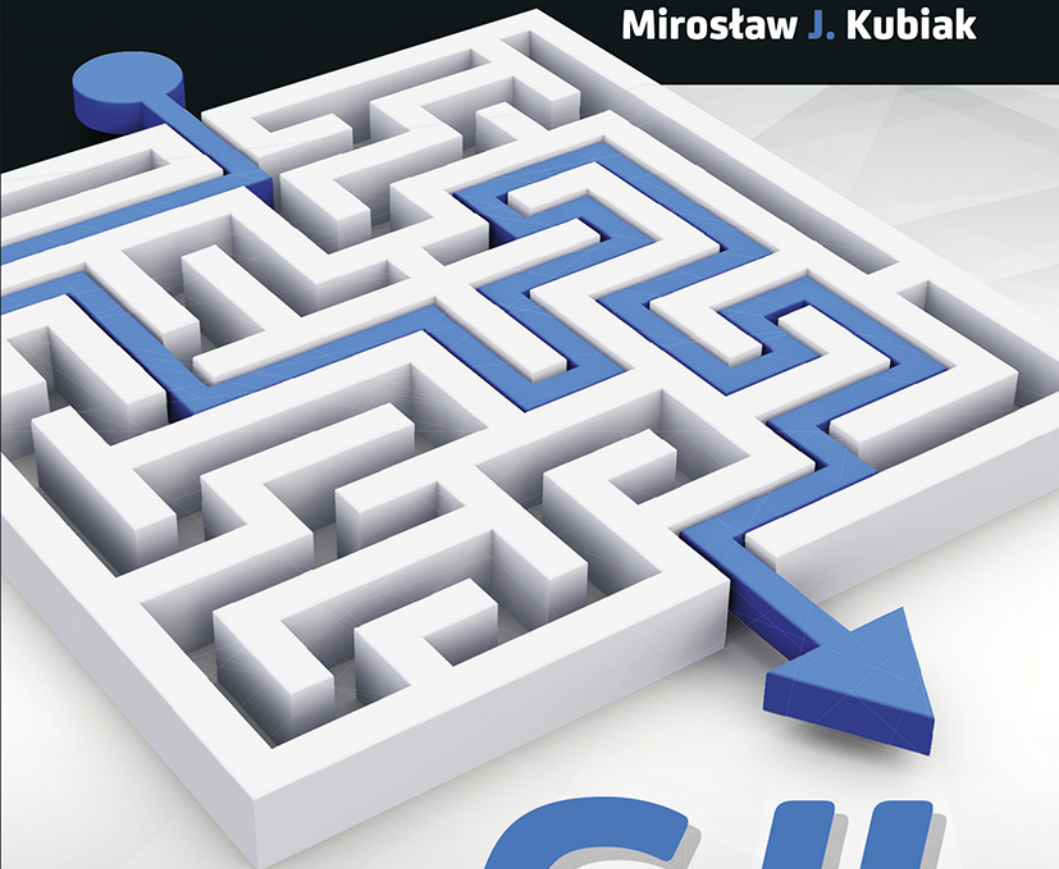


Mirostaw J. Kubiak



# C#

**Zadania z programowania  
z przykładowymi rozwiązaniami**

WYDANIE III

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](mailto: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/cshza3>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

ISBN: 978-83-283-8200-8

Copyright © Helion S.A. 2021

Printed in Poland.

- Kup książkę
- Poleć książkę
- Oceń książkę

- Księgarnia internetowa
- Lubię to! » Nasza społeczność

# Spis treści

<b>Od autora</b>	<b>5</b>
<b>Rozdział 1. Jak język C# komunikuje się z użytkownikiem?</b>	<b>9</b>
Informacje ogólne	9
Obsługa sytuacji wyjątkowych	18
<b>Rozdział 2. Instrukcje sterujące przebiegiem programu — instrukcje wyboru</b>	<b>23</b>
Instrukcje wyboru	23
Instrukcja if ... else	24
Instrukcja switch ... case	24
<b>Rozdział 3. Instrukcje sterujące przebiegiem programu — instrukcje iteracyjne</b>	<b>35</b>
Instrukcje iteracyjne	35
Pętla for	36
Pętla do ... while	37
Pętla while	37
<b>Rozdział 4. Tablice i kolekcje</b>	<b>69</b>
Tablice	69
Kolekcje	69
Tablice jednowymiarowe	70
Tablice dwuwymiarowe	74
Pętla foreach	90
Działania na macierzach	97
Łańcuchy tekstowe	105
Konkatenacja	108

Programowanie uogólnione i klasy generyczne	109
Proste metody generyczne	110
Proste klasy generyczne	111
Listy generyczne	114
<b>Rozdział 5. Elementy programowania obiektowego</b>	<b>117</b>
Informacje ogólne	117
Klasy, pola, metody	118
Rekurencja	129
Klasa Osoba	134
Dziedziczenie	136
<b>Rozdział 6. Pliki tekstowe i pliki o dostępie swobodnym</b>	<b>141</b>
Informacje ogólne	141
Pliki tekstowe	141
Pliki o dostępie swobodnym	156
Serializacja	157
<b>Rozdział 7. Wprowadzenie do współbieżności</b>	<b>161</b>
Informacje ogólne	161
Wprowadzenie do programowania równoległego	162
Wielowątkowość	172
Mój pierwszy wątek	172
Praca z wątkami	176
Priorytety wątków	181
Klasa Task	183
Moje pierwsze zadanie	184
Praca z zadaniami	186
Synchronizacja zadań	188
<b>Rozdział 8. Podążając w kierunku funkcyjnego paradygmatu programowania</b>	<b>193</b>
Wstęp	193
Co to jest paradygmat programowania?	194
Czym jest programowanie funkcyjne?	195
Funkcyjna natura biblioteki LINQ	196
<b>Polecana literatura</b>	<b>199</b>
Bibliografia	199
Zbiory zadań z programowania	200

Rozdział 2.

# Instrukcje sterujące przebiegiem programu — instrukcje wyboru

*W tym rozdziale przedstawiłem typowe zadania, wraz z przykładowymi rozwiązaniami, wykorzystujące instrukcje wyboru.*

## Instrukcje wyboru

Instrukcje sterujące przebiegiem programu są jednym z najważniejszych elementów w każdym języku programowania. Instrukcje te, w połączeniu z wyrażeniami, umożliwiają zapisanie dowolnego algorytmu programu.

Instrukcje sterujące w języku C# można podzielić na:

- ◆ instrukcje wyboru,
- ◆ instrukcje iteracyjne (znane jako pętle),
- ◆ instrukcje skoku.

W niniejszym rozdziale przedstawiam typowe zadania z wykorzystaniem instrukcji wyboru, w rozdziale 3. natomiast zadania z wykorzystaniem instrukcji iteracyjnych.

W języku C# istnieją dwie instrukcje wyboru, które służą do przeprowadzania operacji na podstawie wartości wyrażenia:

- ◆ instrukcja `if ... else`,
- ◆ instrukcja `switch ... case`.

## Instrukcja `if ... else`

Instrukcja `if ... else` służy do sprawdzania poprawności wyrażenia warunkowego i — w zależności od tego, czy dany warunek jest prawdziwy, czy nie — pozwala wykonać różne bloki programu.

Jej ogólna postać jest następująca:

```
if (warunek)
{
    ..... // Instrukcje do wykonania, kiedy warunek jest prawdziwy.
}
else
{
    ..... // Instrukcje do wykonania, kiedy warunek jest fałszywy.
}
```

Blok `else` jest opcjonalny, a instrukcja warunkowa w wersji skróconej ma postać:

```
if (warunek)
{
    ..... // Instrukcje do wykonania, kiedy warunek jest prawdziwy.
}
```

## Instrukcja `switch ... case`

Instrukcja `switch ... case` pozwala w wygodny i przejrzysty sposób sprawdzić ciąg warunków i wykonać kod w zależności od tego, czy są one prawdziwe, czy fałszywe. Jej ogólna postać jest następująca:

```
switch (wyrażenie)
{
    case wartość_1 : instrukcje_1;
    break;
    case wartość_2 : instrukcje_2;
    break;
    .....
    case wartość_n : instrukcje_n;
```

```

        break;
        default : instrukcje;
    }

```

Instrukcja `break` przerywa wykonanie całego bloku `case`. **UWAGA:** jej brak może doprowadzić do uzyskania nieoczekiwanych wyników i pojawienia się błędów w programie.

### Zadanie

#### 2.1

Napisz program, który dla trzech liczb,  $a$ ,  $b$ ,  $c$ , wprowadzonych z klawiatury sprawdza, czy tworzą one trójkę pitagorejską.



Wskazówka

W teorii liczb trójka pitagorejska to takie trzy liczby całkowite dodatnie  $a$ ,  $b$ ,  $c$ , które spełniają równanie Pitagorasa:  $a^2 + b^2 = c^2$ .

### Listing 2.1. Przykładowe rozwiązanie

```

using System;

namespace Zadanie_21 // Zadanie 2.1.
{
    class Program
    {
        static void Main(string[] args)
        {
            int a, b, c;

            Console.WriteLine("Program sprawdza, czy wczytane liczby a,
            ↪b, c to trójka pitagorejska.");
            Console.WriteLine("Podaj a.");
            a = int.Parse(Console.ReadLine());
            Console.WriteLine("Podaj b.");
            b = int.Parse(Console.ReadLine());
            Console.WriteLine("Podaj c.");
            c = int.Parse(Console.ReadLine());

            if ((a * a + b * b) == c * c)
            {
                Console.Write("Liczby ");
                Console.Write("a = " + a + ", ");
                Console.Write("b = " + b + ", ");
                Console.Write("c = " + c);
                Console.WriteLine(" są trójką pitagorejską.");
            }
            else
            {
                Console.Write("Liczby ");

```

```

        Console.WriteLine("a = " + a + ", ");
        Console.WriteLine("b = " + b + ", ");
        Console.WriteLine("c = " + c);
        Console.WriteLine(" nie są trójką pitagorejską.");
    }
}
}
}

```

Sprawdzenie twierdzenia Pitagorasa dla wczytanych liczb a, b, c zostało zawarte w następujących liniach kodu:

```

if ((a * a + b * b) == c * c)
{
    Console.WriteLine("Liczby ");
    Console.WriteLine("a = " + a + ", ");
    Console.WriteLine("b = " + b + ", ");
    Console.WriteLine("c = " + c);
    Console.WriteLine(" są trójką pitagorejską.");
}
else
{
    Console.WriteLine("Liczby ");
    Console.WriteLine("a = " + a + ", ");
    Console.WriteLine("b = " + b + ", ");
    Console.WriteLine("c = " + c);
    Console.WriteLine(" nie są trójką pitagorejską.");
}

```

Łatwo sprawdzić, że liczby  $a = 3$ ,  $b = 4$  i  $c = 5$  tworzą trójkę pitagorejską (spełniają twierdzenie Pitagorasa) i na ekranie pojawi się komunikat: *Liczby ... są trójką pitagorejską*, natomiast liczby  $a = 1$ ,  $b = 2$  i  $c = 3$  nie tworzą trójki pitagorejskiej (nie spełniają twierdzenia Pitagorasa) i na ekranie pojawi się komunikat: *Liczby ... nie są trójką pitagorejską*.

Rezultat działania programu dla  $a = 9$ ,  $b = 12$  i  $c = 15$  można zobaczyć na rysunku 2.1.

### Rysunek 2.1.

*Efekt działania programu*  
Zadanie 2.1

```

Program sprawdza, czy wczytane liczby a, b, c to trójka pitagorejska.
Podaj a.
9
Podaj b.
12
Podaj c.
15
Liczby a = 9, b = 12, c = 15 są trójką pitagorejską.

```



**Zadanie****2.2**

Napisz program, który z wykorzystaniem instrukcji `if` oblicza pierwiastki równania kwadratowego  $ax^2 + bx + c = 0$ , w którym zmienne `a`, `b`, `c` to liczby rzeczywiste wprowadzane z klawiatury. Wszystkie zmienne wyświetlamy na ekranie z dokładnością do dwóch miejsc po przecinku.

**Listing 2.2.** *Przykładowe rozwiązanie*

```
using System;
using static System.Math;

namespace Zadanie_22 //Zadanie 2.2.
{
    class Program
    {
        static void Main(string[] args)
        {
            double a, b, c, delta, x1, x2;

            Console.WriteLine("Program oblicza pierwiastki równania
↳ax ^ 2 + bx + c = 0.");
            Console.WriteLine("Podaj a.");
            a = double.Parse(Console.ReadLine());

            if (a == 0)
            {
                Console.WriteLine("Niedozwolona wartość współczynnika
↳a.");
            }
            else
            {
                Console.WriteLine("Podaj b.");
                b = double.Parse(Console.ReadLine());
                Console.WriteLine("Podaj c.");
                c = double.Parse(Console.ReadLine());

                delta = b * b - 4 * a * c;

                if (delta < 0)
                {
                    Console.WriteLine();
                    Console.Write("Dla ");
                    Console.Write("a = {0}, ", a);
                    Console.Write("b = {0}, ", b);
                    Console.Write("c = {0} ", c);
                    Console.Write("brak pierwiastków rzeczywistych.");
                }
                else
                {
```



Jeśli  $\Delta < 0$ , to zostanie wyświetlony komunikat: *...brak pierwiastków rzeczywistych*.

Dla  $\Delta = 0$  równanie kwadratowe ma jeden pierwiastek podwójny, który obliczymy ze wzoru:

$$x_1 = -b / (2 * a);$$

Dla  $\Delta > 0$  równanie kwadratowe ma dwa pierwiastki, które obliczymy ze wzorów:

$$x_1 = (-b - \text{Sqrt}(\Delta)) / (2 * a);$$

$$x_2 = (-b + \text{Sqrt}(\Delta)) / (2 * a);$$

Dla na przykład  $a = 1$ ,  $b = 5$  i  $c = 4$  wartości pierwiastków równania wynoszą odpowiednio:  $x_1 = -4$  i  $x_2 = -1$ .

Dla na przykład  $a = 1$ ,  $b = 4$  i  $c = 4$  trójmian ma jeden pierwiastek podwójny:  $x_1 = -2$ .

Dla na przykład  $a = 1$ ,  $b = 2$  i  $c = 3$  trójmian nie ma pierwiastków rzeczywistych.

Rezultat działania programu dla  $a = 1$ ,  $b = 5$  i  $c = 4$  można zobaczyć na rysunku 2.2.

### Rysunek 2.2.

*Efekt działania*

*programu*

*Zadanie 2.2*

Program oblicza pierwiastki równania  $ax^2 + bx + c = 0$ .

Podaj a.

1

Podaj b.

5

Podaj c.

4

Dla  $a = 1$ ,  $b = 5$ ,  $c = 4$  trójmian ma dwa pierwiastki:  $x_1 = -4$ ,  $x_2 = -1$ .

### Zadanie

#### 2.3

Napisz program, który z wykorzystaniem instrukcji `switch` oblicza pierwiastki równania kwadratowego  $ax^2 + bx + c = 0$ , w którym zmienne  $a$ ,  $b$ ,  $c$  to liczby rzeczywiste wprowadzane z klawiatury. Wszystkie zmienne wyświetlamy z dokładnością do dwóch miejsc po przecinku.



Należy wprowadzić do programu zmienną pomocniczą `liczba_pierwiastków`.

### Listing 2.3. Przykładowe rozwiązanie

```
using System;
using static System.Math;

namespace Zadanie_23 //Zadanie 2.3.
{
    class Program
    {
        static void Main(string[] args)
        {
            double a, b, c, delta, x1, x2;
            byte liczba_pierwiastków = 0;

            Console.WriteLine("Program oblicza pierwiastki równania
            ↳  $ax^2 + bx + c = 0$ .");
            Console.WriteLine("Podaj a.");
            a = double.Parse(Console.ReadLine());

            if (a == 0)
            {
                Console.WriteLine("Niedozwolona wartość współczynnika
                ↳ a.");
            }
            else
            {
                Console.WriteLine("Podaj b.");
                b = double.Parse(Console.ReadLine());
                Console.WriteLine("Podaj c.");
                c = double.Parse(Console.ReadLine());

                delta = b * b - 4 * a * c;

                if (delta < 0) liczba_pierwiastków = 0;
                if (delta == 0) liczba_pierwiastków = 1;
                if (delta > 0) liczba_pierwiastków = 2;

                switch (liczba_pierwiastków)
                {
                    case 0:
                        {
                            Console.WriteLine();
                            Console.Write("Dla ");
                            Console.Write("a = {0}, ", a);
                            Console.Write("b = {0}, ", b);
                            Console.Write("c = {0} ", c);
                        }
                    default:
                        {
                            Console.WriteLine();
                            Console.Write("Dla ");
                            Console.Write("a = {0}, ", a);
                            Console.Write("b = {0}, ", b);
                            Console.Write("c = {0} ", c);
                        }
                }
            }
        }
    }
}
```



**Rysunek 2.3.**

*Efekt działania programu*  
Zadanie 2.3

Program oblicza pierwiastki równania  $ax^2 + bx + c = 0$ .

Podaj a.

1

Podaj b.

4

Podaj c.

4

Dla  $a = 1, b = 4, c = 4$  trójmian ma jeden pierwiastek podwójny  $x_1 = -2$ .

**Zadanie****2.4**

Napisz program, który oblicza wartość niewiadomej  $x$  z równania  $ax + b = c$ . Wartości  $a, b, c$  należą do zbioru liczb rzeczywistych i są wprowadzane z klawiatury. Dodatkowo należy zabezpieczyć program na wypadek sytuacji, kiedy wprowadzona wartość  $a = 0$ . Wszystkie zmienne wyświetlamy z dokładnością do dwóch miejsc po przecinku.

**Listing 2.4. Przykładowe rozwiązanie**

```
using System;

namespace Zadanie_24 //Zadanie 2.4.
{
    class Program
    {
        static void Main(string[] args)
        {
            double a, b, c, x;

            Console.WriteLine("Program oblicza wartość x z równania
↳liniowego ax + b = 0.");
            Console.WriteLine("Podaj a.");
            a = double.Parse(Console.ReadLine());

            if (a == 0)
            {
                Console.WriteLine("Niedozwolona wartość współczynnika
↳a.");
            }
            else
            {
                Console.WriteLine("Podaj b.");
                b = double.Parse(Console.ReadLine());
                Console.WriteLine("Podaj c.");
                c = double.Parse(Console.ReadLine());
            }
        }
    }
}
```



```
Console.WriteLine("Program losuje liczbę od 0 do 9.  
↳Zgadnij ją.");  
  
losuj_liczbę = Round(10 * (r.NextDouble()));  
zgodnij_liczbę = double.Parse(Console.ReadLine());  
  
if (zgodnij_liczbę == losuj_liczbę)  
{  
    Console.WriteLine("Gratulacje! Zgadłeś liczbę!");  
}  
else  
{  
    Console.WriteLine("Bardzo mi przykro, ale wylosowana  
↳liczba to {0}.", losuj_liczbę);  
}  
}  
}
```

Funkcja `Round()` w poniższej linijce kodu:

```
losuj_liczbę = Round(10 * (r.NextDouble()));
```

umożliwia zaokrąglenie liczby zmiennoprzecinkowej do liczby całkowitej.

Rezultat działania programu można zobaczyć na rysunku 2.5.

### Rysunek 2.5.

*Efekt działania*

*programu*

*Zadanie 2.5*

```
Program losuje liczbę od 0 do 9. Zgadnij ją.  
9  
Gratulacje! Zgadłeś liczbę!
```



# 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** 

**C# — wieloparadygmataowy język programowania** opracowany przez firmę Microsoft — z roku na rok zdobywa coraz większą popularność wśród profesjonalistów z branży IT. Przejrzysta struktura kodu, wygoda stosowania, potężne możliwości i wsparcie ze strony platformy .NET — wszystko to sprawia, że są duże szanse, by ten trend utrzymywał się przez kolejne lata, zapewniając osobom znającym C# stały strumień ofert pracy w najlepszych firmach informatycznych świata. Jeśli chcesz należeć do tego grona, sięgnij po odpowiednie źródło wiedzy.

**Nowe, rozszerzone wydanie** książki *C#. Zadania z programowania z przykładowymi rozwiązaniami*, zawierające cenne wskazówki i informacje na temat najnowszych wersji języka, pomoże Ci skutecznie rozwinąć umiejętności programistyczne. Dowiesz się, jak komunikować się z użytkownikiem programu, prawidłowo i wydajnie korzystać z instrukcji sterujących, przechowywać dane przy użyciu tablic, łańcuchów znakowych i kolekcji obiektów, a także odczytywać i zapisywać pliki tekstowe i binarne. Opanujesz również podstawy programowania obiektowego, funkcyjnego i współbieżnego. A wszystko to na konkretnych przykładach i z naciskiem na użycie w praktyce.

- Operacje wejścia-wyjścia i obsługa wyjątków
- Instrukcje warunkowe i instrukcje pętli
- Tablice, łańcuchy znakowe, kolekcje
- Operacje na plikach i strumieniach
- Podstawy programowania obiektowego
- Wprowadzenie do współbieżności
- Podstawy programowania funkcyjnego

**Poszerz swoją wiedzę o C# — i dołącz do najlepszych!**

	<p><i>Sprawdź nasze szkolenia!</i></p> <p><b>SKOLENIA</b></p>  <p><b>AKADEMIA IT &amp; BUSINESS</b></p> <p><b>HELIONSZKOLENIA.PL</b></p>	<p><b>KOD KORZYŚCI</b> Sięgnij po więcej! ▶</p>  <p>ISBN 978-83-283-8200-8</p>  <p>9 788328 382008</p> <p>Cena: 44,90 zł</p>
<p> <b>helion.pl</b></p> <p> <b>HELION SA</b> ul. Kościuszki 1c 44-100 Gliwice tel.: 32 230 98 63 helion@helion.pl</p>		
<p><b>INFORMATYKA W NAJLEPSZYM WYDANIU</b></p>		