

Wstęp (15)

Podziękowania (23)

Listy kontrolne (25)

Tabele (27)

Rysunki (29)

Część I: Proces budowy oprogramowania (35)

1. Budowa oprogramowania (37)

- 1.1. Czym jest budowa oprogramowania (37)
- 1.2. Znaczenie procesu budowy oprogramowania (40)
- 1.3. Jak korzystać z tej książki (41)

2. Metafory procesu programowania (43)

- 2.1. Znaczenie metafor (43)
- 2.2. Jak korzystać z metafor w programowaniu (46)
- 2.3. Popularne metafory programowania (47)

3. Przed programowaniem - przygotowania (57)

- 3.1. Przygotowania i ich znaczenie (58)
- 3.2. Określanie rodzaju budowanego oprogramowania (65)
- 3.3. Definicja problemu (70)
- 3.4. Określenie wymagań (72)
- 3.5. Architektura (77)
- 3.6. Ilość czasu poświęcanego na przygotowania (89)

4. Kluczowe decyzje konstrukcyjne (95)

- 4.1. Wybór języka programowania (95)
- 4.2. Konwencje programowania (100)
- 4.3. Twoje położenie na fali technologii (101)
- 4.4. Wybór podstawowych praktyk programowania (103)

Część II: Pisanie dobrego kodu (107)

5. Projektowanie (109)

- 5.1. Podstawowe problemy projektowania (110)
- 5.2. Podstawowe pojęcia projektowania (113)
- 5.3. Heurystyki - narzędzia projektanta (122)
- 5.4. Techniki projektowania (146)
- 5.5. Uwagi o popularnych metodykach pracy (155)

6. Klasy z klasą (161)

- 6.1. Abstrakcyjne typy danych (162)
- 6.2. Dobry interfejs klasy (169)
- 6.3. Problemy projektowania i implementacji (179)
- 6.4. Przesłanki dla utworzenia klasy (188)
- 6.5. Specyfika języka (192)

- 6.6. Pakiety klas (192)

7. Procedury wysokiej jakości (197)

- 7.1. Przesłanki utworzenia procedury (200)
- 7.2. Projektowanie na poziomie procedur (204)
- 7.3. Dobra nazwa procedury (207)
- 7.4. Jak długa może być procedura? (209)
- 7.5. Jak używać parametrów procedur (211)
- 7.6. Używanie funkcji (217)
- 7.7. Makra i procedury inline (218)

8. Programowanie defensywne (223)

- 8.1. Zabezpieczanie programu przed niewłaściwymi danymi wejściowymi (224)
- 8.2. Asercje (225)
- 8.3. Mechanizmy obsługi błędów (230)
- 8.4. Wyjątki (234)
- 8.5. Ograniczanie zasięgu szkód powodowanych przez błędy (239)
- 8.6. Kod wspomagający debugowanie (241)
- 8.7. Ilość kodu defensywnego w wersji finalnej (245)
- 8.8. Defensywne podejście do programowania defensywnego (246)

9. Proces Programowania w Pseudokodzie (251)

- 9.1. Budowanie klas i procedur krok po kroku (251)
- 9.2. Pseudokod dla zaawansowanych (253)
- 9.3. Budowanie procedur metodą PPP (256)
- 9.4. Alternatywy dla pseudokodu (269)

Część III: Zmienne (273)

10. Zmienne w programie (275)

- 10.1. Podstawowa wiedza o danych (276)
- 10.2. Deklarowanie zmiennych (277)
- 10.3. Inicjalizowanie zmiennych (278)
- 10.4. Zakres (282)
- 10.5. Trwałość (289)
- 10.6. Czas wiązania (290)
- 10.7. Związek między typami danych i strukturami sterowania (292)
- 10.8. Jedno przeznaczenie każdej zmiennej (293)

11. Potęga nazwy zmiennej (297)

- 11.1. Wybieranie dobrej nazwy (297)
- 11.2. Nazwy a rodzaje danych (303)
- 11.3. Potęga konwencji nazw (308)
- 11.4. Nieformalne konwencje nazw (310)
- 11.5. Standardowe prefiksy (317)
- 11.6. Nazwy krótkie a czytelne (319)

- 11.7. Nazwy, których należy unikać (322)

12. Podstawowe typy danych (327)

- 12.1. Liczby (327)
- 12.2. Liczby całkowite (329)
- 12.3. Liczby zmiennoprzecinkowe (331)
- 12.4. Znaki i ciągi znakowe (333)
- 12.5. Zmienne logiczne (336)
- 12.6. Typy wyliczeniowe (338)
- 12.7. Stałe nazwane (343)
- 12.8. Tablice (345)
- 12.9. Tworzenie własnych typów (aliasy) (346)

13. Inne typy danych (355)

- 13.1. Struktury (355)
- 13.2. Wskaźniki (359)
- 13.3. Dane globalne (371)

Część IV: Instrukcje (383)

14. Struktura kodu liniowego (385)

- 14.1. Instrukcje, które wymagają określonej kolejności (385)
- 14.2. Instrukcje, których kolejność nie ma znaczenia (388)

15. Instrukcje warunkowe (393)

- 15.1. Instrukcje if (393)
- 15.2. Instrukcje case (398)

16. Pętle (405)

- 16.1. Wybieranie rodzaju pętli (405)
- 16.2. Sterowanie pętlą (410)
- 16.3. Łatwe tworzenie pętli - od wewnątrz (422)
- 16.4. Pętle i tablice (424)

17. Nietypowe struktury sterowania (427)

- 17.1. Wiele wyjść z procedury (427)
- 17.2. Rekurencja (429)
- 17.3. Instrukcja goto (434)
- 17.4. Nietypowe struktury sterowania z perspektywy (444)

18. Metody oparte na tabelach (449)

- 18.1. Metody oparte na tabelach - wprowadzenie (449)
- 18.2. Tabele o dostępie bezpośrednim (451)
- 18.3. Tabele o dostępie indeksowym (462)

- 18.4. Tabele o dostępie schodkowym (464)
- 18.5. Inne metody wyszukiwania w tabelach (467)

19. Ogólne problemy sterowania (469)

- 19.1. Wyrażenia logiczne (469)
- 19.2. Instrukcje złożone (bloki) (480)
- 19.3. Instrukcje puste (481)
- 19.4. Praca z głębokimi zagnieżdżeniami (482)
- 19.5. Programowanie strukturalne (490)
- 19.6. Struktury sterujące i złożoność (493)

Część V: Sprawna praca z kodem (497)

20. Jakość oprogramowania (499)

- 20.1. Składowe jakości (499)
- 20.2. Metody podwyższania jakości (502)
- 20.3. Skuteczność metod podwyższania jakości (505)
- 20.4. Kiedy przeprowadzać kontrolę jakości (509)
- 20.5. Ogólna Zasada Jakości Oprogramowania (509)

21. Programowanie zespołowe (513)

- 21.1. Przegląd metod programowania zespołowego (514)
- 21.2. Programowanie w parach (517)
- 21.3. Formalne inspekcje (519)
- 21.4. Inne metody programowania zespołowego (526)

22. Testowanie (533)

- 22.1. Rola testów programisty (534)
- 22.2. Zalecane podejście do testów programisty (537)
- 22.3. Praktyczne techniki testowania (539)
- 22.4. Typowe błędy (550)
- 22.5. Narzędzia wspomagające testowanie (556)
- 22.6. Usprawnianie testów (561)
- 22.7. Gromadzenie informacji o testach (563)

23. Debugowanie (569)

- 23.1. Wprowadzenie (569)
- 23.2. Wyszukiwanie defektu (574)
- 23.3. Usuwanie defektu (585)
- 23.4. Debugowanie a psychologia (588)
- 23.5. Narzędzia debugowania - oczywiste i mniej oczywiste (591)

24. Refaktoryzacja (597)

- 24.1. Ewolucja oprogramowania i jej odmiany (598)
- 24.2. Refaktoryzacje - wprowadzenie (599)

- 24.3. Wybrane refaktoryzacje (605)
- 24.4. Bezpieczne przekształcanie kodu (613)
- 24.5. Strategie refaktoryzacji (615)

25. Strategie optymalizacji kodu (621)

- 25.1. Wydajność kodu (622)
- 25.2. Optymalizowanie kodu (625)
- 25.3. Rodzaje otyłości i lenistwa (632)
- 25.4. Pomiar (637)
- 25.5. Iterowanie (639)
- 25.6. Strategie optymalizacji kodu - podsumowanie (640)

26. Metody optymalizacji kodu (645)

- 26.1. Struktury logiczne (646)
- 26.2. Pętle (651)
- 26.3. Przekształcenia danych (660)
- 26.4. Wyrażenia (665)
- 26.5. Procedury (674)
- 26.6. Reimplementacja w języku niskiego poziomu (675)
- 26.7. Im bardziej świat się zmienia, tym więcej zostaje bez zmian (677)

Część VI: Środowisko programowania (681)

27. Jak rozmiar programu wpływa na jego budowę (683)

- 27.1. Wielkość projektu a komunikacja (684)
- 27.2. Skala rozmiarów projektów (684)
- 27.3. Wpływ wielkości projektu na liczbę błędów (685)
- 27.4. Wpływ wielkości projektu na efektywność pracy (687)
- 27.5. Wpływ wielkości projektu na wykonywaną pracę (687)

28. Zarządzanie w programowaniu (695)

- 28.1. Zachęcanie do budowy dobrego kodu (696)
- 28.2. Zarządzanie konfiguracją (698)
- 28.3. Budowanie harmonogramu (705)
- 28.4. Pomiar (712)
- 28.5. Ludzkie traktowanie programistów (715)
- 28.6. Współpraca z przełożonymi (721)

29. Integracja (725)

- 29.1. Znaczenie metod integracji (725)
- 29.2. Częstość integracji - końcowa czy przyrostowa? (727)
- 29.3. Przyrostowe strategie integracji (730)
- 29.4. Codzienna kompilacja i test dymowy (738)

30. Narzędzia programowania (747)

- 30.1. Narzędzia do projektowania (748)
- 30.2. Narzędzia do pracy z kodem źródłowym (748)
- 30.3. Narzędzia do pracy z kodem wykonywalnym (754)
- 30.4. Środowiska narzędzi programowania (758)
- 30.5. Budowanie własnych narzędzi (759)
- 30.6. Narzędzia przyszłości (761)

Część VII: Rzemiosło programisty (765)

31. Układ i styl (767)

- 31.1. Wprowadzenie (768)
- 31.2. Techniki formatowania (774)
- 31.3. Style formatowania (776)
- 31.4. Formatowanie struktur sterujących (782)
- 31.5. Formatowanie instrukcji (789)
- 31.6. Formatowanie komentarzy (800)
- 31.7. Formatowanie procedur (802)
- 31.8. Formatowanie klas (804)

32. Kod, który opisuje się sam (813)

- 32.1. Zewnętrzna dokumentacja programu (813)
- 32.2. Styl programowania jako dokumentacja (814)
- 32.3. Komentować czy nie komentować (817)
- 32.4. Zasady pisania dobrych komentarzy (821)
- 32.5. Metody pisania komentarzy (828)
- 32.6. Normy IEEE (849)

33. Cechy charakteru (855)

- 33.1. Czy osobowość jest bez znaczenia? (856)
- 33.2. Inteligencja i skromność (857)
- 33.3. Ciekawość (858)
- 33.4. Uczciwość intelektualna (862)
- 33.5. Komunikacja i współpraca (865)
- 33.6. Kreatywność i dyscyplina (865)
- 33.7. Lenistwo (866)
- 33.8. Cechy, które znaczą mniej, niż myślisz (867)
- 33.9. Nawyki (869)

34. Powracające wątki - przegląd (873)

- 34.1. Walka ze złożonością (873)
- 34.2. Wybierz swój proces (875)
- 34.3. Pisz programy dla ludzi, nie tylko dla komputerów (877)
- 34.4. Programuj do języka, a nie w nim (879)
- 34.5. Konwencje jako pomoc w koncentracji uwagi (880)
- 34.6. Programowanie w kategoriach dziedziny problemu (881)
- 34.7. Uwaga, spadające odłamki! (884)
- 34.8. Iteruj, iteruj i jeszcze raz iteruj (886)

- 34.9. Nie będziesz łączył religii z programowaniem (887)

35. Gdzie znaleźć więcej informacji (891)

- 35.1. Programowanie (892)
- 35.2. Szersze spojrzenie na budowę oprogramowania (893)
- 35.3. Periodyki (895)
- 35.4. Plan czytelniczy programisty (896)
- 35.5. Stowarzyszenia zawodowe (898)

Bibliografia (899)

Skorowidz (919)

Steve McConnell (947)