Mirosław J. Kubiak

C#

Zadania z programowania z przykładowymi rozwiązaniami

C# w analizie konkretnych przykładów

- *Instrukcje sterujące przebiegiem programu*
- *Tablice i kolekcje*
- *Programowanie obiektowe i pliki tekstowe*

Helion

Redaktor prowadzący: Ewelina Burska

Projekt okładki: Jan Paluch

Materiały graficzne na okładce zostały wykorzystane za zgodą Shutterstock.

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)

Kody wykorzystane w książce można znaleźć pod adresem:
ftp://ftp.helion.pl/przyklady/cshzap.zip

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
http://helion.pl/user/opinie?cshzap
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

ISBN: 978-83-246-3981-6

Printed in Poland.

Spis treści

Rozdział 2.

Instrukcje sterujące przebiegiem programu — instrukcje wyboru

W tym rozdziale przedstawiono typowe zadania, wraz z przykładowymi rozwiązaniami, z wykorzystaniem instrukcji wyboru.

Instrukcje wyboru

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

Instrukcje sterujące w języku C# można podzielić na:

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

W rozdziale 2. zostaną przedstawione typowe zadania z wykorzystaniem instrukcji wyboru, w rozdziale 3. zaś 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 na wykonanie różnych bloków 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.

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 Zadanie21 // 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.Write(" są trójką pitagorejską.");
            }
            else
            {
                Console.Write("Liczby ");
                Console.Write("a = " + a + ", ");
                Console.Write("b = " + b + ", ");
                Console.Write("c = " + c);
                Console.Write(" nie są trójką pitagorejską.");
            }

            Console.Read(); // naciśnij klawisz Enter
        }
    }
}
```

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

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

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

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

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

Rysunek 2.1. *Efekt działania programu Zadanie 2.1*

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;

namespace Zadanie22 // 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. Naciśnij
                                   klawisz ENTER.");
            }
            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
            {
                if (delta == 0)
                {
                    x1 = -b/(2*a);
                    Console.WriteLine();
                    Console.Write("Dla ");
                    Console.Write("a = {0}, ", a);
                    Console.Write("b = {0}, ", b);
                    Console.Write("c = {0} ", c);
                    Console.Write("trójmian ma jeden pierwiastek
                                    podwójny x1 = {0:##.##}.", x1);
                }
                else
                {
                    x1 = (-b-Math.Sqrt(delta))/(2*a);
                    x2 = (-b+Math.Sqrt(delta))/(2*a);
                    Console.WriteLine();
                    Console.Write("Dla ");
                    Console.Write("a = {0}, ", a);
                    Console.Write("b = {0}, ", b);
                    Console.Write("c = {0} ", c);
                    Console.WriteLine("trójmian ma dwa pierwiastki: x1 =
                                        {0:##.##}, x2 = {1:##.##}.", x1, x2);
                }
            }
        }

        Console.Read(); // naciśnij klawisz Enter
    }
  }
}
```

W pierwszej części programu sprawdzamy, czy wartość współczynnika `a` jest równa zero. Ilustrują to następujące linijki kodu:

```
if (a==0)
        {
           Console.WriteLine("Niedozwolona wartość współczynnika a. Naciśnij
                              klawisz ENTER.");
        }
```

```
else
{
.............
 }
```

Jeśli wartość współczynnika `a = 0`, to zostanie wyświetlony komunikat: *Niedozwolona wartość współczynnika a...* i program zostanie zakończony. Dla `a` różnego od zera program będzie oczekiwał na wprowadzenie wartości `b` i `c`. Po ich wprowadzeniu zostanie obliczona `delta` według wzoru:

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

Jeśli `delta < 0`, to zostanie wyświetlony komunikat: *...brak pierwiastków rzeczywistych*.

Dla `delta = 0` równanie kwadratowe ma jeden pierwiastek podwójny, który obliczymy ze wzoru:

```
x1 = -b/(2*a);
```

Dla `delta > 0` równanie kwadratowe ma dwa pierwiastki, które obliczymy ze wzorów:

```
x1 = (-b-Math.Sqrt(delta))/(2*a);
x2 = (-b+Math.Sqrt(delta))/(2*a);
```

Dla np. `a = 1, b = 5, c = 4` wartości pierwiastków równania wynoszą odpowiednio: `x1 = -4` i `x2 = -1`.

Dla np. `a = 1, b = 4, c = 4` trójmian ma jeden pierwiastek podwójny: `x1 = -2`.

Dla np. `a = 1, b = 2, c = 3` trójmian nie ma pierwiastków rzeczywistych.

Rezultat działania programu dla `a = 1, b = 5, c = 4` można zobaczyć na rysunku 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: x1 = -4, x2 = -1.
```

Rysunek 2.2. *Efekt działania programu Zadanie 2.2*

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_pierwiastkow`.

Listing 2.3. *Przykładowe rozwiązanie*

```
using System;

namespace Zadanie23 // Zadanie 2.3
{
    class Program
    {
        static void Main(string[] args)
        {
            double a, b, c, delta, x1, x2;
            byte liczba_pierwiastkow = 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. Naciśnij
                                   klawisz ENTER.");
            }
            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_pierwiastkow = 0;
                if (delta == 0) liczba_pierwiastkow = 1;
                if (delta > 0) liczba_pierwiastkow = 2;

                switch (liczba_pierwiastkow)
                {
                    case 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.");
                        }

                        break;
                    case 1:
                        {
                            x1 = -b/(2*a);
                            Console.WriteLine();
                            Console.Write("Dla ");
                            Console.Write("a = {0}, ", a);
                            Console.Write("b = {0}, ", b);
                            Console.Write("c = {0} ", c);
                            Console.Write("trójmian ma jeden pierwiastek
                                           podwójny x1 = {0: ##.##}.", x1);
```

```
                    }
                    break;

                case 2:
                    {
                        x1 = (-b-Math.Sqrt(delta))/(2*a);
                        x2 = (-b+Math.Sqrt(delta))/(2*a);
                        Console.WriteLine();
                        Console.Write("Dla ");
                        Console.Write("a = {0}, ", a);
                        Console.Write("b = {0}, ", b);
                        Console.Write("c = {0} ", c);
                        Console.Write("trójmian ma dwa pierwiastki x1 =
                                        {0: ##.##} i ", x1);
                        Console.Write("x2 = {0: ##.##}.", x2);
                    }
                    break;
            }
        }
        Console.Read(); // naciśnij klawisz Enter
    }
  }
}
```

Zmienna pomocnicza `liczba_pierwiastkow` przyjmuje trzy wartości w zależności od znaku zmiennej `delta`. Ilustrują to następujące linijki kodu:

```
if (delta < 0) liczba_pierwiastkow = 0;
if (delta == 0) liczba_pierwiastkow = 1;
if (delta > 0) liczba_pierwiastkow = 2;
```

Rezultat działania programu dla `a = 1`, `b = 4`, `c = 4` można zobaczyć na rysunku 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 x1 = - 2.
```

Rysunek 2.3. *Efekt działania programu Zadanie 2.3*

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 Zadanie24 // 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. Naciśnij
                                    klawisz ENTER.");
            }
            else
            {
                Console.WriteLine("Podaj b.");
                b = double.Parse(Console.ReadLine());
                Console.WriteLine("Podaj c.");
                c = double.Parse(Console.ReadLine());

                x = (c-b)/a;

                Console.WriteLine();
                Console.Write("Dla ");
                Console.Write("a = {0:##.##}, ", a);
                Console.Write("b = {0:##.##}, ", b);
                Console.Write("c = {0:##.##} ", c);
                Console.Write("wartość x = {0:##.##}.", x);
            }
            Console.Read(); // naciśnij klawisz Enter
        }
    }
}
```

Rezultat działania programu można zobaczyć na rysunku 2.4.

```
Program oblicza wartość x z równania liniowego ax+b = 0.
Podaj a.
1
Podaj b.
6
Podaj c.
2

Dla a = 1, b = 6, c = 2 wartość x = - 4.
```

Rysunek 2.4. *Efekt działania programu Zadanie 2.4*

Zadanie 2.5. Napisz program, w którym użytkownik zgaduje całkowitą liczbę losową z przedziału od 0 do 9 generowaną przez komputer.

W języku C# liczby pseudolosowe generujemy za pomocą klasy:

```
Random r = new Random(); .
```

Listing 2.5. *Przykładowe rozwiązanie*

```
using System;

namespace Zadanie25 // Zadanie 2.5
{
    class Program
    {
        static void Main(string[] args)
        {
            double losuj_liczbe, zgadnij_liczbe;

            Console.WriteLine("Program losuje liczbę od 0 do 9. Zgadnij ją.");
            Random r = new Random();
            losuj_liczbe = Math.Round(10*(r.NextDouble()));
            zgadnij_liczbe = double.Parse(Console.ReadLine());

            if (zgadnij_liczbe == losuj_liczbe)
            {
                Console.WriteLine("Gratulacje! Zgadłeś liczbę!");
            }
            else
            {
                Console.WriteLine("Bardzo mi przykro, ale wylosowana liczba
                                   to {0}.", losuj_liczbe);
            }

            Console.Read(); // naciśnij klawisz Enter
        }
    }
}
```

Funkcja `Round()` w poniższej linijce kodu:

```
losuj_liczbe = Math.Round(10*(r.NextDouble()));
```

umożliwia zaokrąglenie liczby zmiennoprzecinkowej do liczby całkowitej.

Rezultat działania programu można zobaczyć na rysunku 2.5.

```
Program losuje liczbę od 0 do 9. Zgadnij ją.
5
Bardzo mi przykro, ale wylosowana liczba to 2.
```

Rysunek 2.5. *Efekt działania programu Zadanie 2.5*

» C#, obecny na rynku od ponad dziesięciu lat, należy do tych języków programowania, bez których trudno się obejść — również dlatego, że jest jedynym językiem zaprojektowanym specjalnie dla platformy .NET i środowiska uruchomieniowego CLR. Dobry programista, student lub nauczyciel informatyki, a także każdy człowiek zainteresowany programowaniem powinien znać podstawy tego języka i umieć rozwiązywać konkretne zadania. Podobnie zresztą powinien opanować najważniejsze zagadnienia dotyczące programowania w językach Java, C++ i Turbo Pascal oraz umieć stosować je w praktyce. Ten rewelacyjny zbiór zadań pozwala szybko i przy minimalnym wysiłku opanować umiejętności programistyczne w obiektowym języku C# poprzez analizę typowych rozwiązań stosowanych w działających programach.

» Książka ***C#. Zadania z programowania z przykładowymi rozwiązaniami*** na konkretnych przykładach pokaże Ci, jak program napisany w języku C# komunikuje się z użytkownikiem poprzez operacje wejścia-wyjścia. Dowiesz się, jakie operatory będą Ci potrzebne i do czego można ich używać. Poznasz instrukcje sterujące przebiegiem programu (iteracyjne oraz wyboru), a także funkcje tablic jednowymiarowych i dwuwymiarowych oraz kolekcji. Następnie nauczysz się wykorzystywać obiektowe właściwości języka C# oraz zapisywać odpowiednie informacje w plikach tekstowych — a wszystko to zajmie Ci zaledwie chwilę. Rozwiązanie zamieszczonych tu zadań to droga na skróty do pełnego zrozumienia i zastosowania zalet języka C#.

- *Komunikacja języka C# z użytkownikiem*
- *Instrukcje wyboru*
- *Instrukcje iteracyjne*
- *Tablice i kolekcje*
- *Programowanie obiektowe*
- *Pliki tekstowe*

Na skróty do celu!

helion.pl
księgarnia internetowa

Nr katalogowy: 7722

Księgarnia internetowa
http://helion.pl

Zamówienia telefoniczne:
0 801 339900
0 601 339900

Helion

Sprawdź najnowsze promocje:
http://helion.pl/promocje
Książki najchętniej czytane:
http://helion.pl/bestsellery
Zamów informacje o nowościach:
http://helion.pl/nowosci

Helion SA
ul. Kościuszki 1c, 44-100 Gliwice
tel.: 32 230 98 63
e-mail: helion@helion.pl
http://helion.pl

Cena: 24,90 zł

ISBN 978-83-246-3981-6

9 788324 639816

Informatyka w najlepszym wydaniu