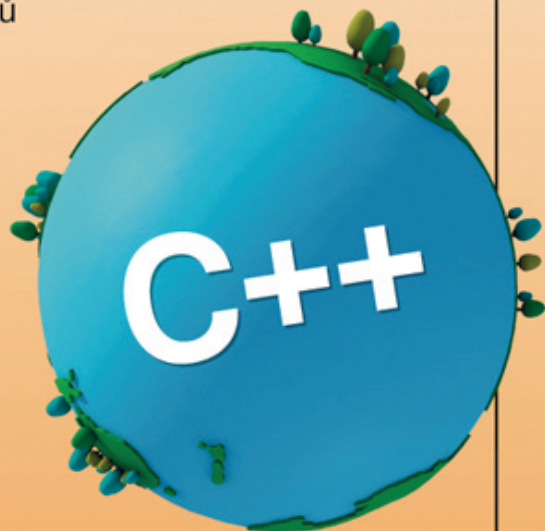


knihovna programátora

- Programovací jazyky C a C++ podle platných standardů
- Včetně připravovaného standardu C++0x
- Referenční přehled
- Příklady použití popisovaných konstrukcí



Jazyky

MIROSLAV VIRIUS

C a C++

kompletní průvodce – 2., aktualizované vydání

Upozornění pro čtenáře a uživatele této knihy

Všechna práva vyhrazena. Žádná část této tištěné či elektronické knihy nesmí být reprodukována a šířena v papírové, elektronické či jiné podobě bez předchozího písemného souhlasu nakladatele. Neoprávněné užití této knihy bude **trestně stíháno**.

Používání elektronické verze knihy je umožněno jen osobě, která ji legálně nabyla a jen pro její osobní a vnitřní potřeby v rozsahu stanoveném autorským zákonem. Elektronická kniha je datový soubor, který lze užívat pouze v takové formě, v jaké jej lze stáhnout s portálu. Jakékoliv neoprávněné užití elektronické knihy nebo její části, spočívající např. v kopírování, úpravách, prodeji, pronajímání, půjčování, sdělování veřejnosti nebo jakémkoliv druhu obchodování nebo neobchodního šíření je zakázáno! Zejména je zakázána jakákoliv konverze datového souboru nebo extrakce části nebo celého textu, umisťování textu na servery, ze kterých je možno tento soubor dále stahovat, přitom není rozhodující, kdo takovéto sdílení umožnil. Je zakázáno sdělování údajů o uživatelském účtu jiným osobám, zasahování do technických prostředků, které chrání elektronickou knihu, případně omezují rozsah jejího užití. Uživatel také není oprávněn jakkoliv testovat, zkoušet či obcházet technické zabezpečení elektronické knihy.





Copyright © Grada Publishing, a.s.

Upozornění pro čtenáře a uživatele této knihy

Všechna práva vyhrazena. Žádná část této tištěné či elektronické knihy nesmí být reprodukována a šířena v papírové, elektronické či jiné podobě bez předchozího písemného souhlasu nakladatele. Neoprávněné užití této knihy bude **trestně stíháno**.

Jazyky C a C++

kompletní průvodce – 2., aktualizované vydání

Miroslav Vírů

Vydala Grada Publishing, a.s.
U Průhonu 22, Praha 7
jako svou 4475. publikaci

Odpovědný redaktor Pavel Němeček
Sazba Tomáš Brejcha
Počet stran 368
První vydání, Praha 2011

© Grada Publishing, a.s., 2011

V knize použité názvy programových produktů, firem apod. mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.

Vytiskla Tiskárna PROTISK, s.r.o., České Budějovice

ISBN 978-80-247-3917-5 (tištěná verze)

ISBN 978-80-247-7417-6 (elektronická verze ve formátu PDF)

© Grada Publishing, a.s. 2012

Předmluva	17
-----------------	----

1.

Úvod

1.1 První program	19
1.1.1 Co je co	20
1.1.2 Překlad a sestavení	23
1.2 Programovací jazyky C a C++	25
1.2.1 Standardy	26
1.3 Objektově orientované programování	27
1.3.1 Základní pojmy OOP	28
1.3.2 Některé další pojmy	30

2.

Základní pojmy

2.1 Popis jazyků C a C++	31
2.2 Množina znaků	32
2.2.1 Univerzální jména znaků	33
2.3 Identifikátor	34
2.3.1 Oblast platnosti, oblast viditelnosti	35
2.4 Klíčová slova	35
2.5 Lexikální konvence a zápis programu	37
2.6 Průběh překladu	37
2.6.1 Průběh překladu podrobně	38
2.6.2 Pozorovatelné chování programu	39
2.7 Definice a deklarace	40
2.7.1 Deklarace	40
2.7.2 Definice	40
2.7.3 Pravidlo jediné definice	41
2.8 L-hodnota a r-hodnota	42
2.8.1 L-hodnota, r-hodnota a další (C++0X)	43
2.9 Zarovnání	43
2.10 Běh programu	44
2.10.1 Inicializace globálních proměnných	44
2.10.2 Ukončení programu	45

3.

Základní datové typy

3.1 Celá čísla	47
3.1.1 Celočíselné literály	49
3.1.2 Další celočíselné typy v C99 a v C++0x	50
3.2 Znakové typy	51
3.2.1 Znakové literály	52
3.3 Logické hodnoty	53
3.3.1 Typ bool (C++)	53
3.3.2 Typ _Bool (C99)	54
3.4 Operace s celými čísly	54
3.4.1 Přiřazování	54
3.4.2 Aritmetické operace	54
3.4.3 Relace	56
3.4.4 Logické operace	56
3.4.5 Bitové operace	56
3.4.6 Další operace	58
3.5 Reálná čísla	58
3.5.1 Reálné typy v C99	59
3.5.2 Reálné literály	59
3.5.3 Operace s reálnými čísly	60
3.6 Komplexní čísla (jen C99)	61
3.6.1 Operace s komplexními čísly	62
3.6.2 Přiřazování	62
3.6.3 Aritmetické operace	62
3.6.4 Logické operace	62
3.6.5 Další operace s komplexními čísly	63
3.7 Typ void	63

4.

Výčtové typy, struktury a unie

4.1 Výčtové typy	65
4.1.1 Deklarace výčtového typu	65
4.1.2 Použití výčtového typu	68
4.1.3 Rozsah výčtového typu	68
4.1.4 Operace s výčtovými typy	69
4.1.5 Přetěžování operátorů	70
4.2 Struktury	70
4.2.1 Deklarace struktury	70
4.2.2 Složky struktur	72
4.2.3 Inicializace	73

4.2.4	Bitová pole	74
4.2.5	Otevřená pole	75
4.2.6	Literály typu struktura	75
4.2.7	Přetěžování operátorů	76
4.3	Unie	76
4.3.1	Deklarace unie	76
4.3.2	Složky unií	77
4.3.3	Inicializace unií	78
4.3.4	Literály typu unie	78
4.3.5	Anonymní unie (jen C++)	78
4.3.6	Přetěžování operátorů	79

5.

Ukazatele, pole a reference

5.1	Pole	81
5.1.1	Jednorozměrná pole	81
5.1.2	Inicializace	82
5.1.3	Použití polí	83
5.1.4	Vícerozměrná pole	85
5.1.5	Předávání pole jako parametru funkce	86
5.1.6	Literály typu pole (C99)	87
5.2	Ukazatele	87
5.2.1	Inicializace ukazatelů	89
5.2.2	Dereferencování	90
5.2.3	Dynamické přidělování a navrácení paměti	91
5.2.4	Uvolňování dynamicky alokované paměti	94
5.2.5	Aritmetické operace s ukazateli	95
5.2.6	Ukazatele na funkce	96
5.2.7	Ukazatele na statické prvky tříd	97
5.2.8	Restringované ukazatele v C99	97
5.3	Reference (jen v C++)	99
5.3.1	Reference na funkce	101
5.3.2	„Konstantní“ reference	101
5.3.3	Reference na r-hodnotu (C++0x)	101

6.

Proměnné a deklaráce

6.1	Syntax deklaráce	103
6.1.1	Mnemotechnické uspořádání deklaráce	103
6.1.2	Popis deklaráce	103
6.1.3	Význam základních tvarů deklarátoru	104

6.1.4	Automatické odvození typu (C++0x)	105
6.1.5	Specifikace decltype (C++0x)	106
6.1.6	Označení typu	106
6.1.7	Deklarace nového jména typu	106
6.2	Paměťové třídy	107
6.2.1	Specifikátory paměťových tříd	107
6.2.2	Automatické proměnné (paměťová třída auto)	107
6.2.3	Registrové proměnné (paměťová třída register)	108
6.2.4	Statické proměnné (paměťová třída static)	108
6.2.5	Externí proměnné (paměťová třída extern)	109
6.2.6	Měnitelné složky konstant (paměťová třída mutable)	109
6.2.7	Proměnné lokální v podprocesu (paměťová třída thread_local)	109
6.3	Jiné specifikátory	109
6.3.1	Cv-modifikátory	109
6.3.2	Volací konvence	112
6.3.3	Další modifikátory	112
6.4	Atributy (C++0x)	113
6.4.1	Specifikace zarovnání	113
6.4.2	Další atributy	113
6.5	Doba života, oblast platnosti a viditelnost	114
6.5.1	Oblast platnosti identifikátoru	114
6.5.2	Viditelnost identifikátoru	115
6.5.3	Doba života proměnné	116
6.6	Rozdělení identifikátorů	116
6.6.1	Jazyk C	116
6.6.2	Jazyk C++	117
6.7	Deklarace asm	118
6.8	Deklarace static_assert (C++0x)	118

7.

Jmenné prostory

7.1	Deklarace jmenného prostoru	119
7.2	Přejmenování prostoru jmen	120
7.3	using	121
7.3.1	Direktiva using	121
7.3.2	Deklarace using	123
7.4	Koenigovo vyhledávání	124
7.4.1	K čemu to je	125
7.4.2	Koenigovo vyhledávání a šablony	126

8.

Operátory a výrazy

8.1 Výraz	127
8.1.1 Konstantní výraz	127
8.2 Přehled operátorů	129
8.2.1 Priorita a asociativita	131
8.2.2 Pořadí vyhodnocování operandů	131
8.3 Konverze	131
8.3.1 Celočíselná a reálná rozšíření	132
8.3.2 Obvyklé aritmetické konverze	135
8.3.3 Konverze číselných typů	135
8.3.4 Konverze ukazatelů	136
8.3.5 Standardní konverze	137
8.4 Popis jednotlivých operátorů	137
8.4.1 Operátory pro přístup k datům	137
8.4.2 Aritmetické operátory	144
8.4.3 Inkrementace a dekrementace ++,-	146
8.4.4 Relační operátory	147
8.4.5 Bitové operace	150
8.4.6 Logické operátory	153
8.4.7 Přiřazovací operátory	156
8.4.8 Alokace a uvolňování paměti	157
8.4.9 Přetypování	162
8.4.10 Dynamická identifikace typu typeid	170
8.4.11 Operátor sizeof	172
8.4.12 Zjištění adresy &	173
8.4.13 Podmínkový operátor (podmíněný výraz) ?:	174
8.4.14 Operátor čárka ,	176
8.4.15 Operátor throw	177
8.4.16 Operátor decltype (C++0x)	178

9.

Příkazy

9.1 Jednoduché příkazy	179
9.1.1 Výrazový příkaz	179
9.1.2 Prázdný příkaz	180
9.1.3 Deklarace jako příkaz	180
9.2 Složený příkaz (blok)	180
9.3 Větení programu	180
9.3.1 Podmíněný příkaz if	180
9.3.2 Příkaz switch	182

9.4	Cykly	185
9.4.1	Příkaz while	186
9.4.2	Příkaz for	187
9.4.3	Cyklus do-while	188
9.4.4	Příkaz cyklu pro procházení kontejneru (jen C++0x)	189
9.5	Přenos řízení	190
9.5.1	Příkaz continue	190
9.5.2	Příkaz break	191
9.5.3	Příkaz return	191
9.5.4	Příkaz goto a návěští	192
9.5.5	Příkaz throw	193
9.5.6	Příkaz __leave	193
9.5.7	Ukončení programu pomocí funkce exit()	193
9.5.8	Dlouhý skok pomocí longjmp()	193
9.6	Příkaz asm	193

10.

Funkce

10.1	Definice a deklarace funkce	195
10.1.1	Definice funkce	195
10.1.2	Deklarace funkce	198
10.1.3	Parametry funkce	199
10.1.4	Lokální proměnné	200
10.1.5	Paměťová třída funkcí	200
10.1.6	Modifikátor inline	200
10.1.7	Zastaralý způsob deklarace	201
10.1.8	Spolupráce C s C++	202
10.1.9	Konstantní funkce (C++0x)	203
10.2	Výměna dat mezi funkcemi	203
10.2.1	Vracená hodnota	204
10.2.2	Parametry	206
10.2.3	Funkce s proměnným počtem parametrů	211
10.2.4	Globální proměnné	213
10.2.5	Statické proměnné	213
10.3	Volací konvence	213
10.3.1	Volací konvence jazyka C	214
10.3.2	Volací konvence jazyka Pascal	214
10.3.4	Standardní konvence	214
10.3.5	Registrová konvence	215
10.4	Funkce main()	215

10.4.1	Parametry funkce main()	215
10.5	Rekurze a rezie volání funkcí	216
10.5.1	Rekurzivní volání funkce	216
10.5.2	Rezie volání funkce	217
10.6	Přetěžování funkcí	217
10.6.1	Přetěžování obyčejných funkcí	218
10.6.2	Přetěžování metod (členských funkcí)	218
10.6.3	Která přetížená funkce se zavolá?	219
10.6.4	Přetěžování, překrytí, zastínění	220
10.7	Lambda-výrazy (C++0x)	220
10.7.1	Deklarace lambda-výrazu	221
10.7.2	Záchyt	222
10.8	Modulární programování a funkce	224

11.

Třídy a objekty

11.1	Deklarace třídy	225
11.1.1	Specifikace přístupových práv	226
11.1.2	Třídy a OOP	227
11.2	Datové složky	229
11.2.1	Nestatické datové složky	229
11.2.2	Statické datové složky	230
11.3	Členské funkce (metody)	232
11.3.1	Nestatické členské funkce	232
11.3.2	Statické členské funkce	234
11.3.3	Definice metody uvnitř třídy	235
11.3.4	Definice vně definice třídy	235
11.4	Ukazatel this	236
11.5	Přístup ke složkám tříd	237
11.5.1	Přístup zevnitř třídy	237
11.5.2	Přístup z vnějšku třídy	238
11.5.3	Spřátelené funkce	238
11.6	Dědění	240
11.6.1	Předkové	240
11.6.2	Přístupová práva pro zděděné složky	241
11.6.3	Přetěžování a zastínění	242
11.6.4	Virtuální dědění	243
11.7	Polymorfismus	245
11.7.1	Časná a pozdní vazba	245

11.7.2	Virtuální metody (C++03)	246
11.7.3	Virtuální destruktor	248
11.7.4	Abstraktní třídy, čistě virtuální metody	248
11.8	Upřesnění v deklaraci třídy (C++0x)	249
11.8.1	Třídy, od nichž nelze odvozovat potomky	250
11.8.2	Explicitní deklarace překrytí a zastínění	250
11.9	Zvláštní metody	251
11.9.1	Konstruktory	251
11.9.2	Kopírovací konstruktor	256
11.9.3	Dědění konstruktorů (C++0x)	258
11.9.4	Konstruktor pro konstantní výrazy (C++0x)	259
11.9.5	Volání jiného konstruktoru téže třídy (C++0x)	259
11.9.6	Destrukory	260
11.9.7	Pořadí volání konstruktorů a destruktorů	260
11.9.8	Volání virtuálních metod z konstruktorů a destruktorů ...	261
11.10	Vytváření instancí	262
11.10.1	Konstantní a nestálé instance	262
11.10.2	Pole instancí	262
11.11	Lokální třídy	263
11.12	Vnořené typy	263
11.12.1	Vnořené třídy	264
11.13	Ukazatele na instance	265
11.14	Struktury a unie	266
11.14.1	Struktury	266
11.14.2	Unie	267
11.15	Třídní ukazatele	267
11.15.1	Ukazatel na datovou složku	267
11.15.2	Ukazatel na metodu	269
11.15.3	Ukazatele na statické složky	271

12.

Přetěžování operátorů

12.1	Základní pravidla	273
12.1.1	Omezení	273
12.2	Operátory, které lze přetěžovat jako metody i jako volné funkce	274
12.2.1	Přetěžování unárních operátorů	274
12.2.2	Přetěžování binárních operátorů	276
12.3	Operátory, které lze přetěžovat jen jako metody	277

12.3.1	Přetěžování operátoru volání funkce	277
12.3.2	Přetěžování přiřazovacího operátoru	278
12.3.3	Přetěžování operátoru indexování	280
12.3.4	Přetěžování operátoru ->	281
12.3.5	Operátor přetypování (konverzní funkce)	281
12.4	Operátory pro práci s pamětí	282
12.4.1	Co můžeme změnit	282
12.4.2	Přetěžování operátoru new	283
12.4.3	Přetěžování operátoru delete	285
12.4.4	Operátory new a delete a výjimky	286
12.5	Uživatелеm definované literály (C++0x)	288
12.5.1	Parametry literálového operátoru	288
12.5.2	Surové a hotové literály	289

13.

Výjimky

13.1	Proč výjimky?	291
13.1.1	Oč jde	292
13.2	Klasické řešení v C: dlouhý skok	292
13.2.1	Použití dlouhého skoku	292
13.3	Výjimky v C++	294
13.3.1	Schéma použití výjimek	294
13.3.2	Co se děje	295
13.3.3	Částečné ošetření	297
13.3.4	Výjimky a funkce	297
13.3.5	Neošetřené a neočekávané výjimky	299
13.3.6	Standardní třídy výjimek	300
13.3.7	Výjimky a alokace paměti	303
13.4	Strukturované výjimky v jazyce C	303
13.4.1	Schéma použití SEH	304
13.4.2	Co se děje	305
13.4.3	Obsluha a filtr	305
13.4.4	Vznik strukturovaných výjimek	306
13.4.5	Filtr	307
13.4.6	Nepokračovatelné výjimky	308
13.4.7	Koncovka bloku	308
13.4.8	Neošetřené výjimky	309

14.

Šablony

14.1 Deklarace šablony	313
14.1.1 Instance šablony	314
14.2 Parametry šablon	314
14.2.1 Typové parametry	315
14.2.2 Hodnotové parametry	315
14.2.3 Šablonové parametry	316
14.3 Šablony volných funkcí	316
14.3.1 Vytváření instancí	317
14.3.2 Explicitní (úplná) specializace	318
14.3.3 Přetěžování šablon volných funkcí	319
14.4 Šablony objektových typů	322
14.4.1 Šablony metod	323
14.4.2 Šablony statických datových složek	324
14.4.3 Vnořené šablony	325
14.4.4 Vytváření instancí	327
14.4.5 Specializace	327
14.4.6 Přátelé	329
14.4.7 Dědění	333
14.5 Organizace programu	334
14.5.1 Exportní šablony (C++03)	334
14.5.2 Externí šablony (C++0x)	335
14.6 Šablony s proměnným počtem parametrů (C++0x)	335
14.6.1 Parametry variadické šablony	335
14.6.2 Rozvoj balíku parametrů	335
14.7 Alias	339
14.8 Různá omezení	339

15.

Dynamická identifikace typů

15.1 Určení typu za běhu	341
15.1.1 Operátor typeid	341
15.1.2 Třída typeid	342
15.1.3 Operátor dynamic_cast	342
15.2 Příklady užití RTTI	342
15.2.1 Rozhodování podle typu	342
15.2.2 Ladění	343
15.2.3 Příslušnost k hierarchii	343

16.

Preprocesor

16.1 Úvod	345
16.2 Direktivy preprocesoru	347
16.2.1 Prázdná direktiva	347
16.2.2 Vkládání souborů	347
16.2.3 Makra	348
16.2.4 Zrušení definice makra	352
16.2.5 Podmíněný překlad	352
16.2.6 Vyvolání chyby	355
16.2.7 Číslování řádků	355
16.2.8 Direktiva závislá na implementaci	355
16.3 Předdefinovaná makra	357
Literatura	359
Rejstřík	361

Předmluva

Otevřeli jste knihu, která vám poskytne referenční příručku programovacích jazyků C a C++ podle platných standardů, a to včetně připravovaného nového standardu jazyka C++ označovaného zatím C++0x.

Co v této knize najdete

V úvodní kapitole najdete příklad jednoduchého programu, základní informace o vytváření zdrojového textu programu a o postupu při překladu. Tato kapitola se poněkud vymyká z rámce celé knihy, neboť spíše než referenční příručku připomíná první kapitolu učebnice. Jejím cílem je poskytnout rámec, který by mu usnadnil pochopení dalších kapitol i čtenářům, kteří s jazyky C a C++ nemají žádnou zkušenost, nebo ji mají jen velmi malou. V závěru první kapitoly najdete také několik slov o historii těchto jazyků, o jejich mezinárodních standardech, a také velice stručný výklad základních pojmů objektově orientovaného programování se zřetelem k C a C++.

Ve druhé kapitole se seznámíte se způsobem popisu jazyků C a C++ a se základními stavebními prvky, jako je množina znaků, identifikátory, klíčová slova atd. Následující kapitoly popisují základní datové typy, uživatelem definované neobjektové typy, výrazy, příkazy, operátory atd. – ostatně to najdete v obsahu. Každý významový celek začíná zpravidla popisem syntaxe, za nímž následuje stručné vysvětlení významu a podstatné informace shrnuté do bodů a doplněné příklady. Tento postup ale nemělo smysl dodržovat vždy.

Převážnou většinu příkladů jsem odzkoušel na současných překladačích jazyků C a C++ na PC. V několika případech jsem převzal příklady přímo ze standardů [1] a [3]; zpravidla se jednalo o rysy jazyka, které běžné současné překladače ještě neimplementují nebo je neimplementují v souladu se standardy. (Může vám připadat podivné, že překladače jazyka C z roku 2010 neimplementují v plném rozsahu novinky standardu [3] tohoto jazyka z roku 1999, ale je to tak; pro tvůrce většiny překladačů je ovšem důležitější kompatibilita s jazykem C++ – a ještě standard [1] jazyka C++ z roku 2003 byl založen na standardu [2] jazyka C z roku 1990. Většiny příkladů týkajících se C++0x jsem převzal z návrhu nového standardu [4]; některé z nich bylo možno vyzkoušet v současných překladačích, ovšem zdaleka ne všechny. Poznamenejme, že dokument [4], z něhož jsem čerpal, je označen jako *konečný návrh standardu*.)

Jak tato kniha vznikla

V roce 1999 vydalo nakladatelství Grada Publishing knihu *Programovací jazyky C a C++ podle normy ANSI/ISO – kompletní kapselní průvodce* (D. Louis, P. Mejzlík, M. Virius), která shrnovala oba jazyky i jejich knihovny do jediného svazku. V době, kdy jsme ji psali, tedy v letech 1997–1998, ovšem nebylo ještě k dispozici konečné znění prvního standardu jazyka C++, vydané v září 1998, ani nový standard jazyka C, vydaný v r. 1999. Neobsahovala tedy řadu důležitých informací. Také příklady byly přizpůsobeny nejrozšířenějším překladačům té doby – a ty se od standardu v mnoha ohledech odchylovaly. Ostatně ani struktura knihy nebyla právě šťastná, neboť byla přizpůsobena striktním požadavkům edice (10 kapitol, co nejvíce „postupů“ apod.).

Proto jsem se v po dohodě s nakladatelstvím Grada Publishing rozhodl tuto knihu od základu přepracovat. Z původního díla jsem převzal některé příklady a části textu, avšak i ty jsem přizpůsobil novým standardům obou jazyků. Tato verze vyšla v nakladatelství Grada Publishing pod názvem *Jazyky C a C++ – Kompletní kapesní průvodce programátora* (M. Víríus, 2005).

Na začátku roku 2011 mne nakladatelství Grada požádalo, abych tuto knihu připravil k novému vydání. Protože koncem roku 2011 má vyjít nový standard C++ s řadou zásadních novinek, rozhodl jsem se k dalšímu přepracování celé knihy, aby obsahovala oba jazyky podle standardů, jež budou platit v době, kdy vyjde. Výsledek držíte v ruce.

Typografické a jiné konvence

Domnívám se, že nemá smysl zdůrazňovat, že *kurziva* znamená zdůraznění a *neproporcionální písmo* že používám k zápisu ukázek zdrojového textu, identifikátorů, klíčových slov apod. a výstupu počítače. To pozná každý průměrně inteligentní čtenář na první pohled (a programátoři bývají více než průměrně inteligentní).

Způsob a význam popisu *syntaktických konstrukcí*, který v této knize používám, je vysvětlen v kapitole 2.1 na str. 31.

[1] V hranatých závorkách jsou uvedeny odkazy na seznam literatury (str. 359).

C90 Pro jazyk C podle normy ISO 9899:1990 [2] používám zkratku C90.

C99 Pro jazyk C podle normy ISO 9899:1999 [3], která nahradila standard [2], používám zkratku C99.

C++03 Pro jazyk C++ podle standardu ISO 14882:2003 [1] používám zkratku C++03.

C++0x Pro jazyk C++ podle připravovaného standardu ISO, jehož vydání se očekává koncem roku 2011, používám zkratku C++0x.

Poděkování

Chci poděkovat všem, kteří mi při přípravě této knihy pomohli svými radami a připomínkami.

Přes veškerou péči, kterou jsem této knize věnoval, se v ní mohou vyskytnout chyby. Pokud na nějakou přijdete, dejte mi prosím vědět; na webové stránce <http://tjn.fjfi.cvut.cz/~virius/errata.htm> uveřejním opravu.

Miroslav Víríus
miroslav.virius@fffi.cvut.cz

1.

Úvod

Je zvykem, že knihy o programovacím jazyce C nebo C++ začínají příkladem, který vypíše nějaký vtipný text a skončí. Pak následuje stručné vysvětlení jeho významu, informace o způsobu překladač apod. Autor si tím vytvoří rámec, který mu umožní předvádět příklady na spustitelných programech. I když v referenční příručce nic takového není nezbytné, přidržme se této tradice, abychom usnadnili její používání i čtenářům, kteří tyto jazyky znají jen povrchně a potřebují je rychle začít používat.

1.1 První program

V tomto oddílu si ukážeme první programy v jazyce C++ a v C, seznámíme se s pravidly pro vytváření zdrojového textu, s postupem překladač atd. Protože některé z těchto věcí mohou do značné míry záviset na použitém překladači, uvedeme pouze základní informace; podrobnosti musíte hledat v dokumentaci ke svému překladači.

Zdrojový text našeho prvního programu v C++ je jednoduchý:

```
/* Soubor PrvniPlus.CPP
   První program v C++
*/
#include <iostream>           // 4
using namespace std;        // 5
int main() {                 // 6
    cout << "Hello, World << endl"; // 7
    return 0;                // 8
}                             // 9
```

Podobný program v jazyce C může vypadat takto:

```
/* Soubor Prvni.c
   První program v jazyce C
*/
#include <stdio.h>
int main() {
    printf("Hello, World\n"); /* 6 */
    return 0;
}
```

Tento program uložíme do textového souboru. Pro programy v jazyce C++ se typicky používá přípona `.cpp`, pro programy v jazyce C přípona `.c`. Lze se ale setkat i s jinými – např. v některých unixových systémech se pro programy v C++ používají přípony `.C` nebo `.CC`. Podobně jako v jiných programovacích jazycích, i v C a v C++ platí, že tento soubor nesmí obsahovat formátování, tedy např. velikost a řez písma apod. Zpravidla jde o soubory v kódování ASCII (nebo v jiném jednobajtovém kódování, které váš počítač používá), dnešní překladače však už zpravidla akceptují zdrojové soubory v kódování Unicode.

Jestliže tento program přeložíme a spustíme, vypíše

```
■ Hello, World
```

Než se ale pustíme do překladu a sestavování tohoto programu, vysvětlíme si (bez nároku na úplnost) význam jeho jednotlivých částí.

1.1.1 Co je co

Pokud výslovně nezdůrazním něco jiného, týká se výklad jak C, tak C++. Veškeré pojmy z jazyků C a C++, které si zde naznačíme, budou znovu vysvětleny v referenční části této knihy. Začneme nepochybně nejnápadnější součástí zdrojového textu – komentářem.

Komentář

První tři řádky v obou programech představují *víceřádkový komentář*. Ten začíná dvojicí znaků `/*` (zapsaných bezprostředně zas sebou) a končí dvojicí znaků `*/`, opět zapsaných bezprostředně zas sebou. Překladač ignoruje tyto znaky a vše mezi nimi.

Vedle toho máme v C++ a v C99 k dispozici jednořádkový komentář. Ten začíná dvojicí lomítek, `//`, a končí na konci řádku.

Jednořádkové komentáře jsme v prvním výpisu použili k očíslování některých řádků.

Struktura programu a funkce `main()`

Programy v jazycích C a C++ jsou tvořeny posloupností deklarácí. To zní možná podivně, ale v principu to znamená, že se skládají z deklarácí funkcí,¹ typů a proměnných. Zjednodušeně řečeno, právě jedna z těchto funkcí musí mít jméno `main` a program začne prvním příkazem této funkce. Po skončení funkce `main()`² program skončí.

Podívejme se na první z výše uvedených výpisů.

Deklarace funkce `main()` začíná v prvním výpisu na 6. řádku. Nejprve je uvedena specifikace typu výsledku (zde je to `int`, tedy celé číslo se znaménkem), pak následuje identifikátor funkce a za ním prázdné závorky, které říkají, že tato funkce nemá žádné parametry. Tyto závorky jsou nezbytné.

Pak následuje otevírací složená závorka `{`, již začíná tělo funkce. Toto tělo končí uzavírací složenou závorkou `}` na 9. řádku. Tělo funkce tvoří příkazy uzavřené mezi těmito závorkami.

¹ V C a v C++ se nerozlišuje mezi funkcí a procedurou; jakýkoli podprogram se nazývá „funkce“.

² Jméno (identifikátor) této funkce je `main`. V textu knihy budeme k identifikátorům funkcí připisovat závorky, aby bylo na první pohled jasné, že se jedná o funkci. Bude-li to potřebné, uvedeme v závorkách i typ parametrů.

Řetězcová konstanta

V 7. řádku souboru `PrvniPlus.CPP` najdeme zápis `"Hello, World"`. To je řetězcová konstanta – posloupnost znaků uzavřená mezi uvozovky.

Výstup v C++

Výstup je jazycích C a C++ je realizován pomocí funkcí nebo objektů, které se nacházejí ve standardní knihovně. Tzv. *standardní výstupní proud* (typicky přeměrovatelný výstup na konzolu) je v C++ představován objektem `cout`. Vystupující hodnoty do tohoto proudu vkládáme pomocí operátoru `<<`. Zápisem

```
■ cout << "Hello, World" << endl;
```

do tohoto proudu vkládáme postupně dvě věci: zaprvé řetězcovou konstantu představující vypisovaný text a zadruhé tzv. manipulátor `endl`, který způsobí přechod na nový řádek.

Výstup v C

Podívejme se nyní na druhý výpis, tj. na soubor `Prvni.c`. Jazyk C používá pro vstupní a výstupní operace knihovní funkce. Jednou z nich je funkce `printf()`, která vypíše zadaný znakový řetězec do standardního výstupu. Tuto funkci lze volat i s více parametry.

Hlavičkové soubory

Ani v C, ani v C++ *nesmíme použít nic, co překladač nezná*. To platí i pro součásti standardních knihoven – překladač je nezná, dokud mu o nich neřekneme. Proto je třeba překladači předem oznámit, že budeme používat objekt `cout`, resp. funkci `printf()`, které jsou definovány ve standardní knihovně. Abychom však nemuseli jejich deklarace vypisovat do svých programů ručně, jsou připraveny v tzv. hlavičkových souborech. Tyto hlavičkové soubory vkládáme do svých programů tzv. direktivou preprocesoru `#include`, tedy např. zápisem

```
■ #include <iostream>
```

Tuto direktivu musíme zapsat na samostatný řádek. Jméno souboru uzavřeme mezi lomené závorky tvořené znaky „menší než“ a „větší než“. Poznamenejme, že standardní hlavičkové soubory v jazyce C++ nemají žádnou příponu.

Hlavičkový soubor `<iostream>`³ obsahuje základní nástroje pro vstupní a výstupní operace.

V jazyce C mají standardní hlavičkové soubory typicky příponu `.h`. Soubor `<stdio.h>` obsahuje nástroje jazyka C pro vstupní a výstupní operace.

Poznamenejme, že standardní knihovnu jazyka C můžeme používat i v C++. To znamená, že uvedený program můžeme přeložit i překladačem C++. Překladače C++ mají vlastní verzi hlavičkových souborů z jazyka C. Jejich obsah je prakticky stejný jako v C90, jazyk C++ však umožňuje používat je dvojím způsobem:

³ Jméno tohoto souboru je `iostream`. Lomené závorky připojujeme, abychom zdůraznili, že jde o hlavičkový soubor. S touto praxí se běžně setkáváme nejen v odborné literatuře, ale i v textu standardu těchto jazyků.