

Sztuka dobrego programowania / Krzysztof Jassem, Andrzej Ziemkiewicz. – Warszawa, 2016

Spis treści

Wstęp	13
Przygotowanie środowiska pracy	14
Instalacja systemu Linux	15
Pobranie OpenSUSE	15
Przygotowanie instalacji	15
Instalacja	15
Uruchamianie programów	16
Edycja kodu źródłowego	16
Przejsście do trybu tekstowego	16
Skompilowanie programu	17
Uruchomienie programu	17
Polecane źródła w Internecie	17
 Część I	
 Lekcja 1. Optymalizowanie funkcji	21
Elementy matematyki	21
Podstawy języka C	22
Program, instrukcja	22
Komentarze	23
Identyfikatory	23
Słowa kluczowe	23
Białe znaki	23
Podstawowe typy numeryczne	24
Zmienne i operator przypisania	25
Operatory	26
Instrukcja warunkowa	27
Instrukcje pętli	28
Pętla while	28
Pętla do while	29
Pętla for	29
Funkcje	30
Funkcje rekurencyjne	31
Rozwiązanie zadania	32
Krok 1. Pierwsze zadanie pomocnicze	32
Krok 2. Rachunki matematyczne	32
Krok 3. Drugie zadanie pomocnicze	33
Krok 4. Uogólnienie rachunków	34
Krok 5. Pierwsza wersja kodu funkcji rozwiązującej zadanie	34

Krok 6. Dodanie warunku zakończenia rekurencji	35
Krok 7. Optymalizacja liczby operacji	35
Krok 8. Optymalizacja liczby zmiennych	36
Krok 9. Usunięcie rekurencji	36
Krok 10. Sprawdzenie poprawności argumentów	38
Wnioski	39
Polecane źródła w Internecie	39

Lekcja 2. Działania na bitach **41**

Sito Eratostenesa	41
Podstawy Języka C	41
Operacje na bitach	41
Operacja and	42
Operacja or	42
Operacja xor	42
Operacja not	42
Przesunięcie bitowe w lewo	42
Przesunięcie bitowe w prawo	43
Rzutowanie	43
Dyrektywy preprocesora	44
Tablice	45
Adresy zmiennych	46
Wskaźniki	47
Wskaźniki a tablice	48
Alokowanie pamięci dla tablicy	49
Rozwiązanie zadania	50
Krok 1. Podejście standardowe	50
Krok 2. Zmniejszenie liczby operacji	51
Krok 3. Oszczędność pamięci	52
Krok 4. Poprawa efektywności algorytmu	57
Wnioski	58
Polecane źródła w Internecie	59
Podstawy informatyki	61

Lekcja 3. Alokowanie pamięci **61**

Podstawy języka C	62
Funkcja malloc()	62
Funkcja calloc()	63
Funkcja realloc()	64
Funkcja free()	64
Przykład zarządzania pamięcią	65
Zmienne automatyczne i statyczne	66
Rozwiązanie zadania	67
Wnioski	72
Polecane źródła w Internecie	72

Część II

Lekcja 4. Program główny	75
Podstawy języka C	75
Funkcja main()	75
Funkcja exit ()	75
Standardowe strumienie wejścia i wyjścia	76
Napisy	76
Konwersja napisu do liczby	77
Rozwiązanie zadania	78
Kroki. Zadanie pomocnicze	78
Krok 2. Unikanie prostych błędów	80
Krok 3. Zastosowanie nazwy programu	81
Krok 4. Obsługa opcji	83
Krok 5. Wyświetlenie komunikatu o błędzie	85
Krok 6. Zastosowanie zmiennych środowiskowych	85
Wnioski	86
Polecane źródła w Internecie	86
 Lekcja 5. Przetwarzanie opcji	 87
Podstawy języka C	87
Przekazywanie argumentów funkcji	87
Instrukcja wielokrotnego wyboru: switch	89
Standardowe opcje programu	90
Rozwiązanie zadania	90
Wnioski	94
Polecane źródła w Internecie	94
 Lekcja 6. Przetwarzanie parametrów wejściowych - plików	 95
Podstawy języka C	95
Odczyt i zapis do pliku	95
Trójargumentowy operator warunkowy	98
Sterowanie preprocesorem	98
Prototyp funkcji	98
Atrybut extern	100
Kompilacja programu	102
Rozwiązanie zadania	103
Kroki. Rozdzielenie kompetencji między funkcje	103
Krok 2. Rozdzielenie kompetencji między pliki	106
Krok 3. Utworzenie plików nagłówkowych	108
Krok 4. Przetwarzanie danych wejściowych	109
Krok 5. Kompilacja	110
Aneks	111
Polecane źródła w Internecie	116

Część III

Lekcja 7. Debugowanie programu	119
Na czym polega debugowanie	119
Podstawy języka C	120
Buforowanie	120
Wyrażenie przecinkowe	121
Funkcja char()	121
Rozwiązanie zadania	122
Krok 1. Wydruki kontrolne w kodzie programu	122
Krok 2. Zastosowanie makra	124
Krok 3. Makro ze zmienną liczbą parametrów	126
Krok 4. Debugowanie wybierane dynamicznie	128
Krok 5. Obsługa parametru wywołania +d	130
Wnioski	131
Aneks	132
Modyfikacje w funkcji set_opt()	132
Modyfikacje w funkcji main()	132
Modyfikacja wypisywania pomocy rozszerzonej	133
Polecane źródła w Internecie	133
 Lekcja 8. Budowanie złożonych programów	 135
Podstawy teoretyczne	135
Struktura pliku Makefile	135
Rozwiązanie zadania	136
Kroki. Tworzenie pliku Makefile	136
Deklaracje zmiennych	136
Reguły kompilacji	137
Krok 2. Optymalizacja pliku Makefile	138
Krok 3. Realizowanie celów specjalnych	139
make clean	139
make distclean	140
make dist	140
make install	142
Krok 4. Sprawdzenie poprawności listy zależności	142
Wnioski	143
Polecane źródła w Internecie	143
 Lekcja 9. Udostępnianie programu	 145
Narzędzia do udostępniania programów	145
Krok 1. Skanowanie katalogu	145
Krok 2. Edytowanie pliku configure.ac	146
Krok 3. Edytowanie pliku config.h.in	149
Krok 4. Edytowanie pliku makefile.in	149
Krok 5. Wywołanie programu autoconf	149
Krok 6. Uruchomienie programu configure	150
Krok 7. Korzystanie z pliku config.h	152

Krok 8. Budowanie programu	152
Krok 9. Dystrybucja	153
Wnioski	153
Polecane źródła w Internecie	153

Część IV

Lekcja 10. Dynamiczne struktury danych 157

Funkcje rekurencyjne	157
Rekurencyjne struktury danych	159
Dynamiczne struktury danych	160
Listy	160
Drzewa binarne	162
Porównanie przetwarzania list i drzew	163
Realizacja dynamicznych struktur danych w języku C	164
Struktury	164
Rozwiązanie zadania	165
Reprezentacja węzła drzewa	165
Kroki. Tworzenie drzewa	166
Krok 2. Wstawianie elementu na początek i na koniec listy	168
Krok 3. Zamiana drzewa na listę	170
Krok 4. Zamiana listy na drzewo	175
Krok 5. Wypisywanie zawartości drzewa	180
Krok 6. Wyszukiwanie informacji w drzewie	181
Krok 7. Optymalizacja wyszukiwania	181
Krok 8. Baza z wieloma kluczami	182
Wnioski	183
Polecane źródła w Internecie	183

Lekcja 11. Wyrażenia regularne 185

Automaty skończone	185
Wyrażenia regularne	187
Metaznaki	188
Rozpoznawanie wyrażeń regularnych	189
Rozwiązanie zadania	189
Krok 1. Sformułowanie zadania dla konkretnego zastosowania	189
Krok 2. Budowanie tablicy sterującej	190
Wnioski	197
Polecane źródła w Internecie	198

Lekcja 12. Interpreter poleceń 199

Gramatyki formalne	200
Narzędzia informatyczne	201
Generator parsera yacc	202
Deklaracje języka C	203
Deklaracje parsera	203

Reguły gramatyki	203
Kod dodatkowy	204
Generator leksera lex	204
Reguły leksykalne	205
Rozwiązanie zadania	205
Krok 1. Tworzenie pliku <i>mag.y</i>	205
Krok 2. Generowanie parsera	207
Krok 3. Tworzenie pliku <i>mag.l</i>	208
Krok 4. Generowanie analizatora leksykalnego	209
Krok 5. Tworzenie prostego programu głównego	210
Krok 6. Modyfikacja pliku Makefile	212
Krok 7. Wywołanie funkcji <i>yyparse()</i> w pętli	213
Krok 8. Sterowanie reakcjami systemu	215
Modyfikacja leksera	217
Modyfikacja funkcji <i>main()</i>	217
Modyfikacja parsera	218
Krok 9. Rozwijanie interpretera	222
Krok 10. Rozwiązywanie konfliktów	225
Wnioski	226
Polecane źródła w Internecie	227
Pliki towarzyszące książce	229
Skorowidz	233