Programowanie w VisualBasic
PODSTAWY

1 Definiowanie zmiennych

Definiowanie zmiennych w języku `VisualBasic` odbywa się z użyciem słow kluczowych `Dim` i `As` oraz słowa kluczowego określającego typ zmiennej. Najpopularniejsze typy zmiennych w zależności od ich zawartości:

- **Integer** - liczba całkowita (np. A = 4)
- **Single/Double** - liczba zmiennoprzecinkowa (np. B = 3.567)
- **String** - ciąg znaków (np. C = "dowolny tekst")

Przykładowo, aby zdefniować zmienną A typu całkowitego, B typu zmiennoprzecinkowego i C jako ciąg znaków należy użyć następującej składni:

```
Dim A As Integer, B As Double, C As String
```

2 Procedury

Procedura jest wydzielonym fragmentem programu, który zostaje wykonany po wywołaniu nazwy procedury. Procedury typu `Sub` wykonują zadania, ale nie zwracają wartości.

```
Sub NazwaProcedury(ListaArgumentow)
    ...
    treść procedury
    ...
End Sub
```

3 Odczytywanie i zapisywanie wartości

Aby odczytać wartości lub zapisać je do arkusza należy w pierwszej kolejności odwołać się do konkretnego arkusza używając jego nazwy, a następnie do konkretnej komórki podając najpierw numer wiersza, a następnie numer kolumny. Numery kolumn określane są za pomocą kolejnych liczb całkowitych, a nie liter.

Odczytywanie zawartości komórki A2 i zapisywanie jej do zmiennej X:

```
X = Arkusz1.Cells(2,1).Value
```

Zapis wartości zmiennej Y do komórki B3:

```
Arkusz1.Cells(3,2).Value = Y
```

4 Wyświetlanie okien informacyjnych

Wyświetlanie okna informacyjnego zawierającego zdefiniowaną treść:

```
Dim Msg As String

Msg = "Tekst do wiadomości w oknie"
MsgBox Msg, vbOKOnly + vbInformation, "Tytul okna"
```

Prametr **vbOKOnly** pozwala na wyświetlenie w okienku przycisku **OK**, a parametr **vbInformation** odpowiedzialny jest za wyświetlanie w tym samym oknie ikony informacyjnej.

5 Komunikacja z użytkownikiem

Język `VisualBasic` umożliwia wyświetlanie okien dialogowych z miejscem do wpisania przez użytkownika odpowiedniej wartości, która może być następnie zapisana do zmiennej:

```
Dim Tresc As String, Tytul As String

Tresc = "Podaj dowolona wartosc:"
Tytul = "Tytul okna..."

A = Application.InputBox(Tresc, Tytul)
```

Programowanie w VisualBasic

STRUKTURA INSTRUKCJI WARUNKOWEJ

Instrukcja warunkowa wykorzystywana jest w programach, które wymagają od użytkownika decyzji odnośnie rodzaju wykonanych działań oraz w sytuacjach, kiedy wykonanie segmentów kodu (zbioru operacji) zależy od spełnienia określonego warunku lub warunków.

1 Podstawowa struktura instrukcji warunkowej

Podstawowa struktura instrukcji warunkowej składa sie ze słów kluczowych `If`, `Then`, `Else` i `End If` oraz warunku występującego po słowie kluczowym `If`. W przypadku, gdy warunek jest spełniony, wykonywany jest blok instrukcji T. Jeśli warunek nie jest spełniony, wykonywany jest blok instrukcji **F**.

```
If warunek Then
    ...
    blok instrukcji T
    ...
Else
    ...
    blok instrukcji F
    ...
End If
```

Nie jest możliwe jednoczesne wykonanie obu bloków instrukcji **T** i **F** oraz niewykonanie żadnego z nich.

Możliwe jest zastosowanie skróconej struktury instrukcji warunkowej z pominięciem słowa kluczowego `Else` oraz bloku instrukcji wykonywanych w przypadku niespełnienia zadanego warunku. Jeżeli zadany warunek jest spełniony, wykonywany jest blok instrukcji **T**, a w przeciwnym razie nie są wykonywane żadne operacje.

```
If warunek Then
    ...
    blok instrukcji T
    ...
End If
```

2 Rozszerzona struktura instrukcji warunkowej

Istnieje rozszerzona struktura instrukcji warunkowej zawierająca więcej niż jeden warunek. Kolejne warunki do sprawdzenia zadawane są po słowie kluczowym `ElseIf`. Tak, jak w przypadku podstawowej struktury instrukcji warunkowej, tutaj również cała konstrukcja powinna być zakończona słowem kluczowym `Else` i blokiem instrukcji **F**.

```
If warunek1 Then
    ...
    blok instrukcji T1
    ...
ElseIf warunek2 Then
    ...
    blok instrukcji T2
    ...
Else
    ...
    blok instrukcji F
    ...
End If
```

3 Operatory porównania i operatory logiczne

Podczas konstruowania warunków występujących w instrukcjach warunkowych możliwe jest wykorzystywanie operatorów porównania:

- $>$ / $>=$ większy od / większy lub równy od
- $<$ / $<=$ mniejszy od / mniejszy lub równy od
- $=$ / $<>$ równy / różny

Dostępnymi w języki `VisualBasic` operatorami logicznymi są między innymi:

- **Not**(warunek) - negacja warunku
- warunek1 **And** warunek2 - koniunkcja warunków
- warunek1 **Or** warunek2 - alternatywa warunków

Dozwolone jest łączenie wielu operatorów porównania i operatorów logicznych w jednym warunku. Odpowiednią kolejność sprawdzania warunków zapewnić można stosując nawiasy okrągłe.

Programowanie w VisualBasic

STRUKTURA PĘTLI

Pętla jest jedną z podstawowych struktur programistycznych - umożliwia cykliczne wykonywanie ciągu instrukcji określoną liczbę razy, do momentu zajścia pewnych warunków lub tak długo, aż pewne warunki są spełnione.

1 Pętla z licznikiem (For ... Next)

Pętla `For ... Next` powtarza blok instrukcji określoną liczbę razy. Stosujemy ją wtedy, gdy z góry wiadomo ile razy pętla ma być wykonana. Po każdym **obrocie pętli** (wykonaniu bloku instrukcji wewnątrz pętli) **iterator pętli** I zwiększany jest od wartości początkowej P do wartości końcowej K z krokiem S. Definicję kroku można pominąć, jeśli ma on wynosić 1.

```
For I = P To K [Step S]
    ...
    blok instrukcji
    ...
Next I
```

W celu zakończenia wykonywania pętli `For ... Next` wykorzystywane jest polecenie `Exit For`. **Przerywanie działania pętli `For ... Next` jest jednak niewskazane.**

2 Pętla warunkowa (Do ... Loop)

Pętla warunkowa `Do ... Loop` ma w języku `VisualBasic` kilka postaci. Jest to pętla warunkowa, a więc jej wykonywanie zależne jest od spełnienia (lub nie) założonego warunku. W najprostszej postaci składa się z dwóch słów kluczowych `Do` i `Loop` oraz bloku instrukcji zawartego między nimi.

```
Do
    ...
    blok instrukcji
    ...
Loop
```

W powyższej formie pętla ta jest pętlą **nieskończoną**. Aby zakończyć działanie pętli `Do ... Loop`, w bloku instrukcji należy wykorzystać polecenie `Exit Do`, umieszczając je wewnątrz instrukcji warunkowej `If ... Then`. Pętla w takiej postaci zostanie zakończona, kiedy założony warunek zostanie spełniony.

```
Do
    If warunek Then
        Exit Do
    End If
Loop
```

Rozszerzeniami podstawowej pętli `Do ... Loop` są pętle, które już w swojej strukturze zawierają warunek, który decyduje o ich wykonaniu bądź nie. Nie jest wtedy konieczne korzystanie z polecenia `Exit Do` do przerywania działania pętli.

2.1 Pętla Do ... Loop z wyrażeniem While

Blok instrukcji wewnątrz pętli jest wykonywany tak długo, jak długo założony warunek jest **prawdziwy**.

```
Do While warunek
    ...
    blok instrukcji
    ...
Loop
```

```
Do
    ...
    blok instrukcji
    ...
Loop While warunek
```

2.2 Pętla Do ... Loop z wyrażeniem Until

Blok instrukcji wewnątrz pętli jest wykonywany tak długo, jak długo założony warunek jest **fałszywy**.

```
Do Until warunek
    ...
    blok instrukcji
    ...
Loop
```

```
Do
    ...
    blok instrukcji
    ...
Loop Until warunek
```

Umieszczenie słów kluczowych `While` i `Until` po słowie kluczowym `Do` lub `Loop` decyduje o tym, czy może zaistnieć sytuacja, w której blok instrukcji wewnątrz pętli `Do ... Loop` nie zostanie wykonany ani raz.