Technologie Informacyjne - Energetyka

Laboratorium 1

*Uruchomienie edytora VBA. Procedura i instrukcja wyjścia. Sposoby uruchamiania makr. Operacje arytmetyczne. Zmienne ich deklarowanie i wymuszanie deklarowania. Typy zmiennych, zmienne tekstowe. Wprowadzenie do funkcji i sposobu ich edycji.*

Język VBA pozwala nam przede wszystkim na automatyzację wielu czynności, czy też funkcji w programach pakietu Office. Możemy dodawać makra, przyciski odpowiadające za przypisane funkcje, automatyzować procesy, manipulować danymi, integrować pliki z bazami danych MS Access, automatyzować zadania w MS PowerPoint i wiele więcej.

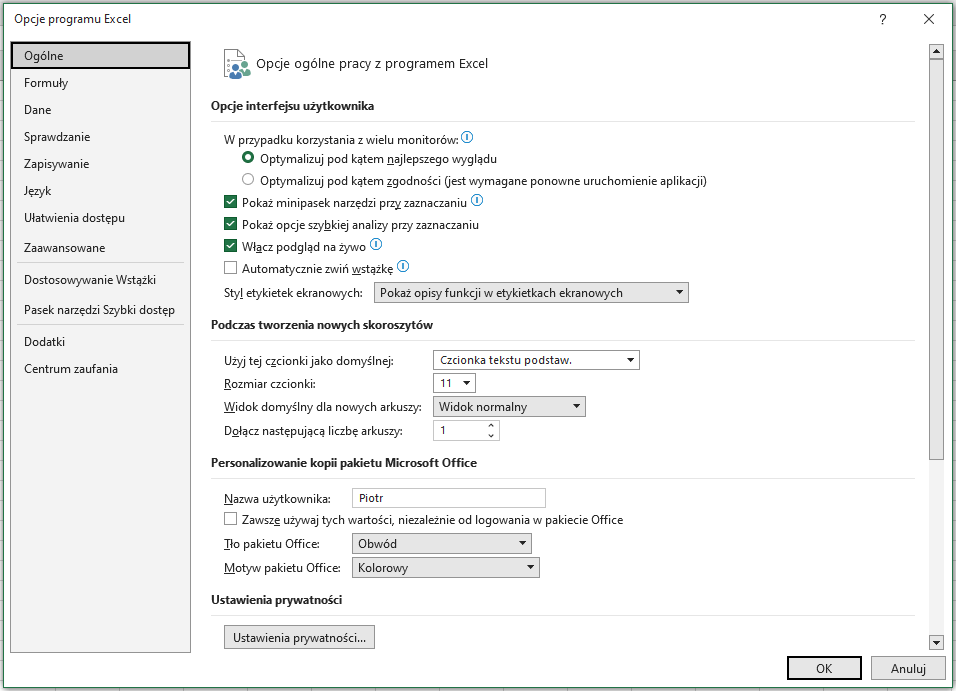
**1. Uruchomienie edytora VBA, Procedura i Instrukcja Wyjścia, Sposoby Uruchamiania Makr**

**Uruchomienie Edytora VBA**

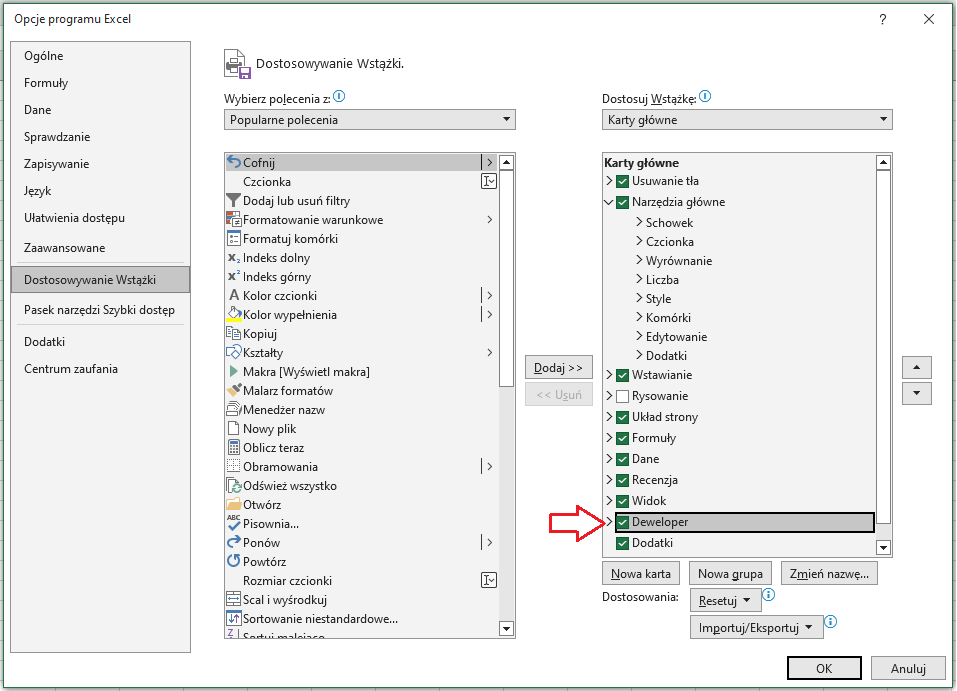
Edytor VBA (Visual Basic for Applications) można uruchomić bezpośrednio z programów pakietu Microsoft Office, takich jak Excel. Ta opcja musi jednak zostać najpierw uruchomiona w opcjach pakietu MS Office. Aby to zrobić należy kolejno:

* Otworzyć program MS Excel.
* Kliknij zakładkę **Plik** w lewym górnym rogu, następnie z listy wybierz **Opcje**

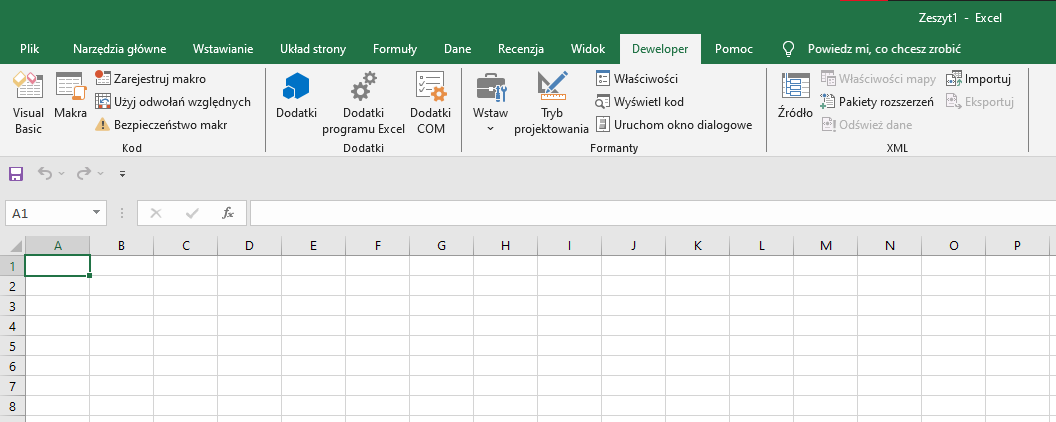
Wyskoczy następujące okienko :



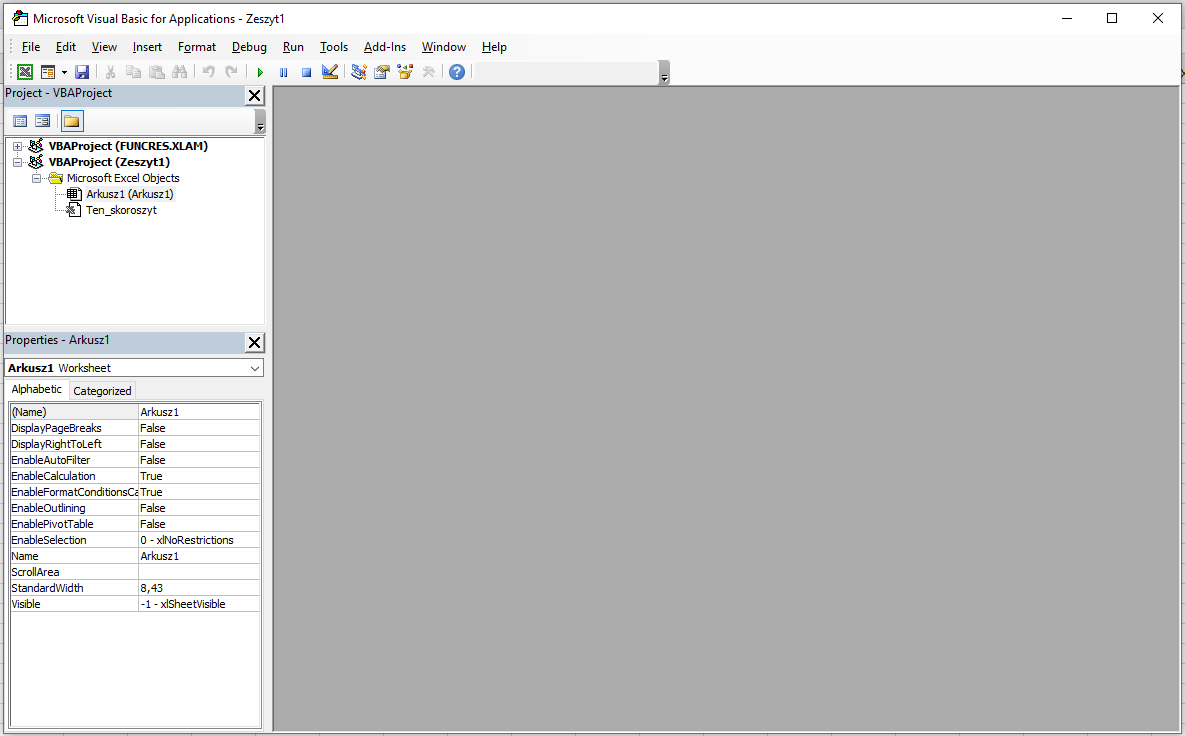
* Wybierz opcje **Dostosowywanie Wstążki**
* Po prawej stronie, wybierz z listy **Karty Główne**, oraz zaznacz opcję **Deweloper**



* Zapisz zmiany naciskając przycisk **Ok**
* Wybierz z listy nową zakładkę Deweloper, a następnie Visual Basic, aby otworzyć edytor VBA.



Widok uruchomionego edytora VBA prezentuje się następująco :



Warto również zauważyć, że pomimo ustawionej Polskiej wersji językowej, domyślnie edytor języka VBA operuje w języku Angielskim, jak również jego procedury oraz polecenia.

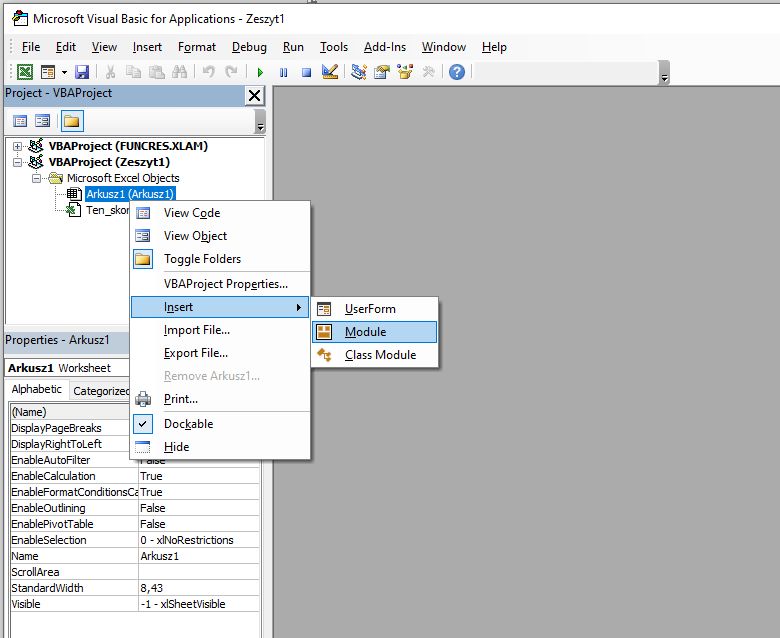
**Procedura i Instrukcja Wyjścia**

**Procedura** to zbiór instrukcji, które można wykonać jako jedną jednostkę pracy. W VBA mamy procedury **Sub** oraz **Function**.

* **Sub** - używane do wykonywania czynności, które nie zwracają wartości.
* **Function** - używane do wykonywania czynności, które zwracają wartość.

Aby móc zacząć pracować w edytorze, dodawać procedury dla naszego arkusza, należy wprowadzić **Moduł**. W tym celu :

* Naciśnij prawym przyciskiem myszy na otworzony arkusz z prawej listy oraz wybierz opcję **Insert**
* Wybierz wprowadzenie **Module**



W tym momencie możemy przystąpić do pisania naszego kodu. Wstawienie modułu pomaga nam również zachować nasz kod w przypadku, kiedy został by usunięty dany arkusz z pliku Excel.

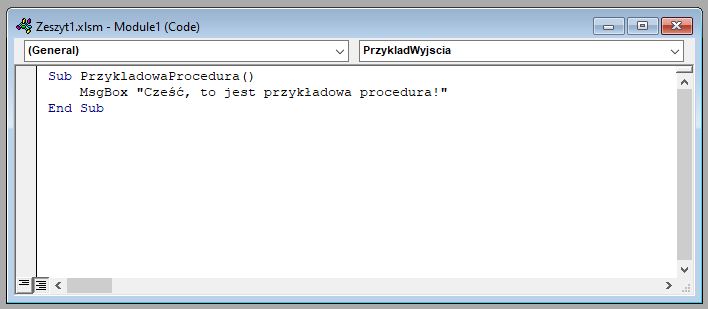
Przetestuj obie procedury na poniższych przykładach (aby uruchomić procedurę naciśnij zieloną strzałkę Run, bądź wciśnij klawisz F5) :

**Przykład procedury Sub :**

Sub PrzykladowaProcedura()

MsgBox "Cześć, to jest przykładowa procedura!"

End Sub

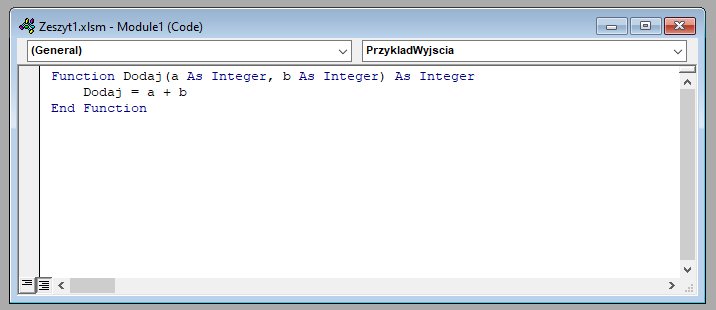


**Przykład procedury Function :**

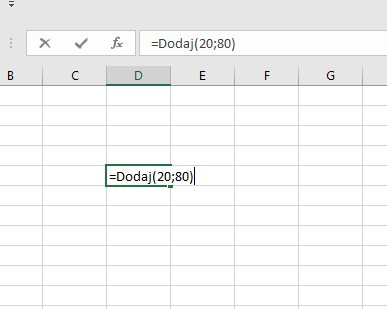
Function Dodaj(a As Integer, b As Integer) As Integer

Dodaj = a + b

End Function



W przykładzie procedury Function po jej wpisaniu przejdź na okno arkusza. Następnie w dowolnym pustym polu wpisz użycie funkcji : =Dodaj(20;80)



Naciśnij Enter. W tym momencie powinniśmy otrzymać wynik z dodawania – liczbę 100. Możesz sprawdzić również dodawanie na innych liczbach.

**Instrukcja wyjścia**: Exit Sub lub Exit Function używane są do natychmiastowego wyjścia z procedury.

**Przykład instrukcji wyjścia :**

Sub PrzykladWyjscia()

On Error GoTo ErrorHandler

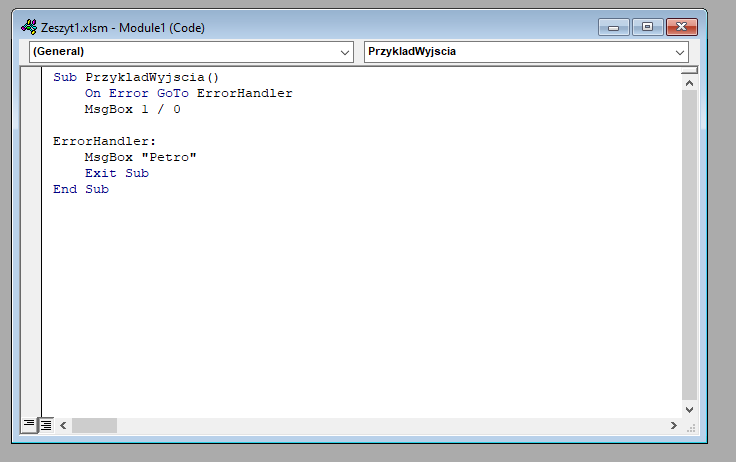
MsgBox 1 / 0

ErrorHandler:

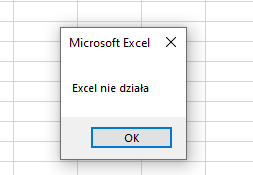
MsgBox "Wystąpił błąd!"

Exit Sub

End Sub



Uruchom instrukcję. W tym momencie zgodnie z wpisanym w pole tekstem MsgBox otrzymamy informację o błędzie „Wystąpił błąd!”. Zamień w instrukcji tekst „Wystąpił błąd!” na „Excel nie działa”. Po uruchomieniu uzyskamy inny tekst z informacji o błędzie.



**Sposoby Uruchamiania Makr**

* Bezpośrednio z edytora VBA: Umieść kursor wewnątrz procedury i naciśnij F5.
* Przypisanie do przycisku lub innego obiektu na arkuszu Excela.
* Używając skrótów klawiaturowych, które można przypisać makru.

**2. Operacje Arytmetyczne, Zmienne, Ich Deklarowanie i Wymuszanie Deklarowania**

Podstawowe operacje arytmetyczne to : +, -, \*, /.

**Przykład operacji :**

Sub OperacjeArytmetyczne()

Dim a As Integer, b As Integer, suma As Integer

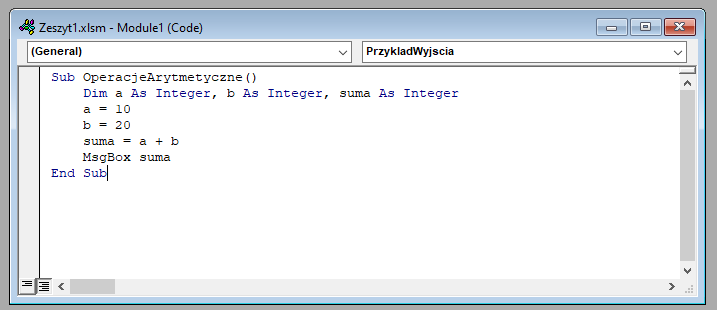
a = 10

b = 20

suma = a + b

MsgBox suma

End Sub



W tym przykładzie uzyskaliśmy wynik jedynie dodawania podanych liczb. Co jeśli byśmy chcieli przetestować wszystkie operacje? Możemy w tym celu wykorzystać poniższe makro, które wyświetli nam po kolej wszystkie wyniki każdego z operacji arytmetycznych :

Sub RozneOperacje()

Dim a As Integer: a = 10

Dim b As Integer: b = 20

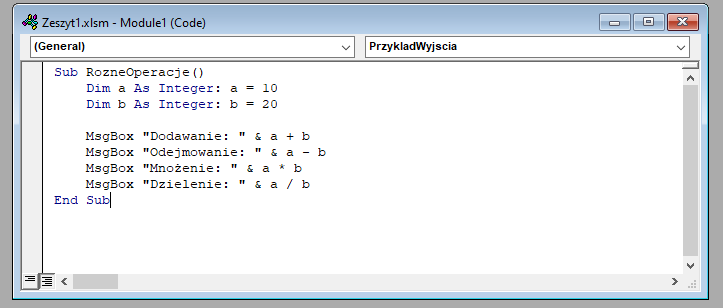
MsgBox "Dodawanie: " & a + b

MsgBox "Odejmowanie: " & a - b

MsgBox "Mnożenie: " & a \* b

MsgBox "Dzielenie: " & a / b

End Sub



**Zmienne i Ich Deklarowanie**

Zmienną deklarujemy, używając słowa Dim, a następnie nazwy zmiennej i typu.

**Przykład operacji wykorzystania deklaracji zmiennych :**

Sub DeklaracjaZmiennych()

Dim imie As String, wiek As Integer, wzrost As Double

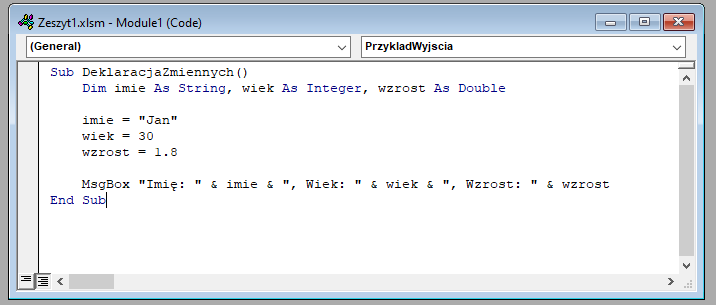
imie = "Jan"

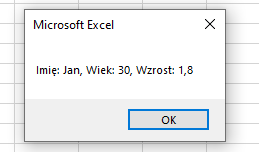
wiek = 30

wzrost = 1.8

MsgBox "Imię: " & imie & ", Wiek: " & wiek & ", Wzrost: " & wzrost

End Sub





#### Wymuszanie Deklarowania Zmiennych - Aby wymusić deklarowanie zmiennych, na początku modułu należy umieścić Option Explicit.

**Przykład :**

Option Explicit

Sub Przyklad()

Dim a As Integer

a = 10

MsgBox a

End Sub

**3. Typy Zmiennych, Zmienne Tekstowe, Wprowadzenie do Funkcji i Sposobu Ich Edycji**

Podstawowe typy zmiennych w VBA to:

* Integer
* String
* Boolean
* Double
* Variant (domyślny typ, jeśli nie określono inaczej)

**Zmienne Tekstowe**

Zmienne tekstowe używane są do przechowywania ciągów znaków. Napisy, imiona, dane do faktury itp. Możemy z powodzeniem przechowywać jako ciąg znaków. Dlaczego więc nie używać ich do wszystkich danych? Przede wszystkim na tym typie danych nie możemy wykonywać operacji arytmetycznych, czy też innych funkcji związanych z operacjami na liczbach, jak wyliczanie średniej.

**Deklarowanie zmiennej tekstowej zapisujemy w przykładowy sposób:**

Dim napis As String

napis = "To jest tekst"

MsgBox napis

**Concatenation (Łączenie Tekstów)**

Zmienne tekstowe możemy łączyć za pomocą operatora &. W ów czas połączymy teksty ze sobą.

**Przykład łączenia zmiennych tekstowych:**

Sub LaczenieTekstu()

Dim czesc1 As String, czesc2 As String, calosc As String

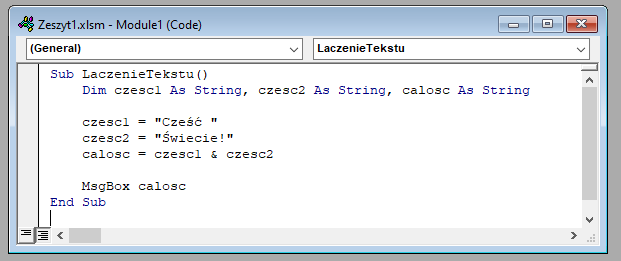
czesc1 = "Cześć "

czesc2 = "Świecie!"

calosc = czesc1 & czesc2

MsgBox calosc

End Sub



**Funkcje i Ich Edycja**

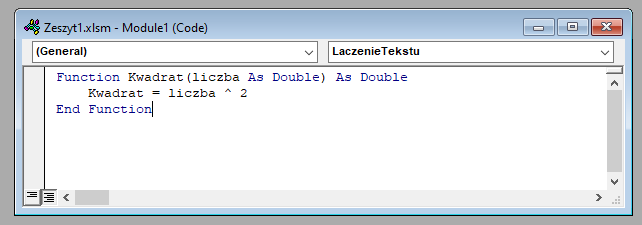
Funkcje jak wspominaliśmy są procedurami, które zwracają wartość.

**Przykład funkcji, która zwraca wartość w postaci podanej liczby podniesionej do drugiej potęgi:**

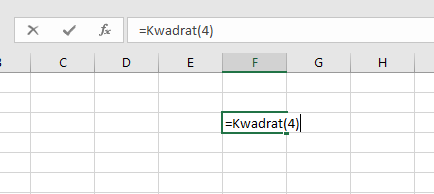
Function Kwadrat(liczba As Double) As Double

Kwadrat = liczba ^ 2

End Function

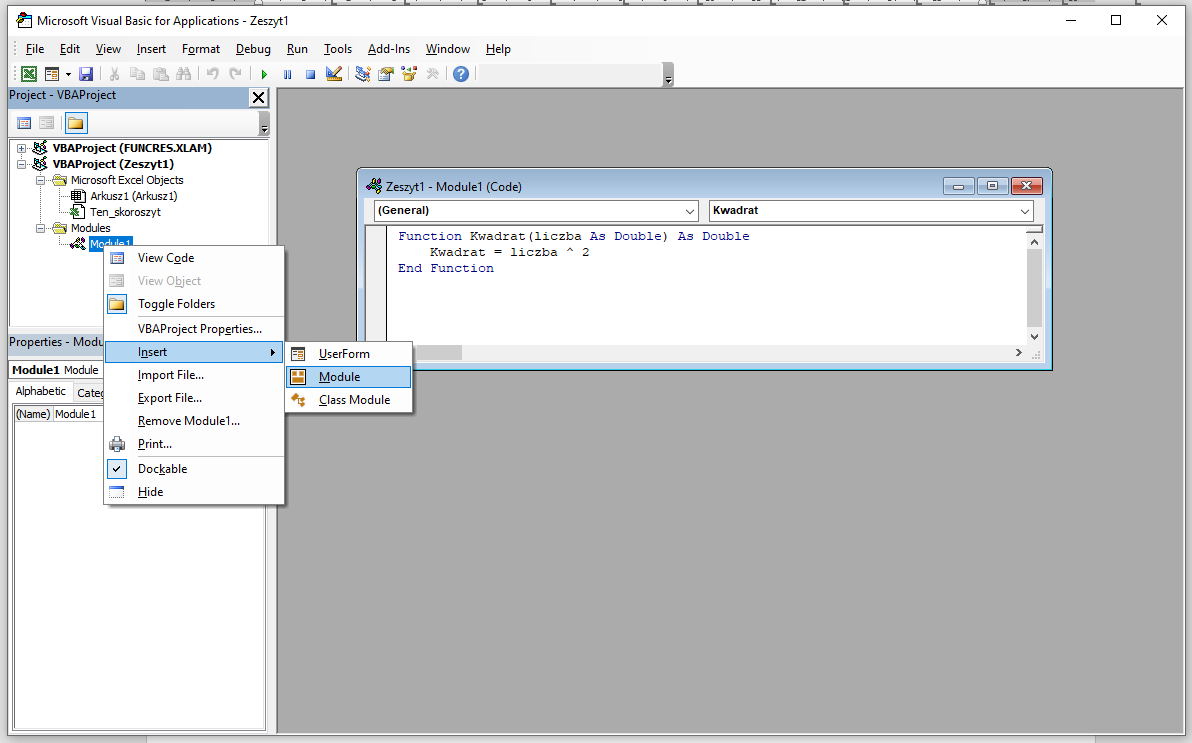


W przykładzie procedury Function po jej wpisaniu przejdź na okno arkusza. Następnie w dowolnym pustym polu wpisz użycie funkcji : =Kwadrat(4)



Po naciśnięciu klawisza Enter uzyskamy wynik w postaci liczby 16.

A co jeśli potrzebujemy korzystać z dwóch podobnych funkcji – podnoszenia liczby do kwadratu oraz do sześcianu na przemian? W tym celu pozostawmy nasz moduł jak w ostatnim przykładzie. Dodajmy nowy moduł do naszego arkuszu.



Następnie wpiszmy ponownie ostatni przykład do nowego modułu. Powinniśmy również go zmienić. W przeciwnym wypadku będziemy mieć przypisane 2 identyczne funkcje.

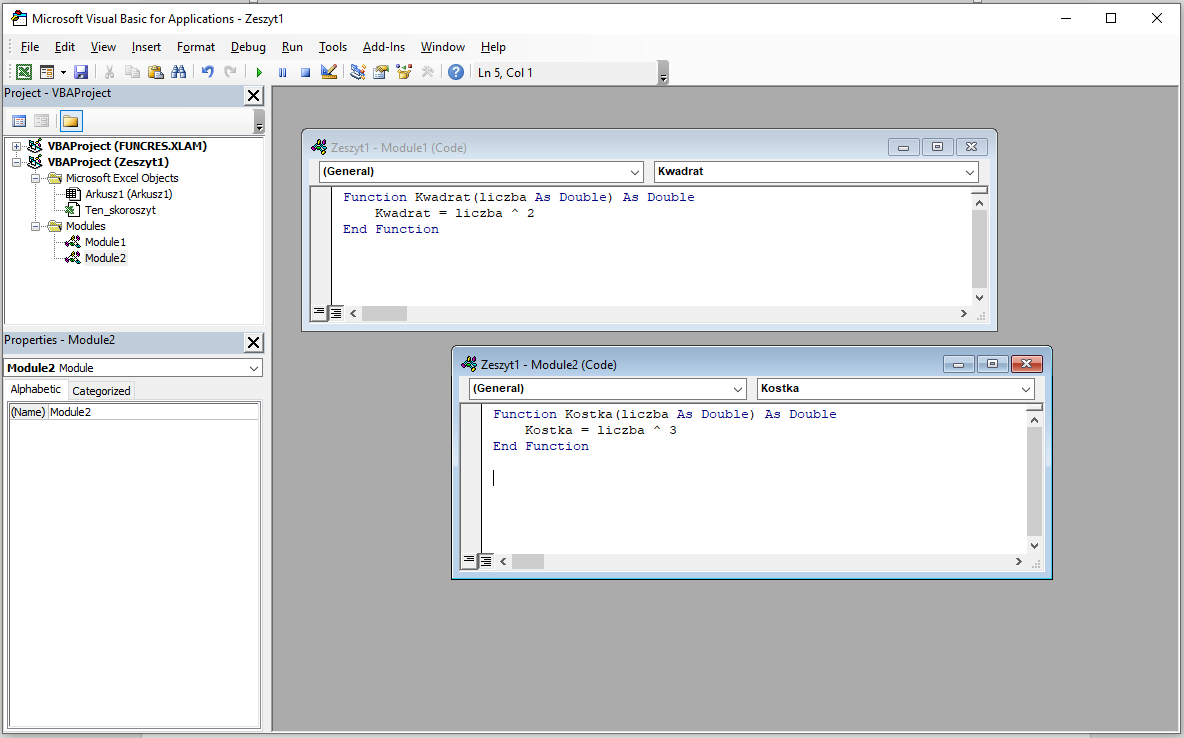
**Przykład funkcji podnoszenia liczby do 3 potęgi wyglądać może następująco :**

Function Kostka(liczba As Double) As Double

Kostka = liczba ^ 3

End Function

Oba moduły powinny wyglądać tak :



Na koniec przetestuj na dwóch polach w arkuszu kalkulacyjnym komendy wykorzystujące nasze dodane funkcje :

=Kwadrat(4)

Obliczy nam wynik liczby 4 podniesionej do 2 potęgi. Dam na wynik 16.

=Kostka(4)

Obliczy nam wynik liczby 4 podniesionej do 3 potęgi. Dam na wynik 64.