

Istnieje bardzo wiele nowoczesnych języków programowania, które pozwalają na szybkie wdrożenie i pracę. Takim językiem na pewno nie jest C. Niektóre jego cechy bardzo utrudniają tworzenie bezpiecznego i bezawaryjnego kodu. Warto więc dogłębnie poznać C — przy bardzo prostej składni i niewielkich wymaganiach sprzętowych ma potężne możliwości!

Niniejsza książka jest bardzo dobrym podręcznikiem dla początkujących programistów. Nauczysz się C, wykonując 52 sprytnie skonstruowane zadania zilustrowane kodem i specjalnie opracowanymi klipami wideo. Duży nacisk został położony na dogłębną analizę tworzonego kodu — autor zmusza Czytelnika do zrozumienia znaczenia każdej linii programu, do koncentracji i dokładności. Zachęca też do praktykowania tzw. programowania defensywnego, dzięki któremu możliwe jest podniesienie jakości i bezpieczeństwa tworzonego oprogramowania. Wartościowym elementem książki są wskazówki, jak zepsuć napisany kod, a następnie go zabezpieczyć. Bardzo ułatwia to unikanie wielu poważnych, często spotykanych błędów.

Spis treści

### **Podziękowania (12)**

#### **Ta książka tak naprawdę nie jest o języku C (13)**

- 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)

### **Przygotowania (20)**

- Linux (20)
- OS X (20)
- Windows (21)
- Edytor tekstu (21)
  - Nie używaj IDE (22)

### **Ćwiczenie 1. Odkurzenie kompilatora (24)**

- Omówienie kodu w pliku (24)
- Co powinieneś zobaczyć? (25)
- Jak to zepsuć? (26)
- Zadania dodatkowe (26)

### **Ćwiczenie 2. Użycie pliku Makefile podczas kompilacji (28)**

- Użycie narzędzia make (28)
- Co powinieneś zobaczyć? (29)
- Jak to zepsuć? (30)
- Zadania dodatkowe (30)

### **Ćwiczenie 3. Sformatowane dane wyjściowe (32)**

- Co powinieneś zobaczyć? (33)
- Zewnętrzne badania (33)
- Jak to zepsuć? (33)
- Zadania dodatkowe (34)

#### **Ćwiczenie 4. Użycie debugera (36)**

- Sztuczki z GDB (36)
- Krótki przewodnik po GDB (36)
- Krótki przewodnik po LLDB (37)

#### **Ćwiczenie 5. Nauka na pamięć operatorów w C (40)**

- Jak uczyć się na pamięć? (40)
- Listy operatorów (41)

#### **Ćwiczenie 6. Nauka na pamięć składni C (46)**

- Słowa kluczowe (46)
- Składnia struktur (47)
- Słowo zachęty (50)
- Słowo ostrzeżenia (51)

#### **Ćwiczenie 7. Zmienne i typy (52)**

- Co powinieneś zobaczyć? (53)
- Jak to zepsuć? (54)
- Zadania dodatkowe (54)

#### **Ćwiczenie 8. Konstrukcje if, else-if i else (56)**

- Co powinieneś zobaczyć? (57)
- Jak to zepsuć? (57)
- Zadania dodatkowe (58)

#### **Ćwiczenie 9. Pętla while i wyrażenia boolowskie (60)**

- Co powinieneś zobaczyć? (60)
- Jak to zepsuć? (61)
- Zadania dodatkowe (61)

#### **Ćwiczenie 10. Konstrukcja switch (62)**

- Co powinieneś zobaczyć? (64)
- Jak to zepsuć? (65)
- Zadania dodatkowe (65)

#### **Ćwiczenie 11. Tablice i ciągi tekstowe (66)**

- Co powinieneś zobaczyć? (67)
- Jak to zepsuć? (68)
- Zadania dodatkowe (69)

#### **Ćwiczenie 12. Wielkość i tablice (70)**

- Co powinieneś zobaczyć? (71)
- Jak to zepsuć? (72)
- Zadania dodatkowe (73)

#### **Ćwiczenie 13. Pętla for i tablica ciągów tekstowych (74)**

- Co powinieneś zobaczyć? (75)
- Zrozumienie tablicy ciągów tekstowych (76)
- Jak to zepsuć? (76)
- Zadania dodatkowe (77)

#### **Ćwiczenie 14. Tworzenie i użycie funkcji (78)**

Co powinienes zobaczyć? (79)  
Jak to zepsuć? (80)  
Zadania dodatkowe (80)

#### **Ćwiczenie 15. Wskaźniki, przerażające wskaźniki (82)**

Co powinienes zobaczyć? (84)  
Poznajemy wskaźniki (85)  
Praktyczne użycie wskaźników (86)  
Leksykon wskaźnika (87)  
Wskaźniki nie są tablicami (87)  
Jak to zepsuć? (87)  
Zadania dodatkowe (88)

#### **Ćwiczenie 16. Struktury i prowadzące do nich wskaźniki (90)**

Co powinienes zobaczyć? (93)  
Poznajemy struktury (94)  
Jak to zepsuć? (94)  
Zadania dodatkowe (95)

#### **Ćwiczenie 17. Alokacja pamięci stosu i sterty (96)**

Co powinienes zobaczyć? (102)  
Alokacja stosu kontra sterty (102)  
Jak to zepsuć? (103)  
Zadania dodatkowe (104)

#### **Ćwiczenie 18. Wskaźniki do funkcji (106)**

Co powinienes zobaczyć? (110)  
Jak to zepsuć? (110)  
Zadania dodatkowe (111)

#### **Ćwiczenie 19. Opracowane przez Zeda wspaniałe makra debugowania (112)**

Problem obsługi błędów w C (112)  
Makra debugowania (113)  
Użycie dbg.h (115)  
Co powinienes zobaczyć? (118)  
W jaki sposób CPP obsługuje makra? (118)  
Zadania dodatkowe (120)

#### **Ćwiczenie 20. Zaawansowane techniki debugowania (122)**

Użycie makra debug() kontra GDB (122)  
Strategia debugowania (124)  
Zadania dodatkowe (125)

#### **Ćwiczenie 21. Zaawansowane typy danych i kontrola przepływu (126)**

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)

### **Ćwiczenie 22. Stos, zakres i elementy globalne (134)**

- 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)

### **Ćwiczenie 23. Poznaj mechanizm Duffa (142)**

- Co powinieneś zobaczyć? (145)
- Rozwiązanie łamigłówki (145)
  - Dlaczego w ogóle mam się tak męczyć? (146)
- Zadania dodatkowe (146)

### **Ćwiczenie 24. Dane wejściowe, dane wyjściowe i pliki (148)**

- Co powinieneś zobaczyć? (150)
- Jak to zepsuć? (151)
- Funkcje wejścia-wyjścia (151)
- Zadania dodatkowe (152)

### **Ćwiczenie 25. Funkcje o zmiennej liczbie argumentów (154)**

- Co powinieneś zobaczyć? (158)
- Jak to zepsuć? (158)
- Zadania dodatkowe (158)

### **Ćwiczenie 26. Projekt logfind (160)**

- Specyfikacja logfind (160)

### **Ćwiczenie 27. Programowanie kreatywne i defensywne (162)**

- 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)

### **Ćwiczenie 28. Pośrednie pliki Makefile (174)**

- 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)

### **Ćwiczenie 29. Biblioteki i linkowanie (182)**

- Dynamiczne wczytywanie biblioteki współdzielonej (183)
- Co powinieneś zobaczyć? (185)
- Jak to zepsuć? (187)
- Zadania dodatkowe (187)

### **Ćwiczenie 30. Zautomatyzowane testowanie (188)**

- Przygotowanie frameworka testów jednostkowych (189)
- Zadania dodatkowe (193)

### **Ćwiczenie 31. Najczęściej spotykane niezdefiniowane zachowanie (194)**

- 20 najczęściej spotykanych przypadków niezdefiniowanego zachowania (196)
- Najczęściej spotykane niezdefiniowane zachowanie (196)

### **Ćwiczenie 32. Lista dwukierunkowa (200)**

- 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)

### **Ćwiczenie 33. Algorytmy listy dwukierunkowej (212)**

- Sortowanie bąbelkowe i sortowanie przez scalanie (212)
- Test jednostkowy (213)
- Implementacja (215)
- Co powinieneś zobaczyć? (217)
- Jak można usprawnić kod? (218)
- Zadania dodatkowe (219)

### **Ćwiczenie 34. Tablica dynamiczna (220)**

- Wady i zalety (227)
- Jak można usprawnić kod? (228)
- Zadania dodatkowe (228)

### **Ćwiczenie 35. Sortowanie i wyszukiwanie (230)**

- 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)

### **Ćwiczenie 36. Bezpieczniejsze ciągi tekstowe (246)**

- Dlaczego stosowanie ciągów tekstowych C to niewiarygodnie kiepski pomysł? (246)
- Użycie bstrlib (248)
- Poznajemy bibliotekę (249)

### **Ćwiczenie 37. Struktura Hashmap (250)**

- Testy jednostkowe (257)
- Jak można usprawnić kod? (259)
- Zadania dodatkowe (260)

### **Ćwiczenie 38. Algorytmy struktury Hashmap (262)**

- Co powinieneś zobaczyć? (267)
- Jak to zepsuć? (268)
- Zadania dodatkowe (269)

### **Ćwiczenie 39. Algorytmy ciągu tekstowego (270)**

- Co powinieneś zobaczyć? (277)
- Analiza wyników (279)
- Zadania dodatkowe (280)

### **Ćwiczenie 40. Binarne drzewo poszukiwań (282)**

- Jak można usprawnić kod? (295)
- Zadania dodatkowe (295)

### **Ćwiczenie 41. Projekt devpkg (296)**

- 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)

### **Ćwiczenie 42. Stos i kolejka (318)**

- Co powinieneś zobaczyć? (321)
- Jak można usprawnić kod? (321)
- Zadania dodatkowe (322)

**Ćwiczenie 43. Prosty silnik dla danych statystycznych (324)**

Odchylenie standardowe i średnia (324)  
Implementacja (325)  
Jak można użyć tego rozwiązania? (330)  
Zadania dodatkowe (331)

**Ćwiczenie 44. Bufor cykliczny (334)**

Testy jednostkowe (337)  
Co powinienś zobaczyć? (337)  
Jak można usprawnić kod? (338)  
Zadania dodatkowe (338)

**Ćwiczenie 45. Prosty klient TCP/IP (340)**

Modyfikacja pliku Makefile (340)  
Kod netclient (340)  
Co powinienś zobaczyć? (344)  
Jak to zepsuć? (344)  
Zadania dodatkowe (344)

**Ćwiczenie 46. Drzewo trójkowe (346)**

Wady i zalety (354)  
Jak można usprawnić kod? (355)  
Zadania dodatkowe (355)

**Ćwiczenie 47. Szybszy router URL (356)**

Co powinienś zobaczyć? (358)  
Jak można usprawnić kod? (359)  
Zadania dodatkowe (360)

**Ćwiczenie 48. Prosty serwer sieciowy (362)**

Specyfikacja (362)

**Ćwiczenie 49. Serwer danych statystycznych (364)**

Specyfikacja (364)

**Ćwiczenie 50. Routing danych statystycznych (366)**

**Ćwiczenie 51. Przechowywanie danych statystycznych (368)**

Specyfikacja (368)

**Ćwiczenie 52. Hacking i usprawnianie serwera (370)**

**Zakończenie (372)**

**Skorowidz (373)**