

Andrzej Stasiewicz



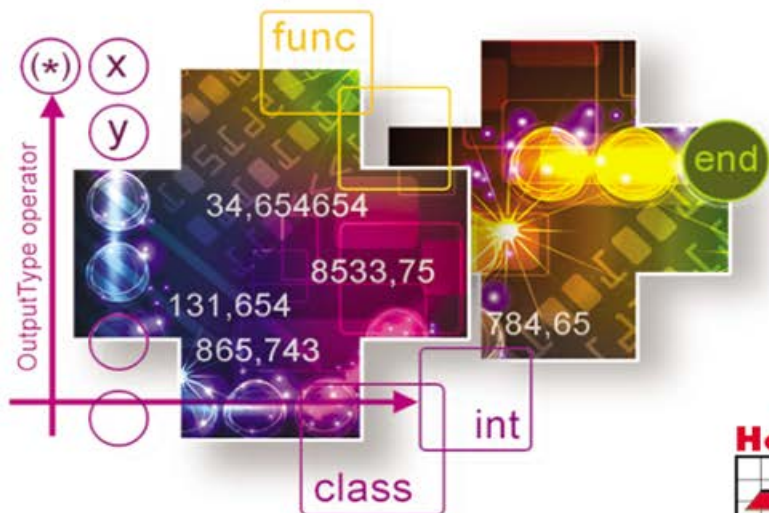
ĆWICZENIA

C++11

Nowy standard

Wypróbuj **nowe możliwości C++!**

Dowiedz się, co nowego w języku C++
Poznaj znaczenie najważniejszych rozszerzeń
Nauucz się wykorzystywać je w praktyce



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: Michał Mrowiec

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?cwcp11>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Kody źródłowe wybranych przykładów dostępne są pod adresem:

<ftp://ftp.helion.pl/przyklady/cwcp11.zip>

ISBN: 978-83-246-3935-9

Copyright © Helion 2012

Printed in Poland.

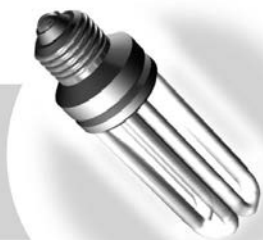
- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

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

Spis treści

Wstęp	5
Rozdział 1. Narzędzia programistyczne	9
Rozdział 2. Słowo kluczowe <code>auto</code> , czyli kompilator określa typ	13
Rozdział 3. Słowo kluczowe <code>decltype</code> , czyli typ taki sam jak tamten	19
Rozdział 4. Słowo kluczowe <code>constexpr</code> , czyli wyrażenia stałe	23
Rozdział 5. Słowo kluczowe <code>nullptr</code>	31
Rozdział 6. Lepsze typy wyliczeniowe <code>enum class</code>	35
Rozdział 7. Inicjalizowanie tablic	39
Rozdział 8. Inicjalizowanie klas na podobieństwo tablic	47
Rozdział 9. Krotki (rekordy)	53
Rozdział 10. Metody oznaczone <code>default</code> lub <code>delete</code>	61
Rozdział 11. Bezpieczne wskaźniki <code>unique_ptr</code> i <code>shared_ptr</code>	69
Rozdział 12. Kopiowanie i przenoszenie	79
Rozdział 13. Pętla <code>for(... : ...)</code> dla kolekcji danych	87
Rozdział 14. Prostsze tworzenie obiektów	91

Rozdział 15. Słowo kluczowe explicit i mocniejsza ochrona przed przypadkowymi konwersjami	95
Rozdział 16. Operator sizeof() zna rozmiary elementów klasy	99
Rozdział 17. Szablony ze zmienną liczbą argumentów	101
Rozdział 18. Funkcje i wyrażenia lambda	113
Zakończenie	123



13

Pętla `for(... : ...)` dla kolekcji danych



Standard `c++11` wprowadza uproszczoną pętlę `for(...)`, przebiegającą przez całą kolekcję podaną jako argument. Pętla ta jest odpowiednikiem znanej z innych języków pętli `foreach(...)` — rób coś dla każdego elementu kolekcji.

Ć W I C Z E N I E

13.1 Wypisanie wszystkich elementów tablicy za pomocą nowej pętli `for(...)`

Zadeklaruj tablicę i za pomocą nowej pętli `for()` dla kolekcji wyprowadź na ekran jej elementy (rysunek 13.1):

```
...  
int tablica[5] = {1, 2, 3, 4, 5};  
for(int element : tablica)  
{  
    cout << element << endl;  
}  
...
```

Nowa pętla `for()` ma dwa pola: pole określenia zmiennej o typie zgodnym z typami w tablicy i pole określenia samej tablicy.

```

c:\MinGW32\a.exe
1
2
3
4
5
Aby kontynuować, naciśnij dowolny klawisz . . .

```

Rysunek 13.1. Nowa pętla for() przebiega po tablicy i wyświetla jej elementy

Ć W I C Z E N I E

13.2 Modyfikacja wszystkich elementów tablicy za pomocą nowej pętli for(...)

Zadeklaruj dla odmiany tablicę `vector` i za pomocą nowej pętli `for()` dla kolekcji zmodyfikuj elementy kolekcji:

```

...
#include <vector>
...
int main()
{
    vector<double> v={1, 2, 3};
    for( double &r : v)
    {
        r = 3.14;
    }

    cout << v[ 0] << ", " << v[ 1] << ", " << v[ 2] << endl;

    system("PAUSE");
    return EXIT_SUCCESS;
}

```

W stosunku do poprzedniego ćwiczenia zmienna robocza `r` została zadeklarowana jako referencja (odwołanie) i pętla `for` dla kolekcji umieszcza pod nią kolejno wszystkie elementy tablicy. Modyfikacja referencji oznacza modyfikację oryginalnego elementu tablicy.

```

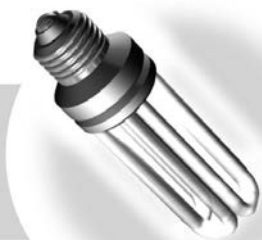
c:\MinGW32\a.exe
3.14, 3.14, 3.14
Aby kontynuować, naciśnij dowolny klawisz . . .

```

Rysunek 13.2. Modyfikacja kolekcji za pomocą nowej pętli dla zakresów dla kolekcji

Podsumowanie

Większość iteracji po kolekcjach odbywa się „od początku do końca”.
Dlaczego tę pętlę otrzymaliśmy tak późno?



14

Prostsze tworzenie obiektów



W standardzie c++11 znajdujemy dwa nowe drobiazgi:

- ❑ zadeklarowane w typach użytkownika (klasach, strukturach) dane mogą być inicjalizowane bezpośrednio, a nie — jak dotąd — w konstruktorach;
- ❑ konstruktory typów użytkownika mogą wywoływać inne konstruktory, co dotychczas też było zabronione.

Ć W I C Z E N I E

14.1 Inicjalizowanie ustroju klasy bezpośrednio i za pomocą konstruktorów

Zadeklaruj klasę, w której zademonstrujesz zarówno bezpośrednie inicjalizowanie zmiennej, jak i stare rozwiązanie — czyli inicjalizację zmiennej za pomocą konstruktora:

```
...
class Stara
{
public:
    int a;
    Stara():a(17){cout << "Konstruktor" << endl;}
};
```

```

class Nowa
{
public:
    int a = 17;
    Nowa(){ cout << "Konstruktor" << endl;}
};

int main()
{
    Stara s;
    cout << s.a << endl;

    Nowa n;
    cout << n.a << endl;

    system("PAUSE");
    return EXIT_SUCCESS;
}

```

Dwie klasy mają pewną zmienną. Klasa *Stara* inicjalizuje tę zmienną za pomocą do tej pory jedynej dostępnej metody, czyli w konstruktorze (tutaj za pomocą tzw. listy inicjalizacyjnej konstruktora). Klasa *Nowa* inicjalizuje swoją zmienną przez bezpośrednie przypisanie wartości w momencie deklaracji klasy. Konstruktory mogą ewentualnie zmienić tę wartość.

```

C:\WINDOWS\system32\cmd.exe
c:\MinGW\bin>g++ -std=c++0x 14_1.cpp
14_1.cpp:17:22: error: ISO C++ forbids initialization of member 'a'
14_1.cpp:17:22: error: making 'a' static
14_1.cpp:17:22: error: ISO C++ forbids in-class initialization of non-const static member 'a'
c:\MinGW\bin>_

```

Rysunek 14.1. Niestety, kompilator jeszcze nie zna bezpośredniego inicjalizowania. Zgodnie z dotychczasowym standardem języka domaga się, by bezpośrednio inicjalizowana zmienna była statyczna i stała

Ć W I C Z E N I E

14.2 Konstruktor może inicjalizować klasę przez wywołanie innego konstruktora

Przygotuj klasę i zaopatrz ją w takie konstruktory, by jeden z nich wywoływał inny:

```

...
class T
{

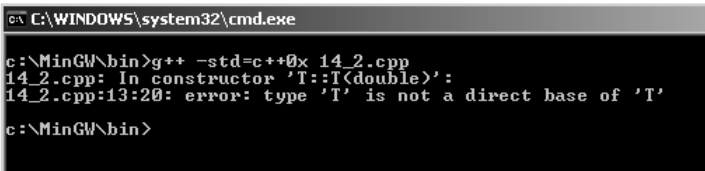
```

```
private:
    int a;
public:
    T():a(17){cout << "Konstruktor 'T()'" << endl;}
    T(double r):T(){cout << "Konstruktor 'T(double r)'" << endl;}
};

int main()
{
    T a;
    T b(3.14);

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

Typ `T` w konstruktorze z argumentem wywołuje inny konstruktor tejże klasy. Do tej pory nie było to możliwe — każdy konstruktor musiał przeprowadzać niezależnie inicjalizację obiektu. Konstruktor mógł wywoływać innego konstruktora tylko w obrębie drzewa dziedziczenia. Mówiąc inaczej — konstruktor klasy pochodnej wywoływał konstruktora klasy bazowej. W obrębie jednej klasy takie wywołania były niemożliwe.



```
c:\WINDOWS\system32\cmd.exe
c:\MinGW\bin>g++ -std=c++0x 14_2.cpp
14_2.cpp: In constructor 'T::T(double)':
14_2.cpp:13:20: error: type 'T' is not a direct base of 'T'
c:\MinGW\bin>
```

Rysunek 14.2. Kompilator jeszcze nie realizuje opisywanego tu usprawnienia. Komunikat głosi, że wywołanie konstruktora przez konstruktor byłoby możliwe, gdyby nasza klasa dziedziczyła po sobie samej

Podsumowanie

Chyba każdy młody programista, który deklarował pierwszą klasę w swoim życiu, zastanawiał się, dlaczego nie może zainicjować jej pól wartościami. Byłyby to jakby wartości domyślne, wstępne, które konstruktory mogą zmienić.

Dodatkowo otrzymujemy mechanizm wywoływania jednego konstruktora przez innego. Jeśli obydwa konstruktory tak samo (lub podobnie) inicjalizują klasę — po co mamy powtarzać ten sam kod?

PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ 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 WYDAWNICZA



Helion SA

C++11. Nowy standard. ĆWICZENIA

Opanuj nowości
w standardzie C++11. Praktycznie!



C++ to jeden z najpopularniejszych języków programowania. Nie dzieje się tak bez powodu — jego duże możliwości, logiczna struktura oraz zwiezłość i przejrzystość kodu zdobywają serca zarówno amatorów traktujących programowanie hobbystycznie, jak i profesjonalistów tworzących zaawansowane aplikacje dla wielkich korporacji. C++ to język żywy i jako taki stale się rozwija. Dowodem tego są kolejne aktualizacje standardu, w tym ostatnia, wprowadzona zaledwie kilka miesięcy temu. Choć w C++11 brak zmian o charakterze rewolucyjnym, proponowane ulepszenia mogą znacznie ułatwić programistom codzienną pracę. Niestety wielu z nas w ogóle z nich nie korzysta, ponieważ nie znamy potencjalnych korzyści, a często nawet nie mamy pojęcia o istnieniu niektórych nowości.

Czas to zmienić. Czas sięgnąć po książkę *C++11. Nowy standard. Ćwiczenia*. W prosty sposób prezentuje ona najciekawsze i najbardziej przydatne możliwości z najnowszego standardu języka. Autor przedstawia niezbędne narzędzia programistyczne oraz sposoby ich używania, opisuje nowe słowa kluczowe i sytuacje, w których należy je stosować, wyjaśnia zmiany wprowadzone w systemie typów języka oraz podaje nowe metody inicjalizacji tablic i klas. W książce została też poruszona tematyka krotek, bezpiecznych wskaźników, nowych rodzajów konstruktorów, dodatkowych opcji związanych z szablonami oraz funkcji i wyrażeń lambda. Wszystko to jest poparte krótkimi ćwiczeniami, dzięki którym utrwalisz wiedzę w praktyce.

- Narzędzia do tworzenia i kompilowania programów
- Nowe słowa kluczowe i ich znaczenie
- Nowe metody inicjalizacji tablic i klas
- Tworzenie i przetwarzanie rekordów
- Używanie bezpiecznych wskaźników
- Korzystanie z nowych wzorców funkcji i typów
- Możliwości związane z funkcjami i wyrażeniami lambda

helion.pl
księgarnia
internetowa

Nr katalogowy: 7518



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/novosci>

Helion SA

ul. Kościuszki 1c, 44-100 Gliwice

tel.: 32 230 98 63

e-mail: helion@helion.pl

<http://helion.pl>

sięgnij po WIECEJ



KOD KORZYŚCI

ISBN 978-83-246-3935-9



Cena 21,90 zł

Informatyka w najlepszym wydaniu

9 788324 639359