ZACHODNIOPOMORSKI UNIWERSYTET TECHNOLOGICZNY
INSTYTUT TECHNOLOGII MECHANICZNEJ

PROGRAMOWANIE W EXCELU W JĘZYKU

VISUAL BASIC FOR APPLICATIONS

wersja 2 zmieniona

mgr Tadeusz Ziębakowski

Szczecin, 2013

Wstęp

Opracowanie zawiera zestaw ćwiczeń przeznaczonych do zajęć laboratoryjnych z przedmiotów *Podstawy informatyki i algorytmizacji* oraz *Informatyka II* realizowanych na I roku studiów. Celem tych ćwiczeń jest wprowadzenie studenta w podstawowe zagadnienia algorytmizacji i programowania, głównie w zakresie programowania strukturalnego. Jako narzędzie programistyczne wybrano język *Visual Basic for Applications*, który jako język skryptowy może być wykorzystywany do tworzenia zarówno prostych makr jak i rozbudowanych aplikacji. Wszystkie ćwiczenia osadzone są w strukturze arkusza kalkulacyjnego, który oferuje łatwy dostęp do danych i jest zarazem źródłem ciekawych problemów programistycznych. Niezbędne do ćwiczeń pliki Excela z danymi, można pobrać ze strony internetowej autora http://tezet.zut.edu.pl lub z zasobów serwera dydaktycznego. Ponieważ zakłada się, że student na wykładzie otrzymuje wiedzę niezbędną do pracy na zajęciach laboratoryjnych, opis ćwiczeń nie został wzbogacony o szerszy materiał pojęciowy. Podstawowe informacje na temat języka *Visual Basic for Applications* można jednak znaleźć w dodatku zamieszczonym na końcu opracowania.

Zawartość:

Ć W I C Z E N I E 1

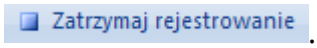
Zagadnienia:

- Nagrywanie makr i ich edycja w edytorze VBA.
- Podstawowe elementy języka Visual Basic for Applications: zmienne i ich typy, instrukcje deklaracji i podstawienia.
- Dostęp do danych arkusza kalkulacyjnego za pomocą obiektu Range.
- Instrukcja warunkowa ***If.***
- Procedura ***MsgBox.***

1. Nagrywanie makra

Zadanie 1. **Rejestrowanie makra.**

W nowym arkuszu wykonaj następujące czynności:

1. Na karcie **Deweloper** wybierz polecenie **Zarejestruj makro** w grupie **Kod**.
2. Zmień nazwę makra na. „Adam” i wybierz klawisz skrótu na **Ctrl-a** i potwierdź OK – rozpoczął się tryb nagrywania makra.
3. W wybranej komórce wpisz napis „Adam”, potwierdź wpisany napis myszką  na pasku formuły.
4. Zakończ nagrywanie makra przyciskiem 

.
5. Wybierz nową komórkę i naciśnij **Ctrl-a**.
6. Przejdź do edytora VBA za pomocą kombinacji klawiszy **Alt+F11** i kliknij na znak **+** przy **Modules** i dwukrotnie na **Module1:**

```
Sub Adam()
'
'  Adam Makro
'
'  Klawisz skrótu: Ctrl+a
'
   ActiveCell.FormulaR1C1 = "Adam"
End Sub
```

Napisy na zielono zaczynające się od apostrofu to komentarz

7. Zmień napis „Adam” na „Adam i Ewa” i ponownie uruchom makro w nowej wybranej komórce arkusza.

2. Obiekt *Range*

Zadanie 2

Wpisz w edytorze VBA w **Module1** makro:

```
Sub Dodaj()
      Range("C1")= Range("A1")+ Range("B1")
End Sub
```

W Arkuszu wpisz do komórek A1 i B1 dowolne wartości liczbowe i wykonaj makro.

Uwaga. W VBA istnieje również prostsza forma zapisu zmiennej związanej z komórką arkusza kalkulacyjnego. Zamiast np. `Range("C1")` można też stosować zapis `[C1]`, zatem powyższe makro można zapisać nastepująco:

```
Sub Dodaj1()
      [C1]= [A1]+ [B1]
End Sub
```

Ostrzeżenie. W VBA można utworzyć zmienną o nazwie np. `A1`. Jeśli jest taka zmienna to VBA nie interpretuje zapisu `[A1]` jako odwołania do komórki `A1`. Aby odwołać się do komórki `A1` należy wówczas zastosować zapis `Range("A1").`

Zadanie 3

Należy obliczyć wartość wyrażenia:

$$\frac{1-\sqrt{a^2-ab+b^2}}{1+\sqrt{a^2+ab+b^2}} \qquad (*)$$

dla wartości *a,b* znajdujących się w komórkach odpowiednio A1 i B1. W tym celu napisz następujące makro:

```
Sub Wynik()
      Dim a As Double ,b As Double ,c As Double
      a = [A1]
      b = [B1]
      c =
      [C1] = c
End Sub
```

tu należy wypisać wyrażenie zmiennych *a* i *b* obliczające (*), skonstruowane na zasadach podobnych do pisania formuł EXCELA

Po wpisaniu wartości do A1 i B1 i uruchomieniu makra, w komórce C1 pojawi wynik.
Przykładowe wyniki przedstawia tabela poniżej.

a	b	c
1	1	0
1	-1	-0,366025
0	0	1

Zadanie 4

Napisz makro, które zamienia wartości dwóch komórek, np. A2 i B2. (Wskazówka: wprowadź dodatkową zmienną typu **`Variant`** w celu zapamiętania wartości jednej z komórek).

3. Instrukcja warunkowa If ...Then...Else...i procedura wewnętrzna MsgBox

Dzielenie dwóch liczb często wymaga sprawdzenia czy dzielnik jest różny od 0. Takiego sprawdzenia można dokonać za pomocą instrukcji warunkowej.

Zadanie 5

Napisz makro:

```
Sub Sprawdź()
  If [B3] <> 0 Then [C3] = [A3]/[B3] Else [C3]= "Błąd"
End Sub
```

Wstawiaj do komórek A3 i B3 różne wartości w tym również 0 i uruchamiaj makro. Następnie przepisz makro w alternatywnej postaci i sprawdź jego działanie:

```
Sub Sprawdź1()
  If [B3] <> 0 Then
      [C3] = [A3]/[B3]
  Else
      [C3]= "Błąd dzielenia przez zero"
  End If
End Sub
```

Następnie zamiast instrukcji `[C3]= "Błąd dzielenia przez zero"` wpisz:

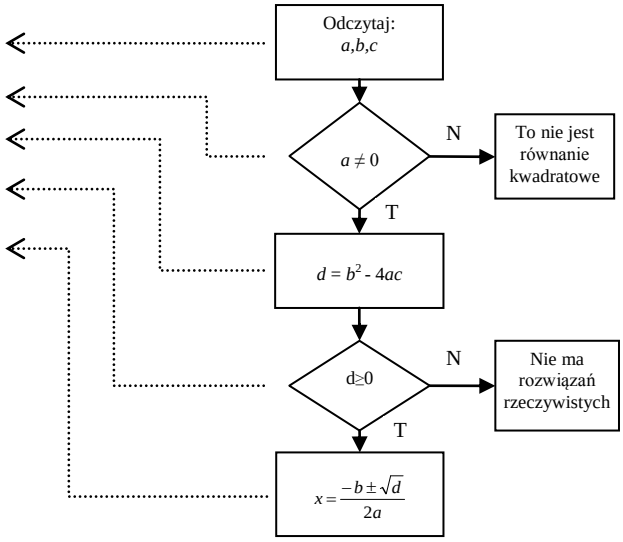
```
[C3]= ""
MsgBox "Błąd dzielenia przez zero"
```

Sprawdź działanie makra.

Zadanie 6

Uzupełnij zarys makra przedstawionego poniżej, którego zadaniem jest rozwiązanie równania kwadratowego $ax^2+bx+c=0$ w liczbach rzeczywistych Wykorzystaj schemat blokowy algorytmu znajdującego się obok. Współczynniki *a,b,c* wpisz do komórek A1,B1,C1. Pierwiastki, jeśli są, makro powinno umieścić w komórkach D1 i E1. Dodatkowe komunikaty można wpisać do wybranej komórki lub wyświetlić poleceniem **`MsgBox`**.

```
Sub równanie_kwadratowe()
Dim a As Single, b As Single, c As Single, d As Single
a = [A1]
b = [B1]
c = [C1]
If ..... Then
   .....
   .....
   If ..... Then
       [D1]= .....
       [E1]= .....
   Else
       .....
       .....
   End If
Else
   .....
   .....
End If
End Sub
```



Przetestuj działanie makra na przykładowych danych:

a	*b*	*c*	wynik
1	1	-2	$x_1 = 1$ lub $x_2 = -2$
0	1	1	To nie jest równanie kwadratowe
1	0	1	Nie ma rozwiązań rzeczywistych

Zadania domowe

Zaprojektuj makro, które rozwiązuje układ dwóch równań liniowych z dwoma niewiadomymi:

$$\begin{cases} a_1 x + b_1 y = c_1 \\ a_2 x + b_2 y = c_2 \end{cases}$$

Współczynniki a_i , b_i , c_i można wpisać do zakresu A1:C2. Makro powinno zbadać kiedy mamy do czynienia z układem:
- oznaczonym i wtedy makro powinno wypisać rozwiązania,
- nieoznaczonym,
- sprzecznym.

Poza tym makro powinno zbadać pewne „złośliwe" przypadki, gdy współczynniki a_i , b_i , c_i nie wyznaczają układu równań z dwoma niewiadomymi. Makro można przetestować na następujących danych:

1	1	2
1	-1	0

układ oznaczony
$x = 1$ oraz $y = 1$

1	1	1
1	1	1

układ nieoznaczony

1	1	1
1	1	-1

układ sprzeczny

0	1	1
0	1	-1

to nie jest układ równań z dwoma niewiadomymi

Wskazówki:

Jeśli $a_1 = a_2 = 0$ lub/i $b_1 = b_2 = 0$ to nie mamy dwóch niewiadomych.

Przyjmijmy $D = a_1 b_2 - b_1 a_2$, $D_x = b_2 c_1 - b_1 c_2$, $D_y = -a_2 c_1 + a_1 c_2$. Z algebry wiadomo, że mogą zachodzić trzy podstawowe przypadki:

a) jeśli $D \neq 0$ to mamy jedno rozwiązanie $x = D_x / D$ i $y = D_y / D$.

b) Jeśli $D = 0$, $D_x = 0$ *i* $D_y = 0$ to mamy nieskończenie wiele rozwiązań: dla dowolnego x, $y = (c_i - a_i x)/b_i$ spełnia układ równań (zakładamy że przynajmniej jedno $b_i \neq 0$)

c) Jeśli $D = 0$ oraz $D_x \neq 0$ lub $D_y \neq 0$ to układ nie ma rozwiązań.

Ć W I C Z E N I E 2

Zagadnienia:

- Obiekt ***ActiveCell***
- Funkcje VBA - tworzenie i użycie w formułach arkusza kalkulacyjnego
- Wywoływanie funkcji VBA z poziomu innych funkcji VBA i makr

1. Obiekt ActiveCell

W nowym arkuszu Excela uruchom edytor VBA i wstaw nowy moduł:
Insert → Module

Zadanie 1

Wpisz w edytorze VBA makro:

```
Sub Dodaj_1()
ActiveCell = ActiveCell + 1
End Sub
```

Każde uruchomienie makra zwiększy wartość aktywnej (wybranej) komórki o 1 .

Zadanie 2

W edytorze VBA wpisz makro:

```
Sub Przykład_ActiveCell()
    ActiveCell = "środek"
    ActiveCell(2) = "w dół"
    ActiveCell(0) = "w górę"
    ActiveCell(1, 2) = "w prawo"
    ActiveCell(1, 0) = "w lewo"
End Sub
```

W arkuszu kalkulacyjnym wybierz komórkę aktywną, wokół której są puste komórki i uruchom makro. Powinien się pokazać następujący obraz:

	w górę	
w lewo	środek	w prawo
	w dół	

Uwaga: Zapis `ActiveCell(k)` jest równoważny zapisowi `ActiveCell(k,1)`. `ActiveCell(1)` jest równoważny zapisowi `ActiveCell` . Wszystkie komórki z tej samej kolumny co `ActiveCell` można wyrazić za pomocą `ActiveCell(k)` dla $k \neq 1$.

Zadanie 3

Wpisz w edytorze VBA poniższe makro, a następnie wybierz komórkę aktywną i kilka razy uruchom to makro.

```
Sub W_prawo()
    ActiveCell = "W prawo"
    ActiveCell(1, 2).Select
End Sub
```

Zmodyfikuj powyższe makro w taki sposób, aby startując z komórki pustej kolejne uruchomienia makra wypisywały zamiast napisu „W prawo" kolejne liczby całkowite.

Zadanie 4

Zaprojektuj dwa podobne makra jak w zadaniu 3, które wypisują kolejne liczby całkowite: pierwsze makro - pionowo w dół, drugie - na ukos w dół i prawo.

Zadanie 5

Należy pobrać arkusz **równanie kwadratowe.xlsm** z odpowiedniej lokalizacji sieciowej**.** Widok tego arkusza pokazany jest na rysunku poniżej.

	A	B	C	D	E	F	G
1	a	b	c	delta	x1	x2	
2	0,5	0	1,5	-3	Nie ma rozwiązan w liczbach rzeczywistych		
3	-3	1,5	1	14,25	-0,38	0,88	
4	3,5	-3	3,5	-40	Nie ma rozwiązan w liczbach rzeczywistych		
5	-1	-1	2	9	-2,00	1,00	
6	0,5	2	-3,5	11	1,32	-5,32	
7	0	0	-3,5	0	To nie jest równanie kwadratowe		
8	-4,5	-0,5	0	0,25	-0,11	0,00	
9	-4	-4	2	48	-1,37	0,37	
10	2,5	-3	3,5	-26	Nie ma rozwiązan w liczbach rzeczywistych		
11	0,5	-2	0,5	3	3,73	0,27	
12	2	-4	0,5	12	1,87	0,13	
13	-0,5	-3	-2,5	4	-5,00	-1,00	

Należy zaprojektować makro, które po ustawieniu aktywnej komórki w kolumnie A pobierze z tego samego wiersza wartości współczynników równania kwadratowego z kolumn A, B, C, obliczy deltę i pierwiastki równania kwadratowego podobnie jak w zadaniu 6 z poprzedniego ćwiczenia i wstawi je również do komórek z tego samego wiersza w kolumnach odpowiednio D, E, F. Po wykonaniu tych czynności makro powinno ustawić aktywną komórkę o jeden wiersz w dół. Zaczynając od komórki A2 kolejne uruchomienia tego makra powinny wypełnić wynikami powyższy arkusz.

2. Funkcje VBA

Zadanie 6

Zadanie to należy wykonać w drugim arkuszu (kopii arkusza 1) skoroszytu **równanie kwadratowe.xlsm.**

1. W edytorze VB, w nowym module napisz następującą funkcję VBA:

```
Function delta1(a As Single, b As Single, c As Single) As Single
   delta1 = b ^ 2 - 4 * a * c
End Function
```

W arkuszu kalkulacyjnym w komórce D2 napisz formułę wykorzystującą funkcję delta z argumentami A2,B2,C2 i przeciągnij ją aż do wiersza 30. Uwaga: Jedynka w nazwie funkcji `dellta1` jest wprowadzona w celu uniknięcia konfliktu z istniejącą w EXCEL'u funkcją DELTA.

MACIERZ.ODW =pierwiastek1(A2;B2;C2)

	A	B	C	D	E	F	G
1	a	b	c	delta	x1	x2	
2	0,5	0	1,5	-3	=pierwiastek1(A2;B2;C2)		
3	-3	1,5	1	14,25	-0,38	0,88	
4	3,5	-3	3,5	-40	Nie ma rozwiązan w liczbach rzeczywistych		
5	-1	-1	2	9	-2,00	1,00	
6	0,5	2	-3,5	11	1,32	-5,32	
7	0	0	-3,5	0	To nie jest równanie kwadratowe		
8	-4,5	-0,5	0	0,25	-0,11	0,00	
9	-4	-4	2	48	-1,37	0,37	
10	2,5	-3	3,5	-26	Nie ma rozwiązan w liczbach rzeczywistych		
11	0,5	-2	0,5	3	3,73	0,27	
12	2	-4	0,5	12	1,87	0,13	
13	-0,5	-3	-2,5	4	-5,00	-1,00	

2. Zaprojektuj funkcje **pierwiastek1** i **pierwiastek2** o nagłówkach:

```
Function pierwiastek1(a As Single, b As Single, c As Single)
Function pierwiastek2(a As Single, b As Single, c As Single)
```

obliczające w liczbach rzeczywistych pierwiastki równania kwadratowego o współczynnikach *a,b,c*. Podobnie jak w punkcie 1 wywołaj te funkcje za pomocą odpowiednich formuł tak, aby w kolumnach E i F pojawiły się wartości tych pierwiastków. W przypadku braku rozwiązań w liczbach rzeczywistych lub w przypadku zerowania się czynnika kwadratowego równania, pierwsza z tych funkcji powinna zasygnalizować taką sytuację odpowiednim napisem, tak jak to widać na powyższym rysunku. Spróbuj wykorzystać wewnątrz jednej z tych funkcji napisaną wcześniej funkcję `delta`.

3. Zaprojektuj makro w działaniu podobne do makra z zadania 5, które posłuży do wypełnienia drugiej kopii arkusza ***równanie kwadratowe***. W makrze tym oprócz zmiennej **ActiveCell** należy wykorzystać wcześniej napisane funkcje **delta**, **pierwiastek1** i **pierwiastek2**. Po ustawieniu komórki A2 jako aktywnej makro powinno pobrać wartości współczynników równania kwadratowego z komórek A2, B2, C2 i wpisać deltę i pierwiastki równania kwadratowego do komórek odpowiednio D2, E2, F2. Po wykonaniu tych czynności makro powinno ustawić komórkę A3 jako aktywną. Kolejne uruchomienia makra powinny generować wyniki w kolejnych wierszach arkusza. Wyniki należy umieścić w nowej (trzeciej) kopii arkusza z danymi znajdującej się w pobranym pliku.

Zadanie domowe

1. Zaprojektuj funkcje, które rozwiązują układ dwóch równań liniowych z dwoma niewiadomymi:

$$\begin{cases} a_1 x + b_1 y = c_1 \\ a_2 x + b_2 y = c_2 \end{cases}$$

Przetestuj te funkcje na danych podanych w zadaniu domowym ćwiczenia 1.

2. Wyznacz kolejne sumy częściowe szeregu geometrycznego $\sum_{n=1}^{\infty} q^{n-1}$ w kolejnych komórkach arkusza dla $q = 0{,}5$. Wskazówka: Szereg ten jest zbieżny gdy |q| < 1. Każda suma częściowa jest postaci $S_n = 1 + q + q^2 + \ldots + q^{n-1}$. Sumy te spełniają związek rekurencyjny $S_{n+1} = qS_n + 1$, który można wykorzystać w tym zadaniu.

Ć W I C Z E N I E 3

Zagadnienia:

- Funkcje wewnętrzne VB
- Instrukcja wyboru `Select Case`
- Operacje na łańcuchach – operator „&" oraz funkcje wewnętrzne VB operujące na łańcuchach: `Len, Left, Right, Mid, InStr, Repalce`.

Zadanie 1 - arkusz *DODATKI DO PENSJI*

Należy pobrać arkusz **dodatki do pensji.xlsm** z odpowiedniej lokalizacji sieciowej.

	A	B	C	D	E	F	G	H	I	J	K
1	WYSŁUGA LAT i DODATEK SZKODLIWY										
2											
3	L.p.	Nazwisko	Imię	Stanowisko	Pensja	Data zatrudnienia	Lata pracy	Lata pracy 1	Wysługa	Wysługa 1	Dodatek szkodliwy
4	1	Arendarska	Anna	księgowa	3 200 zł	2009-03-01	?	?	?	?	?
5	2	Bednarski	Bartosz	magazynier	2 300 zł	1995-02-01	?	?	?	?	?
6	3	Celiński	Cezary	spawacz	3 500 zł	2004-06-01	?	?	?	?	?
7	4	Doliński	Damian	praktykant	1 750 zł	2012-07-01	?	?	?	?	?
8	5	Ejsmond	Edmund	ślusarz	2 200 zł	2004-08-01	?	?	?	?	?
9	6	Flajterski	Filip	malarz	2 500 zł	1992-09-01	?	?	?	?	?
10	7	Gocławski	Grzegorz	dyrektor	5 900 zł	1991-10-01	?	?	?	?	?
11	8	Holenderska	Hanna	sprzątaczka	1 900 zł	1992-11-01	?	?	?	?	?
12	9	Izdebski	Ignacy	spawacz	3 800 zł	1992-12-01	?	?	?	?	?
13	10	Jowalski	Jan	malarz	2 950 zł	1998-01-01	?	?	?	?	?

Jednym z dodatków do wynagrodzenia jakie otrzymują pracownicy w Polsce z tytułu przepracowanych lat jest tzw. ***wysługa***. W zakresie od 5 do 20 lat jest ona równa w procentach liczbie pełnych przepracowanych lat. Na przykład pracownik, który przepracował 7 lat otrzymuje dodatkowo 7% pensji zasadniczej. Pracownicy, którzy przepracowali mniej niż 5 pełnych lat nie otrzymują wysługi, a pracownicy, którzy przepracowali 20 i więcej pełnych lat otrzymują tylko 20% wysługi.

Polecenia do wykonania.

1. W edytorze VB, w nowym module napisz funkcję VBA:

```
Function Lata_pracy(data As Date) As Integer
   Lata_pracy = Year(Now()) - Year(data)
End Function
```

zwracającą wartość ilości przepracowanych lat obliczanych na podstawie różnicy lat pomiędzy rokiem bieżącym a rokiem, w którym pracownik został zatrudniony. Zostały tutaj wykorzystane funkcje Visual Basica **`Year(…)`** oraz **`Now()`**. W arkuszu kalkulacyjnym użyj funkcji **`lata_pracy`** w odpowiedniej formule umieszczonej w kolumnie o nagłówku ***lata pracy***.

Uwaga: Sposób liczenia przepracowanych lat pracy zaproponowany powyżej jest niedokładny. Gdyby na przykład pracownik został zatrudniony w ostatnim dniu ubiegłego roku, to odejmując lata dostajemy 1, natomiast w rzeczywistości nie przepracował on do dnia dzisiejszego pełnego roku.

2. Napisz funkcję VBA o podobnym nagłówku lecz nazwie **`lata_pracy1`**, która zwraca wartość ilości faktycznie przepracowanych lat obliczanych za pomocą funkcji **`WorksheetFunction.YearFrac(…)`**. Zapoznaj się w pomocy Visual Basic z opisem tej funkcji. Użyj funkcji **`Int`** do odrzucenia części ułamkowej wartości zwracanej przez funkcję **`YearFrac`**. W arkuszu kalkulacyjnym w kolumnie o nagłówku ***lata pracy*** zamień wywołanie funkcji **`lata_pracy`** na **`lata_pracy1`**.

3. Napisz funkcję VBA o nagłówku:

   ```
   Function Wysługa(lata As Integer, pensja As Currency) As Currency
   ```

   której wynikiem jest wartość wysługi lat obliczonej wg formuły z treści zadania. W tym celu wykorzystaj składnię instrukcji **`If`** zastosowaną w przykładowej funkcji **`Wartość1`** zamieszczonej w Dodatku na str.41. Zastosuj funkcję **`Wysługa`** w formule umieszczonej w kolumnie o nagłówku ***wysługa.*** Jako jeden argumentów przyjmij wartości znajdujące się w kolumnie ***lata pracy.***

4. Napisz funkcję VBA o nagłówku:

   ```
   Function Wysługa1(data As Date, pensja As Currency) As Currency
   ```

   wyznaczającą podobnie jak poprzednia wysługę lat. Wywołaj funkcję `lata_pracy1` wewnątrz tej funkcji, w celu wyznaczenia lat pracy na podstawie argumentu `data`. Ponadto zastosuj zamiast instrukcji **`If`** instrukcję **`Select Case`** wzorując się na przykładzie funkcji **`Wartość2`** zamieszczonej w Dodatku na str. 41. Wywołaj tę funkcję w arkuszu w nowej kolumnie o nagłówku ***wysługa1***.

5. Zaprojektuj funkcję VBA o nazwie **`Dodatek_szkodliwy`**, która wyznaczy wartość dodatku szkodliwego zgodnie z tabelą:

stanowisko	lata pracy	dodatek szkodliwy
spawacz, malarz, ślusarz	<10	500 zł
spawacz, malarz, ślusarz	>=10	1000 zł
pozostali	dowolna liczba	0 zł

6. Zaprojektuj makro, które posłuży do wypełnienia tabeli nr 3 w arkuszu ***dodatki do pensji***. Działanie tego makra powinno być podobne do makra opisanego w ćwiczeniu 2 zadania 6.3. Oprócz zmiennej **`ActiveCell`** należy wykorzystać wcześniej napisane funkcje **`Wysługa`** i **`Dodatek_szkodliwy`**.

7. Napisz i następnie zastosuj w tabeli nr 4 funkcje VBA, które na podstawie kolumny ***Nazwisko i imię*** z tabeli nr 3 wyznaczają:
   a) płeć (wypisuje napisy mężczyzna albo kobieta),
   b) inicjały,
   c) imię i nazwisko (w odwrotnej kolejności niż podane w tabeli nr 3).

Zadanie domowe

Dopisz w kolumnie ***Nazwisko i imię*** w tabeli nr 3 do niektórych nazwisk drugie imię. Przeprojektuj funkcje z zadania 7,b,c tak aby uwzględniały występowanie drugiego imienia.

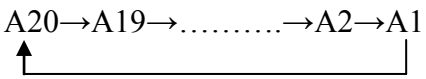
Ć W I C Z E N I E 4

Zagadnienia:

- Pętla **For**

Zadanie 1

Należy zaprojektować makro, które zamienia cyklicznie wartości w komórkach od A1 do A20 wg następującego schematu:



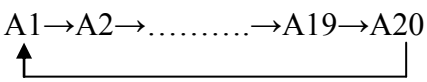
Przepisz w edytorze VBA projekt makra poniżej, zastępując instrukcje ujęte w nawiasy klamrowe instrukcją pętli **For**.

```
Sub zamień()
Dim z As Range, x As Variant, k As Integer
Set z = [a1:a20]
x = z(1)
z(1) = z(2)
z(2) = z(3)
'
'
'
z(19) = z(20)
z(20) = x
End Sub
```

```
For k = … to …
…
Next k
```

Zadanie 2

Napisz makro, które zamienia cyklicznie wartości w komórkach od A1 do A20 wg następującego schematu:



Wskazówka: Należy użyć instrukcji **For** z frazą **Step**:

```
For k = … to … Step …
```

Zadanie 3

Napisz makro, które zamienia wartości w komórkach od A1 do A20 wg następującego schematu:

A1 ↔A20, A2 ↔A19, A3 ↔A18 itd. aż do A10 ↔A11

Zmodyfikuj makro w taki sposób, aby zamieniało na podobnej zasadzie wartości w komórkach zakresu zaznaczonego w Excelu, tzn. pierwszą z ostatnią, druga z przedostatnią itd. Związanie zmiennej x typu `Range` z wybranym zakresem komórek w arkuszu realizuje się instrukcjami:

```
Dim z As Range
Set z = Selection
```

Ilość komórek w zakresie określa wyrażenie `z.Count`.

Zadanie 4

Za pomocą formuły `=sin(wiersz())` wypełnij komórki w zakresie D1:D30. Następnie w edytorze VBA:

a) napisz makro, które koloruje na czerwono wartości dodatnie a na niebiesko wartości ujemne (wskazówka: `[D1].Interior.ColorIndex = 3` wprowadza w komórce [D1] czerwony kolor tła zaś `[D1].Interior.ColorIndex = 5` niebieski);
b) napisz makro, które zlicza wartości ujemne i wartości dodatnie;
c) wyznacz za pomocą makra wartość maksymalną/minimalną.

Zadanie domowe

Oblicz przybliżoną wartość liczby π z dokładnością do 0.0001 wykorzystując szereg:

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \ldots$$

Wypisz również kolejne sumy częściowe tego szeregu w wybranej kolumnie.

Ć W I C Z E N I E 5

Zagadnienia:

- Arkusz jako tablica 2-wymiarowa.
- Tablice i funkcje tablicowe

Zadanie 1

W nowym arkuszu kalkulacyjnym, w edytorze VBA napisz i uruchom następujące makro, które wpisuje do komórek ich adres w formacie zbliżonym do R1C1:

```
Sub wypisz()
Dim z As Range, i As Integer, j As Integer
Set z = [a1]
For i = 1 To 20
    For j = 1 To 10
        z(i, j) = "k" & j & "w" & i
    Next j
Next i
End Sub
```

Zadanie 2

Wzorując się na przykładzie powyżej napisz makro **wypisz2** , które wpisuje do komórki jej nazwę w formacie: oznaczenie literowe kolumny + numer wiersza np. do komórki C2 wpisuje napis „C2”. W tym celu wprowadź tablicę 1-wymiarową za pomocą instrukcji:

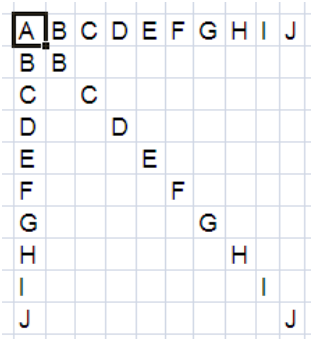
```
Dim t(1 To 10) As String
t(1) = "A": t(2) = "B": t(3) = "C": t(4) = "D": t(5) = "E"
t(6) = "F": t(7) = "G": t(8) = "H": t(9) = "I": t(10) = "J"
```

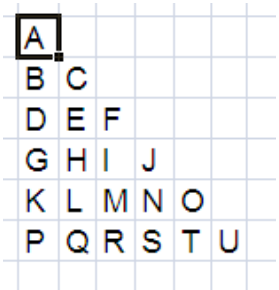
Uwaga: Zamiast wypisywać instrukcje typu `t(…) = "…"` można napisać:

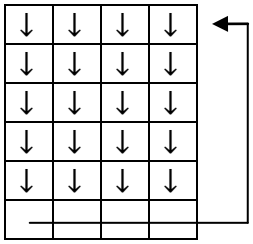
```
For i = 1 To 10          'Funkcja Chr podaje znak w kodzie ASCII
    t(i) = Chr(64 + i)   'Litera A jest na 65 miejscu w tabeli
Next i                   'kodów ASCII, litera B na 66 itd.
```

Zadanie 3

Napisz makra, które począwszy od komórki aktywnej wypisują kolejne litery alfabetu tak jak to pokazują rysunki poniżej.







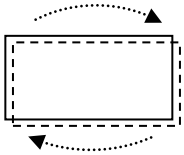
Zadanie 4

Napisz makro, które przesuwa cyklicznie **wiersze** wg schematu przedstawionego obok w zakresie A1:J20. Przetestuj je w zakresie z komórkami wypełnionymi przez makro z zadania 2.

Zadanie 5

Napisz makro, które przesuwa cyklicznie **kolumny** wg schematu podobnego jak w zadaniu 4 w zakresie A1:J20 z komórkami wypełnionymi przez makro z zadania 2.

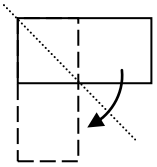
Zadanie 6



Przepisz i przetestuj na dowolnym niepustym prostokątnym zakresie w arkuszu kalkulacyjnym funkcję tablicową wypisaną poniżej, która obraca dowolny prostokątny zakres o 180 stopni (do góry nogami):

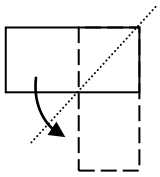
```
Function obrot(z1 As Range) As Variant
Dim z2() As Variant, nc As Integer
Dim nr As Integer, i As Integer, j As Integer
nc = z1.Columns.Count   'Wyznaczenie ilości kolumn
nr = z1.Rows.Count      'Wyznaczenie ilości wierszy
ReDim z2(1 To nr, 1 To nc)    'zmiana rozmiaru tablicy z2
For i = 1 To nr
    For j = 1 To nc
        z2(i, j) = z1(nr + 1 - i, nc + 1 - j)
    Next j
Next i
obrot = z2
End Function
```

Zadanie 7



Napisz funkcje tablicowe, które transponują macierz prostokątną względem jednej lub drugiej przekątnej.

Zadanie domowe:



Napisz makro, które wypisuje tzw. trójkąt Pascala, którego 10 pierwszych wierszy przedstawia rysunek poniżej. Zasada tworzenia tego trójkąta jest następująca: począwszy od liczby 1 wpisanej w górnym wierzchołku trójkąta kolejne elementy są sumą 2 elementów znajdujących się nad i w lewo do góry na ukos (por. rysunek), przy założeniu, że na zewnątrz trójkąta są puste komórki, które mają wartość 0. Dobrze napisane makro nie powinno zawierać więcej niż 10 linijek kodu.

Trójkąt Pascala

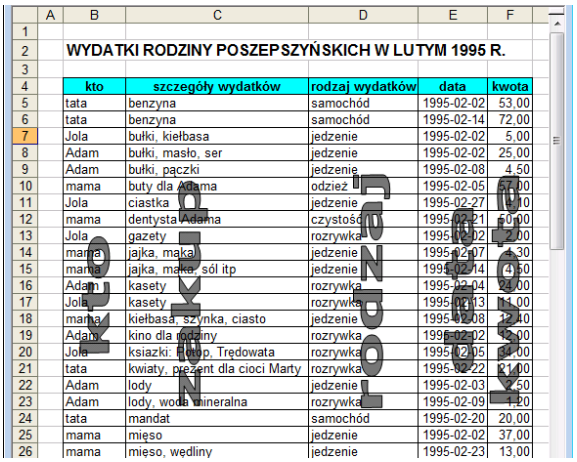
1									
1	1								
1	2	1							
1	3	3	1						
1	4	6	4	1					
1	5	10	10	5	1				
1	6	15	20	15	6	1			
1	7	21	35	35	21	7	1		
1	8	28	56	70	56	28	8	1	
1	9	36	84	126	126	84	36	9	1

Ć W I C Z E N I E 6

Zagadnienia:

- Pętla `For` w połączeniu z instrukcją `If`

Ćwiczenie to dotyczy arkusza **rodzina.xlsm**, który należy pobrać z odpowiedniej lokalizacji sieciowej. W arkuszu tym należy nadać nazwy zakresom komórek zgodnie z rysunkiem poniżej (należy nazwać zakresy bez nagłówków). W drugiej części tego dokumentu podane są **przykłady makr i funkcji**, które mogą być pomocne w realizacji ćwiczenia. Przykłady te można skopiować do edytora VBA (w nowym module), odpowiednio sformatować, a następnie makra należy uruchomić, zaś przykładową funkcję użyć w odpowiedniej formule.



WYDATKI RODZINY POSZEPSZYŃSKICH W LUTYM 1995 R.

kto	szczegóły wydatków	rodzaj wydatków	data	kwota
tata	benzyna	samochód	1995-02-02	53,00
tata	benzyna	samochód	1995-02-14	72,00
Jola	bułki, kiełbasa	jedzenie	1995-02-02	5,00
Adam	bułki, masło, ser	jedzenie	1995-02-02	25,00
Adam	bułki, paczki	jedzenie	1995-02-08	4,50
mama	buty dla Adama	odzież	1995-02-05	80,00
Jola	ciastka	jedzenie	1995-02-27	9,10
mama	dentysta Adama	czystość	1995-02-21	50,00
Jola	gazety	rozrywka	1995-02-02	2,00
mama	jajka, mąka	jedzenie	1995-02-07	4,30
mama	jajka, mąka, sól itp	jedzenie	1995-02-14	4,50
Adam	kasety	rozrywka	1995-02-04	24,00
Jola	kasety	rozrywka	1995-02-13	11,00
mama	kiełbasa, szynka, ciasto	jedzenie	1995-02-08	12,40
Adam	kino dla rodziny	rozrywka	1995-02-02	8,00
Jola	ksiazki: Filip, Trędowata	rozrywka	1995-02-05	34,00
tata	kwiaty, prezent dla cioci Marty	rozrywka	1995-02-22	21,00
Adam	lody	jedzenie	1995-02-03	4,50
Adam	lody, woda mineralna	rozrywka	1995-02-09	1,20
tata	mandat	samochód	1995-02-20	20,00
mama	mięso	jedzenie	1995-02-02	37,00
mama	mięso, wędliny	jedzenie	1995-02-23	13,00

Zadania

1. Napisać funkcję, która obliczy ile razy robiły zakupy dzieci.
2. Napisać makro, które obliczy ile pieniędzy wydali rodzice.
3. Napisać funkcję tablicową, która obliczy ilość oraz kwotę wydatków zarówno rodziców jak i dzieci (wywołanie funkcji w czterech komórkach).
4. Napisać funkcję, która obliczy ile pieniędzy wydano w określonym dniu (data jako parametr funkcji)
5. Napisać makro, które wypisze w wybranym obszarze osoby, które kupowały jajka.
6. Napisać funkcję tablicową, która wyznaczy daty dwóch dni: pierwszego, w którym wydano najwięcej pieniędzy i drugiego, w którym wydano najmniej pieniędzy, przy czym należy pominąć te dni, w których w ogóle nie wydawano pieniędzy.
   Wskazówka: należy wykorzystać funkcję napisaną w punkcie 4.
7. Napisać makro, które tworzy listę wydatków zrobionych w niedziele w miesiącu lutym. (por. help: funkcja Weekday)
8. Napisać funkcję, która obliczy ile pieniędzy ogółem wydała wskazana osoba.
9. Napisać funkcję, która obliczy ile pieniędzy ogółem wydano dla wskazanego rodzaju wydatków.
10. Za pomocą makra wykonać zestawienie, które podaje dla każdego dnia w miesiącu lutym 1995 r. liczbę wydatków i łączną kwotę wydatków.
11. Wypisać za pomocą makr lub/i funkcji wszystkie daty dni miesiąca lutego 1995 r., w których tata nie robił zakupów (nie wydawał pieniędzy).

12. Za pomocą funkcji tablicowej wykonać zestawienie wydatków każdej z osób w poszczególnych dniach tygodnia.

	mama	tata	Jola	Adam
niedziela	96,90 zł	- zł	34,00 zł	- zł
poniedziałek	- zł	31,00 zł	15,10 zł	7,10 zł
wtorek	58,80 zł	80,80 zł	78,00 zł	- zł
środa	12,40 zł	36,90 zł	20,75 zł	4,50 zł
czwartek	54,70 zł	248,00 zł	7,00 zł	53,20 zł
piątek	- zł	- zł	4,70 zł	2,50 zł
sobota	56,50 zł	52,75 zł	29,98 zł	30,50 zł

13. Za pomocą makra, żółtym kolorem zaznaczyć 3 najmniejsze kwoty wydatków. (tło komórki w kolorze żółtym)
14. Napisać makro, które wypisze w wybranym obszarze daty trzech dni, w których wydatki rodziny były największe, a obok kwoty tych wydatków.
15. Uogólnić makro z p. 14 na *k*- maksymalnych wartości. Zastosować procedurę dla $k = 5$ w zagadnieniu wypisania dat 5 dni, w których zanotowano największą ilość wydatków. (uwaga: jeśli *n* jest liczbą wszystkich wartości – elementów zbioru, to powinno być $k <= n$; gdy $k = n$ dostajemy procedurę sortowania).
16. Wypisz bez powtórzeń listę wszystkich rodzajów wydatków.

Przykłady.

1. Zliczanie elementów.

```
Function zlicz_wydatki(dzień As Date, daty As Range) As Integer
'
'Zlicza wydatki w określonym dniu
'
Dim i As Integer
zlicz_wydatki = 0
For i = 1 To daty.Count
If dzień = daty(i) Then zlicz_wydatki = zlicz_wydatki + 1
Next i
End Function
```

2. Sumowanie wartości liczbowych.

```
Sub suma_wydatków_rozrywka()
'
'Oblicza sumę wydatków z grupy rozrywka.
'
Dim kwota As Range, rodzaj As Range, i As Integer, suma As Currency
Set kwota = Range("kwota")
Set rodzaj = Range("rodzaj")
suma = 0
For i = 1 To rodzaj.Count
    If rodzaj(i) = "rozrywka" Then suma = suma + kwota(i)
Next i
MsgBox "suma wydatków z grupy rozrywka wynosi " & suma & " zł"
End Sub
```

3. Zliczanie i sumowanie razem

```
Function zlicz_sumuj_wydatki(dzień As Date, daty As Range, kwoty As
Range)
'
'Zlicza i sumuje wydatki dla określonej daty - funkcja tablicowa
'Wywołanie funkcji : należy zaznaczyć dwie komórki obok siebie,
'wprowadzić formułę z funkcją i nacisnąć Ctr-Shift-Enter
'
Dim zlicz_sumuj(1 To 2) As Variant, i As Integer
zlicz sumuj(1) = 0: zlicz_sumuj(2) = 0
For i = 1 To daty.Count
    If dzień = daty(i) Then
        zlicz_sumuj(1) = zlicz_sumuj(1) + 1          'zliczanie
        zlicz_sumuj(2) = zlicz_sumuj(2) + kwoty(i)   'sumowanie
    End If
Next i
zlicz_sumuj_wydatki = zlicz_sumuj
End Function
```

4. Obliczanie maksymalnej wartości.

```
Sub dzien_max()
'
'Podaje dzień, w którym zanotowano największy pojedynczy wydatek.
'
Dim kwota As Range, dzien As Date, data As Range, i As Integer
Dim max As Currency
Set kwota = Range("kwota")
Set data = Range("data")
For i = 1 To kwota.Count
    If kwota(i) > max Then
      max = kwota(i)
      dzien = data(i)
    End If
Next i
MsgBox "największy wydatek  " & max & " zł  był w dniu  " & dzien
End Sub
```

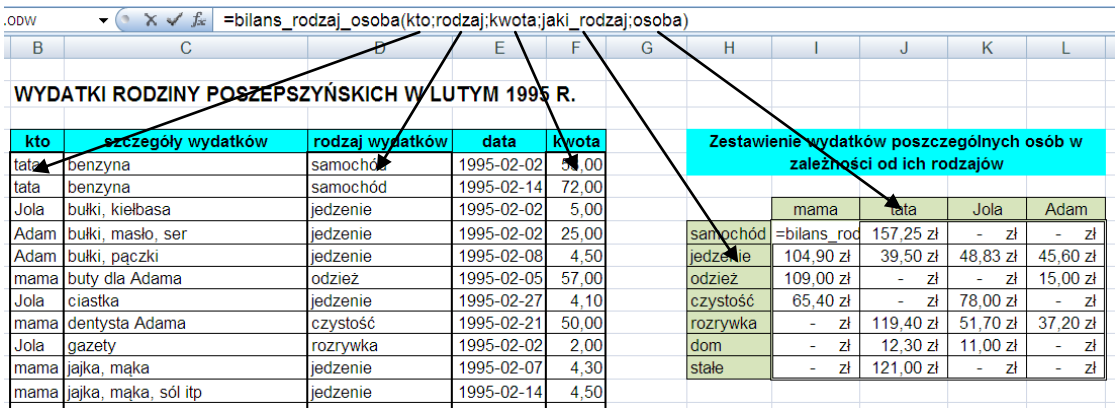
5. Tworzenie listy.

```
Sub lista_zakupów()
'
'Wypisuje listę zakupów rodziców.
'
Dim zakup As Range, kto As Range, lista As Range
Dim i As Integer, j As Integer
Set zakup = Range("zakup")
Set kto = Range("kto")
Set lista = Range("h10:h60")
j = 1
For i = 1 To kto.Count
    If kto(i) = "mama" Or kto(i) = "tata" Then
      lista(j) = zakup(i)
      j = j + 1
    End If
Next i
End Sub
```

6. Zaznaczanie komórek kolorem.

```
Sub jajka()
'
'Zaznacza na żółto zakupy jajek
'
Dim zakup As Range, i As Integer
Set zakup = Range("zakup")
For i = 1 To zakup.Count
 If InStr(zakup(i),"jajka") > 0 Then zakup(i).Interior.ColorIndex =
6
Next i
End Sub
```

7. Tworzenie zestawień tabelarycznych.



kto	szczegóły wydatków	rodzaj wydatków	data	kwota
tata	benzyna	samochód	1995-02-02	50,00
tata	benzyna	samochód	1995-02-14	72,00
Jola	bułki, kiełbasa	jedzenie	1995-02-02	5,00
Adam	bułki, masło, ser	jedzenie	1995-02-02	25,00
Adam	bułki, pączki	jedzenie	1995-02-08	4,50
mama	buty dla Adama	odzież	1995-02-05	57,00
Jola	ciastka	jedzenie	1995-02-27	4,10
mama	dentysta Adama	czystość	1995-02-21	50,00
Jola	gazety	rozrywka	1995-02-02	2,00
mama	jajka, mąka	jedzenie	1995-02-07	4,30
mama	jajka, mąka, sól itp	jedzenie	1995-02-14	4,50

	mama	tata	Jola	Adam
samochód	=bilans_roc	157,25 zł	- zł	- zł
jedzenie	104,90 zł	39,50 zł	48,83 zł	45,60 zł
odzież	109,00 zł	- zł	- zł	15,00 zł
czystość	65,40 zł	- zł	78,00 zł	- zł
rozrywka	- zł	119,40 zł	51,70 zł	37,20 zł
dom	- zł	12,30 zł	11,00 zł	- zł
stałe	- zł	121,00 zł	- zł	- zł

```
Function bilans_rodzaj_osoba(kto As Range, rodzaj As Range, _
            kwota As Range, jaki_rodzaj As Range, osoba As Range)

'Funcja tablicowa:
'bilans wydatków poszczególnych osób w zależności od ich rodzajów
'
Dim tb(), i As Integer, j As Integer, k As Integer
ReDim tb(1 To jaki_rodzaj.Count, 1 To osoba.Count)
For i = 1 To jaki_rodzaj.Count
    For j = 1 To osoba.Count
        For k = 1 To kto.Count
            If jaki_rodzaj(i) = rodzaj(k) And osoba(j) = kto(k) Then
                tb(i, j) = tb(i, j) + kwota(k)
            End If
        Next k
    Next j
Next i
bilans_rodzaj_osoba = tb()
End Function
```

Ć W I C Z E N I E 7

Zagadnienia:

- Generowanie tabeli danych na podstawie tabeli 2-wymiarowej.
- Wyszukiwanie i wypisywanie elementów określonych warunkiem za pomocą funkcji tablicowej VBA.

Ćwiczenie to dotyczy arkusza **sprzedaz herbaty.xlsm** przedstawionego poniżej.

SPRZEDAŻ HERBATY

		Cena 1 kg	29,10 zł	31,50 zł	35,10 zł	29,10 zł	60,90 zł	32,10 zł	27,00 zł	33,60 zł	29,40 zł	29,10 zł	30,30 zł	43,20 zł	33,30 zł	29,40 zł
		Nazwa herbaty	Assam	Black Lychee	Ceylon	Churmee	Darjeeling	Dragonwell	Gunpowder	Gyokuru	Jasmine	Keemun	Lapsang Souchong	Oolong	Pi Lo Chun	Russian Blend
Pracownik	Dąbrowski		55 kg	20 kg	30 kg	50 kg	115 kg	60 kg	35 kg	45 kg	10 kg	115 kg	35 kg	60 kg	15 kg	55 kg
	Górecki			10 kg	35 kg	15 kg	110 kg	20 kg	45 kg	15 kg	25 kg	50 kg	65 kg	60 kg	25 kg	90 kg
	Grabowski		70 kg		15 kg	50 kg	65 kg	35 kg	15 kg	40 kg	15 kg	130 kg		50 kg	45 kg	55 kg
	Halski		45 kg	75 kg		30 kg	95 kg	45 kg	15 kg	20 kg	60 kg	85 kg	25 kg	60 kg	50 kg	115 kg
	Kowalski			15 kg		15 kg		35 kg	10 kg	30 kg	10 kg	30 kg	45 kg	40 kg	15 kg	25 kg
	Lipiński		50 kg	95 kg	65 kg	120 kg	80 kg	45 kg	70 kg	70 kg	90 kg	90 kg	30 kg	70 kg	95 kg	85 kg
	Nowak		35 kg	30 kg	10 kg	45 kg	30 kg			45 kg	35 kg	65 kg			25 kg	15 kg
	Urbaniak			20 kg	30 kg	25 kg	40 kg	15 kg	35 kg	15 kg	10 kg	80 kg	20 kg	15 kg	-10 kg	10 kg
	Zieliński		30 kg	20 kg			25 kg	30 kg	10 kg	40 kg	20 kg	55 kg	20 kg	10 kg	60 kg	40 kg

rys.1

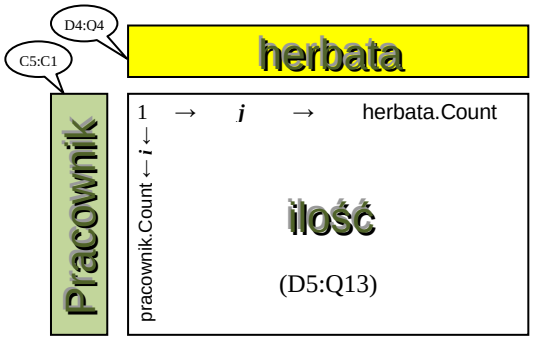
Zadanie 1

Celem zadania jest zaprojektowanie makra, które w nowym arkuszu o nazwie ***Arkusz2*** wygeneruje tabelę danych o strukturze przedstawionej na rys.2 na bazie danych zgromadzonych w ***Arkusz1*** (rys.1). Dla każdej wartości kilogramów zamówionej herbaty makro powinno odczytać z tego samego wiersza nazwisko pracownika i z tej samej kolumny nazwę herbaty i zapisać te trzy dane w poszczególnych wierszach nowej tabeli. W wygenerowanej tabeli powinno być zatem tyle wierszy ile jest niezerowych liczb określających ilość sprzedanej herbaty.

Pracownik	nazwa herbaty	ilość (kg)
Dąbrowski	Assam	55
Grabowski	Assam	70
Halski	Assam	45
Lipiński	Assam	50
Nowak	Assam	35
Zieliński	Assam	30
Dąbrowski	Black Lychee	20
Górecki	Black Lychee	10
Halski	Black Lychee	75
Kowalski	Black Lychee	15
Lipiński	Black Lychee	95
Nowak	Black Lychee	30
Urbaniak	Black Lychee	20
Zieliński	Black Lychee	20
Dąbrowski	Ceylon	30
Górecki	Ceylon	35
Grabowski	Ceylon	15
Lipiński	Ceylon	65

n

rys.2



rys.3

Aby zrealizować powyższe zadanie wykonaj następujące polecenia:

a) W arkuszy kalkulacyjnym wprowadź nazwy ***herbata, pracownik, ilość*** zgodnie ze schematem przedstawionym na rys.3.

b) W arkuszu ***Arkusz2*** w komórkach A1,B1,C1 wpisz odpowiednio: ***Pracownik***, ***nazwa herbaty*** i ***ilość***.

c) W edytorze VBA napisz makro `wypisz_tabele` , które powinno zawierać:

- deklarację zmiennych określających zakresy: `herbata`, `pracownik`, `ilość` oraz `tabela`,
- powiązanie tych zmiennych z zakresami arkusza za pomocą instrukcji **`Set`**, przy czym powiązanie zmiennej `tabela` można zrobić albo poprzez odwołanie się do odpowiedniej, wcześniej zdefiniowanej nazwy zakresu komórek w ***arkusz2*** albo też wprost:

```
Set tabela = Worksheets("Arkusz2").Range("a2")
```

- deklarację zmiennych `i,j,n` typu integer, indeksujących odpowiednio wiersze i kolumny tabeli wyjściowej i wiersze tabeli docelowej,
- blok pętli przebiegających po wszystkich komórkach zakresu ***ilość***. W bloku tym dla *n*-tego wiersza w zakresie `tabela` należy określić wartości: `tabela(n,1)`, `tabela(n,2)` oraz `tabela(n,3)` na podstawie danych `pracownik(…)`, `herbata(…)`,`ilość(…,…)`.

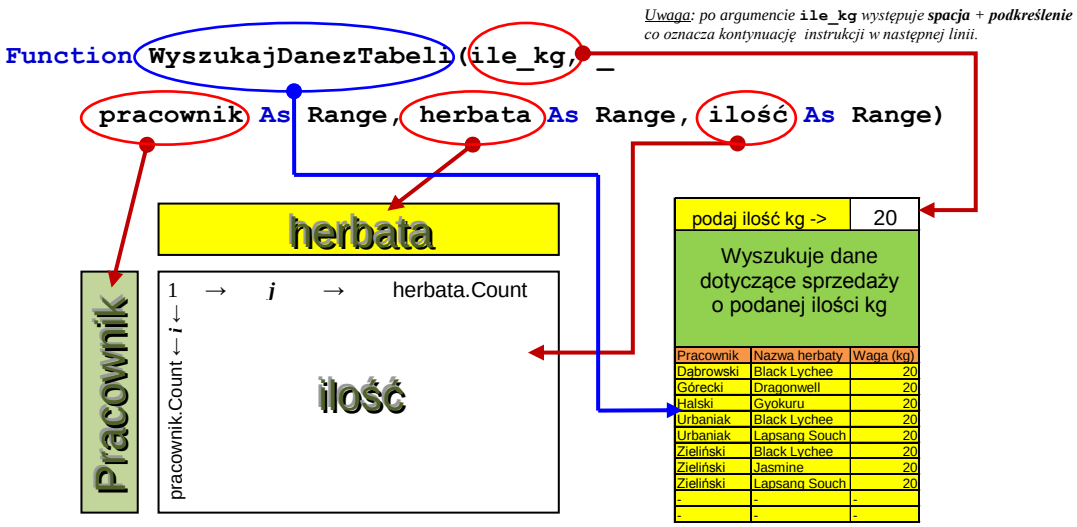
Sposób zapisu danych przedstawiony w tabeli danych w ***Arkusz2*** jest jednym z najbardziej rozpowszechnionych formatów gromadzenia danych charakterytycznym m.in. dla tzw. relacyjnych systemów baz danych. W samym Excelu jest wiele narzędzi operujących na tego typu tabelach do których m.in. należy ***raport tabeli przestawnej***.

Zadanie 2

W arkuszu kalkulacyjnym wybierz na wstążce ***Wstawianie*** przycisk ***Tabela przestawna*** i postępując wg instrukcji kreatora w nowym arkuszu przekształć dane zebrane w ***Arkuszu2*** do postaci z ***Arkusz1***.

Zadanie 3

Należy zaprojektować funkcję tablicową, która wypisuje wszystkie dane dotyczące sprzedaży w formie analogicznej jak w zadaniu 1 (rys.2) ale tylko dla określonej przez parametr ilości herbaty. Na rys.4 pokazano projekt nagłówka takiej funkcji, a także schematycznie przedstawiono sposób przekazywania danych poprzez argumenty funkcji.



rys.4

W projekcie tej funkcji zadeklaruj dynamiczną tablicę o nazwie `tabela` o `ilość.Count` wierszach i 3 kolumnach. Wypełnianie tej tablicy zaprogramuj podobnie jak w zadaniu 1. Użyj tej funkcji tablicowej w arkuszu w pustym zakresie komórek o szerokości 3 kolumn i ok. 20 wierszy. Dopisz odpowiednie nagłówki i opisy nad wygenerowaną tabelą wzorując się na rys.4.

Zadanie 4

Napisz funkcję podobną do tej z poprzedniego zadania, która oprócz ***nazwiska***, ***nazwy herbaty*** i ***ilości*** wypisuje w czwartej kolumnie ***cenę*** sprzedanej herbaty.

Zadanie 5

Zaprojektuj funkcję tablicową, która w pięciu kolejnych komórkach wypisuje 5 maksymalnych i wzajemnie różnych wartości sprzedanej herbaty. Uwaga: można wykorzystać dostępną w VBA funkcję Excela `WorksheetFunction.Large`. Pomoc na temat tej funkcji można uzyskać wypisując jej nazwę + **F1**.

Zadanie 6

Zaprojektuj funkcję tablicową, która w podobny sposób jak w zadaniu 3 wypisuje dane dotyczące sprzedaży, przy czym należy wypisać te dane, dla których ilość sprzedanej herbaty osiąga trzy ***maksymalne*** wartości.

Zadanie 7

Treść jak zadanie 6 tylko zamiast ***maksymalne*** wartości wstawiamy ***minimalne i niezerowe*** wartości.

Zadanie domowe

Napisz funkcję tablicową, która dla wskazanego w argumencie funkcji pracownika podaje listę sprzedaży herbat z podaniem ich ilości.

Dąbrowski	
Assam	55 kg
Black Lychee	20 kg
Ceylon	30 kg
Chunmee	50 kg
Darjeeling	115 kg
Dragonwell	60 kg
Gunpowder	35 kg
Gyokuru	45 kg
Jasmine	10 kg
Keemun	115 kg
Lapsang Souchong	35 kg
Oolong	60 kg
Pi Lo Chun	15 kg
Russian Blend	55 kg

Górecki	
Black Lychee	10 kg
Ceylon	35 kg
Chunmee	15 kg
Darjeeling	110 kg
Dragonwell	20 kg
Gunpowder	45 kg
Gyokuru	15 kg
Jasmine	25 kg
Keemun	50 kg
Lapsang Souchong	65 kg
Oolong	60 kg
Pi Lo Chun	25 kg
Russian Blend	90 kg
-	-

Grabowski	
Assam	70 kg
Ceylon	15 kg
Chunmee	50 kg
Darjeeling	65 kg
Dragonwell	35 kg
Gunpowder	15 kg
Gyokuru	40 kg
Jasmine	15 kg
Keemun	130 kg
Oolong	50 kg
Pi Lo Chun	45 kg
Russian Blend	55 kg
-	-
-	-

Ć W I C Z E N I E 8

Zagadnienia:

Podstawy projektowania aplikacji w Excelu:

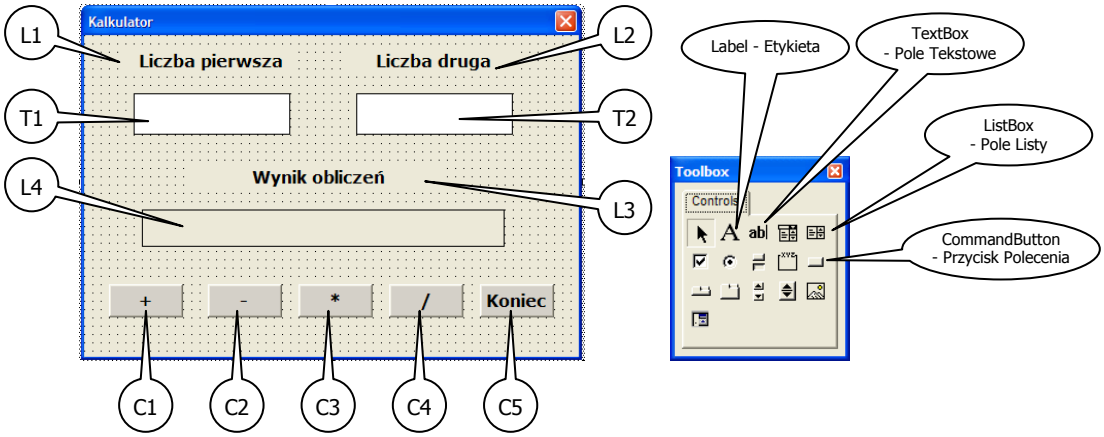
- Projektowanie formularzy z elementami sterującymi (kontrolkami).
- Tworzenie procedur obsługi zdarzeń elementów formularza.

Ćwiczenie to należy wykonać w opracowanym w ćwiczeniu 7 skoroszycie **sprzedaz herbaty.xlsm** (zadanie 1 jednak nie odnosi się do danych z arkusza kalkulacyjnego i może być wykonane w dowolnym innym pliku Excela).

Zadanie 1

Celem zadania jest zaprojektowanie prostego kalkulatora realizującego podstawowe działania arytmetyczne.

1. Zaprojektuj formularz kalkulatora pokazany na rys.1. W tym celu w edytorze VB z menu *Insert* kliknij pozycję *UserForm*. W oknie *Properties* ustaw właściwość *Caption* oraz *Name* na `Kalkulator` zamiast `UserForm1`.



2. Wykorzystując okno *Toolbox* wstaw wzorując się na rysunku powyżej następujące kontrolki:
   - Cztery etykiety L1,L2,L3 i L4. W oknie *properties* ustaw właściwości:
     - *Font* – wielkość 12 + pogrubienie,
     - *TextAlign* (położenie tekstu) na `2-fmTextAlignCenter` (wypośrodkowanie),
     - *Caption* (napisy) – ustaw zgodnie z rysunkiem na *Liczba 1* (L1), *Liczba 2* (L2), *Wynik obliczeń* (L3) i wymaż napis *Label4* (L4).
     - *BorderStyle* (styl obwódki) (L4) zmień na `fmBorderStyleSingle`.
     - (*Name*) czyli nazwę zamiast *Label4* zmień na *Wynik*.
   - Dwa pola tekstowe T1 iT2. Ustaw właściwości:
     - *Font* – wielkość 12 + pogrubienie,
     - *Text* – wymaż `TextBox1` i `TextBox2`,
     - *BorderStyle* (styl obwódki) zmień na `fmBorderStyleSingle`,
     - (*Name*) zmień na *Ed_liczba1* i *Ed_liczba2*.
   - Pięć przycisków C1,C2,C3,C4 i C5. Ustaw właściwości:
     - *Font* – wielkość 12 + pogrubienie,

- *TextAlign* (położenie tekstu) na `2-fmTextAlignCenter`
- *Caption* (napisy) – ustaw zgodnie z rysunkiem na: `+`, `-`, `*`, `/` i **`Koniec`**.
- (*Name*) zmień odpowiednio na `Kl_plus`, `Kl_minus`, `Kl_mno`, `Kl_dziel`, `Kl_koniec`.

Uwagi odnośnie oprogramowania kalkulatora

Końcowym etapem projektu jest oprogramowanie kalkulatora. Z punktu widzenia systemu operacyjnego każde kliknięcie na wybrany przycisk kalkulatora jest pewnym zdarzeniem, na które oprogramowanie powinno w określony sposób zareagować np. przez wyświetlenie w polu wynik jakiegoś napisu. Sposób reakcji określa procedura obsługi zdarzenia, którą twórca programu powinien zaprojektować. Nazwa takiej procedury w VBA tworzona jest automatycznie wg schematu ⟨***nazwa obiektu-kontrolki***⟩ + ⟨***podkreślenie***⟩ + ⟨***zdarzenie***⟩ np. procedura obsługi kliknięcia na przycisk `+` ma nazwę `Kl_plus_Click`. Procedury obsługi zdarzeń mogą przetwarzać dane reprezentowane przez zmienne i stałe w języku VBA, przez obiekty arkusza kalkulacyjnego jak i przez właściwości związane z obiektami (kontrolkami) zaprojektowanego formularza. Te ostatnie są identyfikowane za pomocą nazwy kontrolki i po kropce podanej właściwości np. tekst wpisany do pola tekstowego T1 określa napis `Ed_liczba1.Text` (por. p.2). W przypadku kontrolek nie jest stosowana ich jawna deklaracja w instrukcji `Dim`. Właściwości obiektów mogą być różnych typów, np. właściwość `Text` jest typu `string`. Informację odnośnie typu właściwości można uzyskać klawiszem F1 ustawiając kursor na nazwie właściwości. Jeśli wpiszemy liczbę do pola tekstowego to będzie ona interpretowana jako tekst (`string`) i nie będzie mogła być używana w operacjach arytmetycznych. Dlatego też stosuje się tzw. funkcje konwersji, które dokonują przekształcenia danej jednego typu na daną innego typu. Przykładem może być funkcja `CDbl`, która zamienia tekst na liczbę typu **`Double`**. Uwaga: VBA niejawnie używa tego typu funkcji w instrukcjach przypisania. Przykład: przy deklaracji

```
Dim s As string, d As double
```

poprawna jest instrukcja:

```
d = s
```

która w rzeczywistości jest instrukcją:

```
d = CDbl(s)
```

W sytuacji gdy tekst `s` nie ma formatu liczbowego zostanie w obu przypadkach wygenerowany błąd. Można go zignorować wypisując wcześniej polecenie:

```
On Error Resume Next
```

3. Wykonaj teraz następujące czynności:
   - W module ***Arkusz1*** napisz makro:

```
Sub Uruchom_kalkulator()
    Kalkulator.Show
End Sub
```

W arkuszu Excela osadź przycisk i przypisz do niego to makro. Uruchom kalkulator, a następnie zamknij go w prawym górnym rogu formularza.

- Będąc w oknie projektu formularza kliknij dwukrotnie na przycisk **Koniec**. Uzupełnij procedurę obsługi tego klawisza do postaci:

```
Private Sub Kl_koniec_Click()
'Instrukcja usuwa formularz z pamięci operacyjnej
      Unload Me
End Sub
```

Uruchom kalkulator i sprawdź działanie procedury.

- Będąc w oknie projektu formularza kliknij dwukrotnie na przycisk +. Uzupełnij procedurę obsługi tego klawisza do postaci:

```
Private Sub Kl_plus_Click()
Dim x As Double, y As Double
On Error Resume Next
x = Ed_liczba1.Text
y = Ed_liczba2.Text
    Wynik.Caption = x + y
End Sub
```

Uruchom kalkulator i sprawdź działanie procedury.

- W podobny sposób oprogramuj pozostałe przyciski. Napisz jedną z procedur obsługi przycisku np. przycisku mnożenia bez użycia zmiennych `x`, `y` wykorzystując funkcję `CDbl`. W procedurze obsługi przycisku dzielenia uwzględnij przypadek dzielenia przez 0. Procedura powinna zasygnalizować błąd dzielenia przez 0.

Zadanie 2

Celem kolejnego zadania jest zaprojektowanie formularza przetwarzającego dane z arkusza kalkulacyjnego. Zadanie to bazuje na opracowanym w ćwiczeniu 7 w zakresie zadania 3 arkuszu ***sprzedaż herbaty.xls***.

1. Zaprojektuj formularz, który po uruchomieniu wygląda tak jak na rysunku poniżej.

Formularz zawiera na górze pole tekstowe, poniżej pole listy, oraz przycisk z napisem **Koniec**. Nazwij te kontrolki odpowiednio: `Okno_tekst`, `Okno_listy` i `Klawisz_koniec`. Oprócz tych kontrolek w formularzu

osadzone są etykiety, które służą do opisu formularza. Ustaw odpowiednio właściwości takie jak *Font*, *Text*, *Caption*, wszystkich kontrolek. Oprócz tego ustaw właściwość *ColumnCount* kontrolki `Okno_listy` na 3. Następnie kliknij w oknie projektu dwukrotnie na przycisk **Koniec** i dopisz `Unload Me` do procedury `Klawisz_koniec_Click()`. Następnie dopisz procedurę obsługi zdarzenia `AfterUpdate` dla kontrolki `Okno_tekst` pokazaną poniżej:

```
Private Sub Okno_tekst_AfterUpdate()
   Okno_listy.List  =  WyszukajDanezTabeli(CDbl(Okno_tekst.Text),
_
   [pracownik], [herbata], [ilość])
End Sub
```

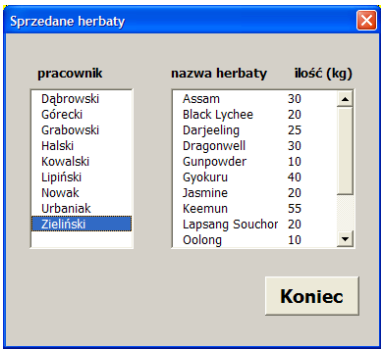
Zdarzenie `AfterUpdate` polega na wpisaniu wartości do pola tekstowego i naciśnięciu **ENTER**. Napis `[pracownik]` oznacza to samo co `Range("pracownik")`.

2. Zmodyfikuj formularz opracowany w poprzednim punkcie dodając nową kolumnę w oknie `Okno_listy`, w której będą podane ceny sprzedaży herbaty (cena 1kg razy ilość). Wskazówka: wykorzystaj funkcje tablicową opracowaną w ćwiczeniu 7 zadanie 4.

3. Dodaj do formularza na wysokości klawisza **Koniec** etykietę o nazwie `Licznik`. Zmodyfikuj procedurę `Okno_tekst_AfterUpdate` tak aby w etykiecie `Licznik` była wyświetlana ilość wierszy wyświetlanych w oknie `Okno_listy`. Wskazówka: wykorzystaj funkcję *WorksheetFunction.CountIf* będącą angielskim odpowiednikiem funkcji arkusza kalkulacyjnego *LICZ.JEŻELI*.

Zadanie domowe

Zadanie to należy wykonać na podstawie arkusza ***sprzedaż herbaty.xls***.
Zaprojektuj formularz, który po uruchomieniu wygląda tak jak na rysunku poniżej. Formularz powinien po kliknięciu na wybrane nazwisko podać listę w drugim oknie.



Wskazówki: Osadź w formularzu dwa okna typu *ListBox*. W jednym oknie wypisz listę pracowników korzystając z właściwości `RowSource`. Aby wyświetlić zawartość drugiego okna wykorzystaj w procedurze obsługi kliknięcia pierwszego okna funkcję z zadania domowego ćwiczenia 7 w celu określenia właściwości `List` drugiego okna.

Ć W I C Z E N I E 9

Zagadnienia:

- Tworzenie złożonych programów: dzielenie programu na procedury i funkcje

Ćwiczenie to dotyczy planu zajęć dla kierunków niestacjonarnych na Wydziale Inżynierii Mechanicznej i Mechatroniki (dawniej Wydziału Mechanicznego) ZUT. Plan ten przedstawiony jest na rysunku na stronie następnej. Odpowiedni arkusz z planem w Excelu należy pobrać z zasobu wskazanego przez prowadzącego ćwiczenia. Pobrany arkusz z planem powinien być niepokolorowany. Zakres kilku komórek pod napisem **Oznaczenia** należy w Excelu nazwać ***legenda***, zaś duży pokolorowany na rysunku zakres nad napisem **Oznaczenia** należy nazwać ***plan***.

Zadanie 1.

Celem zadania jest pokolorowanie planu zajęć za pomocą makra w taki sposób, aby różne przedmioty były pokolorowane różnymi kolorami zgodnie ze schematem kolorów ustawionym w ***legendzie*** (por. rys.). Główny problem, który należy rozwiązać to znaleźć metodę rozpoznawania komórki na planie na podstawie oznaczenia w legendzie. Na przykład, każdy bez problemu oceni, że napis MAT$_L$ SWZ na planie odpowiada w legendzie napisowi MAT, który określa przedmiot ***Materiałoznawstwo***, a nie odpowiada np. napisowi MT określającemu ***Mechanikę techniczną I***. Problem ten w VBA można sprowadzić do napisania funkcji o nagłówku:

```
Function zgodne_napisy(kom_legenda As Range, _
                       kom_plan As Range) As Boolean
```

gdzie `kom_legenda` i `kom_plan` oznaczają odpowiednio komórkę legendy i planu. Funkcja `zgodne_napisy` ma odpowiedzieć na pytanie (FALSE albo TRUE) czy komórka w legendzie odpowiada komórce na planie. Przy opracowaniu tej funkcji można wykorzystać fakt, że napis na planie tworzony jest w ten sposób, że do oznaczenia przedmiotu z legendy dodawana jest litera W,L lub C (wykład, laboratorium, ćwiczenia) czasem napis W/C, ale zawsze w indeksie dolnym. Można zatem poszukać pierwszego znaku, który jest w indeksie dolnym i porównać napis znajdujący się przed tym znakiem z napisami w legendzie. Aby sprawdzić cz *k*-ty znak jest w indeksie dolnym należy sprawdzić czy właściwość:

```
….Characters(Start:=k, Length:=1).Font.Subscript
```

ma wartość `True`. Po napisaniu tej funkcji można ją przetestować w Excelu.
Funkcję tę następnie należy wykorzystać w makrze kolorowania, które powinno sprawdzać pary: komórka na planie i komórka legendy i ewentualnie kopiować kolor wnętrza komórki legendy do komórki na planie.

ROZKŁAD ZAJĘĆ DYDAKTYCZNYCH

Koloruj

kolory ustawia się w komórkach E32:E38

Tworzy tabelę danych w Arkuszu 2

Plan dnia

dla niestacjonarnych studiów **pierwszego** stopnia Wydziału Mechanicznego

Kierunek: **TRANSPORT**

Rok akademicki: 2006/2007 Semestr: **2**

Nr zjazdu	Godziny zajęć: Data i dzień zjazdu	Sobota / Niedziela	Grupa	1 10^{00} / 8^{00}	2 10^{50} / 8^{50}	3 11^{40} / 9^{40}	4 12^{30} / 10^{30}	5 13^{20} / 11^{20}	6 14^{10} / 12^{10}	7 15^{00} / 13^{00}	8 15^{50} / 13^{50}	9 16^{40} / 14^{40}	10 17^{30} / 15^{30}	11 18^{20} / 16^{20}
	3.03.2007	sobota	T_n 11	FFT_W 205			MT_W 205		M_W 205			GI_W 329		---
1	4.03.2007	niedziela	T_n 11	M_C 205			MT_W 205			FFT_C 205			---	---
	10.03.2007	sobota	T_n 11	M_W 205			I_W 301		I_L 104/121			---	---	---
2	11.03.2007	niedziela	T_n 11	M_C 205			MT_C 329		GI_w		GI_P 10		---	---
	24.03.2007	sobota	T_n 11	MT_W 50		MT_C 50		EE_L BMW SWZ				GI_P 10		---
3	25.03.2007	niedziela	T_n 11	MAT_L SWZ			EE_W 233		I_W 301		I_L 104/121			---
	31.03.2007	sobota	T_n 11	FFT_W 205		FFT_C 205		GI_P 10		MAT_L SWZ			---	---
4	1.04.2007	niedziela	T_n 11	$MT_{W/C}$ 50			EE_W 50		I_W 301		I_L 104/121			---
	14.04.2007	sobota	T_n 11	M_W 306			MT_W 306		MT_C 306		GI_P 10		---	---
5	15.04.2007	niedziela	T_n 11	M_C 305			EE_W 305		EE_L BMW SWZ				---	---
	21.04.2007	sobota	T_n 11	EE_W 205			GI_W 10		GI_P 10		M_W 205			---
6	22.04.2007	niedziela	T_n 11	M_C 306			EE_W 306		EE_L BMW SWZ				---	---
	28.04.2007	sobota	T_n 11	$MT_{W/C}$ 50			EE_W 50		I_W 301		I_L 104/121			---
7	29.04.2007	niedziela	T_n 11	EE_L BMW SWZ				GI_W 10		GI_P 10		MAT_L SWZ		
	12.05.2007	sobota	T_n 11	$MT_{W/C}$ 50			EE_W 50		MAT_W 111			---	---	---
8	13.05.2007	niedziela	T_n 11	EE_L BMW SWZ				GI_W 10		GI_P 10		---	---	---
	19.05.2007	sobota	T_n 11	GI_P 10		MT_W 205		FFT_W 205		M_W 205			---	---
9	20.05.2007	niedziela	T_n 11	MAT_L SWZ			M_C 305			FFT_C 305			---	---
	2.06.2007	sobota	T_n 11	MT_W 50		MT_C 50		GI_P 10		FFT_W 205		FFT_C 205		---
10	3.06.2007	niedziela	T_n 11	MT_W 50		MT_C 50		MAT_L SWZ			EE_W 50			---
	16.06.2007	sobota	T_n 11	GI_P 10		$MT_{W/C}$ 50			I_W 301		I_L 104/121			---
11	17.06.2007	niedziela	T_n 11	MT_W 325		MT_C 325		MAT_W 111		EE_W 111		---	---	---

Oznaczenie	Nazwa przedmiotu	Egzamin	Zaliczenie	Punkty
M	Matematyka II	1	1	4
I	Informatyka II		2	4
MAT	Materiałoznawstwo II	1	1	3
MT	Mechanika techniczna I	1	1	5
GI	Grafika inżynierska II		2	4
EE	Elektrotechnika i elektronika		2	5
FFT	Finanse firm transportowych		2	3

Zadanie 2.

Celem zadania jest napisanie makra, które wygeneruje w ***Arkusz 2*** tabelę, której początek przedstawiony jest na rysunku poniżej.

Data	Dzień	Od	Do	Przedmiot	Rodzaj zajęć	Sala
3.03.2007	sobota	10:00	12:30	Finanse firm transportowych	W	205
3.03.2007	sobota	12:30	14:10	Mechanika techniczna I	W	205
3.03.2007	sobota	14:10	16:40	Matematyka II	W	205
3.03.2007	sobota	16:40	18:20	Grafika inżynierska II	W	329
4.03.2007	niedziela	08:00	10:30	Matematyka II	C	205
4.03.2007	niedziela	10:30	13:00	Mechanika techniczna I	W	205
4.03.2007	niedziela	13:00	15:30	Finanse firm transportowych	C	205
10.03.2007	sobota	10:00	12:30	Matematyka II	W	205
10.03.2007	sobota	12:30	14:10	Informatyka II	W	301
10.03.2007	sobota	14:10	16:40	Informatyka II	L	104/121
11.03.2007	niedziela	08:00	10:30	Matematyka II	C	205
11.03.2007	niedziela	10:30	12:10	Mechanika techniczna I	C	329
11.03.2007	niedziela	12:10	13:50	Grafika inżynierska II	W	
11.03.2007	niedziela	13:50	15:30	Grafika inżynierska II	P	10
24.03.2007	sobota	10:00	11:40	Mechanika techniczna I	W	50
24.03.2007	sobota	11:40	13:20	Mechanika techniczna I	C	50
24.03.2007	sobota	13:20	16:40	Elektrotechnika i elektronika	L	BMW SWZ

Zadanie to podzielimy na kilka mniejszych zadań cząstkowych.

1. Napisz funkcję o nagłówku:

```
Function szer(kom_plan As Range) As Integer
```

   która podaje ilość godzin zajęć na podstawie ilości scalonych w Excelu komórek.

2. Napisz funkcję o nagłówku:

```
Function nazwa_przedmiotu(kom As Range, _
                          legenda As Range) As String
```

   która podaje nazwę przedmiotu na podstawie oznaczenia w komórce `kom` na planie wykorzystując dane z zakresu legenda. Dane z zakresu, w którym są nazwy przedmiotów można pobrać stosując właściwość `Offset` do zakresu `legenda`.

3. Napisz funkcję o nagłówku:

```
Function rodzaj_zajęć(kom As Range) As String
```

   która podaje oznaczenie rodzaju zajęć: ***W*** -wykład, ***L*** – laboratoria, ***C*** - ćwiczenia itd.

4. Napisz funkcję o nagłówku:

```
Function dodaj_dwukropek(s As String) As String
```

   Funkcja ta powinna zamieniać format czasu zapisany niestandardowo w Excelu w postaci gg^{mm} na format gg:mm. Ponieważ tekst w komórce jako właściwość `Value` nie ma formatu więc funkcja powinna zamienić np. napis ”1050” na ”10:50”.

5. Napisz funkcję o nagłówku:

```
Function sala_zajęć(kom As Range) As String
```

   która podaje oznaczenie sali, w której odbywają się zajęcia.

6. Napisz funkcję tablicową o nagłówku:

```
Function czas_zajęć(kom_plan As Range, plan As Range)
```

   Funkcja ta powinna podać w postaci tablicy 2-elementowej czas rozpoczęcia i zakończenia zajęć określonych przez komórkę na planie `kom_plan`.

7. Napisz funkcję tablicową o nagłówku:

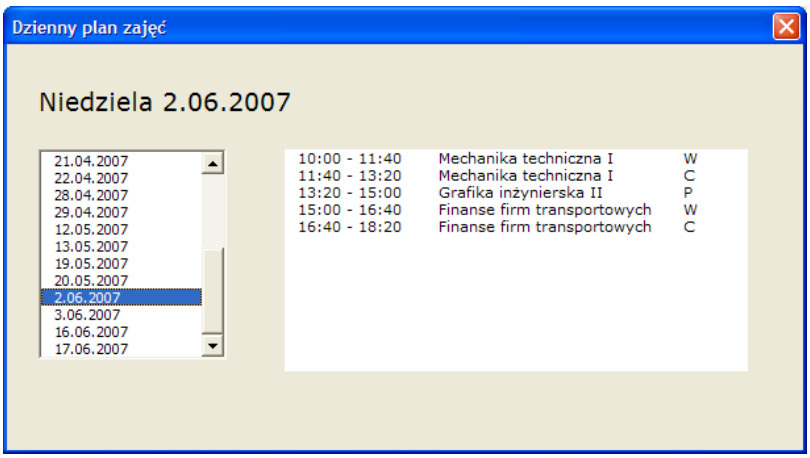
```
Function data_zajęć(kom_plan As Range, plan As Range)
```

   Funkcja ta powinna podać w postaci tablicy 2-elemntowej datę zajęć i dzień tygodnia (sobota/niedziela) zajęć określonych przez komórkę na planie `kom_plan`.

8. Jako ostatni element należy napisać makro `twórz_tabelę`, które przebiegając wszystkie komórki na planie wypisze tabelę w Arkusz 2 zgodnie z podanym wzorem. Makro to powinno wykorzystywać funkcje opisane w punktach 1 - 7.

Zadanie 3

Zaprojektuj formularz, który po uruchomieniu wygląda tak jak na rysunku poniżej. W formularzu osadzone są dwa pola typu *listbox* i etykieta. Formularz po kliknięciu na wybraną datę powinien wyświetlić w drugim oknie listę zajęć, które odbywają się w danym dniu z podaniem godziny rozpoczęcia i zakończenia zajęć, nazwy przedmiotu i rodzaju zajęć. Poza tym w etykiecie powinien być pokazany dzień tygodnia i wybrana data. Wykorzystaj funkcje opracowane w zadaniu 2.



D O D A T E K

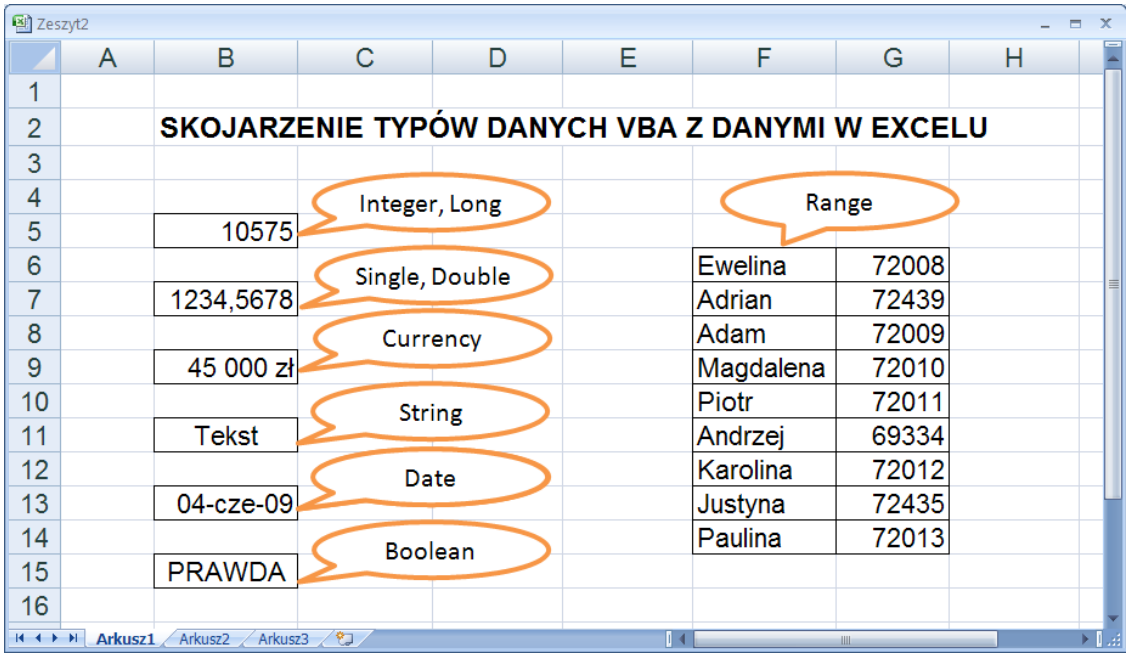
W dodatku zamieszczono wybrane informacje na temat języka *Visual Basic for Applications*, które należy sobie przyswoić w celu opracowania ćwiczeń. Wybór ten tylko w niewielkim stopniu określa podstawy tego języka.

1 Zmienne

Każde makro lub funkcja VBA jest zapisem czynności, które wykonywane są na pewnym określonym zbiorze informacji, nazywanych danymi. Dane mogą być zapisane bądź w komórkach arkusza kalkulacyjnego bądź w pewnym odrębnym obszarze pamięci operacyjnej komputera, który z kolei wewnątrz makra reprezentowany jest poprzez zmienne. Każda zmienna określona jest poprzez podanie nazwy, ta zaś tworzona jest z liter i cyfr (pierwszym znakiem w nazwie powinna być litera). Podobnie jak komórki w programie Excel zmienne mogą przechowywać dane różnych typów, przy czym w Visual Basic różnorodność typów jest znacznie większa (co więcej można tworzyć własne typy danych, czego nie będziemy omawiać). Typy danych mają swoje nazwy. W tabeli poniżej wypisane są podstawowe typy danych dostępne w VBA.

typ	opis	operacje	operator
Byte	liczby całkowite: od 0 do 255	dodawanie odejmowanie mnożenie dzielenie całkowite reszta z dzielenia	+ - * \ Mod
Integer	liczby całkowite: od -32 768 do 32 767		
Long	liczby całkowite: od -2 147 483 648 do 2 147 483 647		
Single	liczby rzeczywiste: od ok.$-3{,}4\cdot10^{38}$ do ok. $3{,}4\cdot10^{38}$	dodawanie odejmowanie mnożenie dzielenie potęgowanie	+ - * / ^
Double	liczby rzeczywiste: od ok.$-1{,}8\cdot10^{308}$ do ok. $1{,}8\cdot10^{308}$		
Currency	liczby wyrażające walutę -922 337 203 685 477,5808 do 922 337 203 685 477,5807		
Boolean	wartości logiczne: True (prawda), False (fałsz)	koniunkcja alternatywa alternatywa wykluczna negacja równoważność implikacja	And Or Xor Not Eqv Imp
String	łańcuchy znaków	łączenie łańcuchów	& , +
Date	daty i czas	specjalne funkcje daty i czasu	
Variant	łączy w sobie wszystkie powyższe typy danych	jak dla Double, String, Boolean	
		operatory relacji (wynik jest typu Boolean):	=,<,<=, >,>=,<>

W języku Visual Basic typ zmiennej można określić albo poprzez nadanie zmiennej określonej wartości albo za pomocą specjalnej instrukcji deklaracji **Dim** (patrz p.3.1). Przy pisaniu bardziej złożonych makr często trudno uniknąć pomyłek w wypisywaniu nazw zmiennych i tworzą się „nowe zmienne". Błędy te są często trudne do wykrycia. Dlatego też w VB można wymusić, aby wszystkie zmienne były zadeklarowane za pomocą instrukcji **Dim**. Pozwala to w przypadku użycia zmiennej o błędnej nazwie odnaleźć ją jako niezadeklarowanej na etapie interpretacji kodu. Wymuszenie to uzyskuje się wypisując w pierwszej linii danego modułu napis **Option Explicit**.



Elementy struktury arkusza kalkulacyjnego są w *Visual Basicu for Applications* reprezentowane przez tzw. obiekty i opisywane za pomocą „specjalnych" zmiennych, służących do identyfikowania m.in. arkuszy, komórek czy zakresów komórek W niniejszym opracowaniu ograniczymy się do opisu trzech obiektów: **Range, ActiveCell** i **Selection**

ActiveCell.Value lub krótko **ActiveCell** - przechowuje wartość wybranej (aktywnej) komórki,

ActiveCell.FormulaR1C1 - przechowuje formułę aktywnej(wybranej) komórki,

ActiveCell.Cells(1,2).Value lub krótko **ActiveCell(1,2)** pozwala wpisać wartości do komórki położonej na prawo od aktywnej, podobnie:

ActiveCell(1,0) położonej na lewo względem aktywnej,

ActiveCell(2,1) położonej w dół względem aktywnej,

ActiveCell(0,1) położonej w górę względem aktywnej,

Uwaga:

Zapis **ActiveCell(k)** jest równoważny zapisowi **ActiveCell(k,1).**

ActiveCell(1) jest równoważny zapisowi **ActiveCell.** Wszystkie komórki z tej samej kolumny co **ActiveCell** można wyrazić za pomocą **ActiveCell(k)** dla **k ≠ 1**.

Selection.Value lub krótko **Selection** - pozwala wpisać wartości do wybranego zakresu komórek,

Range("A1:C5").Value lub krótko **Range("A1:C5")** pozwala wpisać wartości do zakresu „A1:C5”,

Selection.Cells(1,2).Value lub krótko **Selection(1,2)** pozwala wpisać wartości do komórki w pierwszym wierszu i drugiej kolumnie w wybranym obszarze.

Range("A1:C5").Cells(5).Value lub **Range("A1:C5")(5)** pozwala wpisać wartości do 5-tej komórki w zakresie „A1:C5” tj. komórki B2.

Range("A1:C5").Cells.Count
podaje liczbę komórek w zakresie, w tym przypadku 15.

Range("A1:C5").Columns.Count
podaje liczbę kolumn w zakresie, w tym przypadku 3.

Range("A1:C5").Rows.Count
podaje liczbę wierszy w zakresie, w tym przypadku 5.

2 Wyrażenia

Wyrażenia tworzymy podobnie jak formuły w arkuszu, z tym, że zamiast adresów komórek wstawiamy zmienne. W wyrażeniach możemy wykorzystywać operatory, funkcje wewnętrzne Visual Basic, wszystkie funkcje Excela (w wersji angielskiej), a także istnieje możliwość definiowania własnych funkcji. Przy tworzeniu wyrażeń należy zwrócić szczególną uwagę na zgodność typów zmiennych, funkcji i używanych operatorów.

3 Instrukcje

Czynności, które ma realizować makro zapisujemy w postaci tzw. instrukcji. Instrukcje zapisujemy w kolejnych liniach. Jeśli chcemy kilka instrukcji zapisać w jednej linii oddzielamy je dwukropkami. Uwaga: napisy zaczynające się od apostrofu nie są instrukcjami lecz komentarzami. Omówimy kilka najważniejszych instrukcji.

3.1 Instrukcja deklaracji

Instrukcja ta przydziela pamięć zmiennej i określa jej typ i jest następującej postaci:

```
Dim zmienna1 As typ1, zmienna2 As typ2, ...
```

Przykład:

Poniżej zadeklarowano3 zmienne: pierwszą typu **Variant** (typ domyślny), drugą typu **Double**, trzecią typu **String**.

```
Dim liczba, numer As Double, napis As String
```

3.2 Instrukcja przypisania

Instrukcja ta ma postać:

```
zmienna = wyrażenie
```

W wyniku wykonania instrukcji zmienna otrzymuje wartość wyrażenia.

Przykłady:

```
d = b^2 - 4*a*c
i = i + 1
ActiveCell.FormulaR1C1 = "Jaś"
```

Pierwsza instrukcja nadaje zmiennej `d` wartość wyrażenia `b^2 - 4*a*c`, gdzie `a,b,c` są zmiennymi. Druga powiększa wartość zmiennej `i` o jeden. Trzecia wstawia do aktywnej komórki napis `"Jaś"`.

3.3 Instrukcja wywołania procedury

Instrukcja ta ma postać:

```
<nazwa procedury> parametr1, parametr2,...
```

Przykład:

```
MsgBox "Czas na naukę pisania makr!"
```

Instrukcja ta wyświetla komunikat `"Czas na naukę pisania makr!"` wykorzystując procedurę Visual Basica `MsgBox`.

3.4 Instrukcja warunkowa

Instrukcja ta może występować w składni jednowierszowej lub składni blokowej. Instrukcja ta wykonuje ciąg instrukcji w zależności od wartości logicznej wyrażenia –warunku lub kilku wyrażeń – warunków. Składnia jednowierszowa:

```
If warunek Then instrukcje Else instrukcje_else
```

lub w wersji krótszej:

```
If warunek Then instrukcje
```

wykonywane są *instrukcje* w przeciwnym przypadku *instrukcje_else*. W wersji krótszej gdy warunek nie jest spełniony następuje przejście do instrukcji następnej po **If**. W składni jednowierszowej instrukcje oddziela się dwukropkami.

Inną możliwością jest zastosowanie składni blokowej. Pozwala ona na kolejne sprawdzanie wielu warunków.

```
If Warunek_1 Then
      Ciąg instrukcji wykonywany gdy Warunek_1 jest prawdziwy
      [ElseIf Warunek_2 Then
            Ciąg instrukcji wykonywany gdy Warunek_2 jest prawdziwy
      [ElseIf Warunek_3 Then
            Ciąg instrukcji wykonywany gdy Warunek_3 jest prawdziwy
            ...
      [ElseIf Warunek_n Then]]]
            Ciąg instrukcji wykonywany gdy Warunek_n jest prawdziwy
            (poprzednie nie są prawdziwe)
      [Else
            Ciąg instrukcji wykonywany gdy żaden z poprzednich
            warunków nie jest prawdziwy]
End If
```

Uwaga: nawiasy […] oznaczają opcjonalną część instrukcji.

3.5 Instrukcja wyboru Select Case

Select Case jest instrukcją, która zastępuje zagnieżdżone instrukcje **If...ElseIf**, gdy konieczne jest wybieranie pomiędzy wieloma dostępnymi opcjami. **Select Case** sprawdza podane warunki i wybiera tylko jeden odpowiedni blok kodu. Składnia:

```
Select Case Wyrażenie
Case Wartość_1
      Ciąg instrukcji wykonywany gdy Wyrażenie = Wartość_1
Case Wartość_2
      Ciąg instrukcji wykonywany gdy Wyrażenie = Wartość_2
...
Case Wartość_n
      Ciąg instrukcji wykonywany gdy Wyrażenie = Wartość_n
[Case Else
      Ciąg instrukcji wykonywany gdy Wyrażenie jest różne od
      wszystkich Wartości od 1 do n. ]
End Select
```

Uwaga: nawiasy […] oznaczają opcjonalną część instrukcji.

3.6 Instrukcja pętli For

```
For licznik = początek To koniec Step krok
      instrukcje
Next licznik
```

licznik jest zmienną numeryczną, *początek* i *koniec* są wartościami początkowymi i końcowymi licznika. Wykonywane są instrukcje, po każdym wykonaniu licznik zwiększany jest o wartość *krok* lub o 1 gdy fraza **Step** jest pominięta. Wykonywanie zostanie przerwane gdy *licznik* przekroczy wartość *koniec*. Działanie pętli for można dodatkowo przerwać instrukcją **Exit For** umieszczoną wewnątrz pętli.

3.7 Instrukcja pętli Do

Instrukcja ta maże mieć następujące warianty składni:

```
Do While warunek
      instrukcje
Loop
```

albo

```
Do Until warunek
      instrukcje
Loop
```

albo

```
Do
      instrukcje
Loop While warunek
```

albo

```
Do
      instrukcje
Loop Until warunek
```

Jedna lub kilka instrukcji oznaczonych przez *instrukcje* powtarzanych jest tak długo, jak długo *warunek* jest spełniony – wariant ze słowem **While** - albo dopóki nie stanie się prawdziwy - wariant ze słowem **Until**, przy czym *warunek* można sprawdzać na początku lub na końcu.

3.8 Tablice – deklaracje tablic

Tablice to zbiory danych tego samego typu, odpowiednio uporządkowane za pomocą wartości numerycznych zwanych indeksami. Określa się wymiar tablicy jako ilość indeksów porządkujących tablicę. W poniższym wierszu zadeklarowano tablicę 2-wymiarową o elementach typu **Integer**

```
Dim MojaTab(1 To 10,1 To 10) As Integer
```

Tę tablicę możemy interpretować jako macierz 10 na 10: pierwszy z tych indeksów określa wiersze, drugi kolumny. Tablica składa się ze 100 elementów. Ten sposób deklaracji określa nam tzw. tablicę statyczną o stałej ilości elementów.
Jeśli w chwili deklaracji trudno określić wielkość tablicy, można użyć deklaracji tablicy dynamicznej. Nie ma ona rozmiaru do chwili użycia instrukcji **ReDim**. Sama deklaracja np. tablicy o elementach typu **String** ma postać:

```
Dim NowaTab() As String
```

W miejscu kodu, w którym tablica jest faktycznie wykorzystywana, umieszczamy instrukcję **ReDim** określającą rozmiar tablicy. Instrukcja

```
ReDim NowaTab(1 To 7)
```

ustala rozmiar tablicy wstępnie zadeklarowanej jako dynamiczna na siedem elementów.

4 Procedury i Funkcje

Każde makro w Excelu jest zapisywane jako procedura Visual Basic. Podczas pisania bardziej rozbudowanych makr często zdarza się, że taki sam lub podobny fragment makra występuję w kilku miejscach. Można wówczas taki powtarzający się fragment zapisać w postaci dodatkowej procedury lub funkcji.
Procedura lub funkcja - wyodrębniona sekwencja instrukcji, stanowiąca pewną całość, posiadająca jednoznaczną nazwę i ustalony sposób wymiany informacji z pozostałymi częściami programu lub makra.

Stosowanie procedur i funkcji na ogół skraca zapis, a także ułatwia pisanie dużych rozbudowanych makr dzięki podzieleniu go na odrębne logicznie spójne części. Różnica pomiędzy procedurą a funkcją polega na sposobie przekazywania wartości końcowych i sposobie ich wywoływania. Procedury wywoływane są specjalną instrukcją (patrz p.3) natomiast funkcje wykorzystuje się do budowania wyrażeń. Procedury i funkcje definiuje się przy pomocy instrukcji **Sub** i **Function**:

```
Sub nazwa(lista_argumentów)
      instrukcje
End Sub

Function nazwa(lista_argumentów) As typ
      instrukcje
      nazwa = wyrażenie
End Function
```

lista_argumentów jest listą zmiennych oddzielonych przecinkami i jest nieobowiązkowa podobnie jak *typ*, który określa typ zwracanego wyniku przez funkcję. W definicji funkcji typ wyrażenia (*wyrażenie)* powinien być zgodny z typem funkcji . Ponadto w definicji procedury i funkcji mogą się pojawić instrukcje **Exit Sub** i odpowiednio **Exit Function** przerywające działanie procedury lub funkcji.
Szczególnym przypadkiem funkcji są funkcje tablicowe, które poprzez swoją nazwę zwracają zamiast pojedynczej wartości całą tablicę:

```
Function nazwa(lista_argumentów)
      instrukcje
      nazwa = tablica
End Function
```

Funkcje można używać w formułach w Excelu i można je odnaleźć jako *funkcje użytkownika*. Funkcje tablicowe wykorzystujemy w formułach tablicowych wprowadzanych za pomocą kombinacji klawiszy **Ctr-Shift-Enter**. Uwaga: każda funkcja aby była widoczna w Excelu musi być wypisana w nowym module VB.

5 Przykłady

Przykład 1

Funkcja wyznacza normę euklidesową wektora o współrzędnych x,y,x. Funkcja ta jest dostępna w arkuszu jako funkcja użytkownika. Należy pamiętać aby w formule Excela parametry funkcji oddzielać średnikami.

```
Function Norma(x As Single,y As Single,z As Single) _
    As Single
    Norma = Sqr(x ^ 2 + y ^ 2 + z ^ 2)
End Function
```

Uwaga: Podkreślenie poprzedzone spacją oznacza kontynuację instrukcji w następnym wierszu.

Przykład 2.

Makro `Main` wywołuje dwie procedury: `Sygnał`, która wysyła krótkie sygnały dźwiękowe (procedura `Beep`) w ilości określonej przez parametr i procedurę `Komunikat` , która wypisuje napis `"Czas na kolejny przykład!"`.

```
Sub Main()
      'źródło: plik pomocy Visual Basic
      Sygnał 100
      Komunikat
End Sub

Sub Sygnał(ile_dźw)
      For licznik = 1 To ile_dźw
            Beep
      Next licznik
End Sub

Sub Komunikat()
      MsgBox "Czas na kolejny przykład!"
End Sub
```

Przykład 3

Makro w kolumnie A mnoży po kolei przez 2 każdą liczbę, aż do napotkania komórki pustej.

```
Sub Podwajaj()
Dim i As Integer
    i = 1
    Do While Range("A1")(i) <> ""
        Range("A1")(i) = 2 * Range("A1")(i)
        i = i + 1
    Loop
End Sub
```

Przykład 4

Poniższe makro sprawdza wartość liczbową komórki aktywnej i na tej podstawie wpisuje do komórki obok na prawo napis albo „`dodatnia`” albo „`ujemna`” albo „`zero`”. Dalej makro ustawia komórkę aktywną o jedną w dół. Na rysunku obok po ustawieniu `A2` jako komórki aktywnej kilkakrotne uruchomienie makra wypisuje napisy w kolumnie B.

	A	B
1	Liczba	Warość
2	-0,03	ujemna
3	1,79	dodatnia
4	-0,89	ujemna
5	0,13	dodatnia
6	1,05	dodatnia
7	-2,01	ujemna
8	0	zero
9	1,30	dodatnia
10	2,07	dodatnia
11	-0,59	ujemna
12	-1,69	ujemna
13	0,86	dodatnia
14	-2,21	ujemna
15	0	zero

```
Sub wartość()
    If ActiveCell > 0 Then
        ActiveCell(1, 2) = "dodatnia"
    ElseIf ActiveCell < 0 Then
        ActiveCell(1, 2) = "ujemna"
    Else
        ActiveCell(1, 2) = "zero"
    End If
    ActiveCell(2, 1).Select
End Sub
```

Przykład 5

Poniższa funkcja realizuje podobne zadanie jak makro z przykładu 4. Aby otrzymać podobny obraz jak na rysunku z przykładu 4 należy w komórce `B2` wpisać formułę z użyciem tej funkcji i przeciągnąć tę formułę w dół.

```
Function wartość1(liczba As Single) As String
    If liczba > 0 Then
        wartość1 = "dodatnia"
    ElseIf liczba < 0 Then
        wartość1 = "ujemna"
    Else
        wartość1 = "zero"
    End If
End Function
```

Przykład 6

Funkcja ta działa tak samo jak funkcja w przykładzie 5. Zamiast instrukcji **`If`** została użyta instrukcja **`Select Case`**.

```
Function wartość2(liczba As Single) As String
    Select Case liczba
    Case Is > 0
        wartość2 = "dodatnia"
    Case Is < 0
        wartość2 = "ujemna"
    Case Else
        wartość2 = "zero"
    End Select
End Function
```

Przykład 7

Przykład funkcji tablicowej. Funkcja ta z dowolnego prostokątnego obszaru wypisuje do jednokolumnowego zakresu dane z niepustych komórek.

as				**pas**
		las		
			44	
	Ala			
		ma		
kota		**i**		**psa**

=>

as
pas
las
44
Ala
ma
kota
i
psa

```
Function BezPustych(z As Range)
'
'wpisuje do jednokolumnowego zakresu dane z niepustych komórek
'
Dim kolumna, i As Integer, k As Integer
ReDim kolumna(1 To z.Count, 1 To 1)
k = 1
For i = 1 To z.Count
    If z(i) <> "" Then
        kolumna(k, 1) = z(i)
        k = k + 1
    End If
Next i
For i = k To z.Count
    kolumna(i, 1) = ""
Next i
BezPustych = kolumna
End Function
```