

METODIKA NASTAVE PROGRAMIRANJA: KONTROLA TOKA PROGRAMA

Marko Arambašić¹, Olivera Hrnjaković², Aleksandar Kupusinac³, Rade Doroslovački⁴

^{1,2,3,4}Univerzitet u Novom Sadu, Fakultet tehničkih nauka, Novi Sad, Srbija

¹marko.arambasic@uns.ac.rs, ²olja.hrnjakovic@uns.ac.rs, ³sasak@uns.ac.rs, ⁴rade.doroslovacki@uns.ac.rs

Kratak sadržaj: Kontrole toka predstavljaju jednu od osnovnih tehnika koje početnici treba da savladaju i ispravno koriste, kako u programskom kodu, različitim poslovnim procesima gde se poznavanje kontrola toka podrazumeva, tako i u drugim granama nauke gde je kontrola toka bitan faktor. U ovom radu predstavljen je način kako bi početnici lako savladali kontrole pristupa uz pozivanje na Bulovu logiku, grafove koji ih prikazuju, takođe i na primere iz života koji to ilustruju.

Cljučne reči: Kontrole toka, Bulova logika, grafovi, poslovni procesi, programiranje

PROGRAMMING TEACHING METHODOLOGY: PROGRAM FLOW CONTROL

Abstract: Control flows are one of the basic techniques that beginners must master and know how to use properly, in program code, in different business processes where the knowledge of control flows implies, also in other branches of science where control flow is an important factor. This paper represents a way for beginners to easily master control flow with Boolean operations, graphs that depict them, and real life examples.

Key Words: Control flows, Boolean operations, graphs, business processes, programming

1. UVOD

Razvoj računarskih nauka i informatike doneo je sa sobom velike promene, kako u industriji, tako i u društvu. U poslednje vreme smo svedoci velikog interesovanja za IT industriju koja se stalno razvija i dolazi sa svih strana. Domaća privreda prati svetsku praksu, teži da svoje biznis modele razvija prateći tehnološke trendove, država u IT industriji vidi jedan od potencijala za razvoj, a međunarodne organizacije i mediji nastoje da isprate sva ova dešavanja. Sve to dovodi do porasta popularnosti ove sfere među ljudima. Roditelji savetuju decu da se školuju za IT sektor, u školama se otvaraju specijalizovana odeljenja za obdarene učenike za računarstvo i informatiku, raste popularnost takmičenja iz programiranja, a programiranje postaje sve poželjnije zanimanje.

U današnje doba, IT sektor predstavlja jedan od ključnih faktora za razvitak privrede na globalnom nivou, što donosi prosperitet mnogim državama širom sveta, pa tako i Srbiji. Još jedna od dobrih posledica do koje dovodi razvoj IT sektora je to što učestvuje i u razvitku drugih grana privrede, poboljšavajući stanje u državi. Da bi IT sektor dostigao svoj zenit razvitka, a sa njim i privreda potreban je velik broj mladih učenih ljudi. Pretpostave su da će se IT kadar razvijati u budućnosti velikom brzinom. To znači da će i potreba za novim stručnim kadrom rasti i biće potreban velik broj IT stručnjaka od kojih će zavisiti brzina razvoja IT sektora kao i njen kvalitet. Zato je i ključno da njihovo obrazovanje i obučavanje bude na visokom nivou [4].

Jedna od osnovnih stvari sa kojima se početnik tokom učenja programiranja susreće jeste kontrola toka. Principi kontrole toka su se koristili kako u prvim programskim jezicima u nekom od oblika, tako i u modernim jezicima. Savladanje kontrole toka je veoma važno, jer kontrola toka predstavlja osnovni pojam iz programiranja koji se javlja u gotovo svim sferama računarstva. Zato je potrebno početniku objasniti kontrole toka na način na koji će ga razumeti i savladati tako da znaju da implementiraju njihove koncepte nezavisno od programskog jezika koji će koristiti.

Postoje dva načina za učenje kontrole toka. Prvi način za učenje kontrole pristupa, jeste direktno kroz primere nekog programskog koda gde je implementirana neka kontrola toka. Drugi način je učenje kontrole pristupa na osnovu teorijskih konceptata uz primere napisane u obliku dijagrama i pseudokoda. Prvi pristup se u velikom broju slučajeva pokazao kao pogrešan, jer se početnik više fokusira na sintaksu programskog jezika u kome je primer napisan, nego na same koncepte kontrole toka. Zbog ovog problema, objašnjavanje kontrola toka preko njihovih principa ima prednost i smanjuje se šansa da se početnik veže samo za neki programski jezik, što je kod drugog načina učenja veoma moguće.

Programi sa kojima se početnici susreću na početku i pomoću kojih uče osnovne stvari, uglavnom imaju linijsku strukturu. Kod takvih programa naredbe se uglavnom izvršavaju redom, onako kako su napisane, jedna za drugom, od početka do kraja, s tim da se svaka naredba izvrši jednom. Nakon jedne izvršene naredbe, prelazi se na sledeću i tako redom do kraja programa. Nemoguće je izraziti sve svoje ideje, za rešavanje problema, koji se rešava linijskom strukturom programa i tada je potrebno uvesti kontrole toka pristupa.

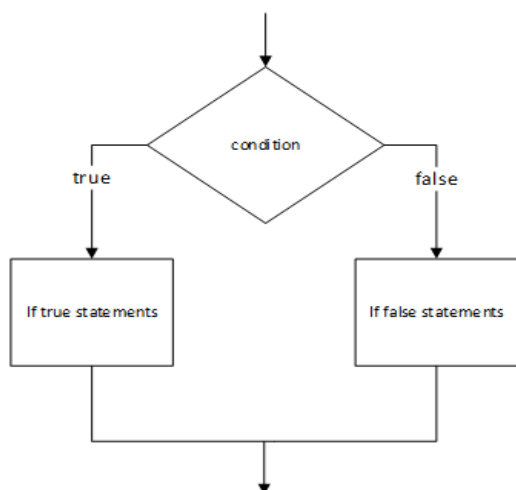
Istraživanjem o konceptima kontrole toka, uočeno je da ne postoji mnogo literature koje početniku mogu da objasne ovako krucijalnu stvar na način na koji će on to najbolje razumeti bez poziva na neke programske jezike. Jedan od primera jeste knjiga „*The C Programming Language*“. Početnici često pokušavaju da nauče osnovne koncepte, u koje spadaju i kontrole toka, preko raznih knjiga koje su usko vezane za neki programski jezik, zbog manjka literature često upadaju u problem da moraju da savladaju prvo sintaksu nekog programskog jezika pa tek onda da savladaju suštinu koja je mnogo važnija i koja se ne mora dobro shvatiti ovakvim načinom učenja. Ovaj rad ističe značaj učenja programiranja tako da je fokus na razjašnjavanju koncepata, a ne sintakse nekog programskog jezika. Ovaj pristup potkrepljen je primerom savladavanja kontrole toka programa Analizom ovog problema, sa kojim se početnici susreću, date su neke osnovne ideje, na osnovu kojih bi korisniku bilo lakše da savlada osnove kontrole toka.

2. METODE KONTROLE TOKA

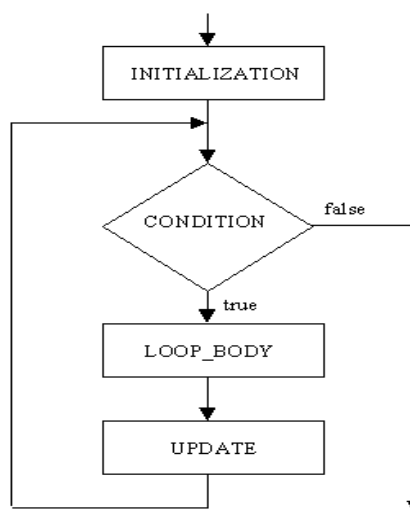
Kontrole toka se mogu podeliti u dve grupe [2]:

- **Grananje (selekcija)** - gde se deo koda programa izvršava uslovno, u zavisnosti od rezultata nekog poređenja (Npr. „Ovaj kod izvrši samo ako je Promenljiva x manja od 10.“). Na slici 1 je prikazan njen grafički prikaz.
- **Petlje (iteracija)** - ponavljanje izvršenja istih iskaza (određeni broj puta, dok se ne dostigne uslov koji je unapred zadat). Na slici 2 je prikazan njen grafički prikaz.

Prvi način predstavljanja algoritama početnicima koji se obično i najčešće koristi jeste korišćenje blok dijagrama algoritma. Razlog za to je to što se nakon objašnjavanja osnovnih celina koje ga čine, dijagram jednostavno i brzo čita. Linije (grane) predstavljaju tok izvršavanja procesa, dok pravougaonici predstavljaju operaciju, odnosno naredbu koja se izvršava. Rombom su predstavljeni uslovi koji diktiraju dalji tok izvršavanja programa. Linije se završavaju sa strelicama koji govore u kom smeru ide proces, pa je početnicima veoma intuitivno da tumače dešavanje u okviru grafova, što pomaže u objašnjenju kontrole toka procesa. Razlika u metodama je u tome što se kod metode grananja ponavlja izvršavanje nekog skupa operacija do ispunjenja uslova, dok kod metode grananja, proces ide u smeru u zavisnost od uslova (bez povratne grane). Zajednički deo obe metode je što zavise od uslova, koji određuje kako će se proces dalje izvršavati. U sledećem poglavlju je dato detaljnije objašnjenje konstruisanja i funkcionisanja uslova.



Slika 1. Grafički prikaz metode grananja



Slika 2. Grafički prikaz metode petlje

Drugi način predstavljanja metoda kontrola toka jeste preko „pseudo koda“ koji ima široku upotrebu pri upoznavanju sa raznim algoritmima i konceptima koje ne zavise od platforme na kojoj su implementirani. Pseudokod je neformalan način opisivanja programa koji ne zahteva nikakvu strogu sintaksu programskog jezika ili temeljna tehnološka razmatranja. Koristi se za kreiranje nacрта ili grubog nacрта programa. Pseudo kod sažima protok programa, ali isključuje osnovne detalje, što i pomaže početniku da lakše shvati ideje bez drugih informacija koje ga ne zanimaju u tom trenutku [3]. Koncept metode grananja napisan u pseudo kodu je dat na slici 3.

Verovatno najbolji način za objašnjavanje kontrole toka početniku jeste objedinjujući prethodno opisane metode. Korišćenje pseudo koda i grafičkog prikaza zajedno, omogućava početniku da prihvati sve osnovne ideje, ali sa dodatkom neke logike koja će se „izvršavati“. Ovakav način, početnika približava objašnjenju ideja kroz sintaksu nekog programskog jezika, prikazujući tok programa preko dijagrama koji je lako pratiti. Ovom metodom početnik će brzo shvatiti ideje kontrole toka, njihovo označavanje u nekom programskom jeziku (sličnost u sintaksi pri pisanju kontrola toka u pseudo kodu i većini programskih jezika), što će mu omogućiti brzu i laku adaptaciju na bilo koji platformu.

Pseudocode

- Or, in general:

```
IF (<CONDITION>)
    THEN <Statements>;
    ELSE <Statements>;
ENDIF;
```

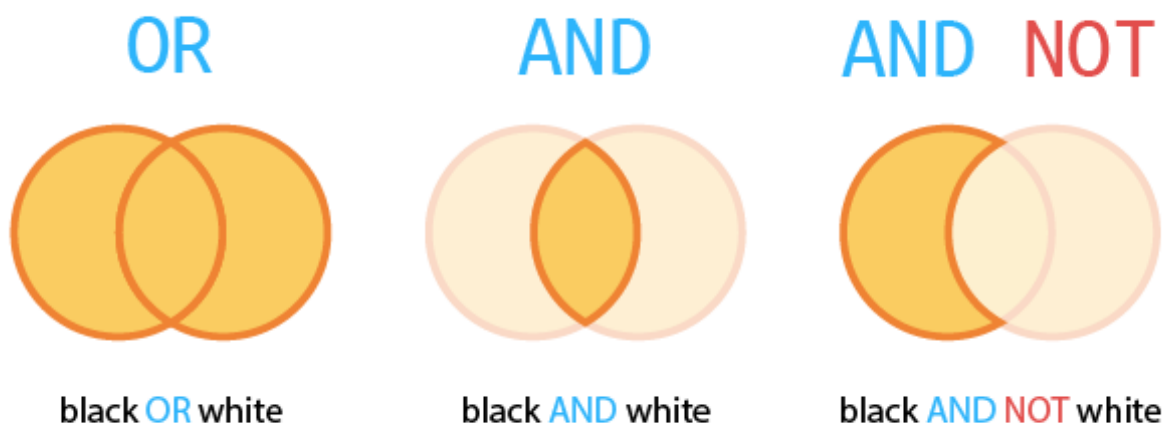
Slika 3. Primer metode grananja kontrole toka u pseudo kodu

3. USLOVI KONTROLE TOKA UZ OBJAŠNJENJE BULOVE LOGIKE

Kao što je već rečeno, obe metode kontrole toka koriste uslove za njihovu realizaciju u procesu. Uslovi uključuju korišćenje Bulove logike, pomoću koje se početnici mogu upoznati sa tehnikama njihovog građenja i funkcionisanja u kontrolama toka.

Bulova logika, odnosno Bulova algebra, deo je matematičke logike koja je dobila ime po engleskom matematičaru Džordžu Bulu, čiji je rad sredinom devetnaestog veka omogućio osnovu Bulove logike. Zbog toga se i u računarstvu uvodi novi tip „bool“ koji može da ima samo dve vrednosti true ili false. Bulova logika ima veliku primenu u svim savremenim programskim jezicima, gde se često koristi da bi zabeležio rezultat neke logičke operacije. Ovo između ostalog, omogućava građenje uslova kontrole toka tako da uslovi mogu da budu i složeniji izraz [1].

Kao što smo već par puta naglasili, u praksi se programiranje često počinje učiti od nekog programskoj jezika, pa se kroz njega provuku i nauče kontrole toka. Važno je naglasiti da nemaju svi programski jezici primitivni tip podataka specijalizovan za Bulovu logiku. Međutim, to nema negativni uticaj na objašnjavanje kreiranja uslova u grananju ili seekciji. Često je taj programski jezik „C“ ili neki slični jezik koji ne podržava „bool“ tip podataka. Zato je potrebno povući paralelu između primitivnog tipa „int“ i „bool“. U ovakvim programskim jezicima se bool vrednost „true“ zamenjuje sa int vrednošću „1“, dok se vrednost „false“ zamenjuje sa „0“. Važno je učenike ili studente naučiti da razlikuju upotrebu „int“ tipa podataka kao celobrojne vrednosti i „int“ tipa podataka koji ima dve moguće vrednosti: netačno (vrednost 0) i tačno (sve različito od 0).



Slika 4: Bulove operacije na primeru skupova.

Na slici 4 su prikazane Bulove operacije koje se mogu koristiti u uslovima kontrole toka. Važno je napomenuti, da uslov može da sadrži više delova. Svaki deo se gradi samostalno upotrebom bulovih ili relacionih operatora. Zadovoljenje uslova zavisi od svih njegovih delova.

Kao primer, razmotrićemo situaciju u kojoj izvršimo neku operaciju, u procesu, na osnovu toga da li je promenljiva 'x' manja od 10 i 'y' veći ili jednak sa 5 ($x < 10$ & $y \geq 5$). Da bi se ovo uradilo, potrebno je proveriti da li je iskaz „x je manje od 10” istinit ili neistinit. Takođe, potrebno je proveriti da li je i drugi iskaz istinit ili neistinit. Tačnije, mora se znati vrednost rezultata tog poređenja. U ovom primeru uslov se sastoji od dva dela koji su povezani sa Bulovim operatorom (u ovom primeru je to "I" operator). Uslov je ispunjen ako su oba dela tačna.

4. ZAKLJUČAK

Uz koncepte, grafove i primere iz nekog realnog procesa u radu je dato objašnjenje kontrole pristupa. Uz ovakav metod učenja, početnik može veoma lako savladati kontrole pristupa i steći osećaj koliko su kontrole toka bitne u poslovnim procesima, programiranju, kao i u različitim dijagramima. Početnicima je potrebno objasniti osnovne delove „programiranja” na nekom konceptualnom nivou, uz oslonac na njihovo prethodno znanje, korišćenjem raznih matematičkih teorema, grafovskim prikazima i raznim primerima. Prikaz primera u nekom programskom jeziku i njegovo objašnjenje je često pogrešan pristup kada je reč o podučavanju početnika. Početnik se tada više fokusira na sintaksu programskog jezika nego na suštinu koja je mnogo bitnija.

5. LITERATURA

- [1] Kontrola toka programa - <https://www.bpa.edu.rs/FileDownload?filename=1860a04c-29b2-4979-bc64-29451480ce79.pdf&originalName=7-Kontrola%20toka%20programa.pdf>
- [2] Norbert Zeh, *Control Flow - Principles of Programming Languages*, Dalhousie University, 2018god.
- [3] Pseudocode - <https://economictimes.indiatimes.com/definition/pseudocode>
- [4] Šta je potrebno srpskoj IT industriji - <https://startit.rs/sta-je-potrebno-srpskoj-it-industriji/>