

# **Programowanie w C : sprytne podejście do trudnych zagadnień, których wołałbyś unikać (takich jak język C) / Zed A. Shaw. – Gliwice, 2016**

## Spis treści

<b>Podziękowania</b>	<b>12</b>
<b>Ta książka tak naprawdę nie jest o języku C</b>	<b>13</b>
Niezdefiniowane zachowania	14
C to język zarazem świetny i paskudny	15
Czego się nauczysz?	16
Jak czytać tę książkę?	16
Wideo	17
Podstawowe umiejętności	18
Czytanie i pisanie	18
Zwracanie uwagi na szczegóły	18
Wychwytywanie różnic	19
Planowanie i debugowanie	19
<b>Przygotowania</b>	<b>20</b>
Linux	20
OS X	20
Windows	21
Edytor tekstu	21
Nie używaj IDE	22
<b>Ćwiczenie 1. Odkurzenie kompilatora</b>	<b>24</b>
Omówienie kodu w pliku	24
Co powinieneś zobaczyć?	25
Jak to zepsuć?	26
Zadania dodatkowe	26
<b>Ćwiczenie 2. Użycie pliku Makefile podczas kompilacji</b>	<b>28</b>
Użycie narzędzia make	28
Co powinieneś zobaczyć?	29
Jak to zepsuć?	30
Zadania dodatkowe	30
<b>Ćwiczenie 3. Sformatowane dane wyjściowe</b>	<b>32</b>
Co powinieneś zobaczyć?	33
Zewnętrzne badania	33
Jak to zepsuć?	33
Zadania dodatkowe	34

<b>Ćwiczenie 4. Użycie debugera</b>	<b>36</b>
Sztuczki z GDB	36
Krótki przewodnik po GDB	36
Krótki przewodnik po LLDB	37
<b>Ćwiczenie 5. Nauka na pamięć operatorów w C</b>	<b>40</b>
Jak uczyć się na pamięć?	40
Listy operatorów	41
<b>Ćwiczenie 6. Nauka na pamięć składni C</b>	<b>46</b>
Słowa kluczowe	46
Składnia struktur	47
Słowo zachęty	50
Słowo ostrzeżenia	51
<b>Ćwiczenie 7. Zmienne i typy</b>	<b>52</b>
Co powinieneś zobaczyć?	53
Jak to zepsuć?	54
Zadania dodatkowe	54
<b>Ćwiczenie 8. Konstrukcje if, else-if i else</b>	<b>56</b>
Co powinieneś zobaczyć?	57
Jak to zepsuć?	57
Zadania dodatkowe	58
<b>Ćwiczenie 9. Pętla while i wyrażenia boolowskie</b>	<b>60</b>
Co powinieneś zobaczyć?	60
Jak to zepsuć?	61
Zadania dodatkowe	61
<b>Ćwiczenie 10. Konstrukcja switch</b>	<b>62</b>
Co powinieneś zobaczyć?	64
Jak to zepsuć?	65
Zadania dodatkowe	65
<b>Ćwiczenie 11. Tablice i ciągi tekstowe</b>	<b>66</b>
Co powinieneś zobaczyć?	67
Jak to zepsuć?	68
Zadania dodatkowe	69
<b>Ćwiczenie 12. Wielkość i tablice</b>	<b>70</b>
Co powinieneś zobaczyć?	71
Jak to zepsuć?	72
Zadania dodatkowe	73
<b>Ćwiczenie 13. Pętla for i tablica ciągów tekstowych</b>	<b>74</b>

Co powinieneś zobaczyć?	75
Zrozumienie tablicy ciągów tekstowych	76
Jak to zepsuć?	76
Zadania dodatkowe	77
<b>Ćwiczenie 14. Tworzenie i użycie funkcji</b>	<b>78</b>
Co powinieneś zobaczyć?	79
Jak to zepsuć?	80
Zadania dodatkowe	80
<b>Ćwiczenie 15. Wskaźniki, przerażające wskaźniki</b>	<b>82</b>
Co powinieneś zobaczyć?	84
Poznajemy wskaźniki	85
Praktyczne użycie wskaźników	86
Leksykon wskaźnika	87
Wskaźniki nie są tablicami	87
Jak to zepsuć?	87
Zadania dodatkowe	88
<b>Ćwiczenie 16. Struktury i prowadzące do nich wskaźniki</b>	<b>90</b>
Co powinieneś zobaczyć?	93
Poznajemy struktury	94
Jak to zepsuć?	94
Zadania dodatkowe	95
<b>Ćwiczenie 17. Alokacja pamięci stosu i sterty</b>	<b>96</b>
Co powinieneś zobaczyć?	102
Alokacja stosu kontra sterty	102
Jak to zepsuć?	103
Zadania dodatkowe	104
<b>Ćwiczenie 18. Wskaźniki do funkcji</b>	<b>106</b>
Co powinieneś zobaczyć?	110
Jak to zepsuć?	110
Zadania dodatkowe	111
<b>Ćwiczenie 19. Opracowane przez Zeda wspaniałe makra debugowania</b>	<b>112</b>
Problem obsługi błędów w C	112
Makra debugowania	113
Użycie dbg.h	115
Co powinieneś zobaczyć?	118
W jaki sposób CPP obsługuje makra?	118
Zadania dodatkowe	120
<b>Ćwiczenie 20. Zaawansowane techniki debugowania</b>	<b>122</b>

Użycie makra debug() kontra GDB	122
Strategia debugowania	124
Zadania dodatkowe	125
<b>Ćwiczenie 21. Zaawansowane typy danych i kontrola przepływu</b>	<b>126</b>
Dostępne typy danych	126
Modyfikatory typu	126
Kwalifikatory typów	127
Konwersja typu	127
Wielkość typu	128
Dostępne operatory	129
Operatory matematyczne	130
Operatory danych	130
Operatory logiczne	131
Operatory bitowe	131
Operatory boolowskie	131
Operatory przypisania	131
Dostępne struktury kontroli	132
Zadania dodatkowe	132
<b>Ćwiczenie 22. Stos, zakres i elementy globalne</b>	<b>134</b>
Pliki ex22.h i ex22.c	134
Plik ex22_main.c	136
Co powinieneś zobaczyć?	138
Zakres, stos i błędy	139
Jak to zepsuć?	140
Zadania dodatkowe	141
<b>Ćwiczenie 23. Poznaj mechanizm Duffa</b>	<b>142</b>
Co powinieneś zobaczyć?	145
Rozwiązanie łamigłówek	145
Dlaczego w ogóle mam się tak męczyć?	146
Zadania dodatkowe	146
<b>Ćwiczenie 24. Dane wejściowe, dane wyjściowe i pliki</b>	<b>148</b>
Co powinieneś zobaczyć?	150
Jak to zepsuć?	151
Funkcje wejścia-wyjścia	151
Zadania dodatkowe	152
<b>Ćwiczenie 25. Funkcje o zmiennej liczbie argumentów</b>	<b>154</b>
Co powinieneś zobaczyć?	158
Jak to zepsuć?	158
Zadania dodatkowe	158
<b>Ćwiczenie 26. Projekt logfind</b>	<b>160</b>

Specyfikacja logfind	160
<b>Ćwiczenie 27. Programowanie kreatywne i defensywne</b>	<b>162</b>
Nastawienie programowania kreatywnego	162
Nastawienie programowania defensywnego	163
8 strategii programisty defensywnego	164
Zastosowanie ośmiu strategii	164
Nigdy nie ufaj danym wejściowym	164
Unikanie błędów	168
Awarie powinny być wczesne i otwarte	169
Dokumentuj założenia	170
Preferuj prewencję zamiast dokumentacji	170
Automatyzuj wszystko	171
Upraszczaj i wyjaśniaj	171
Myśl logicznie	172
Kolejność nie ma znaczenia	172
Zadania dodatkowe	173
<b>Ćwiczenie 28. Pośrednie pliki Makefile</b>	<b>174</b>
Podstawowa struktura projektu	174
Makefile	175
Nagłówek	176
Docelowe wersje programu	177
Testy jednostkowe	178
Operacje porządkujące	180
Instalacja	180
Sprawdzenie	180
Co powinieneś zobaczyć?	181
Zadania dodatkowe	181
<b>Ćwiczenie 29. Biblioteki i linkowanie</b>	<b>182</b>
Dynamiczne wczytywanie biblioteki współdzielonej	183
Co powinieneś zobaczyć?	185
Jak to zepsuć?	187
Zadania dodatkowe	187
<b>Ćwiczenie 30. Zautomatyzowane testowanie</b>	<b>188</b>
Przygotowanie frameworka testów jednostkowych	189
Zadania dodatkowe	193
<b>Ćwiczenie 31. Najczęściej spotykane niezdefiniowane zachowanie</b>	<b>194</b>
20 najczęściej spotykanych przypadków niezdefiniowanego zachowania	196
Najczęściej spotykane niezdefiniowane zachowanie	196
<b>Ćwiczenie 32. Lista dwukierunkowa</b>	<b>200</b>

Czym są struktury danych?	200
Budowa biblioteki	200
Lista dwukierunkowa	202
Definicja	202
Implementacja	204
Testy	207
Co powinieneś zobaczyć?	210
Jak można usprawnić kod?	210
Zadania dodatkowe	211
<b>Ćwiczenie 33. Algorytmy listy dwukierunkowej</b>	<b>212</b>
Sortowanie bąbelkowe i sortowanie przez scalanie	212
Test jednostkowy	213
Implementacja	215
Co powinieneś zobaczyć?	217
Jak można usprawnić kod?	218
Zadania dodatkowe	219
<b>Ćwiczenie 34. Tablica dynamiczna</b>	<b>220</b>
Wady i zalety	227
Jak można usprawnić kod?	228
Zadania dodatkowe	228
<b>Ćwiczenie 35. Sortowanie i wyszukiwanie</b>	<b>230</b>
Sortowanie pozycyjne i wyszukiwanie binarne	233
Unie w języku C	234
Implementacja	235
Funkcja RadixMap_find() i wyszukiwanie binarne	241
RadixMap_sort() i radix_sort()	242
Jak można usprawnić kod?	243
Zadania dodatkowe	244
<b>Ćwiczenie 36. Bezpieczniejsze ciągi tekstowe</b>	<b>246</b>
Dlaczego stosowanie ciągów tekstowych C to niewiarygodnie kiepski pomysł?	246
Użycie bstrlib	248
Poznajemy bibliotekę	249
<b>Ćwiczenie 37. Struktura Hashmap</b>	<b>250</b>
Testy jednostkowe	257
Jak można usprawnić kod?	259
Zadania dodatkowe	260
<b>Ćwiczenie 38. Algorytmy struktury Hashmap</b>	<b>262</b>
Co powinieneś zobaczyć?	267
Jak to zepsuć?	268

Zadania dodatkowe	269
<b>Ćwiczenie 39. Algorytmy ciągu tekstowego</b>	<b>270</b>
Co powinieneś zobaczyć?	277
Analiza wyników	279
Zadania dodatkowe	280
<b>Ćwiczenie 40. Binarne drzewo poszukiwań</b>	<b>282</b>
Jak można usprawnić kod?	295
Zadania dodatkowe	295
<b>Ćwiczenie 41. Projekt devpkg</b>	<b>296</b>
Co to jest devpkg?	296
Co chcemy zbudować?	296
Projekt	297
Biblioteki Apache Portable Runtime	297
Przygotowanie projektu	299
Pozostałe zależności	299
Plik Makefile	299
Pliki kodu źródłowego	300
Funkcje bazy danych	302
Funkcje powłoki	305
Funkcje poleceń programu	309
Funkcja main() w devpkg	314
Ostatnie wyzwanie	316
<b>Ćwiczenie 42. Stos i kolejka</b>	<b>318</b>
Co powinieneś zobaczyć?	321
Jak można usprawnić kod?	321
Zadania dodatkowe	322
<b>Ćwiczenie 43. Prosty silnik dla danych statystycznych</b>	<b>324</b>
Odchylenie standardowe i średnia	324
Implementacja	325
Jak można użyć tego rozwiązania?	330
Zadania dodatkowe	331
<b>Ćwiczenie 44. Bufor cykliczny</b>	<b>334</b>
Testy jednostkowe	337
Co powinieneś zobaczyć?	337
Jak można usprawnić kod?	338
Zadania dodatkowe	338
<b>Ćwiczenie 45. Prosty klient TCP/IP</b>	<b>340</b>
Modyfikacja pliku Makefile	340
Kod netclient	340

Co powinieneś zobaczyć?	344
Jak to zepsuć?	344
Zadania dodatkowe	344
<b>Ćwiczenie 46. Drzewo trójkowe</b>	<b>346</b>
Wady i zalety	354
Jak można usprawnić kod?	355
Zadania dodatkowe	355
<b>Ćwiczenie 47. Szybszy router URL</b>	<b>356</b>
Co powinieneś zobaczyć?	358
Jak można usprawnić kod?	359
Zadania dodatkowe	360
<b>Ćwiczenie 48. Prosty serwer sieciowy</b>	<b>362</b>
Specyfikacja	362
<b>Ćwiczenie 49. Serwer danych statystycznych</b>	<b>364</b>
Specyfikacja	364
<b>Ćwiczenie 50. Routing danych statystycznych</b>	<b>366</b>
<b>Ćwiczenie 51. Przechowywanie danych statystycznych</b>	<b>368</b>
Specyfikacja	368
<b>Ćwiczenie 52. Hacking i usprawnianie serwera</b>	<b>370</b>
<b>Zakończenie</b>	<b>372</b>
<b>Skorowidz</b>	<b>373</b>