Laslo Kraus

PROGRAMSKI JEZIK

C#

sa rešenim zadacima

(C# 6)

AKADEMSKA MISAO
Beograd, 2016

Laslo Kraus

PROGRAMSKI JEZIK C# SA REŠENIM ZADACIMA

Recenzenti
Dr Igor Tartalja
Dr Đorđe Đurđević

Izdaje i štampa
AKADEMSKA MISAO
Bulevar kralja Aleksandra 73, Beograd

Lektor
Anđelka Kovačević

Dizajn naslovne strane
Zorica Marković, akademski slikar

Tiraž
300 primeraka

ISBN 978-86-7466-626-5

Dušanki i Katarini

Predgovor

Ova knjiga predstavlja udžbenik za programski jezik *C#* za široki krug čitalaca. Knjigu mogu da koriste i početnici u programiranju, ali poznavanje osnovnih pojmova iz objektno-orijentisanog programiranja i programskih jezika *C/C++* ili *Java* znatno olakšava da se savlada materija iz ove knjige.

Programski jezik *C#* izložen je u obimu koji može da zadovoljava i naprednije neprofesionalne programere. Od ogromne standardne biblioteke klasa, koja prati jezik *C#*, objašnjeni su samo delovi koji su potrebni za efikasno programiranje pri rešavanju većine problema relativno visoke složenosti.

Jedini način da se nauči neki programski jezik jeste da se pišu programi na njemu. Ova knjiga je u potpunosti podređena tom osnovnom načelu.

Uvodno poglavlje 1 sadrži informacije korisne pre svega za početnike: osnovne pojmove objektno-orijentisanog programiranja, glavne osobine programskog jezika *C#* i načine obrade programa u tekstualnom i grafičkom okruženju.

U poglavlju 2 obrađeni su podaci koji su predmet obrade u programima. Objašnjen je pojam tipa podataka, kao i načini definisanja i korišćenja skalarnih brojčanih, znakovnih, logičkih i tekstualnih podataka. Da bi što pre mogli da se pišu potpuni programi u ovom poglavlju su obrađene i ulazna i izlazna konverzija brojčanih podataka. Naime, programi koji ne čitaju početne podatke i ispisuju rezultate nemaju nikakvu upotrebnu vrednost.

Poglavlje 3 posvećeno je operatorima i načinima sastavljanja izraza pomoću njih. Tu su obrađeni operatori jezika *C#* koji čitaocu s prosečnim matematičkim obrazovanjem ne bi trebalo da budu strani. Prikazivanje nekih specifičnijih operatora je odloženo za kasnije. Kao proširenje skupa operatora dat je i pregled važnijih standardnih bibliotečkih funkcija koje postoje u jeziku *C#*. Odabrane su funkcije koje se odnose na opštepoznate pojmove iz matematike i na obradu već objašnjenih tipova podataka.

U poglavlju 4 prikazane su naredbe koje predstavljaju jedinične obrade u programima. Pored prostih naredbi obrađene su i složene naredbe jezika *C#* koje omogućavaju sastavljanje složenih programa na efikasan i pregledan način: uslovno izvršavanje (selekcije), višestruko izvršavanje (ciklusi) delova programa i upravljačke naredbe (skokovi).

Iz matematike dobro poznati nizovni tipovi: vektori, matrice i višedimenzionalni nizovi obrađeni su u poglavlju 5.

Izučavanje objektno-orijentisanog programiranja počinje u poglavlju 6 o klasama. Klase su složeni tipovi podataka koji programerima omogućuju modeliranje objekata iz realnog života. Sastoje se od polja čije vrednosti čine stanja primeraka klasa i metoda koje definišu moguće načine korišćenja tih polja. Primerci klasa su podaci klasnih tipova i nazivaju se

objektima. Jedan od važnih elemenata ispravnog rada programa je pravilna inicijalizacija objekata.

Programski jezik *C#* omogućava grupisanje srodnih klasa u prostore imena. U poglavlju 7 obrađeni su prostori imena: njihovo formiranje i korišćenje. Prikazani su i važniji elementi dva najčešće korićena standardna prostora imena.

Poglavlje 8 posvećeno je sledećem fundamentalnom konceptu objektno-orijentisanog programiranja, nasleđivanju. U grupama podataka koje su predstavljene klasama mogu da se uoče podgrupe sa nekim specifičnijim osobinama. Te podgrupe, predstavljene potklasama, nasleđuju osobine širih grupa, natklasa. Član podgrupe uvek može da se koristi i kao član opštije nadgrupe. U ovom poglavlju obrađeni su i interfejsi, specijalni apstraktni tipovi koji samo definišu određene osobine koje klase koje ih ostvaruju treba da poseduju, a ne i način kako se te osobine ostvaruju.

U poglavlju 9 obrađeni su vrednosni tipovi, tipovi čije promenljive podatke sadrže neposredno. Za razliku od njih, promenljive nizovnih i klasanih tipova tipova sadrže samo upućivače na mesta u memoriji gde se nalaze elementi nizova, odnosno primerci klasa.

Jezik *C#* omogućava da rukovanje izuzecima (greškama) vremenski ne opterećuje program sve dok se izuzetak ne pojavi. Kada se izuzetak pojavi, kontrola se automatski predaje jednom od rukovalaca izuzecima koje je definisao programer. Način rukovanja izuzecima prikazan je u poglavlju 10.

U poglavlju 11 objašnjeni su generički tipovi i metode. Generičke klase i metode omogućavaju pisanje klasa i metoda za obradu podataka, čiji tipovi nisu poznati u vreme njihovog pisanja, na bezbedan način sa stanovišta provera ispravnog korišćenja tipova podataka. Generički interfejsi, na sličan način, opisuju očekivane osobine budućih klasa u uslovima kada tipovi pojedinih korišćenih podataka nisu poznati. Na kraju ovog poglavlja su ukratko objašnjene i dve najčešće korišćene standardne generičke klase za rad sa zbirkama podataka.

U poglavlju 12 izloženi su osnovni koncepti konkurentne obrade, obrade koja se sastoji od istovremenog odvijanja više tokova kontrole, više istovremenih obrada. Ti tokovi kontrola nazivaju se niti, čija je korektna sinhronizacija osnova dobrog programa s konkurentnom obradom.

Rukovodeći se već istaknutim načelom o učenju jezika, ova knjiga, pored manjih primera u toku izlaganja, na kraju svakog poglavlja sadrži nekoliko rešenih zadataka sa detaljnim objašnjenjima. Više zadataka može da se nađe u autorovoj zbirci *Rešeni zadaci iz programskog jezika C#* (videti [8]). Kroz zadatke u ovoj knjizi i u zbirci skreće se pažnja na neke specifičnosti jezika *C#* i daju se ideje za elegantno i efikasno rešavanje nekih problema.

Ova knjiga je više nego udžbenik za programski jezik *C#.* Kroz rešene zadatke prikazane su i primene osnovnih elemenata objektno-orijentisanog programiranja, a na primerima koji se uopšteno sreću u računarskoj tehnici: obradi redova, stekova, listi, tekstova itd. Posebna pažnja posvećena je i inženjerskim aspektima programiranja. Nije dovoljno da program rešava zadati problem, već treba da bude i „lep", razumljiv, da se lako održava, da troši što manje memorije i da bude što efikasniji.

Svi programi koji se nalaze u ovoj knjizi potpuni su u smislu da mogu da se izvršavaju na računaru. Provereni su na nekoliko računara korišćenjem nekoliko različitih prevodilaca za jezik *C#.*

Izvorni tekstovi svih programa iz ove knjige mogu da se preuzmu sa Interneta, sa adrese `home.etf.rs/~kraus/knjige/`. Na istoj adresi objaviće se ispravke eventualnih, naknadno otkrivenih grešaka u knjizi. Svoja zapažanja čitaoci mogu da upute autoru elektronskom poštom na adresu `kraus@etf.rs`.

Laslo Kraus

Beograd, oktobar 2016.

Sadržaj

1 Uvod

1.1 O programskom jeziku *C#*

Programski jezik *C#* viši je programski jezik opšte namene koji u potpunosti podržava paradigmu objektno-orijentisanog programiranja.

C# je više od programskog jezika koji ima određenu sintaksu i semantiku. Ona je pravo programsko okruženje koje pomoću ogromne biblioteke standardnih programskih modula podržava obrade bez kojih danas ne može da se zamisli nijedan ozbiljan program. Tu spadaju komunikacija sa korisnikom preko prozora, višenitna obrada, rad preko *Interneta*, troslojna arhitektura rada s bazama podataka itd. Standardi programskih jezika do pojave jezika *C#* te elemente nisu obuhvatali. Svaki proizvođač je uvodio nestandardna rešenja za njih, ograničavajući time prenosivost programa s jedne na drugu platformu.

Programski jezik *C#* je srodnik jezika *C++* koji je direktni potomak jezika *C*, a u velikoj meri liči i na jezik *C#*.

Pogramski jezik *C* pojavio se početkom 1970-tih godina i u osnovama je promenio dotadašnji, nesistematizovani, stil programiranja. Kroz podršku za *strukturirano programiranje* obezbedio je da mogu da se napišu, razumeju i održavaju znatno veći programski sistemi nego pre. Pošto se u centru pažnje nalaze postupci koji se primenjuju na podatke, ta tehnika programiranja naziva se i *proceduralno programiranje*.

Pošto su jezik *C* projektovali programeri koji su i sami pisali programe, drugi programeri su ga rado prihvatili i uskoro je postao jedan od najrasprostranjenijih programskih jezika, koji se koristi i danas. Za autora jezika *C* smatra se Denis Riči (*Dennis Ritchie*) iz Belovih laboratorija (*Bell Laboratories*).

Početkom 1980-tih godina računarski programi narasli su do takvih veličina da su se tehnike proceduralnog programiranja pokazale neefikasnim. Rešenje se našlo u paradigmi *objektno-orijentisanog programiranja*, kod koje se u centru pažnje nalaze podaci (objekti) na koje se primenjuju neki postupci. Od programskih jezika koji su projektovani da podržavaju nove tehnike programiranja najveći uspeh imala je nadogradnja jezika *C*, koja je danas poznata pod imenom *C++* (++ je u jeziku *C* operator povećavanja!). Preko 95% jezika *C* usvojeno je bez izmena i u jeziku *C++*.

C++ je hibridni jezik u smislu što, pored novijeg objektno-orijentisanog programiranja, podžava i starije proceduralno programiranje.

Za autora jezika *C++* smatra se Bjarn Strostrup (*Bjarne Stroustrup*).