**Wydział Zarządzania AGH**
Katedra Informatyki Stosowanej

# Podstawy VBA cz. 3

**Programowanie komputerowe**

# Program wykładu

- Typy danych
- Deklaracja zmiennych i stałych
- Zmienne tablicowe
- Wyrażenia
- Operatory

# Dane i typy danych

- Programy manipulują danymi, które są przechowywane w zmiennych lub stałych. Zmienne mają różne atrybuty i mogą przechowywać różne typy danych (liczby, tekst, data i tak dalej). Stąd termin **typ danych**.
- Typ danych określa sposób, w jaki są interpretowane bity przechowywane w pamięci. W węższym rozumieniu typ danych jest zbiorem wartości, które może przyjmować zmienna. Pojęcie typu danych jest fundamentalne w nowocześnie pojmowanym programowaniu.
- W języku VB wyróżniamy 11 podstawowych (standardowych, predefiniowanych) typów danych. Oprócz tego typy mogą być definiowane przez użytkownika, np. typ “konto” z operacjami “wpłata”, “wypłata” i “stan konta”.

# Dane i typy danych

| Typ | Zawartość | Wielkość pamięci | Zakres |
|---|---|---|---|
| Byte | całkowita | 1 bajt | 0 do 255 |
| Boolean | całkowita | 2 bajty | True lub False |
| Integer | całkowita | 2 bajty | -32,768 do 32,767 |
| Long | całkowita | 4 bajty | -2,147,483,648 do 2,147,483,647 |
| Single | liczba zmiennopozycyjna | 4 bajty | Ujemne:-3.402823E38 do -1.401298E-45<br>Dodatnie: 1.401298E-45 do 3.402823E38 |
| Double | liczba zmiennopozycyjna | 8 bajtów | Ujemne:-0.79769313486232E308 do -4.94065645841247E-324<br>Dodatnie: 4.94065645841247E-324 do 1.79769313486232E308 |
| Currency | liczba z ustaloną kropką dziesiętną | 8 bajtów | -922,337,203,685,477.5808 do +922,337,203,685,477.5807 |
| Date | data (miesiąc/dzień/rok) | 8 bajtów | 1/1/100 do 1/31/9999 |
| Object | adres obiektu | 4 bajty | dowolna referencja do obiektu |
| String | łańcuch znaków | 10 bajtów + długość stringu (dla zmiennej długości stringu);<br>długość stringu (dla stringów o ustalonej długości) | 0 do 2 miliardów znaków<br>(0 do 65,400 dla Windows 3.1 i wersji wcześniejszych) |
| Variant | dowolne dane (poza stringiem o ustalonej długości i typem zdefiniowanym przez użytkownika) | 16 bajtów (dla danych liczbowych);<br>22 bajty + długość stringu (dla stringu) | Wartość liczbowa (dla danych liczbowych);<br>jak w przypadku stringów zmiennej długości |

# Zmienne i typy zmiennych

**Zmienna** (ang. *variable*) para uporządkowana: nazwa i przyporządkowana jej wartość, tzn. reprezentant danych określonego typu, istniejący jako miejsce w pamięci.

**Zmienna** – obiekt mogący zmieniać swoją wartość w trakcie wykonywania programu.

Zmienna jest dostępna w programie za pomocą nazwy użytej w deklaracji. Podczas wykonywania programu zmiennej można przypisywać różne wartości, zawsze jednak zgodne z zadeklarowanym typem. Do przypisania wartości zmiennej służy instrukcja przypisania.

Typy zmiennych są tym samym, co omówione poprzednio typy danych.

# Stałe i typy stałych

**Stała** (ang. *constant*) nazwa lub zmienna programowa o przypisanej wartości początkowej, nie zmieniająca się w trakcie wykonania programu.

**Stała** – obiekt, który po jednokrotnym przypisaniu, nie może zmieniać wartości w trakcie wykonywania programu.

Stała jest dostępna w programie za pomocą nazwy użytej w deklaracji.

# Nazwy zmiennych/stałych

Deklarując w VBA zmienną/stałą należy pamiętać, że nazwa zmiennej/stałej:

- musi zaczynać się od litery (nie może zaczynać się od cyfry),
- nie może zawierać znaków (symboli) ze zbioru: { kropka, przecinek, spacja, !, $, @, #, [,], {,},} oraz operatorów,
- nie może być dłuższa niż 255 znaków,
- nie może być taka sama jak nazwa: procedur, funkcji, symboli, stałych oraz słów kluczowych języka VBA, np. nie można zadeklarować zmiennej o nazwie *Me*, ponieważ jest to słowo kluczowe reprezentujące aktywny formant.

# Deklaracje zmiennych/stałych

Przed użyciem zmienna powinna być zadeklarowana. Jeśli zmienna nie jest zadeklarowana, to jej domyślnym typem jest typ Variant. **Uwaga na marnotrawstwo pamięci i czasu obliczeń**!

Deklaracja zmiennej, niezależnie od typu przechowywanych danych, ma następującą składnię:

**Dim | Private | Public** *NazwaZmiennej* [**As** TypDanych]

Deklarację stałej przeprowadza się wg poniższego schematu:

**[Private | Public] Const** *NazwaStałej* [**As** TypDanych] = wartość

# Deklaracje zmiennych/stałych

- Zmienna zadeklarowana przez **Dim|Private** w module jest dostępna dla wszystkich procedur modułu i tylko w module, w którym została zadeklarowana.
- Zmienna zadeklarowana przez **Dim** w procedurze jest dostępna tylko wewnątrz procedury.
- Zmienna zadeklarowana jako **Public** jest dostępna we wszystkich procedurach i we wszystkich modułach w aplikacji, chyba że zostało użyte **Option Private Module**, które powoduje, że zmienne są publiczne tylko wewnątrz projektu, w którym rezydują.
- Deklaracja **Option Explicit** wymusza definiowanie wszystkich zmiennych.

# Deklaracje zmiennych/stałych

**Dim** *Numer* **As** *Byte*

deklaracja zmiennej typu bajtowego

**Public** *Numer1* **As** *Integer*

deklaracja zmiennej typu całkowitoliczbowego, ważna we wszystkich modułach, o ile podana na początku modułu

**Private** *Numer2* **As** *Double*

deklaracja zmiennej typu rzeczywistego, ważna w module, na początku którego została podana

**Dim** *Cena* **As** *Single*

deklaracja zmiennej o nazwie Cena, typ - liczby rzeczywiste pojedynczej precyzji

**Const** *KONIECROKU* **As** *Date* = #12/31/2020#

deklaracja stałej o nazwie KONIECROKU (typ data) z wartością 31 grudnia 2020

***Const*** *NAZWISKO* ***As*** String = "Kowalski"

deklaracja stałej o nazwie NAZWISKO (typ – ciąg znaków)

# Deklaracje tablic

**Tablica** jest zbiorem zmiennych tego samego typu danych. Jest ona stosowana do zgrupowania razem zmiennych zależnych. Na przykład, można stworzyć tablicę 100 liczb całkowitych w celu przechowywania numerów identyfikacyjnych NR_ID urzędników jakiegoś przedsiębiorstwa, zamiast tworzenia 100 niezależnych zmiennych.

**Każdy element tablicy jest zmienną z własnym numerem/ami indeksu**; przeprowadzenie zmian w jednym elemencie tablicy nie wpływa na inne elementy.

# Deklaracja tablic

**Dim | Private | Public** *NazwaTablicy*(indeksy) [**As** TypDanych]

**Dim** Sprzedaż(5)

**Dim** NrRachunku(700 **To** 799) **As** Integer

**Dim** Współrz(4, 1 **To** 3) **As** Byte

**Dim** Współrz(0 **To** 4, 1 **To** 3) **As** Byte

Do elementu tablicy odwołujemy się podając nazwę tablicy i indeks umieszczony w nawiasach:

Sprzedaż(0), Sprzedaż(1).....Sprzedaż(5)

NrRachunku(700).....NrRachunku(799)

Numer początkowy indeksu tablic możemy zmienić z 0 na 1 stosując instrukcję **Option Base 1**. Instrukcje tę należy wpisać na początku modułu w sekcji deklaracji, czyli przed pierwszą występującą w nim procedurą.

# Deklaracja tablic

**Dim** Współrz(4, 1 **To** 3) **As Byte**

Do elementu tablicy odwołujemy się podając nazwę tablicy i indeks umieszczony w nawiasach:

Współrz(0,1), Współrz(0,2) , Współrz(0,3)

Współrz(1,1), Współrz(1,2) , Współrz(1,3)

Współrz(2,1), Współrz(2,2) , Współrz(2,3)

Współrz(3,1), Współrz(3,2) , Współrz(3,3)

Współrz(4,1), Współrz(4,2) , Współrz(4,3)

# Dynamiczna deklaracja tablic

**Dim | Private | Public** *NazwaTablicy*() [**As** TypDanych]

**ReDim** [**Preserve**] *NazwaTablicy*(indeksy) [**As** TypDanych]

```
Sub DeklDyn()
Dim n As Byte
Dim l As Byte
Dim MojaTabl() As Integer    ' Deklaracja tablicy dynamicznej.
n = InputBox("Podaj rozmiar tablicy")
ReDim MojaTabl(n)        ' Określenie rozmiaru tablicy.
l = 1
10: MojaTabl(l) = InputBox("Podaj " & Str(l) & "-szy element wektora")
If l < n Then
   l = l + 1
   GoTo 10
End If
ReDim Preserve MojaTabl(20)  ' Zmień rozmiar z zachowaniem wcześniej
     wprowadzonych wartości.
End Sub
```

# Typy użytkownika

Niezależnie od wbudowanych typów danych można tworzyć struktury danych będące kombinacją kilku istniejących już typów.

```
Type Klient
    Nr_ewid As Long
    Nazwisko As String
    Adres As String
    Miasto As String
    Kod As String
    Telefon As String
End Type

Dim MoiKlienci(1000) As Klient
```

# Typy użytkownika

- Dostęp do elementów struktury następuje przez podanie nazwy struktury, po niej kropki, a następnie - nazwy konkretnego elementu:

  MoiKlienci (10).Nr_ewid = 345216

  MoiKlienci (10).Nazwisko =„Jan Kowalski”

  MoiKlienci (10).Adres =„ ul. Gwiazdy 4”

  MoiKlienci (10).Miasto =„Kraków”

  MoiKlienci (10).Kod =„ 01-773”

  MoiKlienci (10).Telefon =„ 12-34-567”

# VBA - wyrażenia

- ***Wyrażeniem*** nazywamy regułę obliczania, w wyniku której otrzymujemy pewną wartość. Wyrażenie składa się z sensownej kombinacji ***argumentów*** i ***operatorów***. Wyrażenie nie stanowi samodzielnej instrukcji języka VBA - jest jedynie jej częścią składową.
- W zależności od rodzaju operatorów i argumentów wyróżnia się:
  - wyrażenia arytmetyczne, `a+b`, `a*(b-d)`, `sin(x)/z`
  - wyrażenia logiczne, `not v`
  - wyrażenia porównania, `(a<b) and (b>c)`
  - wyrażenia konkatenacji. `"Ala "&"ma"`

# VBA - wyrażenia

- Operatory dzielimy na:
  - jednoargumentowe,
  - dwuargumentowe.
- Jeżeli w wyrażeniu występuje więcej niż jeden operator, to kolejność wykonywania działań jest wyznaczona przez następujące zasady:
  - w pierwszej kolejności wykonywane są działania w najbardziej wewnętrznych nawiasach,
  - najpierw wykonywane są operacje o wyższym priorytecie,
  - ciąg operatorów o równym priorytecie wykonywany jest od strony lewej do prawej.

# Wyrażenia arytmetyczne

- Wyrażenie arytmetyczne stosuje się do obliczania wartości liczbowych za pomocą operacji arytmetycznych. W wyrażeniach tego typu argumenty połączone są operatorami arytmetycznymi. Każde wyrażenie arytm. składa się z elementarnych wyrażeń postaci:

  *operator_jednoargumentowy* argument

  argument1 *operator_dwuargumentowy* argument2

# Operatory arytmetyczne

| Operator | Operacja |
|---|---|
| + (jednoarg.) | identyczność |
| - (jednoarg.) | zmiana znaku |
| ^ | potęgowanie |
| * | mnożenie |
| / | dzielenie |
| \ | dzielenie całkowite |
| mod | reszta z dzielenia |
| + | dodawanie |
| - | odejmowanie |

# Operatory arytmetyczne

| Wykonywana operacja | Symbol operatora | Przykład | Wynik działania (zwracana wartość) |
|---|---|---|---|
| Dodawanie | + | 2+5 | 7 |
| Odejmowanie | - | 10-8 | 2 |
| Mnożenie | * | 2*3 | 6 |
| Dzielenie | / | 10/4 | 2,5 |
| Dzielenie całkowite | \ | 10\4 | 2 |
| Reszta z dzielenia | Mod | 10 mod 3 | 1 |
| Potęgowanie | ^ | 3^2 | 9 |
| Zmiana znaku liczby | - | 5 | -5 |

# Operatory arytmetyczne

| Zapis matematyczny | Zapis w języku VBA |
|---|---|
| $ab + \frac{c}{d}$ | a*b+c/d |
| $\frac{a*b+c}{d}$ | (a*b+c)/d |
| *-a+b* | jak po lewej |
| *-(a+b)* | jak po lewej |
| *abc* | a*b*c |
| $\frac{a}{b}(c+d)$ | a/b*(c+d) lub a*(c+d)/b |
| $\frac{a}{b}c+d$ | a/b*c+d lub a*c/b+d |
| $\frac{a+b}{c+\frac{b}{a-b}}$ | (a+b)/(c+b/(a-b)) |

# Wyrażenia logiczne

- Operatory logiczne służą do wykonywania operacji logicznych na argumentach typu *Boolean*

  **not** argument

  argument1 ***operator_dwuargumentowy*** argument2

| Operator | Operacja |
|---|---|
| **not** (jednoarg.) | negacja |
| **and** | koniunkcja |
| **or** | alternatywa |
| **xor** | różnica symetryczna |

# Wyrażenia logiczne

| | | | | |
|---|---|---|---|---|
| x | true | false | true | false |
| y | false | false | true | true |
| **not** x | false | true | false | true |
| **not** y | true | true | false | false |
| x **and** y | false | false | true | false |
| x **or** y | true | false | true | true |
| x **xor** y | true | false | false | true |

# Wyrażenia porównania

- Wyrażenia porównania (relacje) dotyczą dowolnych argumentów dowolnych typów o typach zgodnych. Wynikiem działania operatorów porównania jest wartość logiczna – *prawda* lub *fałsz* (*true* lub *false*).

  argument1 ***operator_porównania*** argument2

| **Nazwa operatora** | **Symbol** | **Przykład** | **Wynik działania** |
|---|---|---|---|
| Nierówność | < > | 2 < > 1 | Prawda |
| | | 2 < > 2 | Fałsz |
| Równość | = | 3 = 3 | Prawda |
| | | 3 = 5 | Fałsz |
| Większe | > | 4 > 0 | Prawda |
| | | 4 > 6 | Fałsz |

# Wyrażenia porównania

| Nazwa operatora | Symbol | Przykład | Wynik działania |
|---|---|---|---|
| Większe lub równe | > = | 5 >=5 | Prawda |
| | | 5 >=1 | Prawda |
| | | 5 >= 8 | Fałsz |
| Mniejsze | < | 6 < 8 | Prawda |
| | | 6 < 1 | Fałsz |
| Mniejsze lub równe | <= | 7 <= 7 | Prawda |
| | | 7 <= 8 | Prawda |
| | | 7 <= 1 | Fałsz |
| Porównanie łańcuchów | Like | “AGH” like “AGH” | Prawda |
| | | „Agh” like „AGH” | Fałsz |
| | | 1 like 1 | Prawda |
| | | 1 like 2 | Fałsz |

# Wyrażenia konkatenacji

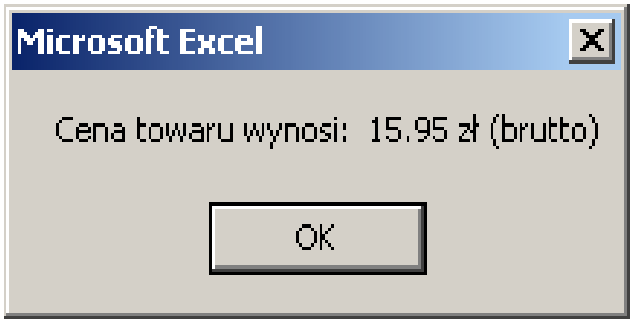
Operatory znakowe wykorzystuje się do łączenia łańcuchów znaków. Podstawowym operatorem znakowym jest **&** (ampersand). Operator ten umożliwia przeprowadzenie konkatenacji dwóch lub większej liczby ciągów znakowych, jak i ciągów znakowych z wartością przechowywaną przez zmienną lub stałą.

```
Sub Przyklad()

Dim Napis As String
Dim Cena As Single
Dim Informacja As String

Napis = "Cena towaru wynosi: "
Cena = 15.95
Informacja = Napis & Str(Cena) & " zł (brutto)"
MsgBox Informacja

End Sub
```



# Priorytety operatorów

Jeżeli w wyrażeniu wystąpi kilka **operatorów**, każda część jest obliczana i sprawdzana w oparciu o kolejność **priorytetów operatorów**.

Jeżeli wyrażenie zawiera operatory należące do różnych kategorii, najpierw wykonywane są działania operatorów arytmetycznych, potem operatorów porównania i operatorów logicznych.

Wszystkie operatory porównania mają taki sam priorytet i wykonywane są w takim porządku, w jakim znajdują się w wyrażeniu, w kolejności od lewej do prawej.

# Priorytety operatorów

| **Arytmetyczne** | **Logiczne** |
|---|---|
| Potęgowanie (^) | Not |
| Negacja (–) | And |
| Mnożenie i dzielenie (*, /) | Or |
| Dzielenie całkowite (\) | Xor |
| Reszta z dzielenia (Mod) | |
| Dodawanie i odejmowanie (+, –) | |
| Łączenie ciągów (&) | |