

Siddhartha Rao

PREVOD
DEVETOG IZDANJA

POKRIVA
C++20 i
C++23

NAUČITE SAMOSTALNO

C++

jedna **lekcija** DNEVNO

C++

PREVOD DEVETOG IZDANJA

jedna **lekcija** DNEVNO

SKORO 300 UZORAKA KODA ZA KOMPJILIRANJE

Učeći jednu lekciju dnevno, možete da steknete sve veštine koje su vam potrebne za početak programiranja u C++ jeziku. Ovaj kompletan vodič će vam pomoći da brzo ovladate osnovama objektno-orijentisanog programiranja, kao i naprednim funkcijama i konceptima jezika C++. Potpuno ažurirana za standard C++20, ova praktična knjiga je dizajnirana da vam pomogne da napišete C++ kod koji je brži, jednostavniji i pouzdaniji.

- Naučite osnove C++ jezika i objektno-orijentisanog programiranja
- Koristite Standard Template Library (STL) da brzo razvijete moćnije i pouzdanije aplikacije
- Naučite moderne funkcije C++20 verzije, kao što su koncepti, opsezi, prikazi, adapteri i moduli
- Saznajte koje bi dokazane akcije trebalo da koristite, a koje ne, da biste iskoristili najbolju praksu i izbegli zamke, od prvog dana
- Testirajte svoje znanje i stručnost vežbama nakon svake lekcije
- Naučite da koristite skoro 300 uzoraka koda za kompajliranje koji su dostupni za preuzimanje besplatno i koji su detaljno objašnjeni u knjizi

Siddhartha Rao je potpredsednik u SAP SE-u, zadužen za Security Response za vodeći svetski trgovinski softver. Evolucija C++-a ubedila ga je da je sada moguće programiranje bržih, jednostavnijih i moćnijih aplikacija nego ikada do sada.

Kategorija: Programiranje

Pokriva: C++23, C++20 C++17, C++14, C++11

Nivo: Početni-Srednji

Učite kada imate vremena, tempom koji vama odgovara

- Nije potrebno iskustvo u programiranju: ubrzo ćete umeti da pišete dobro organizovane, efikasne C++ programe!
- Ovladajte objektno-orijentisanim konceptima kao što su klase, nasleđivanje, polimorfizam, enkapsulacija i apstrakcija
- Kreirajte pouzdane programe bogate funkcijama pomoću STL klasa, kontejnera i algoritama
- Pojednostavite svoj kod pomoću automatskog utvrđivanja tipa i drugih funkcija
- Programirajte objekte funkcija pomoću modernih C++ Lambda izraza
- Ubrzajte učenje korišćenjem skoro 300 uzoraka koda objašnjenih u knjizi
- Iskoristite nove C++20 koncepte, opsege, prikaze, adaptere i module
- Pregledajte poboljšanja koja se očekuju u verziji C++23

Siddhartha Rao

NAUČITE SAMOSTALNO

C++

PREVOD DEVETOG IZDANJA

jedna lekcija DNEVNO

 kompiuter
biblioteka

SAMS



Izdavač:



Obalskih radnika 15, Beograd

Tel: 011/2520272

e-mail: kombib@gmail.com

internet: www.kombib.rs

Urednik: Mihailo J. Šolajić

Za izdavača, direktor:

Mihailo J. Šolajić

Autor: Siddhartha Rao

Prevod: Slavica Prudkov

Lektura: Nemanja Lukić

Slog : Zvonko Aleksić

Znak Kompjuter biblioteke:

Miloš Milosavljević

Štampa: „Pekograf“, Zemun

Tiraž: 500

Godina izdanja: 2022.

Broj knjige: 555

Izdanje: Prvod IX izdanja

ISBN: 978-86-7310-578-9

Sams Teach Yourself C++ in One Hour a Day

by Siddhartha Rao

ISBN: 978-0-13-733468-1

Copyright © 2022 by Pearson Education, Inc.

All right reserved. No part of this book may be reproduced or transmitted in any form or by means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from the Publisher.

Autorizovani prevod sa engleskog jezika edicije u izdanju „Pearson Education, Inc“, Copyright © 2022.

Sva prava zadržana. Nije dozvoljeno da nijedan deo ove knjige bude reprodukovano ili snimljeno na bilo koji način ili bilo kojim sredstvom, elektronskim ili mehaničkim, uključujući fotokopiranje, snimanje ili drugi sistem presnimavanja informacija, bez dozvole izdavača.

Zaštitni znaci

Kompjuter Biblioteka i „Pearson Education, Inc“ su pokušali da u ovoj knjizi razgraniče sve zaštitne oznake od opisnih termina, prateći stil isticanja oznaka velikim slovima.

Autor i izdavač su učinili velike napore u pripremi ove knjige, čiji je sadržaj zasnovan na poslednjem (dostupnom) izdanju softvera. Delovi rukopisa su možda zasnovani na predizdanju softvera dobijenog od strane proizvođača. Autor i izdavač ne daju nikakve garancije u pogledu kompletnosti ili tačnosti navoda iz ove knjige, niti prihvataju ikakvu odgovornost za performanse ili gubitke, odnosno oštećenja nastala kao direktna ili indirektna posledica korišćenja informacija iz ove knjige.

O autoru

Siddhartha Rao je potpredsednik SAP SE-a, zadužen za bezbednost proizvoda. SAP SE je najpouzdaniji svetski dobavljač poslovnog softvera i cloud servisa. Softverski inženjer u srcu, Siddhartha je ubeđen da je brzi razvoj programskog jezika C++ pokrenuo ovo doba mašinskog učenja i veštačke inteligencije. Funkcije predstavljene u verziji C++20 omogućavaju da programirate jednostavnije, ali moćnije aplikacije nego ikada ranije. Prilikom pisanja ove knjige Siddhartha se pobrinuo da skoro 300 primera kompajliranja koda bude praćeno detaljnim analizama načina na koji funkcionišu.

Siddhartha bi želeo da čuje vaše komentare, recenzije i povratne informacije!

Kratak sadržaj

DEO I: OSNOVE

LEKCIJA 1

Početak rada 1

LEKCIJA 2

Anatomija C++ programa 13

LEKCIJA 3

Upotreba promenljivih i deklarisanje konstanti 27

LEKCIJA 4

Upravljanje nizovima i znakovnim nizovima..... 59

LEKCIJA 5

Upotreba izraza, iskaza i operatora 81

LEKCIJA 6

Kontrolisanje toka programa 111

LEKCIJA 7

Organizovanje koda pomoću funkcija 151

LEKCIJA 8

Objašnjenje pokazivača i referenci..... 179

DEO II: OSNOVE OBJEKTNO-ORIJENTISANOG C++ PROGRAMIRANJA**LEKCIJA 9****Klase i objekti219****LEKCIJA 10****Implementiranje nasleđivanja277****LEKCIJA 11****Polimorfizam311****LEKCIJA 12****Tipovi operatora i preklapanje operatora341****LEKCIJA 13****Operatori za eksplicitnu konverziju387****LEKCIJA 14****Uvod u makroe i šablone401****DEO III: UČENJE O BIBLIOTECI STANDARD TEMPLATE LIBRARY (STL)A****LEKCIJA 15****Standard Template Library431****LEKCIJA 16****STL klasa string443****LEKCIJA 17****STL klase dinamičkog niza465****LEKCIJA 18****STL klase list i forward_list487****LEKCIJA 19****STL klase set i multiset507****LEKCIJA 20****STL klase map i multimap525****DEO IV: LAMBDA IZRAZI I STL ALGORITMI****LEKCIJA 21****Razumevanje objekata funkcije549**

LEKCIJA 22	
Lambda izrazi	565
LEKCIJA 23	
STL algoritmi	579
LEKCIJA 24	
Prilagodljivi kontejneri: stack i queue	615
LEKCIJA 25	
Upotreba bit indikatora pomoću biblioteke STL	633
DEO V: NAPREDNI C++ KONCEPTI	
LEKCIJA 26	
Razumevanje pametnih pokazivača	645
LEKCIJA 27	
Upotreba tokova podataka za ulaz i izlaz	661
LEKCIJA 28	
Rukovanje izuzecima	683
LEKCIJA 29	
C++20 koncepti, opsezi, prikazi i adapteri.....	699
LEKCIJA 30	
C++20 programske niti	717
LEKCIJA 31	
C++20 moduli i C++23	725
DEO VI: DODACI	
DODATAK A	
Upotreba brojeva: binarni i heksadecimalni	733
DODATAK B	
C++ ključne reči	739
DODATAK C	
Pisanje odličnog C++ koda	741

DODATAK D	
ASCII kodovi.....	745
DODATAK E	
Odgovori	751
INDEKS	751

Sadržaj

DEO I: OSNOVE

LEKCIJA 1

Početak rada

.....	1
Kratka istorija C++ jezika	1
Povezanost sa C jezikom	1
Prednosti C++ jezika	2
Evolucija C++ standarda	2
Ko koristi programe napisane C++ jezikom?	3
Programiranje C++ aplikacije.....	3
Koraci za izgradnju izvršnog fajla.....	3
Analiziranje i ispravljanje grešaka	4
Integrirana razvojna okruženja.....	4
Programiranje prve C++ aplikacije	5
Izgradnja i izvršavanje prve C++ aplikacije	7
Razumevanje grešaka kompajlera	9
Šta je novo u verziji C++20?.....	10
Rezime	10
Pitanja i odgovori	10
Radionica	11
Kviz	11
Vežbe.....	12

LEKCIJA 2**Anatomija C++ programa**

.....	13
Delovi programa Hello World	13
Pretprocesorska komanda #include	14
Telo programa: main()	15
Vraćanje vrednosti	16
Koncept imenskih prostora	17
Komentari u C++ kodu	19
Funkcije u C++ jeziku	20
Osnovni unos pomoću iskaza std::cin i ispis pomoću iskaza std::cout.....	22
Rezime	24
Pitanja i odgovori	25
Radionica	25
Kviz.....	25
Vežbe.....	26

LEKCIJA 3**Upotreba promenljivih i deklarisanje konstanti27**

Šta je promenljiva?	27
Ukratko o memoriji i adresiranju	27
Deklarisanje promenljivih za pristup i upotrebu memorije	28
Deklarisanje i inicijalizacija više promenljivih datog tipa	30
Razumevanje opsega promenljive	30
Globalne promenljive	32
Konvencije imenovanja	34
Uobičajeni C++ tipovi promenljivih koje kompajler podržava.....	35
Upotreba tipa bool za skladištenje logičkih (Boolean) vrednosti	36
Upotreba tipa char za skladištenje vrednosti karaktera	37
Koncept označenih i neoznačenih celih brojeva	37
Tipovi označenog celog broja short, int, long i long long	38
Izbegavanje grešaka prekoračenja selektovanjem odgovarajućih tipova podataka ..	39
Tipovi sa pokretnim zarezom float i double	41
Određivanje veličine promenljive pomoću operatora sizeof()	42
Izbegavanje grešaka konverzije skraćivanjem pomoću inicijalizacije liste	43
Automatsko utvrđivanje tipa pomoću ključne reči auto	44
Upotreba ključne reči typedef za zamenu tipa promenljive	46
Šta je konstanta?	46
Literalne konstante	47
Deklarisanje promenljivih kao konstanti pomoću ključne reči const	47
Izrazi konstante pomoću ključne reči constexpr	49
C++20 direktne funkcije pomoću ključne reči consteval.....	49
Nabrajanja	51
Nabrajanja u opsegu.....	53
Definisanje konstanti pomoću komande #define	54
Ključne reči koje ne možete da koristite kao nazive promenljivih ili konstanti	55

Rezime	56
Pitanja i odgovori	56
Radionica	58
Kviz	58
Vežbe	58

LEKCIJA 4

Upravljanje nizovima i znakovnim nizovima.....59

Šta je niz?	59
Potreba za nizovima	59
Deklarisanje i inicijalizacija statičkih nizova	60
Kako su podaci uskladišteni u nizu	61
Pristup podacima uskladištenim u nizu	63
Modifikovanje podataka uskladištenih u nizu	64
Višedimenzionalni nizovi	67
Deklarisanje i inicijalizacija višedimenzionalnih nizova	67
Pristup elementima u višedimenzionalnom nizu	68
Dinamički nizovi	70
Znakovni nizovi C stila	72
C++ znakovni nizovi: Upotreba klase std::string	75
Rezime	77
Pitanja i odgovori	78
Radionica	79
Kviz	79
Vežbe	79

LEKCIJA 5

Upotreba izraza, iskaza i operatora.....81

Iskazi	81
Složeni iskazi ili blokovi	83
Upotreba operatora	83
Operator dodele (=)	83
Razumevanje l-vrednosti i r-vrednosti	83
Operatori za sabiranje (+), oduzimanje (-), množenje (*), deljenje (/) i modulo (%) operaciju	84
Operatori za povećanje (++) i smanjivanje (--)	85
Upotrebiti sufiks ili prefiks?	85
Operatori jednakosti (==) i (!=)	87
Relacioni operatori (poređenja)	88
C++20 operator trosmernog poređenja (<=>)	90
Logičke operacije NOT, AND, OR i XOR	92
Upotreba C++ logičkih operatora NOT (!), AND (&&) i OR ()	94
Operatori nad bitovima NOT (~), AND (&), OR () i XOR (^)	98
Operatori nad bitovima pomeranja udesno (>>) i pomeranja ulevo (<<)	100
Složeni operatori dodele	102
Upotreba operatora sizeof() za određivanje memorije koju promenljiva zauzima ...	104
Prioritet operatora i asocijativnost	106

Rezime	109
Pitanja i odgovori	109
Radionica	110
Kviz	110
Vežbe	110

LEKCIJA 6

Kontrolisanje toka programa 111

Uslovno izvršenje upotrebom iskaza if ... else	111
Uslovno programiranje upotrebom iskaza if ... else	112
Uslovno izvršavanje iskaza unutar bloka	115
Ugnežđeni iskazi if	117
Uslovna obrada pomoću iskaza switch-case	121
Uslovno izvršavanje pomoću operatora ?:	124
Izvršenje koda u petljama	126
Osnovna petlja upotrebom iskaza goto	126
Petlja while	128
Petlja do ... while	130
Petlja for	132
Petlja for zasnovana na rasponu	135
Modifikovanje ponašanja petlje pomoću iskaza continue i break	137
Petlje koje se ne završavaju: beskonačne petlje	138
Kontrolisanje beskonačnih petlji	139
Programiranje ugnežđenih petlji	142
Upotreba ugnežđenih petlji za prelaz višedimenzionalnog niza	143
Upotreba ugnežđenih petlji za izračunavanje Fibonačijevih brojeva	145
Rezime	146
Pitanja i odgovori	147
Radionica	148
Kviz	148
Vežbe	148

LEKCIJA 7

Organizovanje koda pomoću funkcija 151

Potreba za funkcijama	151
Šta je prototip funkcije?	153
Šta je definicija funkcije?	154
Šta je poziv funkcije i šta su argumenti?	154
Programiranje funkcije pomoću više parametara	155
Programiranje funkcija bez parametara ili bez vraćenih vrednosti	157
Parametri funkcije sa podrazumevanim vrednostima	158
Rekurzija – funkcije koje same sebe pozivaju	160
Funkcije sa više vraćenih iskaza	162
Upotreba funkcija za korišćenje različitih formi podataka	164
Preklapanje funkcija	164
Prosleđivanje niza vrednosti u funkciju	166
Prosleđivanje argumenata pomoću reference	167

Kako mikroprocesor rukuje pozivima funkcija	169
Umetnute funkcije	171
Automatsko utvrđivanje vraćenog tipa	173
Lambda funkcije	174
Rezime	176
Pitanja i odgovori	176
Radionica	177
Kviz	177
Vežbe	178

LEKCIJA 8

Objašnjenje pokazivača i referenci 179

Šta je pokazivač?	179
Deklarisanje pokazivača	180
Određivanje adrese promenljive pomoću operatora za referenciranje (&).....	181
Upotreba pokazivača za čuvanje adresa	182
Pristup pokazanim podacima pomoću operatora za dereferenciranje (*)	185
Šta je veličina pokazivača?	188
Dinamička dodela memorije	189
Upotreba operatora new i delete za dinamičku dodelu i za otpuštanje memorije	190
Efekat operatora inkrementiranja (++) i dekrementiranja (--) na pokazivače	194
Upotreba ključne reči const u pokazivačima	197
Prosleđivanje pokazivača funkcijama	198
Sličnosti između nizova i pokazivača	199
Uobičajene programerske greške kada se koriste pokazivači	202
Čurenje memorije	203
Kada pokazivači pokazuju na nevalidne lokacije memorije	203
Neupareni pokazivači (takođe ih nazivamo lutajući ili divlji pokazivači)	205
Provera da li je uspešan zahtev dodele pomoću operatora new	206
Najbolja praksa programiranja pokazivača	209
Šta je referenca?	210
Šta reference čini korisnim?	211
Upotreba ključne reči const u referencama	213
Prosleđivanje argumenata u funkcije pomoću reference.....	213
Rezime	215
Pitanja i odgovori	215
Radionica	217
Kviz	217
Vežbe	217

DEO II: OSNOVE OBJEKTNO-ORIJENTISANOG C++ PROGRAMIRANJA

LEKCIJA 9

Klase i objekti219

Koncept klasa i objekata	219
Deklarisanje klase	220

Objekat kao instanca klase	221
Pristupanje članovima pomoću operatora tačke (.)	222
Pristupanje članovima pomoću operatora pokazivača (->)	222
Ključne reči public i private	224
Apstrakcija podataka pomoću ključne reči private	226
Konstruktori	228
Deklarisanje i implementiranje konstruktora	228
Kada i kako upotrebiti konstruktore	229
Preklapanje konstruktora	231
Klasa bez podrazumevanog konstruktora	233
Parametri konstruktora sa podrazumevanim vrednostima	234
Konstruktori sa listama inicijalizacije	236
Destruktor	238
Deklarisanje i implementiranje destruktora	238
Kada i kako upotrebiti destruktor	239
Konstruktor kopiranja	241
Površno kopiranje i povezani problemi	242
Obezbeđivanje dubinskog kopiranja pomoću konstruktora za kopiranje	244
Upotreba konstruktora za pomeranje za poboljšanje performansi	249
Različiti oblici primene konstruktora i destruktora	250
Klasa koja ne dozvoljava kopiranje	250
Singleton klasa koja dozvoljava jednu instancu	251
Klasa koja zabranjuje instanciranje u steku	254
Upotreba konstruktora za konvertovanje tipova	256
Pokazivač this	259
Upotreba operatora sizeof() u klasi	260
Razlika između ključnih reči struct i class	262
Deklarisanje ključne reči friend za class	263
Unija, specijalan mehanizam za skladištenje podataka	265
Deklarisanje unije	265
Gde se upotrebljava unija	266
Upotreba grupne inicijalizacije u klasama i strukturama	269
Ključna reč constexpr u klasama i objektima	272
Rezime	273
Pitanja i odgovori	273
Radionica	275
Kviz	275
Vežbe	275

LEKCIJA 10

Implementiranje nasleđivanja277

Osnove nasleđivanja	277
Nasleđivanje i izvođenje	278
C++ sintaksa za izvođenje	280
Ključna reč specifikatora pristupa – protected	282
Inicijalizacija osnovne klase – prosleđivanje parametara u osnovnu klasu	285
Izvedena klasa redefiniše metode osnovne klase	287
Pozivanje redefinisanih metoda osnovne klase	290

Pozivanje metoda osnovne klase u izvedenoj klasi	290
Izvedena klasa skriva metode osnovne klase	292
Redosled konstrukcije	294
Redosled destrukcije	295
Privatno nasleđivanje	297
Zaštićeno (protected) nasleđivanje	300
Problem isecanja	303
Višestruko nasleđivanje	304
Izbegavanje nasleđivanja pomoću ključne reči final.....	307
Rezime	308
Pitanja i odgovori	309
Radionica	309
Kviz	309
Vežbe	310

LEKCIJA 11

Polimorfizam311

Osnove polimorfizma	311
Potreba za polimorfnim ponašanjem	311
Polimorfno ponašanje implementirano pomoću virtuelnih funkcija	313
Potreba za virtuelnim destruktorima	316
Kako funkcionišu virtuelne funkcije? Razumevanje tabele virtuelne funkcije	320
Apstraktne osnovne klase i potpuno virtuelne funkcije	324
Upotreba virtuelnog nasleđivanja za rešavanje Diamond problema	327
Specifikator override za ukazivanje namere menjanja vrednosti	331
Upotreba specifikatora final za sprečavanje menjanja vrednosti funkcije	332
Virtuelni konstruktori za kopiranje?	333
Rezime	337
Pitanja i odgovori	337
Radionica	338
Kviz	339
Vežbe	339

LEKCIJA 12

Tipovi operatora i preklapanje operatora341

Šta su operatori u jeziku C++?	341
Unarni operatori	342
Unarni operatori za inkrementiranje (++) i dekrementiranje (--)... ..	343
Operatori za konverziju tipova	346
Operatori za dereferenciranje (*) i selekciju člana (->)	350
Binarni operatori	352
Binarni operatori sabiranja (a+b) i oduzimanja (a-b)	353
Operatori dodele sabiranja (+=) i dodele oduzimanja (-=).....	356
Operatori jednakosti (==) i nejednakosti (!=).....	358
Operatori <, >, <= i >=	361
C++20 Trosmerni operator poređenja (<=>).....	363
Operator dodele kopiranja (=)	366

Operator indeksiranja ([])	370
Operator funkcije ()	373
Konstruktor pomeranja i operator dodele pomeranja za programiranje visoke performanse	374
Problem neželjenih koraka kopiranja	374
Deklarisanje konstruktora za pomeranje i operatora dodele pomeranja	375
Korisnički definisani literali	380
Operatori koji ne mogu da budu preklopljeni	382
Rezime	383
Pitanja i odgovori	384
Radionica	384
Kviz	385
Vežbe	385

LEKCIJA 13

Operatori za eksplicitnu konverziju387

Potreba za eksplicitnom konverzijom	387
Zašto eksplicitna konverzija C-stila nije popularna za neke C++ programere	388
C++ operatori eksplicitne konverzije	388
Upotreba operatora static_cast	389
Upotreba operatora dynamic_cast i identifikacije tipa u vreme pokretanja	390
Upotreba operatora reinterpret_cast	394
Upotreba operatora const_cast	394
Problemi sa C++ operatorima eksplicitne konverzije	396
Rezime	397
Pitanja i odgovori	397
Radionica	398
Kviz	398
Vežbe	399

LEKCIJA 14

Uvod u makroe i šablone401

Pretprocesor i kompajler	401
Upotreba makro ključne reči #define za definisanje konstanti	402
Upotreba makroa za zaštitu protiv višestrukih uključenja	404
Upotreba ključne reči #define za pisanje makro funkcija	406
Čemu služe zagrade?	408
Upotreba makroa assert za validaciju izraza	409
Prednosti i mane upotrebe makro funkcija	410
Uvod u šablone	411
Sintaksa za deklarisanje šablona	411
Različiti tipovi deklaracija šablona	412
Funkcije šablona	412
Šabloni i bezbednost tipa	415
Klase šablona	415
Deklarisanje šablona pomoću više parametara	416
Deklarisanje šablona pomoću podrazumevanih parametara	417

Primer klase šablona: HoldsPair	417
Instanciranje i specijalizacija šablona	419
Klase šablona i statički članovi	421
Šabloni promenljivih	423
Upotreba tvrdnje static_assert za izvršenje provere u vreme kompajliranja	427
Upotreba šablona u praktičnom C++ programiranju.....	428
Rezime	429
Pitanja i odgovori	429
Radionica	430
Kviz	430
Vežbe	430

DEO III: UČENJE O BIBLIOTECI STANDARD TEMPLATE LIBRARY (STL/A)

LEKCIJA 15

Standard Template Library431

STL kontejneri	431
Sekvencijalni kontejneri	431
Asocijativni kontejneri	432
Adapteri kontejnera	434
STL iteratori	434
STL algoritmi	435
Interakcija između kontejnera i algoritama pomoću iteratora	436
Upotreba ključne reči auto za omogućavanje kompajleru da definiše tip	438
Biranje odgovarajućeg kontejnera	438
STL klase znakovnih nizova	441
Rezime	441
Pitanja i odgovori	441
Radionica	442
Kviz	442

LEKCIJA 16

STL klasa string443

Potreba za klasama za manipulaciju znakovnim nizovima	443
Upotreba STL klase string	445
Instanciranje STL klase string i kreiranje kopija	445
Pristupanje sadržajima karaktera klase std::string	448
Nadovezivanje jednog znakovnog niza na drugi	450
Pronalaženje karaktera ili podniza u znakovnom nizu	451
Skraćivanje STL znakovnog niza	453
Obrtanje znakovnog niza	455
Konverzija veličine slova znakovnog niza	457
Implementacija STL znakovnog niza zasnovana na šablonu	458
Operator ""s u klasi std::string	459
Upotreba klase std::string_view (izmenjena u verziji C++20)	460
Rezime	462
Pitanja i odgovori	462

Radionica	463
Kviz	463
Vežbe	463

LEKCIJA 17

STL klase dinamičkog niza465

Karakteristike klase <code>std::vector</code>	465
Tipične operacije klase <code>vector</code>	466
Instanciranje klase <code>vector</code>	466
Ubacivanje elemenata na kraj vektora pomoću funkcije <code>push_back()</code>	468
Inicijalizacija liste	469
Ubacivanje elemenata na određenu poziciju pomoću funkcije <code>insert()</code>	469
Pristupanje elementima u klasi <code>vector</code> pomoću semantike niza	472
Pristupanje elementima u vektoru pomoću semantike pokazivača	474
Uklanjanje elemenata iz klase <code>vector</code>	475
Razumevanje konceptata veličine i kapaciteta	477
STL klasa <code>deque</code>	479
Rezime	483
Pitanja i odgovori	483
Radionica	484
Kviz	484
Vežbe	485

LEKCIJA 18

STL klase `list` i `forward_list`487

Karakteristike klase <code>std::list</code>	487
Osnovne operacije klase <code>list</code>	488
Instanciranje objekta <code>std::list</code>	488
Ubacivanje elemenata na početak ili na kraj klase <code>list</code>	490
Ubacivanje elemenata na sredinu klase <code>list</code>	491
Brisanje elemenata iz klase <code>list</code>	494
Obrtanje i sortiranje elemenata u klasi <code>list</code>	496
Obrtanje elemenata pomoću funkcije <code>list::reverse()</code>	496
Sortiranje elemenata	497
Sortiranje i uklanjanje elemenata iz liste koja sadrži instance klase	499
Klasa <code>std::forward_list</code>	502
Rezime	505
Pitanja i odgovori	505
Radionica	506
Kviz	506
Vežbe	506

LEKCIJA 19

STL klase `set` i `multiset`507

Uvod u STL klase <code>set</code>	507
Osnovne operacije STL klasa <code>set</code> i <code>multiset</code>	508

Instanciranje objekta <code>std::set</code>	508
Ubacivanje elemenata u klasu <code>set</code> ili <code>multiset</code>	510
Pronalaženje elemenata u STL klasama <code>set</code> ili <code>multiset</code>	512
Brisanje elemenata u STL klasama <code>set</code> ili <code>multiset</code>	514
Prednosti i mane upotrebe STL klasa <code>set</code> i <code>multiset</code>	519
STL Hash Set implementacija klasa <code>std::unordered_set</code> i <code>std::unordered_multiset</code>	519
Rezime	523
Pitanja i odgovori	523
Radionica	524
Kviz	524
Vežbe	524

LEKCIJA 20

STL klase `map` i `multimap`525

Uvod u STL klase <code>map</code>	525
Osnovne operacije klasa <code>std::map</code> i <code>std::multimap</code>	526
Instanciranje klase <code>std::map</code> ili <code>std::multimap</code>	527
Ubacivanje elemenata u STL klasu <code>map</code> ili <code>multimap</code>	529
Pronalaženje elemenata u STL kontejneru <code>map</code>	531
Pronalaženje elemenata u STL klasi <code>multimap</code>	534
Brisanje elemenata iz STL klasa <code>map</code> ili <code>multimap</code>	535
Unošenje prilagođenih predikata za sortiranje	537
STL-ov kontejner ključa-vrednosti zasnovan na heš tabeli	541
Kako funkcionise heš tabela	541
Upotreba klasa <code>unordered_map</code> i <code>unordered_multimap</code>	542
Rezime	546
Pitanja i odgovori	546
Radionica	547
Kviz	548
Vežbe	548

DEO IV: LAMBDA IZRAZI I STL ALGORITMI

LEKCIJA 21

Razumevanje objekata funkcije549

Objekti funkcije i predikati	549
Tipični primeri primene objekata funkcije	550
Unarne funkcije.....	550
Unarni predikati.....	554
Binarne funkcije	557
Binarni predikati	559
Rezime	562
Pitanja i odgovori	562
Radionica	562
Kviz	563
Vežbe	563

LEKCIJA 22**Lambda izrazi565**

Šta je lambda izraz?	565
Kako se definiše lambda izraz	566
Preuzimanje promenljivih	566
Parametri.....	567
Tipovi rezultata	567
Lambda izraz za unarnu funkciju	568
Lambda izraz za unarni predikat	570
Lambda izraz sa stanjem - upotrebljena lista preuzimanja [...].....	571
Lambda izraz za binarnu funkciju	573
Lambda izraz za binarni predikat.....	575
Rezime	577
Pitanja i odgovori	577
Radionica	578
Kviz.....	578
Vežbe.....	578

LEKCIJA 23**STL algoritmi579**

Šta su STL algoritmi?	579
Klasifikacija STL algoritama	580
Nemutirajući algoritmi.....	580
Mutirajući algoritmi	581
Upotreba STL algoritama	583
Pronalaženje elemenata pomoću vrednosti ili uslova	583
Brojanje elemenata pomoću vrednosti ili uslova	585
Pretraživanje elementa ili raspona u kolekciji	587
Inicijalizacija elemenata u kontejneru na specifične vrednosti	589
Upotreba algoritma <code>std::generate()</code> za inicijalizaciju elemenata na vrednosti generisane u vreme pokretanja	591
Obrada elemenata u rasponu pomoću algoritma <code>for_each()</code>	592
Izvršavanje transformacija u rasponu pomoću algoritma <code>std::transform()</code>	595
Operacije kopiranja i uklanjanja elemenata	597
Zamena vrednosti i zamena elementa u skladu sa uslovom	600
Sortiranje i pretraživanje u sortiranoj kolekciji i brisanje duplikata	601
Particionisanje raspona	604
Ubacivanje elemenata u sortiranu kolekciju.....	606
Izvršavanje operacije preloma pomoću algoritma <code>std::accumulate()</code> u verziji C++20	608
Ograničeni algoritmi verzije C++20.....	609
Rezime	611
Pitanja i odgovori	612
Radionica	613
Kviz.....	613
Vežbe.....	613

LEKCIJA 24**Prilagodljivi kontejneri: stack i queue615**

Karakteristike ponašanja stekova i redova čekanja	615
Stek	615
Red čekanja	616
Upotreba STL klase stack	616
Instanciranje klase stack	617
Funkcije članovi klase stack	618
Ubacivanje elemenata na početak i uklanjanje sa početka pomoću funkcija push() i pop().....	619
Upotreba STL klase queue	620
Instanciranje klase queue	621
Funkcije članovi klase queue	622
Ubacivanje elemenata na kraj i uklanjanje sa početka klase queue pomoću funkcija push() i pop()	623
Upotreba STL klase priority_queue	625
Instanciranje klase priority_queue	625
Funkcije članovi klase priority_queue	627
Ubacivanje elemenata na kraj i njihovo uklanjanje sa početka klase priority_queue pomoću funkcija push() i pop()	627
Rezime	630
Pitanja i odgovori	630
Radionica	631
Kviz.....	631
Vežbe.....	631

LEKCIJA 25**Upotreba bit indikatora pomoću biblioteke STL633**

Klasa bitset	633
Instanciranje klase std::bitset	633
Upotreba klase std::bitset i njenih članova	635
Korisni operatori u klasi std::bitset	635
Metodi članovi klase std::bitset	636
Klasa vector<bool>	639
Instanciranje klase vector<bool>.....	639
Funkcije i operatori klase vector<bool>.....	640
Rezime	642
Pitanja i odgovori	642
Radionica	643
Kviz.....	643
Vežbe.....	643

DEO V: NAPREDNI C++ KONCEPTI

LEKCIJA 26

Razumevanje pametnih pokazivača	645
Šta su pametni pokazivači?	645
Problem sa upotrebom konvencionalnih (neobrađenih) pokazivača	645
Kako pomažu pametni pokazivači?	646
Kako su implementirani pametni pokazivači?	647
Tipovi pametnih pokazivača	648
Dubinsko kopiranje	649
Mehanizam generisanja kopije po upisivanju.....	651
Pametni pokazivači nabrojane reference.....	651
Pametni pokazivači povezane reference	652
Destruktivno kopiranje	652
Upotreba klase <code>std::unique_ptr</code>	655
Popularne biblioteke pametnih pokazivača.....	658
Rezime	658
Pitanja i odgovori	658
Radionica	659
Kviz.....	659
Vežbe.....	659

LEKCIJA 27

Upotreba tokova podataka za ulaz i izlaz	661
Koncept tokova podataka	661
Važne C++ klase i objekti toka podataka	662
Upotreba klase <code>std::cout</code> za pisanje formatiranih podataka u konzolu	664
Menjanje prikaza formata broja pomoću klase <code>std::cout</code>	664
Poravnanje teksta i podešavanje širine polja pomoću klase <code>std::cout</code>	667
Upotreba klase <code>std::cin</code> za unos	668
Upotreba klase <code>std::cin</code> za unos u klasičan tip podataka	668
Upotreba klase <code>std::cin::get</code> za unos u <code>char*</code> bafer	669
Upotreba klase <code>std::cin</code> za unos u klasu <code>std::string</code>	670
Upotreba klase <code>std::fstream</code> za rukovanje fajlom	672
Otvaranje i zatvaranje fajla pomoću metoda <code>open()</code> i <code>close()</code>	672
Kreiranje i pisanje tekstualnog fajla pomoću funkcije <code>open()</code> i operatora <code><<</code>	674
Čitanje tekstualnog fajla pomoću funkcije <code>open()</code> i operatora <code>>></code>	675
Pisanje u binarni fajl i čitanje iz njega	677
Upotreba klase <code>std::stringstream</code> za konverzije znakovnog niza	678
Rezime	680
Pitanja i odgovori	680
Radionica	681
Kviz.....	681
Vežbe.....	681

LEKCIJA 28**Rukovanje izuzecima683**

Šta je izuzetak?	683
Šta izaziva izuzetke?	684
Implementiranje bezbednosti izuzetka pomoću ključnih reči try i catch	684
Upotreba funkcije catch(...) za rukovanje svim izuzecima	685
Hvatanje izuzetka tipa	686
Generisanje izuzetka tipa pomoću ključne reči throw	687
Kako funkcioniše rukovanje izuzecima	689
Klasa std::exception	691
Prilagođena klasa izuzetka izvedena iz klase std::exception	692
Rezime	695
Pitanja i odgovori	695
Radionica	696
Kviz	696
Vežbe	696

LEKCIJA 29**C++20 koncepti, opsezi, prikazi i adapteri.....699**

Koncepti	700
Definisanje prilagođenih koncepata pomoću ključne reči requires	703
Upotreba koncepata sa klasama i objektima	705
Biblioteka opsega, prikazi i adapteri	708
Prikazi i adapteri	709
Adapteri koje obezbeđuje biblioteka opsega	711
Kombinovanje više adaptera	713
Rezime	715
Pitanja i odgovori	715
Radionica	716
Kviz	716
Vežbe	716

LEKCIJA 30**C++20 programske niti717**

Višenitni rad	717
Šta je programska nit?	717
Zašto je potrebno programirati višenitne aplikacije?	718
Upotreba biblioteke programskih niti verzije C++20	718
Kako programske niti mogu da prenose podatke?	721
Upotreba muteksa i semafora za sinhronizovanje programskih niti	722
Rezime	722
Pitanja i odgovori	723
Radionica	723
Vežbe	723

LEKCIJA 31

C++20 moduli i C++23	725
Moduli.....	725
Problem sa sintaksom #include<header>	725
C++20 moduli	726
Programiranje modula.....	727
Upotreba modula	728
Zašto je import Module; superiorniji za pretprocesor #include<header>.....	729
Očekivane funkcije u verziji C++23.....	729
Učenje C++ jezika se ovde ne zaustavlja!	730
Online dokumentacija.....	730
Zajednice za smernice i pomoć	730
Rezime	731
Pitanja i odgovori	731
Radionica	732
Vežbe.....	732

DEO VI: DODACI**DODATAK A**

Upotreba brojeva: binarni i heksadecimalni	733
Decimalni numerički sistem.....	734
Binarni numerički sistem.....	734
Zašto računari koriste binarni sistem?.....	735
Šta su bitovi i bajtovi?.....	735
Koliko bajtova čini kilobajt?	736
Heksadecimalni numerički sistem	736
Zašto su potrebni heksadecimalni brojevi?.....	736
Konvertovanje u različitu bazu	737
Generički proces konverzije	737
Konvertovanje decimalnog broja u binarni	737
Konvertovanje decimalnog broja u heksadecimalni	738

DODATAK B

C++ ključne reči	739
-------------------------------	------------

DODATAK C

Pisanje odličnog C++ koda	741
--	------------

DODATAK D

ASCII kodovi	745
ASCII tabela karaktera koji se mogu štampati.....	746

DODATAK E

Odgovori	751
Odgovori za lekciju 1	751
Kviz	751
Vežbe	751
Odgovori za lekciju 2	752
Kviz	752
Vežbe	752
Odgovori za lekciju 3	753
Kviz	753
Vežbe	753
Odgovori za lekciju 4	754
Kviz	754
Vežbe	755
Odgovori za lekciju 5	755
Kviz	755
Vežbe	756
Odgovori za lekciju 6	757
Kviz	757
Vežbe	757
Odgovori za lekciju 7	760
Kviz	760
Vežbe	761
Odgovori za lekciju 8	761
Kviz	761
Vežbe	761
Odgovori za lekciju 9	762
Kviz	762
Vežbe	762
Odgovori za lekciju 10	764
Kviz	764
Vežbe	764
Odgovori za lekciju 11	765
Kviz	765
Vežbe	765
Odgovori za lekciju 12	767
Kviz	767
Vežbe	767
Odgovori za lekciju 13	768
Kviz	768
Vežbe	768
Odgovori za lekciju 14	769
Kviz	769
Vežbe	769
Odgovori za lekciju 15	770
Kviz	770
Odgovori za lekciju 16	771
Kviz	771

Vežbe.....	771
Odgovori za lekciju 17	774
Kviz.....	774
Vežbe.....	775
Odgovori za lekciju 18	778
Kviz.....	778
Vežbe.....	779
Odgovori za lekciju 19	780
Kviz.....	780
Vežbe.....	780
Odgovori za lekciju 20	784
Kviz.....	784
Vežbe.....	784
Odgovori za lekciju 21	784
Kviz.....	784
Vežbe.....	785
Odgovori za lekciju 22	787
Kviz.....	787
Vežbe.....	787
Odgovori za lekciju 23	789
Kviz.....	789
Vežbe.....	789
Odgovori za lekciju 24	790
Kviz.....	790
Vežbe.....	790
Odgovori za lekciju 25	791
Kviz.....	791
Vežbe.....	791
Odgovori za lekciju 26	792
Kviz.....	792
Vežbe.....	792
Odgovori za lekciju 27	793
Kviz.....	793
Vežbe.....	793
Odgovori za lekciju 28	794
Kviz.....	794
Vežbe.....	794
Odgovori za lekciju 29	794
Kviz.....	794
Vežbe.....	794
Odgovori za lekciju 30	796
Vežba.....	796
Odgovori za lekciju 31	796
Vežba.....	796

Uvod

Brz napredak programskog jezika C++ pokrenuo je ovo doba mašinskog učenja i veštačke inteligencije. Možete da upotrebite C++20 da biste programirali jednostavnije, ali moćnije aplikacije nego ikada ranije. U ovoj knjizi, u kojoj detaljno analiziramo skoro 300 primera kompajliranja koda, predstavimo važne funkcije C++20 jezika i opisati njihov interni rad.

Osim što ćemo objasniti osnove C++ programskog jezika, učićete o funkcijama jezika koje su neophodne u profesionalnom C++ programiranju. Između ostalog, učićete o osnovama objektno-orijentisanog programiranja, osnovnim funkcijama i algoritmima Standard Template Library biblioteke i o konceptima, rasponima, prikazima i adapterima verzije C++20. Bez obzira na to da li ste početnik ili profesionalni programer, ova knjiga će vam biti veoma korisna.

Kome je ova knjiga namenjena?

Knjiga počinje opisom osnova C++ jezika. Da biste razumeli kako sve funkcioniše, potrebne su vam želja za učenjem ovog programskog jezika i radoznalost. Postojeće znanje C++ programiranja može biti prednost, ali nije preduslov. Ovo je, takođe, knjiga koju možete da koristite kao referencu ako već znate C++, ali želite da naučite i poboljšanja koja su dodata u taj jezik. Ako ste profesionalni programer, deo III, „Učenje o biblioteci Standard Template Library (STL)“, deo IV, „Lambda izrazi i STL algoritmi“ i deo V, „Napredni C++ koncepti“ pomoći će vam da kreirate bolje, praktičnije C++ aplikacije.

NAPOMENA Posetite veb sajt izdavača i registrujte ovu knjigu na adresi informat.com/register za lakši pristup ažuriranjima, fajlovima za preuzimanje i spiskovima grešaka dostupnim za ovu knjigu.

Organizacija ove knjige

U zavisnosti od aktuelnog nivoa stručnosti u C++ jeziku, možete da izaberete odeljak od kojeg želite da počnete učenje. Ova knjiga je podeljena u pet delova:

- Deo I, „*Osnove*“ - počinje učenjem pisanja jednostavnih C++ aplikacija. Upoznaćete ključne reči koje se najčešće viđaju u C++ kodu.
- Deo II, „*Osnove objektno-orijentisanog C++ programiranja*“ - učićete o principima objektno-orijentisanog programiranja, kao što su kapsuliranje, apstrakcije, nasleđivanja i polimorfizam. O važnosti programiranja konstruktora copy i upotrebe konstruktora move za optimizaciju performansi učićete u lekciji 9, „Klase i objekti“, a o trosmernom operatoru poređenja, koji nazivamo i spaceship operator zbog njegovog oblika (`<=>`), učićete u lekciji 12, „Tipovi operatora i preklapanje operatora“. U lekciji 14, „*Predstavljanje makroa i šablona*“, naučićete da pišete moćan generički C++ kod.
- Deo III, „*Biblioteka Standard Template Library (STL)*“ - pomoći će vam da napišete efikasan i praktičan C++ kod, korišćenjem STL klasa i kontejnera. Na primer, u ovom delu saznaćete kako `std::string` čini operacije spajanja znakovnog niza bezbednim i jednostavnim. U lekcijama 17, „*STL klase dinamičkog niza*“ i 18, „*STL list i forward_list*“ naučićete da koristite standardizovane dinamičke nizove i povezane liste tako da ne morate da programirate sopstvene. U lekciji 20, „*STL mapa i multimap*“ upoznaćete upotrebu parova ključ/vrednost koji su sačuvani u asocijativnim kontejnerima, kao što su `std::map` i `std::multimap`.
- Deo IV, „*Lambda izrazi i STL algoritmi*“ - počinje objašnjenjem kako da programirate objekte funkcije. U lekciji 22, „*Lambda izrazi*“, predstavljena je implementacija neimenovanih objekata funkcije. U lekciji 23 „*STL algoritmi*“ učićete o različitim algoritmima koji vam pomažu pri izvršenju operacija u kontejnerima, kao što je pronalaženje, uklanjanje i sortiranje elemenata.
- Deo V, „*Napredni C++ koncepti*“ - počinje objašnjenjem pametnih pokazivača i rukovanja izuzecima, koji značajno doprinose stabilnosti i kvalitetu aplikacije. Objašnjene su ključne funkcije predstavljene u verziji C++20. U lekciji 29, „*C++20 koncepti, opsezi, prikazi i adapteri*“ naučićete da potvrdite parametre šablona upotrebom koncepta i strukturnih prikaza elemenata u opsegu upotrebom adaptera. U lekciji 31, „*C++20 moduli i C++23*“ učićete o modulima koji će zameniti tradicionalne fajlove zaglavlja, a lekciju završavamo predstavljanjem poboljšanja jezika za koje se očekuje da će biti izdati u sledećoj verziji standardnog C++ jezika, u verziji C++23..

Konvencije upotrebljene u ovoj knjizi

Unutar lekcija pronaći ćete sledeće elemente koji obezbeđuju dodatne informacije:

NAPOMENA Ovi okviri obezbeđuju dodatne informacije u vezi onoga što čitate.

PAŽNJA Ovi okviri upozoravaju na probleme ili sporedne efekte koji se mogu desiti u posebnim situacijama.

SAVET Ovi okviri predstavljaju najbolju praksu u pisanju C++ programa.

URADITE	NE RADITE
Upotrebite „Uradite/Ne radite“ okvire da biste pronašli rezime osnovnih principa lekcije.	Nemojte da previdite korisne informacije koje se nalaze u ovim okvirima.

U ovoj knjizi upotrebljeni su različiti fontovi da biste razlikovali kod i običan tekst. Kod, komande i termini koji se odnose na programiranje prikazani su računarskim fontom.

Primeri koda za ovu knjigu

Primeri koda za ovu knjigu dostupni su besplatno na stranici <https://github.com/learncpnow/9E.git>.

Želimo da vas čujemo!

Kao čitaoci ove knjige, vi ste nam najvažniji kritičari i komentatori. Mi cenimo vaše mišljenje i želimo da znamo šta smo, po vašim ocenama, uradili dobro, šta smo mogli da uradimo bolje, iz koje oblasti biste želeli da publikujemo sledeće knjige i željni smo da čujemo vaše mudre reči koje želite da nam pošaljete.

Očekujemo vaše komentare. Možete da nam pošaljete e-mail i napišete šta vam se dopalo ili nije dopalo u našoj knjizi i šta možemo da uradimo da bismo poboljšali naše knjige.

Imajte na umu da ne možemo da vam pomognemo u vezi sa tehničkim problemima koji su u vezi sa temom ove knjige.

Kada nam pišete, navedite naslov knjige i ime autora, kao i vaše ime i e-mail adresu. Mi ćemo pažljivo pregledati vaše komentare i podeliti ih sa autorom i urednicima koji su radili na knjizi.

Email: community@informit.com

Početak rada

Dobrodošli u Sams Teach Yourself C++ za sat vremena dnevno! Spremni ste da postanete vešt C++ programer.

U ovoj lekciji otkrićete:

- zašto je C++ standard u razvoju softvera
- kako da unesete, kompajlirate i povežete prvi radni C++ program
- šta je novo u verziji C++20

Kratka istorija C++ jezika

Svrha programskog jezika je da olakša potrošnju računarskih resursa. C++ nije nov - to je jezik koji je popularan i neprestano se razvija. U vreme pisanja ove knjige, najnovija verzija C++ jezika je C++20, koju je ratifikovao International Organization for Standardization (ISO), a izdata je u decembru 2020. godine. Prethodne izmene jezika su bile označene kao verzija C++17 (izdata 2017. godine), verzija C++14 (izdata 2014. godine) i verzija C++11 (izdata 2011. godine).

Povezanost sa C jezikom

Bjarne Stroustrup u Bell Labs-u je razvio C++ 1979. godine. C++ jezik je projektovan da bude naslednik jezika C, od koga se razlikuje po činjenici da je dizajniran da bude objektno-orijentisan jezik, koji implementira koncepte, kao što su nasleđivanje, apstrakcija, polimorfizam i kapsuliranje. C++ uključuje klase koje se koriste za skladištenje podataka člana i metoda člana. Ovi metodi člana su funkcije koje izvršavaju operacije na podacima člana. Ova organizacija pomaže programeru da modeluje podatke i akcije koje želi da izvrši na podacima. Popularni C++ kompajleri nastavili su da podržavaju i programiranje C jezikom i da pružaju jednostavnu integraciju za oba sveta, što je rezultiralo odličnom kompatibilnošću sa starijim verzijama.

NAPOMENA Poznavanje C programiranja nije preduslov za učenje jezika C++. Ako je vaš cilj da naučite objektno-orijentisan programski jezik, kao što je C++, nije potrebno da počinjete od učenja proceduralnog jezika, kao što je C, već možete početi direktno da koristite ovu knjigu.

Prednosti C++ jezika

C++ smatramo programskim jezikom srednjeg nivoa. To je svestran jezik i možete da ga koristite za programiranje aplikacija visokog nivoa koje ne treba da budu svesne specifičnosti hardvera na kom se pokreću. C++ takođe možete da koristite za programiranje biblioteka niskog nivoa, koje funkcionišu bliže hardveru, kao što su drajveri uređaja. Prema tome, C++ pruža optimalnu srednju putanju kao pomoć programerima da razviju kompleksne aplikacije koje ne zahtevaju nikakve kompromise u pogledu performansi ili upravljanja resursima.

Uprkos postojanju programskih jezika kao što su Java, C# i drugi jezici, C++ je i dalje veoma popularan i još uvek se razvija. Ovi drugi jezici su *interpretirani* izvršnom komponentom koja upravlja resursima za programera. Izvršenje olakšava programiranje i apstrahuje sistemске resurse do te mere da su jezici neprikladni za mnoga izračunavanja visoke performanse. Stoga, C++ ostaje izabrani jezik u slučajevima gde je potrebna potpuna kontrola nad upotrebom memorije i nad performansama. Slojevita arhitektura, gde veb server programiran C++ jezikom služi druge komponente programirane u jezicima HTML, Java, JavaScript ili .NET, je uobičajena. C++ je izabrani jezik za veštačku inteligenciju i mašinsko učenje, drajvere uređaja, baze podataka, operativne sisteme, servise, pa čak i za kompajlere i interpretere drugih programskih jezika.

Evolucija C++ standarda

Zbog svoje popularnosti C++ jezik je prihvaćen i usvojen na mnogim različitim platformama (operativnim sistemima) koje koriste sopstvene C++ kompajlere. Njegov razvoj je izazvao odstupanja specifična za kompajlere i, samim tim, probleme interoperabilnosti i probleme promene platforme. Stoga se pojavila potreba za standardizacijom jezika i proizvođačima kompajlera su obezbeđene standardne specifikacije jezika koje bi trebalo da koriste.

ISO Committee je 1998. godine ratifikovao prvu standardnu verziju jezika C++ ISO/IEC 14882:1998. Od tada je taj standard pretrpeo velike promene kojima je poboljšana upotrebljivost jezika i proširena podrška standardne biblioteke. Kao što je ranije pomenuto, u vreme pisanja ove knjige aktuelna ratifikovana verzija standarda je ISO/IEC 14882:2020, popularno nazvana C++20.

NAPOMENA Aktuelni standard možda nisu u potpunosti podržali svi popularni kompajleri. Stoga ćete u ovoj knjizi učiti o najnovijim dodacima jeziku C++20 i objasniti ćemo dobro podržane funkcije jezika koje vam pomažu u pisanju dobrih, funkcionalnih C++ aplikacija.

Ko koristi programe napisane C++ jezikom?

Lista aplikacija, operativnih sistema, interpretera, veb servisa, baza podataka i poslovnog softvera programiranih u jeziku C++ je veoma duga. Bez obzira na to šta ste i šta radite na računaru, postoji mogućnost da već koristite softver programiran u jeziku C++. Na primer, V8 JavaScript Engine od Google-a je programiran u jeziku C++. On je sastavni deo popularnih pretraživača i serverske tehnologije, kao što je Node.js.

C++ je izabrani jezik za istraživačke radove fizičara, matematičara i istraživača podataka. Uspom veštačke inteligencije upotrebom algoritama mašinskog učenja podstaknut je računarsvom visokih performansi koje omogućava C++.

Programiranje C++ aplikacije

Kada pokrenemo omiljen pretraživač ili aplikaciju za obradu reči, mi u stvari ukazujemo procesoru da pokrene izvršni fajl konkretnog programa. Izvršni fajl je završeni proizvod koji bi trebalo da izvrši ono što je programer želeo da postigne.

Koraci za izgradnju izvršnog fajla

Programiranje korišćenjem jezika C++ je prvi korak ka kreiranju izvršnog fajla koji se pokreće na operativnom sistemu. Osnovni koraci u kreiranju aplikacija u C++ jeziku su sledeći:

1. Unesite C++ kod pomoću editora teksta. Editor teksta je obično editor koda ili integrisano razvojno okruženje (IDE).
2. Kompajlirajte kod pomoću C++ *kompajlera* koji kreira verziju mašinskog jezika koji se nalazi u fajlovima objekta.
3. Povežite fajlove objekta pomoću *povezivača* da biste dobili izvršni fajl (na primer, .exe u Windowsu)

Kompajliranje je korak u kojem je kod u C++ jeziku, koji je sadržan u tekstualnim fajlovima i ima ekstenziju .cpp, konvertovan u kod bajtova koji procesor može da izvrši. Kompajler konvertuje po jedan fajl koda generisanjem fajla objekta koji ima ekstenziju .o ili .obj i ignoriše zavisnosti koda u drugom fajlu.



SAVET Popularni kompajleri uključuju `g++` od GNU Projecta, `clang++` od LLVM-a i Microsoft Visual C++ (MSVC) kompajler. `g++` i `clang++` su uglavnom u upotrebi u Linux i macOS okruženjima, dok je MSVC izabrani kompajler za Windows.

U vreme pisanja ovog teksta nijedan kompajler ne obećava punu podršku za C++20, ali `g++` i MSVC su bolji od ostalih.

Povezivač (eng. linker) je alat koji rešava istaknute zavisnosti između .obj fajlova. Prilikom uspešnog povezivanja alat kreira izvršni fajl koji programer može da izvrši ili distribuirati.

Proces uspešnog kompajliranja i povezivanja nazivamo *izgradnja izvršnog fajla*.

SAVET Upotreba online kompajlera koji je dostupan putem pretraživača može da bude najbrži način za početak editovanja, kompajliranja i izvršavanja jednostavnih C++ aplikacija. Potražite „online C++ compiler“ i isprobajte opcije koje pronađete. Za bilo koji kompajler, prilikom kompajliranja novih funkcija jezika predstavljenih u ovoj knjizi, obratite pažnju da podržava C++20.

Analiziranje i ispravljanje grešaka

Aplikacije se retko kompajliraju i izvršavaju pri prvom pokretanju, kao što bi trebalo. Velika ili složena aplikacija programirana u bilo kom jeziku (uključujući i C++) treba da se pokrene više puta, što je deo testiranja i identifikovanja grešaka u kodu, koje nazivamo *programske greške*. Nakon što su programske greške ispravljene, izvršni fajl je ponovo izgrađen i proces testiranja se nastavlja. Stoga, pored tri koraka - programiranja, kompajliranja i povezivanja - razvoj softvera uključuje i korak koji nazivamo *otklanjanje grešaka*, u kojem programer analizira greške u kodu i ispravlja ih. Dobra razvojna okruženja obezbeđuju alatke i funkcije koje pomažu pri otklanjanju grešaka.

Integrirana razvojna okruženja

Mnogi programeri često koriste integrirano razvojno okruženje (IDE) u kojem su koraci programiranja, kompajliranja, povezivanja i otklanjanja grešaka integrirani unutar jedinstvenog korisničkog interfejsa.

Ako želite da programirate u C++ jeziku korišćenjem IDE-a, možete da instalirate jedan od mnogih besplatno dostupnih C++ IDE-a i kompajlera da biste započeli učenje. Neki od popularnih IDE-a su Eclipse i Code::Blocks za Linux sisteme, Xcode za macOS sisteme i Microsoft Visual Studio za Windows sisteme.

URADITE	NE RADITE
Snimite C++ fajlove korišćenjem ekstenzije <code>.cpp</code> .	Nemojte da upotrebite ekstenziju <code>.c</code> za C++ fajlove, zato što će neki kompajleri prevesti te fajlove kao C programe, umesto kao C++.
Za pisanje koda upotrebite jednostavan editor teksta, editor koda ili integrisano razvojno okruženje.	Za pisanje koda nemojte da koristite programe za obradu teksta, jer funkcije za automatsko ispravljanje i formatiranje teksta mogu da dovedu do grešaka kompajliranja.

Programiranje prve C++ aplikacije

Sada, kada poznajete alatke i korake, vreme je da programirate prvu C++ aplikaciju koja prati tradiciju i prikazuje „Hello World!“ na ekranu.

Ako programirate na Linux ili macOS sistemu, upotrebite editor teksta koji vam je poznat (ja sam upotrebio `gedit` na Ubuntu sistemu) za kreiranje fajla koji sadrži kod prikazan u programskom kodu 1.1. Zatim snimite fajl čiji se naziv završava ekstenzijom `.cpp` - na primer, `Hello.cpp`.

Ako koristite macOS sa Xcode IDE-om, pratite sledeće korake da biste kreirali novi C++ projekat:

1. Otvorite New Project Wizard pomoću opcije menija File, New, Project.
2. Selektujte Command Line Tool i kliknite Next.
3. Izaberite naziv proizvoda, kao što je Hello. Za jezik izaberite C++ i kliknite Next.
4. Zamenite automatski generisan sadržaj u fajlu `main.cpp` isečkom koda koji je prikazan u programskom kodu 1.1.

Ako koristite Windows sa Microsoft Visual Studio IDE-om, pratite sledeće korake da biste kreirali nov C++ projekat:

1. Otvorite New Project Wizard pomoću opcije menija File, New, Project.
2. Selektujte C++, za tip izaberite Console App i kliknite Next.
3. Dodelite naziv projektu, kao što je Hello. Kliknite Create.
4. Zamenite automatski generisan sadržaj u fajlu `Hello.cpp` isečkom koda koji je prikazan u programskom kodu 1.1.



PROGRAMSKI KOD 1.1 Hello.cpp, program Hello World

```
1: #include<iostream>
2:
3: int main()
4: {
5:     std::cout << "Hello World!" << std::endl;
6:     return 0;
7: }
```

Ova jednostavna aplikacija samo prikazuje „Hello World!“ na ekranu, korišćenjem linije koda `std::cout`. Linija koda `std::endl` daje instrukcije za `cout` da završi konkretnu liniju unosom novog reda, a aplikacija se zatvara i operativnom sistemu vraća 0.

SAVET Možda vidite varijante linije 5 u programskom kodu 1.1, koja je sledeća:

```
5:     std::cout << "Hello World!\n";
```

Izlaz programa se neće promeniti. "Hello World!\n" uključuje nov red u formi `\n` i nije dodatno unet upotrebom linije `cout::endl`.

Neki primeri koda u ovoj knjizi koriste `'\n'` kao pomoć da se linija koda uklopi u liniju na štampanoj stranici.

NAPOMENA Da biste čitali program, može biti korisno da znate kako da izgovarate specijalne karaktere i ključne reči.

Na primer, `#include` možete da čitate kao „hash-include“. Ostale verzije su „sharp-include“ ili „pound-include“, u zavisnosti od toga odakle ste.

Slično tome, možete da čitate `std::cout` kao „standard-c-out“, a `endl` kao „end-line“.

PAŽNJA Važni su detalji, što znači da bi trebalo da kucate kod na potpuno isti način kao što je prikazano u programskom kodu. Kompajleri su striktni i ako slučajno na kraj iskaza unesete `;`, a potrebno je da unesete `,`, možete da očekujete grešku kompajliranja i izveštaj o grešci! Iz istog razloga ne bi trebalo da koristite program za obradu teksta za editovanje koda.

SAVET Editori koda, kao što je Visual Studio Code, dostupni su besplatno za Linux, macOS i Windows sisteme. Ako ne koristite IDE, sledeća najbolja opcija je da upotrebite Visual Studio Code za editovanje koda.

Izgradnja i izvršavanje prve C++ aplikacije

Ako koristite g++ kompajler od GCC-a, otvorite terminal, kliknite na direktorijum koji sadrži `hello.cpp` i otvorite kompajler i program za povezivanje, korišćenjem komandne linije:

```
g++ -o hello Hello.cpp -std=c++20
```

Alternativno, ako koristite clang++ kompajler na macOS sistemu, pratite prethodno navedene korake i izvršite sledeću komandnu liniju:

```
clang++ -o hello Hello.cpp
```

Ove komande daju instrukcije kompajleru g++ i clang++ da kreiraju izvršni fajl pod nazivom `hello`, kompajliranjem C++ fajla `Hello.cpp`.

OMOGUĆITE C++20 FUNKCIJE KADA KORISTITE KOMPJLER G++ ILI CLANG++

Da biste kompajlirali kod specifičan za verziju C++20 korišćenjem kompajlera g++ ili clang++ dodajte parametar `-std=c++20` u komandnu liniju:

```
g++ -o hello Hello.cpp -std=c++20
```

Ili

```
clang++ -o hello Hello.cpp -std=c++20
```

Parametar će vam biti potreban kada kompajlirate kod koji koristi funkcije verzije C++20.

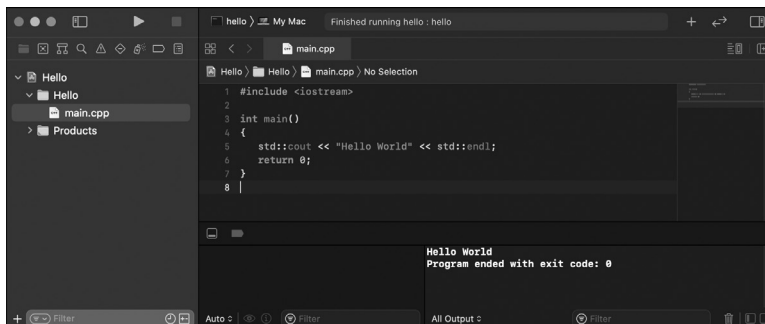
Ako koristite macOS sistem sa Xcode IDE-om, možete da izgradite i pokrenete aplikaciju tako što ćete selektovati opciju `Product, Run`.

Program sastavljen u Xcode IDE-u bi trebalo da izgleda slično onom koji je ilustrovan na slici 1.1.



Slika 1.1

Jednostavan „Hello World“ C++ program editovan u Xcode 12.5 IDE-u na macOS sistemu.



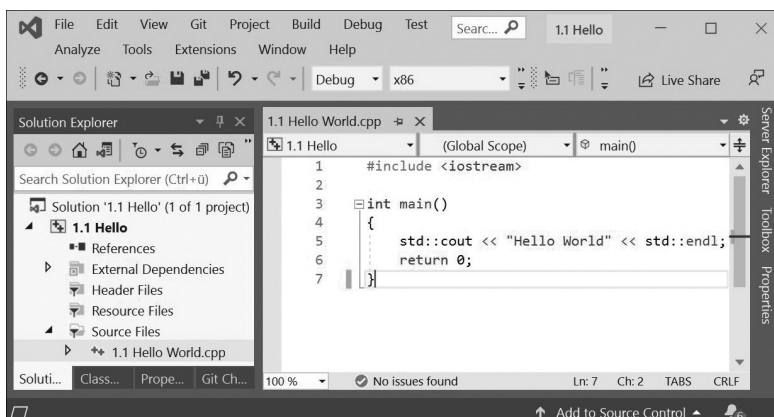
SAVET Da biste kompajlirali C++20 kod u Xcode IDE-u potrebno je da ga eksplicitno omogućite podešavanjem opcije C++ Language Dialect na C++20. Ova opcija se nalazi u meniju Build Settings.

Ako koristite Microsoft Visual Studio na Windows sistemu, pritisnite tastere Ctrl+F5 da biste pokrenuli program direktno pomoću IDE-a. Ova prečica će kompajlirati, povezati i izvršiti aplikaciju.

Program kreiran u Microsoft Visual Studio IDE-u bi trebalo da izgleda slično kao što je ilustrirano na slici 1.2.

Slika 1.2

Jednostavan „Hello World“ C++ program editovan u Microsoft Visual Studio 2019 IDE-u.



SAVET Da biste kompajlirali C++20 kod u Microsoft Visual Studio IDE-u potrebno je da ga eksplicitno omogućite podešavanjem opcije C++ Language Standard na ISO C++20 Standard. Ova opcija se nalazi u meniju Project, Properties.

Izvršavanje fajla `./hello` ili fajla `Hello.exe` vraća sledeći ispis:

```
Hello World!
```

Čestitam! Započeli ste učenje jednog od najpopularnijih i najmoćnijih programskih jezika svih vremena!

ZNAČAJ C++ ISO STANDARDA

Programski kod 1.1 prikazuje da usklađenost sa C++ ISO standardom pomaže pri razvoju koda koji može da bude kompajliran i izvršen na više operativnih sistema.

ISO standard za C++ omogućava konzistentnu podršku za kompajler i prenosivost koda na različite platforme. Zato, kao programer možete da doprete do više korisnika, bez potrebe za programom koji je specifičan za svaki operativni sistem koji oni koriste.

Razumevanje grešaka kompajlera

Kompajleri su veoma precizni u svojim zahtevima, ali samo dobri kompajleri se trude da ukažu gde se nalaze greške u kodu. Da biste to videli na delu, možete namerno da unesete grešku u programski kod 1.1 tako što ćete izbrisati tačku-zarez (;) na kraju linije 5 ispred `return` u liniji 6. Sa tačke gledišta kompajlera, započeli ste nov iskaz u liniji 6 bez završavanja prethodnog iskaza u liniji 5 - što je protiv pravila. Ako sada pokušate da kompajlirate kod, dobićete grešku, kao što je sledeća:

```
1.1 Hello World.cpp(6,2): error C2143: syntax error: missing ';' before 'return'
```

Ova poruka o grešci iz Visual C++ kompajlera počinje nazivom fajla koji sadrži grešku, brojem linije u kojoj se suočio sa novim iskazom pre zaključenja prethodnog (u ovom slučaju 6) i opisom greške i brojem greške (u ovom slučaju C2143). Možete da ispravite ovu grešku dodavanjem znaka tačka-zarez na početak šeste linije i videćete da će program biti kompajliran bez problema!

NAPOMENA Upotreba znaka tačka-zarez (;) je validan način za završavanje iskaza u C++ jeziku. U C++ jeziku prelom linije neće automatski završiti iskaze kao u nekim drugim jezicima, kao što je VBScript.

U C++ jeziku iskazi mogu da se protežu kroz više linija. Takođe je moguće da imate više iskaza u jednoj liniji, u kojoj je svaki iskaz završen znakom tačka-zarez.

Šta je novo u verziji C++20?

Revizije C++ standarda dovele su do jezika koji je jednostavniji za upotrebu za programiranje bez kompromitovanja mogućnosti pisanja aplikacija visokih performansi.

ISO je objavio verziju C++20 decembra 2020. godine, što predstavlja ogroman korak u modernizaciji jezika. Predstavljeni su trosmerni operator poređenja (takođe ga nazivamo i spaceship operator), validacija parametara šablona, biblioteka novih raspona koja uključuje prikaze i adaptere koji olakšavaju moćne operacije u kolekcijama, dalja standardizacija višenitnog rada, podrška za sinhronizaciju pomoću korutina i poboljšani lambda izrazi. Takođe, u verziji C++20 predstavljeni su *moduli* - dugo očekivana funkcija koja prevazilazi nedostatke uključivanja fajlova zaglavlja (.h) i znatno ubrzava proces kompajliranja u velikim projektima. Izmene u verziji C++20 su toliko drastične da u vreme pisanja ovog teksta nijedan kompajler ne obećava podršku za C++20 standard, mada ovde pomenuti kompajleri odlično napreduju ka tom cilju.

C++ je jezik koji se i dalje razvija. Funkcije koje očekujemo u sledećoj reviziji, koju očekujemo 2023. godine, predstavljene su na kraju knjige, u lekciji 31, „C++20 moduli i C++23“.

Rezime

U ovoj lekciji naučili ste da programirate, kompajlirate, povezujete i izvršavate prvi C++ program. U njoj je, takođe, predstavljen kratak pregled evolucije C++ jezika i demonstrirana je efikasnost standarda prikazom da isti program može da bude kompajliran korišćenjem različitih kompajlera na različitim operativnim sistemima.

Pitanja i odgovori

P Mogu li da ignorišem poruke upozorenja kompajlera?

- O U određenim slučajevima kompajler će prikazati poruke upozorenja. Upozorenja se razlikuju od grešaka po tome što je konkretna linija sintaktički tačna i može da se kompajlira. Međutim, verovatno postoji bolji način da se konkretna linija napiše, a dobri kompajleri će prikazati upozorenje, zajedno sa preporukom za ispravku.

Trebalo bi da obratite pažnju na ova upozorenja i da, u skladu sa njima, poboljšate aplikaciju. Nemojte ignorisati poruke upozorenja osim ako ste sigurni da su one lažno pozitivne.

P Po čemu se razlikuje interpretirani od kompajliranog jezika?

- O** Jezici kao što su JavaScript, Ruby i Python su interpretirani. Interpretirani jezik koristi interpreter, koji čita (interpretira) kod u fajlu skripta i izvršava željene akcije. Kompajlirani jezici, kao što je C++, koriste korak izgradnje za kreiranje izvršnih fajlova koji sadrže instrukcije za izvršenje na procesoru.

P Šta su greške pri izvršenju i po čemu se one razlikuju od grešaka u vreme kompajliranja?

- O** Greške koje se dešavaju kada izvršavate aplikaciju nazivamo *greške pri izvršenju*. Možda ste već iskusili poznatu grešku „Access Violation“ na Windows sistemu - to je greška pri izvršenju koju uzrokuje izvršni fajl koji ima grešku. Greške u vreme kompajliranja prekidaju proces izgradnje. Ukazuju na sintaktičke probleme u kodu i potrebno je da ih programer ispravi pre nego što može da nastavi kompajliranje.

P Zašto je potrebno eksplicitno omogućiti C++20 funkcije pri upotrebi g++, clang++ ili MSVC kompajlera? Zašto nisu podrazumevano uključene?

- O** Izmene u jeziku C++ uključuju nove funkcije i poboljšanja postojećih. Kompajleri treba da obezbede kompatibilnost sa starijim verzijama i stabilnost izvršenja postojećeg koda. Oni to rade predstavljanjem novih funkcija (neke su još u eksperimentalnoj fazi) koje podržavaju C++ standard korišćenjem novih parametara komandne linije da postojeći skriptovi izgradnje ne bi bili ugroženi.

Radionica

Radionica sadrži kviz da biste proverili koliko dobro razumete ono što ste naučili u ovoj lekciji i vežbe za sticanje iskustva u upotrebi onoga što ste naučili. Pokušajte da odgovorite na pitanja u kvizu i obavite vežbe pre nego što proverite odgovore u Dodatku E, i uverite se da razumete odgovore pre nego što pređete na sledeću lekciju.

Kviz

1. Ignorišući sintaksu, koja je razlika između kompajliranog jezika, kao što je C++ i interpretiranog jezika, kao što je JavaScript?
2. Šta radi program za povezivanje?
3. Koji su koraci u normalnom ciklusu razvoja programa?



Vežbe

1. Pogledajte sledeći program i pokušajte da pogodite šta on izvršava, a da ga ne pokrenete:

```
1: #include<iostream>
2: int main()
3: {
4:     int x = 8;
5:     int y = 6;
6:     std::cout << std::endl;
7:     std::cout << x - y << " " << x * y << " " << x + y;
8:     std::cout << std::endl;
9:     return 0;
10: }
```

2. Unesite program iz vežbe 1, a zatim ga kompajlirajte i povežite. Šta izvršava ovaj program? Izvršava li ono što ste mislili?
3. Šta mislite u čemu je greška u sledećem programu:

```
1: include<iostream>
2: int main()
3: {
4:     std::cout << "Hello Buggy World " << std::endl;
5:     return 0;
6: }
```

4. Ispravite grešku u programu iz vežbe 3, kompajlirajte ga, povežite i pokrenite. Šta izvršava ovaj program?