

Witold Wrotek

VBA

dla **Excela 2019** 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 Wydawnictwo HELION 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 Wydawnictwo HELION 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

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: helion@helion.pl
WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<http://helion.pl/user/opinie/vbae19>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Kody źródłowe wybranych przykładów dostępne są pod adresem:

<ftp://ftp.helion.pl/przyklady/vbae19.zip>

ISBN: 978-83-283-5556-9

Copyright © Helion 2019

Printed in Poland.

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

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

Spis treści

Wstęp	7
Co to jest VBA?	7
Basic	7
Visual Basic	8
Visual Basic for Applications	8
Czy VBA jest przeżytkiem?	9
VBA i Office 2007, 2010, 2013, 2016, 2019	10
Zmieniony wygląd okna programu	10
Makropolecenia w Microsoft Office 2019	11
Bezpieczeństwo w Microsoft Office 2019	12
Jaką funkcję może pełnić VBA?	12
Kiedy warto stosować VBA?	13
Kiedy nie warto stosować VBA?	14
Jak napisać najprostszy program w VBA?	14
Podsumowanie	15
Rozdział 1. Makropolecenia	17
Pięć wersji Microsoft Office	18
Planowanie makropolecenia	18
Microsoft Office 2019	19
Planowanie i rejestrowanie makropolecenia	19
Podsumowanie	47
Rozdział 2. Jak uruchomić edytor Visual Basic for Applications	49
Gdzie w Excelu jest edytor VBA?	50
Czy makropolecenie może spowodować szkody?	56
Podsumowanie	67
Rozdział 3. Okno edytora Visual Basic for Applications	69
Okno Project	69
Okno Properties	71
Okno Code	73
Pasek menu	74
Pasek narzędziowy	74

Pierwszy program	75
Przełączanie między widokami	89
Wyrównywanie obiektów	90
Strzelanie z armaty do komara	94
Kiedy korzystać z makropoleceń, a kiedy z programów napisanych w VBA?	95
Podsumowanie	95
Rozdział 4. Zmienne	97
Nazwy zmiennych w VBA	98
Pułapki systemu komunikatów	103
Typy danych	104
Pułapki braku deklaracji	113
Wymuszanie deklarowania zmiennych	118
Zasięg deklaracji	123
Zmienne lokalne	123
Zmienne obowiązujące wewnątrz całego modułu	123
Zmienne globalne	124
Przekazanie wartości przez zmienną globalną	128
Przekazanie wartości przez wywołanie procedury z parametrem	130
Deklaracja typu i instrukcja przypisania	132
Komórka arkusza jako zmienna	137
Tekst jako wartość zmiennej	143
Podsumowanie	146
Rozdział 5. Komunikacja z użytkownikiem	147
Wprowadzanie danych	148
Wyświetlanie komunikatów	159
Poprawność wprowadzanych danych	172
Wycinanie tekstu	179
Podsumowanie	184
Rozdział 6. Korzystanie z obiektów	185
Obiekty	185
Właściwości	185
Metody	186
Zaznaczanie komórki	186
Elektroniczny sufler	188
Usuwanie zawartości i formatowania	190
Usuwanie zawartości	192
Usuwanie formatowania	193
Usuwanie wartości mniejszych od progowej	194
Właściwości	195
Przypisanie wartości komórce	195
Kopiowanie zawartości komórek	200
Nadawanie komórce koloru z użyciem nazwy koloru	202
Nadawanie komórce koloru z użyciem kodu koloru	203
Nadawanie koloru zawartości komórki	206
Przesuwanie aktywnej komórki	210
Podsumowanie	212

Rozdział 7. Instrukcje warunkowe	213
Porównywanie	214
Sterowanie wykonywaniem procedur	215
Skok do etykiety	215
Podjmowanie decyzji	225
Wybór jednej z trzech lub więcej opcji	227
Wykonanie grupy instrukcji określoną liczbę razy	232
Pętle zagnieżdżone	238
Wykonywanie pętli, gdy warunek jest spełniony	242
Podsumowanie	244
Rozdział 8. Elementy sterujące arkusza	245
Pole listy	245
Pole kombi (listy rozwijanej)	253
Pasek Toolbox i elementy sterujące arkusza	256
Właściwości	258
Podsumowanie	262
Rozdział 9. Zdarzenia	263
Lista zdarzeń dla skoroszytu	265
Lista zdarzeń dla arkusza	271
Lista zdarzeń dla aplikacji	274
Komunikacja z programem	277
Lista zdarzeń dla formularzy	281
Podsumowanie	284
Rozdział 10. Metody i właściwości dla zakresu	285
Kopiowanie zakresu komórek	285
Sortowanie zakresu komórek	292
Filtrowanie zakresu komórek	298
Wyszukiwanie informacji	302
Podsumowanie	305
Rozdział 11. Podprogramy	307
Śledzenie pracy programu	314
Procedury zagnieżdżone	317
Procedury zapętlone	319
Podsumowanie	321
Rozdział 12. Ściągawka z VBA	323
Metody	323
Funkcje	330
Instrukcje	339
Operatory	342
Operatory arytmetyczne	342
Operatory porównania	345
Operator konkatencji	347
Operatory logiczne	349
Podsumowanie	350

Rozdział 13. Funkcje arkuszowe	351
Odpowiedniki funkcji arkuszowych w VBA	354
Podsumowanie	375
Dodatek A Wybrane kody błędów VBA	377
Dodatek B Programowanie obiektowe	381
Programowanie proceduralne a obiektowe	381
Właściwości	382
Metody	383
Zdarzenia	383
Kolekcje	383
Modele obiektowe	384
Metoda kropkowa	384
Obiekty aktywne	384
Zakończenie	387
Skorowidz	389

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 przyklad95()
Range("B2").Select
End Sub
```

4. Uruchom program.
5. W oknie *Macros* zaznacz `przyklad95`.
6. Wyświetl arkusz Excela (rysunek 6.2).

Rysunek 6.2.
Zaznaczenie zostało przeniesione do komórki *B2*

	A	B
1		
2		

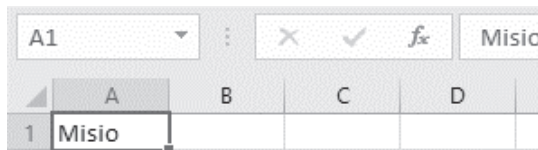
Przykład 96.

Napisz program, którego zastosowanie pozwoli wyczyścić komórkę A1.

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

Rysunek 6.3.

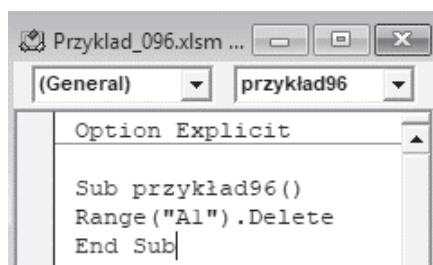
Dane do testowania



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 umożliwia usunięcie zawartości komórki.

Rysunek 6.4.

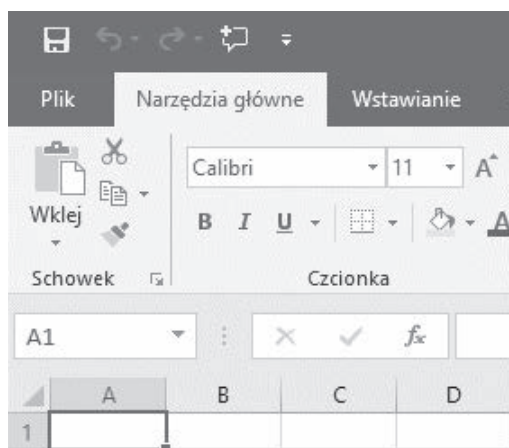
Program, którego zastosowanie pozwala wyczyścić komórkę A2



4. Uruchom program.
5. W oknie *Macros* zaznacz przykład96.
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.

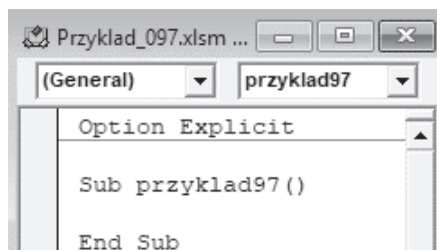
Przykład 97.

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.

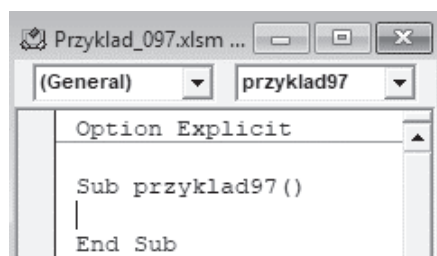
Szkielet programu



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

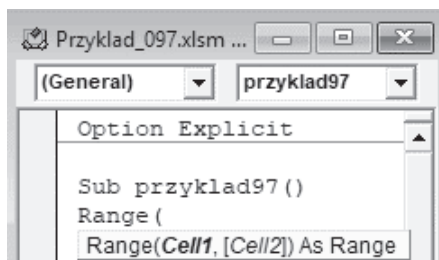
Rysunek 6.7.

Znak wstawiania został umieszczony między liniami ograniczającymi procedurę

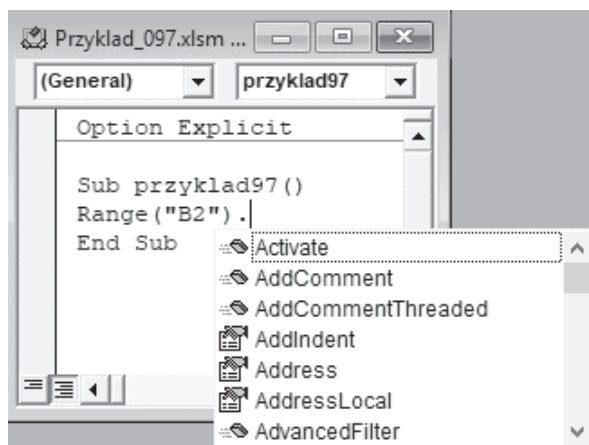


4. Podaj nazwę obiektu. W naszym przypadku będzie to zakres komórek. Wpisz `Range(`. W oknie edytora pojawiła się podpowiedź (rysunek 6.8).
5. Zakresem będzie jedna komórka. Wpisz `"B2")`.
6. Po wpisaniu kropki w oknie edytora pojawiła się podpowiedź (rysunek 6.9).
7. Metody i właściwości są uporządkowane 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`.

Rysunek 6.8.
Edytor podpowiada,
jak można zdefiniować
zakres

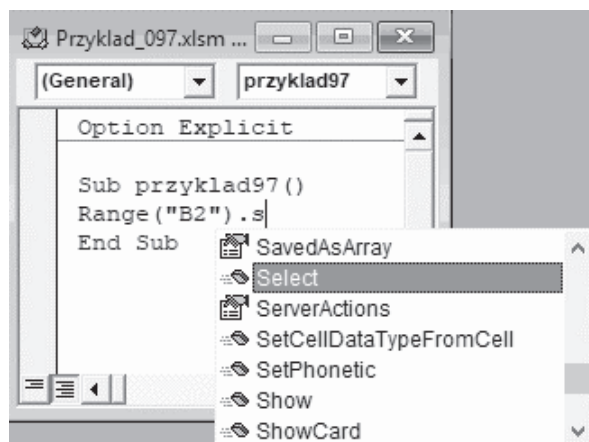


Rysunek 6.9.
Lista metod
i właściwości



8. Została wyświetlona lista metod i właściwości o nazwach rozpoczynających się od litery S. Zaznacz metodę Select (rysunek 6.10).

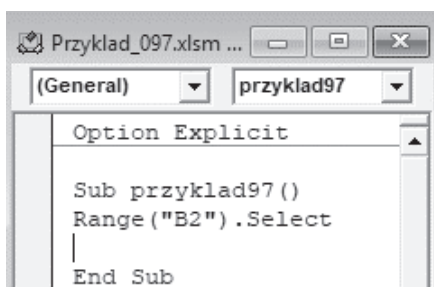
Rysunek 6.10.
Lista metod
i właściwości
o nazwach
rozpoczynających
się od litery S



9. Potwierdź wybór naciśnięciem klawisza *Enter*.
10. Do programu została wstawiona metoda Select. Program jest gotowy (rysunek 6.11).

Rysunek 6.11.

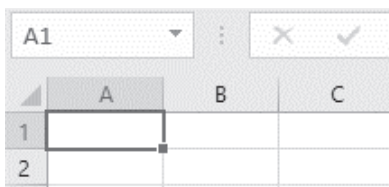
Program
po dodaniu metody



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

Rysunek 6.12.

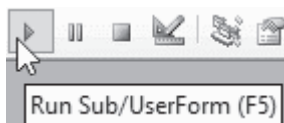
Arkusz z wybraną
komórką A1



12. Wyświetl okno edytora.
13. Uruchom program kliknięciem ikony (rysunek 6.13).

Rysunek 6.13.

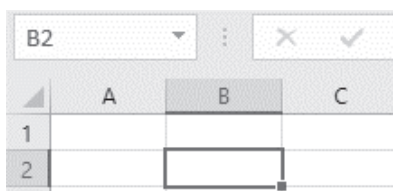
Kliknięcie ikony
spowoduje
uruchomienie
programu



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

```

Option Explicit

Sub przykład98 ()
Range ("A1:C3") .Clear

End Sub

```

3. Wyświetl okno arkusza Excel. Domyślnie wszystkie komórki są puste. Wpisz dane do komórek (rysunek 6.16).

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

4. Wyświetl okno edytora.
5. Uruchom program kliknięciem ikony (rysunek 6.17).

Rysunek 6.17.

Kliknięcie ikony spowoduje uruchomienie programu

```

Option Explicit

Sub przykład98 ()
Range ("A1:C3") .Clear

End Sub

```

6. Program został wykonany.
7. 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.

1. Wyświetl okno *Code*.
2. 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.

Rysunek 6.19.

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

```

Przykład_099.xlsm - Arkusz1 (Code)
(General)
Option Explicit

Sub przyklad99()
Range("A1:C3").ClearContents
End Sub

```

3. Wyświetl okno arkusza Excel. Domyślnie wszystkie komórki są puste. Wpisz dane do komórek (rysunek 6.20).

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

4. Wyświetl okno edytora.
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 ograniczonym obszarze

 A screenshot of a VBA code editor window titled 'Przykład_100.xlsm - Arkusz1 (Code)'. The 'General' tab is selected. The code in the editor is:


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

Rysunek 6.26.

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

```

Option Explicit
Dim komórka As Variant

Sub przyklad101()
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

```

Rysunek 6.27.

Dane wpisane do Arkusza1

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

- 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.

Rysunek 6.28.

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

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

Właściwości

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

Przypisanie wartości komórce

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

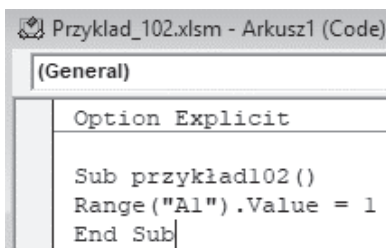
Przykład 102.

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



```

Option Explicit

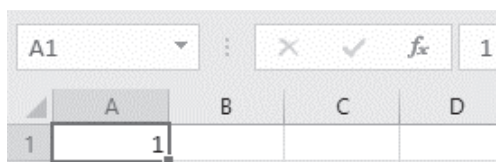
Sub przykład102()
    Range("A1").Value = 1
End Sub

```

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



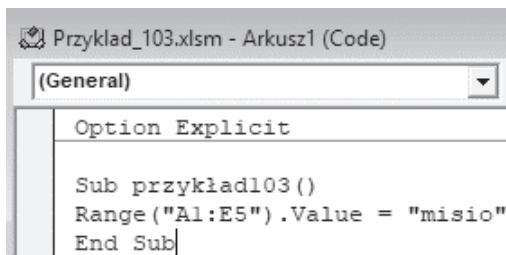
Przykład 103.

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.

Rysunek 6.31.

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



```

Option Explicit

Sub przykład103()
    Range("A1:E5").Value = "misio"
End Sub

```

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

Rysunek 6.32.

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

	A	B	C	D	E
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



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.

- Wyświetl okno *Code*.
- 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 *Variant*. 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 przyklad104()
    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.VIII.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.VIII.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.

```
(General) przyklad106
Option Explicit
Dim komórka As Variant
Sub przyklad106()
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
```

Rysunek 6.36. Program, za pomocą którego można zastąpić wartość liczbową z przedziału od 0 do 9 jej słownym odpowiednikiem

3. Wyświetl okno arkusza Excel. W *Arkusz1* wpisz dane (rysunek 6.37).

Rysunek 6.37.

Dane wpisane
do *Arkusz1*

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

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.

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

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*.

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 przyklad107()
    Range("B1:B3").Value = Range("A1:A3").Value
End Sub

```

3. Wyświetl arkusz Excela.
4. W komórki od *A1* do *A3* wpisz odpowiednio: 1, 2, 3 (rysunek 6.40).

Rysunek 6.40.
Arkusz z zestawem
danych

	A	B
1	1	
2	2	
3	3	

5. Wyświetl okno edytora VBA.
6. Naciśnij klawisz *F5*.
7. Program został wykonany. Wyświetl arkusz Excela.
8. 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
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.

1. Wyświetl okno *Code*.
2. 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 przyklad108()
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. skopiować 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).

Rysunek 6.43.

Arkusz 1. z zestawem danych

	A	B	C	D	E
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).

Rysunek 6.44.

Arkusz 2. z zestawem danych skopiowanych z arkusza 1.

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

< > Arkusz1 **Arkusz2** Arkusz3

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.

Rysunek 6.45.

Komórki od *A1* do *A3* będą miały kolor żółty

```
Przyklad_109.xlsm - Arkusz1 (Code)
(General)
Option Explicit
Sub przyklad109 ()
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).

Rysunek 6.46.

Komórki, których
kolor został zmieniony

	A	B
1		
2		
3		



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
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 przykład110()
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).

Rysunek 6.47.

Numer wiersza
odpowiada wartości
parametru `ColorIndex`

	A	B
1	Black	
2		
3	Dark Gray	
4	Medium-Dark Gray	
5	Black	
6	Light Gray	
7	Dark Gray	
8	Medium-Dark Gray	
9	Black	
10	Dark Gray	
11	Black	
12	Dark Gray	
13	Dark Gray	
14	Dark Gray	
15	Medium-Dark Gray	
16	Dark Gray	
17	Medium-Dark Gray	
18	Dark Gray	
19		
20	Light Gray	
21	Black	
22	Medium-Dark Gray	
23	Dark Gray	

Przykład 111.

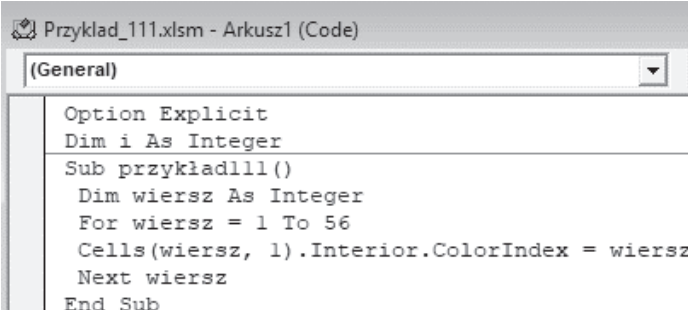
Napisz program, którego zastosowanie pozwoli nadawać komórkom z obszaru od A1 do A56 kolory z palety barw dostępnej za pośrednictwem właściwości `ColorIndex`. Wykorzystaj instrukcję pętli.

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. Wartość zmiennej `wiersz` jest

Rysunek 6.48.

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



```
Przyklad_111.xlsm - Arkusz1 (Code)
(General)
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
```

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 określany 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.

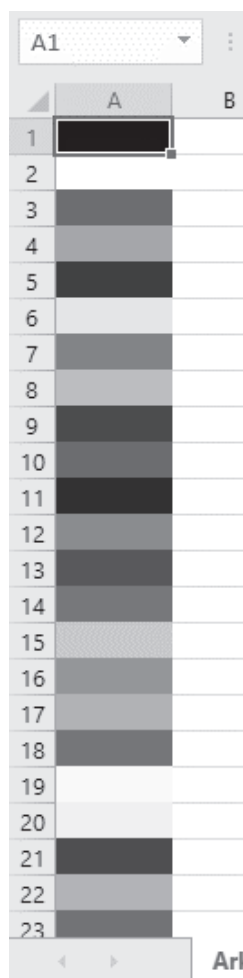
4. Uruchom program.
5. Wyświetl okno arkusza Excel.
6. W komórkach od `A1` do `A56` zostały wyświetlone próbki kolorów (rysunek 6.49).

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`.

Rysunek 6.49.

Rezultat identyczny jak na rysunku 6.47 osiągnięto znacznie mniejszym nakładem pracy

**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).
2. Wyświetl okno `Code`.

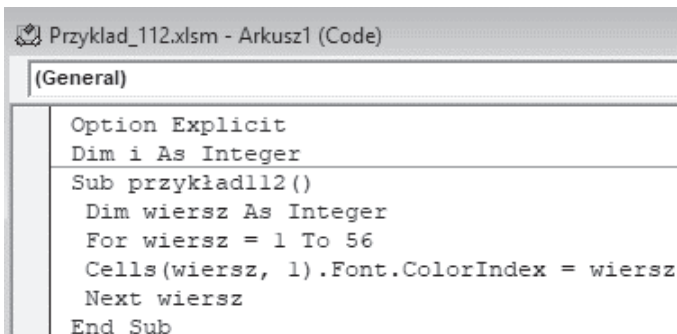
Rysunek 6.50.
*Domyślnie tekst
 ma kolor czarny*

	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

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.

Rysunek 6.51.

Program, którego zastosowanie pozwoli zamienić napisy czarne w wielobarwne



```
Przyklad_112.xlsm - Arkusz1 (Code)
(General)
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).

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

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`.

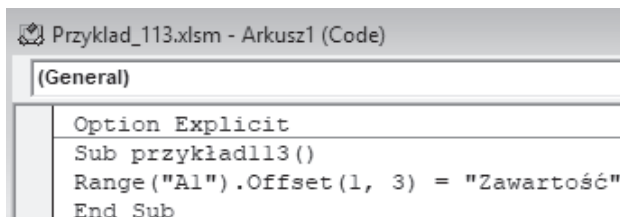
Przykład 113.

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.

Rysunek 6.53.

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



```
Przyklad_113.xlsm - Arkusz1 (Code)
(General)
Option Explicit
Sub przyklad113()
Range("A1").Offset(1, 3) = "Zawartość"
End Sub
```

3. Uruchom program.
4. Wyświetl okno arkusza Excel.
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
3				

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 przyklad115()
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).

Rysunek 6.56.

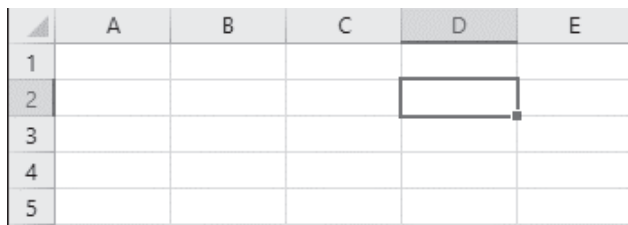
Komórka zaznaczona domyślnie

	A	B	C	D	E
1					
2					
3					
4					
5					

4. Uruchom program. Naciśnij klawisz *F5*.
5. Wyświetl okno arkusza Excel (rysunek 6.57).

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, 331
- Activate, 323
- AddComment, 323
- ADO, 324
- adres
 - komórki
 - wpisywanie, 154
 - obiektu zmiennej, 342
 - właściwości, 342
- adresowanie
 - bezwzględne, 45, 47
 - względne, 46
- AdvancedFilter, 323
- amortyzacja środka trwałego, 334
- And, 349
- Anuluj, 164, 165
- API, Application Programming Interface, 10
- AppActivate, 339
- AppleScript, 333
- ApplyNames, 323
- ApplyOutlineStyles, 323
- Arccos, 331
- Arccosec, 331
- Arccotan, 331
- Arcsec, 331
- Arcsin, 331
- arkusz
 - elementy sterujące, 245, 256
- Asc, 331
- Atn, 331
- atrybuty
 - pliku, 332, 342
 - katalogu, 332
- AutoFill, 323
- AutoFilter, 323

- AutoFit, 323
- AutoOutline, 323

B

- BASIC, 7, 8
- Beep, 339
- bezpieczeństwo, 12
- bieżący
 - katalog, 339
 - napęd, 339
- błąd, 377
 - obsługa, 341
 - odpowiadający kodowi, 332
 - opis, 177
 - zdefiniowany przez użytkownika, 331
 - źródło, 326
- Boolean, 339
- BorderAround, 324, 326
- buttons, 160
- Byte, 339

C

- Calculate, 324
- Call, 339
- CallByName, 331
- carriage return, 148
- CBool, 331
- CByte, 331
- CCur, 331
- CDate, 331
- CDbl, 331
- CDec, 331
- ChDir, 339
- ChDrive, 339
- CheckSpelling, 324

- Choose, 331
 - Chr, 331
 - ciąg bez końcowych spacji, 334
 - ciąg bez początkowych spacji, 333
 - CInt, 331
 - Clear, 324
 - ClearComments, 324
 - ClearContents, 324
 - ClearFormats, 324
 - ClearNotes, 324
 - ClearOutline, 324
 - CLng, 331
 - Close, 339
 - ColorIndex, 329, 330
 - ColumnDifferences, 324
 - COM, 10
 - Command, 331
 - Component Object Model, 10
 - Consolidate, 324
 - Const, 339
 - context, 148, 160
 - Copy, 324
 - CopyFromRecordset, 324
 - CopyPicture, 324
 - Cos, 331
 - Cosec, 331
 - cosecans hiperboliczny, 332
 - cosinus hiperboliczny, 332
 - Cotan, 331
 - cotangens hiperboliczny, 332
 - CreateNames, 324
 - CreateObject, 331
 - CSng, 331
 - CStr, 331
 - CurDir, 331
 - Currency, 339
 - Cut, 324
 - CVar, 331
 - CVErr, 331
 - czas systemowy, 169, 335
 - część
 - całkowita liczby, 333
 - daty, 331
- D**
- dane
 - eksportowanie, 324
 - grupowanie, 325
 - kopiowanie, 323
 - scalenie, 324
 - sortowanie, 326
 - wpisywanie, 155
 - wprowadzanie, 148
 - DAO, 324
 - data
 - i czas systemowy, 338
 - i godzina, 332
 - systemowa, 169, 331, 334, 339
 - DataSeries, 324
 - Date, 175, 331, 333, 339
 - Date\$, 169
 - DateAdd, 331
 - DateDiff, 331
 - DatePart, 331
 - DateSerial, 331
 - DateValue, 331
 - Day, 332
 - DDB, 332
 - Decimal, 339
 - Declare, 339
 - default, 148
 - DefBool, 339
 - DefByt, 339
 - DefCur, 339
 - DefDate, 339
 - DefDbl, 339
 - DefDec, 339
 - DefInt, 340
 - DefLng, 340
 - DefObj, 340
 - DefSng, 340
 - DefStr, 340
 - DefVar, 340
 - deklaracja, 342
 - funkcji, 341
 - nazw i argumentów procedury, 341
 - typu, 132
 - zdarzenia, 340
 - zmiennych, 118, 340
 - Delete, 324
 - DeleteSetting, 340
 - Deweloper, 32
 - DialogBox, 324
 - Dim, 340
 - Dir, 332
 - Dirty, 324
 - DLL, 339
 - długość
 - pliku w bajtach, 332
 - tekstu, 181
 - Do Until, 157
 - Do...Loop, 340
 - dodawanie, 343
 - DoEvents, 332
 - dolne ograniczenia indeksu tablicy, 341
 - domyślny
 - tryb porównywania ciągu, 341
 - typ danych, 339

Double, 339
drukowanie
 obiekt, 325
dzielenie, 343
 scalonego obszaru, 326
dzień
 miesiąca, 332
 tygodnia, 335
dźwięk, 339

E

EditionOptions, 324
edytor VBA, 9, 49
eksportowanie danych, 324
elektroniczny sufler, 188
elementy sterujące arkusza, 245, 256
End, 340
End Sub, 44, 47
Enum, 340
Environ, 332
EOF, 332
Eqv, 349
Erase, 340
Error, 332, 333, 340
Event, 340
Example, 327
Excel 5, 9
Exit Do, 340
Exit For, 340
Exit Function, 340
Exit Property, 340
Exit Sub, 340
Exp, 332
ExportAsFixedFormat, 324

F

FileAttr, 332
FileCopy, 340
FileDateTime, 332
FileLen, 332
FillDown, 324
FillLeft, 325
FillRight, 325
FillUp, 325
Filter, 332
filtrowanie
 listy, 323
 zakresu komórek, 298
Find, 325
FindNext, 325
FindPrevious, 325

folder
 atrybuty, 332
For Each...Next, 340
For...Next, 341
Format, 332
FormatCurrency, 332
FormatDateTime, 332
FormatNumber, 332
FormatPercent, 332
formaty
 wyczyszczenie, 324
formuła
 wyczyszczenie, 324
FreeFile, 332
Function, 341
FunctionWizard, 325
funkcja, 330
 InputBox, 148, 149
 Or, 350
 wykładnicza, 332
funkcje arkusza, 330, 351
 odpowiedniki VBA, 354
FV, 332

G

generator liczb losowych, 342
Get, 341
GetAllSettings, 332
GetAttr, 332
GetObject, 332
GetSetting, 332
godzina, 332
godzina systemowa, 334
 ustawienie, 342
GoSub...Return, 341
GoTo, 177, 341
Group, 325
grupowanie danych, 325

H

HArcCos, 332
HArcCosec, 332
HArcCota, 332
HArcsec, 332
HArcsin, 332
HArcTan, 332
HCos, 332
HCosec, 332
HCotan, 332
helpfile, 148, 160
Hex, 332

Hour, 332
 HSec, 333
 HSin, 333
 HTan, 333

I

ID zadania, 334
 If...Then...Else, 341
 Ignoruj, 164
 If, 333
 ikony paska Toolbox, 257
 iloczyn logiczny, 349
 Imp, 349
 Implements, 341
 implikacja, 349
 indeks tablicy, 335
 Input, 333
 Input #, 341
 InputBox, 148–151, 171, 173, 175, 333
 Insert, 325
 InsertIndent, 325
 InStr, 333
 InStrRev, 333
 instrukcja

- GoTo etykieta, 215, 244
- przypisania, 132

 instrukcje, 339–342

- warunkowe, 213
- porównywanie, 214

 Int, Fix, 333
 Integer, 172, 177, 340
 interfejs, 341
 IPmt, 333
 IRR, 333
 IsArray, 333
 IsDate, 333
 IsEmpty, 333
 IsError, 333
 IsMissing, 333
 IsNull, 333
 IsNumeric, 333
 IsObject, 333

J

język

- BASIC, 8

 Join, 333
 Justify, 325

K

karta

- Deweloper, 53
- Dodatki, 10

 katalog

- atrybuty, 332
- bieżący, 339
- utworzenie, 341
- zmiana nazwy, 341

 Kill, 341
 klasa, 341
 klucz w rejestrze Windows, 340
 kod

- ASCII, 331
- błędu, 377
- RGB koloru, 334
- znaku, 331

 kolejny dostępny numer pliku, 332
 kolekcje, 383
 kolor ramki, 329, 330
 komentarz, 323, 342

- wyczyszczenie, 324

 komórki

- kopiowanie zawartości, 200
- nadawanie koloru, 202, 203, 206
- nazwa, 323
- odczytanie, 326
- przesuwanie aktywnej, 210
- przypisanie wartości, 195
- scalenie, 325
- skopiowanie, 324
- uaktywnienie, 323
- usuwanie formatowania, 190, 193
- usuwanie zawartości, 190, 192, 194
- wstawienie, 325
- wypełnianie, 325
- zaznaczanie, 186

 komunikacja

- z programem, 277
- z użytkownikiem, 147

 komunikat, 159, 160

- Invalid character, 103
- o błędzie, 124, 174
- wyświetlanie, 159

 koniec pliku tekstowego, 332
 konstrukcja

- If ... Then, 225, 244
- Select Case, 228, 244

 kontrolka CommandButton, 261
 kontynuowanie wyszukiwania, 325

kopiowanie

- ADO, 324
- danych, 323
- komórek, 19, 324
- obrazu, 324
- pliku, 340
- zakresu komórek, 285

L

- LBound, 333
- LCase, 333
- Left, 333
- Len, 181, 333
- Let, 341
- liczba, 332
 - bajtów, 333
 - decymalna, 335
 - heksadecymalna, 335
 - konwersja, 332
 - losowa, 334
 - okresów spłaty, 334
 - reprezentująca czas, 335
 - sekund od północy, 335
 - spacji, 334
 - zaokrąglona, 334
 - znaków, 181, 333
- Line Input #, 341
- linefeed character, 148
- lista
 - filtrowanie, 323
 - właściwości, 259
 - zdarzeń, 265
- ListNames, 325
- listy
 - nieukrytych nazw, 325
 - rozwijane, 253
- Load, 341
- Loc, 333
- Lock, 341
- LOF, 333
- Log, 333
- logarytm naturalny, 333
- lokalna
 - tablica, 341
 - zmienna, 341
- Long, 340
- Loop, 157
- LSet, 341
- LTrim, 182, 333

Ł

- łańcuch znaków, 334
 - odwrócony, 335
 - po konwersji, 335
- łączenie ciągów znaków, 347

M

- MacID, 333
- MacScript, 333
- makropolecenia, 10, 11, 21
 - blokowanie, 12
 - edycja, 43
 - przycisk, 35
 - rejestrwanie, 21
 - testowanie, 24
 - uruchomienie, 326
 - warunki początkowe, 18
 - zapisywanie, 38
- makropolecenie Kopiowanie_komórek, 55
- małe litery, 333
- Merge, 325
- metoda, 186, 323, 327, 383
 - AutoFilter, 298
 - Find, 302
 - kropkowa, 384
 - Sort, 292
- metody dla zakresu, 285
- Microsoft Office 2000, 9
- Mid, 180, 333, 341
- miesiąc, 334
- Minute, 333
- MIRR, 334
- MkDir, 341
- Mod, 343
- modalność, 163
- modele obiektowe, 384
- moduł prywatny, 341
- Month, 334
- MonthName, 334
- msdn.microsoft.com, 326
- MsgBox, 159, 164, 171, 173, 334

N

- najmniejszy indeks tablicy, 333
- największy indeks tablicy, 335
- Name, 341
- napeł
 - bieżący, 339
- napis
 - nad przyciskiem, 168

- nazwa
 - komórka, 323
 - miesiąca, 334
 - pliku lub katalogu, 332
 - zmiennych, 98
 - negacja, 349
 - Nie, 164
 - nierównoważność logiczna, 349
 - nieukryte nazwy, 325
 - Not, 349
 - notatki
 - wyczyszczenie, 324
 - Now, 334
 - NPer, 334
 - NPV, 334
 - Null, 333
- O**
- obiekt, 185
 - drukowanie, 325
 - OLE, 332
 - Range
 - wstawianie, 325
 - usunięcie, 324
 - wycięcie, 324
 - wyczyszczenie, 324
 - zakres, 324, 326
 - obiekty aktywne, 384
 - Object, 340
 - obraz
 - skopiowanie, 324
 - obsługa
 - błędów, 341
 - zdarzeń, 332
 - obszar
 - dzielenie, 326
 - obwiednia, 326
 - obwódka, 327
 - Oct, 334
 - odczyt
 - danych z otwartego pliku, 341
 - danych z sekwencyjnego pliku, 341
 - pojedynczej linii, 341
 - zawartości komórki, 326
 - odpowiedniki funkcji arkuszowych, 354
 - odwołanie, 46
 - odwrócony łańcuch znaków, 335
 - OK, 164, 165
 - okno
 - tytuł, 149
 - okno aplikacji
 - uaktywnienie, 339
 - okno
 - Code, 73
 - dialogowe, 148, 170, 333
 - edytora, 57
 - edytora VBA, 69
 - komunikatu, 159, 160, 334
 - Makro, 55
 - Project, 69
 - Properties, 71
 - VBA, 9
 - OLE, 331–333
 - On Error, 341
 - On Error GoTo, 172
 - On Error Resume Next, 172
 - On...GoSub, 341
 - On...GoTo, 341
 - Open, 341
 - operator konkatencji, 347
 - operatory
 - arytmetyczne, 342
 - logiczne, 349
 - porównania, 345
 - opis błędu, 177
 - Option Base, 341
 - Option Compare, 341
 - Option Explicit, 341
 - Option Private, 341
 - Or, 349
 - otworenie pliku tekstowego, 341
- P**
- Parameters, 327
 - Parse, 325
 - Partition, 334
 - pasek
 - menu, 74
 - narzędziowy, 74
 - Toolbox, 256, 257
 - PasteSpecial, 325
 - pętla, 340, 342
 - Do While, 244
 - For ... Next, 232, 242, 244
 - programowa, 340
 - zagnieżdżona, 238
 - Phonetic, 326
 - pierwsza liczba w łańcuchu znaków, 335
 - pierwszy program, 75
 - pisownia
 - sprawdzanie, 324
 - pliki
 - .docm, 11
 - .docx, 11
 - .dotm, 11

- .dotx, 11
 - .xlsm, 11
 - .xlsx, 11
 - .xltm, 11
 - .xltx, 11
 - atrybuty, 332, 342
 - pozycja, 334
 - tekstowy, 341
 - zmiana nazwy, 341
 - Pmt, 334
 - podjęmowanie decyzji, 225
 - podgląd wydruku, 325
 - podprogramy, 307
 - podtyp zmiennej, 335
 - podzbiór tablicy łańcuchów znaków
 - z uwzględnieniem filtru, 332
 - pole
 - edycji, 170
 - kombi, 253
 - listy, 245
 - polecenie Notepad, 62
 - połączenie łańcuchów, 333
 - pomoc
 - przycisk, 162
 - wyświetlanie, 324, 326
 - porównanie łańcuchów znaków, 335
 - potęgowanie, 343
 - pozycja
 - w pliku, 334
 - wydruku, 335
 - wystąpienia ciągu znaków, 333
 - PPmt, 334
 - Print #, 341
 - PrintOut, 325
 - PrintPreview, 325
 - Private, 341
 - procedura, 339
 - obsługi błędów, 172, 341
 - zagnieżdżona, 317
 - zapętłona, 319
 - zewnątrzna, 339
 - procent, 332
 - program
 - rozgałęzienie, 341
 - zatrzymanie, 342
 - programowanie
 - obiektywne, 381
 - proceduralne, 381
 - prompt, 148, 159
 - Property
 - Get, 341
 - Let, 341
 - Set, 341
 - prośba o wpisanie ciągu znaków, 156
 - przedział czasu, 331
 - przekazanie wartości
 - przez wywołanie procedury, 130
 - przez zmienną globalną, 128
 - przekształcanie wyrażenia, 331
 - przeliczanie, 324
 - przełączanie między widokami, 89
 - Przerwij, 164
 - przycisk, 164, 167
 - OK, 164
 - Pomoc, 162
 - w oknie komunikatu, 160
 - przykład zastosowania metody, 327
 - przypisanie
 - makra, 33
 - wartości wyrażenia zmiennej, 341
 - Public, 341
 - pułapki
 - braku deklaracji, 113
 - systemu komunikatów, 103
 - Put, 341
 - PV, 334
- Q**
- QBColor, 334
- R**
- RaiseEvent, 341
 - Randomize, 342
 - Range, 326
 - Rate, 334
 - ReDim, 342
 - rejestrowanie makra, 21
 - Rem, 342
 - Remarks, 327
 - RemoveDuplicates, 325
 - Replace, 334
 - Reset, 342
 - Resume, 342
 - reszta z dzielenia, 343
 - Return value, 327
 - RGB, 334
 - Right, 179, 334
 - Rmdir, 342
 - Rnd, 334
 - rok, 335
 - Round, 334
 - RowDifferences, 326
 - rozdzielenie zakresu danych, 325
 - rozgałęzienie programu, 341
 - rozsunięcie tekstu, 325

rozszerzenie
 .docm, 11
 .docx, 11
 .dotm, 11
 .dotx, 11
 .xlsm, 11
 .xlsx, 11
 .xltm, 11
 .xltx, 11

RSet, 342
 RTrim, 183, 334
 Run, 326

S

SaveSetting, 342

scalenie

danych, 324
 komórek, 325

Sec, 334

secans hiperboliczny, 333

Second, 334

Seek, 334, 342

sekundy, 334

Select, 326

Select Case, 342

SendKeys, 342

seria danych, 324

Set, 342

SetAttr, 342

SetPhonetic, 326

Sgn, 334

Shell, 334

Show, 326

ShowDependents, 326

ShowErrors, 326

Sin, 334

Single, 340

składnia metody, 327

skok

bezwarunkowy, 341
 do etykiety, 215
 do podprogramu, 341

skoroszyt

lista zdarzeń, 265
 makr osobistych, 22
 programu Excel, 41
 programu Excel z obsługą makr, 39

skrót klawiaturowy, 22

Alt+F11, 67
 Ctrl+Alt+Del, 321

skrypt AppleScript, 333

SLN, 334

Sort, 326

sortowanie

danych, 326
 zakresu komórek, 292

Space, 334

Spc, 334

Speak, 326

SpecialCells, 326

Split, 334

sprawdzanie pisowni, 324

Sqr, 335

stała, 339

vbAbortRetryIgnore, 161

vbApplicationModal, 163

vbCritical, 162

vbDefaultButton1, 163

vbDefaultButton2, 163

vbDefaultButton3, 163

VbDefaultButton4, 163

vbExclamation, 162

vbInformation, 162

vbMsgBoxRight, 164

vbMsgBoxRtlReading, 164

vbOKCancel, 160

vbOKOnly, 160

vbQuestion, 162

vbRetryCancel, 161

vbSystemModal, 163

vbYesNo, 161

vbYesNoCancel, 161

Static, 342

sterowanie

dostępem, 341
 wykonywaniem procedur, 215

Stop, 342

Str, 335

StrComp, 335

StrConv, 335

String, 172, 335, 340

StrReverse, 335

Sub, 44, 47, 342

SubscribeTo, 326

suma logiczna, 349

Switch, 335

SYD, 335

symulacja

błędu, 340
 pisania z klawiatury, 342

Syntax, 327

szerokość

kolumny, 323
 pliku, 342

Ś

ścieżka bieżącego katalogu, 331
 śledzenie pracy programu, 314

T

Tab, 335
 tabela danych, 326
 Table, 326
 tablica, 333
 lokalna, 341
 zainicjowanie elementów, 340
 zmiana wymiarów, 342
 Tak, 164
 Tan, 335
 tekst
 długość, 181
 rozsunięcie, 325
 wycinanie, 179
 TextToColumns, 326
 Time, 335, 342
 Time\$, 169
 Timer, 335
 TimeSerial, 335
 TimeValue, 335
 title, 148, 160
 Trim, 183
 Trim s, 335
 typ
 danych, 104, 335
 danych użytkownika, 342
 wyliczeniowy, 340
 zmiennej, 172
 Type, 342
 TypeName, 335
 tytuł okna, 149

U

uaktywnienie
 komórki, 323
 okna aplikacji, 339
 UBound, 335
 UCase, 335
 Ungroup, 326
 Unload, 342
 Unlocks, 341
 UnMerge, 326
 uruchomienie makropolecenia, 326
 usunięcie
 folderu, 342
 obiektu, 324
 obiektu z pamięci, 342

pliku, 341
 zdublowanych obiektów, 325
 znaków białych, 182
 utworzenie
 katalogu, 341
 nazw na podstawie etykiet tekstowych, 324
 obiektów Phonetic, 326
 serii danych, 324
 tabeli danych, 326

V

Val, 335
 Variant, 340
 VarType, 335
 VBA, Visual Basic for Applications, 7
 kiedy stosować, 13
 vbAbortRetryIgnore, 161
 vbApplicationModal, 163
 vbCritical, 162
 vbDefaultButton1, 163
 vbDefaultButton2, 163
 vbDefaultButton3, 163
 VbDefaultButton4, 163
 vbExclamation, 162
 vbInformation, 162
 vbMsgBoxHelpButton, 162
 vbMsgBoxRight, 164
 vbMsgBoxRtlReading, 164
 vbOKCancel, 160
 vbOKOnly, 160
 vbQuestion, 162
 vbRetryCancel, 161
 vbSystemModal, 163
 vbYesNo, 161
 vbYesNoCancel, 161
 Visual Basic, 8

W

wartość
 bezwzględna, 331
 ciągu wyrażeń logicznych, 335
 inwestycji, 334
 netto, 334
 kapitału dla raty pożyczki, 334
 właściwości, 331
 z listy argumentów, 331
 warunkowe wykonanie instrukcji, 341, 342
 wczytanie adresu komórki, 153
 Weekday, 335
 WeekdayName, 335
 wejście aplikacji w rejestrze Windows, 332

- wewnętrzna stopa zwrotu dla ciągu okresowych przepływów gotówkowych, 334
 - While...Wend, 342
 - Width #, 342
 - wielkie litery, 335
 - With, 342
 - właściwości, 185, 195, 258, 382
 - dla zakresu, 285
 - kontrolek, 262
 - właściwość
 - AutoSize, 261
 - Height, 260
 - wprowadzanie danych, 148
 - Write #, 342
 - wstawianie
 - komórki, 325
 - obiektu Range, 325
 - wycięcia, 325
 - wstążka, 10
 - wstrzymanie programu, 333
 - wycięcie
 - obiektów, 324
 - tekstu, 179
 - wstawianie, 325
 - wyczyszczenie
 - formatów, 324
 - formuły, 324
 - komentarzy, 324
 - notatek, 324
 - obiektu, 324
 - wydruk
 - podgląd, 325
 - wygenerowanie zdarzenia, 341
 - wyjście z bloku pętli
 - Do...Loop, 340
 - For Each...Next, 340
 - For...Next, 340
 - z funkcji, 340
 - z procedury Property, 340
 - z procedury Sub, 340
 - wymuszenie deklaracji zmiennych, 341
 - wypełnianie komórek, 323, 325
 - wyrównanie
 - do lewej, 341
 - do prawej, 342
 - obiektów, 90
 - wyróżnienie
 - zakres, 323
 - wysłanie kodów wciśniętych klawiszy, 342
 - wysokość wiersza
 - dopasowanie, 323
 - wyszukiwanie, 325
 - informacji, 302
 - kontynuowanie, 325
 - wyświetlanie
 - komunikatów, 159
 - okna komunikatu, 159
 - pomocy, 324, 326
 - zaznaczonego zakresu, 326
 - wywołanie metody, 327
 - wznowienie wykonywania programu, 342
- X**
- Xor, 349
 - xpos, 148
- Y**
- Year, 335
 - ypos, 148
- Z**
- zainicjowanie elementów tablicy, 340
 - zakończenie
 - bloku, 340
 - procedury, 340
 - programu, 340
 - zakres, 334
 - filtrowanie komórek, 298
 - kopiowanie komórek, 285
 - metody, 285
 - obiekt, 324, 326
 - sortowanie komórek, 292
 - właściwości, 285
 - wyróżnienie, 323
 - zależność, 326
 - załadowanie obiektu, 341
 - zamiana
 - ciągu znaków na datę, 331
 - daty na liczbę, 331
 - liczby na znaki, 335
 - zamknięcie plików, 342, 339
 - zapis
 - danych do pliku, 341, 342
 - w rejestrze Windows, 342
 - zasięg deklaracji, 123
 - zastąpienie znaków, 341
 - zatrzymanie programu, 342
 - zaznaczenie, 186, 323, 326
 - zaznaczony zakres
 - wyświetlenie, 326
 - zdarzenia, 263, 383
 - aplikacji, 274
 - arkusza, 271
 - formularzy, 281
 - skoroszytu, 266

- zdarzenie
 - deklaracja, 340
- zdublowane obiekty, 325
- zgodność wsteczna, 10
- zmiana
 - nazwy pliku lub katalogu, 341
 - wymiarów tablicy, 342
- zmiennie, 97
 - globalne, 124, 321
 - komórka arkusza, 137
 - lokalne, 123, 341
 - na poziomie procedury, 342
 - nazwy, 98
 - przekazanie wartości, 128
 - środowiskowe systemu operacyjnego, 332
 - tekst, 143
 - wymuszanie deklarowania, 118
- znak, 331
 - liczby, 334
- znaki
 - z łańcucha, 333
 - z pliku, 333
- źródło błędów, 326

Ż

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

Włącz supermoce Excela — skorzystaj z VBA!



- Poznaj możliwości makropoleceń
- Odkryj konstrukcje języka VBA
- Naucz się wykorzystywać je w Excelu

Excela używa się dosłownie wszędzie — pełni funkcję uniwersalnego programu do przeprowadzania obliczeń naukowych, statystycznych i finansowych, analizy najrozmaitszych danych, wizualizacji wyników i tworzenia rozbudowanych raportów. Nie bez znaczenia są też oferowane przez niego możliwości automatyzacji działań oraz prowadzenia interakcji z użytkownikami arkusza i innymi aplikacjami pakietu Microsoft Office — co zapewnia zastosowanie makropoleceń opracowanych za pomocą języka Visual Basic for Applications.

Jeśli chcesz wejść na wyższy poziom i dowiedzieć się, jak upraszczać i przyspieszać swoją pracę z Excelem, sięgnij po tę książkę! W niezwykle prosty sposób prezentuje ona możliwości makr, metody ich tworzenia, konstrukcje języka VBA i najrozmaitsze sposoby ich zastosowania. Wiedza nie jest tu oderwana od rzeczywistości — zilustrowano ją praktycznymi przykładami, dzięki czemu szybko nauczysz się nie tylko programować, ale też rozwiązywać prawdziwe problemy, z którymi na co dzień mierzy się wielu użytkowników arkusza kalkulacyjnego.

- 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

Z VBA w Excelu trafisz prosto do celu!

	<p>Sprawdź nasze szkolenia!</p>  <p>AKADEMIA IT & BUSINESS</p>	<p>KOD KORZYŚCI Sięgnij po więcej! ►</p> 
 helion.pl	<p>WWW.SZKOLENIA.HELION.PL</p>	<p>ISBN 978-83-283-5556-9</p>  <p>9 788328 355569</p>
<p>INFORMATYKA W NAJLEPSZYM WYDANIU</p>		<p>Cena: 59,00 zł</p>