती है। (फ्लो चार्ट के लिए Fig 2 देखें)

Fig 2



इस उदाहरण में, मान 1 सेल (1,1) में रखा जायेगा। रो के मान में हर बार वृद्धि होती है जब हर बार लूप निष्पादित होता है। कन्क्रीमेंट किया हुआ मान सेल (row,1) में आता है। लूप तब तक निष्पादित होता है जब तक रो का मान 10 से नीचे होता है उसके बाद लूप बंद हो जाता है। कम से कम एक बार लूप को चलाने के बाद लूप का निष्पादन बंद हो जाता है। (Fig 2)

```
Dim row As Integer

row = 0

Do

row = row + 1

Cells(row, 1) = row

Loop While row < 10
```

The While ….. Wend loop (The While ….. Wend loop)

While ….. Wend लूप जब तक की कंडीशन सही है तब तक लूप के अंदर की स्टेटमेंट को निष्पादित करता है।

इस उदाहरण में कंडीशन लूप की शुरूआत में ही जाँच ली जाएगी। यह कोड 5 बार hello को प्रिंट करेगा, उसके बाद काउंटर का मान प्रिंट करेगा यानि प्रोग्राम के अंत में 5 प्रिंट करेगा।

```
Dim Counter

Counter = 0

While Counter < 5

Counter = Counter + 1

Debug.Print "hello"

Wend

Debug.Print Counter
```

एक्जीट स्टेटमेन्ट (The Exit Statement)

एग्जिट स्टेटमेंट प्रोसीजर या ब्लाक से बाहर आकर प्रोसीजर कॉल या ब्लाक परिभाषा के बाद की स्टेटमेंट पर तुरंत कण्ट्रोल को ट्रान्सफर करता है। यह Exit Do, Exit For, Exit While, Exit Select आदि के रूप में हो सकता है पर यह इस पर आधारित है की इसे कहाँ उपयोग किया जा रहा है। Exit स्टेटमेंट का उदाहरण निम्न है :

```
Do While True

Count = Count + 1

Debug.Print Count

If Count = 5 Then

Debug.Print "stop at 5"

Exit Do

End If

Loop
```

इस उदाहरण में लूप की कंडीशन count=5 होने पर लूप को रोक देगी।

VBA में ऐरे (Arrays in VBA)

उद्देश्य : इस पाठ के अन्त में आप निम्नलिखित कार्य करने योग्य होंगे :

- **VBA में एक ऐरे का वर्णन और घोषणा करना**
- **स्टेटिक और गतिशील ऐरे के बीच अंतर करना**
- **एक बहुआयामी ऐरे घोषित, पॉप्युलेट और रीड करना**
- **VBA में redim और preserve बयानों का वर्णन करना।**

परिचय (Introduction)

एक ऐरे एक ही डेटा प्रकार और उसी नाम के वैरिएबल का एक समूह है। यदि हमारे पास ऐसी वस्तुओं की एक सूची है जो समान प्रकार के हैं, तो हमें प्रत्येक आइटम के लिए चर का उपयोग करने के बजाय चर की एक ऐरे घोषित करने की आवश्यकता है, उदाहरण के लिए यदि हमें प्रत्येक नाम के लिए दस अलग-अलग चर घोषित करने के बजाय दस नाम दर्ज करने की आवश्यकता है, तो हमें सभी नामों को रखने वाले केवल एक ऐरे घोषित करने की आवश्कता है। ऐरे में व्यक्तिगत एलिमेंट या आइटम इसकी इन्डेक्स या सबस्क्रिप्ट द्वारा पहचाना जाता है।

जब ऐरे का उपयोग किया जाता है, डेटा व्यवस्थित तरीके से संग्रहित किया जाता है। डेटा के साथ काम करने के अलावा, आसान और तेज़ हाता है जब लूप स्टेटमेंट्स जैसे फॉर-नेक्स्ट इत्यादि का उपयोग करके पुनरावृत्तियों को दिया जाता है। निम्नलिखित उदाहरण स्कूल में दस छात्रों के रखने के लिए एक ऐरे चर घोषित करता है।

Dim students(10) As Integer

पिछले उदाहरण में ऐरे "छात्रों" में दस एलिमेंट होते हैं। एलिमेंट के सूचकांक डिफ़ॉल्ट रूप से 0 से 9 तक होते हैं। ऐरे में चर, अब छात्रों (0), छात्रों (1) आदि के रूप में पहचाने जाते हैं, जो पहले एलिमेंट और दूसरे एलिमेंट को इंगित करते हैं।

ऐरे के प्रकार (Types of Arrays)

1 स्टेटिक ऐरे Static Arrays

2 गतिशील ऐरे Dynamic Arrays

स्टेटिक ऐरे (Static array)

एक स्टेटिक ऐरे एक ऐरे है जिसे मंद कथन में आकार दिया गया है जो ऐरे घोषित करता है। उदाहरण के लिए

Dim Students(10) as String

Dim StaticArray(1 To 10) As Long

आप स्टेटिक ऐरे के डेटा प्रकार के आकार को नहीं बदल सकते हें। जब आप स्टेटिक ऐरे मिटाते है, तो कोई स्मृति मुक्त नहीं होती है। इरेस सभी एलिमेंट को ऐरे के डेटा प्रकार के आधार पर इपने डिफ़ॉल्ट मान (0, vbNullString, खाली या कुछ भी नहीं) पर सेट करता है।

एक ऐरे घोषित करना (Declaring an Array)

आप Dim कथन का उपयोग कर एक ऐरे चर घोषित करते हैं।

Dim StudentName(3) As String

ऐरे को किसी अन्य विधि में भी घोषित किया जाता है जहाँ टाइप या वैरिएबल नाम कोष्ठक के एक या अधिक जोड़े के साथ जोड़ा जाता है । यह इंगित करने के लिए कि यह एक ऐरे रखेगा। ऐरे घोषित करने के बाद, आप रीडीम स्टेटमेंट का उपयोग करके अपना आकार परिभाषित कर सकते हैं।

निम्न उदाहरण एक आयामी ऐरे चर घोषित करता है और रेडीम स्टेटमेंट का उपयोग करके ऐरे के आयामों को भी निर्दिष्ट करता है।

Dim arr As Integer()

ReDim arr(10)

निम्नलिखित उदाहरण, आयामों को अलग करने के लिए कोष्टक के अंदर कोमा लगाकर एक बहुआयामी ऐरे चर घोषित करता है।

Dim arrayName (num1, num2) as datatype

एक जंजीर ऐरे चर घोषित करने के लिए। नोस्टेड ऐरे के प्रत्येक स्तर के लिए परिवर्तनीय नाम के बाद कोष्टक की एक जोड़ी जोड़े।

Dim arr()()() As integer

VBA ऐरे में आप ऐरे के निचले और ऊपरी सीमाओं के लिए कोई मान निर्दिष्ट कर सकते हैं। एलिमेंट 0 को ऐरे में पहला एलिमेंट नहीं होना चाहिए। उदाहरण के लिए, निम्नलिखित पूरी तरह से कानूनी कोड है (जब तक निचली बाउंड ऊपरी बाउंड से कम या बराबर होती है - यदि निचली बाउंड ऊपरी सीमा से अधिक हो तो त्रुटि उत्पन्न होती है)।

यदि आप स्पष्ट रूप से किसी ऐरे के निचले बाउंड की घोषणा नहीं करते हैं। उपस्थित होने पर विकल्प बेस स्टेटमेंट के मूल्य के आधार पर निचली बाउंड या तो 0 या 1 मानी जाएगी। यदि मॉड्यूल में विक्लप बेस मौजूद नहीं है, तो 0 माना जाता है। उदाहरण के लिए, कोड

Dim Arr(10) As Long

कोड या तो 10 या 11 एलिमेंट की एक ऐरे घोषित करता है। घोषणा ऐरे में एलिमेंट की संख्या निर्दिष्ट नहीं करता है। इसके बजाय, यह ऐरे के ऊपरी बाउंड को निर्दिष्ट करता है। यदि आपके मॉड्यूल में "ऑप्शन बेस" कथन नहीं है, तो निचली बाउंड शून्य मानी जाती है, और उपरोक्त घोषणा Dim Arr(0 से 10) As Long के बराबर होती है।

यदि आपके पास 0 या 1, की एक आप्शन बेस स्टेटमेंट है, ऐरे का लोअर बाउंड उस मान पर सेट हो जाएगा।

तब कोड : Dim Arr(10) As Long, Dim Arr(0 To 10) As Long या Dim Arr(1 To 10) As Long, आप्शन बेस के मान के आधार पर सामान ही होंगे।

ऐरे के लोअर और अपर बाउंड को निर्दिष्ट करना एक अच्छी प्रोग्रामिंग प्रैक्टिस होती है। जब आप कोड को मोड्यूल या कहीं और कॉपी पेस्ट करते हैं तो गलतियों से बचाता है।

ऐरे में मानों को स्टोर करना (Storing values in an array)

ऐरे में मानों को निम्न तरीकों से भरा जाता है -

1 Marks(0)=55
Marks(1)=67
Marks(2)=55
Marks(3)=67
Marks(4)=74

2 Dim marks As Integer() = {55, 67, 87, 48, 90, 74}

3 Dim marks = New Integer() {1, 2, 4, 8}

4 Dim doubles = {1.5, 2, 9.9, 18}

आप ऐरे में एलिमेंट्स के प्रकार को अलग से भी निर्दिष्ट कर सकते हैं ऐरे लिटरल का उपयोग करके। इस मामले में, ऐरेलिटरल के मान ऐरे एलिमेंट के टाइप के होने चाहिए। निम्न उदाहरण इन्टिजर संख्याओं की सूची से डबल टाइप के ऐरे को बना रहा है :

```
Dim marks As Double() = {55, 67, 87, 48, 90, 74}
```

ऐरे द्वारा इटरेटिंग करना (Iterating through an Array)

ऐरे के साथ उनके मानों को अक्सेस करने के लिए for… next, Do …while आदि लूप स्टेटमेंट का उपयोग कर सकते हैं। ऐरे कोड का उदाहरण नीचे दिआ हुआ है -

```
Sub array_test()
Dim arr(5) As Integer
Dim n As Integer
arr(0) = 89
arr(1) = 56
arr(2) = 78
arr(3) = 45
arr(4) = 99
For n = LBound(arr) To UBound(arr) - 1
Debug.Print arr(n)
Next
End Sub
```

यहाँ LBound() और UBound() फंक्शन ऐरे ले लोअर और अपर बाउंड को लौटायेंगे।

बहु आयानी ऐरे (Multi dimensional Arrays)

बहु आयामी ऐरे में एक से अधिक रो और कॉलम होते हैं।

उदाहरण के लिए, Dim MyArray(5, 4) As Integer

Dim MyArray(1 To 5, 1 To 6) As Integer

निम्नलिखित उदाहरण में हम एक ऐरे परिभाषित करेंगे जिसके हर दो रो में 3 एलिमेंट होंगे।

```
Sub array_test()
Sub array_test()
Dim m, n As Integer
Dim arr(2, 4) As String
arr(0, 0) = "printer"
arr(0, 1) = "scanner"
arr(0, 2) = "mouse"
arr(0, 3) = "monitor"
arr(1, 0) = "usb"
arr(1, 1) = "ps2"
arr(1, 2) = "firewire"
arr(1, 3) = "serial"
For m = 0 To 2
   For n = 0 To 3
  Debug.Print arr(m, n); Spc(2);
   Next n
  Debug.Print
Next m
End Sub
```

गतिशील ऐरे (Dynamic Arrays)

एक ऐरे है जिसको Dim कथन में आकार नहीं दिया जाता है। इसके बजाय, यह ReDim कथन के साथ आकार दिया जाता है। गतिशील ऐरे चर उपयोगी होते हैं तब, जब हम पहले से नहीं जानते कि ऐरे में कितने तत्वों को संगेहित करने की आवश्यकता है या जब हमें बाद के चरण में ऐरे आयामों को बदलने की आवश्यकता है।

उदाहरण : Dim DynamicArray() As Long

ReDim DynamicArray(1 To 10)

यदि ऐरे ReDim कथन के साथ आकार दिया गया है, तो ऐरे (स्थैतिक या डायनामिक ऐरे) के रूप में आवंटित होती है। स्टेटिक सरणियों हमेशा आवंटित किए जाते हैं और कभी खाली नहीं रहते। एक गतिशील ऐरे के आकार को बदल सकते हैं, लेकिन नहीं डेटा प्रकार को नहीं। जब आप एक गतिशील ऐरे को मिटाते हैं, उस ऐरे के लिए आवंटित मेमोरी रिलीज़ हो जाती है। आपको इसे मिटाने के बाद इसका इस्तेमाल करने के लिए ReDim करना चाहिए।

गतिशील ऐरे को अभी तक ReDim बयान के साथ नहीं आकार दिया गया है, याइरेस स्टेटमेंट से दिलोकेट हो गयी है तो इस ऐरे को खली या अन एलोकेटेड ऐरे कहा जाता है। स्टेटिक एरे कभी भी अन एलोकेटेड या खाली नहीं होते हैं।

रेडिम स्टेटमेंट (ReDim Statement):

आप डायनामिक चर को खाली कोष्ठक के साथ घोषित कर सकते हैं यानि इंडेक्स डायमेंशन को खाली छोड़ना होगा। उसके बाद आप ReDim स्टेटमेंट द्वारा एक डायनामिक ऐरे का साइज़ दे सकते हैं या रेसाइज़ कर सकते क्योंकि यह अब घोषित हो चुकी है। ऐरे को दोबारा आकर देने के लिए अपर बाउंड देना आवश्यक है जबकि लोअर बाउंड वैकल्पिक है। यदि आप लोअर बाउंड का उल्लेख नहीं करते हैं, तो यह मॉड्यूल के लिए आप्शनबेस सेटिंग द्वारा निर्धारित किया जाता है जो डिफ़ोल्ट रूप से 0 है। आप मॉड्यूल के घोषणा अनुभाग में आप्शनबेस 1 निर्दिष्ट कर सकते हैं और फिर इंडेक्स 1 से शुरू होगा। इसका मतलब यह होगा कि 3 एलिमेंट्स वाली ऐरे के संबंधित इंडेक्समान 1, 2 और 3 होंगे। आप्शन बेस 1 इंटर नहीं करने का मतलब 0, 1 और 2 के सूचकांक मान होंगे।

निम्न उदाहरण A1 नामक एक ऐरे को गतिशील सरणी के रूप में घोषित करता है। ऐरे का आकार सेट नहीं होता है और फिर इसे 3 एलिमेंट्स वाली ऐरे में बदल दिया जाता है (ऑप्शनबेस 1 के द्वारा)

```
Sub arr_test()

'declare a dynamic array

Dim A() As String

ReDim A(3) As String

A(1) = "COPA"

A(2) = "DTPO"

A(3) = "MASE"

debug.print  A(1)  & " , " & A(2) & " , " & A(3)

End Sub
```

जब आप ReDim की वर्ड का उपयोग करते हैं तो आप वर्तमान में ऐरे में संग्रहित किसी भी मौजूदा डेटा को मिटा सकते हैं।

निम्नानुसार redim कथन का उपयोग करके ऊपर दिए गए उदाहरण में वर्णित ऐरे में एक और तत्व जोड़ें और इसे मान दें।

```
ReDim A(4) As String

A(4) = "CHNM"
```

अब जब ऐरे के मान फिर से प्रदर्शित होते हैं, तो पहले के मान रिक्त होंगे, क्योंकि वे redim स्टेटमेंट द्वारा मिटा दिए गए हैं। नीचे दिए गए उदाहरण यह दिखाता है :

```
Sub arr_test()

'declare a dynamic array

Dim A() As String

ReDim A(3) As String

A(1) = "COPA"

A(2) = "DTPO"

A(3) = "MASE"

ReDim A(4) As String

A(4) = "CHNM"

Debug.Print A(1) & " , " & A(2) & " , " & A(3) & "," & A(4)

End Sub
```

इस प्रोग्राम का परिणाम यह होगा , , ,CHNM

मौजूदा डेटा खोए बिना सरणी का आकार बदलने के लिए। आपको Redim के साथ "Preserve" का उपयोग करना होगा।

उदाहरण के लिए ReDim Preserve A(4) As String.

VBA में स्ट्रिंग मैनीपुलेशन (String manipulation in VBA)

उद्देश्य : इस पाठ के अन्त में आप निम्नलिखित कार्य करने योग्य होंगे :

- **VBA में स्ट्रिंग फंक्शन का वर्णन करना**
- **VBA में स्ट्रिंग रूपांतरण फंक्शन का वर्णन करना**
- **VBA में स्ट्रिंग एक्सट्रैक्शन फंक्शन का वर्णन करना**
- **VBA में स्ट्रिंग फोमेटिंग फंक्शन का वर्णन करना।**

परिचय (Introduction)

VBA मेंस्ट्रिंग, डेटा चर का एक प्रकार है जो टेक्स्ट, संख्यात्मक मानों, तारीख और समय और अल्फा नूमेरीक (वर्ण) से मिलकर बनता है। स्ट्रिंग्स अक्सर सभी प्रकार के डाटा को स्टोर करने के लिए इस्तेमाल किया जाता है और यह VBA का एक महत्त्वपूर्ण हिस्सा है। इसके लिए वेरिएबल की घोषणा करने के लिए आप स्ट्रिंग चर या वैरिएंटर डाटा टाइप का उपयोग कर सकत हैं।

स्ट्रिंग मैनीपुलेशन (String Manipulation)

VBA में स्ट्रिंग हैंडल करने के बहुत सारे फंक्शन्स हैं। निम्नलिखित कुछ उदाहरण हैं जहाँ आपको स्ट्रिंग हैंडलिंग फंक्शन की आवश्यकता पड़ेगी :

- यह जांचने के लिए की कही एक स्ट्रिंग में दूसरी स्टिंग निहित तो नहीं है।
- स्ट्रिंग के एक हिस्से को पार्स करना।
- स्ट्रिंग के एक हिस्से को दुसऋ वैल्यू से बदलना।
- स्ट्रिंग के लंबाई ढूँढना आदि।

स्ट्रिंग का संयोजन (String Concatenation)

दो या दो से अधिक स्ट्रिंग्स को "+" ऑपरेटर के द्वारा जोड़ा जा सकता है।

उदाहरण : Dim A, B, C As String

A = "www"

B = " and"

C = " the Internet"

Debug.Print A + B + C

यह "www and the Internet" प्रिंट करेगा।

यदि आपको दो या अधिक विभिन्न प्रकार के डाटा टाइप के चरों को जोड़ना है तब "&" ऑपरेटर उपयोग में आएगा।

उदाहरण : Dim person As String, pay As Integer

person = "Jaya"

pay = 25000

Debug.Print "The payment for "& person & " is " & pay

यह प्रिंट करेगा : The payment for Jaya is 25000

Len() फंक्शन (Len() function)

एक पूर्णांक जो या तो स्ट्रिंग में केरेक्टर की संख्या या नाममात्र बाइट की संख्या जो एक चर में स्टोर करने के लिए आवश्यक है वह रिटर्न करता है।

Syntax : Len (String)

उदाहरण : 1

Dim str As String

str = "Computer operator and programming assistant"

Debug.Print "The length of the string is "& Len(str)

यह प्रिंट करेगा : The length of the string is 43

Example : 2

Dim p, q As Integer, r As Double, dob As Date

p = Sqr(25)

q = 3333

r = 45.6789

dob = #1/1/1990#

Debug.Print "Size of p Is : " & Len(p)

Debug.Print "Size of q Is : " & Len(q)

Debug.Print "Size of r Is : " & Len(r)

Debug.Print "Size of dob Is : " & Len(dob)

यह प्रिंट करेगा : Size of p Is : 1

Size of q Is : 2

Size of r Is : 8

Size of dob Is : 8

Left()

यह एक स्ट्रिंग लौटाएगा जो स्ट्रिंग के बायीं ओर से कुछ निर्दिष्ट कैरक्टरों की संख्या निहित किए हुए हैं।

Syntax: Left(String ,Int)

Right()

यह एक स्ट्रिंग लौटाएगा जो स्ट्रिंग के दायीं ओर से कुछ निर्दिष्ट कैरक्टेरों की संख्या निहित किए हुए हैं।

Syntax: Right(String, Int)

Mid()

यह एक स्ट्रिंग लौटाएगा जो किसी स्ट्रिंग के सारे कैरक्टेरों को निहित किए हुए हैं।

Syntax:Mid(String, Int, Int)

यह एक स्ट्रिंग लौटाएगा जो किसी स्ट्रिंग के सारे कैरक्टेरों को किसी निर्दिष्ट पोसशिन से किये हुए हैं।

Syntax: Mid(String, Int, Int)

यह एक स्ट्रिंग लौटाएगा जो किसी स्ट्रिंग की निर्दिष्ट पोसिशन से शुरू कुछ निर्दिष्ट कैरक्टेरों की संख्या निहित किए हुए हैं। उदाहरण हैं :

1 Dim s AsString
  s =" Indiana Jones"
  debug.print Left(s)
  This will print : India
2 Dim s AsString
  s =" FUNDAMENTALLY"
  debug.print right(s, 5)
  This will print : TALLY
3 Dim s AsString
  s =" wholehearted"
  debug.print mid(s, 6)
  This will print : hearted
4 Dim s AsString
  s =" wholehearted"
  debug.print mid(s, 6,4)
  यह प्रिंट करेगा : hear

Ltrim()

यह एक ऐसी स्ट्रिंग लौटाता है जो बिना लीडिंग स्पेस के साथ (LTrim)

Syntax :LTrim(String)

RTrim()

यह एक ऐसी स्ट्रिंग लौटाता है जो बिना ट्रेलिंग स्पेस के साथ किसी निर्दिष्ट स्ट्रिंग की प्रति को निहित किए हुए हैं।

Syntax :RTrim(String)

Trim()

यह एक ऐसी स्ट्रिंग लौटाता है जो बिना ट्रेलिंग स्पेस लीडिंग स्पेस के साथ किसी निर्दिष्ट स्ट्रिंग की प्रति को निहित किए हुए हैं।

Syntax : Trim(String)

Examples: Dim A as String

A = " Adjustment "

Debug.Print "For everyone" &LTrim(A) & "is a must"

Debug.Print "For everyone"; RTrim(A) & "is a must"

Debug.Print "For everyone"; Trim(A) & "is a must"

यह प्रिंट करेगा : For everyoneAdjustment is a must

For everyone Adjustmentis a must

For everyone Adjustmentis a must

Instr()

यह स्ट्रिंग की पहली घटना की प्रारंभिक स्थिति को किसी अन्य स्ट्रिंग के भीतर निर्दिष्ट करने वाला एक पूर्णांक देता है।

Syntax: InStr([start,]string1, string2[, compare])

Example:

A = "hairdresser"

B = "dress"

Debug.Print "The second string starts at position no. " &InStr(1, A, B) "

यह प्रिंट करेगा : The second string starts at position no. 5

स्टिंग को बदलना (Replacing Strings)

Replace() फ़ंक्शन वर्णों के अनुक्रम को 30 वें वर्णों के दूसरे सेट में बदल देता है।

Replace(source_string, find_string, replacement_string).

Eample: Debug.Print Replace("majordrawback", "drawback", "advantage")

Debug.Print Replace("majority", "aj", "in", 1)

Debug.Print Replace("think and think", "i", "a", 1, 1)

यह प्रिंट करेगा : major advantage

minority

thank and think

Val()

VAL() फ़ंक्शन एक स्ट्रिंग को इनपुट के रूप में स्वीकार करता है और उस

स्ट्रिंग में मिली संख्याओं को वापस देता है। जब इसे पहली गैर-संख्यात्मक मिलती है तो VAL फ़ंक्शन स्ट्रिंग को पढ़ना बंद कर देगा।

Syntax: Val(String)

Example: Dim s1, s2, s3 As String

s1 = "6 feet 1 inch is his height"

s2 = "5 - 6 kms is the distance to my office from here"

s3 = "011 22222222 is my telephone number"

Debug.Print Val(s1)

Debug.Print Val(s2)

Debug.Print Val(s3)

यह प्रिंट करेगा : 6

5

1122222222

स्ट्रिंग कन्वर्शन फंक्शन (The String Conversion Functions)

LCase()

स्ट्रिंग या करैक्टर को लोअर केस में बदल कर लैटाता है।

Syntax :LCase(String)

UCase()

स्ट्रिंग या करैक्टर को अपर केस में बदल कर लौटाता है।

Syntax :UCase(String)

उदाहरण :

Dim A, B as String

A="IF YOU FEAR YOU WILL BECOME WEAK"

B = "be a strong person"

Debug.PrintLcase(A)

Debug.PrintUCase(B)

यह प्रिंट करेगा : if you fear you will become weak

BE A STRONG PERSON

Str()

Str() फ़ंक्शन एक संख्या को एक स्ट्रिंग में परिवर्तित करता है।

CStr()

CStr() फंक्शन किसी प्रकार के मान को स्ट्रिंग में कनवर्ट करने के लिए उयोग किया जाता है।

Syntax: Str(number as variant)

उदाहरण :

1 Dim Number As Double

Number = 1450.5

Debug.Print "The string is "&str(Number)

यह प्रिंट करेगा : The string is 1450.5

2 Dim Date_of_birthAs Date

Date_of_birth = #1/1/1990#

Debug.Print CStr(Date_of_birth)

यह प्रिंट करेगा : 01/01/1990

Asc()

Asc() फ़ंक्शन एक इंटीजर मान देता है जो किसी वर्ण या किसी स्ट्रिंग में पहले वर्ण में संबंधित ASCII कोड का प्रतिनिधित्व करता है।

Syntax :Asc(String)

उदाहरण : Asc("A") will return 65

Chr()

Chr() फ़ंक्शन निर्दिष्ट ASCII कोड से जुड़े कैरेक्टरों को वापस देता है।

Syntax :Chr(Integer)

उदाहरण :Chr(68) will return the character "D"

स्ट्रिंग को रिवर्स करना (Reversing a String)

StrReverse(String)

StrReverse() एक स्ट्रिंग देता है जिसमें एक निर्दिष्ट स्ट्रिंग का करैक्टर क्रम उलट दिया जाता है।

उदाहरण : Dim A As String

A = "desserts"

Debug.Print StrReverse(A)

यह प्रिंट करेगा : stressed

Format() फंक्शन (Format() function)

Format() फ़ंक्शन एक वेरिएंट (स्ट्रिंग) देता है जिसमें एक प्रारूप होता है। एक औपचारिक एक्सप्रेशन में निहित निर्देशों के अनुसार। इसका उपयोग फॉर्मेट की हुई तिथियों के साथ ही फॉर्मेट की हुई स्ट्रिंग करने के लिए किया जा सकता है।

Syntax(for FormattingStrings) : Format(String, Format)

यूजर द्वारा परिभाषित किए गए स्ट्रिंग फोर्मट्स (Format Function)

यह स्ट्रिंग्स के लिए प्रारूप एक्सप्रेशन बनाने के लिए आप निम्न में से किसी भी करैक्टर (वर्ण) का उपयोग कर सकते हैं :

Example :Dim x As String

x = "change case"

Debug.Print Format(x, ">")

This will print "CHANGE CASE"

वर्ण	विवरण
@	वर्ण प्लेसहोल्डर। एक वर्ण या एक स्पेस को प्रदर्शित करता है। यदि स्ट्रिंग में उस स्थिति में एक वर्ण होता है जहाँ प्रारूप स्ट्रिंग में प्रतीक (@) दिखाई देता है तो यह उसको प्रदर्शित करती है अन्यथा, उस स्थिति में एक स्थान प्रदर्शित करती हे। प्लेसहोल्डर को दाएं से बाएं तक भर दिया जाता है। जब तक फॉर्मेट स्ट्रिंग में कोई विस्मयादिबोधक बिंदु का वर्ण (!) नहीं होता।
&	वर्ण प्लेसहोल्डर। एक वर्ण कोय कुह नहीं प्रदर्शित करता है। यदि स्ट्रिंग में उस स्थिति में एक वर्ण होता है जहाँ प्रारूप स्ट्रिंग में प्रतीक (&) दिखाई देता है तो यह उसको प्रदर्शित करती है अन्यथा, उस स्थिति में कुछ भी प्रदर्शित नहीं करती है। प्लेसहोल्डर को दाएं से बाएं तक भर दिया जाता है। जब तक फॉर्मे स्ट्रिंग में कोई विस्मयादिबोधक बिंदु वर्ण (!) नहीं होता।
<	लोअरकेस लागू करता है। लोअरकेस प्रारूप में सभी वर्णों को प्रदर्शित करता है।
>	अपरकेस लागू करता है। अपरकेस प्रारूप में सभी वर्णों को प्रदर्शित करता है।
!	प्लेसहोल्डर्स को सही भरने के लिए जोर से ऊपर उठायें। डिफ़ॉल्ट रूप से प्लेसहोल्डर को दाएं से बाएं भरता है।

VBA में बिल्ट इन फंक्शन्स Built in Functions in VBA

उद्देश्य : इस पाठ के अन्त में आप निम्नलिखित कार्य करने योग्य होंगे :

- **VBA में गणित कार्यों का वर्णन करना**
- **VBA में तार्किक कार्यों का वर्णन करना**
- **VBA में दिनांक/समय कार्यों का वर्णन करना**
- **VBA में रूपांतरण कार्यों का वर्णन करना।**

परिचय (Introduction)

VBA के कार्यों में बिल्ट-इन फंक्शन का संग्रह है जो आपके लिए विभिन्न प्रकार के कार्यों और गणना करता है। डेटा प्रकारों को बदलने, तिथियों पर गणना करने, सरल से जटिल गणित के लिए, वित्तीय गणना करने, टेक्स्ट स्ट्रिंग प्रबंधित करने, मानों को फॉर्मेट करने एवं दूसरी तालिकाओं में डेटा प्राप्त करने के लिए फंक्शन है। VBA के बिल्ट फंक्शन का उपयोग करने से उपयोगकर्ता के लिए कोडिंग बहुत आसानी हो जाएगी। हमने पहले से ही हमारे पूर्व अभ्यासों में कई बिल्ट इनफंक्शन को उपयोग किया है जैसे कि msgbox फंक्शन और कई स्ट्रिंग मैनिपुलेशन फंक्शंस।

MS Excel: VBA Functions (VBA Formulae) - Category wise

एक्सेल में आमतौर पर प्रयुक्त VBA फंक्शंस, श्रेणी द्वारा क्रमबद्ध, यहाँ दिखाए जाते हैं।

स्ट्रिंग फंक्शन (String Functions) : स्ट्रिंग फंक्शन पहले से ही सम्बंधित थ्योरी Ex. 2.2.09 में समझाया गया है।

मैथ्स फंक्शन (Math Functions)

टेवल 1 गणितीय श्रेणी में कुछ सामान्य बिल्ट-इन फंक्शन को सूचीबद्ध करता है।

टेबल 1

फंक्शन का नाम	विवरण
Abs	एक संख्या का पूर्ण मूल्य देता है।
Cos	निर्दिष्ट कोण के कोसाइन देता है।
Cosh	निर्दिष्ट कोण के हाइपरबॉलिक कोसाइन देता है।
Exp	e की निर्दिष्ट पॉवर देता है (नेचुरल लॉगरिदम का आधार)।
Fix	एक संख्या का पूर्णांक वाला हिस्सा देता है।
Format	संख्यात्मक एक्सप्रेशन लेता है और इसे फॉर्मेट किए हुए स्ट्रिंग के रूप में देता है।
Int	एक संख्या का पूर्णांक वाला हिस्सा देता है।
Log	एक निर्दिष्ट संख्या में निर्दिष्ट संख्या के निर्दष्ट प्राकृतिक (बेस e) लॉगरिदम या निर्दिष्ट आधार में निर्दिष्ट संख्या का लॉगरिदम लैटाता है।
Rnd	एक या दृच्छिक संख्या उत्पन्न करता है (पूर्णांक मान)।
Round	निकटतम अभिन्न मूल्य या आंशिक अंकों की निर्दिष्ट संख्या में गोलाकार दशमलब या डबल मान देता है।
Sign	एक संख्या के संकेत को इंगित करने वाला इंटीजर मान देता है।
Sin	एक निर्दिष्ट कोण को साइन देता है।
Sqr	निर्दिष्ट संख्या के वर्ण रूट देता है।
Tan	एक निर्दिष्ट कोण के स्पर्शक देता है।
Val	इनपुट के रूप में एक स्ट्रिंग स्वीकार करता है और उस स्ट्रिंग में मिली संख्याओं को वापस देता है।

```
1  Dim a, b as integer                    debug.print sqr(a)
   a=81                                   This will display the square root of 81 ie. 9
```

तार्किक फंक्शन (Logical Functions)

टेबल 2, तार्किक श्रेणी में कुछ सामान्य बिल्ट-इन फंक्शन को सूचीबद्ध करता है -

टेबल 2

फंक्शन का नाम	विवरण
ISDATE	अगर एक्सप्रेशनवैध दिनांक है तो सत्य लौटाता है। इन्यथा, यह गलत लौटाता है।
ISERROR	त्रुटि मानों के लिए जाँच करता है।
ISNULL	अगर अभिव्यक्ति एक शून्य मान है तो सही लौटाता है। अन्यथा, या गलत लौटाता है।
ISNUMERIC	अगर अभिव्यक्ति वैध संख्या है तो सत्य लौटाता है। अन्यथा, यह गलत लौटाता है।

उदाहरण :

```
1  Sub Button1_Click()
   N = TextBox1.Text
   If IsNumeric(N) = True Then
   MsgBox "correct"
   Else
   MsgBox "Insert only numbers"
   End If
```

यह जाँचता है कि टेक्स्ट बॉक्स में दर्ज डेटा एक संख्या है या नहीं।

```
2  Sub Button1_Click()
   N = TextBox1.Text
   If IsDate(N) = True Then
   MsgBox "correct"
   Else
   MsgBox "Insert only dates"
   End If
   End Sub
```

यह जाँचता है कि टेक्स्ट बॉक्स में दर्ज डेटा मान्य दिनांक है या नहीं।

डेट/टाइम फंक्शन (Date / Time Functions)

टेबल 3 डेटा/टाइम श्रेणी में कुछ सामान्य निर्मितफंक्शन सूचीबद्ध करता है।

टेबल 3

फंक्शन	रिटर्न मान
DATE	वर्तमान सिस्टम दिनांक देता है।
DATEADD	एक तारीख लौटाता है जिसके बाद एक निश्चित समय/दिनांक अंतराल जोड़ा गया है।
DATEDIFF	अंतराल के आधार पर दो दिनांक मानों के बीच अंतर होता है।
DATEPART	किसी दिए गए दिनांक का निर्दिष्ट हिस्सा देता है।
DATESERIAL	एक वर्ष, महीने और दिन देने पर दिनांक देता है।
DATEVALUE	एक तिथि के क्रम संख्या को रेट करता है।
DAY	माह का दिन लौटाता है (1 से 31 तक की संख्या) एक तिथि देने पर
FORMAT Dates	एक डेट एक्सप्रेशन लेता है और फॉर्मेट किए हुए स्ट्रिंग लौटाता है।
HOUR	समय के घंटे लौटाता है (0 से 23 तक की संख्या) एक तिथि देने पर।
MINUTE	एक समय मान के मिनट को (0 से 59 देता) देता है।
MONTH	दिनांक मान देने पर महीना लौटाता है (1 से 12 तक की संख्या)
MONTHNAME	माह के 1 से 12 तक की संख्या के रूप में एक स्ट्रिंग देता है।
NOW	वर्तमान सिस्टम दिनांक और समय देता है।
TIMESERIAL	ऑवर, मिनट और सेकंड देने पर समय देता है।
TIMEVALUE	समय का सीरियल मान लौटाता है।
WEEKDAY	डेटा मान देने पर सपथ के दिन का प्रतिनिधित्व करने वाली संख्या लौटाता है।
WEEKDAYNAME	1 से 7 तक नम्बर दिए जाने पर सप्ताह के दिनों का प्रतिनिधित्व करते हुए स्ट्रिंग लौटाएगा।
YEAR	डेटा आग्युमेंट का ईयर वाला हिस्सा लौटाता है।

उदाहरण

1 DateDiff() Function

DateDiff function का सिंटेक्स इस प्रकार है :

DateDiff (interval, date1, date2, [firstdayofweek], [firstweekofyear])

पैरामीटर्स या आर्ग्यूमेंट्स (Parameters or Arguments)

इंटरवल, date1 और date2 के बीच के समय के अंतराल की गड़ना करने के लिए उपयोग किया जाएगा। नीचे टेबल 4 में वैध इंटरवल की सूची प्रदिर्शित कर रहा है।

टेबल 4

अंतराल	वर्णन
yyyy	Year
q	Quarter
m	Month
y	Day of year
d	Day
w	Weekday
ww	Week
h	Hour
n	Minute
s	Second

Date1 और Date2 वे डेट्स है जिनके बीच का अंतर पता करना है।

सप्ताह का पहला दिन वैकल्पिक है। यह एक कांस्टेंट है जो सप्ताह का पहला दिन दर्शा रहा है। यदि यह पेरामीटर हटा दिया जाता है तो एक्सेल यह मानेगा की सन्डे ही सप्ताह का पहला दिन है।

वर्ष का पहला सप्ताह वैकल्पि है। यह एक कांस्टेंट है जो वर्ष का पहला सप्ताह दर्शा रहा है। यदि यह पेरामीटर हटा दिया जाता है तो एक्सेल यह मानेगा कि 1st जनवरी ही वर्ष का पहला सप्ताह किए हुए हैं।

```
Sub test()
Debug.PrintDateDiff("yyyy", "1/12/1999", "31/1/2000")
Debug.PrintDateDiff("q", "1/12/1999", "31/1/2000")
Debug.PrintDateDiff("m", "1/12/1999", "31/1/2000")
End Sub
```

परिणाम होगा

1

4

12

2 डेट फॉर्मेट करना (Format Date)

Syntax

माइक्रोसॉफ्ट Excel के FORMAT फंक्शन का सिंटेक्स इस प्रकार है :

Format (expression, [format, [firstdayofweek, [firstweekofyear]]])

पेरामीटर या आर्ग्युमेंट (Parameters or Arguments)

एक्सप्रेशन एक मान है जिसे फॉर्मेट करना है।

फॉर्मेट वैकल्पिक है। यह एक फॉर्मेट जो एक्सप्रेशन पर लागू किया जाता है। आप या तो अपना खुद का फॉर्मेट परिभाषित कर सकते हैं या फिर एक्सेल के पहले से परिभाषित फॉर्मेट का उपयोग कर सकते हैं जैसे टेबल 5 में दर्शाया गया है।

टेबल 5

फॉर्मेट	वर्णन
General Date	आपकी सिस्टम सेटिंग्स के आधार पर दिनांक प्रदर्शित करता है।
Long Date	आपके सिस्टम की लंबी तिथि सेटिंग के आधार पर दिनांक प्रदर्शित करता है।
Medium Date	आपके सिस्टम की मध्यम दिनांक सेटिंग के आधार पर दिनांक प्रदर्शित करता है।
Short Date	आपके सिस्टम की छोटी दिनांक सेटिंग के आधार पर दिनांक प्रदर्शित करता है।
Long Time	आपके सिस्टम की लंबी समय सेटिंग के आधार पर समय प्रदर्शित करता है।
Medium Time	आपके सिस्टम की मध्यम समय सेटिंग के आधार पर समय प्रदर्शित करता है।
Short Time	आपके सिस्टम की छोटी समय सेटिंग के आधार पर समय प्रदर्शित करता है।

सप्ताह का पहला दिन वैकल्पिक है। यह एक मान है जो सप्ताह का पहला दिन दर्शा रहा है। यदि यह पेरामीटर हटा दिया जाता है तो FORMAT फंक्शन यह मानेगा की सन्डे ही सप्ताह का पहला दिन है। पेरामीटर टेवल 6 में से कोई सा भी मान हो सकता है।

वर्ष का पहला सप्ताह वैकल्पिक है। यह एक मान है जो वर्ष का पहला सप्ताह दर्शा रहा है। यदि यह पेरामीटर हटा दिया जाता है तो FORMAT फंक्शन यह मानेगा कि 1 जनवरी ही वर्ष का पहला सप्ताह लिए हुए है। पेरामीटर टेवल 7 में से कोई सा भी मान हो सकता है।

```
Sub test()
Debug.Print Format(#1/1/1990#, "Short Date")
Debug.Print Format(#1/1/1990#, "Long Date")
Debug.Print Format(#1/1/1990#, "yyyy/mm/dd")
End Sub
```

टेबल **6**

कांस्टेंट	मान	विवरण
vbUseSystem	0	NLS API सेटिंग का उपयोग
VbSunday	1	रविवार (यदि डिफॉल्ट पेरामीटर नहीं है तो)
vbMonday	2	सोमवार
vbTuesday	3	मंगलवार
vbWednesday	4	बुधवार
vbThursday	5	गुरूवार
vbFriday	6	शुक्रवार
vbSaturday	7	शनिवार

The result will be

1/1/1990

टेबल **7**

कांस्टेंट	मान	विवरण
vbUseSystem	0	NLS API सेटिंग का उपयोग।
vbFirstJan1	1	वह सप्ताह जिसमें 1 जनवरी आ रहा है।
vbFirstFourDays	2	पहले सप्ताह जिसमें साल के कम से कम 4 दिन हों।
vbFirstFullWeek	3	साल का पहला पूरा सप्ताह।

Monday, January 01, 1990

1990/01/01

डेटा टाइप को कन्वर्ट करने वाले फंक्शन (Data Type Conversion Functions)

नीचे दिया गया टेबल 8 डाटा टाइप कन्वर्शन श्रेणी में कुछ सामान्य बिल्ट-इन फंक्शन को सूचीबद्ध रहा है।

टेबल 8

फंक्शन	रिटर्न मान का प्रकार	एक्सप्रेशन आर्ग्युमेंट रेंज
CBool	Boolean	कोई वैध स्ट्रिंग या नुमेरिक एक्स्प्रेशन।
CByte	Byte	0 से 255.
CCur	Currency	-922,337,203,685,477.5808 to 922,337,203,685,477.5807.
CDate	Date	कोई वैध डेट एक्स्प्रेशन।
CDbl	Double	-1.79769313486231E308 से -4.94065645841247E-324 निगेटिव मान के लिए; 4.94065645841247E- 324 से 1.79769313486232E308 पोसिटिव मान के लिए।
CDec	Decimal	+/-79,228,162,514,264,337,593,543,950,335 शून्य-स्केल्ड संख्याओं के लिए यानी कोई दशमलव स्थान वाले नंबर हैं। 28 दशमलव स्थानों वाले नंबरों के लिए, श्रेणी +/- 7.9228162514264337593543950335 है। सबसे छोटी संभव गैर-शून्य संख्या 0.0000000000000000000000000001 है।
CInt	Integer	-32,768 से 32,767; फ्रैक्शन राउंड किये गए हैं।
CLng	Long	-2,147,483,648 से 2,147,483,647; fractions are rounded.
CSng	Single	-3.402823E38 से -1.401298E-45 निगेटिव मान के लिए ; 1.401298E-45 से 3.402823E38 पोसिटिव मान के लिए,
CStr	String	CStr के लिए रिटर्न करता है, एक्सप्रेशन आर्ग्युमेंट के अधार पर।
CVar	Variant	नुमेरिक के लिए डबल जैसे रेंज, नॉन-नुमेरिक के लिए सृंग के समान रेंज।

उदाहरण

```
CDate function
Sub test()
Dim lNum As Long
Dim a As String
a = 12345
Debug.PrintCDate(a)
b = "January 1, 1990"
Debug.PrintCDate(b)
c = "1:23:45 PM"
Debug.PrintCDate(c)
End Sub
This will display
10/18/1933
1/1/1990
1:23:45 PM
```