Obsah

Předmluva ke čtvrtému vydání 16
Předmluva k prvnímu vydání prvního dílu 18

1 Úvod 20
1.1 C – vznik, vývoj, charakteristika 20
1.2 Typografické a syntaktické konvence 21
1.3 Styl psaní programů 22

2 Základní pojmy 23
2.1 Způsob zpracování programu 23
2.2 Základní pojmy v jazyce C 24
2.2.1 Zdrojové a hlavičkové soubory 25
2.2.2 Bílé znaky 26
2.2.3 ASCII tabulka 26
2.2.4 Identifikátory 27
2.2.5 Komentáře 28

3 První začátky s C 29
3.1 Jednoduché datové typy a přiřazení 29
3.1.1 Definice proměnných 30
3.1.2 Přiřazení 31
3.2 Hlavní program 31
3.3 Konstanty 33
3.3.1 Celočíselné konstanty 33
3.3.2 Reálné konstanty 34
3.3.3 Znakové konstanty 34
3.3.4 Řetězcové konstanty (literály) 35
3.4 Aritmetické výrazy 35
3.4.1 Unární operátory 35
3.4.2 Binární operátory 36
3.4.3 Speciální unární operátory 36
3.4.4 Přiřazovací operátory 37

4 Terminálový vstup a výstup 39
4.1 Hlavičkový soubor `stdio.h` 39
4.2 Vstup a výstup znaku 39
4.3 Formátovaný vstup a výstup 40
4.3.1 Řídící řetězec formátu 41

5 Řídící struktury 46
5.1 Booleovské výrazy 48
5.1.1 Zkrácené vyhodnocování logických výrazů 48
5.1.2 Priority vyhodnocování výrazů 49
5.2 Podmíněný výraz – ternární operátor 50
5.3 Operátor čárky 51
5.4 Příkaz `if` a příkaz `if--else` 52
5.5 Iterační příkazy – cykly 55
5.5.1 Příkazy `break` a `continue` 55
5.5.2 Příkaz `while` 56
5.5.3 Příkaz `do--while` 57
5.5.4 Příkaz `for` 58
5.6 Příkaz `switch` 60
5.7 Příkaz `goto` 65
5.8 Příkaz `return` 65

6 Vstup ze souboru a výstup do souboru 69
6.1 Začátek práce se souborem 71
6.1.1 Otevření souboru pro čtení 72
6.1.2 Otevření souboru pro zápis 72
6.2 Základní operace s otevřeným souborem 72
6.3 Ukončení práce se souborem 73
6.4 Příklady základní práce se soubory 73
6.5 Testování konce řádky 75
6.6 Testování konce souboru 77
6.6.1 Pomocí symbolické konstanty `EOF` 77
6.6.2 Pomocí standardního makra `feof()` 78
6.7 Testování správnosti otevření a uzavření souboru 78
6.8 Standardní vstup a výstup 82
6.9 Vrácení přečteného znaku zpět do vstupního bufferu 84

7 Typová konverze 87
7.1 Implicitní typová konverze 87
7.2 Explicitní typová konverze 88
8 Preprocesor jazyka C 90
8.1 Makra bez parametrů – příkaz `define` 91
8.2 Makra s parametry 94
8.2.1 Předdefinovaná makra 95
8.3 Vkládání souborů – příkaz `include` 97
8.3.1 Vkládané soubory 97
8.3.2 Standardní hlavičkové soubory 98
8.4 Oddělený překlad souborů – I. 99
8.5 Podmíněný překlad 100
8.5.1 Řízení překladu hodnotou konstantního výrazu 101
8.5.2 Řízení překladu definicí makra 103
8.5.3 Operátor `defined` 104
8.5.4 Direktivy `#elif` a `#error` 104
9 Funkce a práce s pamětí 107
9.1 Alokace paměti 108
9.1.1 Statická alokace 108
9.1.2 Dynamická alokace na hromadě 109
9.1.3 Dynamická alokace v zásobníku 109
9.2 Funkce 110
9.2.1 Definice funkce 110
9.2.2 Procedury a datový typ `void` 112
9.2.3 Rekurzivní funkce 113
9.2.4 Funkce nevracející `int` 113
9.2.5 Problémy s umístěním definice funkcí 113
9.2.6 Konverze návratové hodnoty funkce 116
9.2.7 Parametry funkcí 116
Konverze skutečných parametrů 117
9.3 Oblast platnosti identifikátorů 117
9.3.1 Globální a lokální proměnné 117
9.3.2 Paměťové třídy 121
Třída `auto` 121
Třída `extern` 122
Třída `static` 122
Třída `register` 123

9.3.3 Typové modifikátory 124
Modifikátor `const` 124
Modifikátor `volatile` 125
9.3.4 Bloky 126
9.4 Oddělený překlad souborů – II. 127
9.4.1 Rozšíření platnosti globální proměnné 127
9.4.2 Statické globální proměnné a funkce 128
9.4.3 Jak udržet pořádek ve velkém programu 130
Doporučený obsah `.c` souboru 131
Doporučený obsah `.h` souboru 132
9.5 Inicializace jednoduchých proměnných 140

10 Pointery 144

10.1 Základy práce s pointery 145
10.1.1 Definice dat typu pointer na typ 145
10.1.2 Práce s adresovými operátory 146
10.1.3 Přiřazení hodnoty pointerům a pomocí pointerů 146
10.1.4 Použití pointerů v přiřazovacích příkazech 147
10.1.5 Nulový pointer `NULL` 149
10.1.6 Konverze pointerů 150
10.1.7 Zarovnávání v paměti 150
10.2 Pointery a funkce 150
10.2.1 Volání odkazem 151
10.2.2 Pointer na typ `void` 154
Pointer na typ `void` jako pointer na několik různých typů 154
Pointer na typ `void` jako formální parametr funkce 155
10.2.3 Pointery na funkce a funkce jako parametry funkcí ... 155
10.3 Jak číst komplikované definice – I. 158
10.4 Definice s využitím operátoru `typedef` 159
10.5 Pointerová aritmetika 160
10.5.1 Operátor `sizeof` 161
10.5.2 Součet pointeru a celého čísla 161
10.5.3 Odečítání celého čísla od pointeru 163
10.5.4 Porovnávání pointerů 163
10.5.5 Odečítání pointerů 164
10.6 Dynamické přidělování a navracení paměti 164
10.6.1 Přidělení paměti 165
10.6.2 Uvolňování paměti 167
10.6.3 Příklady přidělování paměti 167

10.6.4 Funkce `calloc()` 168
10.7 Pointer jako skutečný parametr funkce 169
11 Jednorozměrná pole 173
11.1 Základní dovednosti 173
11.2 Pole a pointery 176
11.2.1 Dynamická pole 177
11.2.2 Podobnost statických a dynamických polí 178
11.2.3 Další zvláštnosti a dovednosti při práci s poli 179
Práce s celým polem najednou 179
Přístup do pole pomocí pointerů 179
Jak zjistit velikost pole 181
11.3 Pole měnící svoji velikost 181
11.4 Pole jako parametry funkcí 183
11.5 Pole pointerů na funkce 187
11.6 Jak číst komplikované definice – II. 188
12 Řetězce 192
12.1 Základní informace a definování řetězců 192
12.2 Práce s řetězcem 196
12.2.1 Čtení řetězce z klávesnice 196
Čtení řetězce v daném formátu 196
12.2.2 Tisk řetězce na obrazovku 198
12.2.3 Přístup k jednotlivým znakům řetězce 199
12.2.4 Standardní funkce pro práci s řetězci 200
Délka řetězce 200
Kopírování řetězce 200
Spojení řetězců 200
Nalezení znaku v řetězci 200
Porovnání dvou řetězců 200
Nalezení podřetězce v řetězci 201
Práce s omezenou částí řetězce 201
Práce s řetězcem pozpátku 201
Převody řetězců na čísla 201
12.3 Formátované čtení a zápis z a do řetězce 202
12.4 Řádkově orientovaný vstup a výstup z terminálu 204
12.4.1 Čtení řádky z klávesnice 204
12.4.2 Výpis řádky na obrazovku 205
12.5 Řádkově orientovaný vstup a výstup ze souboru 205

12.5.1 Čtení řádky ze souboru 206
12.5.2 Zápis řádky do souboru 208
12.6 Řídící řetězec formátu pro tisk 209
12.6.1 *konverze* *209*
12.6.2 *modifikátor* *210*
12.6.3 *šířka* *210*
12.6.4 *přesnost* *211*
12.6.5 *příznak* *211*
12.6.6 Příklady různých formátů tisku 212
13 Vícerozměrná pole 215
13.1 Základní definice a přístup k prvkům 215
13.2 Uložení vícerozměrných polí v paměti 216
13.3 Různé způsoby definice dvourozměrných polí 218
13.3.1 Statické dvourozměrné pole 218
13.3.2 Pole pointerů 218
13.3.3 Pointer na pole 219
13.3.4 Pointer na pointer 220
13.3.5 Výhody a nevýhody předchozích čtyř způsobů 220
13.3.6 Dvourozměrné pole jako parametr funkce 222
13.4 Inicializace polí všech rozměrů 224
13.5 Pole řetězců 225
13.6 Parametry funkce `main()` 227
13.7 Externí pole všech rozměrů 229
14 Struktury, uniony a výčtové typy 232
14.1 Struktury 232
14.1.1 Definice a základní dovednosti 232
14.1.2 Struktury a pointery 235
14.1.3 Struktury odkazující samy na sebe 236
14.1.4 Struktura v jiné struktuře 238
14.1.5 Položky struktury jsou pointery 240
14.1.6 Alokace paměti pro jednotlivé položky struktury 244
14.1.7 Struktury a funkce 245
14.1.8 Shrnutí poznatků o práci s jednotlivými strukturami . 248
14.1.9 Pole struktur 249
14.1.10 Inicializace struktur 252
14.2 Výčtový typ 253
14.3 Uniony 255

15 Bitové operace a bitové pole 262
15.1 Operace s jednotlivými bity 262
15.1.1 Bitový součin 263
15.1.2 Bitový součet 263
15.1.3 Bitový exkluzivní součet 263
15.1.4 Operace bitového posunu doleva 264
15.1.5 Operace bitového posunu doprava 264
15.1.6 Negace bit po bitu 265
15.1.7 Způsoby práce se skupinou bitů 266
15.2 Bitové pole 266
16 Tabulka preferencí 269

Obsah druhého dílu
(je uveden pro úplnost – podrobnosti viz v předmluvě)

17 Komplexní pohled na souborový vstup a výstup 272
17.1 Rozdíly mezi binárním a textovým souborem obecně 273
17.1.1 Binární soubory 273
17.1.2 Textové soubory 274
17.2 Rozdíly mezi binárním a textovým režimem otevírání souboru 274
17.3 Otevření souboru 277
17.3.1 Různé způsoby otevírání souboru 278
17.4 Uzavření souboru 281
17.5 Ošetření chyb 283
17.6 Čtení a zápis dat 285
17.6.1 Formátované vstupy a výstupy 286
17.6.2 Neformátované vstupy a výstupy 286
Zpracování jednoho znaku v jeden okamžik 287
Zpracování celé řádky najednou 290
Zpracování celého bloku dat najednou 292
17.7 Bufferování 293
17.7.1 Základní informace 294
17.7.2 Možnosti bufferování poskytnuté ANSI C 295
17.7.3 Funkce pro práci s bufferem 295
Funkce `setbuf()` 295
Funkce `setvbuf()` 296
17.7.4 Řešení problému nebufferovaného `stdin` 297

17.8 Funkce pro zápis bufferu na disk 299
17.8.1 Funkce `fflush()` 299
17.9 Přímý přístup do souboru 300
17.9.1 Funkce `fseek()` – posun v souboru 300
17.9.2 Funkce `ftell()` – zjištění pozice v souboru 300
17.9.3 Typické použití přímého přístupu 302
17.10 Zbývající užitečné funkce ze `<stdio.h>` 306
17.10.1 Funkce `freopen()` – přesměrování proudu 306
17.10.2 Funkce `rename()` – přejmenování souboru 308
17.10.3 Funkce `remove()` – zrušení souboru 308
17.10.4 Funkce `tmpfile()` – pomocný soubor 308
17.10.5 Funkce `tmpnam()` – jméno pomocného souboru 308
17.11 Ukázka použití funkcí z UNIXové knihovny 309
17.12 Zjištění informací o položkách v adresáři 311
17.12.1 Prostředí UNIXu 311
Zjištění pouze jmen položek 312
Zjištění typu a stavu položek 313
17.12.2 Prostředí Windows 316

18 Návaznost překladače ANSI C na okolní prostředí 319
18.1 Standardní hlavičkové soubory 319
18.2 Rezervovaná jména 321
18.3 Standardní hlavičkové soubory bez funkčních prototypů 322
18.3.1 Soubor `<errno.h>` 322
18.3.2 Soubor `<limits.h>` 323
18.3.3 Soubor `<float.h>` 323
18.3.4 Soubor `<stddef.h>` 324
18.4 Předdefinovaná makra preprocesoru 325
18.4.1 Makro `__STDC__` 325
18.4.2 Makro `__FILE__` 325
18.4.3 Makro `__LINE__` 325
18.4.4 Makro `__DATE__` 325
18.4.5 Makro `__TIME__` 326
18.5 Externí proměnné 326
18.5.1 Proměnná `errno` z `<errno.h>` 326
18.5.2 Proměnná `sys_errlist` ze `<stdlib.h>` 327
18.5.3 Proměnná `sys_nerr` ze `<stdlib.h>` 327
18.5.4 Proměnná `environ` ze `<stdlib.h>` 328
18.5.5 Proměnná `_iob` ze `<stdio.h>` 328

19 Popis funkcí a maker ze standardních knihoven 329

19.1 `<assert.h>` – makro používané při ladění 329

19.2 `<ctype.h>` – práce se znaky 329

19.3 `<math.h>` – matematické funkce v přesnosti `double` 330

19.4 `<locale.h>` – přizpůsobení C národnímu prostředí 331

19.5 `<setjmp.h>` – umožnění běžně nedovolených skoků 332

19.6 `<signal.h>` – zpracování signálů 334

19.7 `<stdarg.h>` – práce s proměnným počtem parametrů 337

19.8 `<stdio.h>` – funkce pro vstup a výstup 337

19.9 `<stdlib.h>` – obecně užitečné funkce 339

19.9.1 Konverze řetězců na čísla 339

Funkce pro jednoduchý převod desítkových čísel 339

Funkce pro náročný převod čísel 339

19.9.2 Generátor pseudonáhodných čísel 340

19.9.3 Funkce pracující s dynamickou pamětí 341

19.9.4 Funkce pro spolupráci s operačním systémem 341

Funkce `abort()` 341

Funkce `atexit()` 341

Funkce `exit()` 342

Funkce `system()` 343

19.9.5 Funkce pro hledání a řazení 343

Řazení algoritmem *quick-sort* 343

Binární vyhledávání v seřazeném poli 344

19.10 `<string.h>` – zpracování řetězců 348

19.10.1 Funkce `strspn()` 348

19.10.2 Funkce `strcspn()` 349

19.10.3 Funkce `strpbrk()` 349

19.10.4 Funkce `strtok()` 349

19.10.5 Funkce `memchr()` 350

19.10.6 Funkce `memcmp()` 350

19.10.7 Funkce `memcpy()` 351

19.10.8 Funkce `memmove()` 351

19.10.9 Funkce `memset()` 351

19.11 `<time.h>` – práce s datem a časem 353

19.11.1 Funkce `clock()` 353

19.11.2 Měření doby běhu programu 354

19.11.3 Funkce `time()` 356

19.11.4 Funkce `difftime()` 356

19.11.5 Funkce `localtime()` 356
19.11.6 Funkce `asctime()` 357
19.11.7 Funkce `ctime()` 357
19.11.8 Funkce `strftime()` 358
19.11.9 Funkce `mktime()` 359

20 Funkce s proměnným počtem parametrů 362

20.1 Úvodní informace 362
20.2 Základní princip 363
20.3 Standardní hlavičkový soubor `<stdarg.h>` 365
20.4 Funkce `vprintf()`, `vfprintf()`, `vsprintf()` 366
20.5 Praktické příklady použití FPPP 367
20.5.1 Typické použití `vprintf()`, `vfprintf()`, `vsprintf()` 367
20.5.2 Přesměrování výstupu za běhu programu 368
20.5.3 Seznam parametrů je ukončený domluvenou hodnotou 370
20.5.4 Počet parametrů je předán v pevném parametru 372
20.5.5 Odstrašující případ zcela netypického použití 373

21 Ladění v jazyce C 376

21.1 Ladicí výpisy na `stderr` 378
21.2 Využití podmíněného překladu pomocí příkazů C-preprocesoru 378
21.2.1 Využití existence symbolické konstanty 379
21.2.2 Využití hodnoty symbolické konstanty 380
21.3 Self-testy modulů nebo funkcí 383
21.4 Využití makra ze souboru `<assert.h>` 385
21.4.1 Praktické použití 386
21.4.2 Zrušení aserce 389
21.4.3 Princip *aserce* 389
21.4.4 Chybné použití *aserce* 390
21.5 Prostředky pro ladění dynamických procesů 392

22 Minimální znalosti při programování pod UNIXem 396

22.1 Editor 396
22.2 Překladač a sestavovací program 397
22.3 Kontrola zdrojových textů v C 399
22.4 Odděleně překládané moduly 401
22.5 Ladicí prostředky 404

23 Změny v C podle nového standardu ISO 406
23.1 Stručný přehled norem a nejdůležitějších změn 406
23.2 Milé drobnosti 407
23.2.1 Komentáře do konce řádky 407
23.2.2 Definice kdekoliv v kódu 408
23.2.3 Co bylo zrušeno 408
23.3 Celočíselné datové typy 409
23.3.1 Logické hodnoty 409
23.3.2 Typ `long long int` 410
23.3.3 Celočíselné typy s přesně danou šířkou 410
23.3.4 Nové formátové specifikace 412
23.4 Okrajové hodnoty reálných čísel 413
23.4.1 Nekonečno 413
23.4.2 NaN 413
23.4.3 Denormální čísla 413
23.4.4 Podpora okrajových hodnot reálných čísel 413
23.5 Pole 414
23.5.1 Inicializace polí 414
23.5.2 Proměnná délka pole 415
23.5.3 Minimální velikost pole jako parametr funkce 416
23.5.4 Složené literály 416
23.6 Funkce 416
23.6.1 Identifikátor funkce 417
23.6.2 Funkce mohou být vnořené 418
23.6.3 Modifikátor `inline` 419
23.7 Vícebajtové znaky 420
23.7.1 Kódování a definice proměnných 421
23.7.2 Práce s jednotlivými znaky 421
23.7.3 Práce s řetězci 422
23.7.4 Vstupy a výstupy 423

24 Desatero přikázání pro programátory v C 425

Literatura 430

Rejstřík 431