

Spis treści

Wstęp	17
Krótka historia języka C	18
Nowości i zmiany w C23	19
Jak zorganizowana jest ta książka	21
Kody źródłowe	21
W ramach rozrywki	21
I. Podstawy	
1. Warsztat	25
1.1. Narzędzia i środowisko pracy	25
1.2. Uruchamianie przykładów	26
1.2.1. Pierwszy program	26
1.2.2. Eksperymenty	29
2. Kod źródłowy	30
2.1. Struktura kodu źródłowego	31
2.1.1. Przykładowa struktura	32
2.2. Ograniczenia translacji	34
II. Koncepcja języka	
3. Elementy leksykalne	37
3.1. Zestawy znaków	37
3.1.1. Znaki wielobajtowe i rozszerzone	38
3.1.2. Sekwencje trój- i dwuznakowe	39
3.2. Komentarze	40
3.3. Identyfikatory	41
3.3.1. Słowa kluczowe	41
3.3.2. Kategorie identyfikatorów	42
3.3.3. Zakres identyfikatorów	43
3.3.4. Łączenie identyfikatorów	43
3.3.5. Identyfikator <code>_func_</code>	44
3.4. Deklaracje	44
3.4.1. Składnia deklaracji	45
3.4.2. Deklaratory	45
3.4.3. Inicjatory	47
3.4.4. Diagnostyka: <code>static_assert</code>	49

4. Podstawowe typy danych	50
4.1. Klasyfikacja typów	50
4.2. Typ bool	52
4.3. Typ char	53
4.4. Typy całkowitoliczbowe	53
4.4.1. Standardowe typy całkowitoliczbowe	53
4.4.2. Typ <code>_BitInt(N)</code>	56
4.4.3. Rozszerzone typy całkowitoliczbowe	57
4.5. Typy zmiennoprzecinkowe	58
4.5.1. Standardowe typy zmiennoprzecinkowe	58
4.5.2. Zmiennoprzecinkowe typy dziesiętne	60
4.5.3. Zmiennoprzecinkowe liczby zespolone	61
4.6. Typ void	62
5. Typy wyliczeniowe i typy pochodne	64
5.1. Typy wyliczeniowe	64
5.2. Tablice	67
5.2.1. Tablice o stałej długości	68
5.2.2. Tablice o zmiennej długości	71
5.2.3. Tablice wielowymiarowe	72
5.2.4. Łańcuchy znaków	73
5.3. Wskaźniki	75
5.3.1. Arytmetyka wskaźników	78
5.3.2. Tablice a wskaźniki	80
5.3.3. Wskaźniki do funkcji	80
5.4. Struktury, unie i pola bitowe	81
5.4.1. Struktury	81
5.4.2. Unie	87
5.4.3. Pola bitowe	88
6. Jeszcze o typach	91
6.1. Deklaracja <code>typedef</code>	91
6.2. Czas trwania obiektów	92
6.2.1. Klasy pamięci	93
6.3. Specyfikatory typów	98
6.4. Kwalifikatory typów	99
6.4.1. Kwalifikator <code>const</code>	100
6.4.2. Kwalifikator <code>volatile</code>	102
6.4.3. Kwalifikator <code>restrict</code>	102
6.4.4. Kwalifikator <code>_Atomic</code>	104
6.5. Specyfikator wyrównania	105
6.5.1. Specyfikator <code>alignas</code>	106
6.6. Specyfikatory atrybutów	107
6.6.1. Atrybuty standardowe	108
6.6.2. Testowanie atrybutów	112
6.7. Typy niekompletne	113
6.8. Typy zgodne	114

6.9.	Konwersja typów	115
6.9.1.	Konwersje arytmetyczne	116
6.9.2.	Konwersje typów w przypisaniach i wskaźnikach	119
7.	Stałe	121
7.1.	Stałe całkowitoliczbowe	121
7.1.1.	Rozszerzone typy całkowitoliczbowe	123
7.1.2.	Separator cyfr	123
7.2.	Stałe zmiennoprzecinkowe	124
7.2.1.	Ułamki dziesiętne	124
7.2.2.	Szesnastkowe stałe zmiennoprzecinkowe	125
7.3.	Stałe wyliczeniowe	125
7.4.	Stałe znakowe	126
7.4.1.	Znaki sterujące	127
7.4.2.	Uniwersalna nazwa znaku	128
7.5.	Literały łańcuchowe	129
7.6.	Stałe predefiniowane	129
8.	Wyrażenia i operatory	131
8.1.	Przetwarzanie wyrażeń	131
8.1.1.	Klasyfikatory wyrażeń	131
8.1.2.	Wybór ogólny	132
8.2.	Operatory	134
8.2.1.	Operatory arytmetyczne	136
8.2.2.	Operatory przypisania	137
8.2.3.	Operatory relacji	138
8.2.4.	Operatory logiczne	139
8.2.5.	Operatory bitowe	140
8.2.6.	Alternatywna notacja operatorów logicznych i bitowych	142
8.2.7.	Operator warunkowy <code>?:</code>	142
8.2.8.	Operator sekwencji <code>,</code>	144
8.2.9.	Operatory dostępu do pamięci	144
8.2.10.	Operator wywołania funkcji <code>()</code>	146
8.2.11.	Operator rzutowania <code>(typ)</code>	146
8.2.12.	Literały złożone <code>(typ){}</code>	147
8.2.13.	Operator <code>sizeof</code>	148
8.2.14.	Operator <code>alignof</code>	148
8.2.15.	Operatory typowania <code>typeof</code> i <code>typeof_unqual</code>	150
9.	Instrukcje	152
9.1.	Instrukcja prosta	152
9.2.	Bloki	152
9.3.	Instrukcje sterujące	153
9.3.1.	Instrukcje warunkowe	154
9.3.2.	Pętle	156
9.3.3.	Skoki bezwarunkowe	158

10. Funkcje	161
10.1. Funkcja <code>main</code>	161
10.1.1. Uruchamianie programu	161
10.1.2. Zakończenie programu	162
10.2. Prototypy funkcji	163
10.3. Definicje funkcji	164
10.3.1. Definicje w starym stylu	165
10.3.2. Funkcje anonimowe	166
10.4. Specyfikatory funkcji	166
10.4.1. Specyfikator <code>inline</code>	166
10.4.2. Specyfikator <code>_Noreturn</code>	167
10.5. Wywołanie funkcji, operator <code>()</code>	168
11. Dyrektywy preprocesora	169
11.1. Dołączanie plików	169
11.1.1. Dyrektywa <code>#include</code>	169
11.1.2. Makro <code>_has_include</code>	170
11.1.3. Dyrektywa <code>#embed</code>	171
11.1.4. Makro <code>_has_embed</code>	174
11.2. Makrodefinicje	175
11.2.1. Dyrektywa <code>#define</code>	175
11.2.2. Dyrektywa <code>#undef</code>	176
11.2.3. Operator <code>#</code>	177
11.2.4. Operator <code>##</code>	177
11.2.5. Zmienna liczba argumentów makra	178
11.3. Kompilacja warunkowa	179
11.3.1. <code>#if</code> , <code>#elif</code> , <code>#else</code> , <code>#endif</code>	179
11.3.2. Operator <code>defined</code>	180
11.3.3. <code>#ifdef</code> , <code>#elifdef</code> , <code>#ifndef</code> , <code>#elifndef</code>	181
11.4. Inne dyrektywy	182
11.4.1. Dyrektywa <code>#</code>	182
11.4.2. Dyrektywa <code>#line</code>	182
11.4.3. Dyrektywy <code>#error</code> i <code>#warning</code>	183
11.5. Predefiniowane standardowe makra	184
11.5.1. Makra wymagane przez C23	184
11.5.2. Makra opcjonalne	185
11.6. Dyrektywa <code>#pragma</code>	186
11.6.1. Operator <code>_Pragma</code>	187
 III. Biblioteka standardowa	
12. Wprowadzenie	191
13. Diagnostyka <code><assert.h></code>	193
13.1. Makra <code><assert.h></code>	193
13.1.1. Makro <code>NDEBUG</code>	193
13.1.2. Makro <code>assert</code>	193
13.2. Deklaracja <code>static_assert</code>	194

14. Arytmetyka liczb zespolonych <complex.h>	196
14.1. Konstrukcja liczb zespolonych i urojonych	197
14.1.1. Liczby zespolone	197
14.1.2. Liczby urojone	198
14.1.3. Konstrukcja liczb	199
14.2. Funkcje manipulujące liczbami zespolonymi	200
14.2.1. Operacje podstawowe	200
14.2.2. Funkcje trygonometryczne	201
14.2.3. Funkcje hiperboliczne	201
14.2.4. #pragma CX_LIMITED_RANGE	203
14.3. Dalszy rozwój <complex.h>	204
15. Klasyfikacja znaków <ctype.h>	205
15.1. Funkcje klasyfikujące	205
15.2. Funkcje konwersji znaków	206
15.3. Dalszy rozwój <ctype.h>	207
16. Obsługa błędów <errno.h>	208
16.1. Typy i makra	208
16.1.1. Typ errno_t	208
16.1.2. Makro errno	209
16.2. Dalszy rozwój <errno.h>	210
17. Środowisko zmiennoprzecinkowe <fenv.h>	211
17.1. Dyrektywy #pragma	213
17.1.1. #pragma FENV_ACCESS	213
17.1.2. #pragma FENV_ROUND	213
17.1.3. #pragma FENV_DEC_ROUND	214
17.2. Obsługa wyjątków zmiennoprzecinkowych	214
17.2.1. Makra	214
17.2.2. Funkcje obsługi wyjątków	214
17.3. Zaokrąglania	219
17.3.1. Makra kierunku zaokrąglania	219
17.3.2. Funkcje obsługi zaokrągleń	219
17.4. Środowisko zmiennoprzecinkowe	222
17.4.1. Funkcja fegetenv	222
17.4.2. Funkcja fesetenv	222
17.4.3. Funkcja feholdexcept	224
17.4.4. Funkcja feupdateenv	224
17.5. Dalszy rozwój <fenv.h>	225
18. Charakterystyka typów zmiennoprzecinkowych <float.h>	226
18.1. Dalszy rozwój <float.h>	229

19. Konwersja formatu typów całkowitoliczbowych <inttypes.h>	230
19.1. Funkcje operujące na liczbach typu <code>intmax_t</code>	231
19.1.1. Funkcja <code>imaxabs</code>	231
19.1.2. Funkcja <code>imaxdiv</code>	231
19.1.3. Funkcje <code>strtoimax</code> i <code>strtoumax</code>	232
19.1.4. Funkcje <code>wcstoimax</code> i <code>wcstoumax</code>	232
19.2. Makra specyfikatorów formatu	232
19.3. Dalszy rozwój <inttypes.h>	233
20. Alternatywna notacja operatorów <iso646.h>	234
21. Charakterystyka typów całkowitoliczbowych <limits.h>	235
22. Ustawienia regionalne <locale.h>	238
22.1. Wprowadzenie	239
22.2. Ustawienia regionalne	239
22.2.1. Funkcja <code>setlocale</code>	239
22.2.2. Funkcja <code>localeconv</code>	241
22.3. Dalszy rozwój <locale.h>	243
23. Funkcje matematyczne <math.h>	244
23.1. Klasyfikacja i porównywanie liczb	249
23.2. Funkcje typów zmiennoprzecinkowych	251
23.2.1. Funkcje trygonometryczne i hiperboliczne	251
23.2.2. Funkcje wykładnicze i logarytmiczne	253
23.2.3. Pierwiastkowanie, potęgowanie, wartość bezwzględna	254
23.2.4. Funkcje błędu i funkcje gamma	254
23.2.5. Najbliższa wartość całkowita i zaokrąglenia	255
23.2.6. Operacja modulo	256
23.2.7. Manipulowanie wartościami	256
23.2.8. Różnica pozytywna, minimum i maksimum	257
23.2.9. Podstawowe operacje optymalizowane	258
23.3. Dodatkowe funkcje typów zmiennoprzecinkowych dziesiętnych	260
23.3.1. Funkcje kwantowe i kwantowe wykładnicze	260
23.3.2. Funkcje kodowania dziesiętnego	260
23.4. Rozszerzenia <math.h>	261
23.5. Dalszy rozwój <math.h>	262
24. Skoki odległe <setjmp.h>	263
24.1. Skoki odległe	263
24.1.1. Funkcja <code>setjmp</code>	263
24.1.2. Funkcja <code>longjmp</code>	264
24.1.3. Typ <code>jmp_buf</code>	265
25. Obsługa sygnałów <signal.h>	266
25.1. Wprowadzenie	266
25.2. Obsługa sygnałów	267
25.2.1. Funkcja <code>raise</code>	267
25.2.2. Funkcja <code>signal</code>	268

25.2.3. Typ <code>sig_atomic_t</code>	270
25.3. Dalszy rozwój <code><signal.h></code>	270
26. Wyrównywanie pamięci <code><stdalign.h></code>	271
27. Zmienna ilość argumentów funkcji <code><stdarg.h></code>	272
27.1. Funkcje o zmiennej ilości argumentów	272
27.1.1. Makro <code>va_start</code>	273
27.1.2. Makro <code>va_arg</code>	273
27.1.3. Makro <code>va_end</code>	273
27.1.4. Makro <code>va_copy</code>	274
28. Typy atomowe <code><stdatomic.h></code>	275
28.1. Wolność od blokady	277
28.1.1. Makra z rodziny <code>ATOMIC_typ_LOCK_FREE</code>	277
28.1.2. Funkcja <code>atomic_is_lock_free</code>	278
28.2. Typy atomowe i operacje na nich	278
28.2.1. Atomowe typy całkowitoliczbowe	278
28.2.2. Operacje atomowe	279
28.3. Inicjalizacja obiektów atomowych <code>atomic_init</code>	281
28.4. Szeregowanie pamięci	282
28.4.1. Typ <code>memory_order</code>	282
28.4.2. Makro <code>kill_dependency</code>	283
28.4.3. Bariery pamięci	283
28.5. Atomowe flagi i operacje na nich	284
28.5.1. Typ <code>atomic_flag</code>	284
28.5.2. Makro <code>ATOMIC_FLAG_INIT</code>	284
28.5.3. Funkcja <code>atomic_flag_test_and_set</code>	284
28.5.4. Funkcja <code>atomic_flag_clear</code>	284
28.6. Dalszy rozwój <code><stdatomic.h></code>	285
29. Narzędzia bitowe i bajtowe <code><stdbit.h></code>	286
29.1. Kolejność bajtów	287
29.2. Funkcje	287
29.2.1. Funkcje zliczające bity	288
29.2.2. Wyliczanie liczb binarnych	290
29.3. Dalszy rozwój <code><stdbit.h></code>	290
30. Wartości i typy logiczne <code><stdbool.h></code>	291
31. Kontrolowana arytmetyka liczb całkowitych <code><stdckdint.h></code>	292
31.1. Makra z rodziny <code>ckd_</code>	292
31.2. Dalszy rozwój <code><stdckdint.h></code>	293
32. Definicje wspólne <code><stddef.h></code>	294
32.1. Typy	294
32.1.1. Typ <code>max_align_t</code>	294
32.1.2. Typ <code>nullptr_t</code>	295
32.1.3. Typ <code>ptrdiff_t</code>	295
32.1.4. Typ <code>size_t</code>	296

32.1.5. Typ <code>rsizet</code>	296
32.1.6. Typ <code>wchar_t</code>	296
32.2. Makra	298
32.2.1. Makro <code>NULL</code>	298
32.2.2. Makro <code>offsetof</code>	298
32.2.3. Makro <code>unreachable</code>	299
33. Typy całkowitoliczbowe <code><stdint.h></code>	301
33.1. Dalszy rozwój <code><stdint.h></code>	304
34. Standardowe wejście i wyjście <code><stdio.h></code>	305
34.1. Wprowadzenie	308
34.2. Funkcje obsługi błędów wejścia/wyjścia	309
34.2.1. Funkcja <code>clearerr</code>	309
34.2.2. Funkcja <code>feof</code>	310
34.2.3. Funkcja <code>ferror</code>	310
34.2.4. Funkcja <code>perror</code>	311
34.3. Operacja na plikach	311
34.3.1. Funkcja <code>remove</code>	311
34.3.2. Funkcja <code>rename</code>	311
34.3.3. Funkcje <code>tmpfile</code> , <code>tmpfile_s</code>	312
34.3.4. Funkcje <code>tmpnam</code> , <code>tmpnam_s</code>	313
34.4. Funkcje dostępu do plików	314
34.4.1. Funkcja <code>fclose</code>	314
34.4.2. Funkcja <code>fflush</code>	315
34.4.3. Funkcje <code>fopen</code> , <code>fopen_s</code>	316
34.4.4. Funkcje <code>freopen</code> , <code>freopen_s</code>	317
34.4.5. Funkcja <code>setbuf</code>	318
34.4.6. Funkcja <code>setvbuf</code>	319
34.5. Funkcje formatowanego wejścia/wyjścia	319
34.5.1. Formatowane wyjście: rodzina funkcji <code>fprintf</code>	319
34.5.2. Formatowane wejście – rodzina funkcji <code>fscanf</code>	328
34.6. Znakowe wejście/wyjście	332
34.6.1. Funkcje <code>fgetc</code> , <code>getc</code> i <code>getchar</code>	332
34.6.2. Funkcje <code>fputc</code> , <code>putc</code> i <code>putchar</code>	333
34.6.3. Funkcja <code>fgets</code>	333
34.6.4. Funkcja <code>gets_s</code>	334
34.6.5. Funkcje <code>fputs</code> i <code>puts</code>	334
34.6.6. Funkcja <code>ungetc</code>	334
34.7. Blokowe funkcje wejścia/wyjścia	335
34.7.1. Funkcja <code>fread</code>	335
34.7.2. Funkcja <code>fwrite</code>	336
34.8. Funkcje pozycjonowania plików	336
34.8.1. Funkcja <code>fgetpos</code>	336
34.8.2. Funkcja <code>fseek</code>	337
34.8.3. Funkcja <code>fsetpos</code>	337
34.8.4. Funkcja <code>ftell</code>	337

34.8.5. Funkcja rewind	338
34.9. Dalszy rozwój <stdio.h>	338
35. Narzędzia ogólne <stdlib.h>	339
35.1. Konwersja liczb i funkcje arytmetyczne	342
35.1.1. Konwersja numeryczna	342
35.1.2. Konwersja liczb zmiennoprzecinkowych dziesiętnych	346
35.1.3. Generowanie liczb pseudolosowych	347
35.1.4. Funkcje arytmetyczne liczb całkowitych	348
35.2. Dynamiczne zarządzanie pamięcią	349
35.2.1. Przydział pamięci	349
35.2.2. Zwalnianie pamięci	351
35.2.3. Wyrównanie pamięci	352
35.3. Komunikacja z systemem operacyjnym	352
35.3.1. Zakończenie programu	352
35.3.2. Komunikacja ze środowiskiem	356
35.4. Narzędzia do wyszukiwania i sortowania	358
35.4.1. Funkcje qsort, qsort_s	358
35.4.2. Funkcje bsearch, bsearch_s	359
35.5. Funkcje konwersji znaków oraz łańcuchów wielobajtowych i rozszerzonych	361
35.5.1. Konwersja znaków wielobajtowych i rozszerzonych	361
35.5.2. Konwersja łańcuchów wielobajtowych i rozszerzonych	363
35.6. Obsługa błędów ograniczeń zakresu	365
35.6.1. Typ constraint_handler_t	365
35.6.2. Funkcja set_constraint_handler_s	365
35.6.3. Funkcja abort_handler_s	366
35.7. Dalszy rozwój <stdlib.h>	367
36. Funkcje bez powrotu <stdnoreturn.h>	368
37. Obsługa łańcuchów <string.h>	369
37.1. Kopiowanie	371
37.1.1. Funkcje strcpy, strcpy_s	371
37.1.2. Funkcje strncpy, strncpy_s	372
37.1.3. Funkcja strdup	373
37.1.4. Funkcja strndup	374
37.2. Łączenie	374
37.2.1. Funkcje strcat, strcat_s	374
37.2.2. Funkcje strncat, strncat_s	376
37.3. Porównywanie	376
37.3.1. Funkcja strcmp	376
37.3.2. Funkcja strncmp	377
37.3.3. Funkcja strcoll	377
37.3.4. Funkcja strxfrm	378
37.4. Wyszukiwanie	379
37.4.1. Funkcja strchr	379
37.4.2. Funkcja strcspn	379
37.4.3. Funkcja strpbrk	380

37.4.4. Funkcja <code>strrchr</code>	380
37.4.5. Funkcja <code>strspn</code>	380
37.4.6. Funkcja <code>strstr</code>	380
37.4.7. Funkcje <code>strtok</code> , <code>strtok_s</code>	381
37.5. Pozostałe funkcje	382
37.5.1. Funkcje <code>strerror</code> , <code>strerror_s</code>	382
37.5.2. Funkcja <code>strerrorlen_s</code>	383
37.5.3. Funkcje <code>strlen</code> , <code>strlen_s</code>	384
37.6. Obsługa bloków pamięci	384
37.6.1. Funkcja <code>memchr</code>	384
37.6.2. Funkcja <code>memcmp</code>	385
37.6.3. Funkcje <code>memcpy</code> , <code>memcpy_s</code>	385
37.6.4. Funkcja <code>memccpy</code>	386
37.6.5. Funkcje <code>memmove</code> , <code>memmove_s</code>	386
37.6.6. Funkcje <code>memset</code> , <code>memset_s</code>	386
37.7. Dalszy rozwój <code><string.h></code>	387
38. Funkcje matematyczne niezależne od typu <code><tgmath.h></code>	388
38.1. Makra wspólne dla typów rzeczywistych i zespolonych	388
38.2. Makra typów rzeczywistych	390
38.3. Makra typów zespolonych	393
38.4. Makra dziesiętnych typów zmiennoprzecinkowych	393
39. Wątki <code><threads.h></code>	394
39.1. Wątki	396
39.1.1. Obsługa wątków	396
39.1.2. Jednokrotne wywołanie funkcji	401
39.2. Dostęp do zasobów współdzielonych	402
39.2.1. Muteksy	402
39.2.2. Zmienne warunkowe	406
39.2.3. Pamięć lokalna wątku	410
39.3. Dalszy rozwój <code><threads.h></code>	413
40. Czas <code><time.h></code>	414
40.1. Typy i makra	416
40.1.1. Typ <code>clock_t</code>	416
40.1.2. Typ <code>struct timespec</code>	417
40.1.3. Typ <code>struct tm</code>	417
40.1.4. Typ <code>time_t</code>	418
40.1.5. Makro <code>CLOCKS_PER_SEC</code>	418
40.1.6. Makra z rodziny <code>TIME_</code>	419
40.2. Funkcje reprezentacji czasu	419
40.2.1. Funkcja <code>clock</code>	419
40.2.2. Funkcja <code>difftime</code>	419
40.2.3. Funkcja <code>time</code>	420
40.2.4. Funkcja <code>timespec_get</code>	421
40.2.5. Funkcja <code>timespec_getres</code>	421

40.3. Funkcje konwersji czasu	421
40.3.1. Funkcje asctime, asctime_r, asctime_s	421
40.3.2. Funkcje ctime, ctime_r, ctime_s	423
40.3.3. Funkcje gmtime, gmtime_r, gmtime_s	424
40.3.4. Funkcje localtime, localtime_r, localtime_s	424
40.3.5. Funkcje mktime, timegm	424
40.3.6. Funkcja strftime	425
40.4. Dalszy rozwój <time.h>	429
41. Obsługa znaków unicode <uchar.h>	430
41.1. Konwersja znaków UTF	431
41.1.1. Funkcje c8rtomb, c16rtomb, c32rtomb	431
41.1.2. Funkcje mbrtoc8, mbrtoc16, mbrtoc32	433
42. Obsługa znaków wielobajtowych i rozszerzonych <wchar.h>	435
42.1. Funkcje formatowanego wejścia/wyjścia	438
42.1.1. Stała WEOF	438
42.1.2. Formatowane wyjście – rodzina funkcji fwprintf	439
42.1.3. Formatowane wejście – rodzina funkcji fwscanf	441
42.2. Znakowe wejście/wyjście	442
42.2.1. Znakowe funkcje wyjścia	442
42.2.2. Znakowe funkcje wejścia	443
42.2.3. Funkcje dodatkowe	443
42.3. Manipulowanie łańcuchami	444
42.3.1. Konwersja numeryczna	444
42.3.2. Kopiowanie – wcsncpy, wcsncpy_s, wcsncpy, wcsncpy_s	445
42.3.3. Konkatenacja – wcscat, wcscat_s, wcsncat, wcsncat_s	446
42.3.4. Porównywanie – wcscmp, wcsncmp, wscoll, wcsxfrm	446
42.3.5. Przeszukiwanie	446
42.3.6. Pozostałe funkcje	447
42.3.7. Funkcje obsługi bloków pamięci	447
42.4. Konwersja znaków i łańcuchów	448
42.4.1. Konwersja znaków btowc, wctob	448
42.4.2. Konwersja znaków i łańcuchów wielobajtowych	449
42.5. Dalszy rozwój <wchar.h>	455
43. Klasyfikacja znaków rozszerzonych <wctype.h>	456
43.1. Funkcje klasyfikacji znaków	457
43.1.1. Funkcja iswctype	458
43.1.2. Funkcja wctype	459
43.2. Funkcje konwersji znaków	459
43.2.1. Funkcje tolower i toupper	459
43.2.2. Funkcja towctrans	459
43.2.3. Funkcja wctrans	460
43.3. Dalszy rozwój <wctype.h>	460
Literatura	461
Indeks	463