

Spis treści

Wprowadzenie	19
Rozdział 1. Wprowadzenie do informatyki, internetu i sieci	31
1.1. Wprowadzenie	32
1.2. Hardware i software	32
1.2.1. Prawo Moore'a	32
1.2.2. Organizacja komputera	33
1.3. Hierarchia danych	35
1.4. Język maszynowy, język assemblera i język wysokiego poziomu	37
1.5. Język programowania C	38
1.6. Biblioteka standardowa C	40
1.7. C++ i inne oparte na C języki programowania	41
1.8. Technologia obiektowa	42
1.8.1. Samochód jako obiekt	42
1.8.2. Metoda i klasa	42
1.8.3. Tworzenie egzemplarza	42
1.8.4. Wielokrotne używanie klasy	43
1.8.5. Komunikat i wywołanie metody	43
1.8.6. Atrybut i zmienna egzemplarza	43
1.8.7. Hermetyzacja i ukrywanie informacji	43
1.8.8. Dziedziczenie	43
1.9. Typowe środowisko programowania w języku C	44
1.9.1. Faza 1. — tworzenie programu	45
1.9.2. Fazy 2. i 3. — przetwarzanie i kompilowanie programu	45
1.9.3. Faza 4. — linkowanie programu	45
1.9.4. Faza 5. — wczytywanie programu	46
1.9.5. Faza 6. — wykonywanie programu	46
1.9.6. Problemy, które mogą pojawić się podczas wykonywania programu	46
1.9.7. Standardowe strumienie wejścia, wyjścia i błędów	46
1.10. Przykładowa aplikacja w języku C utworzona na platformach Windows, Linux i macOS ...	46
1.10.1. Uruchomienie aplikacji napisanej w C w oknie wiersza poleceń systemu Windows	47
1.10.2. Uruchomienie aplikacji napisanej w C za pomocą GNU C w systemie Linux	50
1.10.3. Uruchomienie aplikacji napisanej w C za pomocą okna narzędzia Terminal w systemie macOS	52

1.11.	System operacyjny	54
1.11.1.	Windows — własnościowy system operacyjny	55
1.11.2.	Linux — otwarty system operacyjny	55
1.11.3.	Systemy firmy Apple — macOS dla komputerów Mac, iOS dla urządzeń mobilnych iPhone, iPad i iPod touch	55
1.11.4.	System Android firmy Google	56
1.12.	Internet i sieć WWW	56
1.12.1.	Internet — sieć sieci	57
1.12.2.	Sieć WWW — internet przyjazny użytkownikowi	57
1.12.3.	Usługa sieciowa	57
1.12.4.	AJAX	59
1.12.5.	Internet rzeczy	59
1.13.	Wybrana kluczowa terminologia związana z oprogramowaniem	59
1.14.	Na bieżąco z technologiami informatycznymi	62

Rozdział 2. Wprowadzenie do programowania w języku C67

2.1.	Wprowadzenie	68
2.2.	Prosty program w języku C — wyświetlenie wiersza tekstu	68
2.3.	Następny prosty program w języku C — dodawanie dwóch liczb całkowitych	72
2.4.	Koncepcje dotyczące pamięci	77
2.5.	Arytmetyka w języku C	78
2.6.	Podejmowanie decyzji — operatory równości i relacji	82
2.7.	Bezpieczne programowanie w języku C	86

Rozdział 3. Programowanie strukturalne w języku C99

3.1.	Wprowadzenie	100
3.2.	Algorytm	100
3.3.	Pseudokod	100
3.4.	Struktury kontrolne	101
3.5.	Polecenie wyboru if	102
3.6.	Polecenie wyboru if-else	103
3.7.	Polecenie iteracji while	107
3.8.	Studium przypadku tworzenia algorytmu 1 — iteracja kontrolowana przez licznik	108
3.9.	Studium przypadku tworzenia algorytmu 2 — iteracja kontrolowana przez wartownik	111
3.10.	Studium przypadku tworzenia algorytmu 3 — zagnieżdżone polecenia kontrolne	117
3.11.	Operatory przypisania	121
3.12.	Operatory inkrementacji i dekrementacji	121
3.13.	Bezpieczne programowanie w języku C	124

Rozdział 4. Struktury warunkowe programu w języku C143

4.1.	Wprowadzenie	144
4.2.	Podstawy iteracji	144
4.3.	Iteracja oparta na liczniku	144
4.4.	Konstrukcja for	146
4.5.	Konstrukcja — uwagi i obserwacje	149
4.6.	Przykłady użycia polecenia for	150
4.7.	Konstrukcja switch umożliwiająca wybór spośród wielu możliwości	153
4.8.	Konstrukcja do-while	159
4.9.	Polecenia break i continue	160

4.10.	Operatory logiczne	162
4.11.	Mylenie operatorów równości (==) i przypisania (=)	165
4.12.	Podsumowanie programowania strukturalnego	166
4.13.	Bezpieczne programowanie w języku C	171
Rozdział 5. Funkcje w języku C		187
5.1.	Wprowadzenie	188
5.2.	Modularyzacja programów w języku C	188
5.3.	Funkcje biblioteki matematycznej	189
5.4.	Funkcje	191
5.5.	Definicja funkcji	191
5.5.1.	Funkcja square()	191
5.5.2.	Funkcja maximum()	195
5.6.	Więcej o prototypie funkcji	196
5.7.	Stos wywołań funkcji i stos ramek	199
5.8.	Nagłówki	202
5.9.	Przekazywanie argumentów przez wartość i przez referencję	203
5.10.	Generowanie liczb losowych	204
5.11.	Przykład — gra hazardowa i wprowadzenie typu enum	208
5.12.	Klasy przechowywania	211
5.13.	Reguły dotyczące zasięgu	213
5.14.	Rekurencja	216
5.15.	Przykład użycia rekurencji — ciąg Fibonacciego	220
5.16.	Rekurencja kontra iteracja	223
5.17.	Bezpieczne programowanie w języku C	225
Rozdział 6. Tablice w języku C		245
6.1.	Wprowadzenie	246
6.2.	Tablica	246
6.3.	Definiowanie tablicy	247
6.4.	Przykłady tablic	248
6.4.1.	Definiowanie tablicy i stosowanie pętli do przypisania wartości elementom	248
6.4.2.	Inicjalizowanie tablicy w definicji za pomocą listy inicjalizacyjnej	249
6.4.3.	Określenie wielkości tablicy za pomocą stałej symbolicznej i inicjalizowanie elementów tablicy na podstawie wyniku obliczeń	250
6.4.4.	Sumowanie elementów tablicy	251
6.4.5.	Stosowanie tablicy do podsumowania wyników ankiety	252
6.4.6.	Wizualizacja wartości elementów tablicy za pomocą histogramu	254
6.4.7.	Rzut kością 60 000 000 razy i podsumowanie wyników za pomocą tablicy	255
6.5.	Stosowanie tablicy znaków do przechowywania ciągów tekstowych i operowania na nich	256
6.5.1.	Inicjalizacja tablicy znaków za pomocą ciągu tekstowego	256
6.5.2.	Inicjalizacja tablicy znaków za pomocą listy znaków	256
6.5.3.	Uzyskiwanie dostępu do znaków ciągu tekstowego	256
6.5.4.	Umieszczanie danych wejściowych w tablicy znaków	257
6.5.5.	Wyświetlanie tablicy znaków przedstawiającej ciąg tekstowy	257
6.5.6.	Przykład użycia tablicy znaków	257
6.6.	Statyczna i automatyczna tablica lokalna	258
6.7.	Przekazywanie funkcji argumentu w postaci tablicy	260
6.8.	Sortowanie tablicy	264

6.9.	Studium przypadku — obliczanie średniej, mediany i dominanty za pomocą tablic	266
6.10.	Wyszukiwanie elementów w tablicy	270
6.10.1.	Wyszukiwanie liniowe elementu w tablicy	271
6.10.2.	Wyszukiwanie binarne elementu w tablicy	272
6.11.	Tablica wielowymiarowa	275
6.11.1.	Prezentacja tablicy o dwóch indeksach	275
6.11.2.	Inicjalizacja tablicy dwuwymiarowej	276
6.11.3.	Definiowanie elementów w wierszu	278
6.11.4.	Sumowanie elementów tablicy dwuwymiarowej	278
6.11.5.	Operacje na tablicy dwuwymiarowej	278
6.12.	Tablica o zmiennej wielkości	281
6.13.	Bezpieczne programowanie w języku C	284

Rozdział 7. Wskaźniki w języku C305

7.1.	Wprowadzenie	306
7.2.	Definiowanie i inicjalizowanie zmiennej wskaźnika	306
7.3.	Operatory wskaźnika	307
7.4.	Przekazywanie argumentów do funkcji przez referencję	309
7.5.	Stosowanie kwalifikatora <code>const</code> ze wskaźnikiem	314
7.5.1.	Konwersja znaków ciągu tekstowego na wielkie za pomocą wskaźnika niebędącego stałą do danych niebędących stałą	315
7.5.2.	Wyświetlanie po jednym znaku ciągu tekstowego za pomocą wskaźnika niebędącego stałą do danych będących stałą	316
7.5.3.	Próba modyfikacji za pomocą wskaźnika będącego stałą do danych niebędących stałą	318
7.5.4.	Próba modyfikacji za pomocą wskaźnika będącego stałą do danych będących stałą	318
7.6.	Sortowanie bąbelkowe z użyciem przekazywania przez referencję	319
7.7.	Operator <code>sizeof</code>	322
7.8.	Wyrażenia i arytmetyka wskaźnika	325
7.8.1.	Dozwolone operatory dla arytmetyki wskaźnika	325
7.8.2.	Wskaźnik prowadzący do elementu tablicy	325
7.8.3.	Dodawanie liczby całkowitej do wskaźnika	325
7.8.4.	Odejmowanie liczby całkowitej od wskaźnika	326
7.8.5.	Inkrementacja i dekrementacja wskaźnika	326
7.8.6.	Odejmowanie wskaźników	327
7.8.7.	Przypisanie jednego wskaźnika innemu	327
7.8.8.	Wskaźnik do <code>void</code>	327
7.8.9.	Porównywanie wskaźników	327
7.9.	Związek między wskaźnikiem i tablicą	328
7.9.1.	Notacja wskaźnika i przesunięcia	328
7.9.2.	Notacja wskaźnika i indeksu	329
7.9.3.	Brak możliwości modyfikacji nazwy tablicy za pomocą arytmetyki wskaźnika	329
7.9.4.	Ustalanie indeksu i przesunięcia wskaźnika	329
7.9.5.	Kopiowanie ciągu tekstowego za pomocą tablicy i wskaźnika	330
7.10.	Tablica wskaźników	332
7.11.	Studium przypadku — symulacja tasowania i rozdawania kart	333
7.12.	Wskaźnik do funkcji	337
7.12.1.	Sortowanie w kolejności rosnącej lub malejącej	337
7.12.2.	Stosowanie wskaźnika do funkcji podczas tworzenia systemu opartego na menu	340
7.13.	Bezpieczne programowanie w języku C	342

Rozdział 8. Znaki i ciągi tekstowe w języku C	363
8.1. Wprowadzenie	364
8.2. Podstawy dotyczące znaków i ciągów tekstowych	364
8.3. Biblioteka obsługi znaków	366
8.3.1. Funkcje isdigit(), isalpha(), isalnum() i isxdigit()	367
8.3.2. Funkcje islower(), isupper(), tolower() i toupper()	368
8.3.3. Funkcje isspace(), iscntrl(), ispunct(), isprint() i isgraph()	370
8.4. Funkcje konwersji ciągu tekstowego	371
8.4.1. Funkcja strtod()	371
8.4.2. Funkcja strtol()	372
8.4.3. Funkcja strtoul()	373
8.5. Funkcje biblioteki standardowej wejścia-wyjścia	374
8.5.1. Funkcje fgetc() i putchar()	374
8.5.2. Funkcja getchar()	376
8.5.3. Funkcja sprintf()	376
8.5.4. Funkcja sscanf()	377
8.6. Funkcje biblioteki przeznaczonej do operacji na ciągach tekstowych	378
8.6.1. Funkcje strcpy() i strncpy()	378
8.6.2. Funkcje strcat() i strncat()	379
8.7. Funkcje porównania zdefiniowane w bibliotece przeznaczonej do obsługi ciągów tekstowych	380
8.8. Funkcje wyszukiwania zdefiniowane w bibliotece przeznaczonej do obsługi ciągów tekstowych	382
8.8.1. Funkcja strchr()	383
8.8.2. Funkcja strcspn()	383
8.8.3. Funkcja strpbrk()	384
8.8.4. Funkcja strrchr()	384
8.8.5. Funkcja strspn()	385
8.8.6. Funkcja strstr()	386
8.8.7. Funkcja strtok()	386
8.9. Funkcje dotyczące pamięci zdefiniowane w bibliotece przeznaczonej do obsługi ciągów tekstowych	387
8.9.1. Funkcja memcpy()	388
8.9.2. Funkcja memmove()	389
8.9.3. Funkcja memcmp()	390
8.9.4. Funkcja memchr()	390
8.9.5. Funkcja memset()	391
8.10. Pozostałe funkcje w bibliotece przeznaczonej do obsługi ciągów tekstowych	391
8.10.1. Funkcja strerror()	392
8.10.2. Funkcja strlen()	392
8.11. Bezpieczne programowanie w języku C	393
 Rozdział 9. Formatowanie danych wejściowych i wyjściowych w języku C	 407
9.1. Wprowadzenie	408
9.2. Strumienie	408
9.3. Formatowanie danych wyjściowych za pomocą funkcji printf()	408
9.4. Wyświetlanie liczb całkowitych	409
9.5. Liczby zmiennoprzecinkowe	410
9.5.1. Specyfikatory konwersji e, E i F	410
9.5.2. Specyfikatory konwersji g i G	411
9.5.3. Prezentacja specyfikatorów konwersji liczb zmiennoprzecinkowych	411

9.6.	Wyświetlanie ciągów tekstowych i znaków	412
9.7.	Inne specyfikatory konwersji	413
9.8.	Określanie szerokości pola i dokładności podczas wyświetlania danych	414
9.8.1.	Określanie szerokości pola dla wyświetlanej liczby całkowitej	414
9.8.2.	Określanie dokładności dla liczb całkowitych, zmiennoprzecinkowych i ciągów tekstowych ...	415
9.8.3.	Określanie szerokości pola i dokładności wartości	416
9.9.	Stosowanie opcji w ciągu tekstowym formatowania funkcji printf()	416
9.9.1.	Wyrównanie do prawej i lewej strony	416
9.9.2.	Wyświetlanie liczb dodatnich i ujemnych z opcją + i bez niej	417
9.9.3.	Stosowanie opcji w postaci spacji	418
9.9.4.	Stosowanie opcji #	418
9.9.5.	Stosowanie opcji 0	419
9.10.	Wyświetlanie literałów i sekwencje sterujące	419
9.11.	Pobieranie za pomocą funkcji scanf() sformatowanych danych wejściowych	420
9.11.1.	Składnia funkcji scanf()	420
9.11.2.	Specyfikatory konwersji funkcji scanf()	420
9.11.3.	Pobieranie liczb całkowitych za pomocą funkcji scanf()	420
9.11.4.	Pobieranie liczb zmiennoprzecinkowych za pomocą funkcji scanf()	422
9.11.5.	Pobieranie znaków i ciągów tekstowych za pomocą funkcji scanf()	423
9.11.6.	Stosowanie zbioru w funkcji scanf()	423
9.11.7.	Stosowanie szerokości pola podczas pracy z funkcją scanf()	424
9.11.8.	Pomijanie znaków w strumieniu danych wejściowych	425
9.12.	Bezpieczne programowanie w języku C	426

Rozdział 10. Struktury, unie, operacje na bitach i wyliczenia w języku C435

10.1.	Wprowadzenie	436
10.2.	Definicja struktury	436
10.2.1.	Struktura odwołująca się do samej siebie	437
10.2.2.	Definiowanie zmiennej typu struktury	437
10.2.3.	Nazwy tagów struktury	438
10.2.4.	Operacje możliwe do przeprowadzania na strukturze	438
10.3.	Inicjalizacja struktury	439
10.4.	Uzyskanie dostępu do elementu struktury za pomocą operatorów . i ->	439
10.5.	Stosowanie struktur wraz z funkcjami	441
10.6.	Definicja typedef	441
10.7.	Przykład — wysoko wydajna symulacja tasowania i rozdawania kart	442
10.8.	Unia	445
10.8.1.	Deklaracja unii	445
10.8.2.	Operacje możliwe do przeprowadzania na unii	445
10.8.3.	Inicjalizacja unii w deklaracji	446
10.8.4.	Przykład użycia unii	446
10.9.	Operatory bitowe	447
10.9.1.	Wyświetlenie za pomocą bitów liczby całkowitej bez znaku	448
10.9.2.	Uogólnienie funkcji displayBits() i zapewnienie jej przenośności	450
10.9.3.	Stosowanie operatorów bitowych i, lub, wyłączającego lub i dopełnienia	450
10.9.4.	Stosowanie operatorów bitowych przesunięcia w lewo i przesunięcia w prawo	453
10.9.5.	Operatory bitowe przypisania	454
10.10.	Pole bitowe	455
10.10.1.	Definiowanie pola bitowego	455
10.10.2.	Zastosowanie pola bitowego do przedstawienia figury i koloru karty	456
10.10.3.	Nienazwane pole bitowe	458

10.11. Stała wyliczenia	459
10.12. Anonimowe struktury i unie	460
10.13. Bezpieczne programowanie w języku C	460
Rozdział 11. Przetwarzanie plików w języku C	473
11.1. Wprowadzenie	474
11.2. Plik i strumień	474
11.3. Tworzenie pliku o dostępie sekwencyjnym	475
11.3.1. Wskaźnik do FILE	476
11.3.2. Stosowanie funkcji fopen() do otwierania pliku	476
11.3.3. Stosowanie funkcji feof() do sprawdzania pod kątem znacznika końca pliku	477
11.3.4. Stosowanie funkcji fprintf() do zapisywania danych w pliku	477
11.3.5. Stosowanie funkcji fclose() do zamykania pliku	477
11.3.6. Tryb otwarcia pliku	479
11.4. Odczytywanie danych z pliku o dostępie sekwencyjnym	480
11.4.1. Zerowanie wskaźnika położenia w pliku	481
11.4.2. Program pobierający informacje o saldach klientów	481
11.5. Plik o dostępie swobodnym	484
11.6. Tworzenie pliku o dostępie swobodnym	485
11.7. Losowy zapis danych w pliku o dostępie swobodnym	487
11.7.1. Stosowanie funkcji fseek() do umieszczania wskaźnika położenia w pliku	488
11.7.2. Sprawdzanie pod kątem błędów	489
11.8. Odczytywanie danych z pliku o dostępie swobodnym	489
11.9. Studium przypadku — program przetwarzający transakcje	490
11.10. Bezpieczne programowanie w języku C	496
Rozdział 12. Struktury danych w języku C	507
12.1. Wprowadzenie	508
12.2. Struktura odwołująca się do samej siebie	508
12.3. Dynamiczna alokacja pamięci	509
12.4. Lista jednokierunkowa	510
12.4.1. Funkcja insert()	516
12.4.2. Funkcja delete()	516
12.4.3. Funkcja printList()	518
12.5. Stos	518
12.5.1. Funkcja push()	522
12.5.2. Funkcja pop()	523
12.5.3. Zastosowanie stosu	523
12.6. Kolejka	524
12.6.1. Funkcja enqueue()	528
12.6.2. Funkcja dequeue()	529
12.7. Drzewo	529
12.7.1. Funkcja insertNode()	533
12.7.2. Funkcje nawigacji po drzewie: inOrder(), preOrder() i postOrder()	533
12.7.3. Eliminowanie duplikatów	534
12.7.4. Binarne drzewo wyszukiwania	534
12.7.5. Inne operacje przeprowadzane na drzewie binarnym	534
12.8. Bezpieczne programowanie w języku C	534

Rozdział 13. Preprocesor w języku C	547
13.1. Wprowadzenie	548
13.2. Dyrektywa preprocesora #include	548
13.3. Dyrektywa preprocesora #define — stałe symboliczne	548
13.4. Dyrektywa preprocesora #define — makra	549
13.4.1. Makro z jednym argumentem	550
13.4.2. Makro z dwoma argumentami	551
13.4.3. Znak kontynuowania makra	551
13.4.4. Dyrektywa preprocesora #undef	551
13.4.5. Makra i funkcje biblioteki standardowej	551
13.4.6. Nie umieszczaj w makrze wyrażenia powodującego efekty uboczne	551
13.5. Kompilacja warunkowa	552
13.5.1. Dyrektywa preprocesora #if...#endif	552
13.5.2. Komentowanie bloków kodu za pomocą #if...#endif	552
13.5.3. Warunkowa kompilacja kodu używanego podczas debugowania	552
13.6. Dyrektywy preprocesora #error i #pragma	553
13.7. Operatory # i ##	553
13.8. Numery wierszy	554
13.9. Predefiniowane stałe symboliczne	554
13.10. Asercje	554
13.11. Bezpieczne programowanie w języku C	555
Rozdział 14. Inne zagadnienia związane z językiem C	561
14.1. Wprowadzenie	562
14.2. Przekierowanie operacji wejścia-wyjścia	562
14.2.1. Przekierowanie wejścia za pomocą znaku <	562
14.2.2. Przekierowanie wejścia za pomocą znaku 	562
14.2.3. Przekierowanie wyjścia	563
14.3. Zmiennej długości lista argumentów	563
14.4. Stosowanie argumentów powłoki	565
14.5. Kompilowanie programu składającego się z wielu plików kodu źródłowego	566
14.5.1. Deklaracje extern dla zmiennych globalnych w innych plikach	566
14.5.2. Prototypy funkcji	567
14.5.3. Ograniczenie zasięgu za pomocą słowa kluczowego static	568
14.5.4. Plik Makefile	568
14.6. Zakończenie działania programu za pomocą exit() i atexit()	568
14.7. Sufiksy dla literałów liczb całkowitych i zmiennoprzecinkowych	569
14.8. Obsługa sygnałów	570
14.9. Dynamiczna alokacja pamięci — funkcje calloc() i realloc()	572
14.10. Bezwarunkowe odgałęzienie za pomocą goto	573
Rozdział 15. C++ jako lepsza wersja C — wprowadzenie do technologii obiektowej	579
15.1. Wprowadzenie	580
15.2. C++	580
15.3. Prosty program dodający dwie liczby	580
15.3.1. Program w C++ dodający dwie liczby	580
15.3.2. Nagłówek <iostream>	581
15.3.3. Funkcja main()	581
15.3.4. Deklaracje zmiennych	581
15.3.5. Obiekty standardowych strumieni wejścia i wyjścia	581

15.3.6.	Manipulator strumienia std::endl	582
15.3.7.	Wyjaśnienie prefiksu std:::	582
15.3.8.	Połączone strumienie wyjścia	582
15.3.9.	Polecenie return nie jest wymagane w funkcji main()	582
15.3.10.	Przeciążanie operatorów	583
15.4.	Biblioteka standardowa języka C++	583
15.5.	Pliki nagłówkowe	584
15.6.	Funkcja typu inline	585
15.7.	Słowa kluczowe języka C++	587
15.8.	Referencja i parametry przekazywane przez referencję	587
15.8.1.	Parametry referencyjne	588
15.8.2.	Przekazywanie argumentów przez wartość i przez referencję	589
15.8.3.	Referencja jako alias w funkcji	591
15.8.4.	Zwrot referencji przez funkcję	592
15.8.5.	Komunikat błędu do niezainicjalizowanej referencji	593
15.9.	Pusta lista parametrów	593
15.10.	Argumenty domyślne	593
15.11.	Jednoargumentowy operator ustalenia zasięgu	595
15.12.	Przeciążanie funkcji	596
15.13.	Szablony funkcji	599
15.13.1.	Definiowanie szablonu funkcji	599
15.13.2.	Stosowanie szablonu funkcji	600
15.14.	Wprowadzenie do technologii obiektowej i UML	602
15.14.1.	Podstawowe koncepcje technologii obiektowej	602
15.14.2.	Klasy, elementy składowe danych i funkcji	603
15.14.3.	Projekt i analiza zorientowane obiektowo	604
15.14.4.	Zunifikowany język modelowania	604
15.15.	Wprowadzenie do szablonu klasy vector biblioteki standardowej języka C++	605
15.15.1.	Problemy związane z opartą na wskaźnikach tablicą w stylu języka C	605
15.15.2.	Stosowanie szablonu klasy vector	605
15.15.3.	Obsługa wyjątków — przetwarzanie indeksu spoza zakresu	610
15.16.	Zakończenie rozdziału	611
Rozdział 16.	Wprowadzenie do klas, obiektów i ciągów tekstowych	619
16.1.	Wprowadzenie	620
16.2.	Definiowanie klasy z funkcją składową	620
16.3.	Definiowanie funkcji składowej z parametrem	623
16.4.	Dane składowe, funkcje składowe set i get	625
16.5.	Inicjalizacja obiektu za pomocą konstruktora	631
16.6.	Umieszczenie klasy w oddzielnym pliku, aby umożliwić jej wielokrotne użycie	634
16.7.	Oddzielenie interfejsu od implementacji	638
16.8.	Weryfikacja danych wejściowych za pomocą funkcji set	643
16.9.	Zakończenie rozdziału	647
Rozdział 17.	Klasy — zgłaszanie wyjątków	655
17.1.	Wprowadzenie	656
17.2.	Studium przypadku — klasa Time	656
17.3.	Zasięg klasy i dostęp do jej elementów składowych	663
17.4.	Funkcje dostępu i funkcje narzędziowe	664
17.5.	Studium przypadku — konstruktor z argumentami domyślnymi	664

17.6. Destruktory	670
17.7. Kiedy są wywoływane konstruktor i destruktory?	671
17.8. Studium przypadku — pułapka podczas zwrotu odwołania lub wskaźnika do prywatnych danych składowych	674
17.9. Domyślne przypisywanie danych składowych	677
17.10. Obiekty i funkcje składowe typu const	678
17.11. Kompozycja — obiekt jako element składowy klasy	681
17.12. Funkcje i klasy zaprzyjaźnione	686
17.13. Stosowanie wskaźnika this	688
17.14. Statyczne elementy składowe klasy	694
17.15. Zakończenie rozdziału	698

Rozdział 18. Przeciążanie operatorów i klasa string711

18.1. Wprowadzenie	712
18.2. Stosowanie przeciążonych operatorów klasy string biblioteki standardowej	712
18.3. Podstawy przeciążania operatorów	715
18.4. Przeciążanie operatorów dwuargumentowych	717
18.5. Przeciążanie dwuargumentowych operatorów wstawiania danych do strumienia i pobierania danych ze strumienia	717
18.6. Przeciążanie operatorów jednoargumentowych	721
18.7. Przeciążanie jednoargumentowych operatorów prefiks i postfiks ++ i --	722
18.8. Studium przypadku — klasa Date	723
18.9. Dynamiczne zarządzanie pamięcią	728
18.10. Studium przypadku — klasa Array	730
18.10.1. Stosowanie klasy Array	731
18.10.2. Definicja klasy Array	735
18.11. Operator w postaci funkcji składowej kontra operator w postaci funkcji nieskładowej	742
18.12. Konwersja między typami	743
18.13. Konstruktor typu explicit i operatory konwersji	744
18.14. Przeciążanie funkcji operator()	747
18.15. Zakończenie rozdziału	747

Rozdział 19. Programowanie zorientowane obiektowo — dziedziczenie759

19.1. Wprowadzenie	760
19.2. Klasa bazowa i klasa pochodna	760
19.3. Relacje między klasą bazową i klasą pochodną	763
19.3.1. Utworzenie i użycie klasy CommissionEmployee	763
19.3.2. Utworzenie klasy BasePlusCommissionEmployee bez wykorzystania dziedziczenia	767
19.3.3. Tworzenie hierarchii dziedziczenia CommissionEmployee- BasePlusCommissionEmployee	772
19.3.4. Hierarchia dziedziczenia CommissionEmployee-BasePlusCommissionEmployee wykorzystująca dane chronione	776
19.3.5. Hierarchia dziedziczenia CommissionEmployee-BasePlusCommissionEmployee wykorzystująca dane prywatne	779
19.4. Konstruktor i destruktory w klasie pochodnej	784
19.5. Dziedziczenie publiczne, chronione i prywatne	785
19.6. Stosowanie dziedziczenia w tworzeniu oprogramowania	786
19.7. Zakończenie rozdziału	787

Rozdział 20. Programowanie zorientowane obiektowo — polimorfizm	793
20.1. Wprowadzenie	794
20.2. Wprowadzenie do polimorfizmu — polimorficzna gra wideo	794
20.3. Relacje między obiektami w hierarchii dziedziczenia	795
20.3.1. Wywołanie funkcji klasy bazowej z poziomu obiektu klasy pochodnej	796
20.3.2. Skierowanie wskaźnika funkcji pochodnej na obiekt klasy bazowej	798
20.3.3. Wywołanie funkcji składowej klasy pochodnej za pomocą wskaźnika klasy bazowej	799
20.3.4. Funkcje wirtualne i destrukторы wirtualne	801
20.4. Typ pola i konstrukcja switch	808
20.5. Klasa abstrakcyjna i czysta funkcja wirtualna	808
20.6. Studium przypadku — system kadrowo-płacowy oparty na polimorfizmie	810
20.6.1. Tworzenie abstrakcyjnej klasy bazowej Employee	811
20.6.2. Tworzenie konkretnej klasy pochodnej SalariedEmployee	814
20.6.3. Tworzenie konkretnej klasy pochodnej CommissionEmployee	817
20.6.4. Tworzenie pośredniej konkretnej klasy pochodnej BasePlusCommissionEmployee	818
20.6.5. Przykład przetwarzania polimorficznego	820
20.7. (Opcjonalnie) Polimorfizm, funkcje wirtualne i wiązanie dynamiczne „pod maską”	823
20.8. Studium przypadku — system kadrowo-płacowy oparty na polimorfizmie, mechanizmie RTTI, rzutowaniu w dół, operatorach dynamic_cast i typeid oraz klasie type_info	827
20.9. Zakończenie rozdziału	830
Rozdział 21. Dokładniejsza analiza strumieni wejścia i wyjścia	837
21.1. Wprowadzenie	838
21.2. Strumienie	838
21.2.1. Strumienie klasyczne kontra strumienie standardowe	839
21.2.2. Nagłówki biblioteki iostream	839
21.2.3. Obiekty i klasy strumieni wejścia-wyjścia	840
21.3. Strumień wyjścia	842
21.3.1. Dane wyjściowe zmiennych typu char *	842
21.3.2. Wyświetlanie znaków za pomocą funkcji składowej put()	843
21.4. Strumień wejścia	843
21.4.1. Funkcje składowe get() i getline()	844
21.4.2. Funkcje składowe peek(), putback() i ignore() obiektu istream	846
21.4.3. Operacje wejścia-wyjścia zapewniające bezpieczeństwo typu	846
21.5. Niesformatowane operacje wejścia-wyjścia przeprowadzane za pomocą funkcji read(), write() i gcount()	847
21.6. Wprowadzenie do manipulatorów strumienia	848
21.6.1. Podstawa liczb całkowitych w strumieniu — manipulatory dec, oct, hex i setbase	848
21.6.2. Dokładność liczb zmiennoprzecinkowych	849
21.6.3. Szerokość pola — width() i setw()	850
21.6.4. Manipulatory strumienia wyjścia zdefiniowane przez użytkownika	851
21.7. Stany formatu strumienia i manipulatorów strumienia	852
21.7.1. Zera na końcu wartości i przecinek (manipulator showpoint)	853
21.7.2. Wyrównanie wartości (manipulatory left, right i internal)	854
21.7.3. Dopełnienie (funkcja fill() i manipulator setfill)	855
21.7.4. Wewnętrzna podstawa strumienia (manipulatory dec, oct, hex, showbase)	857
21.7.5. Liczby zmiennoprzecinkowe — notacja naukowa oraz z określoną liczbą cyfr po przecinku (manipulatory scientific i fixed)	857
21.7.6. Kontrolowanie wielkości liter (manipulator uppercase)	858

21.7.7. Stosowanie formatu boolowskiego (manipulator boolalpha)	859
21.7.8. Ustawianie i zerowanie stanu formatu za pomocą funkcji składowej flags()	860
21.8. Stany błędu strumienia	861
21.9. Powiązanie strumieni wyjścia i wejścia	863
21.10. Zakończenie rozdziału	863
Rozdział 22. Dokładniejsza analiza obsługi wyjątków	875
22.1. Wprowadzenie	876
22.2. Przykład — obsługa próby dzielenia przez zero	876
22.3. Ponowne zgłoszenie wyjątku	882
22.4. Rozwinięcie stosu	883
22.5. Kiedy używać obsługi wyjątków?	885
22.6. Konstruktor, destruktor i obsługa wyjątków	886
22.7. Wyjątki i dziedziczenie	887
22.8. Przetwarzanie niepowodzenia w wyniku działania operatora new	887
22.9. Klasa unique_ptr i dynamiczna alokacja pamięci	890
22.10. Hierarchia wyjątków biblioteki standardowej	893
22.11. Zakończenie rozdziału	894
Rozdział 23. Wprowadzenie do szablonów niestandardowych	901
23.1. Wprowadzenie	902
23.2. Szablony klas	902
23.3. Szablon funkcji przeznaczonej do przeprowadzania operacji na obiekcie specjalizacji szablonu klasy	906
23.4. Parametry pozbawione typu	908
23.5. Argumenty domyślne parametrów typu szablonu	908
23.6. Przeciążanie szablonu funkcji	909
23.7. Zakończenie rozdziału	909
Dodatek A. Pierwszeństwo operatorów w językach C i C++	913
Dodatek B. Kodowanie znaków ASCII	917
Dodatek C. Systemy liczbowe	919
C.1. Wprowadzenie	920
C.2. Skracanie liczb dwójkowych do ósemkowych i szesnastkowych	922
C.3. Konwersja liczb ósemkowych i szesnastkowych na dwójkowe	923
C.4. Konwersja liczb dwójkowych, ósemkowych i szesnastkowych na dziesiętne	924
C.5. Konwersja liczb dziesiętnych na dwójkowe, ósemkowe i szesnastkowe	925
C.6. Ujemne liczby dwójkowe — notacja dopełnienia do dwóch	926
Dodatek D. Sortowanie — informacje szczegółowe	931
D.1. Wprowadzenie	932
D.2. Notacja dużego O	932
D.3. Sortowanie przez wybieranie	933
D.4. Sortowanie przez wstawianie	936
D.5. Sortowanie przez scalanie	939

Dodatek E. Wielowątkowość oraz inne zagadnienia związane ze standardami C99 i C11	949
E.1. Wprowadzenie	950
E.2. Nowe nagłówki w C99	951
E.3. Wyznaczone metody inicjalizacyjne i złożone literały	951
E.4. Typ bool	954
E.5. Niejawne użycie typu int w deklaracji funkcji	955
E.6. Liczby zespolone	956
E.7. Usprawnienia w preprocesorze	957
E.8. Inne funkcje standardu C99	958
E.8.1. Minimalne ograniczenia zasobów kompilatora	959
E.8.2. Słowo kluczowe restrict	959
E.8.3. Niezawodne kopiowanie liczb całkowitych	959
E.8.4. Element składowy w postaci elastycznej tablicy	960
E.8.5. Złagodzone ograniczenia dotyczące inicjalizacji agregowanych elementów	960
E.8.6. Operacje matematyczne dla typów generycznych	960
E.8.7. Funkcja typu inline	961
E.8.8. Zakończenie działania funkcji bez wyrażenia	961
E.8.9. Predefiniowany identyfikator __func__	961
E.8.10. Makro va_copy	961
E.9. Nowe funkcje w standardzie C11	962
E.9.1. Nowe nagłówki C11	962
E.9.2. Obsługa wielowątkowości	962
E.9.3. Funkcja quick_exit()	969
E.9.4. Obsługa Unicode	969
E.9.5. Specyfikator funkcji _Noreturn	970
E.9.6. Wyrażenia typów generycznych	970
E.9.7. Dodatek L — możliwości w zakresie analizy i niezdefiniowane zachowanie	970
E.9.8. Kontrola wyrównania do granicy	971
E.9.9. Asercje statyczne	971
E.9.10. Typy zmiennooprzecinkowe	971
E.10. Zasoby dostępne w internecie	971
Dodatki w internecie	975
Dodatek F. Użycie debugera Visual Studio	FTP
Dodatek G. Użycie debugera GNU	FTP