

# Obsah

## **0 pastech a propastech ..... 15**

Druhé vydání .....	15
Co v této knize je a co v ní není .....	15
Ukázky .....	16
Jazyk C++ .....	17
Překladače .....	17
Jak tato kniha vznikla .....	17
Terminologie .....	18
Pasti, propasti a Tussmanův zákon .....	18
Poděkování .....	18

Kapitola 1

## **Zahřívací kolo ..... 19**

1.1 Příkazy .....	19
Příkaz switch .....	19
Zatoulané else .....	20
1.2 Operátory .....	21
Rovnost není přiřazení .....	21
Hodnota přiřazovacího výrazu .....	22
&& není &,    není   .....	22
Úplné vyhodnocení logického výrazu .....	23
Ne každý operátor má takovou prioritu, jakou bychom právě potřebovali ...	23
Matematická zkratka v Céčku? .....	24
Není dělení jako dělení .....	24
Asociativita operátorů .....	25
Pořadí, ve kterém se vyhodnocují operandy .....	25
Co se změní dříve? .....	26
Sekvenční body .....	27
1.3 Důsledky lexikální konvence .....	28
1.4 Syntaxe .....	29
Jen středník... .....	29
1.5 Funkce .....	31
Volání funkce .....	31
Prototyp? K čemu to je? .....	31

1.6 Sestavování a proměnné .....	32
Různé tváře jedné proměnné .....	32
Dvě proměnné se stejným jménem .....	33
1.7 Překlad a sestavení .....	34
Jeden soubor .....	34
Několik souborů .....	35
Projekt .....	36

## Kapitola 2

**Makra .....** **37**

2.1 Jedna závorka, žádná závorka .....	37
2.2 Jednou stačí .....	38
2.3 Makra mohou být zatraceně dlouhá .....	39
Může makro nahradit deklaraci typu? .....	39
2.4 Středník v makru .....	39
2.5 Makro jako příkaz: assert() .....	40
2.6 Aha, mezera .....	44
2.7 Preprocesor pracuje se symboly .....	45
2.8 Nedělejme z Céčka Pascal .....	45

## Kapitola 3

**Pole a ukazatele .....** **47**

3.1 Pole nejsou ukazatele a ukazatele nejsou pole .....	47
Ukazatele .....	47
Pole .....	48
Pole podrobněji .....	48
Jednorozměrná pole .....	48
Vícerozměrná pole .....	49
Ukazatel na ukazatel .....	50
Co je špatně v úvodním příkladu .....	51
Jak to mělo být správně .....	52
Jednou ukazatel, podruhé pole .....	52
U funkcí je to jedno .....	52
3.2 Pole .....	53
Počítáme prvky pole .....	53
Ukazatel na prvek za posledním .....	54

3.3 Ukazatele .....	54
Toulavé ukazatele .....	54
Ukazatel na objekt, který již neexistuje .....	55

## Kapitola 4

**Funkce v C++ .....** **59**

4.1 Předávání parametrů odkazem .....	59
Předávání ukazatelů .....	59
Pozor na pole .....	60
Řetězcová konstanta .....	60
Předávání odkazem v C++ .....	60
Kdy je předávání odkazem nezbytné .....	61
Abstraktní třída jako parametr .....	63
Reference na konstanty .....	63
4.2 Výsledek .....	64
Ukazatel na neexistující objekt .....	64
Vracíme výsledek odkazem .....	64
Odkaz na neexistující objekt .....	65
Dereferencovaný ukazatel .....	66
4.3 Jak vrátit z funkce řetězec .....	66
Doporučené způsoby .....	66
Nedoporučené způsoby .....	67
4.4 Standardní konverze .....	68
4.5 Výpustka .....	68
4.6 Funkce podle Kernighana a Ritchieho .....	69
4.7 Pryč s konstantou NULL .....	70
4.8 Kterou funkci vlastně voláme? .....	70
Shoda deklarací .....	71
Volání funkce .....	71
Standardní konverze nemusí být přenositelné .....	75

## Kapitola 5

**Vstupy a výstupy .....** **77**

5.1 Nejběžnější chyby na úvod .....	77
5.2 Čtení souboru .....	78
Binární soubor .....	78
Textový soubor .....	79

Čtení řetězců .....	81
<b>5.3 Znaky a jiná data .....</b>	<b>83</b>
getchar() vrací int .....	84
<b>5.4 Aktualizace souboru .....</b>	<b>85</b>
<b>5.5 Vlastní výstupní operátor .....</b>	<b>86</b>
Šířka pole .....	86
Definice vstupních a výstupních operátorů .....	88
Staré paměťové proudy .....	89
<b>5.6 Proč nepíše? .....</b>	<b>90</b>
Textové a binární soubory .....	90
Binární soubor .....	91
Textový soubor .....	91
Otevření souboru .....	92
Režim přepisování .....	92
Otevření souboru v nesprávném režimu .....	93
Formátovaný výstup v binárním režimu .....	94
Neformátovaný vstup a výstup v textovém režimu .....	94
<b>5.7 Konzola .....</b>	<b>95</b>
Přesměrování .....	95
Je konzola textový soubor? .....	95

## Kapitola 6

**Trocha počítání .....** **97**

<b>6.1 Problémy s celými čísly .....</b>	<b>97</b>
Celočíselné dělení .....	97
Celočíselné přetečení .....	98
Pokud víme, co děláme .....	101
Celočíselné konverze .....	102
Platí asociativní zákon? .....	103
Problémy s reálnými čísly .....	104
Knihovní funkce .....	107
Není arkustangens jako arkustangens .....	108

## Kapitola 7

**Zapouzdření, metody .....** **111**

<b>7.1 Zapouzdření má být vodotěsné .....</b>	<b>111</b>
Dočasné proměnné .....	113

7.2 Volání metod .....	114
Třída je oblast viditelnosti .....	115
Dominance .....	115
7.3 Metody, které volá překladač automaticky .....	117
Další automaticky volané metody .....	119
7.4 Konstantní objekty .....	119
Metody konstantních objektů .....	120
Dvě verze jedné metody .....	121
Když konstanty nejsou konstantní .....	121
Měnitelné složky konstant .....	122

## Kapitola 8

**Dědění .....** **123**

8.1 Přetypování mezi předkem a potomkem .....	123
8.2 Adresa objektu .....	124
Nula se nezmění .....	127
Přetypování v novém stylu .....	127
Operátor <code>dynamic_cast</code> .....	128
Shrnutí .....	134

## Kapitola 9

**Konstruktory a destruktory .....** **135**

9.1 Inicializace předků a atributů .....	135
Konstruktor by měl inicializovat všechny složky instance .....	135
Inicializační část konstruktoru .....	137
Konstruktory virtuálních předků .....	138
Inicializace nepřímého virtuálního předka .....	139
Explicitní volání jiného konstruktoru .....	141
9.2 Konstruktor volá konstruktory předků a destruktory volá destruktory předků .....	142
9.3 Kopírovací konstruktory .....	144
Parametry kopírovacího konstruktoru .....	144
Když nestačí implicitní kopírovací konstruktor .....	145
Přiřazovací operátor nenahradí kopírovací konstruktor .....	146
Implicitní volání konstruktoru .....	147
Deklarace s rovnítkem .....	150

## Kapitola 10

**Virtuální metody . . . . . 151**

10.1 Kdy použít virtuální metody . . . . .	151
Rozhraní musí být stejné . . . . .	153
Implicitní hodnoty parametrů . . . . .	154
10.2 Virtuální metody a vícenásobná dědičnost . . . . .	155
10.3 Virtuální destruktor . . . . .	157
10.4 Konstruktory, destruktory a virtuální metody . . . . .	158
10.5 Zbytečné virtuální metody . . . . .	160
10.6 Deklarace patří do hlavičkových souborů . . . . .	161
10.7 Čistě virtuální destruktor . . . . .	162
Jak se vyhnout pozdní vazbě . . . . .	163
10.8 Shrnutí . . . . .	163

## Kapitola 11

**Správa paměti . . . . . 165**

11.1 Operátory new a delete . . . . .	165
Operátor new nemusí uspět . . . . .	165
Napíšeme-li new, musíme napsat i delete . . . . .	166
delete musí mít své new . . . . .	166
Operátor new vrací ukazatel správného typu . . . . .	167
Zapomeňme na malloc() a free() . . . . .	167
Nekonečný cyklus v new . . . . .	167
Operátor delete[] . . . . .	168
11.2 Vývoj operátoru new . . . . .	169

## Kapitola 12

**Přetěžování operátorů . . . . . 171**

12.1 Pravidla slušného chování . . . . .	171
Aritmetické operátory +, -, *, / a % . . . . .	171
Přirazovací operátor = . . . . .	172
Globální operátor new . . . . .	172
Operátory &&,    a , . . . . .	173
Jen konvence, nic víc . . . . .	173
12.2 Zřetěžené operátory . . . . .	173
12.3 Kdy nevracet výsledek odkazem . . . . .	175
12.4 Přirazovací operátor = . . . . .	177

Když nestačí implicitní přiřazovací operátor . . . . .	177
Přiřazovací operátor se nedědí . . . . .	179
Přiřazovací operátor nemůže zastoupit kopírovací konstruktor . . . . .	179
Shodné objekty a přiřazování . . . . .	179
Přiřazovací operátor a výjimky . . . . .	182
Paměť jen tak okopírovaná . . . . .	185
Když nedodržíme pravidla slušného chování . . . . .	187
<b>12.5 Složené přiřazovací operátory . . . . .</b>	<b>187</b>
Když opravdu záleží na efektivitě . . . . .	189
<b>12.6 Operátory new a delete . . . . .</b>	<b>191</b>
Operátory new a delete lze předefinovat . . . . .	192
new má mít své delete . . . . .	194
Operátor new a pole . . . . .	195
Co vrací přetížený operátor new[]? . . . . .	196
Když new volá jiné new . . . . .	197
Které delete se zavolá? . . . . .	198
<b>12.7 Operátor -&gt; . . . . .</b>	<b>200</b>
<b>12.8 Operátory   , &amp;&amp; a , (čárka) . . . . .</b>	<b>200</b>
<b>12.9 Operátory ++ a -- . . . . .</b>	<b>201</b>

## Kapitola 13

**Výjimky . . . . . 205**

<b>13.1 Pozor na terminologii . . . . .</b>	<b>205</b>
<b>13.2 Výjimku je třeba zachytit . . . . .</b>	<b>205</b>
Výjimky a dědění . . . . .	205
Standardní třídy výjimek . . . . .	206
Žádné jiné konverze . . . . .	207
Černá díra na výjimky . . . . .	208
Výjimky chytáme odkazem . . . . .	209
Jednosměrná ulice . . . . .	210
Pošli to dál . . . . .	212
<b>13.3 Zdánlivé bezpečí . . . . .</b>	<b>213</b>
Únik prostředků při výjimce . . . . .	213
Automatické ukazatele a vlastnictví objektu . . . . .	215
Automatické ukazatele ve standardní knihovně . . . . .	217
Výjimky v destrukturu . . . . .	218
Funkce uncaught_exception() . . . . .	219
Výjimky v konstrukturu . . . . .	219
Výjimky v inicializační části konstrukturu . . . . .	221

13.4 Prvky v kontejneru .....	223
Třída Zasobník .....	224
V čem je problém? .....	225
Co s tím? .....	225
13.5 Závazky je třeba dodržovat .....	226
Specifikace výjimek a šablony .....	227
Funkce volaná z unexpected() .....	228
Výjimky by měly zůstat výjimečné .....	228
13.6 Test na závěr .....	230
Různé cesty .....	230

## Kapitola 14

**Prostory jmen .....** **233**

14.1 using .....	233
Direktiva using je tranzitivní .....	233
Deklarace using vnáší jméno do oboru .....	234
14.2 Koenigovo vyhledávání .....	235
Princip rozhraní .....	237

## Kapitola 15

**Šablony .....** **239**

15.1 Syntaktické chytáky .....	239
Lexikální nedorozumění .....	239
Šablonové parametry šablon .....	240
Dvojí čtení .....	241
Koenigovo vyhledávání .....	244
Kvalifikace this-> .....	244
Řetězec jako skutečný parametr šablony .....	245
15.2 Přátelé .....	246
Přátelé, které se nedaří najít .....	247
Souhrnný příklad .....	248
Definice spřátelené funkce v těle šablony třídy .....	248
15.3 Co se podle šablony nevytoří .....	250
15.4 Různé parametry, různé třídy .....	252
15.5 To není chyba .....	254
Základní typy .....	254
Nepodařené dosazení .....	255



## Kapitola 16

**Standardní knihovna jazyka C++ . . . . . 257**

16.1	Struktura knihoven jazyka C++ . . . . .	257
	Kontejnery . . . . .	258
	Iterátory . . . . .	258
	Algoritmy . . . . .	260
	Příklad: filtr SORT . . . . .	260
16.2	Vkládání prvků do kontejneru . . . . .	262
	Indexovat lze jen existující prvky . . . . .	262
	Efektivita . . . . .	262
	Požadavky na ukládané hodnoty . . . . .	263
	Typ uloženého prvku . . . . .	264
	Dynamicky alokované objekty je třeba zrušit . . . . .	266
	auto_ptr nepatří do kontejnerů . . . . .	267
16.3	Odstraňování prvků z kontejnerů . . . . .	267
	Vlastní odstraňování . . . . .	269
	Velikost kontejneru . . . . .	270
16.4	STL a dědění . . . . .	271
16.5	Diagnostika v STL . . . . .	272
	Může být hůř . . . . .	274

## Kapitola 17

**Chyby objektového návrhu . . . . . 275**

17.1	Nejběžnější problémy s děděním . . . . .	275
	Adaptér . . . . .	275
	„Technické“ dědění . . . . .	277
	Test je–má . . . . .	279
	„Logické“ dědění . . . . .	280
17.2	Kolik tříd potřebujeme? . . . . .	284
	Hejno much (mnoho drobných tříd) . . . . .	284
	Slepenec . . . . .	285
	Je ta datová složka nezbytná? . . . . .	287
17.3	Dědění a zapouzdření . . . . .	288
	Lepší návrh . . . . .	290
17.4	Shrnutí . . . . .	293

## Kapitola 18

**Závěrečné kolo . . . . . 295**

18.1 errno . . . . .	295
18.2 Proměnné operačního systému . . . . .	296
18.3 Optimalizace . . . . .	297
Optimalizace vrácené hodnoty . . . . .	297
18.4 Přetěžováním k efektivitě . . . . .	298
Inicializace globálních proměnných . . . . .	299
18.5 Není exit jako exit . . . . .	301

## Kapitola 19

**Složité deklarace . . . . . 305**

19.1 Začneme od začátku . . . . .	305
19.2 Jak je tedy čist . . . . .	306
19.3 Několik příkladů . . . . .	306
19.4 Označení typu . . . . .	308

## Kapitola 20

**Národní prostředí v C++ . . . . . 311**

20.1 O co jde . . . . .	311
Identifikátory . . . . .	312
Lokalizace a internacionalizace . . . . .	312
20.2 Lokální nastavení a jeho jméno . . . . .	313
Jméno lokálního nastavení . . . . .	313
Zjišťujeme jméno lokálního nastavení . . . . .	313
Používání lokálního nastavení . . . . .	314
20.3 Fazety lokálního nastavení . . . . .	315
Deklarace třídy locale . . . . .	315
Třída locale jako kontejner na fazety . . . . .	316
Přístup k fazetám . . . . .	317
20.4 Znaky národních abeced . . . . .	318
Úzké a široké proudy . . . . .	319
Výstup textu . . . . .	319
Vstup textu . . . . .	322
Konverze řetězců . . . . .	323
Neformátované proudy . . . . .	324

Jak zapsat Unicode do souboru	325
Unicode a malý a velký endián	325
Národní prostředí a datové proudy	326
<b>20.5 Znaky a jejich klasifikace</b>	<b>326</b>
Převod mezi malými a velkými písmeny	326
Klasifikace znaků	328
<b>20.6 Abecední řazení</b>	<b>328</b>
Třídění, nebo řazení?	328
Porovnávání slov	329
Abecední řazení v C++	331
Řadíme vektor řetězců podle abecedy	332
Není všechno zlato	335
Ještě jednou filtr SORT	335
<b>20.7 Formátování čísel</b>	<b>336</b>
Národní zvyklosti	336
Fazety pro formátování čísel	337
Implicitní nastavení	337
Fazeta numpunct	338
Skupiny číslic	339
<b>20.8 Vlastní fazeta</b>	<b>342</b>
Fazeta Obd_vv	342
O čem jsme nehovořili	346
Upozornění	346

## Kapitola 21

**Rozdíly mezi C a C++** ..... **349**

<b>21.1 Co v C++ chybí</b>	<b>349</b>
Deklarace funkce	349
Deklarace proměnných	351
Konstanty	352
<b>21.2 Datové typy</b>	<b>352</b>
Výčtové typy	352
Kompatibilní typy	353
Jméno použité v deklaraci typedef	353
<b>21.3 Struktura je obor viditelnosti</b>	<b>353</b>
<b>21.4 Otevřená pole</b>	<b>354</b>
<b>21.5 Typ void*</b>	<b>355</b>
<b>21.6 Kopírování nestálých objektů</b>	<b>355</b>

21.7 Když totéž neznamená totéž .....	356
Deklarace funkcí .....	356
Konstanty .....	356
Znaky .....	356
Jména struktur .....	357
Makra .....	357
Knihovní funkce .....	358
21.8 Kompatibilita .....	358
Virtuální metody .....	358
Destruktory .....	359
Přístupová práva .....	359
Operátory .....	360
Výjimky, dynamická identifikace typů .....	361
Šablony .....	361
21.9 Konverze .....	362
21.10 Novinky C99, které nejsou součástí C++ .....	363
21.11 ANSI? .....	363

**Literatura .....** **365**

**Rejstřík .....** **367**