

Witold Wrotek

VBA

dla **Excela 2021 PL i 365 PL**

234
praktyczne
przykłady

Helion 

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz wydawca dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz wydawca nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Redaktor prowadzący: Małgorzata Kulik

Projekt okładki: Studio Gravite / Olsztyn

Obarek, Pokoński, Pazdrijowski, Zaprucki

Grafika na okładce została wykorzystana za zgodą Shutterstock.com

Helion S.A.

ul. Kościuszki 1c, 44-100 Gliwice

tel. 32 231 22 19, 32 230 98 63

e-mail: helion@helion.pl

WWW: <https://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<https://helion.pl/user/opinie/vbe213>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

ISBN: 978-83-283-8898-7

Copyright © Helion S.A. 2022

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

Spis treści

Wstęp	9
Co to jest VBA?	9
Basic	9
Visual Basic	10
Visual Basic for Applications	10
Czy VBA jest przeżytkiem?	11
VBA i Office 2007, 2010, 2013, 2016, 2019, 2021	11
Zmieniony wygląd okna programu	12
Makropolecenia w Microsoft Office 2021	12
Makropolecenia w Microsoft Office dla sieci Web	14
Bezpieczeństwo w Microsoft Office 2021	15
Jaką funkcję może pełnić VBA?	15
Kiedy warto stosować VBA?	16
Kiedy nie warto stosować VBA?	17
Jak napisać najprostszy program w VBA?	18
Podsumowanie	18
Rozdział 1. Makropolecenia	21
Siedem wersji Microsoft Office	22
Planowanie makropolecenia	22
Microsoft Office 2021	23
Planowanie i rejestrowanie makropolecenia	23
Podsumowanie	49
Rozdział 2. Jak uruchomić edytor Visual Basic for Applications	50
Gdzie w Excelu jest edytor VBA?	50
Czy makropolecenie może spowodować szkody?	57
Podsumowanie	68

Rozdział 3. Okno edytora Visual Basic for Applications	69
Okno Project	69
Okno Properties	71
Okno Code	71
Pasek menu	72
Pasek narzędziowy	74
Pierwszy program	74
Przełączanie między widokami	88
Wyrównywanie obiektów	90
Strzelanie z armaty do komara	94
Kiedy korzystać z makropoleczeń, a kiedy z programów napisanych w VBA?	94
Podsumowanie	94
Rozdział 4. Zmienne	95
Nazwy zmiennych w VBA	96
Pułapki systemu komunikatów	101
Typy danych	101
Pułapki braku deklaracji	110
Wymuszanie deklarowania zmiennych	115
Zasięg deklaracji	120
Zmienne lokalne	120
Zmienne obowiązujące wewnątrz całego modułu	120
Zmienne globalne	120
Przekazanie wartości przez zmienną globalną	125
Przekazanie wartości przez wywołanie procedury z parametrem	127
Deklaracja typu i instrukcja przypisania	129
Komórka arkusza jako zmienna	133
Tekst jako wartość zmiennej	139
Podsumowanie	142
Rozdział 5. Komunikacja z użytkownikiem	143
Wprowadzanie danych	144
Wyświetlanie komunikatów	155
Poprawność wprowadzanych danych	167
Wycinanie tekstu	174
Podsumowanie	179

Rozdział 6. Korzystanie z obiektów	181
Obiekty	181
Właściwości	181
Metody	182
Zaznaczanie komórki	182
Elektroniczny sufler	183
Usuwanie zawartości i formatowania	186
Usuwanie zawartości	187
Usuwanie formatowania	188
Usuwanie wartości mniejszych od progowej	189
Właściwości	191
Przypisanie wartości komórce	191
Kopiowanie zawartości komórek	194
Nadawanie komórce koloru z użyciem nazwy koloru	197
Nadawanie komórce koloru z użyciem kodu koloru	199
Nadawanie koloru zawartości komórki	202
Przesuwanie aktywnej komórki	204
Podsumowanie	207
Rozdział 7. Instrukcje warunkowe	208
Porównywanie	210
Sterowanie wykonywaniem procedur	210
Skok do etykiety	210
Podejmowanie decyzji	220
Wybór jednej z trzech lub więcej opcji	223
Wykonanie grupy instrukcji określoną liczbę razy	226
Pętle zagnieżdżone	233
Wykonywanie pętli, gdy warunek jest spełniony	237
Podsumowanie	239
Rozdział 8. Elementy sterujące arkusza	240
Pole listy	240
Pole kombi (listy rozwijanej)	247
Pasek Toolbox i elementy sterujące arkusza	250
Właściwości	252
Podsumowanie	256

Rozdział 9. Zdarzenia	257
Lista zdarzeń dla skoroszytu	259
Lista zdarzeń dla arkusza	265
Lista zdarzeń dla aplikacji	268
Komunikacja z programem	270
Lista zdarzeń dla formularzy	274
Podsumowanie	276
Rozdział 10. Metody i właściwości dla zakresu	277
Kopiowanie zakresu komórek	277
Sortowanie zakresu komórek	284
Filtrowanie zakresu komórek	289
Wyszukiwanie informacji	294
Podsumowanie	297
Rozdział 11. Podprogramy	298
Śledzenie pracy programu	305
Procedury zagnieżdżone	307
Procedury zapętlone	310
Podsumowanie	313
Rozdział 12. Ściągawka z VBA	314
Metody	314
Funkcje	322
Instrukcje	330
Operatory	334
Operatory arytmetyczne	334
Operatory porównania	337
Operator konkatencji	339
Operatory logiczne	340
Podsumowanie	342
Rozdział 13. Funkcje arkuszowe	344
Odpowiedniki funkcji arkuszowych w VBA	346
Podsumowanie	367

Dodatek A. Wybrane kody błędów VBA	369
Dodatek B. Programowanie obiektowe	372
Programowanie proceduralne a obiektowe	372
Właściwości	373
Metody	374
Zdarzenia	374
Kolekcje	374
Modele obiektowe	375
Metoda kropkowa	375
Obiekty aktywne	375
Zakończenie	377
Skorowidz	379

Rozdział 6.

Korzystanie z obiektów

Z tego rozdziału dowiesz się:

- ◆ Co to jest obiekt
- ◆ Co to jest właściwość
- ◆ Co to jest metoda

Obiekty

Obiektem jest element składowy aplikacji, np. komórka, zakres komórek, wykres umieszczony na arkuszu, arkusz, skoroszyt, arkusz programu Excel.

Excel zawiera obiekty skoroszytów, skoroszyty zawierają arkusze, arkusze zawierają zakresy, zakresy zawierają komórki. Powstaje więc struktura hierarchiczna.

Właściwości

Obiekty mają właściwości. Właściwość jest cechą obiektu. Właściwościami zakresu komórek (obektu Range) są: zawartość komórki, styl czcionki itp.

Przy odwoływaniu się do właściwości danego obiektu używa się składni:

Object.Property

gdzie:

- ◆ *Object* — nazwa obiektu,
- ◆ *Property* — nazwa właściwości.

Metody

Z obiektami są związane metody. Metoda jest operacją wykonaną na obiekcie — stanowi ją np. wpisanie tekstu do komórki czy zaznaczenie zakresu komórek.

Metoda jest działaniem, które można wykonać na obiekcie. Jest sposobem postępowania, który prowadzi do określonego rezultatu.



W celu posłużenia się metodą należy za nazwą obiektu wstawić kropkę, a następnie nazwę wybranej metody.

Zaznaczanie komórki

Do zaznaczania komórek można wykorzystać metodę `Select`.

Przykład 95.

Napisz program, którego zastosowanie pozwoli zaznaczyć komórkę `B2`.

1. W arkuszu Excel zaznacz komórkę `A1`.
2. Wyświetl okno `Code`.
3. Wpisz w nim kod programu (rysunek 6.1). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W linii drugiej znajduje się nazwa procedury. W linii trzeciej zapisano obiekt (`Range("B2")`). W ten sposób oznaczono pojedynczą komórkę `B2`. Następnie po kropce została zapisana metoda (`Select`), która umożliwia zaznaczenie komórki.

RYSUNEK 6.1.

Program, którego zastosowanie pozwala zaznaczyć komórkę `B2`

```
Option Explicit

Sub przykład95 ()
Range ("B2").Select

End Sub
```

4. Uruchom program.
5. W oknie `Macros` zaznacz `przykład95`.
6. Wyświetl arkusz Excela (rysunek 6.2).

RYSUNEK 6.2.

Zaznaczenie zostało przeniesione do komórki `B2`

	A	B
1		
2		

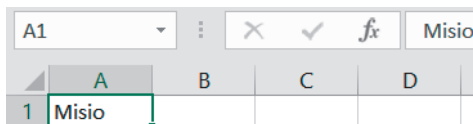
Przykład 96.

Napisz program, którego zastosowanie pozwoli wyczyścić komórkę A1.

1. W komórce A1 wpisz *Misio* (rysunek 6.3).

RYSUNEK 6.3.

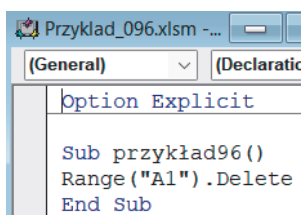
Dane do testowania



2. Wyświetl okno *Code*.
3. Wpisz w nim kod programu (rysunek 6.4). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W linii drugiej znajduje się nazwa procedury. W linii trzeciej zapisano obiekt (Range("A1")). W ten sposób „pokazano” programowi, że jego działanie ma dotyczyć komórki A1. Następnie po kropce została zapisana metoda (Delete), która umożliwi usunięcie zawartości komórki.

RYSUNEK 6.4.

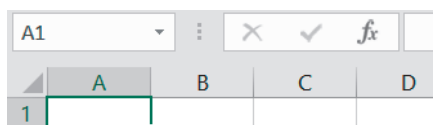
Program, którego zastosowanie pozwala wyczyścić komórkę B2



4. Uruchom program.
5. W oknie *Macros* zaznacz przyklad96.
6. Wyświetl arkusz Excel (rysunek 6.5).

RYSUNEK 6.5.

Zawartość komórki A1 została bezpowrotnie utracona

**Elektroniczny sufler**

Podczas pisania programu dużo problemów sprawiają:

- ♦ potrzeba dokładnej znajomości nazw obiektów i metod,
- ♦ literówki popełniane przy ich wpisywaniu.

Na szczęście edytor VBA został wyposażony w system podpowiedzi. Jak z niego korzystać, dowiesz się z przykładu 97.

Przykład 97.

Napisz program, którego zastosowanie pozwoli zaznaczyć komórkę B2.

1. Wyświetl okno *Code*.
2. Utwórz szkielet programu. Linia kończąca procedurę pojawia się automatycznie po wprowadzeniu jej nagłówka (rysunek 6.6).

RYSUNEK 6.6.

Szkielet programu

```

Option Explicit

Sub przyklad97 ()

End Sub

```

3. Umieść znak wstawiania między obiema liniami (rysunek 6.7).

RYSUNEK 6.7.

Znak wstawiania został umieszczony między liniami ograniczającymi procedurę

```

Option Explicit

Sub przyklad97 ()

|

End Sub

```

4. Podaj nazwę obiektu. W naszym przypadku będzie to zakres komórek. Wpisz Range(. W oknie edytora pojawiła się podpowieź (rysunek 6.8).

RYSUNEK 6.8.

Edytor podpowiada, jak można zdefiniować zakres

```

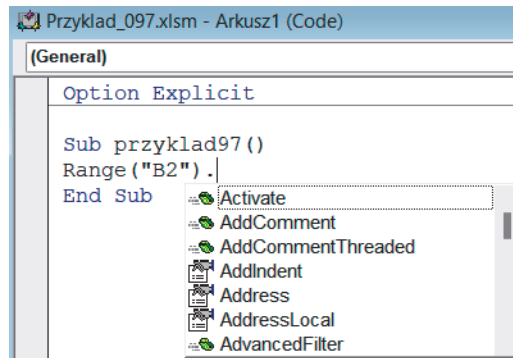
Option Explicit

Sub przyklad97 ()
Range (|
Range(Cell1, [Cell2]) As Range

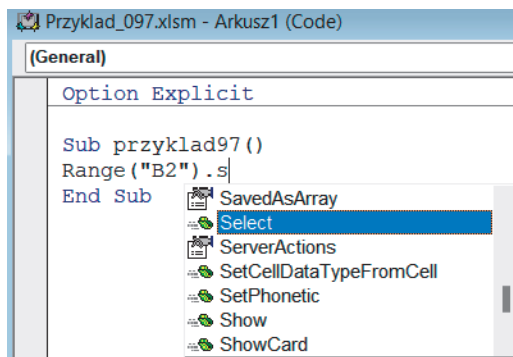
```

5. Zakresem będzie jedna komórka. Wpisz "B2") ..
6. Po wpisaniu kropki w oknie edytora pojawiła się podpowieź (rysunek 6.9).
7. Metody i właściwości są uporządkowane alfabetycznie. Zaznaczenie obiektu osiąga się poprzez zastosowanie metody Select. Nie musisz znać dokładnie jej pisowni. Wystarczy, że wiesz, na jaką literę zaczyna się nazwa. Wpisz literę S.
8. Została wyświetlona lista metod i właściwości o nazwach rozpoczynających się od litery S. Zaznacz metodę Select (rysunek 6.10).
9. Potwierdź wybór naciśnięciem klawisza *Enter*.

RYSUNEK 6.9.
Lista metod
i właściwości

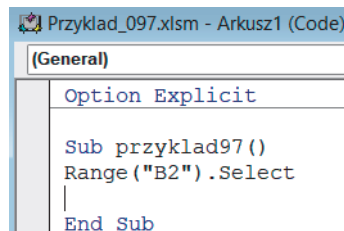


RYSUNEK 6.10.
Lista metod
i właściwości
o nazwach
rozpoczynających
się od litery S



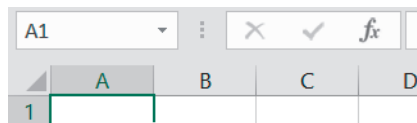
10. Do programu została wstawiona metoda `Select`. Program jest gotowy (rysunek 6.11).

RYSUNEK 6.11.
Program
po dodaniu metody



11. Teraz pora na sprawdzenie, czy program działa. Wyświetl okno arkusza Excel (rysunek 6.12). Domyślnie jest wybrana komórka `A1`.

RYSUNEK 6.12.
Arkusz z wybraną
komórką `A1`

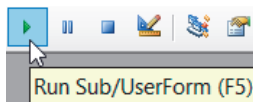


12. Wyświetl okno edytora.

13. Uruchom program kliknięciem ikony (rysunek 6.13).

RYSUNEK 6.13.

Kliknięcie ikony spowoduje uruchomienie programu

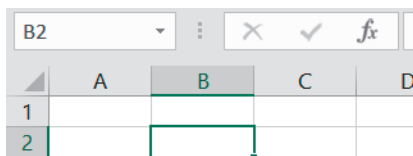


14. Program został wykonany.

15. Wyświetl okno arkusza Excel (rysunek 6.14). Wybraną komórką jest B2.

RYSUNEK 6.14.

Arkusz z wybraną komórką B2



Usuwanie zawartości i formatowania

Usunięcie zawartości i formatowania z komórek wymaga zaznaczenia zakresu komórek i posłużenia się metodą Clear.

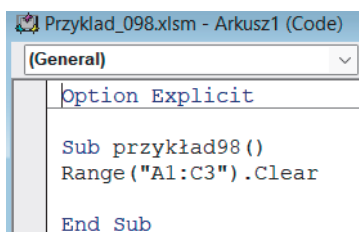
Przykład 98.

Napisz program, którego zastosowanie pozwoli czyścić i przywracać domyślne formatowanie komórek w zakresie od A1 do C3.

1. Wyświetl okno *Code*.
2. Wpisz w nim kod programu (rysunek 6.15). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W linii drugiej znajduje się nazwa procedury. W linii trzeciej znajduje się komenda czyszczenia zawartości i przywracania domyślnego formatowania komórek arkusza leżących w zakresie od A1 do C3.

RYSUNEK 6.15.

Program, którego zastosowanie pozwala usunąć zawartość i formatowanie komórek w obszarze od A1 do C3

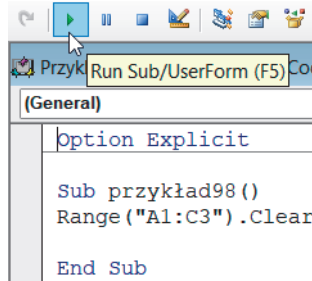


3. Wyświetl okno arkusza Excel. Domyślnie wszystkie komórki są puste. Wpisz dane do komórek (rysunek 6.16).
4. Wyświetl okno edytora.
5. Uruchom program kliknięciem ikony (rysunek 6.17).
6. Program został wykonany.

RYSUNEK 6.16.
Arkusz z danymi

	A	B	C	D
1	a	a	a	a
2	a	a	a	a
3	a	a	a	a
4	a	a	a	a

RYSUNEK 6.17.
Kliknięcie ikony spowoduje uruchomienie programu



- Wyświetl okno arkusza Excel (rysunek 6.18). Komórki z zakresu od A1 do C3 zostały wyczyszczone.

RYSUNEK 6.18.
Z komórek w zakresie od A1 do C3 zostały usunięte zawartość i formatowanie

	A	B	C	D
1				a
2				a
3				a
4	a	a	a	a

Usuwanie zawartości

Usunięcie zawartości z komórek wymaga zaznaczenia zakresu komórek i posłużenia się metodą `ClearContents`.

Przykład 99.

Napisz program, którego zastosowanie pozwoli usunąć zawartość komórek w zakresie od A1 do C3.

- Wyświetl okno *Code*.
- Wpisz w nim kod programu (rysunek 6.19). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W linii drugiej znajduje się nazwa procedury. W linii trzeciej znajduje się komenda usuwania zawartości komórek arkusza leżących w zakresie od A1 do C3.
- Wyświetl okno arkusza Excel. Domyślnie wszystkie komórki są puste. Wpisz dane do komórek (rysunek 6.20).
- Wyświetl okno edytora.

RYSUNEK 6.19.

Program, którego zastosowanie pozwala usunąć zawartość komórek w obszarze od A1 do C3

```
Przykład_099.xlsm - Arkusz1 (Code)
(General)
przykład99
Option Explicit
Sub przykład99()
Range("A1:C3").ClearContents
End Sub
```

RYSUNEK 6.20.

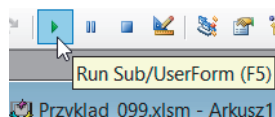
Arkusz z danymi

	A	B	C	D
1	a	a	a	a
2	a	a	a	a
3	a	a	a	a
4	a	a	a	a

5. Uruchom program kliknięciem ikony (rysunek 6.21).

RYSUNEK 6.21.

Kliknięcie ikony spowoduje uruchomienie programu



6. Program został wykonany.

7. Wyświetl okno arkusza Excel (rysunek 6.22). Z komórek z zakresu od A1 do C3 została usunięta zawartość.

RYSUNEK 6.22.

Z komórek z zakresu od A1 do C3 została usunięta zawartość, ale nie formatowanie

	A	B	C	D
1				a
2				a
3				a
4	a	a	a	a

Usuwanie formatowania

Zastąpienie dotychczasowego formatowania przez domyślne formatowanie komórek wymaga zaznaczenia zakresu komórek i posłużenia się metodą `ClearFormats`.

Przykład 100.

Napisz program, którego zastosowanie pozwoli przywrócić domyślne formatowanie komórek w zakresie od A1 do C3.

1. Wyświetl okno *Code*.

2. Wpisz w nim kod programu (rysunek 6.23). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W linii drugiej znajduje się nazwa procedury.

W linii trzeciej znajduje się komenda usuwania formatowania komórek arkusza leżących w zakresie od A1 do C3.

RYSUNEK 6.23.

Program, którego zastosowanie pozwoli usunąć formatowanie komórek w ograniczonej obszarze

```

Option Explicit

Sub przykład100()
    Range("A1:C3").ClearFormats
End Sub

```

- Wyświetl okno arkusza Excel. Domyślnie wszystkie komórki są puste. Wpisz dane do komórek (rysunek 6.24).

RYSUNEK 6.24.

Arkusz z danymi

	A	B	C	D
1	a	a	a	a
2	a	a	a	a
3	a	a	a	a
4	a	a	a	a

- Wyświetl okno edytora.
- Uruchom program. Naciśnij klawisz F5.
- Program został wykonany. Wyświetl okno arkusza Excel (rysunek 6.25). Z komórek z zakresu od A1 do C3 zostało usunięte formatowanie.

RYSUNEK 6.25.

Z komórek z zakresu od A1 do C3 zostało usunięte formatowanie, ale nie dane

	A	B	C	D
1	a	a	a	a
2	a	a	a	a
3	a	a	a	a
4	a	a	a	a

Usuwanie wartości mniejszych od progowej

Do usunięcia zawartości komórek służy metoda `ClearContents`. Użycie jej tylko w stosunku do komórek o wartości mniejszej od progowej wymaga sprawdzania wartości komórek o danym zakresie.

Przykład 101.

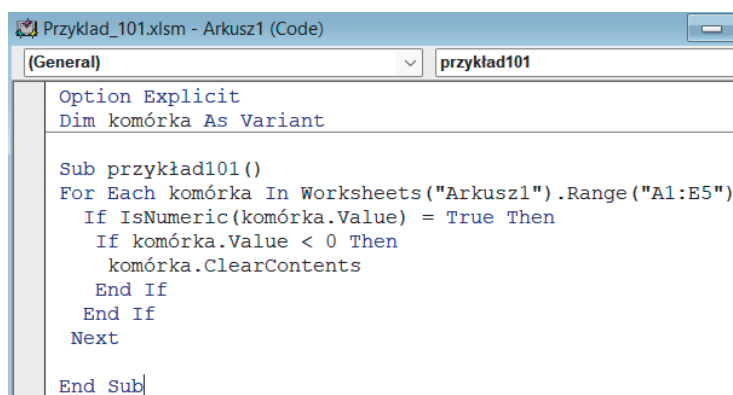
Napisz program, którego zastosowanie pozwoli przeszukiwać *Arkusz1* w zakresie od A1 do E5. Zawartość komórek o wartościach mniejszych od 0 będzie usuwana.

- Wyświetl okno *Code*.

- Wpisz w nim kod programu (rysunek 6.26). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W linii drugiej została zadeklarowana zmienna komórka. Ponieważ nie wiadomo, jakie dane może zawierać arkusz, został zadeklarowany typ Variant. W trzeciej linii znajduje się nazwa procedury. Za pomocą pętli For Each... Next będzie przeszukiwany zakres komórek od A1 do E5 arkusza Excela o nazwie Arkusz1. Przy użyciu metody ClearContents zawartość komórek o wartościach mniejszych od 0 jest usuwana.

RYСУNEK 6.26.

Program, którego zastosowanie pozwala usunąć komórki o wartości mniejszej od 0 Arkusza1 w zakresie od A1 do E5



```

Przykład_101.xlsm - Arkusz1 (Code)
[General] przykład101

Option Explicit
Dim komórka As Variant

Sub przykład101()
For Each komórka In Worksheets("Arkusz1").Range("A1:E5")
If IsNumeric(komórka.Value) = True Then
If komórka.Value < 0 Then
komórka.ClearContents
End If
End If
Next
End Sub

```

- Wyświetl okno arkusza Excel. W Arkusz1 wpisz dane (rysunek 6.27).

RYСУNEK 6.27.

Dane wpisane do Arkusza1

	A	B	C	D	E
1	-1	0	2	-33	
2	8	-1	0	4	
3	a	14.VII.1410	0,1	-0,02	
4	22		9	0.2	
5					

- Wyświetl okno edytora.
- Uruchom program. Naciśnij klawisz F5.
- Program został wykonany. Wyświetl okno arkusza Excel (rysunek 6.28). Komórki z zakresu od A1 do E5 o wartościach mniejszych od zera zostały pozbawione wartości. Ich formatowanie pozostało niezmienione.

RYСУNEK 6.28.

Z komórek z zakresu od A1 do E5 zostały usunięte dane o wartościach mniejszych niż zero

	A	B	C	D	E
1		0	2		
2	8		0	4	
3	a	14.VII.1410	0,1		
4	22		9	0.2	
5					

Właściwości

Właściwością jest cecha obiektu. Może nią być np. liczba wpisana w komórkę, kolor komórki itp.

Przypisanie wartości komórce

Właściwość, która jest przypisana komórce, zależy od wartości Value.

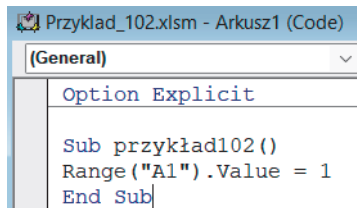
Przykład 102.

Napisz program, którego zastosowanie pozwoli wpisać liczbę 1 do komórki A1.

1. Wyświetl okno *Code*.
2. Wpisz w nim kod programu (rysunek 6.29). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W linii drugiej znajduje się nazwa procedury. W linii trzeciej zakresowi komórek jest przypisywana wartość. Jako zakres została podana jedna komórka, o adresie A1. Zostaje jej przypisana wartość 1.

RYSUNEK 6.29.

Do komórki A1
zostanie wpisana
wartość 1



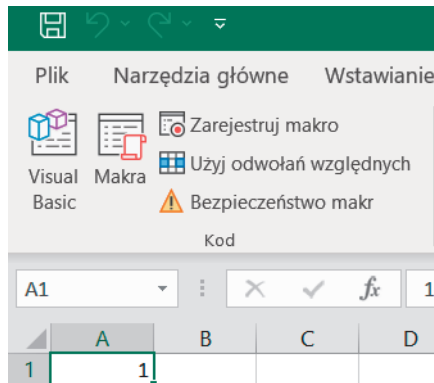
```
Przykład_102.xlsm - Arkusz1 (Code)
(General)
Option Explicit

Sub przykład102()
Range("A1").Value = 1
End Sub
```

3. Uruchom program. Naciśnij klawisz *F5*.
4. Program został wykonany. Do komórki A1 została wstawiona liczba 1 (rysunek 6.30).

RYSUNEK 6.30.

Arkusz z liczbą
wpisaną przez program.
Operacji nie można
cofnąć, ale po wydaniu
polecenia zamknięcia
arkusza program pyta
o zapisanie
wprowadzonych zmian



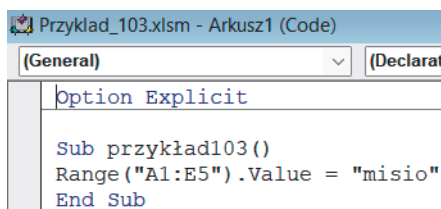
Przykład 103.

Napisz program, którego zastosowanie pozwoli wpisywać słowo misio do komórek w zakresie od A1 do E5.

1. Wyświetl okno *Code*.
2. Wpisz w nim kod programu (rysunek 6.31). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W linii drugiej znajduje się nazwa procedury. W linii trzeciej komórkom należącym do zakresu od A1 do E5 jest przypisywana wartość. Wartością jest ciąg znaków — słowo misio.

RYСУNEK 6.31.

Komórki należące do zakresu od A1 do E5 zostaną wypełnione identycznym ciągiem znaków



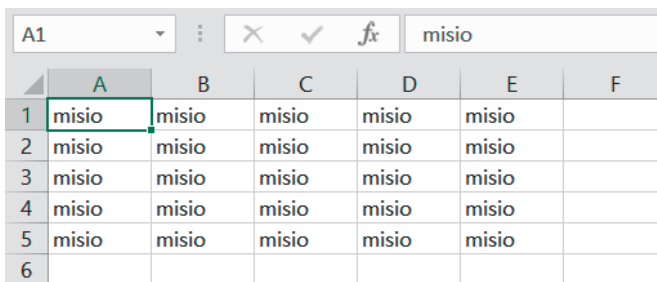
```
Option Explicit

Sub przyklad103()
Range("A1:E5").Value = "misio"
End Sub
```

3. Uruchom program. Naciśnij klawisz *F5*.
4. Program został wykonany. Do komórek od A1 do E5 został wstawiony ciąg znaków misio (rysunek 6.32).

RYСУNEK 6.32.

Rezultatem wykonania programu jest błyskawiczne „rozmnóżenie” słowa „misio”



	A	B	C	D	E	F
1	misio	misio	misio	misio	misio	
2	misio	misio	misio	misio	misio	
3	misio	misio	misio	misio	misio	
4	misio	misio	misio	misio	misio	
5	misio	misio	misio	misio	misio	
6						



Uwaga

Zmian w arkuszu wprowadzonych przez uruchomiony program nie można cofnąć w prosty sposób, np. naciskając klawisze *Ctrl+Z*.

Przykład 104.

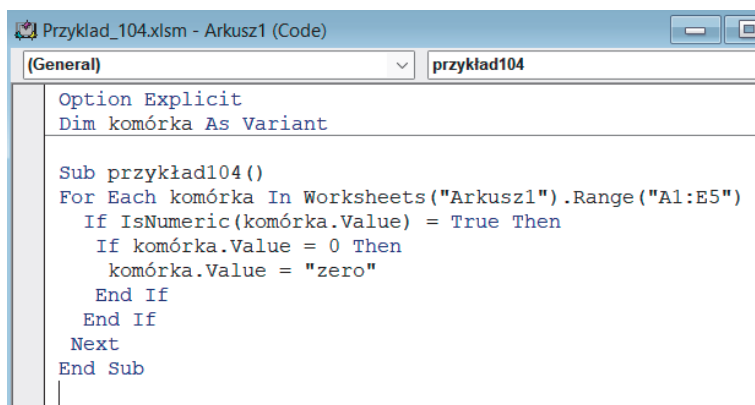
Napisz program, który będzie przeszukiwał *Arkusz1* w zakresie od A1 do E5. W komórkach o wartościach równych zero będzie wpisywane słowo zero.

1. Wyświetl okno *Code*.
2. Wpisz w nim kod programu (rysunek 6.33). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W drugiej linii została zadeklarowana zmienna komórka. Ponieważ nie wiadomo, jakie dane może zawierać arkusz, został zadeklarowany typ *Vari ant*. W trzeciej linii znajduje się nazwa procedury.

Za pomocą pętli For Each... Next będzie przeszukiwany zakres komórek od A1 do E5 arkusza programu Excel o nazwie *Arkusz1*. Przy użyciu metody Value do komórek o wartościach równych 0 będzie wpisywane słowo zero.

RYSUNEK 6.33.

Program, za pomocą którego można zastąpić wartość liczbową jej słownym odpowiednikiem



```

Option Explicit
Dim komórka As Variant

Sub przykład104()
For Each komórka In Worksheets("Arkusz1").Range("A1:E5")
If IsNumeric(komórka.Value) = True Then
If komórka.Value = 0 Then
komórka.Value = "zero"
End If
End If
Next
End Sub

```

3. Wyświetl okno arkusza Excel. W *Arkusz1* wpisz dane (rysunek 6.34).

RYSUNEK 6.34.

Dane wpisane do *Arkusza1*

	A	B	C	D	E
1	-1	0	2	-33	
2	8	-1	0	4	
3	a	14.VII.1410	0,1	-0,02	
4	22		9	0.2	
5					

4. Wyświetl okno edytora.

5. Uruchom program. Naciśnij klawisz F5.

6. Program został wykonany. Wyświetl okno arkusza Excel (rysunek 6.35). W komórki z zakresu od A1 do E5 o wartościach równych 0 został wpisany wyraz zero.

RYSUNEK 6.35.

Oto dowód na to, że wartość komórek pustych wynosi 0

	A	B	C	D	E
1	-1	zero	2	-33	zero
2	8	-1	zero	4	zero
3	a	14.VII.1410	0,1	-0,02	zero
4	22	zero	9	0.2	zero
5	zero	zero	zero	zero	zero

Przykład 105. — do samodzielnego wykonania

Napisz program, za pomocą którego można będzie przeszukiwać *Arkusz1* w zakresie od A1 do E5. W komórkach, w których znajduje się słowo zero, zostanie wpisana liczba 0.

Przykład 106.

Napisz program, za pomocą którego będzie można przeszukiwać *Arkusz1* w zakresie od *A1* do *E5*. W komórkach, w których znajduje się liczba, zostanie wpisana jej wartość słownie.

1. Wyświetl okno *Code*.
2. Wpisz w nim kod programu (rysunek 6.36). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W drugiej linii została zadeklarowana zmienna komórka. Ponieważ nie wiadomo, jakie dane może zawierać arkusz, został zadeklarowany typ *Variant*. W trzeciej linii znajduje się nazwa procedury. Za pomocą pętli *For Each... Next* będzie przeszukiwany zakres komórek od *A1* do *E5* arkusza Excel o nazwie *Arkusz1*. Przy użyciu metody *Value* do komórek o wartościach równych 0 będzie wpisywane słowo zero. Analogiczny warunek jest sprawdzany dla liczb o wartościach do 9 włącznie. Niespełnienie warunku powoduje, że będzie wykonywana kolejna linia programu. Spełnienie warunku powoduje, że do komórki będzie wpisywany ciąg znaków.
3. Wyświetl okno arkusza Excel. W *Arkusz1* wpisz dane (rysunek 6.37).
4. Wyświetl okno edytora.
5. Uruchom program. Naciśnij klawisz *F5*.
6. Program został wykonany. Wyświetl okno arkusza Excel (rysunek 6.38). W komórki z zakresu od *A1* do *E5* zostały wpisane wartości liczbowe wyrażone słownie.

Kopiowanie zawartości komórek

Kopiowanie zawartości komórek polega na przypisaniu komórkom należącym do jednego zakresu wartości komórek znajdujących się w innym zakresie. Operacja kopiowania jest możliwa dzięki nadaniu komórkom w obu zakresach identycznej właściwości *Value*.

Przykład 107.

Napisz program, za pomocą którego będzie można kopiować zawartość obszaru od *A1* do *A3* do obszaru od *B1* do *B3*.

1. Wyświetl okno *Code*.
2. Wpisz w nim kod programu (rysunek 6.39). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W drugiej linii znajduje się nazwa procedury. W trzeciej linii obiektowi, którym jest zakres komórek od *B1* do *B3*, została przypisana zawartość komórek z zakresu od *A1* do *A3*.
3. Wyświetl arkusz Excela.
4. W komórki od *A1* do *A3* wpisz odpowiednio: 1, 2, 3 (rysunek 6.40).

RYSUNEK 6.36.

Program, za pomocą którego można zastąpić wartość liczbową z przedziału od 0 do 9 jej słownym odpowiednikiem

```

Option Explicit
Dim komórka As Variant
Sub przykład106()
For Each komórka In Worksheets("Arkusz1").Range("A1:E5")
If IsNumeric(komórka.Value) = True Then
If komórka.Value = 0 Then
komórka.Value = "zero"
End If
If komórka.Value = 1 Then
komórka.Value = "jeden"
End If
If komórka.Value = 2 Then
komórka.Value = "dwa"
End If
If komórka.Value = 3 Then
komórka.Value = "trzy"
End If
If komórka.Value = 4 Then
komórka.Value = "cztery"
End If
If komórka.Value = 5 Then
komórka.Value = "pięć"
End If
If komórka.Value = 6 Then
komórka.Value = "sześć"
End If
If komórka.Value = 7 Then
komórka.Value = "siedem"
End If
If komórka.Value = 8 Then
komórka.Value = "osiem"
End If
If komórka.Value = 9 Then
komórka.Value = "dziewięć"
End If
End If
Next
End Sub

```

RYSUNEK 6.37.

Dane wpisane do Arkusza1

	A	B	C	D	E
1	1				
2				3	
3					
4		2			
5					

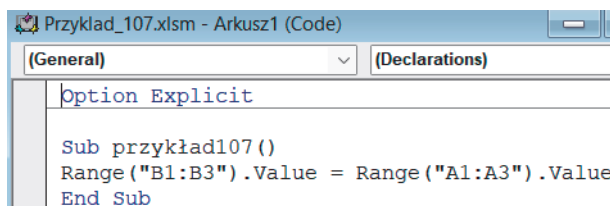
RYSUNEK 6.38.

Wartości liczbowe zostały wyrażone słownie

	A	B	C	D	E
1	jeden	zero	zero	zero	zero
2	zero	zero	zero	trzy	zero
3	zero	zero	zero	zero	zero
4	zero	dwa	zero	zero	zero
5	zero	zero	zero	zero	zero

RYSUNEK 6.39.

Program, za pomocą którego będzie można zakresowi komórek od B1 do B3 przypisać wartość komórek od A1 do A3



```

Option Explicit

Sub przykład107()
Range("B1:B3").Value = Range("A1:A3").Value
End Sub

```

RYSUNEK 6.40.

Arkusz z zestawem danych

	A	B	C	D
1	1			
2	2			
3	3			

- Wyświetl okno edytora VBA.
- Naciśnij klawisz *F5*.
- Program został wykonany. Wyświetl arkusz Excela.
- Do komórek od B1 do B3 została skopiowana zawartość komórek od A1 do A3 (rysunek 6.41).

RYSUNEK 6.41.

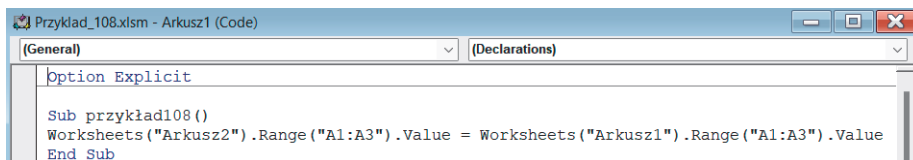
W komórkach od B1 do B3 została wpisana zawartość komórek od A1 do A3

	A	B	C	D
1	1	1		
2	2	2		
3	3	3		

Przykład 108.

Napisz program, za pomocą którego będzie można kopiować zawartość obszaru od A1 do A3 z arkusza 1. do obszaru od A1 do A3 arkusza 2.

- Wyświetl okno *Code*.
- Wpisz w nim kod programu (rysunek 6.42). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W linii drugiej znajduje się nazwa procedury. W linii trzeciej obiektowi, którym jest zakres komórek od B1 do B3, została przypisana zawartość komórek z zakresu od A1 do A3.



```

Option Explicit

Sub przykład108()
Worksheets("Arkusz2").Range("A1:A3").Value = Worksheets("Arkusz1").Range("A1:A3").Value
End Sub

```

RYSUNEK 6.42. Program, za pomocą którego będzie można pomiędzy arkuszami 1. i 2. kopiować wartość komórek od A1 do A3

3. Wyświetl arkusz Excela.

4. W komórki od A1 do A3 wpisz odpowiednio: 1, 2, 3 (rysunek 6.43).

RYСУNEK 6.43.

Arkusz 1. z zestawem danych

	A	B	C	D
1	1			
2	2			
3	3			
4				

Arkusz1 Arkusz2 Arkusz3

5. Wyświetl okno edytora VBA.

6. Naciśnij klawisz F5.

7. Program został wykonany. Wyświetl arkusz Excela i arkusz 2.

8. Do komórek od A1 do A3 arkusza 2. została skopiowana zawartość komórek od A1 do A3 arkusza 1. (rysunek 6.44).

RYСУNEK 6.44.

Arkusz 2. z zestawem danych skopiowanych z arkusza 1.

	A	B	C	D
1	1			
2	2			
3	3			
4				

Arkusz1 **Arkusz2** Arkusz3



Uwaga

Arkusz docelowy musi istnieć przed uruchomieniem programu. W przeciwnym razie wyświetlone zostanie okno z komunikatem błędu.

Microsoft Visual Basic for Applications ×



Subscript out of range

OK

Pomoc

Nadawanie komórce koloru z użyciem nazwy koloru

Aby nadać komórce kolor, wykorzystując nazwę koloru, należy posłużyć się właściwością `Interior.Color`.

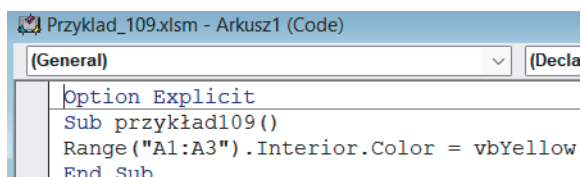
Przykład 109.

Napisz program, za pomocą którego będzie można nadawać komórkom z obszaru od A1 do A3 kolor żółty.

1. Uruchom arkusz kalkulacyjny Excel. Sprawdź, jaki kolor mają komórki. Domyślnie nie mają one żadnego wypełnienia.
2. Wyświetl okno *Code*.
3. Wpisz w nim kod programu (rysunek 6.45). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W drugiej linii znajduje się nazwa procedury. W trzeciej linii obiektowi, którym jest zakres komórek od A1 do A3, zostaje przypisany kolor żółty.

RYСУNEK 6.45.

Komórki od A1 do A3
będą miały kolor żółty

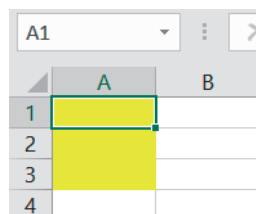


```
Option Explicit
Sub przykład109 ()
Range ("A1:A3").Interior.Color = vbYellow
End Sub
```

4. Uruchom program. Naciśnij klawisz *F5*.
5. Program został wykonany. Wyświetl arkusz kalkulacyjny Excel.
6. Kolor komórek od A1 do A3 został zmieniony (rysunek 6.46).

RYСУNEK 6.46.

Komórki, których kolor
został zmieniony



Uwaga

W tabeli 6.1 zebrano stałe, których przypisanie właściwości *Interior* powoduje wyświetlenie tła komórki w określonym kolorze.

TABELA 6.1. Stałe odpowiadające najczęściej używanym kolorom

Stała	Kolor
vbBlack	Czarny
vbRed	Czerwony
vbGreen	Zielony
vbYellow	Żółty
vbBlue	Niebieski

TABELA 6.1. Stałe odpowiadające najczęściej używanym kolorom (ciąg dalszy)

Stała	Kolor
vbMagenta	Fioletowy
vbCyan	Zielononiebieski
vbWhite	Biały

Nadawanie komórce koloru z użyciem kodu koloru

Szersze możliwości nadawania komórkom koloru stwarza posługiwanie się kodem koloru. Aby przypisać komórce kod koloru, należy posłużyć się właściwością `ColorIndex`.

Przykład 110.

Napisz program, którego zastosowanie pozwoli nadawać komórkom z obszaru od *A1* do *A56* kolory z palety barw dostępnej za pośrednictwem właściwości `ColorIndex`.

1. Uruchom arkusz kalkulacyjny Excel. Sprawdź, jaki kolor mają komórki. Domyślnie nie mają one żadnego wypełnienia.
2. Wyświetl okno *Code*.
3. Wpisz w nim kod programu:

```
Option Explicit
Sub przyk1ad110()
Range("A1").Interior.ColorIndex = 1
Range("A2").Interior.ColorIndex = 2
Range("A3").Interior.ColorIndex = 3
Range("A4").Interior.ColorIndex = 4
Range("A5").Interior.ColorIndex = 5
Range("A6").Interior.ColorIndex = 6
Range("A7").Interior.ColorIndex = 7
Range("A8").Interior.ColorIndex = 8
Range("A9").Interior.ColorIndex = 9
Range("A10").Interior.ColorIndex = 10
Range("A11").Interior.ColorIndex = 11
Range("A12").Interior.ColorIndex = 12
Range("A13").Interior.ColorIndex = 13
Range("A14").Interior.ColorIndex = 14
Range("A15").Interior.ColorIndex = 15
Range("A16").Interior.ColorIndex = 16
Range("A17").Interior.ColorIndex = 17
Range("A18").Interior.ColorIndex = 18
Range("A19").Interior.ColorIndex = 19
Range("A20").Interior.ColorIndex = 20
Range("A21").Interior.ColorIndex = 21
Range("A22").Interior.ColorIndex = 22
Range("A23").Interior.ColorIndex = 23
Range("A24").Interior.ColorIndex = 24
Range("A25").Interior.ColorIndex = 25
Range("A26").Interior.ColorIndex = 26
Range("A27").Interior.ColorIndex = 27
```

```

Range("A28").Interior.ColorIndex = 28
Range("A29").Interior.ColorIndex = 29
Range("A30").Interior.ColorIndex = 30
Range("A31").Interior.ColorIndex = 31
Range("A32").Interior.ColorIndex = 32
Range("A33").Interior.ColorIndex = 33
Range("A34").Interior.ColorIndex = 34
Range("A35").Interior.ColorIndex = 35
Range("A36").Interior.ColorIndex = 36
Range("A37").Interior.ColorIndex = 37
Range("A38").Interior.ColorIndex = 38
Range("A39").Interior.ColorIndex = 39
Range("A40").Interior.ColorIndex = 40
Range("A41").Interior.ColorIndex = 41
Range("A42").Interior.ColorIndex = 42
Range("A43").Interior.ColorIndex = 43
Range("A44").Interior.ColorIndex = 44
Range("A45").Interior.ColorIndex = 45
Range("A46").Interior.ColorIndex = 46
Range("A47").Interior.ColorIndex = 47
Range("A48").Interior.ColorIndex = 48
Range("A49").Interior.ColorIndex = 49
Range("A50").Interior.ColorIndex = 50
Range("A51").Interior.ColorIndex = 51
Range("A52").Interior.ColorIndex = 52
Range("A53").Interior.ColorIndex = 53
Range("A54").Interior.ColorIndex = 54
Range("A55").Interior.ColorIndex = 55
Range("A56").Interior.ColorIndex = 56
End Sub

```

4. Uruchom program.
5. Wyświetl okno arkusza Excel.
6. W komórkach od A1 do A56 zostały wyświetlone próbki kolorów (rysunek 6.47).

Przykład 111.

Napisz program, którego zastosowanie pozwoli nadawać komórkom z obszaru od A1 do A56 kolory z palety barw dostępnej za pośrednictwem właściwości ColorIndex. Wykorzystaj instrukcję pętli.

1. Uruchom arkusz kalkulacyjny Excel. Sprawdź, jaki kolor mają komórki. Domyślnie nie mają one żadnego wypełnienia.
2. Wyświetl okno *Code*.
3. Wpisz w nim kod programu (rysunek 6.48). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W drugiej linii nazwa procedury Dim wiersz As Integer jest deklaracją zmiennej o nazwie wiersz typu Integer. Zmienna będzie przechowywała wartości licznika pętli. W linii For wiersz=1 To 56 została zdefiniowana pętla. Przy pierwszym wykonaniu pętli wartość zmiennej wiersz równa się 1. Ponieważ nie została określona wartość kroku pętli, jest przyjmowana wartość domyślna, czyli 1.

RYSUNEK 6.47.

Numer wiersza
odpowiada wartości
parametru *ColorIndex*

	A
1	
2	
3	Red
4	Green
5	Blue
6	Yellow
7	Pink
8	Cyan
9	DarkRed
10	DarkGreen
11	DarkBlue
12	Olive
13	Purple
14	Teal
15	Grey
16	DarkGrey
17	LightBlue
18	Maroon
19	LightYellow
20	LightCyan
21	DarkPurple
22	LightPink
23	DarkBlue
24	LightBlue
25	DarkBlue
26	Pink
27	Yellow
28	Cyan
29	Purple

RYSUNEK 6.48.

Zamiast definiowania
56 operacji przypisania
wartości, jak
w poprzednim
przykładzie,
można posłużyć się
instrukcją pętli

```
Przyklad_111.xlsm - Arkusz1 (Code)
(General) (Declarations)
Option Explicit
Dim i As Integer
Sub przyklad111()
    Dim wiersz As Integer
    For wiersz = 1 To 56
        Cells(wiersz, 1).Interior.ColorIndex = wiersz
    Next wiersz
End Sub
```

Wartość zmiennej *wiersz* jest zwiększana o wartość 1 przy każdym wykonaniu pętli. Pętla jest wykonywana ostatni raz, gdy wartość tej zmiennej osiągnie 56. Zmienna *wiersz* przybiera wartości numeryczne od 1 do 56 i pełni funkcję licznika pętli. Po podstawieniu do linii `Cells(wiersz, 1).Interior.ColorIndex = wiersz` określa adres komórki i przypisany jej kod koloru. W przykładzie stały numer ma kolumna. Numer wiersza zaś jest określane przez zmienną *wiersz* i przy każdym wykonaniu

pętli jest zwiększany o jeden. Instrukcja Next wiersz powoduje zapętlenie. W instrukcji Next nazwa licznika (wiersz) nie jest wymagana. Dzięki podaniu nazwy licznika w instrukcji kończącej pętlę staje się ona bardziej czytelna.

4. Uruchom program.
5. Wyświetl okno arkusza Excel.
6. W komórkach od A1 do A56 zostały wyświetlone próbki kolorów (rysunek 6.49).

RYСУNEK 6.49.

Rezultat identyczny jak na rysunku 6.47 osiągnięto znacznie mniejszym nakładem pracy



Nadawanie koloru zawartości komórki

Kolorowanie tła komórek nie wyczerpuje wszystkich możliwości wyróżniania kolorem obszarów arkusza. Ciekawy efekt można uzyskać, zmieniając kolor czcionki. Aby przypisać czcionce kod koloru, należy posłużyć się właściwością Font.

Przykład 112.

Napisz program, którego zastosowanie pozwoli nadawać czcionkom z obszaru od A1 do A56 kolory z palety barw dostępnej za pośrednictwem właściwości `ColorIndex`. Wykorzystaj instrukcję pętli.

1. Uruchom arkusz kalkulacyjny Excel. Wpisz tekst do komórek od A1 do A56 (rysunek 6.50).

RYСУNEK 6.50.

Domyślnie tekst
ma kolor czarny

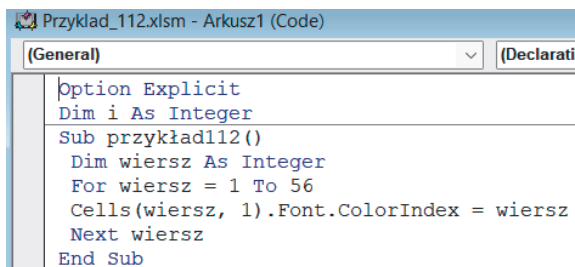
	A
1	tekst
2	tekst
3	tekst
4	tekst
5	tekst
6	tekst
7	tekst
8	tekst
9	tekst
10	tekst
11	tekst
12	tekst
13	tekst
14	tekst
15	tekst
16	tekst
17	tekst
18	tekst
19	tekst
20	tekst
21	tekst
22	tekst
23	tekst
24	tekst
25	tekst
26	tekst
27	tekst
28	tekst
29	tekst

2. Wyświetl okno *Code*.
3. Wpisz w nim kod programu (rysunek 6.51). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W drugiej linii nazwa procedury `Dim wiersz As Integer` jest deklaracją zmiennej o nazwie `wiersz` typu `Integer`. Zmienna będzie przechowywała wartości licznika pętli. W linii `For wiersz=1 To 56` została zdefiniowana pętla. Przy pierwszym wykonaniu pętli wartość zmiennej `wiersz` równa się 1. Ponieważ nie została określona wartość kroku pętli, jest przyjmowana

wartość domyślna, czyli 1. Wartość zmiennej wiersz jest zwiększana o wartość 1 przy każdym wykonaniu pętli. Pętla jest wykonywana ostatni raz, gdy wartość tej zmiennej osiągnie 56. Zmienna wiersz przybiera wartości numeryczne od 1 do 56 i pełni funkcję licznika pętli. Po podstawieniu do linii `Cells(wiersz, 1)` ↪ `.Font.ColorIndex=wiersz` określa adres komórki i przypisany jej kod koloru. W przykładzie stały numer ma kolumna. Numer wiersza zaś jest określany przez zmienną wiersz i przy każdym wykonaniu pętli zwiększany o jeden. Instrukcja `Next wiersz` powoduje zapętlenie. W instrukcji `Next` nazwa licznika (`wiersz`) nie jest wymagana. Dzięki podaniu nazwy licznika w instrukcji kończącej pętlę staje się ona bardziej czytelna.

RYСУNEK 6.51.

Program, którego zastosowanie pozwoli zamienić napisy czarne w wielobarwne



```
Przyklad_112.xlsm - Arkusz1 (Code)
(General) (Declarati
Option Explicit
Dim i As Integer
Sub przyklad112()
    Dim wiersz As Integer
    For wiersz = 1 To 56
        Cells(wiersz, 1).Font.ColorIndex = wiersz
    Next wiersz
End Sub
```

4. Uruchom program.
5. Wyświetl okno arkusza Excel.
6. W komórkach od `A1` do `A56` zostały wyświetlone próbki kolorów (rysunek 6.52).

Przesuwanie aktywnej komórki

W arkuszu kalkulacyjnym musi zostać wybrana przynajmniej jedna komórka. W niej będą się np. pojawiały znaki wpisywane z klawiatury. Jest ona wyróżniona grubszą obwódką.

Jest możliwe pozostawienie zaznaczenia jednej komórki i uczynienie aktywną innej. Jest to wykonalne dzięki właściwości `Offset`. Ma ona następującą składnię:

`Offset(x,y)`

gdzie:

- `x` — przesunięcie w poziomie,
- `y` — przesunięcie w pionie.

Wartości ujemne pozwalają odpowiednio na przesunięcie w lewo lub w górę. Wartości dodatnie pozwalają odpowiednio na przesunięcie w prawo lub w dół.

Oczywiście istnieje również możliwość sterowania z poziomu programu VBA zaznaczeniem komórki. Umożliwia to właściwość obiektu `Application` — `ActiveCell`.

RYSUNEK 6.52.

Fragment arkusza
z rysunku 6.50
po zmianie kolorów
czcionki

	A
1	tekst
2	
3	tekst
4	tekst
5	tekst
6	tekst
7	tekst
8	tekst
9	tekst
10	tekst
11	tekst
12	tekst
13	tekst
14	tekst
15	tekst
16	tekst
17	tekst
18	tekst
19	tekst
20	tekst
21	tekst
22	tekst
23	tekst
24	tekst
25	tekst
26	tekst
27	tekst
28	tekst
29	tekst

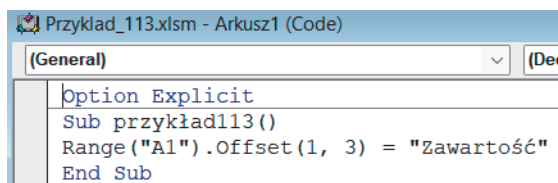
Przykład 113.

Napisz program, którego zastosowanie pozwoli przesuwać aktywną komórkę o jedną komórkę w dół i trzy komórki w prawo, a następnie wstawić do niej tekst Zawartość.

1. Wyświetl okno *Code* (rysunek 4.3).
2. Wpisz w nim kod programu (rysunek 6.53). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W drugiej linii jest widoczna nazwa procedury. W trzeciej linii znajduje się instrukcja zmiany aktywnej komórki. Punktem odniesienia jest komórka *A1*. Właściwość *Offset* pozwala na przesunięcie zaznaczenia o jedną komórkę w dół i trzy komórki w prawo. Ciąg znaków widoczny po znaku równości zostanie wstawiony do zaznaczonej komórki.
3. Uruchom program.
4. Wyświetl okno arkusza Excel.

RYSUNEK 6.53.

Program, którego zastosowanie pozwoli na wstawienie napisu do komórki, która nie jest zaznaczona



```
Przykład_113.xlsm - Arkusz1 (Code)
[General]
Option Explicit
Sub przykład113()
Range("A1").Offset(1, 3) = "Zawartość"
End Sub
```

5. W komórce D2 pojawił się napis (rysunek 6.54).

RYSUNEK 6.54.

Jest zaznaczona komórka A1. Treść została wpisana do komórki odległej o zadane przesunięcie

	A	B	C	D
1				
2				Zawartość

Przykład 114. — do samodzielnego wykonania

Napisz program, którego zastosowanie pozwoli przesuwać aktywną komórkę o trzy komórki w dół i jedną w prawo, a następnie wstawić do niej tekst Zawartość.

Przykład 115.

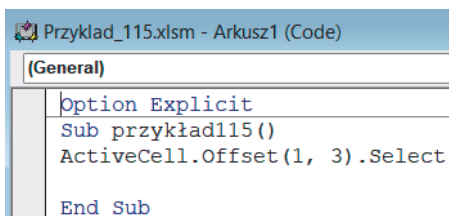
Napisz program, którego zastosowanie pozwoli przesuwać zaznaczenie komórki o jedną komórkę w dół i trzy komórki w prawo.

1. Wyświetl okno *Code*.

2. Wpisz w nim kod programu (rysunek 6.55). Pierwsza linia zawiera wymuszenie deklaracji wszystkich zmiennych. W drugiej linii jest widoczna nazwa procedury. W trzeciej linii znajduje się instrukcja wyboru zaznaczenia komórki. Punktem odniesienia jest bieżąca komórka. Właściwość `Offset(1, 3)` powoduje przesunięcie zaznaczenia względem aktualnego o jedną komórkę w dół i trzy komórki w prawo. Metoda `Offset` pozwala na wybranie komórki i oznaczenie jej pogrubioną ramką.

RYSUNEK 6.55.

Program, którego zastosowanie pozwala przesuwać zaznaczenie komórki o jedną komórkę w dół i trzy komórki w prawo



```
Przykład_115.xlsm - Arkusz1 (Code)
[General]
Option Explicit
Sub przykład115()
ActiveCell.Offset(1, 3).Select
End Sub
```

3. Wyświetl arkusz kalkulacyjny Excel. Sprawdź, czy komórka A1 jest zaznaczona. Jeżeli nie — zaznacz ją (rysunek 6.56).

4. Uruchom program. Naciśnij klawisz *F5*.

5. Wyświetl okno arkusza Excel (rysunek 6.57).

RYSUNEK 6.56.
Komórka zaznaczona
domyślnie

	A	B	C	D	E
1					
2					
3					
4					
5					

RYSUNEK 6.57.
Jest zaznaczona
komórka D2

	A	B	C	D	E
1					
2					
3					
4					
5					

Przykład 116. — do samodzielnego wykonania

Napisz program, który będzie „pytał” o współrzędne zaznaczonej komórki, a następnie „przesuwał” zaznaczanie zgodnie z wprowadzonymi danymi.

Podsumowanie

- ♦ Obiektem jest element składowy aplikacji, np. komórka, zakres komórek, wykres umieszczony na arkuszu, arkusz, skoroszyt.
- ♦ Obiekty mają właściwości. Właściwość jest cechą obiektu. Właściwościami zakresu komórek są: zawartość komórki, styl czcionki itp.
- ♦ Metoda jest operacją wykonaną na obiekcie, umożliwia np. wpisanie tekstu do komórki lub zaznaczenie zakresu komórek.

Skorowidz

A

Abs, 322
Activate, 314
AddComment, 314
ADO, 315
adres
 komórki, 150
 obiektu zmiennej, 334
 właściwości, 334
adresowanie
 bezwzględne, 47, 49
 względne, 47
AdvancedFilter, 314
amortyzacja środka trwałego,
 326
And, 341
Anuluj, 159, 160, 162
API, 12
AppActivate, 331
AppleScript, 325
ApplyNames, 314
ApplyOutlineStyles, 314
Arccos, 322
Arccosec, 322
Arccotan, 322
Arcsec, 322
Arcsin, 322
Asc, 322
Atn, 322
atrybuty
 pliku, 323, 334
 katalogu, 323
AutoFill, 314
AutoFilter, 314
AutoFit, 314
AutoOutline, 314

B

BASIC, 9, 10
Beep, 331

bieżący
 katalog, 331
 napęd, 331
błąd
 obsługa, 333
 odpowiadający kodowi, 323
 opis, 172
 zdefiniowany przez
 użytkownika, 323
 źródło, 317
Boolean, 331
BorderAround, 315, 317
buttons, 155
Byte, 331

C

Calculate, 315
Call, 331
CallByName, 322
carriage return, 144
CBool, 322
CByte, 322
CCur, 322
CDate, 322
CDBl, 322
CDec, 322
ChDir, 331
ChDrive, 331
CheckSpelling, 315
Choose, 322
Chr, 322
ciąg bez końcowych spacji, 326
ciąg bez początkowych spacji,
 325
CInt, 322
Clear, 315
ClearComments, 315
ClearContents, 315
ClearFormats, 315
ClearNotes, 315
ClearOutline, 315
CLng, 322
Close, 331
ColorIndex, 321
ColumnDifferences, 315
COM, 12
Command, 322
Component Object Model, 12
Consolidate, 315
Const, 331
context, 144, 155
Copy, 315
CopyFromRecordset, 315
CopyPicture, 315
Cos, 322
Cosec, 322
cosecans hiperboliczny, 324
cosinus hiperboliczny, 324
Cotan, 322
cotangens hiperboliczny, 324
CreateNames, 315
CreateObject, 322
CSng, 322
CStr, 322
CurDir, 322
Currency, 331
Cut, 315
CVar, 322
CVer, 323
czas dla podanej godziny, 327
czas systemowy, 165
część
 całkowita liczby, 324
 daty, 323

D

dane
 eksportowanie, 315
 grupowanie, 316
 kopiowanie, 314
 scalenie, 315
 sortowanie, 317
 wpisywanie, 150
 wprowadzanie, 144

DAO, 315
 data
 i czas systemowy, 329
 i godzina, 323
 systemowa, 165, 323, 325, 331
 DataSeries, 315
 Date, 170, 323, 324, 331
 Date\$, 165
 DateAdd, 323
 DateDiff, 323
 DatePart, 323
 DateSerial, 323
 DateValue, 323
 Day, 323
 DDB, 323
 Decimal, 331
 Declare, 331
 default, 144
 DefBool, 331
 DefByt, 331
 DefCur, 331
 DefDate, 331
 DefDbt, 331
 DefDec, 331
 DefInt, 331
 DefLng, 331
 DefObj, 331
 DefSng, 331
 DefStr, 331
 DefVar, 331
 deklaracja, 334
 funkcji, 332
 nazw i argumentów
 procedury, 333
 zdarzenia, 332
 zmiennych, 331
 Delete, 315
 DeleteSetting, 331
 Deweloper, 35
 DialogBox, 315
 Dim, 331
 Dir, 323
 Dirty, 315
 DLL, 331
 długość
 pliku w bajtach, 323
 tekstu, 176
 Do Until, 152
 Do...Loop, 331
 dodawanie, 334
 DoEvents, 323

dolne ograniczenia indeksu
 tablicy, 333
 domyślny
 tryb porównywania ciągu,
 333
 typ danych, 331
 Double, 331
 drukowanie
 obiekt, 317
 dzielenie, 334
 scalonego obszaru, 317
 dzień
 miesiąca, 323
 tygodnia, 327
 dźwięk, 331

E

EditionOptions, 315
 Edytor Visual Basic, 11
 eksportowanie danych, 315
 End, 332
 End Sub, 45, 49
 Enum, 332
 Environ, 323
 EOF, 323
 Eqv, 341
 Erase, 332
 Error, 323, 324, 332
 Event, 332
 Example, 318
 Excel 5, 11
 Exit Do, 332
 Exit For, 332
 Exit Function, 332
 Exit Property, 332
 Exit Sub, 332
 Exp, 323
 ExportAsFixedFormat, 315

F

FileAttr, 323
 FileCopy, 332
 FileDateTime, 323
 FileLen, 323
 FillDown, 316
 FillLeft, 316
 FillRight, 316
 FillUp, 316
 Filter, 323
 filtrowanie listy, 314
 Find, 316
 FindNext, 316
 FindPrevious, 316

folder, atrybuty, 323
 For Each...Next, 332
 For...Next, 332
 Format, 323
 FormatCurrency, 323
 FormatDateTime, 323
 FormatNumber, 323
 FormatPercent, 323
 formaty, wyczyszczenie, 315
 formuła, wyczyszczenie, 315
 FreeFile, 323
 Function, 332
 FunctionWizard, 316
 funkcja
 arkusza, 322
 Or, 342
 wykładnicza, 323
 FV, 323

G

generator liczb losowych, 333
 Get, 332
 GetAllSettings, 323
 GetAttr, 323
 GetObject, 323
 GetSetting, 323
 godzina, 324
 godzina systemowa, 325
 ustawienie, 334
 GoSub...Return, 332
 GoTo, 172, 332
 Group, 316
 grupowanie danych, 316

H

HArccos, 323
 HArccosec, 324
 HArccota, 324
 HArcsec, 324
 HArctan, 324
 HCOs, 324
 HCosec, 324
 HCotan, 324
 helpfile, 144, 155
 Hex, 324
 Hour, 324
 HSec, 324
 HSin, 324
 HTan, 324

I

ID zadania, 326
 If...Then...Else, 332
 Ignoruj, 159
 IIf, 324
 iloczyn logiczny, 341
 Imp, 341
 Implements, 332
 implikacja, 341
 indeks tablicy, 327
 Input, 324
 Input #, 332
 InputBox, 144, 145, 147, 167, 168, 170, 324
 Insert, 316
 InsertIndent, 316
 InStr, 324
 InStrRev, 324
 instrukcje, 331, 332, 333, 334
 Int, Fix, 324
 Integer, 168, 172, 331
 interfejs, 332
 IPmt, 324
 IRR, 324
 IsArray, 324
 IsDate, 324
 IsEmpty, 324
 IsError, 324
 IsMissing, 324
 IsNull, 324
 IsNumeric, 324
 IsObject, 325

J

Join, 325
 Justify, 316

K

karta
 Deweloper, 54,
 Dodatki, 12
 katalog
 atrybuty, 323
 bieżący, 331
 utworzenie, 333
 zmiana nazwy, 333
 Kill, 332
 klasa, 332
 Klawisz skrótu, 25
 klucz w rejestrze Windows, 331
 kod
 ASCII, 322
 RGB koloru, 326
 znaku, 322

kolejny dostępny numer pliku,
 323
 kolor ramki, 321
 komentarz, 314, 333
 wyczyszczenie, 315
 komórka
 nazwa, 314
 odczytanie, 317
 scalenie, 316
 skopiowanie, 315
 uaktywnienie, 314
 wstawienie, 316
 wypełnianie, 316
 komunikat, 155, 156, 157
 o błędzie, 169
 koniec pliku tekstowego, 323
 kontynuowanie wyszukiwania,
 316
 kopiowanie
 danych, 314
 komórek, 23

L

LBound, 325
 LCase, 325
 Left, 325
 Len, 176, 325
 Let, 332
 liczba, 323, 324
 bajtów, 325
 decymalna, 327
 heksadecymalna, 327
 konwersja, 324
 losowa, 326
 okresów spłaty, 325
 reprezentująca czas, 327
 sekund od północy, 326
 spacji, 326
 zaokrąglona, 326
 znaków, 176, 325
 Line Input #, 333
 linefeed character, 144
 lista, filtrowanie, 314
 ListNames, 316
 Load, 333
 Loc, 325
 Lock, 333
 LOF, 325
 Log, 325
 logarytm naturalny, 325
 lokalna
 tablica, 333
 zmienna, 333
 Long, 331

Loop, 152
 LSet, 333
 LTrim, 177, 325

Ł

łańcuch znaków, 325, 326
 odwrócony, 326
 po konwersji, 326
 łączenie ciągów znaków, 339

M

MacID, 325
 MacScript, 325
 Makra, 25
 makropolecenia, 12
 blokowanie, 15
 edycja, 44
 przycisk, 38
 rejestrwanie, 24
 testowanie, 28
 uruchomienie, 317
 warunki początkowe, 22
 zapisywanie, 41
 małe litery, 325
 Merge, 316
 metoda, 318
 Mid, 175, 325, 333
 miesiąc, 325
 Minute, 325
 minuty, 325
 MIRR, 325
 MkDir, 333
 Mod, 335
 Modalność, 159
 moduł prywatny, 333
 Month, 325
 MonthName, 325
 msdn.microsoft.com, 318
 MsgBox, 155, 160, 167, 168, 170, 325

N

najmniejszy indeks tablicy, 325
 największy indeks tablicy, 327
 Name, 333
 napęd bieżący, 331
 napis nad przyciskiem, 163
 nazwa
 komórka, 314
 miesiąca, 325
 pliku lub katalogu, 323
 negacja, 341
 Nie, 159, 162

nierównoważność logiczna, 341
nieukryte nazwy, 316
Not, 341
notatki, wyczyszczenie, 315
Now, 325
NPer, 325
NPV, 325
Null, 324

O

obiekt
 drukowanie, 317
 OLE, 323
 Range, 317
 usunięcie, 315
 wycięcie, 315
 wyczyszczenie, 315
 zakres, 315, 317
Object, 331
obraz, skopiowanie, 315
obsługa
 błędów, 333
 zdarzeń, 323
obszar, dzielenie, 317
obwódnia, 317
obwódka, 319
Oct, 325
odczyt
 danych z otwartego pliku, 332
 danych z sekwencyjnego pliku, 332
 pojedynczej linii z otwartego pliku tekstowego, 333
 zawartości komórki, 317
odwołanie, 47
odwrotny cosecans
 hiperboliczny, 324
odwrotny cotangens
 hiperboliczny, 324
odwrotny secans
 hiperboliczny, 324
odwrotny sinus hiperboliczny, 324
odwrotny tangens
 hiperboliczny, 324
odwrócony łańcuch znaków, 326
okno
 aplikacji, 331
 dialogowe, 144, 166, 324
 komunikatu, 155, 156, 157, 325
 tytuł, 145
OLE, 322, 323, 325
On Error, 167, 333
On Error GoTo 0, 168

On Error GoTo line, 167
On Error Resume Next, 167
On...GoSub, 333
On...GoTos, 333
Open, 333
opis błędu, 172
Option Base, 333
Option Compare, 333
Option Explicit, 333
Option Private, 333
Or, 341
otworzenie pliku tekstowego, 333

P

Parameters, 318
Parse, 317
Partition, 325
PasteSpecial, 317
pętla, 331, 334
 programowa, 332
Phonetic, 317
pierwsza liczba w łańcuchu znaków, 327
pisownia, sprawdzanie, 315
plik
 .docm, 13
 .docx, 13
 .dotm, 13
 .dotx, 13
 .xlsm, 13
 .xlsx, 13
 .xltm, 13
 .xltx, 13
 atrybuty, 323, 334
 pozycja, 326
 tekstowy, 333
 zmiana nazwy, 333
Pmt, 325
podgląd wydruku, 317
podtyp zmiennej, 327
podzbiór tablicy łańcuchów znaków z uwzględnieniem filtra, 323
pole edycji, 166
połączenie łańcuchów, 325
pomoc, 162
 przycisk, 158
 wyświetlanie, 315, 317
porównanie łańcuchów znaków, 326
pozycja
 w pliku, 326
 wydruku, 326
 wystąpienia ciągu znaków, 324

PPmt, 325
Print #, 333
PrintOut, 317
PrintPreview, 317
Private, 333
procedura, 331
 obsługi błędów, 167, 333
 obsługi błędu, 333
 zewnętrzna, 331
procent, 323
program
 rozgałęzienie, 333
 zatrzymanie, 334
prompt, 144, 155
Property
 Get, 333
 Let, 333
 Set, 333
prośba o wpisanie ciągu znaków, 152
przedział czasu, 323
przekształcanie wyrażenia, 322
Przerwij, 159
przycisk, 159, 162
 OK, 160
 Pomoc, 158
 w oknie komunikatu, 156, 157
Public, 333
Put, 333
PV, 325

Q

QBColor, 325

R

RaiseEvent, 333
Randomize, 333
Range, 317
Rate, 325
ReDim, 333
rejestrwanie makra, 25
Rem, 333
Remarks, 318
RemoveDuplicates, 317
Replace, 325
Reset, 333
Resume, 333
Return value, 318
RGB, 326
Right, 174, 326
Rmdir, 333
Rnd, 326

rok, 327
 Round, 326
 RowDifferences, 317
 rozdzielenie zakresu danych, 317
 rozgałęzienie programu, 333
 rozsuniecie tekstu, 316
 rozszerzenie
 .docm, 13
 .docx, 13
 .dotm, 13
 .dotx, 13
 .xlsm, 13
 .xlsx, 13
 .xltm, 13
 .xltx, 13
 RSet, 334
 RTrim, 178, 326
 Run, 317

S

SaveSetting, 334
 scalenie
 danych, 315
 komórek, 316
 Sec, 326
 secans hiperboliczny, 324
 Second, 326
 Seek, 326, 334
 sekundy, 326
 Select, 317
 Select Case, 334
 SendKeys, 334
 seria danych, 315
 Set, 334
 SetAttr, 334
 SetPhonetic, 317
 Sgn, 326
 Shell, 326
 Show, 317
 ShowDependents, 317
 ShowErrors, 317
 Sin, 326
 Single, 331
 sinus hiperboliczny, 324
 składnia metody, 318
 skok
 bezwarunkowy, 332
 do podprogramu, 332
 skopiowanie
 ADO, 315
 DAO, 315
 komórek, 315
 obrazu, 315
 pliku, 332

skoroszyt
 makr osobistych, 26
 programu Excel, 43
 skrypt AppleScript, 325
 SLN, 326
 Sort, 317
 sortowanie danych, 317
 Space, 326
 Spc, 326
 Speak, 317
 SpecialCells, 317
 Split, 326
 sprawdzanie pisowni, 315
 Sqr, 326
 stała, 331
 Static, 334
 sterowanie dostępem, 333
 Stop, 334
 stopa zwrotu dla ciągu
 przepływów gotówkowych,
 324
 Str, 326
 StrComp, 326
 StrConv, 326
 String, 167, 326, 331
 StrReverse, 326
 Sub, 45, 49, 334
 SubscribeTo, 317
 suma logiczna, 341
 Switch, 326
 SYD, 326
 symulacja
 błędu, 332
 pisania z klawiatury, 334
 Syntax, 318
 szerokość
 kolumny, 314
 szerokość pliku, 334

Ś

ścieżka bieżącego katalogu, 322

T

Tab, 326
 tabela danych, 317
 Table, 317
 tablica, 324
 lokalna, 333
 zainicjowanie elementów, 332
 zmiana wymiarów, 333
 Tak, 159, 162
 Tan, 326
 tekst

 długość, 176
 rozsuniecie, 316
 TextToColumns, 317
 Time, 326, 334
 Time\$, 165
 Timer, 326
 TimeSerial, 327
 TimeValue, 327
 title, 144, 155
 Trim, 178
 Trim s, 327
 typ
 danych użytkownika, 334
 danych, 327
 zmiennej, 167
 Type, 334
 TypeName, 327
 tytuł okna, 145

U

uaktywnienie
 komórki, 314
 okna aplikacji, 331
 UBound, 327
 UCase, 327
 Ungroup, 317
 Unload, 334
 Unlocks, 333
 UnMerge, 317
 uruchomienie makropolecenia,
 317
 usunięcie
 folderu, 333
 katalogu, 333
 obiektu z pamięci, 334
 obiektu, 315
 pliku, 332
 zduplowanych obiektów, 317
 znaków białych, 177
 utworzenie
 katalogu, 333
 nazw na podstawie
 etykiet tekstowych, 315
 obiektów Phonetic, 317
 tabeli danych, 317
 w serii danych, 315

V

Val, 327
 Variant, 331
 VarType, 327
 VBA, 9, 10

vbAbortRetryIgnore, 156
 vbApplicationModal, 159
 vbCritical, 157
 vbDefaultButton, 162
 vbDefaultButton1, 158
 vbDefaultButton2, 158
 vbDefaultButton3, 159
 VbDefaultButton4, 159
 vbExclamation, 158
 vbInformation, 158
 vbMsgBoxHelpButton, 158, 162
 vbMsgBoxRight, 159
 vbMsgBoxRtlReading, 159
 vbOKCancel, 156
 vbOKOnly, 156
 vbQuestion, 157, 162
 vbRetryCancel, 157
 vbSystemModal, 159
 vbYesNo, 156
 vbYesNoCancel, 156, 162
 Visual Basic, 10
 Visual Basic for Applications, 9

W

wartość
 bezwzględna, 322
 ciągu wyrażeń logicznych, 326
 inwestycji, 325
 kapitału dla raty pożyczki, 325
 netto, 325
 właściwości, 322
 z listy argumentów, 322
 warunkowe
 wykonanie grupy instrukcji, 332
 wykonanie instrukcji, 334
 wczytanie adresu komórki, 149
 Weekday, 327
 WeekdayName, 327
 wejście aplikacji w rejestrze Windows, 323
 While...Wend, 334
 Width #, 334
 wielkie litery, 327
 With, 334
 Write #, 334
 wstawianie
 komórki, 316
 obiektu Range, 317
 wycięcia, 316

wstążka, 12
 wstrzymanie programu, 324
 wycięcie
 obiektów, 315
 wstawianie, 316
 wyczyszczenie
 formatu, 315
 formuły, 315
 komentarza, 315
 notatek, 315
 obiektu, 315
 wydruk, 317
 podgląd, 317
 wygenerowanie zdarzenia, 333
 wyjście
 z bloku pętli
 Do...Loop, 332
 For Each...Next, 332
 For...Next, 332
 z funkcji, 332
 z procedury
 Property, 332
 Sub, 332
 wymuszenie deklaracji zmiennych, 333
 wypełnianie komórek, 314, 316
 wyrównanie
 do lewej, 333
 do prawej, 334
 wyróżnienie, 314
 wysokość wiersza,
 dopasowanie, 314
 wyszukiwanie, 316
 wyświetlanie
 okna komunikatu, 155
 pomocy, 315, 317
 zaznaczonego zakresu, 317
 wywołanie metody, 318
 wznowienie wykonywania programu, 333

X

Xor, 341
 xpos, 144

Y

Year, 327
 ypos, 144

Z

zainicjowanie elementów tablicy, 332
 zakończenie bloku, 332
 procedury, 332
 programu, 332
 zakres, 325
 danych, 317
 obiekt, 315, 317
 wyróżnienie, 314
 zależność, 317
 załadowanie obiektu, 333
 zamiana
 ciągu znaków na datę, 323
 daty na liczbę, 323
 liczby na znaki, 326
 zamknięcie plików, 331, 333
 zapis
 danych do pliku, 334
 w pliku, 334
 w rejestrze Windows, 334
 zmiennej do pliku tekstowego, 333
 zastąpienie znaków, 333
 zatrzymanie programu, 334
 zaznaczenie, 314, 317
 zdublowane obiekty, 317
 zgodność wsteczna, 12
 zmiana
 nazwy pliku lub katalogu, 333
 wymiarów tablicy, 333
 zmienna
 lokalna, 333
 na poziomie procedury, 334
 środowiskowa systemu operacyjnego, 323
 znak, 322
 liczby, 326
 z łańcucha, 325
 z pliku, 324

Ź

źródło błędów, 317

PROGRAM PARTNERSKI

— GRUPY HELION —

- 
1. ZAREJESTRUJ SIĘ
 2. PREZENTUJ KSIĄŻKI
 3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

GRUPA
Helion

Excel z VBA — nowy poziom wydajności

- Odkryj język VBA i jego zastosowania
- Poznaj sztukę tworzenia makropoleceń
- Zautomatyzuj swoją pracę w Excelu

Masz dość wykonywania ciągle tych samych zadań? Musisz pilnie opracować kolejny raport dla szefa? Od wpatrywania się w tabele w Excelu Twoje oczy robią się kwadratowe? A może masz już dość bycia zwykłym użytkownikiem aplikacji i chcesz czegoś więcej? Świetnie! Wsparcie nadchodzi!

Excel to niezwykle popularny i uniwersalny program umożliwiający przeprowadzanie obliczeń naukowych, statystycznych, finansowych, analizę danych, wizualizację wyników i tworzenie rozbudowanych raportów. Ciekawie robi się jednak dopiero wtedy, gdy zachodzi potrzeba automatyzacji działań i prowadzenia interakcji z użytkownikami arkusza lub innymi aplikacjami pakietu Microsoft Office. Ciekawie, a może raczej trudno? Spokojnie, wszystko masz pod ręką, wystarczy skorzystać z tego, co oferuje język Visual Basic for Applications.

Jeśli chcesz wejść na wyższy poziom i dowiedzieć się, jak upraszczać i przyspieszać pracę z Excelem, sięgnij po tę książkę! Dzięki niej szybko i łatwo poznasz możliwości makr, metody ich tworzenia, konstrukcje języka VBA i różne sposoby ich zastosowania. Wiedza nie jest tu oderwana od rzeczywistości, lecz przekazywana w odniesieniu do praktycznych przykładów z życia. A to oznacza, że nauczysz się nie tylko skutecznie programować, lecz również rozwiązywać prawdziwe problemy, z którymi na co dzień mierzy się wielu użytkowników Excela. Ruszaj w drogę, przygoda czeka!

- Makropolecenia i język VBA w Excelu
- Korzystanie z edytora języka VBA
- Komunikacja z użytkownikiem aplikacji
- Zastosowanie zmiennych i obiektów arkusza
- Używanie instrukcji warunkowych i podprogramów
- Obsługa zdarzeń i korzystanie z zakresów arkusza
- Zastosowanie funkcji arkuszowych i elementów sterujących

Automatyzacja pracy w Excelu? Tylko z VBA!

	<p><i>Sprawdź nasze szkolenia!</i></p>  <p>SZKOLENIA AKADEMIA IT & BUSINESS</p>	<p>KOD KORZYŚCI <i>Sięgnij po więcej!</i> ▶</p>	
 helion.pl	<p>HELIONSZKOLENIA.PL</p>	<p>ISBN 978-83-283-8898-7</p>  <p>9 788328 388987</p>	
<p>INFORMATYKA W NAJLEPSZYM WYDANIU</p>		<p>Cena: 69,00 zł</p>	