

OSNOVI PROGRAMIRANJA

Dr Dinu Dragan, dinud@uns.ac.rs

Dr Dušan Gajić, dusan.gajic@uns.ac.rs

Milan Pisarić, piskem@gmail.com



Verzija 1.01

TEME IZUČAVANJA

Teme izučavanja

1. Kratak pregled organizacije računara
2. Proces razvoja programa – apstrakcija, tipovi podataka i programski kod
3. Algoritmi
4. Pregled Visual Studio okruženja
5. Struktura programa
6. Tipovi podataka C# programskog jezika
7. Operatori i izrazi
8. Programske upravljačke strukture
9. Nizovi i matrice
10. Potprogrami (metode) – parametri
11. Mnogo rešavanja problema i vežbanja

Potreban softver




- Visual Studio Community edition 2017
 - Integrirano razvojno okruženje (Integrated Development Environment - IDE)
 - Preuzeti sa: <https://www.visualstudio.com/vs/community/>
- Microsoft .NET Framework 4.6 (uključeno u gornjoj instalaciji)
 - Kolekcija programskih alata koja sadrži sve što je potrebno da bi se napravio i pokretao C# program (ili bilo koji Windows program)
 - (Po potrebi) Preuzeti sa:
 - <https://www.microsoft.com/en-us/download/details.aspx?id=48137>

Instalacija Visual Studio


Visual Studio


Products

Installed

 Visual Studio Community 2017
Acquiring Microsoft.VisualStudio.CoreEditor
1% 
Starting operation
0% 

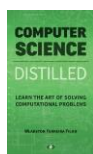
Available

 Visual Studio Enterprise 2017
15.5.4
Microsoft DevOps solution for productivity and

 Visual Studio Professional 2017
15.5.4
Professional developer tools and services for small teams

Literatura

1. **Brošura sa slajdovima, prateći kod + vaše beleške**
2. Joseph Albahari - "C# 6.0 in a Nutshell: The Definitive Reference", 6th Edition, November 2015
3. Troelsen, Andrew - "Pro C# 5.0 and the .NET 4.5 Framework", 6th Edition, 2012
4. Rob Miles "C# Programming Yellow Book", Cheese Edition 8.2, November 2016, <http://www.csharpcourse.com/>
5. Wladston Ferreira Filho, "[Computer Science Distilled: Learn the Art of Solving Computational Problems](#)", Code Energy LLC, Las Vegas, 2017.



Prekvalifikacije za IT

Forum

- Forum se nalazi na adresi: <http://it-girls2.forumotion.me/>
Prilikom prve posete, potrebno je napraviti nalog izborom opcije *Register* u gornjem desnom uglu ekrana

The screenshot shows the Forumotion website interface. At the top, there is a navigation bar with links for Home, Calendar, FAQ, Search, Memberlist, Usergroups, Register, and Log in. Below the navigation bar, the current date and time are displayed as 'Wed Sep 06, 2017 9:11 pm'. The main content area contains a login form with fields for 'Username' and 'Password', a 'Log in' button, and a checkbox for 'Log in automatically'. Below the login form, there is a 'Register' section with a 'Register' button. The page has a dark blue header and a light gray background.

Prekvalifikacije za IT

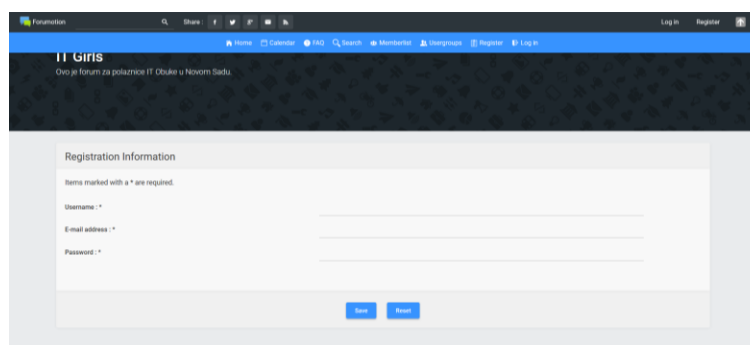
Forum

- Nakon izbora opcije registracije, potrebno je da prihvatite uslove korišćenja foruma izborom opcije *I agree to these terms*

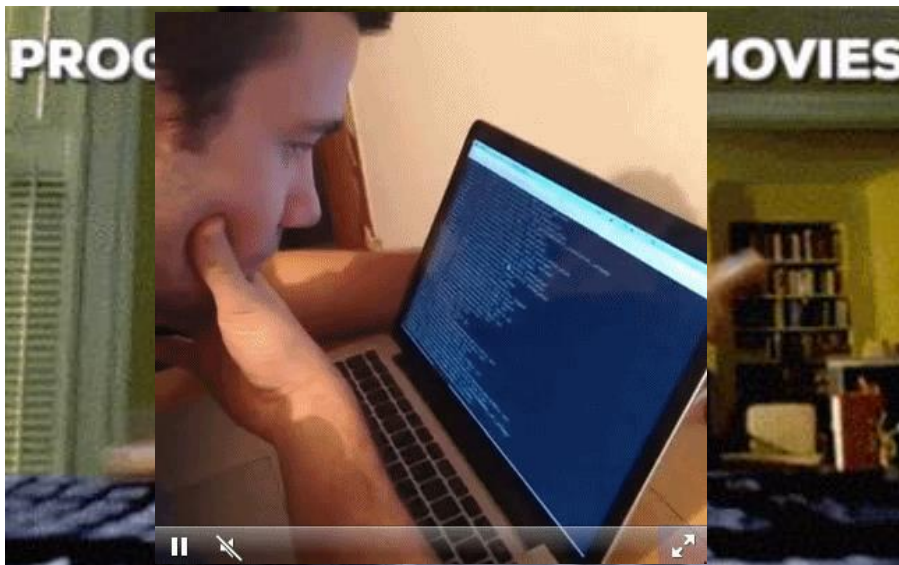
The screenshot shows the 'Forum Terms of service' page. The page has a dark blue header with the same navigation bar as the previous screenshot. The main content area contains the text of the terms of service, which includes a disclaimer about the moderators' responsibility and a list of rules. At the bottom of the page, there are two buttons: 'I agree to these terms' and 'I do not agree to these terms'. The page has a light gray background and a dark blue header.

Forum

- Potom izaberete korisničko ime (*username*), e-mail adresu i šifru (*password*) i izabere Save. Nakon potvrde šifre i verifikacije, koji se zahtevaju na sledećem ekranu, na e-mail ćete dobiti poruku sa linkom za aktivaciju naloga. Nakon aktivacije, Vaš nalog je spreman!

A screenshot of a web browser showing the registration page of a forum titled "IT Girls". The page has a blue header with navigation links like Home, Calendar, Search, Members, Usergroups, Register, and Log In. Below the header, there's a dark banner with the forum title and a subtitle. The main content area is titled "Registration Information" and contains a form with three input fields: "Username: **", "E-mail address: **", and "Password: **". A note above the fields says "Items marked with a * are required." At the bottom of the form are two buttons: "Save" and "Reset".

UVOD



Pisanje programa

- Nije lako napisati program
- Morate reći računaru šta treba da radi i pri tom mu opisati svaki korak koji treba da se izvrši
- Nema mesta za greške, jer računar ne zna da interpretira, slepo izvršava svaku naredbu
- Dobar program je rezultat razmišljanja na pravi način
- **Program nije ništa drugo nego formalno izražena ideja**
 - Kreće se od ideje šta se želi (šta program/računar treba da radi)
 - Okvir onog što će se (što treba da se) uradi, izražava se kroz algoritam
 - Algoritam omogućuje ne samo da se izrazi ideja o zadatku, nego i o koracima koji se moraju preći da bi zadatak rešio (iz čega proizilazi program)

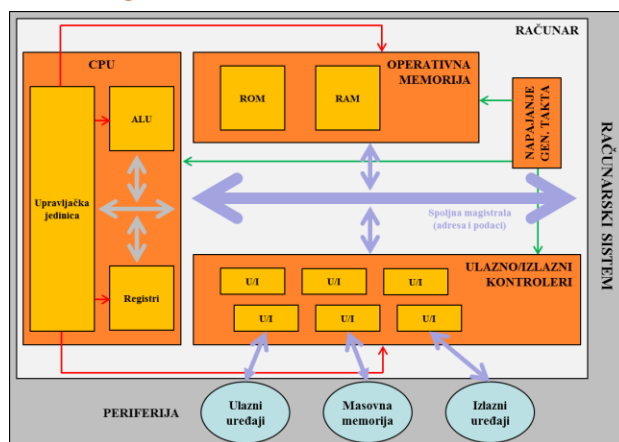
Algoritam ...

- Algoritam je precizno definisana procedura za rešavanje nekog problema.
- Računarstvo je nauka o algoritmima:
 - njihovim formalnim osobinama
 - tačnost, ograničenja, efikasnost, cena
 - njihovim hardverskim realizacijama
 - projektovanje računarskih sistema
 - njihovim jezičkim realizacijama
 - programiranje i programski jezici
 - njihovim primenama
 - inženjerstvo, astronomija, fizika, biologija, hemija...

... Algoritam

- Kada se opisuje algoritam, ne moraju svi delovi (koraci) da budu do detalja opisani (specificirani), sve dok su oni determinisani i dok vode ka jasnom rešenju
- Mogu se pisati na bilo kom jeziku, mogu biti grafički i/ili tekstualni
- Popunjavanjem svih detalja algoritam postaje program

Organizacija računara



- Brzi procesor/memorija naspram sporih izlazno/ulaznih uređaja, preprogramirani set instrukcija
- Računar nije pametan, samo je dobar u brzom ponavljanju rutina

Programiranje

- **Programiranje == davanje uputstva računaru**
- Procesorske (engl. Central Processing Unit – CPU) instrukcije:

c = a + b

Pročitaj vrednost sa lokacije **a**

Pročitaj vrednost sa lokacije **b**

Saberi

Upiši vrednost na lokaciju **c**

- Programске jezike je lakše razumeti nego CPU instrukcije
- Potrebno je prevesti kod na CPU instrukcije da bi ga CPU razumeo

Generacije programskih jezika ...

1. GL – MAŠINSKI JEZICI

- programiranje korišćenjem binarnog koda (0 i 1)
- uporedo sa prvim komercijalnim računarima
- komande zavise od arhitekture procesora, nema portovanja
- potrebna ekspertiza u u elektronici i digitalnim sistemima
- svi programski jezici se na kraju prevode na mašinski

2. GL – ASEMBLERSKI JEZICI

- simboli ili mnemonici koji odgovaraju instrukcijama mašinskog jezika
- prvi put se javlja kompajliranje (prevođenje na mašinski jezik)
- manja zavisnost od hardverske platforme
- i danas se koriste za pisanje pojedinih delova sistemskog softvera (! C)

... Generacije programskih jezika

3. GL – JEZICI VIŠEG NIVOA (KOMPJILERSKI I INTEPRETERSKI)

- nezavisni od fizičke arhitekture
- dorađena verzija jezika druge generacije (proširen alfabet)
- jedna naredba prevodi se u više naredbi mašinskog jezika
- jezičke strukture – utvrđena struktura programskog koda
- podrazumevaju kompajler (C) ili interpreter (Java)

4. GL – PROBLEM ORIJENTISANI JEZICI (SQL)

- jezici posebne namene / problem orijentisani jezici (application specific)
- zahtevaju min. programersko znanje, ali je primena usko specifična
- neproceduralni jezici visokog nivoa izgrađeni oko baze podataka (SQL)

PROCES RAZVOJA PROGRAMA

Proces razvoja programa

„Kako izuzetno kompleksne, neformalne i često protivrečne realne probleme čoveka i njegovog okruženja rešavati (automatizovati) pomoću funkcionalno ograničene ali zato brze mehaničke (formalne) mašine – računara?„

„Etapnom transformacijom polaznog problema (njegove statike i dinamike) u objekte koji su interesantni sa date tačke gledišta (apstrakcija) a koji se mogu matematički (aritmetički i logički) rešiti – jezik računara!“

Razvoj jednostavnih programa ...

1. ANALIZA PROBLEMA

- šta program treba da radi?

IDENTIFIKOVATI PROBLEM – opisati problem u nekoliko rečenica, šta se očekuje od programa.

ODREĐIVANJE ULAZA I IZLAZA – koje podatke program treba da dobije da bi obavio željenu radnju i koje podatke će program vratiti korisniku (uvek barem jedan izlaz)

2. FUNKCIONALNA DEKOMPOZICIJA PROBLEMA

- razlaganje problema na potprobleme, do elementarnih problema koji se znaju rešiti i implementirati
- usputno se identifikuju potrebni podaci

... Razvoj jednostavnih programa ...

3. RAZVOJ ALGORITMA

- postupak (način razmišljanja) kojim se iz datih polaznih podataka, određenim konačnim nizom računskih i/ili logičkih postupaka dolazi do traženih rezultata
- osobine: diskretnost, rezultativnost, determinisanost, masovnost, (i) **efikasnost**
- **primeri:** recept, uputstvo za sklapanje, uputstvo...
- tekstualne ili grafičke notacije za opis algoritama
- nezavisno od programskog jezika (ili nametnuto programskim jezikom)
- piše se u skladu sa projektom razrađenim u prethodnim fazama

... Razvoj jednostavnih programa

4. PROGRAMIRANJE

- pisanje programa prema algoritmu (preslikavanje 1:n)
- kompajlerski ili interpreterski jezici

5. TESTIRANJE

- izvršavanje programa nad test podacima s ciljem da se pronađu greške
- jedinično testiranje
- integrisano testiranje
- sistemsko testiranje

Primeri rešavanja problema

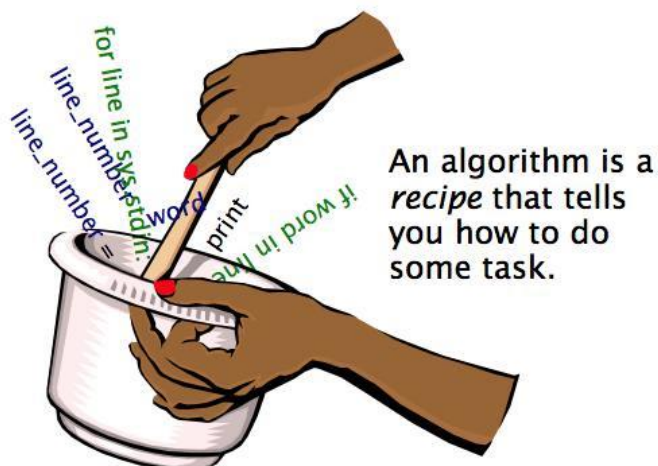
- **Primer 1.** Kalkulator celih brojeva za sabiranje i oduzimanje.
 - Šta su ulazi i izlazi?
 - Kako čuvati podatke?
 - Postoji li potreba za dekompozicijom?
- **Primer 2.** Sortiranje spiska zaposlenih po imenu ili jmbg-u.
 - Šta su ulazi i izlazi?
 - Kako čuvati podatke?
 - Postoji li potreba za dekompozicijom?

Apstrakcija u razvoju programa

- “Koncept ignorisanja svih svojstva subjekta koja nisu relevantna za tekuću svrhu sa ciljem koncentrisanja samo na ona koja to jestu”
- Primeri:
 - ČOVEK => apstrakcija STUDENT
 - ČOVEK => apstrakcija LEKAR
 - ČOVEK => apstrakcija VOZAČ

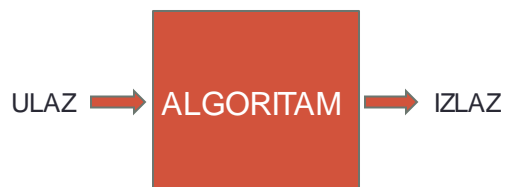
ALGORITMI

Algorithms



Pojam algoritma

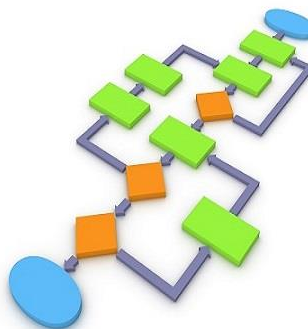
ALGORITAM = UPUTSTVO = RECEPT



- **Algoritam** je precizno definisan postupak sa konačnom listom koraka za rešavanje nekog problema. **Algoritam** prihvata **ulazne vrednosti** i proizvodi **izlazne vrednosti**.

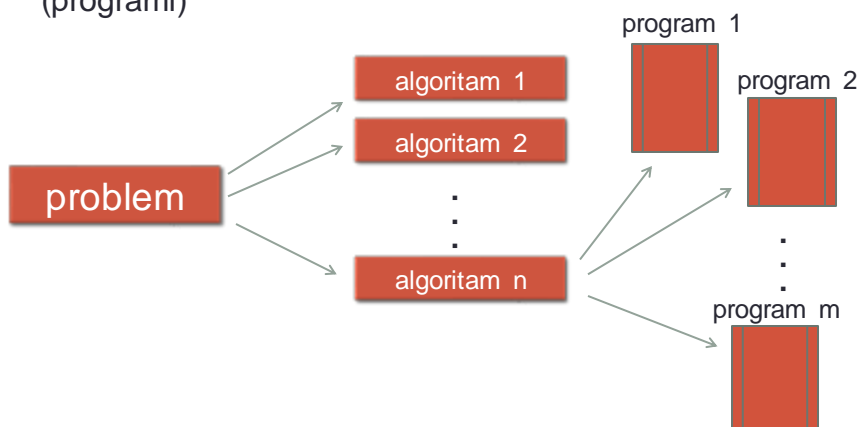
Primeri algoritama

1. KUVANJE
2. KRETANJE PO GRADU
3. MNOŽENJE
4. ODREĐIVANJE NZD
5. PROGRAMIRANJE



Problem, algoritam i program

- Različiti algoritmi kao rešenja istog problema
- Jedan algoritam može imati više različite implementacije (programi)



Osobine algoritma

- Podrazumeva se da algoritam mora prvo da bude ispravan (tačan)!
 1. Diskretnost
 2. Determinisanost
 3. Efektivnost (konačnost)
 4. Rezultativnost
 5. Generičnost (masovnost)
 6. (Optimalnost)

Diskretnost

- Algoritam se sastoji od konačnog broja koraka
- Svaki korak zahteva obavljanje jedne ili više operacija
- U zavisnosti od operacija koje računar može da obavi, uvode se ograničenja za tip operacija koje se mogu koristiti u algoritmu

Determinisanost

- Svaki algoritamski korak mora biti precizno definisan i potpuno jasan
 - Izrazi „13/0“ ili „oduzmi 4 ili 5 od trenutne vrednosti proizvoda“ nisu dozvoljeni
- Posle izvršavanja tekućeg koraka u algoritmu mora biti jednoznačno određeno koji je sledeći korak

Determinisanost

- Svaki algoritamski korak mora biti precizno definisan i potpuno jasan
 - Izrazi „13/0“ ili „oduzmi 4 ili 5 od trenutne vrednosti proizvoda“ nisu dozvoljeni
- Posle izvršavanja tekućeg koraka u algoritmu mora biti jednoznačno određeno koji je sledeći korak

Efektivnost (konačnost)

- Sve operacije koje se javljaju u algoritamskim koracima moraju se izvršiti za konačno (razumno kratko) vreme i moraju biti dovoljno jednostavne da se mogu tačno izvršiti
- Vreme izvršavanja celog algoritma mora biti konačno, tj. prihvatljivog trajanja

Rezultativnost

- Svaki algoritam mora posle konačnog broja koraka generisati traženi rezultat
- Algoritam može imati nula ili više ulaznih podataka, a može generisati jednu ili više izlaznih vrednosti

Generičnost (masovnost)

- Svaki algoritam definiše postupak za rešavanje klase problema, a ne pojedinačnog slučaja
- Pretraživanje ili sortiranje bilo kog podskupa celih ili realnih brojeva, množenje matrica/vektora bilo kog reda...

(Optimalnost)

- Razlika između odličnog, dobrog, upotrebljivog i lošeg koda
- Ono što pravi razliku među programerima
- U nekim slučajevima nije prioritet
- U nekim slučajevima nije ostvarivo
- U nekim slučajevima izuzetno važno!

Načini predstavljanja algoritama

1. **Tekstualni opis na prirodnom jeziku**
2. **Grafički (pomoću dijagrama toka)**
3. Pseudokod
4. (Strukturogram)
5. Programski jezik

Tekstualni opis algoritama

- Koriste se precizne rečenice govornog jezika
- Koristi se za lica koja se prvi put sreću sa pojmom algoritma
- Dobra osobina: razumljivost za širi krug ljudi
- Loše: nepreciznost koja proističe iz same prirode jezika

Tekstualni opis algoritama – primer 1

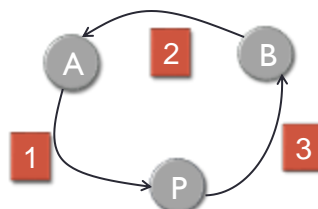
Euklidov algoritam za nalaženje **NZD** dva prirodna broja **a** i **b**

1. Podeliti **a** sa **b** i ostatak zapamtiti u **r**
2. Ako je **r** jednako **0**, **NZD** je **b** i došli smo do kraja algoritma, **inače** preći na **korak 3**
3. Zameniti **a** sa **b**, **b** sa **r**, i preći na **korak 1**

Tekstualni opis algoritama – primer 2

Zameniti sadržaje dve memorijske lokacije **A** i **B**

- Rešenje – potrebna je treća, pomoćna, lokacija
1. sadržaj lokacije **A** zapamtiti u pomoćnoj lokaciji **P**
 2. sadržaj lokacije **B** zapamtiti u lokaciji **A**
 3. sadržaj lokacije **P** zapamtiti u lokaciji **B**
 4. kraj



Tekstualni opis algoritama – primer 3

Pomeranje sadržaja lokacija ulevo

- Ciklički pomeriti u levo sadržaje lokacija **A**, **B** i **C**

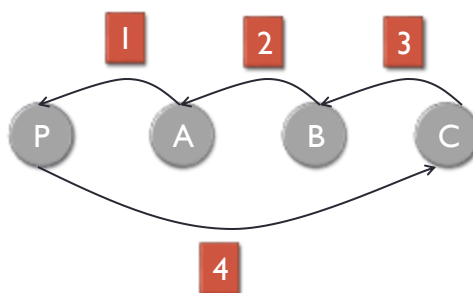
1. iz **A** u **P**

2. iz **B** u **A**

3. iz **C** u **B**

4. iz **P** u **C**

5. kraj



Tekstualni opis algoritama – zadatak

1. Računanje maksimuma tri broja