

Spis treści

Przedmowa 13

Podziękowania 15

Podziękowania do wydania polskiego 17

Wykaz skrótów 19

1. Wprowadzenie 21

- 1.1. Struktura książki 27
- 1.2. Konwencje formatowania 30
- 1.3. O kodzie i projektach 32

2. Wprowadzenie do programowania 35

- 2.1. Model sprzętowy 35
- 2.2. Ekosystem tworzenia oprogramowania 39
- 2.3. Etapy tworzenia oprogramowania 42
- 2.4. Reprezentacja i uruchamianie algorytmów 44
 - 2.4.1. Reprezentowanie algorytmów 44
 - 2.4.2. Korzystanie z kompilatorów dostępnych w Internecie 46
 - 2.4.3. Struktura programu w C++ 48
 - 2.4.4. Analiza kodu 49
 - 2.4.5. (✱) Budowa linuxowej postaci wykonywalnej 50
- 2.5. Przykładowy projekt – kalkulator procentu składanego 53
 - 2.5.1. Analiza procentu składanego 54
 - 2.5.2. Implementacja kalkulatora procentu składanego 55
 - 2.5.3. Budowanie i uruchamianie oprogramowania 58
- 2.6. Przykładowy projekt – zliczanie wystąpień znaków w tekście 59
 - 2.6.1. Analiza problemu i implementacja 59
 - 2.6.2. Uruchomienie kodu C++ za pomocą kompilatora dostępnego w Internecie 61
 - 2.6.3. Kod histogramu – wyjaśnienie 62
- 2.7. Podsumowanie 65
 - Pytania i ćwiczenia 65

3. Podstawy C++ 69

- 3.1. Stałe i zmienne – wbudowane typy danych, ich zakres oraz inicjalizacja 69
- 3.2. Przykładowy projekt – zbieranie ocen studentów 80

3.3.	Nasz przyjaciel debugger	84
3.4.	Podstawowa struktura danych – <code>std::vector</code>	87
3.5.	Przykładowy projekt – implementowanie macierzy w postaci wektora wektorów	93
3.6.	Specjalny wektor do przechowywania tekstu – <code>std::string</code>	95
3.7.	Słowo kluczowe <code>auto</code> oraz <code>decltype</code> do automatycznej dedukcji typu	101
3.8.	Popularne algorytmy standardowe	104
3.9.	Struktury: zbieranie obiektów różnego typu	108
3.10.	(✳) Tablice o stałym rozmiarze	113
3.10.1.	Wielowymiarowe tablice o stałym rozmiarze	115
3.11.	Referencje	117
3.12.	(✳) Wskaźniki	121
3.12.1.	Dostęp do obiektów za pomocą wskaźników	121
3.13.	Instrukcje	126
3.13.1.	Bloki instrukcji oraz dostęp do zmiennych – rola nawiasów klamrowych	127
3.13.2.	Instrukcje C++	129
3.13.2.1.	Instrukcje warunkowe	129
3.13.2.2.	Instrukcje pętli	135
3.13.2.3.	Instrukcje pomocnicze – <code>continue</code> oraz <code>break</code>	140
3.13.2.4.	Instrukcja <code>goto</code>	142
3.13.2.5.	Strukturalna obsługa wyjątków – instrukcja <code>try-catch</code>	142
3.14.	Funkcje	144
3.14.1.	Anatomia funkcji w C++	145
3.14.2.	Przekazywanie argumentów do i z funkcji	150
3.14.2.1.	Przekazywanie argumentów przez kopię (semantyka wartości)	151
3.14.2.2.	Pośrednie przekazywanie argumentów przez referencję	153
3.14.2.3.	(✳) Przekazywanie przez wskaźnik	155
3.14.3.	Mechanizm wywoływania funkcji i funkcje wbudowane	157
3.14.4.	Funkcje rekurencyjne i stos wywołań	159
3.14.5.	Przeciążanie funkcji – rozwiązywanie widoczności za pomocą przestrzeni nazw	160
3.14.6.	Funkcje <code>lambda</code>	163
3.14.7.	(✳) Więcej na temat funkcji <code>lambda</code>	168
3.14.8.	(✳) Wskaźniki do funkcji	174
3.14.9.	(✳) Funkcje w środowisku obiektowym	176
3.15.	Przykładowy projekt – opakowywanie obiektów strukturą z konstruktorem	178
3.15.1.	<code>EMatrix</code> w środowisku obiektowym	181
3.15.2.	Podstawowe operacje z użyciem <code>EMatrix</code>	182
3.15.3.	Operacje wejścia i wyjścia na <code>EMatrix</code>	184
3.15.4.	Proste operacje matematyczne na macierzy <code>EMatrix</code>	186
3.15.5.	Organizowanie plików projektu i uruchamianie aplikacji	188
3.15.6.	Rozszerzanie inicjalizacji macierzy za pomocą prostego generatora liczb losowych	191

3.16.	Przykładowy projekt – reprezentowanie równań kwadratowych	193
3.16.1.	Definicja klasy do reprezentowania wielomianów kwadratowych	194
3.16.2.	Implementacja składowych <i>TQuadEq</i>	194
3.16.3.	<i>TQuadEq</i> w akcji	205
3.17.	Przykładowy projekt – krotki i powiązania strukturalne do konwertowania liczb rzeczywistych	206
3.17.1.	Więcej o <code>std::tuple</code> i powiązaniu strukturalnym	210
3.17.2.	Jak napisać test jednostkowy oprogramowania	212
3.17.3.	Automatyzowanie testów jednostkowych – z użyciem standardowej biblioteki liczb losowych	213
3.18.	Projekt przykładowy – tworzenie komponentu kalkulatora walutowego	216
3.18.1.	Analiza problemu wymiany walut	216
3.18.2.	Projekt oprogramowania <i>CurrencyCalc</i>	219
3.18.3.	Klasa <i>TCurrency</i> reprezentująca rekordy walut	220
3.18.3.1.	Manipulatory wejścia/wyjścia C++	223
3.18.4.	Klasa <i>TCurrencyExchanger</i> do wymiany walut	225
3.18.5.	Łączenie wszystkiego w całość – kompletny program wymiany walut	230
3.19.	Operatory	236
3.19.1.	Podsumowanie operatorów C++	239
3.19.2.	Dalsze uwagi na temat operatorów	263
3.20.	Podsumowanie	264
	Pytania i ćwiczenia	265
4.	Zgłębianie programowania obiektowego	269
4.1.	Podstawowe reguły oraz filozofia projektowania i programowania obiekтового	270
4.2.	Anatomia klasy	274
4.2.1.	Konwencja nazewnictwa i samodokumentujący się kod	277
4.3.	Reguły uzyskiwania dostępu do składowych klasy	277
4.4.	Przykładowy projekt – klasa <i>TComplex</i> do przeciążania operatorów	280
4.4.1.	Definicja klasy <i>TComplex</i>	281
4.4.2.	Definicja składowych klasy <i>TComplex</i>	287
4.4.3.	Funkcje testujące dla klasy <i>TComplex</i>	289
4.5.	Więcej o referencjach	293
4.5.1.	Referencje prawostronne i przekazywania	293
4.5.2.	Referencje kontra wskaźniki	298
4.5.3.	Pułapki z referencjami	299
4.6.	Przykładowy projekt – opanowywanie składowych klasy z użyciem klasy <i>TheCube</i>	301
4.6.1.	Automatyczna kontra jawna definicja konstruktorów	302
4.6.2.	Układ i semantyka obiektu <i>TheCube</i>	312
4.6.3.	Semantyka kopiowania płytkiego i głębokiego	313
4.6.4.	Semantyka konstruktora przenoszącego i przypisania przenoszącego	314
4.6.5.	Implementacja operatorów strumieniowania dla <i>TheCube</i>	316

4.6.6.	Sprawdzanie TheCube	318
4.7.	Projekt przykładowy – przenoszenie EMatrix do klasy	321
4.7.1.	Definicja klasy EMatrix	322
4.7.2.	Implementacja operatorów strumieniowania klasy	324
4.7.3.	Implementacja operatorów arytmetycznych	328
4.7.4.	Testowanie operacji na macierzach	330
4.8.	Wprowadzenie do szablonów i programowania uogólnionego	332
4.8.1.	Uogólnianie klasy przy użyciu szablonów	332
4.8.2.	(✳) Specjalizacje szablonów	337
4.8.3.	Funkcje szablonowe i sprawdzanie typu	338
4.8.4.	Przykładowy projekt – projektowanie klas szablonowych przy użyciu TStack	340
4.8.4.1.	Projekt i implementacja klasy TStackFor	342
4.8.4.2.	Testowanie TStack	345
4.8.5.	Szablonowe funkcje składowe	347
4.9.	Relacje między klasami – „zna”, „ma” oraz „jest”	350
4.10.	Przykładowy projekt – rozszerzanie funkcjonalności poprzez dziedziczenie klas z użyciem TComplexQuadEq	357
4.11.	Funkcje wirtualne i polimorfizm	363
4.12.	(✳) Więcej na temat mechanizmu wirtualnego	370
4.13.	(✳) Ciekawie rekurencyjny wzorzec szablonu i statyczny polimorfizm	373
4.14.	(✳) Klasy domieszki	377
4.15.	Przykładowy projekt – klasa TLongNumberFor do wydajnego przechowywania liczb o dowolnej długości	379
4.15.1.	Reprezentacja Binary-Coded Decimal	381
4.15.2.	Kolejność bajtów	381
4.15.3.	Definicja klasy TLongNumberFor	382
4.15.3.1.	Operacje konwersji typu	385
4.15.3.2.	Funkcja testująca TLongNumberFor	391
4.15.4.	Projektowanie klas dla numeru PESEL	392
4.15.4.1.	Agregowanie klasy PESEL	393
4.15.4.2.	Odziedziczona klasa PESEL	395
4.15.4.3.	Organizacja projektu LongNumber	396
4.15.5.	(✳) Rozszerzanie funkcjonalności klasy TLongNumberFor z użyciem wzorca pełnomocnika	398
4.15.5.1.	Definicja klasy Proxy	399
4.15.5.2.	Testowanie funkcjonalności klasy TLongNumberFor z użyciem wzorca pełnomocnika	402
4.16.	Silne typy	403
4.17.	Podsumowanie	405
	Pytania i ćwiczenia	405
5.	Zarządzanie pamięcią	409
5.1.	Rodzaje magazynów danych	409
5.2.	Dynamiczny przydział pamięci – jak unikać wycieków pamięci	416

5.2.1.	Wprowadzenie do inteligentnych wskaźników i zarządzania zasobami	419
5.2.1.1.	RAII i odwijanie stosu	420
5.3.	Inteligentne wskaźniki – omówienie z przykładami	421
5.3.1.	(✳) Więcej o <code>std::unique_ptr</code>	421
5.3.1.1.	Kontekst użycia <code>std::unique_ptr</code>	421
5.3.1.2.	Wzorec projektowy metody wytwórczej	435
5.3.1.3.	Niestandardowe usuwanie <code>unique_ptr</code>	438
5.3.1.4.	Konstrukcje do unikania podczas korzystania z <code>unique_ptr</code>	439
5.3.2.	(✳) Więcej o <code>shared_ptr</code> i <code>weak_ptr</code>	440
5.4.	Podsumowanie	443
	Pytania i ćwiczenia	443
6.	Zaawansowane programowanie obiektowe	445
6.1.	Obiekty funkcyjne	445
6.2.	Projekt przykładowy – rozszerzanie o wyszukiwanie walut w plikach XML oraz korzystanie z maszyny stanów i wyrażeń regularnych za pomocą biblioteki <code>regex</code>	452
6.2.1.	Dopasowywanie do wzorca za pomocą biblioteki wyrażeń regularnych	453
6.2.2.	Wzorec maszyny stanów	455
6.2.3.	Implementowanie klasy rozszerzonej	456
6.2.4.	Rozszerzenie projektu – wczytywanie informacji o walutach z internetu	463
6.2.5.	Uruchamianie rozszerzonej wersji <code>CurrencyCalc</code>	469
6.2.6.	Tworzenie biblioteki statycznej	474
6.2.7.	System plików C++	475
6.2.8.	Interfejs użytkownika	484
6.2.8.1.	Definicja klasy <code>CC_GUI</code>	485
6.2.8.2.	Definicje składowych klasy <code>CC_GUI</code> i mechanizm wywołania zwrotnego	489
6.2.8.3.	Uruchamianie aplikacji z GUI	497
6.3.	Zegary systemowe i pomiar czasu	498
6.4.	(✳) Mierzenie czasu wykonywania funkcji	502
6.5.	Klasa <code>Range</code>	505
6.5.1.	Programowanie funkcyjne i biblioteka <code>std::ranges</code>	510
6.6.	Przykładowy projekt – parsowanie wyrażeń	511
6.6.1.	Definiowanie wyrażeń języka i zasad gramatyki formalnej	512
6.6.2.	Projektowanie biblioteki przetwarzania wyrażeń	515
6.6.3.	Pierwszy interpreter poleceń	516
6.6.4.	Budowanie drzewa składniowego z użyciem wzorca projektowego kompozytu	520
6.6.4.1.	Wzorec projektowy kompozytu do definiowania węzłów drzewa	521
6.6.4.2.	Implementacja hierarchii <code>TNode</code> i współpraca z wizytatorami	522
6.6.4.3.	Implementacja klasy <code>ValueLeafNode</code>	525
6.6.4.4.	Implementacja klasy <code>BinOperator</code>	526
6.6.4.5.	Implementacja klasy <code>PlusOperator</code>	528
6.6.4.6.	Głębokie kopiowanie obiektów węzła – mechanizm prototypowania	529

6.6.5.	Interpreter do budowy drzew składniowych	531
6.6.6.	Stos dla inteligentnych wskaźników	536
6.6.7.	Przechodzenie po drzewach za pomocą wzorca projektowego wizytatora	540
6.6.7.1.	Wizytator ewaluujący wyrażenie	543
6.6.7.2.	Wizytator wypisujący wyrażenie	545
6.6.8.	Testowanie interpreterów	547
6.6.9.	Reprezentowanie wyrażeń na stosie w odwrotnej notacji polskiej	551
6.6.9.1.	Odwrócona notacja polska	551
6.6.9.2.	Algorytm do ewaluowania wyrażenia RPN	552
6.7.	Podsumowanie	558
	Pytania i ćwiczenia	558
7.	Arytmetyka komputerowa	563
7.1.	Reprezentacja wartości całkowitej	563
7.1.1.	Algorytm konwersji podstawy	565
7.1.2.	Reprezentacje szesnastkowe i ósemkowe	566
7.1.3.	Dodawanie binarne	567
7.1.4.	Wartości ujemne i odejmowanie	568
7.2.	Operacje przesunięcia binarnego	569
7.1.5.	Flagi kontroli arytmetycznej	570
7.1.6.	Reprezentowanie ułamków	573
7.2.	Operacje przesunięcia binarnego	577
7.3.	(✱) Przykładowy projekt – model programowy dla reprezentacji stałoprecyzyjnej	578
7.3.1.	Liczby stałoprecyzyjne i ich arytmetyka	579
7.3.2.	Definicja klasy <i>FxFor</i>	579
7.3.3.	Wybrane metody klasy <i>FxFor</i>	587
7.3.4.	Zastosowania dla <i>FxFor</i>	593
7.4.	Reprezentacje liczb zmiennoprecyzyjnych	596
7.4.1.	Reprezentacja liczb w formacie zmiennoprecyzyjnym	598
7.4.2.	Rozkład liczb zmiennoprecyzyjnych i konsekwencje obliczeniowe	602
7.4.3.	(✱) Błąd przybliżenia wartości rzeczywistej przy użyciu reprezentacji zmiennoprecyzyjnej	606
7.4.4.	Standard IEEE 754 dla arytmetyki zmiennoprecyzyjnej	608
7.4.5.	Standardowy model operacji zmiennoprecyzyjnych	611
7.4.6.	Obliczenia ze świadomością o błędach numerycznych	611
7.4.7.	Przykładowy projekt – ewaluacja algorytmów sumujących	617
7.4.8.	Przykładowy projekt – metoda Newtona do znajdowania miejsc zerowych funkcji	623
7.4.8.1.	Funkcja do obliczania pierwiastka kwadratowego na podstawie iteracji Newtona	627
7.5.	Podsumowanie	629
	Pytania i ćwiczenia	630

8.	Podstawy programowania równoległego	633
8.1.	Podstawowe zagadnienia związane z obliczeniami równoległymi	633
8.2.	Dodawanie równoległości do algorytmów standardowych	637
8.3.	Uruchamianie zadań asynchronicznych	640
8.4.	Zrównoleglanie za pomocą biblioteki OpenMP	643
8.4.1.	Uruchamianie zespołu wątków i zapewnianie ochrony wyłączonego dostępu	644
8.4.2.	Równoległa pętla for i operacje redukcji	646
8.4.3.	Równoległość dla dużych ilości danych	649
8.5.	Podsumowanie	658
	Pytania i ćwiczenia	658

Dodatek 661

A.1.	Dyrektywy preprocesora	661
A.2.	Krótkie wprowadzenie do języka C	666
A.2.1.	Tablice wbudowane	667
A.2.1.1.	Wielowymiarowe tablice o stałym rozmiarze	670
A.2.2.	Przekazywanie tablic do funkcji – funkcja main	671
A.2.3.	Struktury C	676
A.2.4.	Funkcje i operacje wejścia/wyjścia w C	677
A.2.5.	Unie	678
A.2.6.	Operacje wykonywane na pamięci i ciągach znaków	680
A.2.7.	Łączenie kodu C i C++	686
A.3.	Łączenie i organizacja binarna obiektów C/C++	686
A.4.	Graficzny interfejs użytkownika i interfejs sieci Web dla projektów C++	689
A.5.	Konwertowanie wartości bin, oct, dec i hex za pomocą FixBinCalc	691
A.6.	Zestaw narzędzi programistycznych	692
A.6.1.	Narzędzie do generowania projektów (CMake)	692
A.6.2.	Systemy kontroli wersji i repozytoria	697
A.6.3.	Profiler	698
A.7.	Testowanie oprogramowania	699
A.8.	Podsumowanie	706
	Pytania i ćwiczenia	706

Bibliografia 709

Indeks 713