



IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

ISTORIJA INFORMACIONIH TEHNOLOGIJA

Lekcija 12

PRIRUČNIK ZA STUDENTE

IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

Lekcija 12

ISTORIJA INFORMACIONIH TEHNOLOGIJA

- ▼ ISTORIJA INFORMACIONIH TEHNOLOGIJA
- ▼ Poglavlje 1: Internet
- ▼ Poglavlje 2: Istorija Interneta
- ▼ Poglavlje 3: Rast Interneta tokom godina i razlozi za enorman rast
- ▼ Poglavlje 4: Interakcija čovek-računar
- ▼ Poglavlje 5: Istorija razvoja interfejsa
- ▼ Poglavlje 6: Pokazna vežba: JS & HTML
- ▼ Poglavlje 7: Zadaci za samostalni rad: JS & HTML
- ▼ Poglavlje 8: Domaći zadatak
- ▼ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

▼ Uvod

UVOD

U ovoj lekciji ćemo dati istorijski pregled razvoja informacionih tehnologija

U ovom predavanju biće reči o razvoju informacionih tehnologija kroz istoriju. Biće reči o sledećim temama:

- Razvoj interneta
- Razvoj korisničkog interfejsa

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 1

Internet

ŠTA JE TO INTERNET?

Internet je globalni sistem umreženih računara, podataka i korisnika

Prema definiciji U.S. Federal Networking Council [Internet](#) je globalni informacioni sistem koji:

- je logički povezan globalnim jedinstvenim adresnim prostorom baziranim na Internet protokolu (IP) ili njegovim proširenjima ili naslednicima
- je sposoban da podrži komunikacije korišćenjem protokola za upravljanjem prenosa (TCP/IP)
- obezbeđuje, koristi ili čini dostupnim, javno ili privatno, usluge visokog nivoa bazirane na komunikacijama i odgovarajućoj infrastrukturi.

Pojednostavljeno rečeno, **Internet je globalni sistem umreženih računara, podataka i korisnika**. Zahvaljujući Internetu korisnici mogu danas da komuniciraju sinhrono i asinhrono uz zanemarljive troškove. Internet je omogućio da se skupi telefonski razgovori zamene jeftinom Internet telefonijom, a spora zemaljska pošta zameni elektronskom poštom. Sastanci udaljenih korisnika su postali mogući i bez putovanja zahvaljujući tehnologiji video konferencija preko Interneta.

Kamen temeljac Internetu udaren je razvojem paketnog prenosa podataka ([packet switching](#)) ranih 60-tih godina prošlog veka. Do tada su se podaci prenosili komutiranim linijama ([circuit switching](#)) što je zahtevalo da se između izvorišne i odredišne tačke uspostavi stalna hardverska veza koja ostaje u upotrebi sve dok se ne završi komunikacija. Pri paketnom prenosu podaci se na izvorišnoj strani dele na male pakete podataka i obeležavaju podacima o pošiljaocu i primaocu. Paketi se jedan za drugim šalju preko mreže da bi se na odredišnoj strani ponovo sakupili u inicijalnu celinu. Ako neki paket ne stigne na odredišnu stranu u ispravnom obliku zbog bilo koje greške u komunikaciji, od pošiljaoca se zahteva da ponovo pošalje isti paket. Pored toga pojedini paketi mogu da budu preneti do odredišne strane različitim komunikacionim putevima.

KRATKA ISTORIJA INTERNETA NAPISANA OD STRANE NJENIH OSNIVACA

Barry M. Leiner, Vinton G. Cerf, David D. Clark, Robert E. Kahn, Leonard Kleinrock, Daniel C. Lynch, Jon Postel, Larry G. Roberts, Stephen Wolff

The Initial Internetting Concepts

The original ARPANET grew into the Internet. Internet was based on the idea that there would be multiple independent networks of rather arbitrary design, beginning with the ARPANET as the pioneering packet switching network, but soon to include packet satellite networks, ground-based packet radio networks and other networks. The Internet as we now know it embodies a key underlying technical idea, namely that of open architecture networking. In this approach, the choice of any individual network technology was not dictated by a particular network architecture but rather could be selected freely by a provider and made to interwork with the other networks through a meta-level "Internetworking Architecture". Up until that time there was only one general method for federating networks. This was the traditional circuit switching method where networks would interconnect at the circuit level, passing individual bits on a synchronous basis along a portion of an end-to-end circuit between a pair of end locations. Recall that Kleinrock had shown in 1961 that packet switching was a more efficient switching method. Along with packet switching, special purpose interconnection arrangements between networks were another possibility. While there were other limited ways to interconnect different networks, they required that one be used as a component of the other, rather than acting as a peer of the other in offering end-to-end service.

In an open-architecture network, the individual networks may be separately designed and developed and each may have its own unique interface which it may offer to users and/or other providers, including other Internet providers. Each network can be designed in accordance with the specific environment and user requirements of that network. There are generally no constraints on the types of network that can be included or on their geographic scope, although certain pragmatic considerations will dictate what makes sense to offer.

.....

Ostatak članka pogledajte na

<http://www.internetsociety.org/internet/what-internet/history-internet/brief-history-internet>

▼ Poglavlje 2

Istorija Interneta

ARPANET – MAJKA INTERNETA

Prvobitna internet mreža je imala samo 4 čvora

Godine 1969. Bolt, Beranek, and Newman, Inc. (BBN) je projektovala za potrebe Ministarstva odbrane U.S. mrežu koja je Advanced Research Projects Agency Network ([ARPANET](#)). Ideja projekta je bila da se istraživačima omogući deljenje resursa super kompjutera. Inicijalno, mreža je imala samo 4 čvorova. Oni su bili locirani u:

- University of California u Los Angeles-u,
- University of California u Santa Barbari,
- University of Utah,
- Stanford Research Institute.

ARPANET je prvi put demonstriran 1972. godine na International Conference on Computers koja je održana u Vašingtonu. Tada je prvi put javno demonstriran paketni prenos podataka između dva računara. Tokom sledeće dekade na ARPANET je vezano mnogo računara iz univerzitetskih centara i velikih firmi, kao što su IBM, tako da ih je 1983 bilo preko 300. Iste godine je vojni deo mreže odvojen u MILNET u Americi i MINET u Evropi. Sedamdesetih godina stvorene su mnoge lokalne računarske mreže različitih arhitektura koje su želele da se međusobno povežu. To je bio povod da se promoviše ideja otvorene arhitekture, tj. da mreže koje mogu da se povežu na ARPANET mogu da budu bilo koje arhitekture. Brzina prenosa podataka između pojedinih čvorova išla je do 50 Kb/s.

Od 1972. godine ARPANET se proširio van granica U.S. povezujući University College u Londonu i Royal Radar Establishment u Norveškoj. Te godine je Ray Tomlinson, koji je radio za BBN izmislio email tako da je ARPANET masovno počeo da se koristi za komunikaciju. Godine 1974, Bolt, Beranek and Newman su objavili još jedan Internet servis, Telenet, koji je omogućio pristup aplikacijama na udaljenim računarima. Godine 1979. je startovan [User network](#) (USENET) koji je ponudio servis UUCP (UNIX to UNIX CoPy), novosti i grupne e-mail-ove. Stvarane su grupe korisnika koje su bile zainteresovane za vesti iz specifičnih oblasti (newsgroup) koje su primale takozvane NetNews. Na taj način su počele da se stvaraju Internet zajednice.

Sledeći korak ka Internetu je bio odluka ARPANET-a iz 1980. godine da prihvati TCP/IP set protokola kao set pravila koja će se koristiti na mrežama vezanim za ARPANET. Ovi protokoli su bili uključeni u operativni sistem Berkeley UNIX, koji je bio široko rasprostranjen na univerzitetima. Na taj način su protokoli postali nezavisni od velikih proizvođača računara kao što su IBM i DEC i time postali dostupni svima.

VIDEO - ISTORIJA ARPANET-A

History Of The Internet

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

VIDEO - TIM KOJI JE IZGRADIO ARPANET

ARPAnet - the team behind the internet

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

INTERNET DOBA

80-tih godina izraz Internet je počeo da se koristi da bi se opisalo da je ARPANET u stvari mreža lokalnih mreža

80-tih godina izraz Internet je počeo da se koristi da bi se opisalo da je ARPANET u stvari mreža lokalnih mreža. Pored istraživačkih institucija na Internet počinju da se priključuju i druge institucije i organizacije. Sa širenjem broja korisnika javlja se potreba za uvođenjem sistema imena domena (DNS - **Domain Name System**), što je olakšalo dalje širenje Interneta.

Snažan doprinos razvoju Interneta dala je i američka National Science Foundation koja je od 1987. godine finansirala razvoj brzih magistralnih veza (engleski **backbone**) između pojedinih IT centara u Americi. Regionalne mreže koje su se povezale na magistrale imale su sada vrlo brzu vezu do udaljenih računarskih mreža. Kasnije su i druge U.S. federalne agencije izgradile svoje magistrale koje su bile povezane sa postojećim. ARPANET je ugašen 1989. Od 1995. godine komercijalni provajderi Internet usluga su preuzeli upravljanje glavnim magistralama i Internet je nastavio da raste.

Tokom 90-tih godina mnoge komercijalne organizacije su se priključile na Internet. Razvoj HTML jezika i HTTP protokola omogućio je da se Internet koristi kao infrastruktura za mnoge nove servise.

VIDEO - POČETAK UMREŽAVANJA

Networking the Nerds

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

RAZVOJ VEB SERVISA

Navođenje razvoja veb servisa hronološkim redom

Nadalje se navode hronološkim redosledom samo neki:

- **Gopher, 1991** Gopher je razvijen na University of Minnesota sa idejom da se preko Interneta, korišćenjem menija dođe do željenih datoteka. Sa razvojem veba ovaj servis je izgubio na značaju.
- **World Wide Web, 1991** - WWW je kreiran od strane Tim Berners-Lee na CERN-u kao jednostavan način za publikovanje informacija na Internetu. Od 1992. godine WWW je bio javno dostupan.
- **Pristup Internetu, 1992** - Kompanija Delphi Forums, Cambridge, MA, je ponudila svim zainteresovanim pun pristup Internetu i tako postala prvi Internet provajder
- **Mosaic, 1993**—Mosaic je bio prvi grafički čitač za veb. Razvio ga je Marc Andreessen sa nekolicinom studenata Univerziteta u Illinoisu. Prva verzija Mosaic-a je napisana za X Windows grafičko okruženje UNIX-a.
- **Netscape Communications, 1994**—Ponesen uspehom Mosaic-a, Marc Andreessen i Jim Clark formiraju firmu Netscape Communications, koja na bazi Mosaic-a objavljuje veb čitač Netscape Navigator 1.0.
- **Yahoo!, 1994**—Studenti Univerziteta u Stanford-u David Filo i Jerry Yang su razvili mašinu za pretraživanje Interneta i imenik web strana koga su nazvali Yahoo.
- **Java, 1995**— Sun Microsystems je objavio prvu verziju programskog jezika Java, koji je projektovan sa idejom da služi za razvoj veb aplikacija.
- **Microsoft Internet Explorer, 1995**—Microsoft je prvu verziju svog čitača razvio tek 1995. godine, ali je istovremeno započeo veliki rat veb čitača koji i danas traje.
- **Preko 55 miliona Internet hostova, 1999**—Ovo je era eksplozivnog rasta Interneta
- **Google indeksira preko 1.3 milijarde Web stranica, 2001**—Ogroman broj veb sajtova i nagli porast broja stranica na njima dovodi do potrebe da se informacije objavljene na Internetu kritički procenjuju.
- **Bežični uređaji, 2001**—Neki servisi Interneta, kao što su WWW i e-mail, postaju dostupni i preko mobilnih uređaja.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

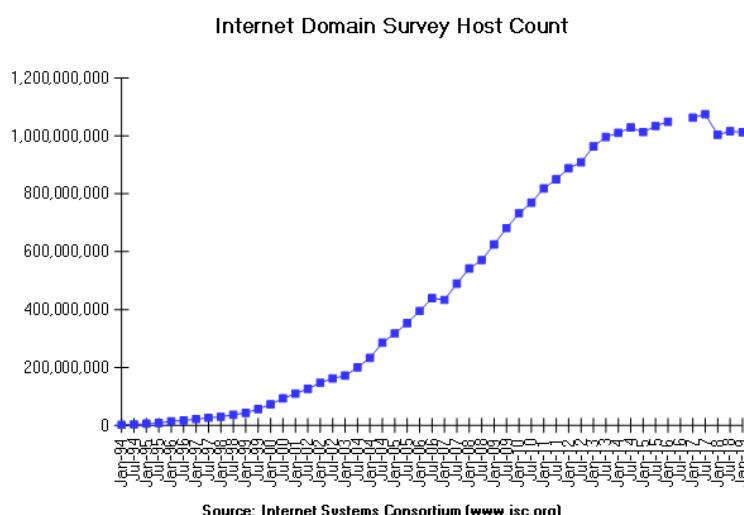
▼ Poglavlje 3

Rast Interneta tokom godina i razlozi za enorman rast

INTERNET KORISNICI I BROJ HOSTOVA NA INTERNETU

Nakon linearog rasta do 1999. godine, počinje eksponencijalni rast broja hostova na internetu

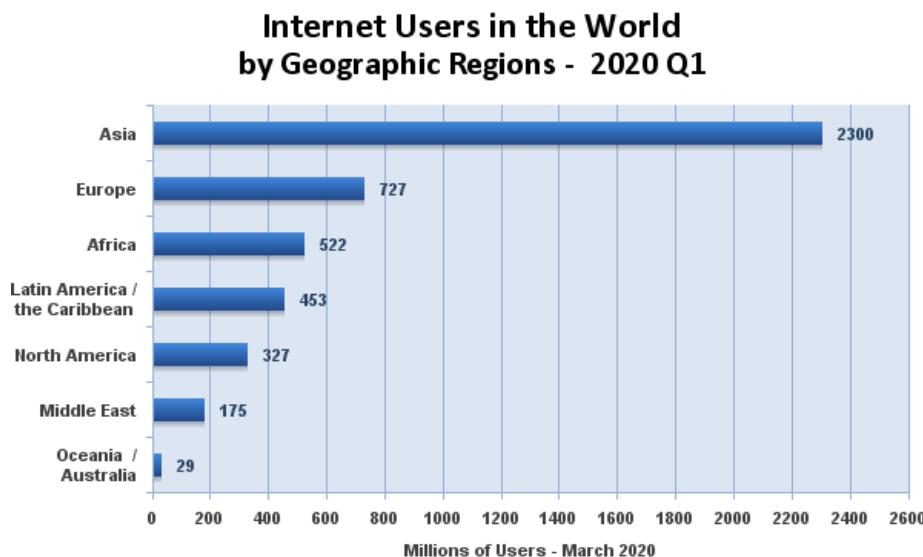
Razvoj Interneta i primena njegovih servisa se može smatrati eksponencijalnim. Na narednom grafikonu je prikazana promena broja čvorova (ili hostova) priključenih na Internet od 1994. do 2015. godine. Očigledno je, da nakon linearog rasta do 1999. godine, počinje eksponencijalni rast.



Slika 3.1 Broj hostovanih domena na svetu

. Internet System Consortium, <https://www.isc.org/survey/> (01.2019.)

Procenjuje se da je 2004. godine Internet koristilo 745 miliona korisnika, 2007. oko 1,32 milijarde, 2010. oko 1,97 milijarde korisnika, a 2012 2,26 milijardi korisnika. Procenjuje se da je broj korisnika Interneta u Srbiji u 2007. godini bio 1,4 miliona, a u 2010. Oko 4,11 miliona.



Source: Internet World Stats - www.internetworldstats.com/stats.htm

Basis: 4,574,150,134 Internet users estimated in March 3, 2020

Copyright © 2020, Miniwatts Marketing Group

Slika 3.2 Broj internet korisnika u svetu

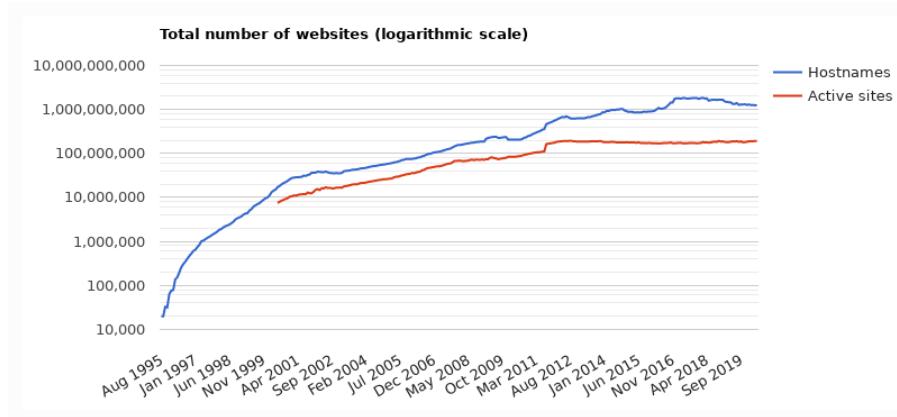
. IWS, [https://internetworldstats.com/stats.htm](http://internetworldstats.com/stats.htm) (03.2020.)

RAZLOZI ZA USPEH INTERNETA

Očigledna je saglasnost promene broja Internet korisnika i broja veb sajtova.

Broj korisnika Interneta naglo se povećao sa razvojem Web-a. Na sledećoj slici prikazana je promena broja veb sajtova u periodu od 1995. do 2010. Tako je krajem 2007. godine na Internetu bilo oko 150 miliona sajtova, pri čemu je samo u 2007. godini postavljeno preko 40 miliona novih sajtova, a krajem 2010. godine ukupan broj sajtova u svim domenima je bio oko 256 miliona.

Očigledna je saglasnost promene broja Internet korisnika i broja veb sajtova.



Slika 3.3 Ukupan broj sajtova u svim domenima

. Netcraft, <https://news.netcraft.com/archives/2020/05/26/may-2020-web-server-survey.html>
(05.2020.)

Razlozi za enorman uspeh Interneta leže u sledećem:

- Sve odluke vezane za Internet su donošene na tehničkoj, a ne na političkoj bazi.
- Internet nije centralizovan već distribuirano organizovan
- Internet je omogućio ljudima da po vrlo niskoj ceni koriste veliki broj informacionokomunikacionih servisa
- Softver za korišćenje Interneta (veb čitači) je besplatan ili vrlo jeftin.

▼ Poglavlje 4

Interakcija čovek-računar

ČIME SE BAVI DISCIPLINA INTERAKCIJA ČOVEK-RAČUNAR

Interakcija čovek - računar je disciplina koja se bavi projektovanjem, evaluacijom i implementacijom interaktivnih računarskih sistema koje koristi čovek

Da bi čovek mogao da koristi računarske sisteme potrebno je ostvariti interakciju između čoveka i sistema. Korisnik ima potrebe da prenese sistemu svoje želje (komande) i podatke koje računar treba da obradi. S druge strane računar ima potrebe da korisniku prezentuje rezultate obrade komande ili podataka. Interakcija između čoveka i računarskog sistema se obavlja preko interfejsa. Mogući prevod termina interfejs (engleski **interface**) na srpski jezik je međuveza ili veznik, ali se u svakodnevnom govoru već odomaćio pojam interfejs. U opštem slučaju, interfejs je termin kojim se označava mesto na kome dva nezavisna sistema dolaze u međudejstvo ili komuniciraju. U računarstvu se pod **interfejsom** podrazumeva prezentacija, komunikacija i interakcija između korisnika i računarskog sistema.

Interakcija čovek - računar (engleski HCI - **Human-computer interaction**) je disciplina koja se bavi projektovanjem, evaluacijom i implementacijom interaktivnih računarskih sistema koje koristi čovek, kao i proučavanjem čovekove interakcije sa računarom.

Interakcija čovek-računar se obavlja preko softverskih i hardverskih komponenata. Tipična softverska komponenta, ali ne i jedina, koja učestvuje u interakciji je grafički korisnički interfejs. Tipične hardverske komponente za interakciju su monitori, zvučnici, štampači, tastature i miševi.

Poznato je da je jedan od najvećih problema u korišćenju računara prve i druge generacije bio vrlo loš interfejs između korisnika i računarskog sistema. Tek su računari III generacije standardno imali monitore i tastature. Sa razvojem PC računara dolazi do standardizacije ulazno-izlaznih kanala, što je bila dobra osnova za razvoj novih ulazno-izlaznih uređaja i njihovu proizvodnju u velikim serijama. Sve ovo je dovelo do razvoja različitih tipova korisničkog interfejsa. Snažan uticaj na razvoj metoda interakcije čovek-računar su imale i druge discipline kao što su računarska grafika, operativni sistemi, ergonomija, kognitivna psihologija i računarske nauke.

▼ 4.1 Korisnički interfejs

TIPOVI KORISNIČKOG INTERFEJSA

Direktni korisnički interfejs

Korisnički interfejs (engl. user interface (UI)) je širok pojam za bilo koji sistem, bilo da je on fizički ili zasnovan na softveru, a koji omogućava korisniku da se poveže sa datom tehnologijom. Mnoge različite vrste korisničkih interfejsa su dizajnirane za različite uređaje i softverske programe. Mnogi od njih imaju neke osnovne sličnosti, iako svaki je jedinstven na svoj način.

Postoji nekoliko različitih tipova direktnog korisničkog interfejsa:

- Interfejs baziran na komandnoj liniji
- Interfejs baziran na sistemu menija
- Objektno-orientisani interfejs
- Ekspert sistem interfejs
- Web interfejs

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

INTERFEJS BAZIRAN NA KOMANDNOJ LINIJI

U ovom slučaju se komunikacija između čoveka i računara odvija preko komandne linije

Interfejs baziran na komandnoj liniji (engl. Command Line Interface (CLI)) je tekstualni interfejs koji se koristi za upravljanje softverom i operativnim sistemom, a omogućava korisniku da odgovori na vizuelne upite kucanjem pojedinačnih komandi u interfejsu i dobijanju odgovora na isti način. U ovom slučaju se komunikacija između čoveka i računara odvija preko komandne linije. Njegov radni mehanizam je veoma lak. Korisnik unosi određenu komandu, pritisne "Enter", a onda čeka na odgovor. Nakon prijema naredbe, CLI je obrađuje i prikazuje izlaz / rezultat na istom ekranu. Korisnik, koristeći tastaturu, unosi komande operativnom sistemu ili nekom aplikativnom softverskom programu. Komande su uobičajeno mnemonici praćeni potrebnim parametrima. Na primer komanda za kopiranje datoteke sa nekog direktorijuma na disku na disketu ima oblik:

copy c:tekstovi/pisma/ponuda.doc a:/ponuda.doc

Razume se da je korišćenje ovakvog interfejsa zahtevalo dobru obučenost korisnika i stalnu kontrolu unetih komandi. Da bi se korisniku olakšalo unošenje čestih komandi korišćeni su funkcionalni tasteri kojima su dodeljivana određena značenja ili kombinacije Ctrl i Alt tastera sa nekim drugim tasterom. Iako danas postoje mnogo intuitivniji korisnički interfejsi neki korisnici preferiraju komandnu liniju iz razloga produktivnosti ili zbog navike.

CLI je sasvim drugačiji od grafičkog korisničkog interfejsa (engl. **Graphic User Interface (GUI)**) koji se trenutno koristi u najnovijim operativnim sistemima.

Kako bi najbolje iskoristiti CLI, korisnik mora biti u mogućnosti da unosi pregršt komandi brzo i to jedan po jedan. Postoji mnogo aplikacija (**mono-processing systems**) koji još uvek koriste CLI. Osim toga, neki programski jezici kao što su Forth , Python i BASIC, nude CLI.

MS DOS je najbolji primer CLI.

INTERFEJSI BAZIRANI NA SISTEMU MENIJA

Meniji se sastoje od liste opcija koja je u nekom okruženju na raspolaganju korisnika

Meniji se sastoje od liste opcija koja je u nekom okruženju na raspolaganju korisnika. Tipično korisnik bira željenu opciju ukucavanjem slova ili broja opcije, pomeranjem pokazivača pomoću strelica na tastaturi i pritiskom na taster Enter ili lociranjem pokazivača i klikom na taster miša. Često izbor jedne opcije u meniju vodi do novog podmenija koji daje detaljne mogućnosti za izabranu opciju. Sistem ovakvih menija ima strukturu hiperarhijskog stabla.

1. New game
2. Save game
3. Change options
4. Help
5. Exit game

Press appropriate number to select desidered option

U odnosu na interfejs baziran na komandnoj liniji, ovakvi interfejsi su mnogo lakši za korišćenje, intuitivni su i lako se uče. Međutim, veliki sistem menija po dubini i širini može da postane kontraproduktivan jer zahteva više vremena za postizanje željenog efekta nego meniji bazirani na komandnoj liniji.

PREDNOSTI I MANE KORIŠĆENJA INTERFEJSA BAZIRANOG NA SISTEMU MENIJA

Prednost je što je izuzetno lak za korišćenje, ali loše dizajniran interfejs baziran na sistemu menija može biti spor

Prednosti

- Izuzetno je lak za korišćenje. Neko ko nikad nije video interfejs se može snaći u njegovom korišćenju
- Nema komandi koje moraju da se uče ili zapamte

- Korak-po-korak instrukcije su date tako da korisnik ne mora da pamti
- Čak i ako korisni ne zna tačno šta da radi, obično može da pogodi svoj put kroz navigaciju u različitim pokušajima
- Meniji ne moraju da budu napravljeni sa vizuelnom reprezentacijom. Oni na primer mogu biti i govornog tipa, tako da su recimo korisni u telefonskom odabiru opcija
- Njima ne trebaju velike količine procesorske snage i memorije
- Prilično je lako za programera da kreira iste menije u različitim programskim jezicima

Mane

- Loše dizajniran interfejs baziran na sistemu menija može biti spor za korišćenje
- Previše ekrana sa kojima mora da se radi može biti iritantno i zamarajuće za korisnika, naročito ako prouzrokuje da se korisnik nervira ili mu postaje dosadno, zato što niz koraka i opcija koje treba da izabere traje dugo
- Na početku rada korisnik često ne može tačno da stigne do podmenija do kojeg želi. U tom slučaju korisnik mora da prođe kroz niz ekrana da bi stigao do opcije do koje želi.
- Meni može da zauzme veliki deo ekrana, tako da korisnik mora stalno da se šetam između aplikacija
- Ako meni je loše osmišljen tada može biti teško čitljiv. Tako ne primer veličina slova može biti suviše mala za lica sa oštećenim vidom. Takođe, boje mogu biti u koliziji što čini tekst teškim za čitanje. Isti problem se može desiti i sa loše odabrani stilom slova (font-om).

OBJEKTNO-ORIJENTISANI, EKSPERT SISTEM I VEB INTERFEJSI

Objektno-orientisani interfejsi se još nazivaju grafički korisnički interfejsi (GUI – graphical user interface) ili interfejsi bazirani na ikonama

Objektno-orientisani interfejsi

Objektno-orientisani interfejsi se još nazivaju grafički korisnički interfejsi (GUI – **graphical user interface**) ili interfejsi bazirani na ikonama. Postali su vrlo popularni od svog uvođenja na Apple Macintosh i kasnije Windows operativnim sistemima. Osnovni elementi ovakvog interfejsa su prozori, ikone, meniji i pokazivači. Zbog toga se ovi interfejsi označavaju skraćenicom **WIMP** (**Window, Icon, Menu, Pointing device**). Po pravilu, korisnik bira pokazivačem jednu od ikona, a klikom pokreće asociranu akciju. OO interfejsi su intuitivni i zahtevaju minimum obuke. WIMP zahteva znatne memoriske i procesorske resurse od računara na kome je implementiran.

Ekspert sistem interfejsi

Ekspert sistem interfejsi koriste tehnologiju obrade prirodnog jezika (NLP - **natural language processing**) da povežu korisnika i programski sistem. Sistem je zasnovan na glasovnom unosu podataka i komandi, prepoznavanju glasa i razčlanjivanju i prepoznavanju reči i rečenica. Za ulaz se, pored mikrofona, koriste i tastatura i miš kao pomoćni ulazni uređaji i zvučnici kao izlazni uređaj. NLP zahteva vrlo snažne računare sa brzim procesorima i velikim memorijskim kapacitetima.

Pored ekspert sistem interfejsa baziranih na glasu postoje i interfejsi bazirani na prepoznavanju napisanih karaktera. Ovakvi interfejsi zahtevaju specijalne ulazne uređaje i ekspertske sisteme za prepoznavanje karaktera.

Veb interfejs

Veb interfejs je zasnovan na metafori uspostavljenoj World Wide Web-om. Datoteke i programi su označeni kao hiperlinkovane strane. U okviru veb interfejsa posebnu kategoriju čine veb forme. Veb forme su projektovane tako da korisnik može da unosi podatke u raspoloživa polja ili da bira neki od ponuđenih odgovora. Kod dobro dizajniranih formi uneti podaci se verifikuju, a samo korišćenje interfejsa je intuitivno.

▼ Poglavlje 5

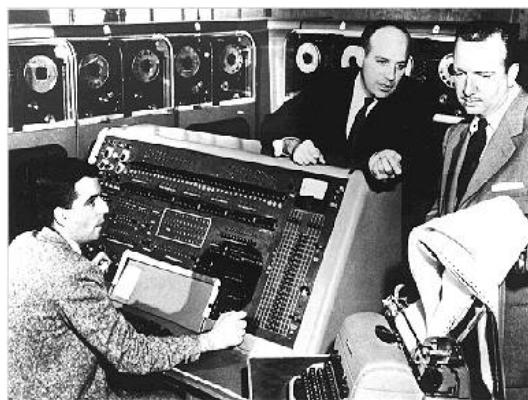
Istorija razvoja interfejsa

KOMUNIKACIJA SA PRVIM RAČUNARIMA

Komunikacija čoveka sa računarima prve generacije se obavljala korišćenjem tastera, papirne bušene trake, bušenih kartica, sijalica i štampača

Tokom cele istorije razvoja računarstva inženjeri su se trudili da razviju nove metode koje će olakšati korisnicima rad sa računarskim sistemima. Njihova rešenja su bila ograničena tehnološkim mogućnostima koja su postojala u pojedinim fazama razvoja računarstva. Tačnije, razvoj interakcije čovek-računar povezan je sa razvojem ulaznih i izlaznih uređaja.

Komunikacija čoveka sa računarima prve generacije se obavljala korišćenjem tastera, papirne bušene trake, bušenih kartica, sijalica i štampača. U to doba nisu korišćene tastature i grafički ulazni uređaji. Nije postojala gotovo nikakva interaktivnost između čoveka i računara u realnom vremenu.



Slika 5.1 UNIVAC I

Prvi tekst bazirani terminal koji se masovno koristio je bio IBM-ov terminal 3270 iz 1970. godine, baziran na katodnoj cevi.



Slika 5.2 Primer karakter orientisanog korisničkog interfejsa



Slika 5.3 IBM PC 1981

PRVI GRAFIČKI KORISNIČKI INTERFEJS

Razvoj prvih grafičkih korisničkih interfejsa započeo je 70-tih godina prošlog veka u Xerox-ovom istraživačkom centru Palo Alto

Godine 1963. Ivan Sutherland sa MIT-a je u okviru svoje doktorske disertacije razvio program Sketchpad koji je omogućavao da se pomoću svetlosnog pera na ekranu crtaju tačke, linije i krugovi. Program je mogao da dodeljuje karakteristike grafičkim objektima i da gradi relacije među njima. Pored toga bile su moguće operacije pomeranja, kopiranja, zumiranja, rotiranja i zapisivanja kreiranih objekata. Ovim je udaren kamen temeljac novoj računarskoj disciplini koja je nazvana kompjuterska grafika, ali je istovremeno ljudima dalo prve ideje o grafičkom korisničkom interfejsu.

Razvoj prvih grafičkih korisničkih interfejsa započeo je 70-tih godina prošlog veka u Xerox-ovom istraživačkom centru Palo Alto (Palo Alto Research Center) gde su prvi put razvijene i primenjene tehnologije kao što su bitmapirani displeji, desktop, prozor, miš i point-and-click editor. Douglas Engelbart je u Xerox-u 1970. patentirao prvog miša sa točkićima. Njegov kolega, Ron Rajder je 1974. godine dao ideju da se miš obrne i da se umesto točkića postavi kuglica. Ovakvi miševi se i danas koriste samo što se na osnovu Apple-ovog izuma umesto metalne kuglice koristi gumena.

U Xerox-u su razvijeni sistemi Altus i STAR koji su koristili miša za pokazivanje i izbor u okviru **grafičkog korisničkog interfejsa**. Xerox je 1981. godine izbacio na tržište radnu stanicu STAR na kojoj je po prvi put primenjena tehnologija desktop-a, ali ona nije doživela veći uspeh. Ova radna stanica je koristila miša i grafički monitor. Korisnički interfejs je omogućavao da

se pomoću miša pokažu i izaberu grafički elementi prikazani na monitoru. Interfejs je imao mogućnosti prikaza višestrukih preklapajućih prozora i pop-up menije, a kasnije su dodate ikone kao što su kanta za otpatke, ormar sa fajlovima i folder dokumenta.



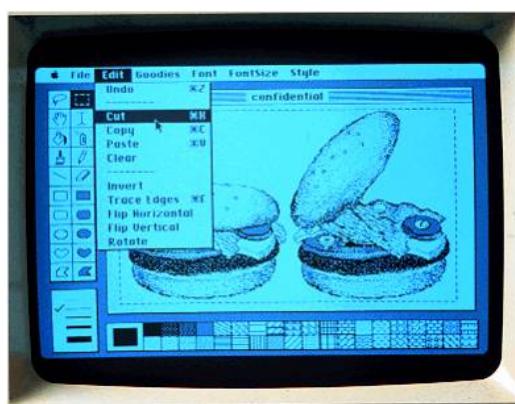
Slika 5.4 Primer grafičkog korisničkog interfejsa

APPLE I KORISNIČKI INTERFEJS

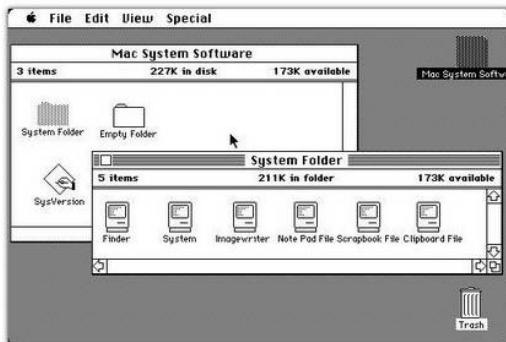
Pregled MAC korisničkih interfejsa

Stiv Džobs, osnivač kompanije Apple se oduševio idejom grafičkog korisničkog interfejsa i pokušao da ga implementira 1983. na računaru Apple Lisa. Na žalost ni ovaj računar nije imao komercijalni uspeh, ali je zato njegov naslednik Apple Macintosh iz 1984. uspeo pre svega zbog marketinške kampanje koja je u prvi plan izbacila revolucionaran grafički korisnički interfejs i što je ceo računar sagrađen kao integrисани sistem. Na ovom računaru su prvi put primjenjeni padajući meniji.

U to doba javlja se krilatica “**what you see is what you get**” (WYSIWYG) kojom se ukazuje na mogućnost da korisnik na monitoru može da vidi isto ono što će dobiti i na štampaču.



Slika 5.5 Prikaz MacPaint-a na prvom Macintosh GUI-u



Slika 5.6 Mac OS 1.0



Slika 5.7 Mac OS 8



Slika 5.8 Mac OS 9

VIDEO - KAKO JE STIV DŽOBS DOBIO IDEJU ZA GUI OD XEROX-A

How Steve Jobs got the ideas of GUI from XEROX

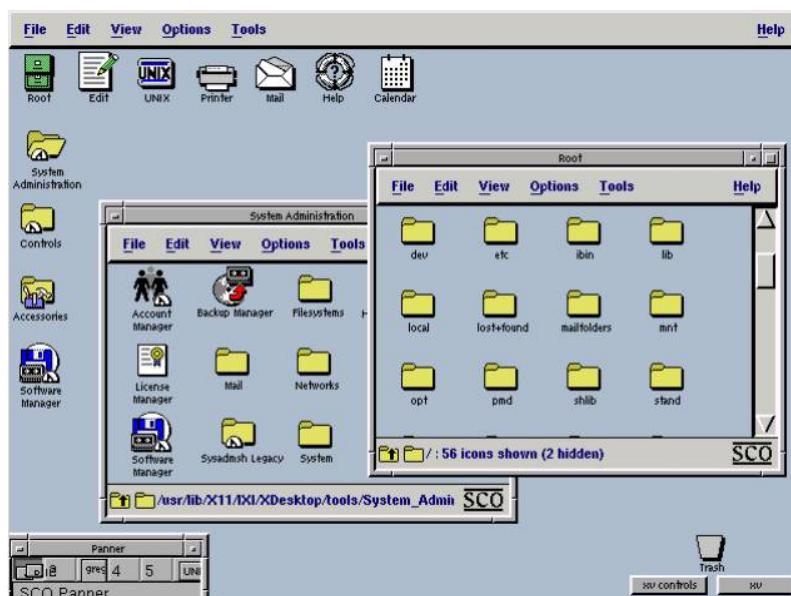
Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

POČETAK MS-A

Windows 1.0 je bio rudimentalni grafički korisnički interfejs koji je radio nad MS DOS operativnim sistemom.

Microsoft pokušao da kopira Apple-ov korisnički interfejs pa je 1985. godine objavio prvu verziju Windows-a. Windows 1.0 je bio rudimentalni grafički korisnički interfejs koji je radio nad MS DOS operativnim sistemom. Windows nije imao značajniji uspeh sve do verzije Windows-a 3.0 i 3.1 iz 1992. godine, ali ni tada njegova funkcionalnost nije bila na nivou Apple-a. Tek je Windows 95 dobio ovu funkcionalnost.

Za UNIX operativni sistem, koji je inicijalno radio sa interfejsom baziranim na komandnoj liniji, razvijeno je nekoliko grafičkih korisničkih interfejsa kao što je X Window System iz 1987. godine i Open Look by koga su razvili AT&T i Sun Microsystems 1989. godine, ali je Motif postao defakto standard.



Slika 5.9 Motif GUI

TENDENCIJE INTERFEJSA

Korišćenje grafičkog korisničkog interfejsa predstavljal je prekretnicu u filozofiji interakcije čovek-računar

Korišćenje grafičkog korisničkog interfejsa predstavljal je prekretnicu u filozofiji interakcije čovek-računar. Do tada se čovek prilagođavao sistemu. Nakon uvođenja grafičkog interfejsa počinje era u kojoj je korisnik u centru pažnje. Postavljaju se novi principi projektovanja korisničkog interfejsa među kojima su glavni:

- Korisnik ima mogućnost da upravlja interfejsom i da bira način na koji će koristiti računar
- Smanjiti potrebu da korisnik pamti procedure komandovanja

- Korisnički interfejs treba da bude konzistentan, što znači da se preporučuje da se isti elementi korisničkog interfejsa koriste i u operativnom sistemu i u programima.

Na dalji razvoj interakcije između korisnika i računara utiče pojava mnogih novih tehnologija. Ovde će se nabrojati samo neki glavni uticaji.

- Pojeftinjenje komponenata vodi ka bržim sistemima sa više memorije što projektantima interfejsa daje više prostora za nova rešenja koja su gladna za procesorskom snagom i memorijom.
- Minijaturizacija hardvera i sve manja potrošnja struje komponenata računarskog sistema omogućuje portabilnost i korišćenje računara na bilo kom mestu.
- Masovna proizvodnja TFT displeja dovela je do sniženja njihove cene i mogućnost da se njihovom primenom promeni i oblik računara. Dobar primer za ovo je tablet PC.
- Povjekili su se mnogi novi ulazni uređaji koji reaguju na glas, gestove, dodir, ili koriste pisaljku.
- Povjekili su se mnogi neračunarski uređaji koji imaju ugrađene računare (**embedded system**) i mogućnost povezivanja na mrežu.
- Povećanje broja korisnika širom sveta.
- Potreba za poslovnom kolaboracijom dovela je do pojave novih grupnih interfejsa koji predstavljaju dobar alat za vođenje sastanaka udaljenih korisnika, projektovanje, učenje i slično.
- Korisnik sam prilagođava interfejs i program svojim potrebama
- Efektivno korišćenje medija (govor, muzika, zvuk, film) znatno obogaćuju komunikaciju. Tako se na primer korišćenjem zvuka mogu sugerisati fizičke osobine objekta i poboljšati verodostojnost slike i informacija.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

✓ Poglavlje 6

Pokazna vežba: JS & HTML

KREIRATI DIGITRON - SA REŠENJEM

Prikaz kreiranja prostog digitrona - zadatkov sa rešenjem

Ukupno predviđeno vreme za pokazne vežbe je 75 minuta.

Predviđeno vreme izrade sledećeg primera je 15 minuta.

Kreirati digitron koji je prikazan na sledećoj slici koristeći JS i HTML:



Slika 6.1 Digitron koji je potrebno kreirati

Rešenje:

```
<html>
<head>
    <title>Digitron</title>

<script language=javascript type="text/javascript">var plus,minus,divide,times
function initialise(){plus=document.calc.operator.options[0]
minus=document.calc.operator.options[1]
divide=document.calc.operator.options[2]
times=document.calc.operator.options[3]}function
calculate(){x=parseInt(document.calc.val1.value)
y=parseInt(document.calc.val2.value)
if(plus.selected)document.calc.answer.value=x+y
if(minus.selected)document.calc.answer.value=x-y
if(divide.selected)document.calc.answer.value=x/y
</script>
```

```
if(times.selected)document.calc.answer.value=x*y}</script>

</head>

<body onLoad="initialise()">
<h2>Digitron</h2>

Primer prostog kalkulatora
<br/>

<form name="calc" action="post">
<input type=text name=val1 size=10>

<select name=operator>
<option value=plus>+
<option value=minus>-
<option value=divide>/
<option value=times>*
</select>

<input type=text name=val2 size=10>
=
<input type=text name=Odgovor size=10>
<input type=button value=answer onClick="calculate()">
</form>

</body>
</html>
```

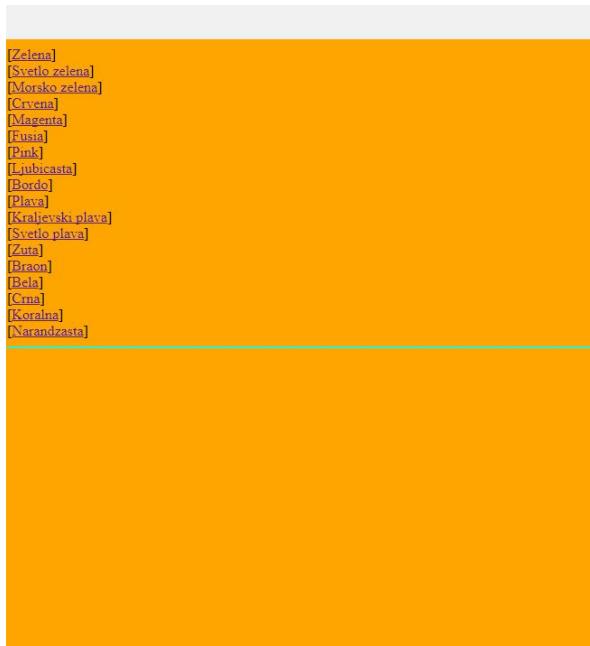
PROMENA BOJE POZADINE - PRIMER

Nakon prelaska mišem preko imena boje, dolazi do promene boje pozadine

Predviđeno vreme izrade sledećeg primera je 10 minuta.

Zadatak:

Potrebno je prilikom prelaska mišem preko imena boje, dolazi do promene boje pozadine. Rešiti koristeći JS.



Slika 6.2 Zadatak

Rešenje:

```
[<a href="/" onmouseover="document.bgColor='green'">Zelena</a>]<br/>
[<a href="/" onmouseover="document.bgColor='green'">Svetlo zelena</a>]<br/>
[<a href="/" onmouseover="document.bgColor='seagreen'">Morsko zelena</a>]<br/>
[<a href="/" onmouseover="document.bgColor='red'">Crvena</a>]<BR>
[<a href="/" onmouseover="document.bgColor='magenta'">Magenta</a>]<br/>
[<a href="/" onmouseover="document.bgColor='fusia'">Fusia</a>]<br/>
[<a href="/" onmouseover="document.bgColor='pink'">Pink</a>]<br/>
[<a href="/" onmouseover="document.bgColor='purple'">Ljubicasta</a>]<BR>
[<a href="/" onmouseover="document.bgColor='navy'">Bordo</a>]<br/>
[<a href="/" onmouseover="document.bgColor='blue'">Plava</a>]<br/>
[<a href="/" onmouseover="document.bgColor='royalblue'">Kraljevski plava</a>]<br/>
[<a href="/" onmouseover="document.bgColor='skyblue'">Svetlo plava</a>]<BR>
[<a href="/" onmouseover="document.bgColor='yellow'">Zuta</a>]<br/>
[<a href="/" onmouseover="document.bgColor='brown'">Braon</a>]<br/>
[<a href="/" onmouseover="document.bgColor='black'">Crna</a>]<br/>
```

```
onmouseover="document.bgColor='white'">Bela</a>]<BR>
[<a href="/" onmouseover="document.bgColor='black'">Crna</a>]<br/>
[<a href="/" onmouseover="document.bgColor='coral'">Koralna</a>]<br/>
[<a href="/" onmouseover="document.bgColor='orange'">Narandzasta</a>]<br/>
<hr color="#00FFFF">
```

GOOGLE SEARCH NA HTML STRANI

Potrebno ubaciti google search na HTML strani

Predviđeno vreme izrade sledećeg primera je 10 minuta.

Zadatak:

Potrebno je dodati google search na HTML strani kao što je prikazao na slici koristeći JS i HTML. Ukoliko se upiše tekst, dolazi do pretrage u novom tabu u google-u.



Slika 6.3 Zadatak

Rešenje:

```
<html>
<center>
<form method=GET action="http://www.google.com/search">
<table bgcolor="#FFFFFF">
<tr>
<td>
<a href="http://www.google.com/">
</a>
<input type=text name=q size=31 maxlength=255 value="">
<input type=hidden name=hl value="en">
<input type=submit name=btnG value="Google Search">
</td>
</tr>
</table>
</form>
</center>
</html>
```

DODAVANJE ELEMENATA U LISTU

Napisati skriptu koja dodaje element u listu

Predviđeno vreme izrade sledećeg primera je 15 minuta.

Zadatak:

Napraviti polje za unos u koje će korisnik unositi elemente liste. Zatim napisati skriptu koja na osnovu unosa korisnika unosi elemente u listu.

Rešenje:

```
function addItem(){
    var li = document.createElement("li");
    var input = document.getElementById("txtInput");
    li.innerHTML = input.value;
    input.value = "";

    document.getElementById("lista").appendChild(li);
}
```

Element "**li**" koji predstavlja listu se kreira koristeći funkciju *createElement*.

U promenljivu **input** se čuva unos korisnika iz polja za unos.

Zatim se u promenljivu **li** čuva unos korisnika pomoću *.value* metode koja uzima vrednost polja za unos.

Poslednji korak je da se novi element doda u listu što se radi pomoću metode *appendChild*.

NASUMIČAN IZBOR REČENICA

Napisati skriptu koja nasumično vadi rečenice iz liste

Predviđeno vreme izrade sledećeg primera je 15 minuta.

Zadatak:

Napisati funkciju u JavaScript-u koja na osnovu niza 6 različitih citata i autora, ispisuje na stranici po jedan citat, ali tako da se citat promeni kada se stranica ponovo osveži (koristiti ugrađenu *Math.random()* funkciju).

```
<html>

<head>
    <title>Slucajni citati</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <!--Ukljucivanje meta taga sa UTF-8 dobijate i latinicna srpska slova-->
</head>
```

```
<body>
    <h1>Slučajni citati</h1>
    <hr />
    <script language="JavaScript">
        //citati i autori su smesteni u 2 posebna niza
        quotes = new Array(6);
        authors = new Array(6);
        quotes[0] =
            "Toliko je bilo stvari u žžživotu kojih smo se bojali. A nije
        trebalo.Trebalo je živeti.";
        authors[0] = "Ivo Andrić";
        quotes[1] =
            "Prijateljstvo se na bira, ono biva, ko zna zbog čega, kao ljubav ";
        authors[1] = "Meša Selimović";
        quotes[2] =
            "Zdrav covek ima hiljadu želja, a bolestan samo jednu - da ozdravi.";
        authors[2] = "Narodna izreka";
        quotes[3] =
            "Nemojte da hendikepirate svoju decu time što ćete im život učiniti
        suviše lakim ";
        authors[3] = "Duško Radović";
        quotes[4] = "Čast se ne može oduzeti, ona se može samo izgubiti";
        authors[4] = "Čehov";
        quotes[5] = "Nema sunca bez svetlosti, ni čoveka bez ljubavi";
        authors[5] = "Gete";
        //izracunavanje slucajnog broja, izmedju 0 i 1
        index = Math.floor(Math.random() * quotes.length);
        //prikaz citata u vidu definicione liste
        document.write("<dl>\n");
        document.write("<dt>" + '"' + quotes[index] + '"\n');
        document.write("<dd>" + "- " + authors[index] + "\n");
        document.write("</dl>\n");
    </script>
    <hr /> Slučajan citat se prikazuje na ekranu. Probajte osvezavanje stranice
    (Refresh/Reload F5) za ponovno ucitavanje stranice i prikaz drugog citata.
    <hr />
</body>

</html>
```

PREGLED TRENUOTNOG VREMENA

Prikazati trenutno vreme

Predviđeno vreme za izradu sledećeg zadatka je 10 minuta.

Zadatak:

Napisati program koji ispisuje trenutno vreme u formatu SATI:MINUTI:SEKUNDE PM/AM.

```
<html>
<head>
    <title>Trenutno vreme</title>
    <script>
        function Vreme() {
            time = new Date();
            cas = time.getHours();
            minuti = time.getMinutes();
            sekunde = time.getSeconds();
            temp = "" + (cas > 12 ? cas - 12 : cas);
            temp += (minuti < 10 ? ":0" : ":") + minuti;
            temp += (sekunde < 10 ? ":0" : ":") + sekunde;
            temp += cas >= 12 ? " P.M." : " A.M.";
            document.vremeForma.cifre.value = temp;
            setTimeout("Vreme()", 1000);
            //posle svakih 1000milisekundi, odnosno 1 sekunde
            //ponovo se ucitava funkcija Vreme()
        }
    </script>
</head>

<body bgcolor="#FFFFFF" onLoad="Vreme()">
    <form name="vremeForma">
        Trenutno vreme je &nbsp <input type="text" name="cifre" size="12" />
    </form>
</body>
</html>
```

▼ Poglavlje 7

Zadaci za samostalni rad: JS & HTML

ZADATAK ZA VEŽBU 1

Koristeći HTML napraviti web sajt prezentaciju namjenjenu ljubiteljima programiranja.

Predviđeno vreme za izradu sledećeg zadatka je 20 minuta.

Koristeći HTML napraviti web sajt prezentaciju namjenjenu ljubiteljima programiranja.

1. Prva strana sajta treba da sadrži linkove na ostale strane
2. Postaviti željenu sliku kao pozadinu na svim web stranama ili osmisliti adekvatno zaglavje. Pozadina i zaglavje treba da budu konzistentni na svim stranicama.
3. Kreirati stranu "Sabiranje" na kojoj se nalazi jedan element za unos teksta i dva dugmeta, "dodaj" i "saberi". Na stranici implementirati skriptu koja računa sumu vrednosti u zadatom nizu.

Korisnik unosi broj članova niza. Klikom na dugme "dodaj", u stranicu se dodaju zadati broj novih elemenata za unos, po jedan za svaki član niza.

Korisnik unosi vrednosti u elemente za unos. Klikom na dugme "saberi", program ispisuje sumu svih unesenih vrednosti.

Ako se desi da je neka vrednost negativan broj, ispisati poruku sa greškom.

4. Kreirati stranu "Jezici" na kojoj će biti date informacije o bar dva programska jezika. Za svaki jezik na strani ispisati podnaslov, paragraf teksta i odgovarajuću sliku

ZADATAK ZA VEŽBU 2

Koristeći HTML napraviti web sajt prezentaciju namjenjenu ljubiteljima programiranja - drugi deo zadataka.

Predviđeno vreme za izradu sledećeg zadatka je 20 minuta.

Koristeći HTML napraviti web sajt prezentaciju namjenjenu ljubiteljima programiranja.

1. Prva strana sajta treba da sadrži linkove na ostale strane.

2. Postaviti željenu sliku kao pozadinu na svim web stranama ili osmisli adekvatno zaglavje. Pozadina i zaglavje treba da budu konzistentni na svim stranicama

3. Kreirati stranu "Bodovi" na kojoj će se nalaziti tri elementa za unos teksta (ime, prezime, broj bodova) i dugme.

Klikom na dugme, podaci iz elementa za unos se upisuju u tabelu na stranici. Tabela treba da ima tri kolone koje odgovaraju elementima za unos, i četvrtu kolonu "Uslov". Svaki klik dugmeta dodaje novi red u tabelu.

Ako student ima više od 35 bodova, u koloni uslov ispisati "Da", u protivnom ispisati "Ne".

U slučaju da je uneseno ime ili prezime kraće od tri karaktera, ispisati poruku sa greškom.

4. Kreirati stranu "Jezici" na kojoj će biti date informacije o bar dva programska jezika. Za svaki jezik na strani ispisati podnaslov, paragraf teksta i odgovarajuću sliku.

ZADATAK ZA VEŽBU 3

Postaviti željenu sliku kao pozadinu na svim web stranama ili osmisli adekvatno zaglavje. Pozadina i zaglavje treba da budu konzistentni na svim stranicama.

Predviđeno vreme za izradu sledećeg zadatka je 20 minuta.

1. Prva strana sajta treba da sadrži linkove na ostale strane.

2. Postaviti željenu sliku kao pozadinu na svim web stranama ili osmisli adekvatno zaglavje. Pozadina i zaglavje treba da budu konzistentni na svim stranicama.

3. Kreirati stranu "Max" na kojoj će se nalaziti tri dugmeta, "+" , "-" i "max". Klik na dugme + u stranu se dodaje jedan element za unos teksta. Klikom na dugme - poslednji dodati element se briše iz strane. Klikom na dugme max, program treba da nadje najveću brojčanu vrednost u elementima za unos, i ispiše je.

4. Kreirati stranu „Queue“ na kojoj će se nalaziti polje za unos teksta i dva dugmeta. Klikom na prvo dugme, tekst iz polja se dodaje u listu. Klikom na drugo dugme, prva dodata stavka iz liste (stavka sa početka liste) se ispisuje u element na strani ili pop up, i briše iz liste. Ukoliko je lista prazna, ispisati poruku sa greškom.

Da biste implementirali ovu funkcionalnost možete koristiti niz i metode push() i shift().

5. Kreirati stranu "Jezici" na kojoj će biti date informacije o bar dva programska jezika. Za svaki jezik na strani u tabeli ispisati ime, godinu kada se jezik pojavio, ime i prezime tvorca jezika.

✓ Poglavlje 8

Domaći zadatak

OPIS DOMAĆEG ZADATKA - DZ12

Kreirati stranu "Množenje" na kojoj se nalaze i dva dugmeta, "dodaj" i "pomnoži". Na stranici implementirati skriptu koja računa proizvod vrednosti u zadatom nizu.

Očekivano vreme izrade zadatka: 45 minuta

Opis domaćeg zadatka:

- Koristeći HTML napraviti web sajt prezentaciju predmeta IT101.
- Prva strana sajta treba da sadrži linkove na ostale strane.
- Postaviti željenu sliku kao pozadinu na svim web stranama ili osmisliti adekvatno zaglavlj. Pozadina i zaglavje treba da budu konzistentni na svim stranicama.
- Kreirati stranu "Poeni". Na ovoj strani napraviti formu za unos predispitnih poena studenta na predmetu IT101. Nakon unosa podataka, korisnik klikne na dugme "Unesi". Klikom na ovo dugme, podaci iz forme se ubacuju u tabelu sa poenima koja se nalazi na istoj stranici. Forma treba da sadrži id, indeks studenta, ime studenta, prezime studenta, šifra predmeta i broj poena.
- Kreirati stranu "IT101" na kojoj će u dva paragrafa biti isписан opis predmeta IT101.
- Strana "autor" mora da sadrži sliku i podatke o studentu (ime, prezime, kontakt email)

Zadatak dostaviti kao IT101-DZ12_Ime_Prezime_BrojIndeksa

✓ Zaključak

ZAKLJUČAK

U ovom predavanju dat je pregled istorijskog razvoja informacionih tehnologija sa fokusom na razvoj internet i razvoj korisničkog interfejsa.

Još jedan interesantan video koji se preporučuje da se pogleda je:

- **Revolution OS - movie about GNU/Linux history**

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Literatura

- [1] Ceruzzi, Paul E., A history of modern computing, 2nd ed., Massachusetts Institute of Technology, The MIT Press, 2003
- [2] Raúl Rojas and Ulf Hashagen, editors, The First Computers—History and Architectures, MIT press, 2000
- [3] Deborah G. Tatar, A programmers Guide to Common Lisp, Digital Press, 1987
- [4] An Illustrated History of Computers, <http://www.computersciencelab.com/ComputerHistory/History.htm>
- [5] The History of Computers, <https://www.thoughtco.com/history-of-computers-4082769>
- [6] Timeline of Computer History, <http://www.computerhistory.org/timeline/>



IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

RAZVOJ INFORMACIONIH SISTEMA

Lekcija 13

PRIRUČNIK ZA STUDENTE

IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

Lekcija 13

RAZVOJ INFORMACIONIH SISTEMA

- ▼ RAZVOJ INFORMACIONIH SISTEMA
- ▼ Poglavlje 1: Uvođenje nove aplikacije
- ▼ Poglavlje 2: Informacioni sistem
- ▼ Poglavlje 3: Razvoj informacionog sistema
- ▼ Poglavlje 4: SDLC - Studija izvodljivosti
- ▼ Poglavlje 5: RAD metoda
- ▼ Poglavlje 6: Agilne metode razvoja
- ▼ Poglavlje 7: Upravljanje promenama
- ▼ Poglavlje 8: Pokazna vežba: GanttProject
- ▼ Poglavlje 9: Zadatak za samostalni rad: Gantov dijagram
- ▼ Poglavlje 10: DOMAĆI ZADATAK
- ▼ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

✓ Uvod

UVOD

Cilj ovog predavanja je da objasni osnovni proces koji se primenjuje prilikom razvoja informacionog sistema

U ovom predavanju biće reči o razvoju informacionih sistema, počevši od toga kako poslovni procesi utiču na razvoj informacionog sistema, pa do samog isporučivanja gotovog informacionog sistema. Na ovom predavanju biće obrađene sledeće teme:

- Šta je to informacioni sistem i kako on utiče na poslovanje
- Vrste informacionog sistema
- Faze u razvoju informacionog sistema

Cilj ovog predavanja je da objasni osnovni proces koji se primenjuje prilikom razvoja informacionog sistema.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 1

Uvođenje nove aplikacije

RAZLOG ZA IT PROMENAMA U ORGANIZACIJI

Razlog može da bude promena hardvera, infrastrukture, neke aplikacije ili celog informacionog sistema

Uvođenje nove aplikacije u rad neke organizacije nikad se ne može posmatrati izolovano. Jedna aplikacija je uvek deo informacionog sistema organizacije koji je opet deo poslovnog sistema, pa postoje mnogi međusobni uticaji koje treba prethodno razmotriti da bi se shvatio proces uvođenja aplikacije. Zbog toga će se ovde najpre dati jedan kratak pogled na informacioni sistem u celini, ukratko će se prikazati proces razvoja informacionog sistema, a onda će se govoriti o uvođenju nove aplikacije i pratećim pitanjima.

Pojava promena u IT okruženju je vrlo česta. Razlog može da bude promena hardvera, infrastrukture, neke aplikacije ili celog informacionog sistema. Promene takođe mogu da budu i organizacione prirode. Uobičajeno je da je uvođenje nove aplikacije ili informacionog sistema deo projekta koji su načinili IT profesionalci. Rezultat gotovo svih projekata su promene. Korisnici rezultata projekta ili oni koji će upravljati novim sistemom se plaše promena. Ljudi ne prihvataju novi sistem samo zato što je on bolji. Kod njih postoji otpor promenama zbog toga što sa uvođenjem novog sistema napuštaju stari sistem koga poznaju i ulaze u zonu nesigurnosti. Proces promena zahteva od ljudi da napuste prošlost i prihvate nove procese i okruženje.

U kreiranju tranzisionog plana treba uključiti ljude koji su pogodjeni promenama. Njima treba dati šansu da prouče stare metode i predložene promene i da eventualno predlože druge mogućnosti koje dovode do cilja. Ljudi koji su uključeni u kreiranje promena uobičajeno imaju pozitivan stav prema njima. Kada je tranzisioni plan završen, treba kreirati tim za upravljanje tranzicijom.

Prilikom kreiranja tranzisionog plana treba imati na umu da postoje dve dimenzije promena: promene sistema i promene kod ljudi. Promene će se izvršiti uspešno samo kada su promene po obe dimenzije simultane i koordinisane.

PROMENE SISTEMA

Faze promena sistema

Promene sistema se mogu opisati sledećim fazama:

- Pojavljuje se neka potreba za promenama izazvana poslovanjem organizacije

- Predlaže se izrada projekta kojim će se definisati novi sistem. Novi sistem može da bude nova aplikacija ili nova organizacija rada ili bilo koja druga novina u radu organizacije.
- Formirani tim projektuje novi sistem
- Na osnovu projekta novog sistema vrši se njegovo uvođenje u organizaciju.
- Novi sistem se koristi na predviđeni način.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 2

Informacioni sistem

DEFINICIJA INFORMACIONOG SISTEMA

Informacioni sistem je deo poslovnog sistema čiji je zadatak da prikuplja, obrađuje, prosleđuje, prikazuje i čuva informacije

Vrlo je teško definisati šta je informacioni sistem (IS), pre svega što je teško utvrditi granice jednog informacionog sistema neke organizacije. Ipak, može se reći da je:

Informacioni sistem je deo poslovnog sistema čiji je zadatak da prikuplja, obrađuje, prosleđuje, prikazuje i čuva informacije.

Ovakva definicija je dovoljno opšta da obuhvati sve aspekte informacionog sistema. Ono što se iz nje može videti je da su mogući informacioni sistemi i bez primene računara. Računarski podržani informacioni sistemi imaju iste funkcije kao i klasični poslovni informacioni sistemi koji su postojali i pre primene računara. Korišćenjem računara, IS su postali efikasniji i pouzdaniji.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Termin IS se koristi i za naziv jedne od pet računarskih disciplina. U tom kontekstu informacioni sistemi su posebna disciplina računarstva koja se bavi integracijom informacionih tehnologija i poslovnih procesa, kako bi se zadovoljile poslovne potrebe za informacijama i rad organizacija učinio efikasnim i efektivnim. Stručnjaci koji se bave informacionim sistemima treba dobro da poznaju savremene informacione tehnologije i poslovne procese, kako bi na najbolji način stvorili informacioni sistem koji će generisati, obrađivati i distribuirati potrebne informacije. Zbog svoje obimnosti i stalnog pomeranja tehnoloških mogućnosti i zahteva, za nijedan informacioni sistem se ne može reći da je završen. Informacioni sistemi uvek zahtevaju pored daljeg razvoja i održavanje i prilagođavanje novim poslovnim zahtevima i standardima. Nije moguć razvoj informacionog sistema bez odličnog poznavanja organizacije rada i poslovnih pravila i potreba organizacije. Zbog toga stručnjaci iz ove oblasti najčešće rade u organizaciji za koju razvijaju informacioni sistem i sarađuju sa stručnjacima iz drugih disciplina kako bi što brže implementirali nova rešenja.

✓ 2.1 Elementi informacionih sistema

ELEMENTI INFORMACIONOG SISTEMA

Osnovni elementi IS-a su hardverska oprema i infrastruktura, softverska oprema, podaci, procedure i osoblje

Postoje različiti informacioni sistemi, kako prema nameni, komponentama i veličini. Ipak, svaki informacioni sistem se sastoji od pet elemenata:

- Hardverske opreme i infrastrukture
- Softverske opreme
- Podataka
- Procedura
- Osoblja

Svaki informacioni sistem ima svoje granice kojim su definisani njegov opseg i domašaj. Ipak, informacioni sistem se ne može posmatrati izdvojeno od ostalih delova poslovnog sistema jer bitno zavisi od njih. U svakom slučaju bilo koji informacioni sistem ima svoje okruženje. Veza između okruženja i IS je preko nekog interfejsa koji treba da, između ostalog, obezbedi zaštitu IS.

Svaki informacioni sistem ima ulaz kojim se elementi iz okoline unose u IS. Analogno tome postoji i izlaz iz IS. Pored toga svaki IS ima i ograničenja koja su nametnuta unutrašnjim i spoljašnjim faktorima (Slika 1).

Zajedničko za sve IS su i karakteristike koje ga opisuju. To su:

- Organizacija - struktura i poredak, hijerarhijske veze koje određuju formalnu komunikaciju i upravljački lanac
- Interakcija - način na koji pojedine komponente IS sarađuju sa drugim komponentama (na primer, marketing sa prodajom i proizvodnjom)
- Međuzavisnost - jedan podsistem zavisi od drugog (izlaz iz jednog često je ulaz u drugi podsistem)
- Integrisanost - mera povezanosti pojedinih komponenata.



Slika 2.1.1 Elementi informacionog sistema

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

✓ 2.2 Vrste informacionih sistema

VRSTE INFORMACIONIH SISTEMA (TPS, MIS, DSS, EIS)

Postoje različite vrste informacionih sistema, koji su više ili manje specijalizovani

Postoje različite vrste informacionih sistema, koji su više ili manje specijalizovani. Ovde se daje jedna moguća podela informacionih sistema.

- **Transaction Processing System** (TPS) ponekad se naziva i **Data Processing System**. Ovi informacioni sistemi se bave prikupljanjem i obradom podataka o poslovnim transakcijama (primeri: banke, pošte, računovodstva).
- **Management Information System** (MIS) su informacioni sistemi koji su namenjeni rukovodećoj strukturi da na osnovu sopstvenih podataka i podataka u okruženju, a korišćenjem matematičkih i statističkih metoda, dobiju informacije kojim će lakše upravljati organizacijom (primeri: marketing, planiranje proizvodnje, planiranje gradskog budžeta).
- **Decision Support System** (DSS) su sistemi za podršku u odlučivanju i koriste se za odlučivanje na osnovu nestruktuiranih podataka iz različitih izvora (primeri: odbrana zemlje, strategija razvoja preduzeća).
- **Executive Information System** (EIS) ima za cilj da olakša i podrži donošenje odluka rukovodicima u preduzeću, a pruža pristup relevantnim informacijama potrebnim da zadovolje strateške ciljeve kompanije.

PRIMERI INFORMACIONIH SISTEMA - TPS, MIS

Prikaz primera prethodno navedenih informacionih sistema

Primeri TPS-a:

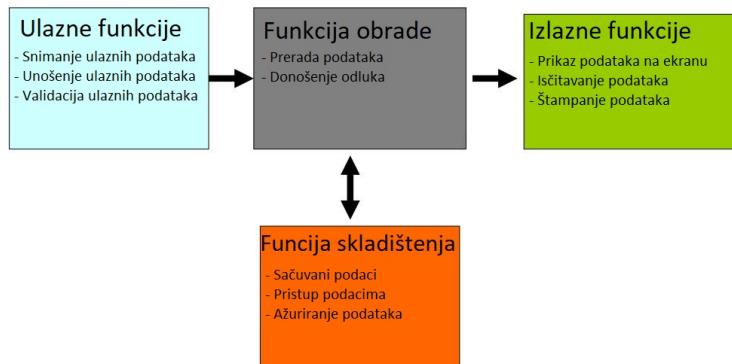
- **Bankomat:** vrše preradu podataka odmah nakon datih instrukcija od strane korisnika
- **Online porudžbine i rezervacije:** omogućavaju korisniku da putem veba kupi, naruči ili rezerviše nešto

Primeri MIS-a:

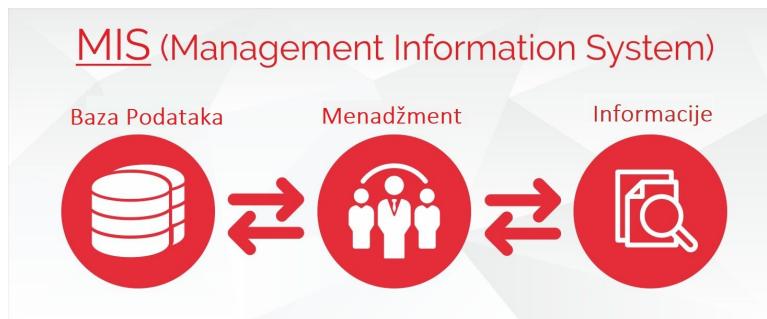
- **Firma:** zbog sakupljenih podataka koje ima, firma može da putem analize podataka i prodaja predviđi nove trendove
- **Menadžment:** kompanije putem menadžmenta mogu da tačno utvrde svoje prednosti i mane, performanse radnika, itd.

Transaction Processing System

FUNKCIONALNOSTI



Slika 2.2.1 - Primer TPS informacionog sistema



Slika 2.2.2 - Prikaz primera MIS informacionog sistema

PRIMERI INFORMACIONIH SISTEMA - DSS, EIS

Prikaz primera prethodno navedenih informacionih sistema, nastavak

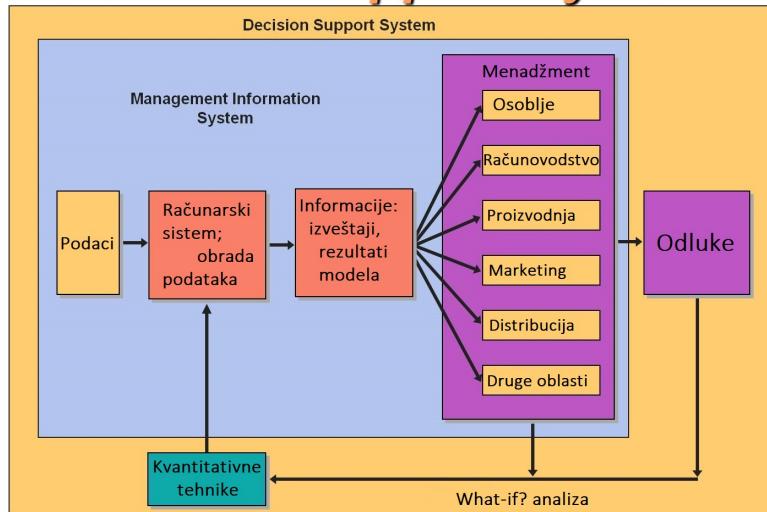
Primeri DSS-a:

- **Komunikacija:** omogućava lakši rad ukoliko na jednom projektu radi više osoba
- **Preduzeće:** sa svim prikupljenim podacima koje preduzeće ima, može se tačno predefinisani strategija koja će se pratiti kako bi se došlo do boljih rezultata

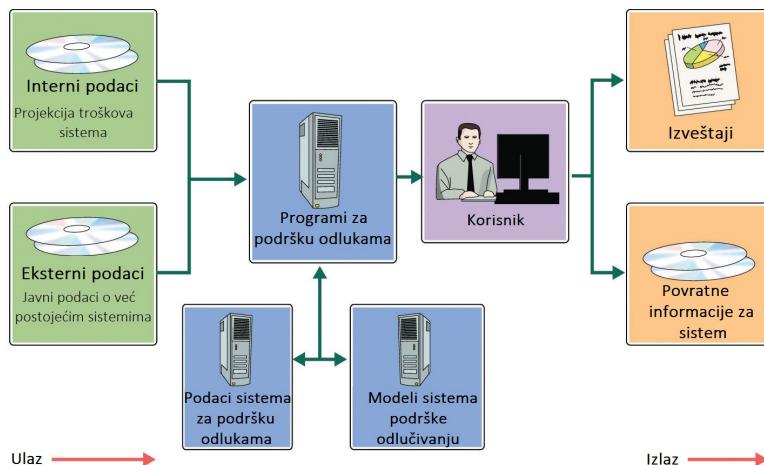
Primeri EIS-a:

- **Marketing:** korišćenjem internih (firminih) i eksternih informacija koje će kasnije menadžerima biti od koristi i olakšati im donošenje odluka u firmi o daljim postupcima

Decision Support System



Slika 2.2.3 - Prikaz DSS informacionog sistema



Slika 2.2.4 - Prikaz primene EIS informacionog sistema

✓ Poglavlje 3

Razvoj informacionog sistema

SDLC

SDLC je jedna od najrasprostranjenijih metoda za razvoj informacionih sistema

Statistika pokazuje da većina projekata informacionih sistema ne daje očekivane rezultate. Razvoj ili nabavka IS i njegova implementacija, održavanje ili zamena je kompleksan, skup i rizičan proces zbog velikog broja tehničkih, poslovnih, upravljačkih, strateških, organizacionih i socijalnih faktora koji utiču na ovaj proces. Nesistematski pristup razvoju IS sigurno neće dovesti do upotrebljivog sistema. Zbog toga su razvijene mnoge metode za razvoj informacionih sistema.

Životni ciklus razvoja sistema (*SDLC - System Development Life Cycle*) je jedna od najrasprostranjenijih metoda za razvoj informacionih sistema. SDLC i varijante ove metode se sreću i pod drugim imenima, kao što su: konvencionalna sistem analiza, **Waterfall** ili V model, **Linear Sequential Model**, **Classic Life Cycle**. Ova metoda se može definisati kao formalni proces razvoja IS kroz niz uzastopnih faza. Svaka faza se sastoji od više podfaza ili koraka. Nakon završetka jednog, prelazi se na naredni korak, mada metod predviđa povratnu spregu, odnosno povratak na prethodni korak ukoliko je to zbog uočenih propusta potrebno. Na žalost, nema saglasnosti među različitim autorima o tačnom broju i nazivu faza. U narednoj tabeli se daje spisak faza za SDLC i **Waterfall** metod (Tabela 1).

Cilj SDLC i drugih sličnih metoda je da se podelom celokupnog posla na upravljive faze reši problem ekstremne kompleksnosti razvoja IS, kako bi se razvio stabilan i funkcionalan informacioni sistem. Jedan od izlaza iz svake faze je obimna dokumentacija koja predstavlja osnov za rad u narednoj fazi. U teoriji, ove faze se obavljaju jedna za drugom, sekvensijalno. Međutim, u praksi, paralelne faze su često uslovljene jedna drugom, tako da je razvojni proces ustvari iterativni proces.

SDLC	Waterfall
Studija izvodljivosti	Sistemski zahtevi
Planiranje	Softverski zahtevi
Analiza	Analiza
Dizajn	Dizajn
Implementacija	Implementacija
Održavanje	Verifikacija
	Održavanje

Slika 3.1 Tabela-1 Faze SDLC i Waterfall metoda

ALTERNATIVNE METODE ZA RAZVOJ INFORMACIONIH SISTEMA

Izvršavanje svih faza SDLC metodologije ne garantuje da će novi sistem biti uspešan

Izvršavanje svih faza SDLC metodologije ne garantuje da će novi sistem biti uspešan. Kruti sistem pristupa omogućuje analitičarima i projektantima potrebnu kontrolu da završe razvojni zadatak, ali je u isto vreme ovakav pristup restriktivan prema zahtevima korisnika. Zahtevi koji su izneti u ranim fazama projekta ne mogu se lako promeniti. Zbog toga su se pojavile mnoge manje ili više uspešne alternativne metode za razvoj informacionih sistema i one će ovde biti samo nabrojane:

- Strukturni pristup razvoju sistema
- Metoda prototipa
- Evolucionarni pristup razvoju sistema (spirali model)
- Inkrementalni pristup razvoju sistema
- Brzi razvoj aplikacija (RAD – **Rapid Application Development**)
- Socio-tehnički pristup razvoju sistema
- Objektno orijentisani pristup
- Metoda „**Cleanroom**“ razvoja softvera
- **Outsourcing**

✓ Poglavlje 4

SDLC - Studija izvodljivosti

ŠTA JE SDLC - STUDIJA IZVODLJIVOSTI

Studija izvodljivosti je neophodna kako bi se ispitalo da li su predložene izmene isplative

Studija izvodljivosti istražuje postojeći sistem u svetlu novih zahteva i razmatra alternativna rešenja. Studija izvodljivosti je neophodna kako bi se ispitalo da li su predložene izmene isplative. Ovo podrazumeva da ukoliko trenutni system ne podržava trenutne poslovne zahteve, studija izvodljivosti treba da ispita i predloži alternativna rešenja. Sistem analitičari određuju da li su identifikovane potrebe za novim sistemom ili aplikacijom neophodne organizaciji, isplative i kompatibilne sa IT arhitekturom organizacije i poslovnom strategijom. Studija izvodljivosti sadrži sledeće korake:

- **Tehnička izvodljivost** - da li novi sistem može da bude razvijen i implementiran korišćenjem postojeće tehnologije. Često se javlja izazov da se obezbede finansije ukoliko su neophodni novi resursi kao što su hardver, softver ili ljudski kadar koji je potreban za rad na novom sistemu.
- **Economic feasibility** - analizira da li su resursi koji su potrebni za realizaciju novih zahteva finansijski isplativi i izvodljivi za kompaniju. Ova analiza efikasno ispituje da li su koristi koje će doneti novi informacioni sistem veće od troškova koje donosi razvoj tog sistema.
- **Pravna izvodljivost** - da li je predložen sistem u saglasnosti sa zakonom
- **Organizaciona izvodljivost** - da li postoji kompatibilnost organizacionih procedura starog sistema i rada novog sistema
- **Vremenska izvodljivost** - da li sistem može biti razvijen na vreme
- **Socijalna izvodljivost** - da li je sistem prihvatljiv za organizaciju i ljude u njoj na svim nivoima
- **Strateška izvodljivost** - da li se predloženi sistem uklapa u strateški poslovni i IT plan

Treba primetiti da je kompletну studiju izvodljivosti nemoguće uraditi u prvoj fazi jer ne postoje mnogi detalji o rešenju. Zbog toga neki autori ovu aktivnost stavljaju nakon faze analize sistema. Izveštaj studije izvodljivosti sadrži i plan projekta za razvoj ili nabavku novog sistema ili aplikacije.

MODELI

U svim ovim slučajevima, kao pomoć u odlučivanju, koristi se neki model odlučivanja

Pod studijom izvodljivosti spade i analiza isplativosti (engl. **cost-benefit analysis**). Upravljanje organizacijama često podrazumeva donošenje poslovnih odluka u uslovima ograničenih materijalnih i ljudskih resursa. Donošenje odluke o tome da li treba započeti novi projekat ili ne je teško ako nema jasnih pokazatelja koji će pomoći u odlučivanju. Poseban problem se javlja kada se od više projekata, zbog ograničenih resursa, treba odlučiti samo za jedan. I u ovom slučaju je potrebna metoda kojom će se svi predlozi proceniti kako bi moglo da se izvrši njihovo nepristrasno upoređenje i donošenje odluke. Jedan od najtežih problema u projektu reinženjeringa poslovnih procesa je sagledavanje finansijskih posledica reinženjeringa. U uslovima očekivanih radikalnih promena vrlo teško je unapred proceniti kakve će to finansijske efekte u budućnosti doneti organizaciji.

U svim ovim slučajevima, kao pomoć u odlučivanju, koristi se neki **model odlučivanja** (engl. **decision model**) koji predstavlja formalnu metodu za ocenjivanje projekta. Postoje dve osnovne kategorije modela odlučivanja:

- metode merenja dobiti
- modeli ograničene optimizacije.

U praksi se najčešće koriste metode merenja dobiti, pa će one ovde biti detaljnije opisane.

Metode merenja dobiti omogućuju da se uporede dobiti iz različitih projekta procenjivanjem na osnovu istih kriterijuma. Postoje tri vrste metoda merenja dobiti:

- **Cost-benefit analiza**
- **Bodovni model**
- **Ekonomski model**

COST-BENEFIT ANALIZA

Cost-benefit analiza izračunava troškove, projektovanu uštedu i projektovani prihod projekta

Cost-benefit analiza (analiza trošak-dobit) izračunava troškove, projektovanu uštedu i projektovani prihod projekta. Ovaj model je dobar izbor onda kada se odluka o izboru projekta donosi na osnovu toga koliko brzo će se povratiti sredstva uložena u projekat bilo uštedama, bilo povećanjem prihoda. Nedostatak korišćenja samo cost-benefit analize je u tome da ona ne uzima u obzir druge važne faktore nematerijalne prirode kao što su na primer strategijske vrednosti. Projekat koji će se otplatiti za najkraće vreme nije obavezno i najvažniji projekat za jednu organizaciju. Zbog toga se dobra cost-benefit analiza bavi i nematerijalnim dobitima. Ali da bi se oni uključili u analizu potrebno je proceniti njihovu vrednost.

Proces cost – benefit analize se može podeliti na sledeće faze:

- Identifikacija mogućih alternativa
- Definisanje alternativa na takav način da je moguće izvršiti pravedno i objektivno upoređivanje
- Procena troškova i dobiti tokom vremena
- Detaljna provera podataka koji nisu pouzdani
- Upoređivanje alternativnih rešenja na osnovu prikupljenih podataka

- Izbor najpovoljnijeg rešenja.

BODOVNI I EKONOMSKI MODELI

Ekonomska model je serija finansijskih proračuna koji obezbeđuju detaljne finansijske podatke celog projekta

Bodovni model (engl. **scoring model**) koristi unapred definisani listu kriterijuma na osnovu koje se projekti procenjuju. Svaki kriterijum ima svoj težinski faktor koji je povezan sa značajem kriterijuma. Za svaki kriterijum projektu se može dodeliti broj bodova u određenom dijapazonu. Bodovni model može da uključi finansijske podatke, ali i takve kriterijume kao što su tržišna vrednost, inovacija, uklapanje u korporacijsku kulturu i slične nenumeričke kriterijume. Na neki način, bodovni model je kombinacija objektivnih i subjektivnih kriterijuma. U nekim organizacijama se ustanovljava minimalni dozvoljeni broj bodova. Ukoliko broj bodova koji dobije projekat ne pređe ovu granicu, projekat se odbacuje ili vraća na doradu. Uvođenjem težinskih faktora organizacija može da neguje svoju kulturu. Na primer, organizacija koja želi da ima inovativne projekte će imati veliki težinski faktor za kriterijum inovativnosti. Kada se u takvoj organizaciji upoređuju dva projekta, moguće je da se bolje oceni projekat koji će u odnosu na druge kasnije vratiti ulaganja ako je inovativan. Nedostatak bodovnog modela je u tome što su rezultati procene onoliko dobri koliko su dobri kriterijumi i težinski faktori. Razvoj dobrog bodovnog modela je kompleksan proces koji zahteva učešće gotovo kompletног rukovodstva organizacije.

Ekonomska model je serija finansijskih proračuna koji obezbeđuju detaljne finansijske podatke celog projekta. Ovaj model je najsloženiji i zahteva iskusne ekonomiste i obilje podataka, mada se neki pokazatelji dobijaju na osnovu prepostavki.

▼ 4.1 SDLC – Faza planiranja

SDLC – FAZA PLANIRANJA - 4 AKTIVNOSTI

Tokom faze planiranja obavljaju se četiri ključne aktivnosti: razmatranje i odobravanje zahteva, zahtev za određivanje prioriteta, alokacija resursa, formiranje tima za svaku aktivnost

Planiranje aktivnosti počinje kada upravni odbor (engl. **steering committee**) primi zahteve da se razvije novi sistem. Ovaj se odbor obično sastoji od podpredsednika, IT osoblja, menadžera i korisnika koji nisu menadžeri.

Tokom faze planiranja obavljaju se četiri ključne aktivnosti:

- Razmatranje i odobravanje zahteva
- Zahtev za određivanje prioriteta
- Alokacija resursa

- Formiranje tima za svaku aktivnost

Projekti kojima je dat najviši prioritet su projekti koji se prvi implementiraju. Drugi projekti obično čekaju ili na odobrenje ili da se obezbede finansije za njihovu realizaciju.

▼ 4.2 SDLC - Faza analize

SDLC - FAZA ANALIZE (ANALIZA TRENUOTNOG SISTEMA)

Faza analiza podrazumeva da se obavi analiza rada trenutnog sistema, sakupljanje i definisanje zahteva za način rada novog sistema i davanja preporuka za novi sistem

Faza analiza podrazumeva da se obavi analiza rada trenutnog sistema, sakupljanje i definisanje zahteva za način rada novog sistema i davanja preporuka za novi sistem na osnovu obavljene analize.

Tehnike koje se koriste za sakupljanje zahteva su:

- Posmatranje podataka i protok informacija
- Analiza dokumenata koji se koriste u organizaciji
- Razgovori sa budućim korisnicima
- Upitnici/ankete
- Prototipovi koji su brzo napravljeni na osnovu zahteva korisnika

U ovoj fazi, informacije koje su prikupljene tokom faze istraživanja sistema obrađuju se detaljno kako bi se formalno utvrdile funkcionalnosti novog sistema. Sveobuhvatni i precizni zahtevi predstavljaju temelj za projektovanja sistema. Zahteve za projektovanje sistema treba pažljivo i sistematski dokumentovati tako da svako, uključujući i korisnike i IT profesionalce, može da koristi ove dokumente u kasnijim fazama.

▼ 4.3 SDLC -Faza dizajna

SDLC -FAZA DIZAJNA (DIZAJN NA OSNOVU SPECIFIKACIJA)

Ono što treba da proistekne iz ove faze su specifikacije za novi sistem koje se odnose na hardver, softver, ljudske resurse, informacije i procedure

U ovoj fazi informacioni sistem se projektuje tako da može da ispunji sve postavljene zahteve. Ono što treba da proistekne iz ove faze su specifikacije za novi sistem koje se odnose na hardver, softver, ljudske resurse, informacije i procedure. **Faza dizajna** ima dve podfaze:

- **Logički dizajn sistema** – tokom ove faze se pokušava da se dođe do odgovora kako će sistem reagovati na zahteve. Tokom ove faze treba razviti prototip. Tokom ove faze se projektuje arhitektura podataka koji se koriste kao ulaz i izlaz za sistem, s tim što se izlazni podaci obično prvo projektuju (svi meniji, izveštaji, i sl.) zbog toga što se na osnovu njih kasnije projektuju ulazi.
- **Fizički dizajn sistema** – definiše komponente koji su neophodni da bi automatizovani procesi radili. U ovoj fazi je neophodno da se definiše hardver i softver. Konkretno, u ovoj fazi je neophodno da se projektuje baza i programi koji će je koristiti. Projektovanje baze je veoma važno. Važno je da se definiše struktura svake tabele, kao i privilegije koje će svako od korisnika imati prilikom pristupanja bazi.

Kada se projektuje sistem, potrebno je da se prate osnovni principi koji omogućavaju modularnost, skalabilnost, održivost i ponovno korišćenje delova ili celog sistema.

Glavne aktivnosti tokom projektovanja sistema su:

- Projektovanje i dizajn interfejsa
- Projektovanje ulaza i izlaza, na primer, sadržaj informacija i njihov format, kako bi sistem mogao da ih procesuje
- Dizajn softvera.

▼ 4.4 SDLC – Faza implementacije

SDLC – FAZA IMPLEMENTIRANJA

Implementacija sistema predstavlja fazu u kojoj se novi sistem, koji je u skladu sa postavljenim zahtevima, instalira i osposobljava za korišćenje

Implementacija sistema predstavlja fazu u kojoj se novi sistem, koji je u skladu sa postavljenim zahtevima, instalira i osposobljava za korišćenje. Ova faza se sastoji iz sledećih aktivnosti:

- Razvoj softvera
- Instaliranje i testiranje novog sistema
- Obuka korisnika
- Prelazak na novi sistem

Sve faze SDLC metode su jednako važne i imaju uticaja na krajnji proizvod. Međutim, poslednji korak, odnosno prelazak na novi sistem, je naročito osetljiv. U ovoj fazu korisnici mogu stvoriti na odbojnost prema novom sistemu i nastalim promenama u poslovanju.

Kako bi korisnici mogli da se efikasno prilagode novom sistemu i kako bi uopšte znali kako da ga koriste, potrebno je da prođu neku vrstu obuke koja će im omogućiti sticanje potrebnih znanja i veština. Obuka korisnika treba da počne što ranije je moguće. Za korisnike koji su dobro obučeni postoji manja verovatnoća da će razviti odbojnost prema novom ili

modifikovanom sistemu. Takođe, tokom obuke, mogu se videti eventualni problemi i propusti koje novi sistem ima, tako da se oni mogu otkloniti pre nego što se sistem pusti u rad.

Tokom procesa implementacije mogu se javiti mnoge promene kod zaposlenih. Ove promene se mogu opisati na sledeći način:

- Kada ljudi postanu svesni predstojećih promena, ljudi prvo osećaju strah zbog neizvesnosti o tome kako će predstojeće promene uticati na njih lično
- U cilju prevazilaženja straha i stvaranja želje da se promene prihvate, neophodno je da se ljudi temeljno informišu o razlozima zašto je promena neizbežna i da im se ukaže na to što može da se desi ako se ne promene ne sprovodu. Na primer, ako novi antivirusni program nije instaliran, postoji velika opasnost od narušavanja integriteta informacionog sistema. U ovoj fazi potrebno je motivisati ljude ne samo da prihvate promene nego i da doprinesu u kvalitet promena i same implementacije.
- Kada su ljudi koji su uključeni u rad pokazuju želju da se izmene sprovedu, oni treba da budu edukovani o tome kako da sprovedu te promene što efikasnije. Samo ljudi koji imaju dovoljno znanja mogu da budu uspešni u implementaciji novog sistema.
- Obrazovani ljudi sada imaju mogućnost da sprovode promene na dnevnoj bazi i da doprinose u tranziciji na novi sistem rada.

RAZVOJ SOFTVERA

Najčešći je slučaj da je kupljeni softver potrebno doraditi i prilagoditi

Ukoliko je kompanija kupila već gotov softver kome nije potrebna nikakva dorada i biće koristan takav kakav jeste, onda se ova faza može preskočiti. Međutim, najčešći je slučaj da je kupljeni softver potrebno doraditi i prilagoditi. Takođe, može se desiti da gotovo rešenje ne postoji i da je potrebno da se razvije novi softver. Razvoj softvera podrazumeva da se prate sledeće aktivnosti kao što su:

- Analiza zahteva
- Razvoj rešenja
- Validacija rešenja
- Implementacija rešenja
- Testiranje
- Dokumentacija

INSTALIRANJE I TESTIRANJE NOVOG SISTEMA

Kada kompanija kupi novi hardver i softver, potrebno je da ih neko instalira i testira njihov rad

Kada kompanija kupi novi hardver i softver, potrebno je da ih neko instalira i testira njihov rad. Potrebno je da se proveri da li svi delovi sistema dobro rade. Od vitalnog je značaja da se svi delovi pažljivo provere pre nego što se puste u rad, tako da se sve potencijalne greške uhvate pre nego što dođe do grešaka u poslovanju. Testovi koji se mogu sprovesti su "unit"

testovi, sistemski testovi, testovi za proveru integrisanosti sistema (engl. **integration tests**) i test prihvatljivosti (engl. **acceptance tests**).

“**Unit**” test verifikuje da li program ili neki objekat rade nezavisno od celog sistema. Sistemski test verifikuje da ceo program i sve funkcionalnosti rade kako treba kao jedna celina. Test za proveru integrisanosti proverava da li aplikacija radi kako treba zajedno sa drugim aplikacijama sa kojima je potrebno da komunicira u radu. Test prihvatljivosti se obavlja sa krajnjim korisnicima koji će raditi na tom sistemu, kako bi se proverilo da li je sistem operativan i u praksi.

OBUKA KORISNIKA

Plan edukacije korisnika treba da sadrži ko će biti edukovan, ko će vršiti edukaciju i gde će ona biti obavljena

Prilikom planiranja edukacije treba odgovoriti na sledeća pitanja:

1. Ko će biti edukovan? Razume se da kroz edukaciju obavezno treba da prođu potencijalni korisnici aplikacije i administratori aplikacije. Pored toga, menadžerima obavezno treba dati jedan opšti pregled, kako bi znali šta mogu očekivati od aplikacije.

2. Ko će vršiti edukaciju? Idealan tim za edukaciju se sastoji od:

- korisnika koji je ekspert u domenu za koji je aplikacija pisana
- IT specijaliste koji poznaje tehnologije koje aplikacija koristi
- profesionalnog predavača koji poznaje metodologije i tehnike edukacije.

3. Gde će se vršiti edukacija? Za mesto edukacije se može odabrat jedna od tri opcije:

- na radnom mestu korisnika
- u prostorijama za obuku organizacije korisnika aplikacije
- van organizacije.

Edukacija u prostorijama organizacije je jeftinija ali je podložna prekidima zbog „hitnih“ poslova.

PRELAZAK NA NOVI SISTEM

Prelazak na novi sistem se može izvesti direktno, paralelno ili postupno

Poslednja aktivnost faze implementacije je prelazak na novi sistem. Prelazak na novi sistem se može izvesti direktno, paralelno ili postupno.

- **Direktno uvođenje** predstavlja prekid korišćenja starog sistema određenog datuma, što podrazumeva da se samo može od tog dana koristiti novi sistem. O ovome se obaveštavaju svi korisnici sistema. U međuvremenu se vrši konverzija podataka. Ovo je najjeftiniji i najbrži način, ali sa druge strane najrizičniji jer u slučaju lošeg funkcionisanja novog sistema nema rezervne varijante. S druge strane, ovakav prelazak na novi sistem ne iziskuje dodatne troškove.

- **Paralelno uvođenje** podrazumeva da se startuje sa korišćenjem novog sistema, ali se ne prekida rad sa starim. Tek kada se dokaže da je novi sistem potpuno efektivan, prekida se sa korišćenjem starog sistema. Ovo je najsigurnija alternativa. U slučaju da novi sistem nije funkcionalan postoji alternativni sistem koji funkcioniše. Pored toga moguće je testirati novi sistem upoređujući njegove izlaze sa starim sistemom. Nedostatak ove metode je što zahteva dodatni rad, što ga čini skupim.
- **Postupno uvođenje** se obično koristi u velikim kompanijama. Ovakvo uvođenje obično podrazumeva da se različiti odseci kompanije prebacuju na novi sistem u različitim vremenskim intervalima. Tokom definisanog vremena se postupno isključuju pojedine funkcije starog sistema i istovremeno uključuju adekvatne funkcije novog.

KONVERTOVANJE PODATAKA

Vrlo je čest slučaj da se prilikom uvođenja novog ili promena postojećeg sistema javlja potreba za konvertovanjem podataka, formata datoteka ili tabela iz starog u novi format

Vrlo je čest slučaj da se prilikom uvođenja novog ili promena postojećeg sistema javlja potreba za konvertovanjem podataka, formata datoteka ili tabela iz starog u novi format. U nekim slučajevima su podaci na staroj opremi koju je vrlo teško povezati sa novom. Posebni problemi se javljaju ako prethodno nije postojala nikakva aplikacija za neku oblast, pa je potrebno uneti podatke iz papirne dokumentacije.

U slučajevima kada postoje podaci u digitalnom obliku najlakši način konverzije je izrada posebnih aplikacija koje će automatski izvršiti prevodenje podataka iz starog u novi format. Ukoliko je potrebno novom sistemu dodati još neke podatke koje stari sistem nije zahtevao, prave se posebne rutine za unos takvih podataka.

Mnogo veći problem je konverzija podataka iz papirnog u digitalni oblik. Organizacija koja radi više godina ima obilje podataka koji su zapisivani u različitim formama. Zbog toga je ovo skup i dugotrajan proces, koji je podložan greškama pri konverziji. Zbog toga se primenjuje metoda „**dan-jedan**“ kojom se delimično mogu ublažiti ovi problemi, a sistem postaje funkcionalan od prvog dana. Metoda dan-jedan se zasniva na činjenici da su samo neki od svih podataka koje ima organizacija aktivni. **Aktivni podaci** su podaci o skorašnjim transakcijama. Može se očekivati da će u bliskoj budućnosti ovi podaci biti potrebni za dodatne obrade, kao što su na primer godišnji izveštaji. **Pasivni podaci** se odnose na transakcije iz prošlosti. Ne očekuje se da će ovi podaci biti potrebni za dalju obradu, osim ako organizacija ne pravi dugogodišnje analize u cilju praćenja trendova. Odnos aktivnih i pasivnih podataka može biti i veći od 1:4. Na primer, vrlo je verovatno da od 100 organizacija sa kojima je posmatrana organizacija radila poslednjih 20 godina, 80 više ne postoji.

Metod dan-jedan omogućuje da se konverzija podatka vrši nakon, a ne pre početka rada sa novim sistemom. Po početku rada sa novom aplikacijom novi podaci o transakcijama se regularno unose. Ako postoji potreba za vezom između novih i starih podataka, unose se i stari podaci. Na početku rada po ovoj metodi unosi se veliki broj starih podataka, ali tokom vremena opada potreba za unosom jer sve manji broj aktivnih podataka nije već unet. Organizacija onda može da odluči hoće li uneti i preostale, pasivne, podatke.

❖ 4.5 SDLC - Održavanje

SDLC - ODRŽAVANJE SISTEMA

Ova faza uključuje aktivnosti kao što su održavanje, nadgledanje performansi i procenjivanje bezbednosti sistema.

Kada je sistem instaliran potrebno je da se obezbedi trajno **održavanje sistema**. Ova faza uključuje aktivnosti kao što su održavanje, nadgledanje performansi i procenjivanje bezbednosti sistema.

Kada se završi period prelaska sa starog na novi sistem, što je tipično oko nekoliko meseci nakon što se pusti novi sistem u rad i nakon što su otklonjene sve greške, obavlja se evaluacija novog sistema. Cilj evaluacije je da se odredi da li novi sistem radi u skladu sa očekivanjima i definisanim specifikacijama. Evaluacija treba da uključi krajnje korisnike, tehničko osoblje, menadžment i projektante. Izveštaj o evaluaciji treba da uključi:

- Ciljne performanse novog sistema
- Trenutno performanse sistema i njihovo upoređenje sa ciljnim performansama
- Listu svih sistemskih karakteristika koje nisu u skladu sa ciljevima
- Akcije koje treba preduzeti kako bi se novi sistem doveo na očekivani nivo
- Rok za izvršenje ovih aktivnosti
- Datum za konačnu kontrolu sistema

U ovoj fazi, odgovornosti koji je imao razvojni tim se prenose na samu organizaciju. Formalno gledano, u ovoj fazi treba potpisati sporazume koji definišu prebacivanje nadležnosti sa razvojnog tima. Prenos odgovornosti treba da bude formalizovan ugovorom koji treba da imaju sledeće elemente:

- Ciljnu performansu novog sistema
- Prag koji se koristi kao donja granica za određivanje lose performance; ako se performansa spusti ispod ovog praga tada se može zaključiti da sistem ne radi na zadovoljavajućem nivou
- Predviđeni životni vek sistema, nakon čega da će biti potrebna dorada ili zamenja sistema
- Uslove održavanja kao što su cena i vreme za potrebne popravke i pronalaženje problema u sistemu
- Uslovi pod kojima će se vršiti izmene u skladu sa novim zahtevima korisnika.

❖ 4.6 Faze SDLC metode

SDLC FAZE

SDLC metoda se može definisati kao formalni proces razvoja IS kroz niz uzastopnih faza

SDLC - Studija izvodljivosti

SDLC - Faza planiranja

SDLC - Faza analize

SDLC - Faza dizajna

SDLC - Faza implementacije

SDLC - Održavanje

SDLC UKRATKO

System Development Life Cycle

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 5

RAD metoda

RAPID APPLICATION DEVELOPMENT - RAD

RAD metodologiju moguće je primeniti kod aplikacija koje se izvode samostalno, ako performanse nisu kritične, ako pouzdanost aplikacije nije kritična

Brzi razvoj aplikacija (Rapid Application Development - **RAD**) je metodologija razvoja softvera (ili sistemskog razvoja) koja se fokusira na brzu izradu radnog modela softvera, dobijanje povratnih informacija od korisnika, a zatim koristeći tu povratnu informaciju da ažurira radni model. Nakon nekoliko iteracija razvoja, razvija se i implementira finalna verzija. RAD metodologiju moguće je primeniti kod aplikacija koje se izvode samostalno, ako performanse nisu kritične, ako pouzdanost aplikacije nije kritična, kada je proizvod namenjen visoko specijalizovanom tržištu i dr.

RAD metodologija se sastoji iz 4 faze:

- **Planiranje zahteva.** Ova faza je slična preliminarnoj analizi, sistemskoj analizi i fazama projektovanja SDLC-a. U ovoj fazi se definišu opšti zahtevi sistema, identificuje se tim i utvrđuje se izvodljivost.
- **Korisnički dizajn.** U ovoj fazi manja grupa korisnika radi sa sistemskim analitičarima, dizajnerima i programerima radi interaktivnog stvaranja dizajna sistema. Jedna tehnika za rad sa svim ovim raznim akterima je takozvana JAD sesija. JAD je akronim zajedničkog razvoja aplikacija (**joint application development**). JAD sesija podrazumeva da sve zainteresovane strane zajedno diskutuju o dizajnu sistema i njegovom najboljem rešenju za sve. Programeri aplikacija takođe sede na ovom sastanku i posmatraju, pokušavajući da razumeju suštinu zahteva.
- **Izrada.** U fazi izrade programeri koji rade sa korisnicima, grade sledeću verziju sistema. Ovo je interaktivni proces, a promene se mogu izvršiti pošto programeri rade na programu. Ovaj korak se izvršava paralelno sa korakom korisničkog dizajna na iterativan način, sve dok se ne razvije prihvatljiva verzija proizvoda.
- **Brza tranzicija.** U ovoj fazi se sistem pušta u rad

Kao što se može videti, RAD metodologija ima manje faza od SDLC-a. Isto tako se može primetiti da mnogi od SDLC koraka su kombinovani i bazira se na učestvovanju korisnika u dizajnu i na iteracijama. Ova metodologija je mnogo pogodnija za manje projekte od SDLC-a i ima dodatnu prednost davanja korisnicima mogućnostima da pruže povratne informacije tokom čitavog procesa. SDLC zahteva više dokumentacije i pažnju na detalje i dobro odgovara velikim projektima. RAD ima više smisla za manje projekte koji se moraju brzo razvijati.

✓ Poglavlje 6

Agilne metode razvoja

AGILNE METODOLOGIJE

Agilne metode razvoja predstavljaju grupu metodologija koje koriste iteracije sa kratkim vremenskim intervalima sa fokusom na kvalitet i pažnju na detalje

Agilne metode razvoja predstavljaju grupu metodologija koje koriste iteracije sa kratkim vremenskim intervalima sa fokusom na kvalitet i pažnju na detalje. U svakoj iteraciji se razdvajaju zadaci na male inkrementalne promene sa minimalnim planiranjem. Iako se smatraju drugačijim od RAD-a, oni dele neke iste principa: iterativni razvoj, interakciju korisnika, sposobnost promene. Agilne metodologije se zasnivaju na "Agilnom manifestu", koji je prvi put objavljen 2001. godine.

Karakteristike agilnih metoda uključuju:

- Mali kros-funkcionalni timovi koji uključuju članove razvojnog tima i korisnike;
- Dnevni sastanci o statusu kako bi razgovarali o trenutnom stanju projekta;
- Kratki vremenski okviri (od nekoliko dana do jedne ili dve nedelje) koji se određuju za svaku inkrementalni napredak (zadatak) koji treba da bude završen u tom periodu;
- Na kraju svake iteracije potrebno je da definisani zadatak bude završen i da ga je moguće prikazati svim zainteresovanim stranama.
- Tim definiše zahteve za iteraciju, razvija kod, definiše i izvršava integrisane testove a korisnici verifikuju rezultate.
- Verifikacija se izvršava mnogo ranije u razvojnom procesu nego kod Waterfall modela

Cilj agilne metodologije je da obezbedi fleksibilnost iterativnog pristupa uz obezbeđivanje kvalitetnog proizvoda.

▼ Poglavlje 7

Upravljanje promenama

UPRAVLJANJE PROMENAMA SISTEMA

Promene u poslovnom sistemu su neminovne, one mogu biti planirane i nametnute.

Promene u poslovnom sistemu su neminovne. One mogu biti planirane i nametnute. **Planirane promene** su posledica svesnog razmišljanja i planiranja. **Nametnute promene** su spontane ineplanirane. Promene mogu biti nametnute iz dva razloga.

1. Rukovodstvo organizacije je donošenjem nekih odluka izazvalo promene za koje nije bilo očigledno da su u vezi sa odlukama. Rukovodstvo nije bilo svesno da će izazvati promene, pa nije moglo ni da ih planira.
2. Promene su nametnute spoljnim faktorima kao što su na primer ekonomija, konkurenčija, politička klima, nagla promena tehnologije ili pojava nove vrste računarskih virusa.

Nametnute promene mogu da se javi čak i onda kada se realizuje dobro planirani program promena zato što ih članovi tima jednostavno nisu detektovali kao posledicu planiranih promena.

Ako se promene posmatraju u vremenskom domenu onda se može reći da promene mogu biti:

- tranzicione i
- kontinualne.

Tranzicione promene su retke, kratkotrajne i diskontinualne. Nekada se nazivaju i epizodne ili radikalne jer zamenjuju jednu strategiju ili program drugim.

Kontinualne promene su dugotrajne, evolucionarne i kumulativne. Nekada se nazivaju i inkrementalne jer se do krajnjeg stanja dolazi malim inkrementima promena tokom dugog vremenskog perioda.

✓ 7.1 Upravljanje projektom

PROJEKAT INFORMACIONIH SISTEMA

Istraživanja pokazuju da je procenat uspešnih projekata informacionih sistema (IS) vrlo mali i pored primene raznih novih tehnologija i alata, kao i višedecenijskog iskustva IT zajednice

Upravljanje projektima je proces, koji podrazumeva planiranje i kontrolu svih aktivnosti tokom razvoja sistema. Cilj upravljanja projektima je da isporuči sistem koji zadovoljava sve zahteve, a da pri tom vodi računa da se projekat ispunjava u planiranim finansijskim okvirima i isplaniranom vremenskom periodu. U manjim preduzećima obično jedna osoba rukovodi projektom. Međutim, u većim kompanijama može biti više ljudi, ali su to obično projekt menadžer (engl. **project manager**) i vođa projekta (engl. **project leader**). Vođa projekta obično kontroliše budžet i raspored projekta, dok projekt menadžer kontroliše aktivnosti u toku razvoja sistema.

Istraživanja pokazuju da je procenat uspešnih projekata informacionih sistema (IS) vrlo mali i pored primene raznih novih tehnologija i alata, kao i višedecenijskog iskustva IT zajednice u projektovanju IS. Jedan od glavnih razloga neuspešnosti je što se IS i IT projekti tretiraju kao i standardni projekti iako ima mnogo razlika kao što su:

- IS projekti imaju slabo definisane granice. Tokom rada na projektu širi se opseg projekta dodavanjem novih funkcija i karakteristika informacionom sistemu.
- Stručnjaci projektnog tima IS projekata su uobičajeno opterećeni svakodnevnim aktivnostima kao što su održavanje i podrška trenutnog sistema. Zbog toga rukovodioci IS projekta pokušavaju da u projekat uključe minimalni broj ljudi. Nedovoljan broj ljudi u fazi snimanja zahteva i poslovnih pravila dovodi do loše definisanih projektnih zahteva.
- Upravljačka struktura organizacije se sastoji od ljudi koji po pravilu nisu IT eksperti. Zbog toga postoje problemi u komunikaciji sa članovima projektnog tima, što rezultira nerazumevanjem ili lošom interpretacijom zahteva i poslovnih pravila.
- IS projekti su pogodniji za timski rad u odnosu na druge projekte.
- Mnogi delovi IS projekta mogu da se simultano odvijaju

DODATNI RAZLOG ZA NEUSPEH IS PROJEKATA

Projektni tim nema članove koji dobro poznaju poslovne procese i specifična znanja vezana za upravljanje projektom

Pored ovog razloga postoje i drugi razlozi zašto su IS projekti tako često neuspešni. Ovde se navode samo neki:

- Postoji otpor unutar organizacije ka promenama u IS
- Projektni tim nema članove koji dobro poznaju poslovne procese i specifična znanja vezana za upravljanje projektom

- Nema dobre motivacije članova projektnog tima.

Da bi jedan IT projekat bio uspešan, potrebno je, između ostalog, sastaviti dobar tim. Projektni tim treba tako koncipirati da članovi tima imaju znanja iz sledećih oblasti:

- Upravljanje opsegom projekta
- Upravljanje vremenom
- Upravljanje troškovima
- Upravljanje kvalitetom
- Upravljanje ljudskim resursima
- Upravljanje komunikacijama
- Upravljanje rizikom
- Upravljanje nabavkom
- Upravljljane integracijom

Posebno je značajan rukovodilac projekta, koji mora da ima znanja iz mnogih od gore navedenih oblasti, iskustvo u rukovođenju ali i niz veština kao što su:

- Liderstvo
- Oralne i pisane komunikacije
- Slušanje
- Organizovanje
- Upravljanje vremenom
- Planiranje
- Rešavanje problema
- Rešavanje konflikta
- Pregovaranje
- Građenje tima

VREMENSKA ORGANIZACIJA

Raspored obavljanja zadataka u projektu se može prikazati vizuelno sa raznim alatima

Kada su aktivnosti za svaku fazu definisane, planiranje rasporeda za obavljanje svih aktivnosti može da počne. Prvi korak je da se odredi po kom redu će se aktivnosti obavljati. Treba napomenuti da su mnogi od zadataka ili aktivnosti međusobno zavisni. Ukoliko ne postoji zavisnost između određenih aktivnosti, onda se oni mogu izvršavati paralelno. Uobičajena zavisnost između dve aktivnosti jeste da izvršavanje jedne aktivnosti uslovjava da počne izvršenje druge aktivnosti. Ovo je takozvana forsirana zavisnost. Zavisnost može biti prouzrokovana nekim spoljnim uticajem. Na primer, instalacija programa ne može da počne dok dobavljač ne dostavi server.

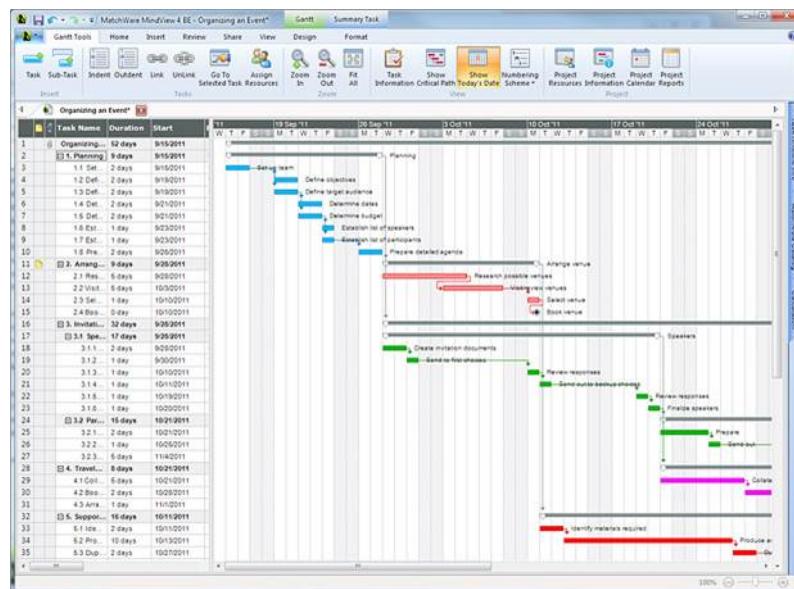
Raspored obavljanja zadataka u projektu se može prikazati vizuelno sa raznim alatima. Jedan od načina prikazivanja je pomoću mrežnih dijagrama (PERT dijagrami) i GANTT dijagrama. Ovi se alati koriste kako bi vizuelno prikazali zavisnosti između njih i raspored koji prikazuje kada je predviđeno da se počne i završi sa određenom aktivnošću. On prikazuje aktivnosti na vertikalnoj osi, a na horizontalnoj osi prikazuje planirano vreme za izvršenje za svaku

aktivnost. PERT dijagrami mogu biti komplikovani od GANTT dijagrama ali su prikladniji za vremensko planiranje većih projekata.

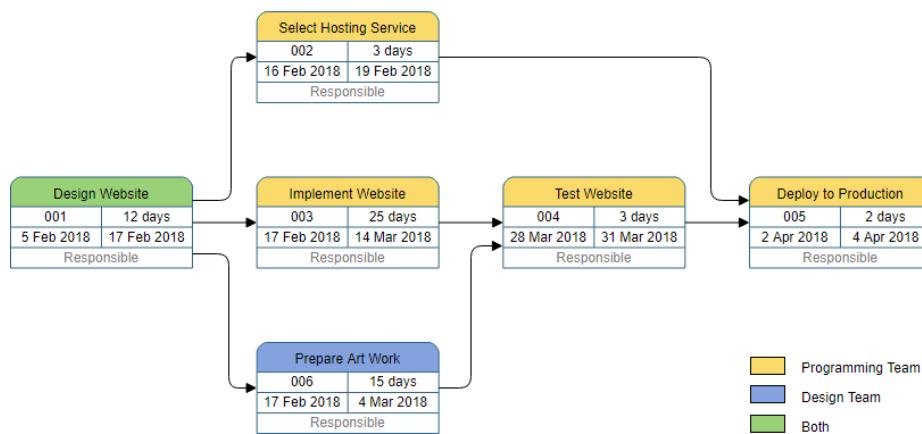
Planiranje rasporeda rada podrazumeva da se isplanira i vremensko trajanje svake od aktivnosti i njihovih delova. Ono što treba imati u vidu prilikom procene potrebnog vremena je potrebni radni sati, mogućnost da neki od angažovanih na projektu rade samo delimično radno vreme, treba uračunati vikende i praznike kada se ne radi i sl. U svakom slučaju, ukupno procenjeno radno vreme za projekat treba da bude toliko koliko je realno potrebno da se završi sa projektom do planiranog datuma.

PERT I GANT PRIMERI

Prikaz primera



Slika 7.1.1 GANT dijagram



Slika 7.1.2 PERT dijagram

KREIRANJE OSNOVNE FORME GANT DIJAGRAMA

Create a Basic Gantt Chart

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 8

Pokazna vežba: GanttProject

PLANIRANJE PROJEKTA

Svaki projekat mora imati definisane ciljeve

Predviđeno vreme pokazne vežbe je 105 minuta.

Svaki projekat mora imati definisane ciljeve:

- * Rok završetka projekta;
- * Ograničeni budžet;
- * Krajnji rezultat / proizvod ili sl. rada na projektu.

S obzirom na to da je svaki projekat drugaciji, ne postoji aprioran, utvrđen način specifikovanja kako svaki od njih treba da se uradi. Novi projekti donose nova pitanja. Za one koji počinju, projektni tim mora da odgovori na pitanja šta treba da se uradi, kako će se to raditi, ko će raditi, kojim redom, za koliko i kada. To znači da:

1. Utvrđuju se ciljevi, zahtevi i obim rada. Ovi elementi specifikuju završene proizvode, željene rezultate i vremenske, troškovne i performansne ciljeve (Šta, za koliko i do kada?).
2. Specifične radne aktivnosti, zadaci ili poslovi za postizanje ciljeva su razloženi, defenisani i popisani (Šta?).
3. Pravi se organizacija projekta koja specifiše odeljenja, podizvođače i menadžere odgovorne za radne aktivnosti (Ko?).
4. Pravi se raspored sa tajmingom radnih aktivnosti, krajnjih rokova i majlstonova (Kada, kojim redom?)
5. Pravi se budžet i plan resursa, koji pokazuje summu i tajming resursa i rashode za radne aktivnosti i slične stavke (Koliko i kada?).
6. Priprema se proračun vremenskih, troškovnih i performansnih projekcija za kompletiranje projekta (Koliko treba vremena, koliko će koštati, i kada će projekat biti završen?).

Dati koraci se uvek prate, jer svaki projekat je po nečemu jedinstven, zahteva različite resurse i mora da bude završen u specifičnom vremenu, trošku i performansnim standardima koji zadovoljavaju zahteve korisnika. Kad god su projekti slični, veliki deo planiranja zasniva se na prošlom iskustvu i starim dosjeima.

GLAVNI PLAN PROJEKTA

Glavni plan projekta treba da reflektuje stanje projekta u skladu sa promenama zahteva i okruženja.

Glavni plan projekta treba da reflektuje stanje projekta u skladu sa promenama zahteva i okruženja. Dokumentovani plan treba da obuhvati potrebne resurse, očekivane izlazne rezultate i potrebnii rad; da sadrži plan angažovanja relevantnih učesnika i projektnog tima; da dokumentuje angažovanje menadžmenta i drugih provajdera resursa i da čini bazu za upravljanje projektom.

Finalni plan IT projekta treba da sadrži minimalno sledeće elemente:

1. Uvod. Ovaj deo plana predstavlja opis ciljeva projekta i ograničenja sa kojima su suočeni realizatori (budžet, vreme, resursi i sl.) i o čemu treba da vodi računa menadžer IT projekta;
2. Organizacija projekta. Sadrži opis načina na koji će razvojni tim biti organizovan, koji će ljudi biti uključeni i sa kojim ulogama u timu;
3. Analiza rizika. Sadrži opis mogućih rizika na projektu, procenu verovatnoće da se ti rizici pojave, kao i strategije za izbegavanje / smanjenje rizika;
4. Zahtevi za hardverskim i softverskim resursima. Sadrži specifikaciju hardvera i softverske podrške za planirani razvoj. Ukoliko je neophodno kupiti hardver, u ovaj deo plana treba uključiti i troškove i raspored isporuke;
5. Podela poslova. Sadrži opis poslova i aktivnosti na projektu i identifikaciju kritičnih trenutaka – majlstonova, kao i rezultata koji treba da se ostvare obavljanjem pojedinačnih aktivnosti;
6. Raspored aktivnosti na projektu. Sadrži opis međuzavisnosti između aktivnosti, procenjeno vreme za izvršavanje svake aktivnosti i alokaciju ljudskih resursa po aktivnostima;
7. Mehanizmi monitoringa i izveštavanja. Propisuje se sadržaj izveštaja menadžera projekta, vreme kada treba ti izveštaji da se podnesu, kao i mehanizmi praćenja progresu realizacije projekta.

GANTOV DIJAGRAM

Gantov dijagram je metoda grafičkog prikazivanja informacija koju se često koristi za utvrđivanje rasporeda aktivnosti

Gantov dijagram ili gantogram dobio je ime po Henry-u Laurecu-u Ganttu (1861-1919), naučniku i inžinjeru koji ga je osmislio 1917. Njemu u čast danas se mladim inovativnim menadžerima za doprinos zajednici i naučna dostignuća dodjeljuje medalja Henry Laurence Gant (The Henry Laurence Gant Medal).

Gantogram je dijagram koji se sastoji od koordinatnog sistema. U koordinatnom sistemu horizontala je vreme, a vertikala resursi na kojima se odvijaju pojedini radni nalozi, po

operacijama i vremenom početka i završetka svake operacije. Vremenski interval se određuje shodno vrsti proizvodnje, odnosno dužini proizvodnog ciklusa.

To je, u stvari, metoda grafičkog prikazivanja informacija koju se često koristi za utvrđivanje rasporeda aktivnosti. Tipični Gantov dijagram grafički prikazuje pogreške u zadatku, vreme potrebno za kompletну izradu zadatka kao i procenat urađenog dela zadatka. Svaki stubić (linija) predstavlja jedan zadatak s vremenskim vrednostima, a redovi sadržaj zadataka.

Gantov dijagram može biti jednostavna verzija: grafikon iscrtan na papiru ili kompleksna automatizovana verzija izražena korištenjem programskih aplikacija poput Microsoft Project ili Excel. Često se primenjuje kod izgradnje infrastrukturnih objekata poput brana i međunarodnih autoputeva, ali i u firmama gdje nadzornik/menadžer pomoću Gantovog dijagrama izrađuje raspored zaposlenih po smenama i pomoću kojeg može na jednostavan način kontrolisati redovnost, tačnost i efikasnost zaposlenih. Koriste ga i vođe timova za izradu rasporeda s popisom igrača i njihovih uloga, termina sastanaka i utakmica. Ovo su samo neki od primera korišćenja gantograma.

PREUZIMANJE GANTTPROJECT SOFTVERA

Kako bi kreirali gantogram koristićemo softver Gantt project.

Kako bi kreirali gantogram koristićemo softver Gantt project. Gantt project je besplatan softver i može se preuzeti na adresi

<https://www.ganttproject.biz/download>

The screenshot shows the GanttProject website homepage. At the top, there's a logo consisting of a stylized yellow and grey gear-like icon followed by the text "GanttProject" in a blue sans-serif font, with a yellow horizontal bar underneath. Below the logo is a yellow banner containing the text "Free project scheduling and management". A message at the top left says "Nov 26, 2011 Want to see GanttProject in your native language? [Help us with translations!](#)". On the left, there's a large orange button with the text "Download GanttProject 2.6 Release build". Below this button is the text "... or ... stay with an older [GanttProject 2.5.5](#)". To the right, there's a video player window titled "Introduction into GanttProject 2.6" showing a screenshot of the software interface. The video player has a play button, a progress bar showing "00:00 / 15:04", and a "YouTube" link. Below the video player is the text "Watch [video tutorial](#) on YouTube". At the bottom, there's a paragraph about the software: "GanttProject is a cross-platform desktop tool for project scheduling and management. It runs on Windows, Linux and MacOSX, it is free and its code is opensource. What can it do?". Below this paragraph is a list of features:

- ❑ **Gantt chart.**
Create work breakdown structure, draw dependencies, define milestones.
- ❑ **Resources.**
Assign human resources to work on tasks, see their allocation on the Resource Load chart.
- ❑ **PERT chart.**
Generate PERT chart from Gantt chart

Slika 8.1 Prikaz sajta za preuzimanje GanttProject

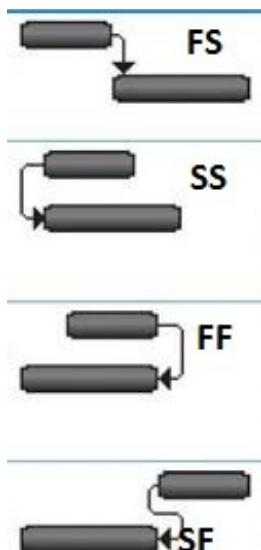
AKTIVNOSTI I VEZE U GANTOVOM DIJAGRAMU

Potrebno je odrediti redne brojeve, koja aktivnost će biti prva, a koja zadnja i veze koje postoje između aktivnosti

Na početku je potrebno napraviti listu taskova (aktivnosti). Nakon kreiranja WBS-a (**work-breakdown-structure**) dobije se lista aktivnosti koje treba da se realizuju da bi projekat bio uspešno završen. Listu aktivnosti kreira ceo projektni tim. Nakon toga potrebno je povezati (linkovati) aktivnosti. Kada imamo sve neophodne aktivnosti koje treba realizovati potrebno je uočiti neku vezu između njih, odnosno vremensku zavisnost. Potrebno je odrediti redne brojeve, koja aktivnost će biti prva, a koja zadnja i veze koje postoje između aktivnosti.

Postoje 4 tipa veza:

- **Finish to start (FS)**-kada se prva aktivnost završi, kreće se sa izvršenjem druge aktivnosti. Poglavlje knjige mora biti napisano da bi moglo da se pregleda i isprave greške.
- **Start to start (SS)**- obe aktivnosti počinju u isto vreme (štampanje knjige i pravljenje sajta na kome će biti predstavljena knjiga).
- **Finish to finish (FF)** - kraj prve aktivnosti određuje kraj druge aktivnosti. Korišćenje zakupljene opreme mora da se završi kada i period zakupa te iste opreme.
- **Start to finish (SF)** - start prethodne aktivnosti određuje završetak naredne aktivnosti. Ovo se vrlo retko koristi.



Slika 8.2 Prikaz veza u gantovom dijagramu

DEFINISANJE AKTIVNOSTI

Potrebno je da sve aktivnosti imaju definisan početak i kraj

Nakon pravljenja liste taskova, potrebno je odrediti početak i kraj aktivnosti. Povezivanje aktivnosti je korak ka pravljenju vremenske linije projekta. Potrebno je da se sve aktivnosti prikažu u vremenu. Potrebno je da sve aktivnosti imaju definisan početak i kraj. Ako je potrebno može se i ručno uneti početak i kraj. Svaka aktivnost u Gantovom dijagramu je predstavljena jednom linijom koja menja svoju dužinu u zavisnosti od trajanja same aktivnosti.

Kako bi se lakše pratilo projekat moraju se dodati i kontrolne tačke. Kontrolna tačka ili milestone je aktivnost čije je trajanje nula i to je takozvano mesto prodaje. Na Gantovom dijagramu je u obliku dijamanta. One predstavljaju kraj jedne faze ili završetak nekog velikog taska, serije taskova ili početak novog. Može se smatrati da je to mesto merenja odnosno provere da li je projekat na pravom putu.

Poslednji korak je dodati resurse. dobra stvar kod gantovog dijagrama je da će imena resursa stajati pored aktivnosti i imaće vizuelni prikaz odeljenih resursa. Radni dan ima 8 sati pa isti resurs ne može da radi paralelno dve aktivnosti koje su predviđene za isti dan i za koje je potreban rad od 8h u toku dana. Ovaj resurs je preopterećen i treba ga zameniti nekim drugim.

PRIMER: DEFINISANJE PROJEKTA

Kompanija WinMar Niš želi da kreira online prodavnicu koja bi trebala pružiti online prodaju računarske opreme. Primarni cilj je prelazak kompanije na online prodaju kako bi se povećala

Predviđeno vreme sledećeg pokaznog primera je 40 minuta.

Pre svega cemo definisati projekat:

Kompanija WinMar Niš želi da kreira online prodavnicu koja bi trebala pružiti online prodaju računarske opreme. Primarni cilj je prelazak kompanije na online prodaju kako bi se povećala prodaja računarske opreme.

Projekat je podeljen na šest osnovnih faza:

1. Projekt menadžment
2. Planiranje i dizajniranje
3. Razvoj
4. Test
5. Puštanje u rad

7. Završetak

Na početku ćemo pripremiti biznis plan i specifikaciju zahteva za softver kako bi tačno sa naručiocem projekta definisali ciljeve. Nakon toga ćemo vršiti istraživanje koja tehnologija je najbolja za izradu internet prodavnice kao i iščitavati dokumentaciju za tehnologiju za koju smo se odlučili. Nakon toga ćemo krenuti za dizajniranjem osmišljavanjem sajta. Zatim sledi razvoj koji obuhvata razvoj sajta, dizajn izgleda i realizacija izgleda sajta. Nakon razvoja naravno sledi testiranje.

FAZE PROJEKTA

Faze projekta su: projekt menadžment, planiranje, razvoj, testiranje, puštanje u rad i zatvaranje projekta

Projekt menadžment

1. Priprema biznis plana
2. Priprema specifikacije zahteva za softver

Planiranje i dizajniranje

1. Istraživanje
2. Čitanje dokumentacije tehnologija za izradu sajta

Razvoj

1. Dizajniranje (osmišljavanje) sajta
2. Razvoj sajta
3. Dizajn izgleda i realizacija izgleda (templejta) sajta

Testiranje

1. Testiranje sajta
2. Obuka zapoljenih

Puštanje u rad

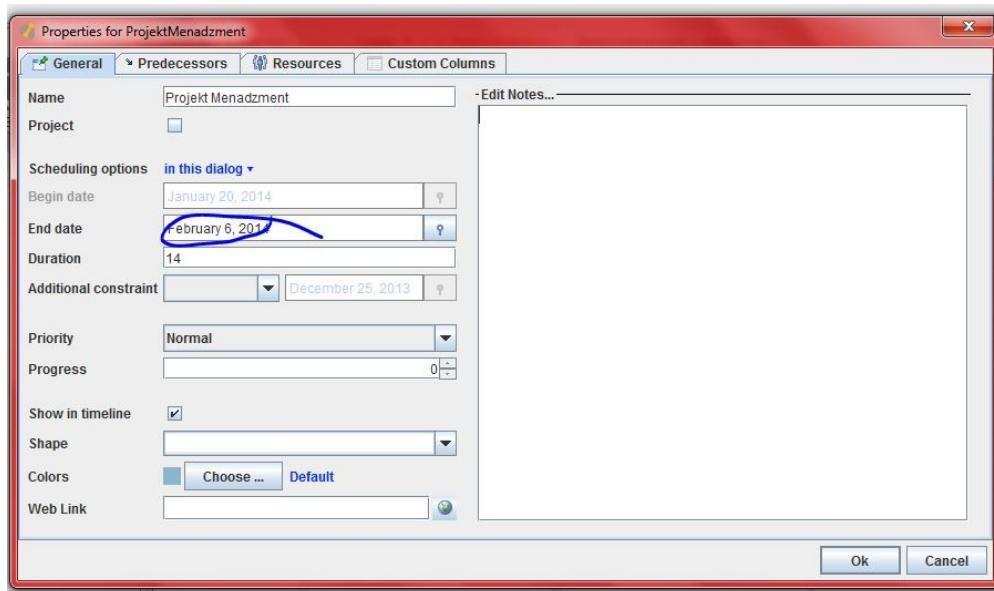
Zatvaranje projekta

KREIRANJE AKTIVNOSTI

Svaka aktivnost treba da ima neki vremenski okvir pa je potrebno definisati od kog do kog datuma se izrađuje svaka aktivnost.

Svaka aktivnost treba da ima neki vremenski okvir pa je potrebno definisati od kog do kog datuma se izrađuje svaka aktivnost. Da ne bismo ponovo definisali to ćemo definisati direkno u GanttProject programu.

1. Na početku ćemo formirati aktivnost projekt menadzment koji se sastoji iz subaktivnosti priprema biznis plana i priprema specifikacije za softver. Definisaćemo datum početka, i kraj aktivnosti. Resurse ćemo dodeliti kasnije kada budemo kreirali.

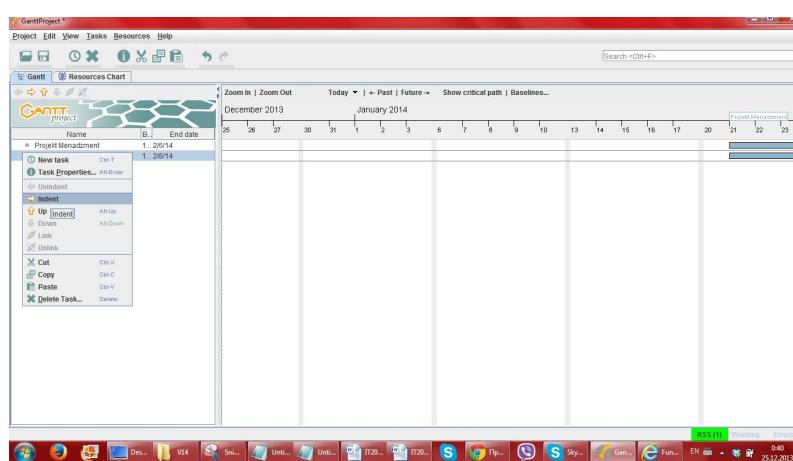


Slika 8.3 Kreiranje aktivnost ProjektMenadzment

DODELJIVANJE PODAKTIVNOSTI

Kako bi se dodelili subaktivnosti to se radi tako što se koristi Indent opcija

Kako bi se dodelili subaktivnosti to se radi tako što se koristi Indent opcija kao što je prikazano na sledećoj slici. Pre korišćenja te opcije potrebno je napraviti nova aktivnost i dodeliti joj se da postane subaktivnost.



Slika 8.4 Dodeljivanje podaktivnosti

Izgled subaktivnosti:

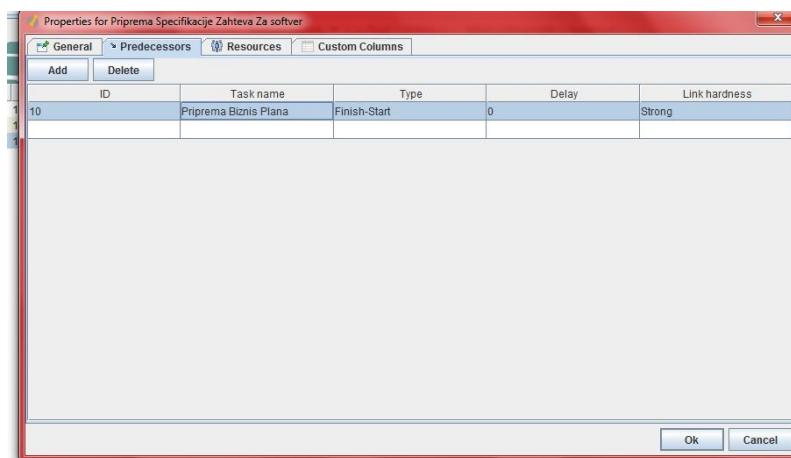
	Name	Begin date	End date
♀	Projekt Menadžment	12/25/13	1/27/14
♀	● Priprema Biznis Plana	1/21/14	1/27/14
♀	● Priprema Specifikacije Zaht...	12/25/13	12/25/13

Slika 8.5 Prikaz podaktivnosti

DEFINISANJE ZAVINOSTI

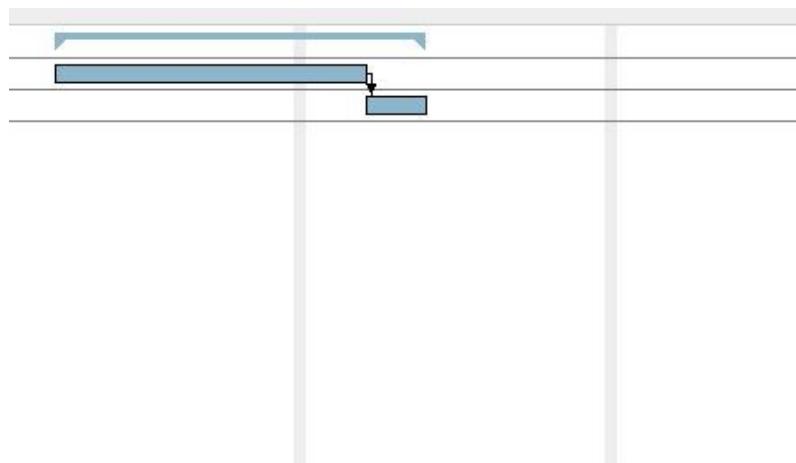
Da bi se definisala zavisnost izmedju aktivnosti potrebno je da prvoj aktivnosti dodamo predhodnika

Priprema Specifikacije Zahteva Za Softver aktivnost se izvršava odmah nakon završetka Priprema Biznis Plana. Da bi se definisala ta zavisnost potrebno je da prvoj aktivnosti dodamo predhodnika kao što je urađeno na sledećoj slici.



Slika 8.6 Dodavanje zavisnosti

Rezultat dijagrama unetih aktivnosti je prikazan na sledećoj slici:

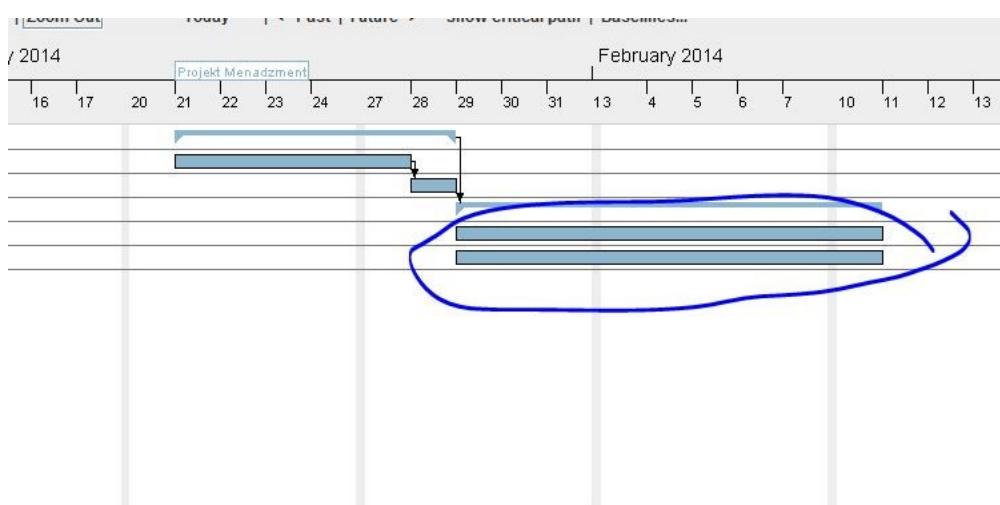


Slika 8.7 Prikaz generisanih dijagrama

PARALELNE AKTIVNOSTI

U sledećoj aktivnošći Planiranje i dizajniranje, subaktivnosti Istraživanje i Čitanje dokumentacije tehnologija za izradu sajta se izvršavaju u isto vreme

Nakon toga videli smo kako izgleda jedna aktivnost sa subaktivnostima i samim tim cemo kreirati na isti nacin ostale aktivnosti vodeći računa koje aktivnosti predhode kojim drugim aktivnostima. U sledećoj aktivnošći Planiranje i dizajniranje, subaktivnosti Istraživanje i Čitanje dokumentacije tehnologija za izradu sajta se izvršavaju u isto vreme. I to je prikazano na sledećoj slici:

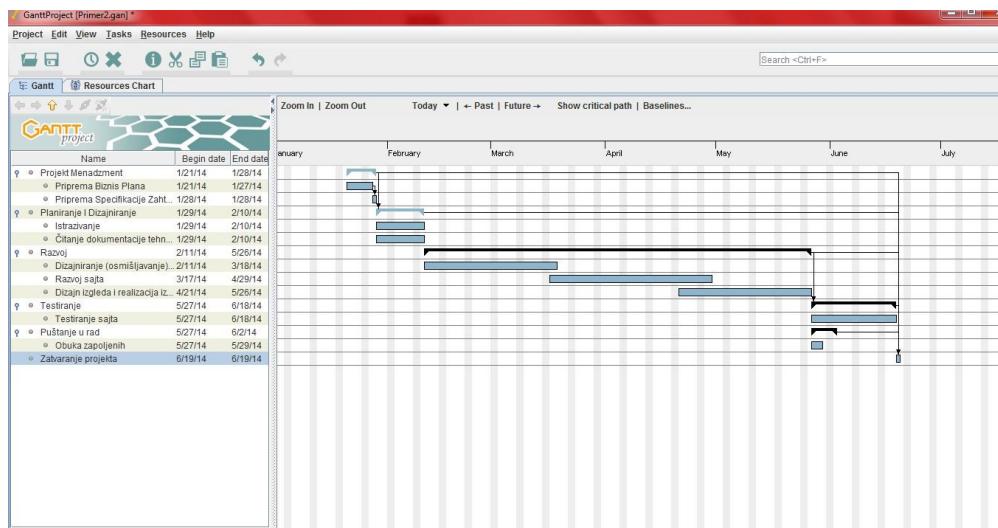


Slika 8.8 Prikaz paralelnih podaktivnosti

REZULTUJUĆI GANTOGRAM

Nakon iscrtavanja svih aktivnosti dobili smo sledeći gantogram

Nakon iscrtavanja svih aktivnosti dobili smo sledeći gantogram:

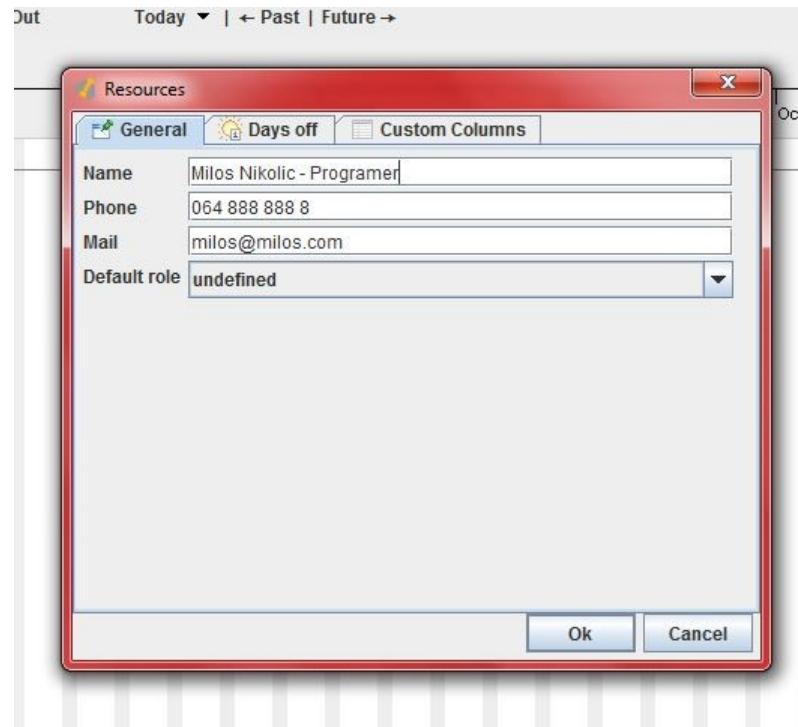


Slika 8.9 Prikaz gantograma

DEFINISANJE RESURSA

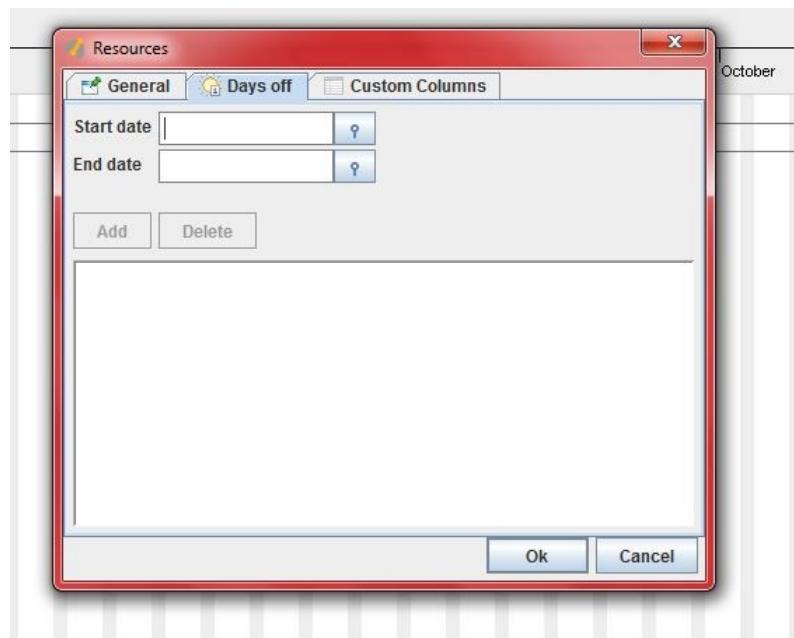
Za resurse se može definisati kojim danima i koliko rade

S obzirom da je ovo prost primer i da nam od resursa treba samo 2 programera dodaćemo te resurse i ubaciti ih posle u gantogram.



Slika 8.10 Definisanje resursa

Za resurse se može definisati i kojim danima rade i koliko rade i ta opcija je prikazana na sledećoj slici:

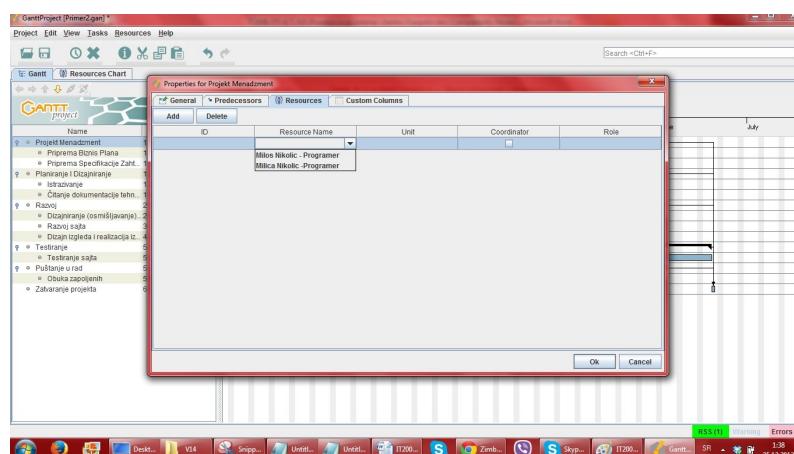


Slika 8.11 Dodeljivanje početka i kraja rada resursa

DODELA RESURSA

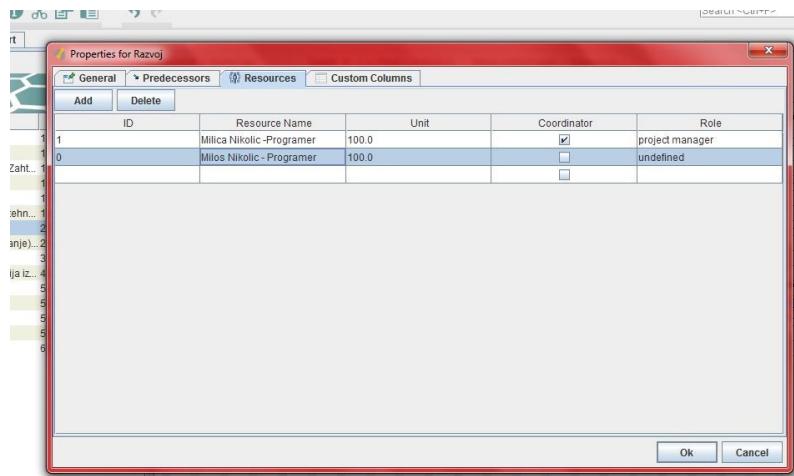
Kako se dodeljuju resursi je prikazano na sledećim slikama

Kako se dodeljuju resursi je prikazano na sledećim slikama:



Slika 8.12 Dodeljivanje resursa

Dodeljivanje više resursa:

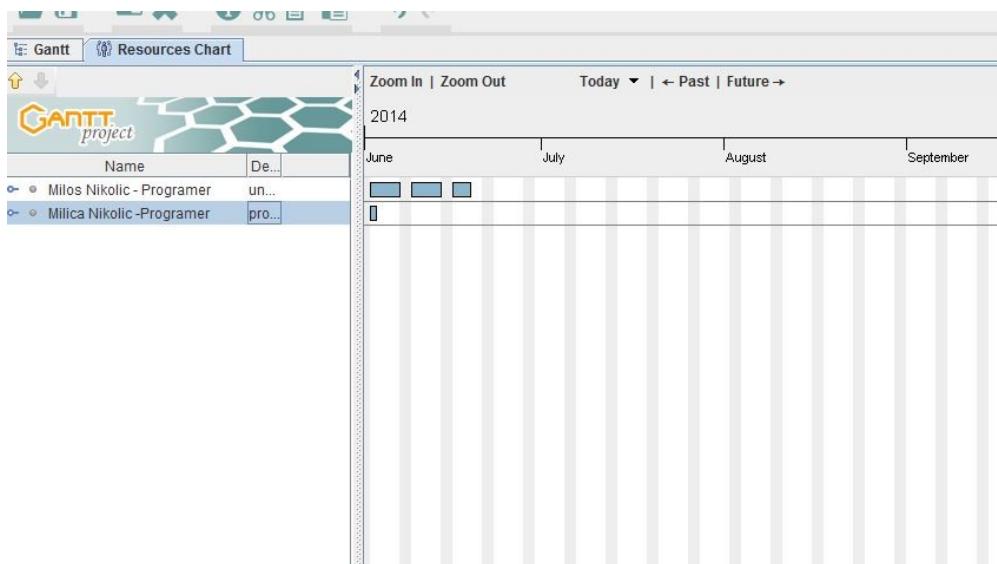


Slika 8.13 Prikaz dodeljenih više resursa

FORMIRANJE DATUMA RADA RESURSA

Kada smo resursima dodelili poslove, automatski se formirao datum rada resursa

Kada smo resursima dodelili poslove, automatski se formirao datum rada resursa i samim tim nismo morali u resursu to da definišemo:



Slika 8.14 Prikaz rada resursa

PRIKAZ JEDNOSTAVNOG PLANIRANJA PROJEKTA

Project Planning for Beginners

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 9

Zadatak za samostalni rad: Gantov dijagram

ZADATACI ZA SAMOSTALNI RAD

Gantov dijagram

Predviđeno vreme za izradu sledećeg zadatka je 30 minuta.

1. Napraviti Gantov dijagram za projekat razvoja informacionog sistema univerziteta. Dijagram treba da sadrži aktivnosti, podaktivnosti i resurse. (Vreme izrade: 10 minuta)
2. Napraviti Gantov dijagram za razvoj mobilne aplikacije koji mora da sadrži aktivnosti, podaktivnosti kao i resurse. (Vreme izrade: 10 minuta)
3. Napraviti Gantov dijagram koji se bazira na razvoju web sajta fakulteta. Dijagram treba da sadrži aktivnosti, podaktivnosti i resurse. (Vreme izrade: 10 minuta)

✓ Poglavlje 10

DOMAĆI ZADATAK

OPIS DOMAĆEG ZADATKA - DZ13

Napraviti plan projekta koji će sadržati cilj projekta, kompaniju za koju se projekat definiše i sam opis projekta

Očekivano vreme izrade zadatka: 35 minuta

Vaš zadatak za ovu nedelju je da osmislite informacioni sistem (tip informacionog sistema je vaš izbor). Za taj informacioni sistem:

1. Napraviti plan projekta koji će sadržati cilj projekta, kompaniju za koju se projekat definiše i sam opis projekta. Nakon toga opisati faze kroz koje mora da prođe kreiranje definisanog projekta. Sve to uraditi po uzoru na vežbu i sniti u word dokumentu.
2. Nakon definisanja plana projekta definisati gantov dijagram projekta u ganttProject programu. Gantsov dijagram treba sadržati:
 - Aktivnosti i subaktivnosti projekta
 - Međusobnu zavisnost aktivnosti i subaktivnosti (definisanjem predhodnika i sledbenika)
 - Svaka aktivnost i subaktivnost mora sadržati datum početka i datum završetka
 - Nakon definisanja aktivnosti definisati resurse sa kojima se raspolaže (programer, vođa projekta, dizajner itd...)
 - Dodeliti resurse aktivnostima
3. Gantsov dijagram snimiti i poslati uz word dokument predmetnom asistentu.

Zadatak dostaviti kao IT101-DZ13_Ime_Prezime_BrojIndeksa

▼ Zaključak

ZAKLJUČAK

Na ovom predavanju bilo je reči o informacionim sistemima i kako oni utiču na poslovanje. Bilo je reči o vrstama informacionog sistema, gde se moglo videti da njihova namena uslovjava razvijanje različitog tipa informacionog sistema, ali i dalje ima za glavni cilj da unapredi efikasnost poslovanja. Detaljno smo obradili faze u razvoju informacionog sistema, mada se te faze mogu koristiti u razvoju bilo kog IT projekta i nisu isključivo specifične samo za informacione sisteme.



IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

IT PROFESIONALIZAM

Lekcija 14

PRIRUČNIK ZA STUDENTE

IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

Lekcija 14

IT PROFESIONALIZAM

- ▼ IT PROFESIONALIZAM
- ▼ Poglavlje 1: Adaptibilnost
- ▼ Poglavlje 2: Potreba za kontinualnim usavršavanjem
- ▼ Poglavlje 3: Osnovna etička načela
- ▼ Poglavlje 4: Elementi sistema komuniciranja
- ▼ Poglavlje 5: Sinhrona i asinhrona komunikacija
- ▼ Poglavlje 6: Pokazna vežba: Obrada teksta u MS Wordu
- ▼ Poglavlje 7: Zadatak za samostalni rad: pravljenje šablona
- ▼ Poglavlje 8: Domaći zadatak
- ▼ ZAKLJUČAK

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

✓ Uvod

UVOD

Cilj ovog predavanja je da predstavi uticaj promena u društvu i promena na tržištu na razvoj IT

Na razvoj informacionih tehnologija utiče mnogo faktora. Cilj ovog predavanja je da predstavi uticaj promena u društvu i promena na tržištu na razvoj informacionih tehnologija. Takođe, kako tehnologije brzo napreduju, tako je neophodno da se ljudi, a naročito IT stručnjaci, kontinualno razvijaju i unapređuju tokom svog profesionalnog razvoja, te čem u ovom predavanju biti reči o **adaptibilnosti, etičkim načelima i važnosti komunikacije kod IT stručnjaka.**

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 1

Adaptibilnost

PODLOŽNOST IT-A PROMENAMA

Rad u oblasti IT je podložan promenama u društvu, na tržištu i novim tehnologijama, te uspešni IT profesionalac mora uvek da se prilagođava novim promenama i da bude adaptibilan

Kao i u drugim oblastima, rad u oblasti IT podložan je različitim uticajima. Problem je u tome što rad IT profesionalca nije izolovan proces, već je samo deo nekih drugih okolnih procesa višeg nivoa. Okolni procesi predstavljaju ili ulaz ili poremećaj koji deluje na proces rada IT profesionalca. Okolni procesi su promenljivi, pa su i njihovi uticaji na rad IT profesionalca promenljivi tokom vremena. Stoga, da bi jedan IT profesionalac opstao i bio uspešan neophodno je da bude **adaptabilan**, odnosno da se stalno prilagođava ovim uticajima koji nisu konstantni. Pomenuti uticaji bi u zavisnosti odakle potiču, mogli da se svrstaju na uticaje koji nastaju zbog:

- promena u društvu
- promena na tržištu
- promena tehnologija.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ 1.1 Uticaj promena u društvu na IT oblast

UTICAJ ZAHTEVA DRUŠTVA NA RAZVOJ INFORMACIONIH TEHNOLOGIJA

Tri osnovna zahteva društva koja imaju uticaj na razvoj tehnologija su dostupnost tehnologija, pristup podacima i zaštita podataka

Računari su danas prisutni u svim aktivnostima ljudi. Zbog toga svaka **promena u društvu**, u većoj ili manjoj meri, **utiče i na rad IT profesionalca**. Ovde će se, radi ilustracije, razmotriti samo neki uticaji.

Smanjenje cena računarske opreme, pa time i olakšana dostupnost svim IT proizvodima doveli su do toga da društvo, koje je tokom 19. i 20. veka prolazilo kroz tehnološku revoluciju,

sada prolazi kroz IT revoluciju. Veliki deo čovečanstva u svakodnevnom radu koristi računare ne samo za obavljanje poslova već i za zadovoljenje svojih svakodnevnih potreba. U nekim zemljama je broj stanovnika koji su „računarski pismeni“ narastao toliko da se govorи o e-društву (e-Society). Ovako veliki broj korisnika, kao i eksplozivan razvoj Interneta postavili su pred IT profesionalce nove zahteve kojima su morali da se prilagode. Navedimo neke zahteve društva koji su bitno uticali na rad IT profesionalaca:

- Dostupnost IT tehnologija svim članovima društva. IT profesionalci su morali da prilagode korisničke interfejse tako da programe mogu koristiti i starije osobe i ljudi sa posebnim zahtevima (slepi, gluvi, nepokretni itd.).
- Pristup podacima i aplikacijama sa bilo kog mesta. Mobilni korisnici računarskih tehnologija zahtevaju da imaju mogućnost pristupa podacima i aplikacijama sa bilo koje geografske lokacije kako bi mogli u svakom trenutku da obavljaju poslovne ili druge aktivnosti. Od IT profesionalaca se očekuje da razviju takve sisteme koje će korisnici moći da koriste ne samo na radnom mestu nego i kod kuće, u hotelu, u transportnim sredstvima ili na bilo kojoj drugoj lokaciji.
- Zaštita podataka. Povećanje broja računarski pismenih osoba dovelo je do pojave neovlašćenog pristupa podacima i drugih težih oblika računarskog kriminala. Stoga se od IT profesionalaca zahteva da razviju takve sisteme koji će garantovati integritet podataka i računarskih sistema uopšte.

UTICAJ DRUŠTVENIH PROMENA NA IT

Niska cena radne snage u Kini i Indiji dovela je do otpuštanja radnika u SAD kako bi kompanije smanjile troškove

Jedna od većih društvenih promena koja je uticala na kretanja u oblasti IT je otvaranje Kine koje je započelo krajem 20. veka. Pored toga, sve jače uključivanje Indije i nekih drugih dalekoistočnih zemalja u IT biznis izazvalo je ozbiljne poremećaje na tržištu IT radne snage zapadnih zemalja. Niska cena radne snage u ovim zemljama dovela je do preseljenja aktivnosti mnogih IT kompanija u ove zemlje, što je sa druge strane izazvalo otpuštanje ogromnog broja radnika uglavnom u SAD i zapadnoj Evropi. Naredni članci ilustruju ove promene i ozbiljnost njihovog uticaja na poslovanje u oblasti IT.

Američka kompanija Sun otpustila je 13.000 radnika tokom četiri godine kako bi smanjila troškove, istovremeno proširujući svoje aktivnosti u Bangalore, Pekingu, Peterburgu, Pragu i Indiji.

Sun otpustio 13.000 radnika

Sun is scaling back its Silicon Valley engineering group and expanding its facilities around the world to save on costs. The company said last week it would grow facilities in Bangalore, Beijing, St. Petersburg and Prague. Published reports said Sun isn't pulling out of SantaClara but won't be hiring much more there for the time being. The ability to quickly hire large numbers of programmers in India and other low-cost locations justified the company's plan to combine its worldwide research staff in those places, Sun says.

Sun recently said it would outsource its internal information technology team to Computer Sciences Corp. in a contract worth \$360 million. The company has reduced its staff to about 30,000, from about 43,000 four years ago.

*Network world, Maj 2005, In Brief: IBM set to slash 13,000 jobs,
<http://www.networkworld.com/news/2005/050905page6briefs.html>.*

OŠTRE KRITIKE IBM-U

Tokom svoje dugoročne prakse da otpušta ljudе iz svojih američkih predstavništva, IBM prilikom bilo kog većeg otpuštanja izaziva sumnju da prebacuje i dalje IT poslove van Amerike

Prebacivanjem poslova u druge zemlje radi jeftinije radne snage dovelo je oštih kritika. Tokom svoje dugoročne prakse da otpušta ljudе iz svojih američkih predstavništva, IBM prilikom bilo kog većeg otpuštanja izaziva sumnju da prebacuje i dalje IT poslove van Amerike, što dovodi do velikog nezadovoljstva stanovništva u periodu velikih kriza i recesija, poput recesije tokom 2009. i 2010. godine.

Od početka 2009. godine i tokom 2010. godine IT kompanije U SAD su otpustile oko 215.000 ljudi.

*Computerworld, Septembar 2010, IT slowly hiring back after layoffs,
<https://www.computerworld.com/article/2515698/it-slowly-hiring-back-after-layoffs.html>*

IBM otpušta

IBM's news that it will shed some 5,000 North American jobs and potentially send more positions overseas has stirred up some bad sentiment toward Big Blue as the economy continues to languish.

IT professionals and others sounded off online regarding IBM's plans (first reported in the Wall Street Journal) to reduce headcount in its Global Business Services division and possibly relocate jobs to lower-cost offshore geographies -- despite the practice being part of Big Blue's long-term strategy.

IBM has publicly stated it would grow its global presence and tap local resources and talent around the world, offshoring jobs overseas for years now. Yet this week's news that IBM would eliminate jobs during the U.S. recession sparked a notably negative reaction to what some industry watchers refer to as a solid business strategy.

Network world, Mart 2009, IBM Layoffs Incite Backlash, <http://www.networkworld.com/news/2009/032709-ibm-layoff-backlash.html>

OTPUŠTANJA U 2016-OJ I NJIHOV UTICAJ

Microsoft kills off another chunk of its smartphone activities

Pogledajmo prvo slučaj Microsoft-a i nastavak promena u poslovanju oko neuspelog poduhvata i ulaskom u industriju pametnih telefona? Do kakvih je promena ovde došlo i zbog čega? Šta sledi inženjerima koji su otpušteni i kakva je njihova dalja perspektiva?

Microsoft kills off another chunk of its smartphone activities

Down, but not out: Microsoft is laying off another 1,850 staff from its smartphone hardware business, but says it isn't leaving the market completely. What's left of the old Nokia business in Finland will be hardest hit by the latest round of lay-offs, with up to 1,350 jobs to go. Microsoft will cut up to 500 more globally, it said Wednesday. Since it bought Nokia's mobile phone activities in 2013, Microsoft has been managing a business in decline. The majority of the staff it acquired from Nokia are gone, and the company's mobile phone market share has stagnated.

Last week, the company sold off its feature-phone business, and hinted that while it would continue to update software for its Lumia smartphones, it would develop no new Lumia hardware. But no new Lumia hardware doesn't necessarily mean no new smartphones. Microsoft CEO Satya Nadella said the company will "continue to innovate across devices," an ambiguous phrase that could mean more hardware is on the way. The company is rumored to be working on a Surface Phone, a companion to its series of Surface tablets that could arrive next year. Nadella also promised innovation in "cloud services across all mobile platforms." But the company's phone efforts will be focused where it can differentiate its offering from the competition, he said. He highlighted security and manageability as two key areas where Microsoft has something to offer phone buyers, be they consumers or enterprises. Development will also continue on Continuum, he said. This can turn compatible high-end phones such as the Lumia 950 or 950XL into presenting tools or even computers by connecting them to a keyboard, mouse and external screen. Some say Continuum is a threat to Apple and Android phone makers; others are less enthusiastic. As a consequence of the latest lay-off plans, Microsoft will set aside **\$200 million** for severance fees and write down the value of its More Personal Computing business, resulting in a total charge of around \$950 million, it said. It expects to make most of the lay-offs by year-end, with the remainder going by mid-2017.

<http://www.networkworld.com/article/3074860/smartphones/microsoft-kills-off-another-chunk-of-its-smartphone-activities.html>

OTPUŠTANJA U 2020-OJ I NJIHOV UTICAJ

Pandemija koja nas je zadesila u 2020. godini, dovodi do velikog broja nezaposlenih

- "IBM will eliminate "several thousand jobs" as of May 22, mainly in the company's technology-services division. Cuts come a month after new CEO Arvind Krishna withdrew IBM's financial outlook amid economic uncertainty caused by the pandemic."

- "On April 28, online travel company TripAdvisor announced it was laying off more than 900 of its employees, amounting to a quarter of its workforce."

- "ZipRecruiter laid off 443 employees and furloughed dozens more on March 27, days after CEO Ian Siegel said the billion-dollar online job-hub company was safe."

- "ClassPass, the billion-dollar fitness platform, furloughed or laid off over half of its 700 employees on April 2 — 22% were laid off and 31% were furloughed."

<https://www.businessinsider.com/coronavirus-layoffs-furloughs-hospitality-service-travel-unemployment-2020?op=1>

▼ 1.2 Uticaj promena na tržištu na IT

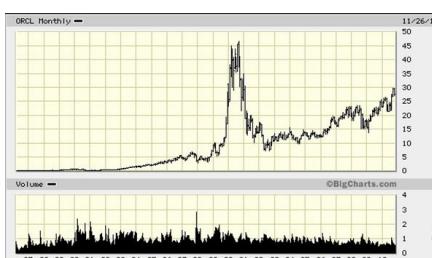
PROMENE NA TRŽIŠTU

Zakon ponude i potražnje u velikoj meri menja uslove poslovanja

Promene na tržištu takođe utiču na poslovanje IT profesionalaca. Zakon ponude i potražnje u velikoj meri menja uslove poslovanja. Kao primer kako promene na tržištu mogu preko noći da promene uslove poslovanja pogledajmo **čuvenu krizu nastalu kao posledicu .com buma**. Nagli razvoj Interneta i veb tehnologija omogućio je malim IT kompanijama koje su pružale usluge u ovoj sferi poslovanje sa velikim zaradama. Ovo je izazvalo veliko interesovanje finansijera koji su investirali velike sume novca u sve firme koje su se bavile informacionim tehnologijama. Vrednost deonica IT kompanija je vrtoglavu rasla, pre svega, zbog jagme za deonicama, a ne zbog realne vrednosti kompanija.

Kada se 2000. godine video da je vrednost deonica ovih kompanija daleko veća od stvarne vrednosti kompanija, došlo je do njihovog velikog pada i zatvaranja velikog broja malih kompanija. Probleme su imale i velike kompanije kao, na primer, Oracle. Na slici 1 je data promena vrednosti deonica ove kompanije u periodu od 1996. do 2005. godine. Jasno se vidi da je 1999. i 2000. godine vrednost deonica povećana četiri puta iako u istom periodu prodaja proizvoda ove kompanije nije imala taj trend. Ovo je dovelo do naglog pada vrednosti deonica krajem 2000. i u nekoliko narednih godina, što je bilo praćeno otpuštanjem velikog broja IT profesionalaca.

Slične probleme imao je i IBM čije su akcije između 1996. i 2000. godine porasle pet puta, da bi se 2002. posle mnogo oscilacija spustile na realni nivo (slika 2) .



Slika 1.1.1 Promena cena deonica kompanije Oracle



Slika 1.1.2 Promena cena deonica kompanije IBM



Slika 1.1.3 Promena cena deonica kompanije Microsoft

KONKURENTNOST NA TRŽIŠTU

Kompanije koje nisu dovoljno fleksibilne i efikasne prinuđene su da otpuštaju veliki broj radnika, uključujući i IT profesionalce

Stalni razvoj postojećih i pojava potpuno novih tehnologija utiču na sniženje cene proizvoda. Kompanije koje nisu dovoljno fleksibilne i efikasne prinuđene su da otpuštaju veliki broj radnika, uključujući i IT profesionalce. Sledeći članak pokazuje jedan ovakav slučaj.

Kompanije često pokušavaju da smanje troškove kako bi ostali konkurentni na tržištu. U sledećem primeru, članak ilustruje otpuštanja u 2012-oj godini kao i poteškoće sa kojima se kompanija susreće pokušavajući da ide u susret novim tehnologijama.

Biggest tech industry layoffs in 2020

Visualizing Layoffs at Prominent Startups Triggered by COVID-19

<https://www.visualcapitalist.com/layoffs-prominent-startups-covid-19/>

MICROSOFT ANNOUNCES 7,800 LAYOFFS, BLAMES NOKIA DEAL

Pogledajmo promene koje su usledile nakon sporazuma između Microsofta i Nokieu 2015-oj

Kako ova otpuštanja mogu uticati na dalji razvoj firme? Da li je postojao drugi način da se premoste gubici? Kako bi vi pokušali da rešite ovaj problem?

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

❖ 1.3 Uticaj promena tehnologija na IT

STALNA PROMENA TEHNOLOGIJA

Brzi razvoj tehnologija utiče na potrebu da IT profesionalci budu adaptibilni. Ono što je danas savremeno već kroz dve do tri godine postaje zastarelo.

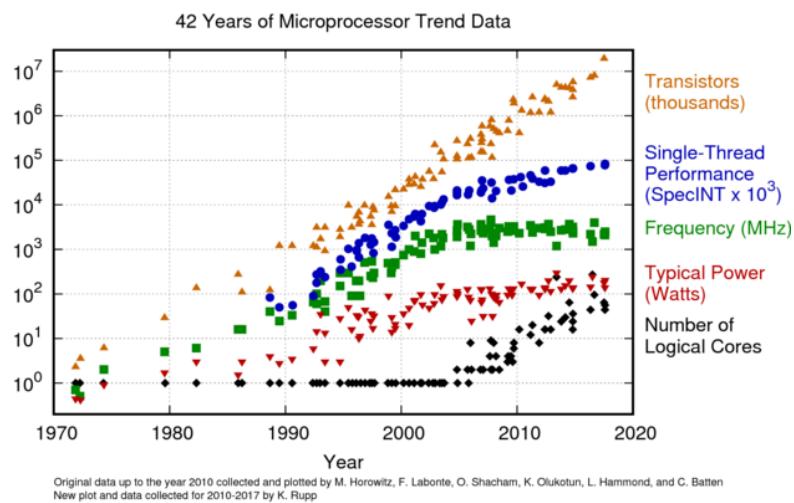
Još jedan razlog zbog koga IT profesionalci treba da budu adaptibilni je **stalna promena tehnologija**. Teško je naći drugu oblast poslovanja u kojoj se tolikom brzinom menjaju tehnologije. Ono što je danas savremeno već kroz dve do tri godine postaje zastarelo.

Brzina promena tehnologija u oblasti hardvera može se opisati Moor-ovim zakonom, koji kaže da se kapacitet računara povećava dva puta svakih 18 meseci. Na slici 1 je prikazana **promena broja tranzistora u mikroprocesorima** od 1971. pa do 2010. godine. Ovakav eksponencijalni rast uticao je na proširenje oblasti primene računara, a time i na povećanu potražnju IT profesionalaca. Međutim, promene u arhitekturi procesora zahtevale su od IT profesionalaca da se stalno usavršavaju.

Paralelno sa hardverom, **velike promene u računarstvu se događaju i u softveru**. Uvedene su mnoge nove programske paradigme, **a zajedno sa njima i novi programski jezici**. Umesto COBOL-a, koji je bio dominantan 60-ih i 70-ih godina prošlog veka, prešlo se na relacione baze podataka i SQL. Umesto proceduralnih jezika, danas se pretežno koriste objektno-orientisani jezici kao što su C++ i Java. **Event-driven paradigma** je postala neizbežna zbog prelaska na grafički korisnički interfejs. Programeri koji nisu prihvatili ove novine ostali su na margini IT razvoja.

Razvoj Interneta i veba doveo je do potrebe uvođenja novih arhitektura aplikacija. Tako se od **klijent-server arhitekture** prešlo na **višeslojnu arhitekturu**, jer je to pre svega ubrzavalo razvoj aplikacija.

Kombinacijom Interneta i bežičnih komunikacija došlo se do potpuno novih mogućnosti za korišćenje računara. Konvergencija računarstva i komunikacionih tehnologija, a pre svega mobilne telefonije, dovela je do pojave novih vrsta uređaja čija je namena istovremeno i obrada podataka i komunikacija.



Slika 1.2.1 Promena broja tranzistora u mikroprocesorima od 1970. pa do 2020. godine

. "42 Years of Microprocessor Trend Data" - Karl Rupp

▼ Poglavlje 2

Potreba za kontinualnim usavršavanjem

POTREBA ZA PROFESIONALNIM RAZVOJEM

IT profesionalac ima potrebu za kontinualnim sticanjem opštih i teoretskih znanja, kao i specijalističkih znanja i veština

IT profesionalac mora da bude adaptabilan i da promptno reaguje na promene bilo da su one nastale u društvu, na tržištu ili su tehnološke prirode. IT profesionalci pružaju pre svega intelektualne usluge, bez obzira da li se radi o uslugama u pravom smislu reči ili se radi o proizvodu koji je zasnovan na intelektualnom radu. Praksa pokazuje da je ova vrsta robe „kvarljiva“ i da joj rok upotrebe ističe uvek kad se pojave nove ili poboljšane tehnologije. Klijenti po pravilu traže najsavremeniju tehnologiju tako da IT profesionalci koji ne prate trend ubrzo postaju „IT dinosauri“. Dobar IT profesionalac zato mora uvek da uskladi svoje znanje i veštine sa zahtevima tržišta.

Jedini način da jedan IT profesionalac opstane u poslu je da prihvati princip „lifelong learning-a“, odnosno princip da čitavog radnog veka mora da dopunjuje i proširuje svoje znanje i veštine.

Može se reći da IT profesionalac ima potrebu za sticanjem:

- **opštih znanja** (kao što su matematika, jezik, komunikologija)
- **teoretskih znanja** (odgovor na pitanje zašto)
- **specijalističkog znanja i veština** (odgovor na pitanje kako da se uradi neki posao).

Nemoguće je sva ova znanja dobiti odjednom, pogotovo što se, kao što se videlo, javljaju nove tehnologije, ali dobar IT profesionalac treba da ima svoj plan profesionalnog razvoja.

Sticanje opštih i teoretskih znanja u principu donosi najmanje novca, ali je sa dugoročnom vrednošću i predstavlja osnovu za specijalističku nadgradnju. Ova znanja su mnogo manje podložna brzim tehnološkim promenama. Za sticanje opštih i teoretskih znanja mogu se koristiti sledeći instrumenti:

- doktorske studije
- diplomske akademske studije (master)
- osnovne akademske studije
- naučni časopisi i stručna literatura
- kongresi, simpozijumi
- neke vrste kurseva (na primer, strani jezik).

SPECIJALISTIČKA ZNANJA

Pri sticanju specijalističkih znanja je važno izvršiti predviđanje koje će tehnologije biti konkurentne, kako bi se na vreme ovladalo njima

Sticanje specijalističkog znanja i veština ima kratkoročnu vrednost, najčešće samo nekoliko godina, ali se ovakvo znanje uobičajeno jako dobro plaća. Za sticanje ovog vrsta znanja se koriste sledeći instrumenti:

- specijalizacije
- sertifikovani kursevi (Microsoft, Cisco, dr.)
- seminari, prezentacije, sajmovi
- e-biblioteke, specijalistički forumi
- vodeći IT sajtovi
- stručni časopisi

Ono što je važno pri sticanju specijalističkih znanja jeste izvršiti predviđanje koje će tehnologije biti konkurentne, kako bi se na vreme ovladalo njima. Zbog toga **IT profesionalac treba stalno da prati kretanja u IT svetu kako bi se na vreme adaptirao za predstojeće promene.**

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

MASOVNA OTPUŠTANJA U IT SEKTORU U 2017-OJ GODINI

Kako automatizacija dovodi do poboljšanja poslovanja, tako je došlo do promene na tržištu rada u Indiji, gde je sve manja potreba za nisko kvalifikovanom IT radnom snagom

Indijske IT kompanije zarađuju oko 155 milijardi dolara godišnje, samo od prihoda koje imaju iz SAD. Međutim, kako automatizacija dovodi do poboljšanja poslovanja, tako je došlo do promene na tržištu rada u Indiji, gde je sve manja potreba za nisko kvalifikovanom IT radnom snagom. Neke od tehnologija koje su unapredile rad firmi, ali na uštrb radnih mesta su 3D štampa i Internet of Things (IoT). Pod takvim sticajem okolnosti, gde Indija proizvodi oko 1,5 milion inženjera (ne samo IT), samo oko 500.000 pronađi posao, a ostali ostaju nezaposleni. Ovo ne ukazuje samo da postoji razvoj tržišta i automatizacija koja je dovela do rashoda u proizvodnji i potražnji radne snage, već i da se inženjeri ne usmeravaju sa veštinama i znanjima koje su potrebne modernom radnom IT okruženju, koje brzo i energično napreduje.

Stoga je neophodno da se IT inženjer prvo bitno priprema sa visoko naprednim i ne-specifično usmerenim znanjima. Na primer, učenje samo za jedan posao, kao što se vidi u ovom slučaju Indije, može biti mač sa dve oštice. Dok se sa jedne strane čovek može spremiti pohađanjem jednog kursa za specifičnu veštinu ili znanje, isto tako on postaje manje koristan, kada ta

tehnologija zastari i firma se odluči da pređe na nove tehnologije koje će im unaprediti obrazovanje i doneti više prihoda.

U Indiji se očekuje da 56 000 inženjera izgubi posao u 2017-oj godini. Ovo i ne bi bio značajan broj, ako se uzme u obzir broj ljudi koji živi i radi u Indiji, ali predviđena za naredne godine su alarmantna. Očekuje se da 200 000 inženjera izgubi posao godišnje u naredne 3 godine. Pogledajte više o masovnom otpuštanju u Indiji na sledećem linku

<http://www.hindustantimes.com/education/layoffs-and-shrinking-job-market-is-this-the-end-of-india-s-engineering-dream/story-uWtw0E8PtsINzsfiXszMpL.html>

▼ Poglavlje 3

Osnovna etička načela

OSNOVNA I PROFESIONALNA ETIČKA NAČELA

Ogromna količina informacionih resursa može postati predmet zloupotrebe, čime se mogu naneti nepopravljive materijalne i nematerijalne štete pojedincima, institucijama i društvu

Posao IT profesionalca, ma koliko bio lep, povlači za sobom i ogromnu odgovornost prema ljudima u radnom okruženju i prema široj društvenoj zajednici. Ogromna količina informacionih resursa kojima upravljaju IT profesionalci može postati predmet zloupotrebe, čime se mogu naneti nepopravljive materijalne i nematerijalne štete pojedincima, institucijama i društvu. Zbog toga IT profesionalac treba da se pridržava kako osnovnih tako i profesionalnih etičkih načela. Osnovna etička načela se odnose na sve ljudе. Profesionalna etika se odnosi na pitanja vezana za profesiju. Ne može se reći da se neko drži etičkih načela ako poštuje samo jednu od ove dve vrste načela. Društvo ili neke interesne zajednice propisuju etička načela radi opšte dobrobiti. Zbirka etičkih načela se naziva etički kod.

OPŠTA ETIČKA NAČELA

Ovde se navode samo neka najvažnija opšta etička načela i njihov veza sa IT profesionalizmom

Osnovna etička načela se vezuju za celo društvo. U različitim kulturama postoje razlike u etičkim načelima, ali su one zanemarljive. Ovde se navode samo neka najvažnija opšta etička načela, ali je istaknuto kako se ona odnose na IT profesionalce:

1. Doprinosite društvu i ljudskom blagostanju

- poboljšanje kvaliteta života
- zaštita ljudskih prava
- respektovanje kulturnih razlika
- minimiziranje uticaja IT tehnologija na zdravlje i bezbednost ljudi
- minimiziranje negativnih uticaja IT tehnologija na okolinu

2. Izbegavajte da povredite druge

- neželjeni gubitak informacija
- slučajno ili namerno oštećenje ili menjanje informacija
- gubitak ili povreda vlasništva

3. Budite iskreni i poverljivi

- sprečavanje konflikta interesa
- prijavljivanje svih problema i ograničenja informacionog sistema
- dužnost da se da prava slika o ličnoj kvalifikaciji

4. Budite pravični i odbacite bilo kakvu diskriminaciju

- jednakost, tolerancija i poštovanje drugih
- bez diskriminacije na bazi rase, pola, religije, godina, invaliditeta, nacije ili drugih sličnih faktora
- onemogućavanje neautorizovanog pristupa resursima

5. Poštovanje vlasničkih prava, uključujući i autorska i patentna prava

- zaštita autorskih prava, poslovnih tajni, uslova iz licencnih ugovora
- sprečavanje neovlašćenog kopiranja softvera

6. Poštovanje intelektualne svojine

- zaštita intelektualne svojine drugih
- bez krađa tuđih ideja

7. Poštovanje privatnosti drugih

- održavanje privatnosti i integriteta i tačnosti podataka osoba
- preduzimanje mera za zaštitu podataka
- dozvola osobama da imaju uvid u sopstvene podatke
- prikupljanje i čuvanje samo neophodnog minimuma podataka o osobama

8. Poštovanje poverenja

- poštovanje poverenja koje su vam ukazali zaposleni, klijenti i korisnici.

✓ 3.1 Profesionalna načela

PROFESIONALNA ETIČKA NAČELA IT PROFESIONALCA

Poštovanje profesionalnih načela obezbeđuje da IT profesionalac obavlja svoju funkciju odgovorno

Profesionalna etička načela treba da definišu ideale i odgovornost IT profesionalca. Ona na izvestan način imaju regulatorni efekt jer štite i klijente i profesionalce. **Poštovanje profesionalnih načela obezbeđuje da IT profesionalac obavlja svoju funkciju odgovorno.**

IT profesionalac treba da poštuje sledeća profesionalna etička načela:

- Težnja ka najvišem kvalitetu proizvoda i usluga i ka efektivnosti u procesu profesionalnog rada
- Postizanje i održavanje profesionalnih sposobnosti
- Poznavanje i poštovanje postojećih zakona vezanih za profesionalni rad
- Prihvati profesionalnu ocenu rada i profesionalno ocenjivati druge
- Davati iscrpne i potpune ocene IT sistema i njegovog uticaja, uključujući i analizu rizika
- Časno ispunjavati obaveze iz ugovora i sporazuma i biti odgovoran
- Edukovati društvo o računarskim tehnologijama i njihovom uticaju na društvo
- Koristiti računarsko-komunikacione resurse samo autorizovano

PRIMER ETIČKOG KODA

Kao primer etičkog koda IT profesionalaca ovde se navodi IEEE etički kod

Mi, članovi IEEE, prepoznajući značaj naših tehnologija na kvalitet života celog sveta i prihvatajući profesionalnu obavezu naše profesije, njenih članova i zajednice kojoj služimo, ovim se obavezujemo na najviše etičko i profesionalno ponašanje i slažemo se:

1. da prihvatimo odgovornost da donosimo inženjerske odluke u skladu sa bezbednošću, zdravljem i blagostanjem ljudi i odmah otkrijemo faktore koji mogu ugroziti ljude ili okruženje
2. da izbegavamo realne i moguće konflikte interesa uvek kada je moguće i da ih obelodanimo zainteresovanim stranama kada stvarno postoje
3. da budemo iskreni i realni pri tvrđenju i procenama na osnovu raspoloživih podataka
4. da odbacimo korupciju u svim njenim formama
5. da poboljšamo razumevanje tehnologije, njenu primenu i potencijalne posledice
6. da održavamo i poboljšavamo našu tehničku sposobnost i da prihvatimo tehnološke poslove za drugu stranu samo ako smo kvalifikovano obučeni ili imamo iskustva ili nakon obznanjivanja naših ograničenja u vezi sa tim posлом
7. da tražimo, prihvatamo i pružamo iskrenu kritiku tehničkog rada, da obznamo i ispravimo greške i da poštено cenimo doprinos drugih
8. da se nepristrasno tretiraju sve osobe bez obzira na faktore kao što su rasa, religija, pol, invaliditet, godine ili nacionalno poreklo
9. da izbegnemo povređivanje drugih osoba, njihovo vlasništvo, reputaciju ili posao pogrešnim ili zlonamernim radnjama
10. da pomognemo kolegama i saradnicima u njihovom profesionalnom razvoju i da im damo podršku u prihvatanju ovog etičkog koda.

Odobreno od strane IEEE Board of Directors

Avgust 1990.

▼ Poglavlje 4

Elementi sistema komuniciranja

KOMUNIKACIJE

Komunikacija se može definisati kao proces razmene poruka između dve ili više osoba

IT profesionalac u svakodnevnom radu ima potrebe za **jednosmernom** i **dvosmernom komunikacijom** sa svojim okruženjem ili udaljenim partnerima. Da bi profesionalno obavljao **posao IT profesionalac treba da ima potrebna znanja i veštine za efikasnu komunikaciju.**

Komuniciranje u najopštijem smislu predstavlja sporazumevanje između dva entiteta. U užem smislu, kada se misli na ljude, komunikacija je proces razmene poruka među ljudima. Ovde će se govoriti samo o komunikaciji između ljudi.

Dakle, komunikacija se može definisati kao proces razmene poruka između dve ili više osoba. Svaka komunikacija ima cilj koji ne mora obavezno da bude ispunjen u procesu komunikacije. Komunikacija se zasniva na razmeni poruka čije značenje treba da bude poznato svim stranama u komunikaciji, jer u suprotnom ako poruke nisu razumljive, ne može biti ostvaren cilj komunikacije.

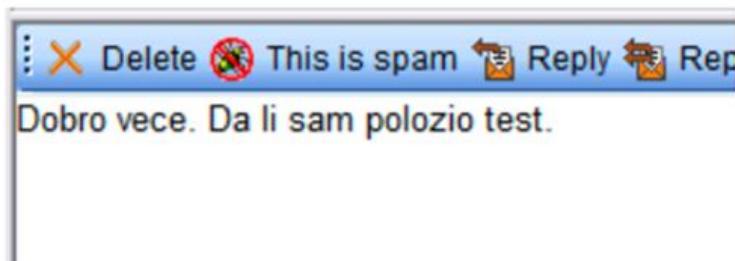
Komunikacija može biti **verbalna** ili **neverbalna**. **Verbalne komunikacije** su **zasnovane na rečima ili simbolima**. **Neverbalne komunikacije** su **zasnovane na gestikulaciji, dodiru, fizičkoj pojavi, pogledima, klimanju glavom itd.**

Komunikacija se uvek odigrava u nekom kontekstu komunikacije koji je definisan mnogim faktorima kao što su: **cilj komunikacije, broj učesnika, karakteristike učesnika, uloge učesnika, prethodna komunikacija između učesnika** itd.

PRIMER

Analiza email-a koji je student poslao profesoru

Da bi se shvatile definicije komunikacije i da bi se nazreti problemi koji se javljaju u komunikaciji upotrebićemo jedan primer iz realnog života. Sledi tekst kompletne poruke prikazane na slici 1, u neizmenjenom obliku, koju je jedan student poslao svom profesoru. **Radi se o e-mail-u, tako da je poznato ime i prezime pošiljaoca.** Ovaj primer treba da ilustruje da se u komunikaciji treba držati pravila komunikacije ako želimo da postignemo njen cilj.



Slika 4.1 Poruka studenta profesoru

U ovom slučaju reč je o **verbalnoj komunikaciji** između **dve osobe**: studenta, koji šalje poruku, i profesora koji je prima. **Kontekst komunikacije** je određen činjenicom da **je poruka dobijena sa servera metropolitan.ac.rs**, pa profesor može da prepostavi da je u pitanju student, tako da su **uloge samo delimično poznate**.

Istina je da je profesor primio poruku, ali je to možda bila i greška studenta. **Iz poruke se ne vidi kome se student obraća**. Umesto toga stoji samo „dobro veče“ iako će profesor možda pročitati poruku u jutarnjim satima. Da je student umesto toga napisao „Poštovani profesore Petkoviću“, profesor bi bio siguran da je poruka upućena njemu. Dalje, profesor **iz zaglavlja može da vidi tačno ime i prezime studenta, ali ne i na kojoj je godini**. Na osnovu imena i prezimena profesor je mogao da proveri da li je bilo prethodne komunikacije, i u ovom slučaju je nije bilo.

Razmotrimo sada **cilj ove komunikacije**. Student je očigledno želeo da sazna da li je položio test iz nekog predmeta. Pošto jedan profesor uobičajeno drži više predmeta i pošto se testovi daju svake nedelje, **profesor iz studentove poruke ne vidi o kom je predmetu i testu reč**. Razume se, profesor se može raspitati kod fakultetskih službi koje predmete sluša ovaj student, ali problem o kom se testu radi ostaje **nerešen**. **Dakle, profesor verovatno neće moći da odgovori studentu, pa tako ni cilj komunikacije neće biti ostvaren**.

Na kraju, profesor iz poruke može da zaključi da je reč o studentu koji ne poznaje ne samo principe komunikacije nego i gramatiku. **Reči „vece“ i „polozio“ su nedozvoljene u poslovnoj komunikaciji**. Na kraju druge rečenice nedostaje znak „?“.

ANALIZA PRIMERA - VIDEO

Analiza email-a koji je student poslao profesoru - video

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

FAKTORI KOJI UTIČU NA KOMUNIKACIJU

Na komuniciranje utiču: običajne norme, socijalna iskustva, standardi ponašanja, jezičke forme izražavanja, umetničke forme izražavanja, naučne forme izražavanja, dr.

Komunikacije su po prirodi vrlo složen fenomen jer na komuniciranje utiče čitav niz faktora. U nekim slučajevima strane koje komuniciraju nisu svesne postojanja ovih uticaja, pa dolazi do nerazumevanja.

Između ostalog, na komuniciranje utiču [3]:

- običajne norme
- socijalna iskustva
- standardi ponašanja
- jezičke forme izražavanja
- umetničke forme izražavanja
- naučne forme izražavanja
- mitološki i religiozno-ritualni simboli i drugo.

ELEMENTI SISTEMA KOMUNICIRANJA

Sistem komuniciranja se sastoji od strana u komuniciranju i instrumenata komunikacije.

Da bi se komunikacija uspešno obavila potrebno je dobro poznavanje svih elemenata sistema komuniciranja. **Sistem komuniciranja se sastoji od strana u komuniciranju i instrumenata komunikacije.**

Strane u komuniciranju su:

- **pošiljalac poruke**
- **primalac poruke.**

Instrumenti komunikacije su:

- **poruka**
- **komunikacioni kanal (medij).**

Sam proces komunikacije je vrlo složen, pa se radi njegovog razumevanja može podeliti na potprocese

Podprocesi u komunikaciji su:

- **Definisanje namere.** Pošiljalac definiše nameru da pošalje neku poruku sa određenim ciljem određenim primaocima.
- **Kodiranje poruke.** Pošiljalac kodira poruku i prilagođava je primaocima i vrsti komunikacionog kanala koji će koristiti. Veliki problemi u komunikaciji nastaju zbog toga što pošiljalac ne izvrši pravilnu transformaciju zamišljene poruke u kodirani oblik poruke. Da bi se uverio da je dobro izvršio kodiranje poruke, pošiljalac treba da izvrši decentralizaciju. Decentralizacija je postupak stavljanja pošiljaoca u poziciju primaoca. Pošiljalac treba da razmisli o tome da li će u kontekstu u kome se šalje poruka ona biti razumljiva primaocu.

- **Upućivanje poruke.** Poruka se preko nekog komunikacionog kanala (govor, pismo, poruka na forumu itd.) šalje primaocima. Ovde se mogu javiti problemi sa kvalitetom komunikacionog kanala, odnosno sa komunikacionim šumom. Na primer, pošiljalac priča nerazgovetno, primalac je dekoncentrisan, e-mail server ne radi, telefonska veza je slaba, u prostoriji je buka itd.
- **Prijem i tumačenje poruke.** Primalac prima i dešifruje poruku. I pri dešifrovanju mogu da nastanu problemi u komunikaciji. Primalac nekada ne može da protumači ili samo delimično može da protumači poruku.
- **Povratna sprega.** Komunikacija ne može biti uspešna ako se pošiljalac ne uveri da je primalac dobro protumačio poruku. Zavisno od vrste poslate poruke pošiljalac se može uveriti u ispravnost prijema poruke na osnovu odgovora ili druge akcije primaoca.

▼ Poglavlje 5

Sinhrona i asinhrona komunikacija

SINHRONE KOMUNIKACIJE

Sinhrone komunikacije se obavljaju između dva entiteta u realnom vremenu

Sinhrone komunikacije se obavljaju između dva entiteta u realnom vremenu. Svaki entitet je u mogućnosti da prima i istovremeno šalje poruke. Pre pojave pisma i pismenosti među ljudima sva komunikacija je bila sinhrona, odnosno ljudi su razgovarali jedni s drugima licem u lice. Sa otkrićem telefona postalo je moguće vršiti sinhronu komunikaciju i sa udaljenim sagovornikom. Korišćenjem Interneta danas se obavlja sinhrona komunikacija ili samo prenosom glasa ili prenosom glasa i slike. Pri tom se koriste tehnologije kao što su: chat, voice over IP, deljenje aplikacija i video conferencing.

Nedostatak sinhrone komunikacije je u tome što obe strane moraju biti raspoložive za komunikaciju u isto vreme. Ovo je u nekim slučajevima teško izvodljivo pogotovo ako u komunikaciji učestvuju više od dve osobe. Razlozi za ovo mogu da budu: trenutne obaveze sagovornika, velika razlika u vremenskim zonama, slaba propusna moć komunikacionih kanala i drugo.

Sinhrone metode komunikacije treba koristiti samo onda kada je potrebno dobiti trenutni odgovor i kada su obe strane zainteresovane za takav oblik komunikacije. Treba imati na umu da je sinhrona komunikacija napadni metod koji možda u tom trenutku nije po volji drugoj strani.

ASINHRONE KOMUNIKACIJE

Asinhrone komunikacije se ne obavljaju u realnom vremenu i ne zahtevaju da obe strane budu istovremeno raspoložive za komunikaciju

Asinhrone komunikacije se ne obavljaju u realnom vremenu i ne zahtevaju da obe strane budu istovremeno raspoložive za komunikaciju. Strana koja inicira komunikaciju može da odabere vreme kada će poslati poruku. Druga strana može takođe da bira vreme kada će primiti poruku i da li će odgovoriti na nju, ako se radi o dvosmernoj komunikaciji. Istorijски gledano, prvi mehanizam asinhronе komunikacije su bila pisma. Danas se koriste i druga sredstva kao što su: telegram, faks, elektronska pošta, veb forumi, veb stranice i slično.

Asinhronne komunikacije, u poslovnom okruženju, u mnogim slučajevima predstavljaju bolje rešenje. One obema stranama ostavljaju više vremena da pripreme efikasnu poruku zasnovanu na faktima, što je u nekim slučajevima u realnom vremenu nemoguće. Pored toga, asinhronne komunikacije omogućuju sagovornicima da usklade vreme komunikacije sa svojim radnim obavezama.

JEDNOSMERNE I DVOSMERNE KOMUNIKACIJE

U jednosmernim komunikacijama jedna strana šalje poruku, a druga može samo da je primi. U dvosmernim komunikacijama obe strane imaju mogućnost da razmenjuju poruke.

Zavisno od učešća strana koje komuniciraju, komunikacije mogu biti jednosmerne ili dvosmerne. **U jednosmernim komunikacijama** jedna strana šalje poruku, a druga može samo da je primi. Tipičan primer za ovo su veb, oglasni pano, radio i televizija. **U dvosmernim komunikacijama** obe strane imaju mogućnost da razmenjuju poruke.

Zavisno od broja učesnika u procesu komunikacije, komunikacije mogu biti između:

- dve osobe
- osobe i grupe
- između dve grupe ili
- kao masovna difuzija informacija (radio, TV, veb).

▼ Poglavlje 6

Pokazna vežba: Obrada teksta u MS Wordu

OBRADA TEKSTA

Obrada teksta je jedna od osnovnih funkcija savremene primene računara.

Predviđeno vreme pokazne vežbe je 45 minuta.

Obrada teksta je jedna od osnovnih funkcija savremene primene računara. Čak i u slučajevima kada korisnik ne koristi neki od specijalizovanih programa kao što su Microsoft Word ili OpenOffice.org Writer, korisnik se bavi obradom teksta. Neki od primera su pisanje e-mail poruka ili unos podataka u odgovarajuću bazu.

Osnovne funkcije su slične kod većine programa koji dozvoljavaju unos teksta, manipulaciju njime i snimanje, tako da korisnici koji poznaju principe rada sa jednim programom, lako mogu po analogiji da koriste neki drugi program. U okviru ovih vežbi pokazaćemo osnovne principe obrade teksta u MS Word i OpenOffice.org Writer programima. Često se za programe za obradu teksta koristi i termin tekst procesor, zbog engleskog termina – **word processor**.

MS Word je program za obradu teksta koji je dugi niz godina važio za osnovni program iz paketa Office. Učešće ovog programa na tržištu programa za obradu teksta je dominantno, pa se format ovih dokumenata (**.doc**) u većini slučajeva smatra standardom.

OpenOffice.org (često se koristi i skraćenica Ooo ili OO.o) Writer je program za obradu teksta koji pruža funkcije i alate koji se mogu uporediti sa opcijama MS Word. Osim toga, postoji mogućnost eksportovanja u PDF format (engl. **Portable Document Format**) bez dodatnog softvera, WYSIWYG editor za kreiranje i prikazivanje Web strana itd.

MS WORD

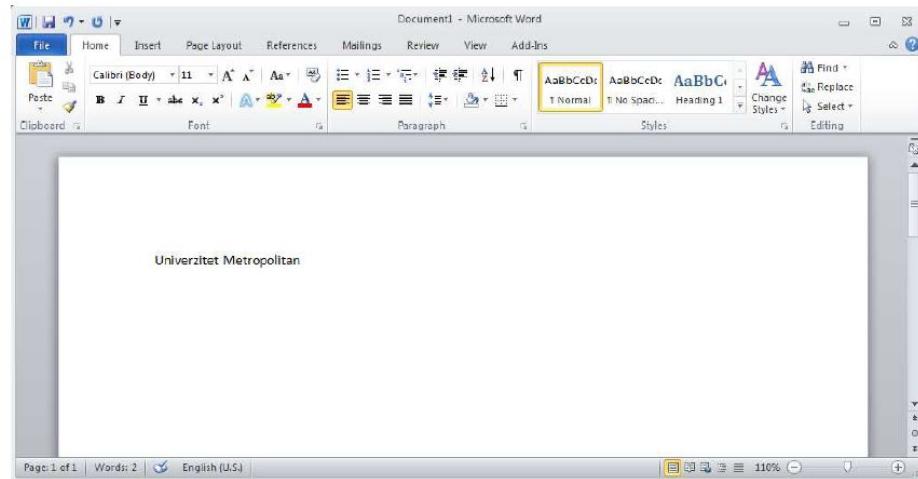
Kada se pokrene Word, u osnovnom prozoru se nalazi prazan dokument

MS Word se može pokrenuti na više različitih načina. Jedan od često korišćenih je uz pomoć opcije

Start→All Programs→Microsoft Office→ Microsoft Office Word. Nakon toga na ekranu se pojavljuje

osnovni prozor Worda.

Kada se pokrene Word, u osnovnom prozoru se nalazi prazan dokument



Slika 6.1.1 MS Word prozor

ELEMENTI MSWORD PROZORA

Svaki korisnik može, u skladu sa svojim potrebama da menja raspored elemenata prozora.

Dok korisnik radi sa dokumentom koristi sledeće elemente prozora:

Element prozora	Svrha
Radna površina	Prikazuje dokument u koji se unosi i uređuje tekst.
Lenjir	Prikazuje širinu dokumenta i njegove margine.
Naslovna traka	Prikazuje ime programa i dokumenta na kome se radi.
Traka sa menijima	Omogućava pristup menijima sa različitim Word naredbama.
Traka sa standardnim alatima	Sadrži osnovne naredbe za rad sa dokumentom.
Statusna traka	Prikazuje informacije o dokumentu i stanje nekih tastera.
Klizač	Koristi se za kretanje po dokumentu

Slika 6.1.2 Tabela-1 Elementi prozora

Raspored osnovnih elemenata prozora može biti drugačiji za svakog korisnika. Svaki korisnik može, u skladu sa svojim potrebama da menja raspored ovih elemenata. Prilikom rada sa Word dokumentom, koriste se različite komande uz pomoć menija ili traka sa različitim alatima. Za korišćenje komandi na trakama dovoljno je kliknuti na željenu opciju. Za korišćenje menija potrebno je, pre svega, kliknuti na naziv menija i otvoriti se prozor sa svim opcijama tog menija.

Osim navedenih načina, za pokretanje pojedinih opcija koriste se tzv. prečice sa tastature. Na primer, za otvaranje dokumenta potrebno je kliknuti na taster **CTRL**, koji se obično nalazi u donjem levom uglu tastature, istovremeno pritiskajući taster **O**. Često se kombinovano kucanje dva ili više tastera piše sa znakom „+”, što u ovom slučaju znači: **CTRL+O**.

UNOS TEKSTA

Tačka umetanja pokazuju mesto u dokumentu na koje će biti unest tekst

Kada se Word pokrene, u novom, praznom dokumentu, u gornjem levom uglu, prikazan je uspravna mala linija koja trepće. Ta linija se naziva tačkom unosa ili tačkom umetanja, koja pokazuje mesto u dokumentu na koje će biti unet tekst. Za unošenje teksta dovoljno je jednostavno kucati odgovarajuće tastere na tastaturi. Dok se tekst unosi, tačka unosa se pomera u desnu stranu.

Ako se prilikom kucanja pogreši treba pritisnuti taster Backspace (najčešće se nalazi u gornjem desnom uglu tastature) za brisanje znaka sa leve strane pointera unosa. Za brisanje znaka sa desne strane pointera koristi se taster Delete.

Na kraju reda nije neophodno pritisnuti nijedan taster, jer Word automatski prelama tekst. Osim tastera sa slovima, prilikom unošenja teksta često se koriste i tasteri sa određenim funkcijama. Na primer, taster Enter koristi se za početak novog paragrafa. Definisanje paragrafa u Word-u je vrlo važno jer se određeni poslovi uređenja teksta odnose samo na pojedine paragrafe.

Dakle pritiskom na taster Enter, Word zavrašava uneti red i tačka unosa se prebacuje na početak narednog reda. Da bi postojao razmak između dva reda potrebno je dva puta kliknuti na taster Enter, nakon završenog paragrafa. Često je potrebno spojiti dva paragrafa u jedan. Jedan od načina je da se tačka unosa postavi na početak drugog paragrafa. Pritiska se taster Backspace, pri čemu se dati paragraf pomera na gore i spaja sa prvim. Postoje preporuke koje se odnose na lepo uređivanje teksta. Na primer, između zareza ili tačke i naredne reči postoji razmak. Taj razmak ne postoji iza reči koja prethodi zarezu ili tački. Potrebno je koristiti naša slova uvek kada je to moguće. Znači, korišćenje c umesto č ili z umesto ž i sl., nije preporučljivo.

POMERANJE TAČKE UNOSA

Za pomeranje tačke unosa dovoljno je kliknuti na bilo koji deo dokumenta u koji je već unet tekst i tačka unosa će se pojaviti na tom mestu.

Za pomeranje tačke unosa dovoljno je kliknuti na bilo koji deo dokumenta u koji je već unet tekst i tačka unosa će se pojaviti na tom mestu. Osim toga, za pomeranje tačke unosa za jednu mesto levo ili desno, odnosno gore ili dole, koriste se tasteri sa odgovarajućim strelicama. U narednoj tabeli navedeni su osnovni načini za pomeranje pointera unosa.

Element prozora	Svrha
Radna površina	Prikazuje dokument u koji se unosi i uređuje tekst.
Lenjir	Prikazuje širinu dokumenta i njegove margine.
Naslovna traka	Prikazuje ime programa i dokumenta na kome se radi.
Traka sa menijima	Omogućava pristup menijima sa različitim Word naredbama.
Traka sa standardnim alatima	Sadrži osnovne naredbe za rad sa dokumentom.
Statusna traka	Prikazuje informacije o dokumentu i stanje nekih tastera.
Klizač	Koristi se za kretanje po dokumentu

Slika 6.1.3 Tabela-2 Načini za pomeranje pointera unosa

KRETANJE KROZ DOKUMENT

Dok korisnik koristi klizač za kretanje po dokumentu, tačka unosa je i dalje mestu na kojem je i bio

Nakon pokretanja obično se vidi samo deo dokumenta i to gornji deo prve strane. Ako željeni deo nije vidljiv potrebno je kretati se po dokumentu da bi se došlo do željenog dela i tada kliknuto na odgovarajuće mesto. U sledećoj tabelia navedeni su načini pomeranja dokumenta i akcije koje je potrebno uraditi.

Pomeranje tačke unosa	Akcija
Jedan znak levo ili desno	Pritisnuti taster \leftarrow ili \rightarrow .
Pomeranje za jednu reč levo ili desnu	Pritisnuti istovremeno taster CTRL i taster \leftarrow ili \rightarrow .
Jedan red gore ili dole	Pritisnuti taster \uparrow ili \downarrow .
Na početak ili kraj reda	Pritisnuti taster Home ili End .
Jedan prozor gore ili dole	Pritisnuti taster Page Up ili Page Down .

Slika 6.1.4 Tabela-3 Kretanje kroz dokument

Veoma je važno napraviti razliku između pomeranja tačke unosa i kretanja po dokumentu. Tačka unosa ostaje na mestu na kojem se nalazi sve dok se ne postavi na neko drugo mesto npr. klikom na željeno mesto. Dok korisnik koristi klizač za kretanje po dokumentu, tačka unosa je i dalje mestu na kojem je i bio. Na primer, uz pomoć klizača moguće je da korisnik gleda sadržaj četvrte strane dok se tačka unosa nalazi na prvoj. Preciznije rečeno, tačka unosa ostaje na mestu poslednjeg unosa teksta sve dok se ne klikne na neko drugo mesto. Ako je potrebno da se korisnik vrati na mesto pointera unosa koristi se kombinacija tastera SHIFT i F5.

IZBOR TEKSTA

Tekst se može izabrati uz pomoć miša ili tastature, pri čemu se izabrani tekst razlikuje od ostalog po boji pozadine koja je sada crna

Mnogi poslovi u Word-u podrazumevaju tzv. izbor teksta. Tekst se može izabrati uz pomoć miša ili tastature, pri čemu se izabrani tekst razlikuje od ostalog po boji pozadine koja je sada crna. Na primer, da bi se izabrao čitav red potrebno kliknuti levo od željenog reda.

Za poništavanje izbora potrebno je kliknuti bilo gde u prozoru (preporučljivo izvan izabranog dela) ili jednostavno pomeriti cursor uz pomoć tastera na tastaturi.

Kretanje kroz dokument	Akcija
Jedan red gore ili dole	U delu sa vertikalnim klizačem kliknuti jednom na strelicu koja pokazuje na gore ili na dole, a nalaze se na vrhu i dnu trake.
Pomeranje više redova na gore ili dole	U delu sa vertikalnim klizačem kliknuti na deo između klizača i strelice na gore ili dole.
Kontinualno pomeranje na gore ili dole	Kliknuti na klizač i držeći levi taster miša pomerati klizač na gore ili dole. Pored cursora pojaviće se informacija o broju strane koja je trenutno prikazana na radnoj površini.

Slika 6.1.5 Tabela-4 Izbor teksta

ČUVANJE DOKUMENTA

Da bi dokument ostao zabeležen u memoriji računara potrebno je da se snimi u odgovarajuću datoteku

Prilikom otvaranja novog praznog dokumenta, Word mu dodeljuje ime koje ima oblik DocumentN gde je N redni broj novog dokumenta, pa ako su otvorena tri nova dokumenta oni mogu imati imena Document1, Document2 i Document3. Ako je pre otvaranja narednog dokumenta već korišćen naziv npr. Document3, naredni novi dokument će dobiti ime Document4. Da bi dokument ostao zabeležen u memoriji računara potrebno je da se snimi u odgovarajuću datoteku. Za snimanje se koristi opcija Save u meniju File ili kombinacija tastera **Ctrl+S**. Nakon toga otvorice se dijalog prozor Save As. U polje File Name unosi se ime datoteke. Preporuka je da ime opisuje kratko sadržaj dokumenta. Direktorijum koji je predložen za snimanje naveden je u polju Save In. Najčešće je to My Documents. Za promenu direktorijuma u koji će biti smeštena datoteka, potrebno je kliknuti na strelicu sa desne strane polja Save In i izabrati željeni direktorijum ili u levom delu prozora izabrati neki od već predloženih. Na kraju potrebno je kliknuti na Save.

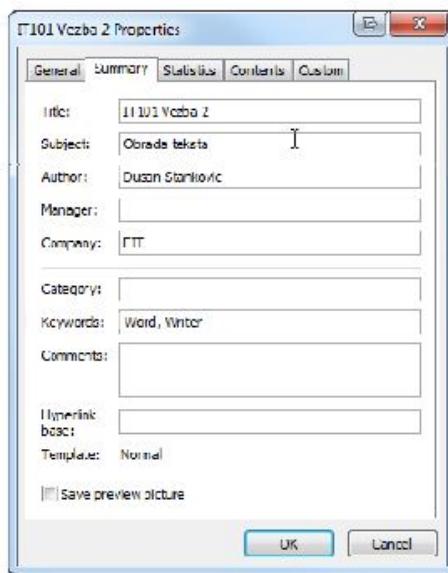
Ovaj postupak se odnosi na prvo snimanje novog dokumenta. Snimanje naknadnih promena podrazumeva automatsko pamćenje informacija, bez prikazivanja dijalog prozora. Kada je dokument snimljen, korisnik može da nastavi rad na njemu.

Preporučljivo je povremeno, čak često, snimati unete promene, kako bi u slučaju nestanka struje ili nekih drugih problema, gubitak bio minimalan. Za snimanje se koriste opcija File>Save, opcija Save na liniji sa standardnim alatima ili kombinacija tastera CTRL+S. Dokumenti se, ako se ne izvrši izbor neke druge opcije, snimaju u .doc formatu.

OSOBINE DOKUMENTA

Svaki dokument ima neke osobine koje daju informacije o dokumentu

Svaki dokument ima neke osobine koje daju informacije o dokumentu, kao što su ime osobe koja je kreirala dokument, datum kada je dokument napravljen, datum poslednjeg menjanja dokumenta itd. Neke osobine unosi korisnik, dok ostale Word generiše automatski. Da bi se videle trenutne osobine dokumenta potrebno je kliknuti na File>Properties.



Slika 6.1.6 Osobine dokumenta

Na primer, ako se klikne karticu Summary dobijaju se informacije o nazivu dokumenta, njegovom autoru itd. Ovi podaci mogu se menjati promenom u datim poljima. Važna je uočiti da naziv (Title) može potpuno da se razlikuje od naziva datoteke ovog dokumenta. Osim navedenih podataka u delu Comments mogu se uneti i komentari koji su korisni ako na dokumentu radi više ljudi. Većina principa koji su gore spomenuti važe i za OpenOffice Writer. Dokumenti pripremljeni uz pomoć Writier-a podrazumevano se snimaju sa .odt ekstenzijom

▼ 6.1 Pokazna vežba: Kreiranje šablonu u MS Wordu

STILOVI

Stil je zbirka zadatih formata koja se može zapamititi pod određenim imenom

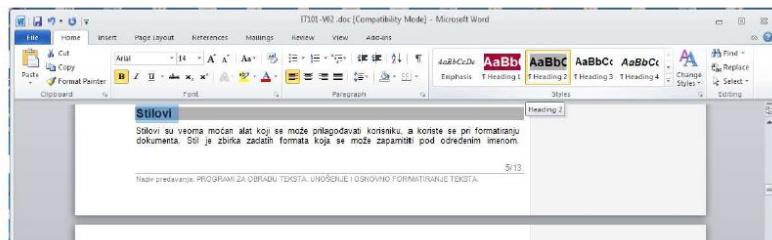
Predviđeno vreme pokazne vežbe je 10 minuta.

Stilovi su veoma moćan alat koji se može prilagođavati korisniku, a koriste se pri formatiranju dokumenta. Stil je zbirka zadatih formata koja se može zapamititi pod određenim imenom.

Definisanje stila je brže nego promena stila svakog pojedinačnog elementa i omogućava dosledno poštovanje željenog formata. Stilovi se obično dele na stilove paragrafa i stilove pojedinačnih znakova. Stilovi paragrafa se odnose na cele paragrafe i mogu uključiti sve elemente formatiranja koji utiču na izgled kao što je npr. razmak između redova.

Osnovni stil koji se podrazumeva prilikom pokretanja je Normal. Stilovi znakova se odnose na bilo koji deo teksta i mogu podrazumevati bilo koji element primenjen na bilo kom znaku, kao što su veličina slova, tip slova, podvučen tekst itd. U ovom slučaju ne postoji podrazumevani stil.

Da bi se primenio stil na željeni tekst potrebno je prvo tekst izabrati, a potom kliknuti na polje Style koje se nalazi na liniji sa alatima za formatiranje. Na listi koja će se pojaviti naziv stila je napisan odgovarajućim pismom i poravnanjem, a simboli označavaju da li se radi o stilu odlomka ili znakova. Stil se bira klikom na naziv.



Slika 6.2.1 Odabir stila

KREIRANJE STILOVA

Korisnik prilikom rada nije ograničen samo na stlove Word-a već ima i mogućnost kreiranja sopstvenih

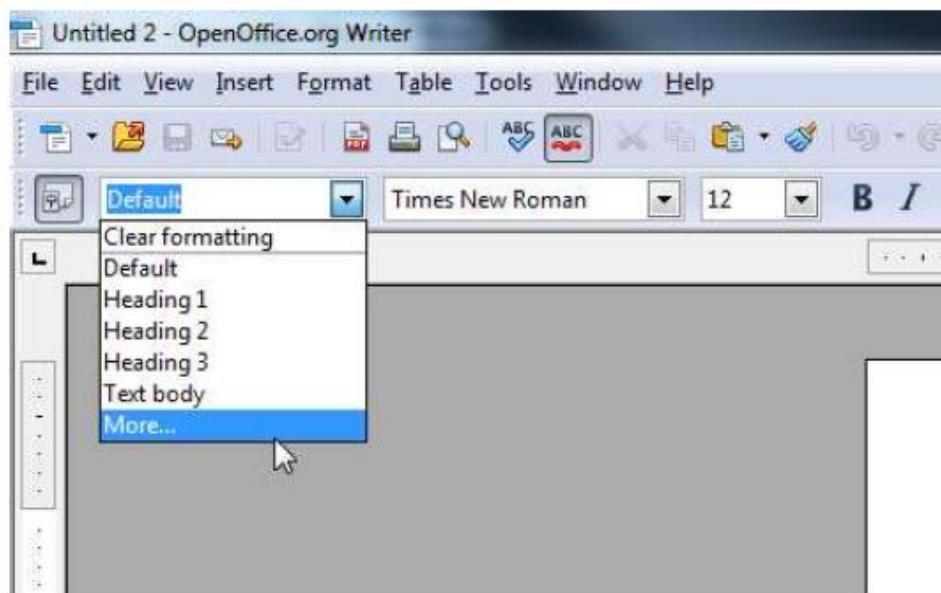
Korisnik prilikom rada nije ograničen samo na stlove Word-a već ima i mogućnost kreiranja sopstvenih. To je i jedna od najvećih prednosti alata za obradu teksta. Sopstveni stil se stvara na sledeći način. Pronađe se deo na koji se želi primeniti novi stil. Taj deo teksta se formatira promenom npr. tipa slova, razmaka između redova i sl. Taj stil će se pojaviti u listi Style, pa ako se na taj naziv klikne moguće je uneti ime novog stila, koje zamenjuje trenutni naziv. Na kraju se pritisne Enter. Osim ovog načina, za kreiranje stilova može se koristiti opcija and u meniju Format>Styles and Formatting. A potom se klikne na New Style....

U delu Style Type bira se Character ili Paragraph u zavisnosti od toga da li se novi stil odnosi na karakter ili paragraf. U polje Name unosi se ime stila. Preporučuje se da ime bude opisno, npr. Italic Indented. Ako se novi stil bazira na nekom već postojećem u delu Based On bira se stil koji se menja. Ako je potrebno da novi stil bude deo šablona datog dokumenta, treba potvrditi opciju Add to Template. Ako se ova opcija ne odabere, stil će biti primenjen samo na trenutni dokument. Potvrda Automatically Update se odnosi samo na stlove paragrafa. Ako je opcija potvrđena sve promene koje se odnose na format u paragrafima formatiranih ovim stilom biće dodata definiciji stila. Na kraju se klikne na opciju Apply za potvrdu svih učinjenih promena

KREIRANJE WRITER STILOVA

Kod OpenOffice Writer-a osim stilova paragrafa i pojedinačnih znakova postoje i stili okvira, strana i listi

Kod OpenOffice Writer-a osim stilova paragrafa i pojedinačnih znakova postoje i stili okvira, strana i listi. Za menjanje ovih stilova, na sličan način kao i u slučaju Word-a, klikne se na deo za stilove na liniji sa alatima za formatiranje.



Slika 6.2.2 OO Writer stili

▼ 6.2 Pokazna vežba: Poslovna komunikacija

KARAKTERISTIKE POSLOVNE KOMUNIKACIJE

Efektivna poslovna poruka obezbeđuje praktične informacije, pruža činjenice, pojašnjava i sažima informacije, jasno ističe kome je komunikacija upućena, ubedjuje i preporučuje

Predviđeno vreme pokazne vežbe je 10 minuta.

Poslovno komuniciranje se sprovodi radi poslovnih aktivnosti. Cilj komunikacije je da se uspešno obavi posao.

Efektivne poslovne poruke imaju niz zajedničkih karakteristika:

- obezbeđuju praktične (upotrebljive) informacije
- pružaju činjenice pre nego utiske

- pojašnjavaju i sažimaju informacije
- jasno ističu kome su upućene
- ubeđuju i preporučuju.

Jedno od najvažnijih pitanja u radu IT profesionalca je organizacija i vođenje **poslovnih razgovora**. U nekim slučajevima uspešnost IT profesionalca direktno zavisi od veštine vođenja poslovnih razgovora. Zato se svaki poslovni razgovor mora pažljivo pripremiti i voditi. Tipičan poslovni razgovor obuhvata sledeće aktivnosti:

- priprema za razgovor
- otpočinjanje razgovora
- informisanje
- argumentovanje
- neutralisanje prigovora
- donošenje odluke – završetak razgovora.

Svaka od navedenih aktivnosti zahteva različita znanja i veštine od IT profesionalca, pa je potrebno da se on posebno edukuje za vođenje poslovnih razgovora.

PREDSTAVLJANJE PUTEM CV-A

CV u profesionalnoj komunikaciji

Deo poslovne komunikacije čini i pisanje CV-a, odnosno njegova priprema. CV predstavlja kratku biografiju veština i znanja kojima se predstavljamo potencijalnom poslodavcu.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

✓ 6.3 Pokazna vežba: Kako napisati poslovno pismo

POSLOVNO PISMO

Svrha poslovnog pisma je da čitaocu pruži određenu informaciju.

Predviđeno vreme pokazne vežbe je 40 minuta.

Poslovno pismo je jedan od osnovnih vidova komunikacije u savremenom poslovanju, čija je svrha da čitaocu pruži određenu informaciju. Način komuniciranja je veoma važan i postoji potreba da se ispoštuju određena pravila. Poslovno pismo će oslikati vas vaše manire i pre nego što se sretnete sa osobom kojoj pišete pa vam zato nudimo nekoliko saveta kako da što bolje ovo pismo napišete.

ELEMENTI POSLOVNOG PISMA

Ukoliko se obraćate osobi koja poseduje neku titulu, zvanje, potrebno je obratiti se sa tom određenom titulom

Adresant - pošiljalac pisma i podaci njegovi se uvek prvi naznačuju u pismu. Oni mogu biti već biti uneti u zaglavlju, kao memorandum kompanije, ili se mogu otkucati (naziv ili ime pošiljaoca, adresa i mesto, telefon, faks, e-mejl, web adresa ili drugi bitni podaci).

Adresat - primalac pisma (osoba, organizacija, kompanija...). Ispod podataka o pošiljaocu se uvek naznačuju podaci adresata - naziv organizacije i eventualno ime osobe kojoj je pismo upućeno, adresa i mesto.

Mesto i datum pisanja. Ovo obavezno naznačite u pismu. U principu se piše u gornjem desnom uglu strane, međutim, ako imate prostora u donjem levom uglu strane, možete i tamo naznačiti mesto i datum.

Oslovijavanje primaoca. Ako pišete konkretnoj osobi, pismo započnite sa "Poštovani gospodine XY," ili "Poštovana gospodo-ice XY.". Ukoliko ne znate ime osobe kojoj pišete, stavite samo "Poštovani". Ukoliko već poznajete osobu kojoj se obraćate i imate sa njom prisniji odnos, oslovljavanje može biti neformalnije, tipa "Dragi gospodine XY" ili "Draga Sanja" i slično. Ukoliko se obraćate osobi koja poseduje neku titulu, zvanje, potrebno je obratiti se sa tom određenom titulom ("Poštovani profesore"). Ukoliko se koncretizuje kome se npr. profesoru obraća potrebno je navesti prezime, nikako ime ("Poštovana profesorka Nikolić"). Vrlo je nekulturno obraćati se po imenu profesoru, i potrebno je voditi računa o takvima stvarima.

Predmet - pismo ne mora da započinje oslovljavanjem adresata, nego da sadrži samo naznaku predmeta pisma, u kratkom i jasnom opisu. Ispod predmeta se pravi jedan red razmaka i počinje prvi pasus pisma. Na primer, " Predmet : Ponuda"

ELEMENTI POSLOVNOG PISMA - NASTAVAK

Na kraju pisma se obavezno potpišite, a uz potpis otkucajte i puno ime i prezime, i ukoliko je potrebno i funkciju ili zvanje.

Telo pisma i pozdrav - u prvom pasusu kratko i jasno iznesite razlog svog pisanja, a u ostaku pisma dajte relevante informacije o temi vašeg pisma. U poslednjem pasusu se obično iznose očekivanja za neku dalju akciju po pitanju teme Vašeg pisma. Pismo završite sa prigodnim pozdravima tipa "Uz poštovanje", "Očekujući Vašu porudžbinu, mi Vas sa poštovanjem pozdravljamo", "Pozdravljam Vas u nadi da ćemo saradivati na obostrano zadovoljstvo", "Očekujući Vaš odgovor, srdačno Vas pozdravljam" i slično.

Potpis pošiljaoca - na kraju pisma se obavezno potpišite, a uz potpis otkucajte i puno ime i prezime, i ukoliko je potrebno i funkciju ili zvanje.

Prilozi - uz pismo šaljete i dodatne dokumente, obavezno to naznačite tako što ćete u donjem levom uglu strane napisati "Prilozi:" i ispod toga ih nabrojati pod crticama.

DUŽINA PISMA

Poslovna pisma u principu ne treba da prelaze jednu stranu.

Dužina pisma - poslovna pisma u principu ne treba da prelaze jednu stranu. Ukoliko imate potrebu da primaocu skrenete pažnju i na podatke kojih ima mnogo, ukratko ih spomenite u pismu, a detaljnije ih predstavite u prilogu koji dostavljate. Poslovna komunikacija treba da bude kratka i jasna, koliko god je to moguće, i da već u početku pisma bude vidljivo koja je svrha pisanja. Ukoliko vaše pismo ipak ima više strana, a šaljete ga poštom, postarajte se da stranice budu zaheftane.

Font - standardne fontove, pogotovo ako pismo šaljete e-mejlom. Ako pošaljete pismo u nekom od nestandardnih fontova, rizikujete mogućnost da primalac neće moći da ga pročita pošto nema taj font. Arial i Times New Roman su možda najprimereniji za ovakva pisma, a maximum veličina fonta je Arial 10 ili Times New Roman 12.

FORME POSLOVNOG PISMA

Pisanje u blok formi prvenstveno podrazumeva da pasusi počinju u istoj ravni i da se između pasusa pravi red razmaka

Blok forma. Pisanje u blok formi prvenstveno podrazumeva da pasusi počinju u istoj ravni i da se između pasusa pravi red razmaka. I podaci primaoca i ostali delovi pisma takođe su poravnati sa osnovnim sadržajem pisma.

Zupčasta forma. U zupčastoj formi se pasusi ne razdvajaju redom razmaka, nego se nastavljaju odmah jedan ispod drugog. Pasusi se razlikuju po tome što se početak svakog od njih malo uvlači. Podaci po se ovde mogu ispisati u sredini strane, poput naslova. Vaš potpis, odnosno potpis pošiljaoca se stavlja u desnom delu strane.

Kombinovana forma. Pismo u ovoj formi se sastavlja isto kao u blok formi, osim što se podaci primaoca, kao i potpis pošiljaoca stavljuju u desni deo deo strane.

Slobodna forma. Generalno nije neophodno da se u dlaku pridržavate svih pravila forme pisama. Raspoloženje elemenata malo možete i da menjate, sve dok ne narušavate jasnost odvajanja pasusa, sadržajnost, preglednost i urednost pisma, odnosno dokle god su svi neophodni elementi na broju i dokle god je na prvi pogled uočljivo ko piše kome, koja je osnovna svrha pisma, kada je pisano i da li ga prati dodatna dokumentacija. Ovo smo mi nazvali "slobodna forma".

ODABIR FORME PISMA

Blok forma je i pogodnija za kraća pisma, obzirom na razmake koji se stavljuju, te se vizuelno dobija na dužini teksta

Kako odabrati formu? Ukoliko pismo šaljete u koverti koja ima prozoriči u desnom delu, obratite pažnju i da su naziv i adresa primaoca otkucani u desnom delu strane (kombinovana

forma), kako bi se potpuno videli kroz prozorčić i kako biste omogućili poštaru da vaše pismo dostavi na pravu adresu. Ukoliko je prozorčić koverte na levoj strani, više će vam odgovarati blok forma. Blok forma je i pogodnija za kraća pisma, obzirom na razmake koji se stavljuju, te se vizuelno dobija na dužini teksta. Sa zupčastom formom štedite na prostoru, tako da više teksta može stati na jednu stranu. Zupčasta forma se naziva još i "evropska, svečana" forma, te se može koristiti pri obraćanju značajnim poslovnim partnerima, za pisanje molbi i slično, mada ovo više i nije striktno pravilo. Blok forma je poznata i kao "američka", pa vam ova određenja možda mogu pomoći da formu pisanja prilagodite geografskom poreklu primaoca.

KREIRANJE PROPRATNOG PISMA

Prikaz kreiranja propratnog pisma

Pod propratnim pismom smatramo dokument koji prati vaš CV prilikom prijavljivanja za posao. Propratno pismo je prilika da se da neki lični pečat biografiji i da se ukaže na neke specifičnosti. Propratno pismo ima neke specifičnosti i to:

1. Propratno pismo obično nije dugačko (200-250 reči) i on se koristi da se u par rečenici daju najvažnije crte naše biografije, kako bi bolje privukli pažnju i istakli naš CV. U određenim izuzecima, kada oglas za posao to nalaže, propratno pismo može biti i duže.
2. Potrebno ga je posmatrati kao uvod u CV, da se prikažu aktivnosti i karakteristike koje su od posebnog značaja za prijavu.
3. Trebalo bi da odgovore na pitanja koja bi poslodavac mogao sebi da postavi čitajući vaš CV.
4. Posmatrajući propratno pismo možemo reći da ima za pitanje i to: Zašto da zaposlim ovog kandidata?
5. Ima formu poslovnog pisma.
6. Treba da zvuči iskreno i da bude puno elana, a kao i CV treba ga prilagoditi konkursu na koji se prijavljujete.
7. Posmatrati ga kao ličnu prezentaciju.

PRIMER PROPRATNOG PISMA

Pogledati propratno pismo. Da li bi ste nešto menjali?

Nikola Mišić
Vojvode Mišića 97
18000 Niš
Mob: 064/388-3477
E-mail: nikolam.011@gmail.com

Vocell
Omladinskih brigada 26
11 000 Beograd

Predmet: Prijava na konkurs za Java Developer-a

Poštovani,

Pišem Vam povodom konkursa objavljenog na sajtu <http://itposlovi.info/>, 01.02.2017. u kome je navedeno da tražite „Java Developer-a“.

Trenutno studiram na Fakultetu informacionih tehnologija Univerziteta Metropolitan, smer Informacione tehnologije i planiram da diplomiram do kraja školske godine. Već četiri meseca obavljam stručnu praksu u kompaniji MJS, gde radim na nekoliko projekata vezanih za razvoj softverskih rešenja, prevashodno koristeći Java i C#. Na studijama sam od prve godine radio na Java projektima, gde sam imao prilike da steknem bliži uvid i unapredim svoja znanja iz ne samo Java, nego i drugih front-end i back-end tehnologija. Radio sam u razvojnim okruženjima Visual Studio, Eclipse, a posedujem i odlično znanja C, C++, SQL, HTML, PHP, CSS, JavaScript, Spring i razvoj Android aplikacija.

Smatram da znanja koja posedujem mogu u velikoj meri doprineti razvoju u oblasti za koju je raspisan konkurs. Takođe, verujem da bi rad u Vašoj kompaniji predstavljao idealnu priliku za profesionalno usavršavanje i ispunjenje mojih dugoročnih ciljeva u karijeri i stoga se nadam da ćete mi pružiti priliku da Vam se i lično predstavim.

Srdačno,

Nikola Mišić

Beograd,

15.09.2017.

Slika 6.3.1 Primer propratnog pisma

▼ Poglavlje 7

Zadatak za samostalni rad: pravljenje šablonu

ZADATAK ZA SAMOSTALNI RAD

Kreiranje templejta

Predviđeno vreme za izradu sledećeg zadatka je 30 minuta.

Zadatak

Koristeći odgovarajuće alate MS Word ili OpenOffice.org Writer potrebno je napraviti 2 šablonu.

1. Prvi šablon koji će koristiti prilikom pisanja lično vašeg propratnog pisma. Templejt je potrebno da sadrži sve elemente propratnog pisma.
2. Drugi šablon je šablon za kreiranje CV-a. Šablon je potrebno da sadrži: *Mesto za fotografiju, Ime, Prezime, Broj telefona, Adresa, Obrazovanje, Radno iskustvo, Rad na računaru, Jezici*.

Za ideju šablonu možete pogledati razne primere na Internetu neke, kako bi dobili ideju za kreiranje sopstvenog šablonu.

✓ Poglavlje 8

Domaći zadatak

OPIS DOMAĆEG ZADATKA - DZ14

Kreirati CV, poslovno pismo i šablon domaćeg zadatka

Očekivano vreme izrade zadatka: 45 minuta

Zadatak 1 (Predviđeno vreme izrade: 15 minuta)

Kreirajte svoj CV. Sa eLearning sistema preuzmite šablon za kreiranje CV dokumenta i popunite ga svojim podacima. Snimite ga kao:

CV_Ime_prezime_briIndexa.

Zadatak 2 (Predviđeno vreme izrade: 20 minuta)

Koristeći odgovarajuće alate MS Word ili OpenOffice.org Writer kreirati propratno pismo koje bi ste poslali uz CV. Izaberite neki trenutno objavljeni IT oglas za posao ili praksu na Infostudu (priložite ga uz domaći zadatak) i napišite za taj posao propratno pismo.

Sačuvati u Word dokumentu pod imenom
IT101-DZ14_Ime_Prezime_briIndexa-2.docx

Zadatak 3 (Predviđeno vreme izrade: 10 minuta)

Koristeći odgovarajuće alate MS Word ili OpenOffice.org Writer napravite šablon domaćeg zadatka, koji ćete koristiti u kasnijim izradama domaćih zadataka, kao i projektnog zadatka, a čiji su osnovni delovi:

Logo Univerziteta/Fakulteta

Domaći Zadatak br.,

Naziv predmeta,

Naslov,

Ime i prezime studenta,

Broj indeksa,

Tekst zadatka,

Rešenje zadatka

Svaki deo u okviru zadatka je potrebno da ima svoj naslov, na primer "Opis zadatka", "Rešenje zadatka" (koristiti heading opcije)

Pre teksta i rešenja zadatka potrebno je da se nađe automatski generisan sadržaj sa brojem strana.

Prilikom slanja maila predmetnom profesoru ili asistentu koristiti pravila poslovne komunikacije. Ovo pravilo je potrebno usvojiti i koristiti do kraja školovanja.

IT101-DZ14_Ime_Prezime_briIndexa-3.docx

▼ ZAKLJUČAK

ZAKLJUČAK

U ovom predavanju smo razmotrili uticaj promena u društvu i tržištu na razvoj informacionih tehnologija. Međutim, za napredovanje tehnologija neophodno je i kontinualno usavršavanje tokom karijere, kao i neizostavne profesionalne komunikacije koja je važna ne samo kod IT stručnjaka, nego kod svih profesionalaca.

Literatura

1. ACM Code of Ethics and Professional Conduct, ACM Council, 1999.
2. IEEE Code of Ethics, IEEE Board of Directors, 1990.
3. Dragoslav Jokić, Anđela Mikić, Komunikologija biznisa, Naučno istraživački centar Užice, 2005.
4. Debra Dž. Džonson, Kompjuterska etika, Službeni glasnik, Beograd, 2006.



IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

SOCIJALNI KONTEKST RAČUNARSTVA

Lekcija 15

PRIRUČNIK ZA STUDENTE

IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

Lekcija 15

SOCIJALNI KONTEKST RAČUNARSTVA

- ▼ SOCIJALNI KONTEKST RAČUNARSTVA
 - ▼ Poglavlje 1: Socijalna informatika
 - ▼ Poglavlje 2: Bioinformatika
 - ▼ Poglavlje 3: E-Commerce
 - ▼ Poglavlje 4: Bibliotečki sistem
 - ▼ Poglavlje 5: eUprava
 - ▼ Poglavlje 6: Uticaj ICT na grafički dizajn
 - ▼ Poglavlje 7: Pokazna vežba: EC, CMS i WordPress
 - ▼ Poglavlje 8: Zadatak za samostalni rad: WordPress
 - ▼ Poglavlje 9: DOMAĆI ZADATAK
 - ▼ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

✓ Uvod

UVOD

U ovom predavanju biće reči o raznim oblastima primene informaciono komunikacionih tehnologija

U ovom predavanju biće reči o raznim oblastima u kojima se uključuje rad informaciono komunikacionih tehnologija, kako bi se unapredio rad i efikasnost. Biće reči o sledećim temama:

- Uticaj IT na društvo
- Digitalna podela
- Primena IT u zdravstvu
- Primena IT u bioinformatici
- Primena IT u elektronskom poslovanju
- Primena IT u obrazovanju
- Primena IT u radu vladinih organizacija i dr.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 1

Socijalna informatika

SOCIJALNI UTICAJ NA DRUŠTVO

Uz relativno mala ulaganja svaka kompanija može da ponudi svoje proizvode čitavom svetu

Odlika vremena u kome živimo je da postoji ogroman uticaj IKT na društvo. Nezavisno od toga da li neka osoba koristi ili ne IKT, one utiču posredno ili neposredno na njen svakodnevni život. Države koriste IKT da upravljaju društvom tako da su svi članovi društva pod uticajem IKT. Međutim, IKT se koriste i u ostalim segmentima života. Generalno, može se reći da se IKT koriste u:

- državnom životu
- poslovnom životu
- privatnom životu.

Kao što je napred rečeno, gotovo sve države koriste IKT kao jedan od instrumenata za upravljanje državom. Svaki građanin odmah po rođenju biva registrovan u informacioni sistem države, a kasnije tokom života, država prikuplja o njemu i dodatne podatke. Na osnovu ovih podataka građani lakše ostvaruju svoja prava vezana za izdavanje ličnih dokumenata, glasanje, školovanje, rad i osiguranje. Sa druge strane, država koristi iste podatke za regulisanje pitanja poreza, vojnih obaveza, stvaranje demografske slike stanovništva, planiranje i upravljanje školskog, zdravstvenog i bezbednosnog sistema, kao i za mnoge druge potrebe.

Ljudi koriste masovno IKT u svom poslovnom životu. Danas je u razvijenim zemljama teško zamisliti poslovanje preduzeća ili institucija bez korišćenja IKT. Nema oblasti poslovanja koja nije pokrivena nekom od IKT. One se koriste za automatizaciju svakodnevnih poslovnih aktivnosti, za upravljanje poslovnim sistemima, nabavkom, proizvodnjom, prodajom, odnosima sa klijentima i partnerima itd. Kompanije koje koriste IT su u prilici da optimiziraju svoje proizvode, da ih prilagode potrebama klijenata i da ih učine efikasnijim. Na taj način one su ostvarivale komparativnu prednost na tržištu.

Razvoj Interneta i servisa koji se ostvaruju putem njegove infrastrukture doveo je do pojave novih oblika komunikacije koji su prethodno bili nezamislivi ili preskupi. Mobilna telefonija i bežični Internet su omogućili ljudima da budu online bilo kada i bilo gde i da na taj način u svakom trenutku koriste informacione resurse svojih institucija ili kompanija. Ovo je omogućilo kompanijama da lakše koordiniraju radom svojih udaljenih odeljenja, a ljudima da budu mobilniji.

Zahvaljujući Web-u i male kompanije su do bile mogućnost da budu vidljive na svetskom tržištu. Uz relativno mala ulaganja svaka kompanija može da ponudi svoje proizvode čitavom svetu. Ovo je učinilo ekonomski tokove mnogo dinamičnijim i otvorenijim.

UTICAJ IKT NA KVALITET ŽIVOTA

Ljudi koriste IKT za učenje, zabavu, kupovinu, informisanje i komunikaciju

Sa padom cena računara i komunikacionih uređaja, ljudi sve više koriste IKT i u privatnom životu. Ljudi koriste IKT za učenje, zabavu, kupovinu, informisanje i komunikaciju. Zahvaljujući primeni IKT kvalitet života ljudi se znatno poboljšao. Evo nekoliko primera:

- Zahvaljujući e-Learning tehnologiji mnogi ljudi koji ne žive u univerzitetskim gradovima ili su prinuđeni da rade i putuju su dobili šansu da studiraju ili pohađaju specijalizovane kurseve.
- Zahvaljujući e-Trgovini ljudi mogu da nađu neki proizvod pod najpovoljnijim uslovima.
- Zahvaljujući vebu ljudi mogu da vide vesti gotovo u trenutku njihovog nastajanja.
- Zahvaljujući GPS sistemu ljudi mogu lako da dođu do željene lokacije.
- Zahvaljujući VoIP tehnologiji ljudi mogu jeftinije i kvalitetnije da komuniciraju.

Kao što se vidi IKT postaju neizbežan faktor modernog društva. Ljudima koji koriste IKT život postaje kvalitetniji, a život bez njih gotovo nezamisliv.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ 1.1 Online zajednice

DRUŠTVENE IMPLIKACIJE ONLAJN ZAJEDNICA

Prve online zajednice su za komunikaciju koristile tehnologije mejling listi

Ljudi sa istim ili sličnim potrebama su od uvek stvarali svoje zajednice. Bez obzira da li su organizovane kao klubovi ili udruženja, zajednice su imale za cilj da svojim članovima omoguće zadovoljenje nekih potreba. Na primer, članovi filatelističkog društva su lakše mogli da dođu do marke koja nedostaje njihовоj kolekciji. Jedan od osnovnih ciljeva zajednica je zbližavanje i bolja komunikacija među članovima. Razvoj Interneta, a pogotovo veba, otvorio je mogućnost da članovi mnogo lakše komuniciraju i da se zbliže. Stvoreni su uslovi da se stvore i zajednice koje zbog male geografske koncentracije ranije nije bilo moguće stvoriti. Zajednice koje koriste Internet kao infrastrukturu za komunikaciju između članova nazivaju se online ili virtuelne zajednice.

Prve online zajednice su za komunikaciju koristile tehnologije mejling listi. Svi članovi zajednice su bili na istoj mejling listi, pa ih je lako i brzo moglo informisati o aktivnostima zajednice. Rukovodstvo zajednice je povremeno distribuiralo news letter svim članovima, mada je i svaki član individualno mogao da komunicira sa zajednicom ili pojedinim njenim članovima. Sa pojavom veba, informacije i drugi sadržaji su mogli da se trajno publikuju na nekom veb sajtu. Tako su zajednice postale vidljive globalno, pa su mogle da privuku nove članove. Danas online zajednice kao najčešći mehanizam asinhronne komunikacije koriste forume ili časkaonice (**chat**) za sinhronu komunikaciju.

FOKUS ZAJEDNICE

Prva stvar koja definiše jednu zajednicu je njen fokus. On je određen interesom njenih članova.

Prva stvar koja definiše jednu zajednicu je njen fokus. On je određen interesom njenih članova. Tako se na Internetu mogu naći online zajednice čiji je fokus na stvaranju:

- Saradničke radne grupe koja radi na istom poslu
- Porodične grupe
- Socijalnog prostora
- Grupe za igre sa ulogama (RPG)
- Grupe za podršku obolelih
- Etničke grupe
- Profesionalne grupe
- Geografski povezane grupe
- Zajednice za podršku softveru ili hardveru
- Intelektualne diskusione grupe
- Grupe sa specijalnim interesima

Ove grupe mogu da budu više ili manje otvorene za prijem novih članova. Svi članovi mogu da doprinose radu zajednice publikovanjem određenog sadržaja koji je podložan kritičkom sagledavanju ostalih članova zajednice. Zbog toga se od svih članova očekuje da prihvate i drugačiji način razmišljanja. Članovi zajednice ne mora da imaju, a najčešće nemaju materijalne koriste od rada u zajednici, ali zato imaju dobru komunikaciju, a nekada i priznanje od zajednice.

Sledeća karakteristika online zajednica je da su svi članovi, bez obzira na rasu, veru, pol, materijalni status i obrazovanje ravnopravni. Ova činjenica je vrlo važna jer omogućava nekim članovima da u online zajednicama dobiju status koji u realnom svetu često nemaju. Zbog mogućnosti da član zajednice može biti registrovan bez provere stvarnog identiteta, članovi zajednice su u mogućnosti da kreiraju svoj novi virtuelni identitet.

Očigledno je da online zajednice imaju veliki uticaj na društvo. Pre svega online zajednice utiču na rušenje geografskih granica. Ljudi u online zajednici postaju stanovnici globalnog sela. Njihova uloga i mesto u zajednici ne zavisi u velikoj meri od njihove geografske lokacije. Pored toga online zajednice ruše i druge barijere koje postoje u realnom svetu, kao što su: nacionalna pripadnost, verska ubeđenja ili materijalni status.

▼ 1.2 Pitanja razlika

KAKO PITANJA RAZLIKA UTIČU NA ICT

Pitanja razlika koje mogu uticati na digitalnu podelu

Korisnici IKT se razlikuju po mnogim karakteristikama. Tako se ljudi razlikuju po svom materijalnom statusu, polu, obrazovanju, nacionalnosti, veri itd. Neke od tih karakteristika mogu da utiču na njihovo korišćenje ovih tehnologija. Ovde će se ukazati samo na neke probleme koje nastaju zbog ovih razlika.

RODNA RAVNOPRAVNOST

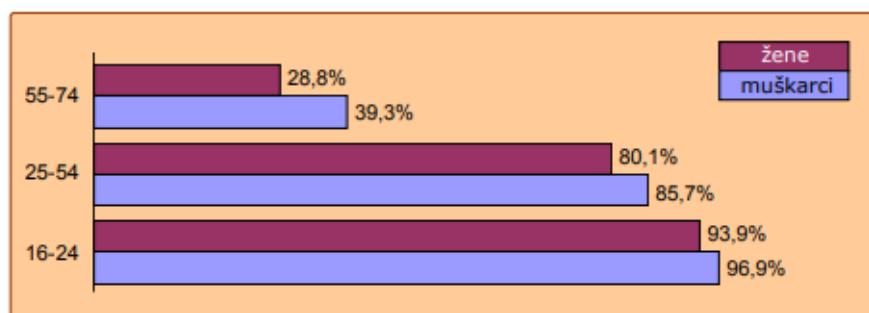
Rodna ravnopravnost se ističe i u IKT

Statistička istraživanja su pokazala da je među korisnicima IKT veći broj muškaraca nego žena. Razloge za ovakvo stanje je vrlo teško otkriti, jer je praksa pokazala da su žene podjednako dobri korisnici i kreatori IKT kao i muškarci. Zbog toga je u društvu pokrenuto pitanje rodne ravnopravnosti (**gender**) u IKT. Ovim ozbiljnim pitanjem se bavi većina međunarodnih organizacija među kojima su i Ujedinjene nacije.

Kao ilustracija za razlike bazirane na polu mogu poslužiti statistički podaci korišćenja Interneta u Srbiji i prikazan je na slici 1 koja je data u nastavku.

Najviše Internet koriste muškarci između 16 i 24 godina. Dolazi do skorog izjednačenja između muškaraca i žena što se tiče korišćenja Interneta. Razlike su minorne.

Graf. 1.17. Korišćenje računara (u poslednja tri meseca), prema polu i starosti

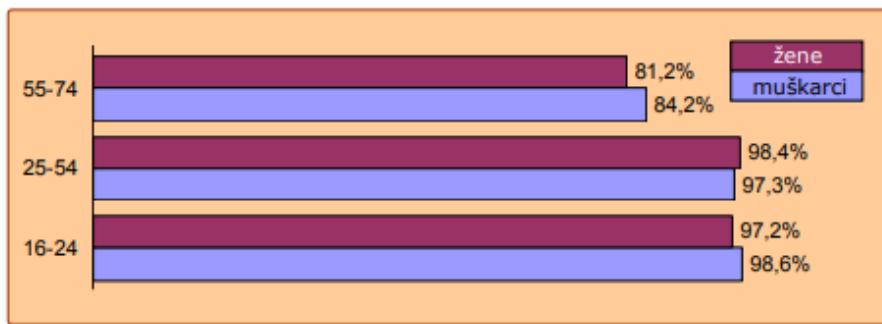


Slika 1.1.1 Korišćenje računara prema polu i starosti 2017

. <https://pod2.stat.gov.rs/ObjavljenePublikacije/G2017/pdf/G20176006.pdf> - Republički zavod za statistiku, 2017 god.

Statistike od 2017. godine ukazuju na to da je stepen korišćenja računara u Srbiji porastao, kao i da su mobilni uređaji sve više zastupljeniji među oba pola i svim generacijama.

Graf. 1.20. Upotreba mobilnog telefona, prema polu i starosti



Slika 1.1.2 Korišćenje mobilnih telefona prema polu i starosti 2017

. <https://pod2.stat.gov.rs/ObjavljenePublikacije/G2017/pdf/G20176006.pdf> - Republički zavod za statistiku, 2017 god.

PROBLEM PRISTUPA IT RESURSIMA

Jedan od najvećih problema u globalnom korišćenju IKT je da neki ljudi uopšte nemaju pristupa ovim tehnologijama

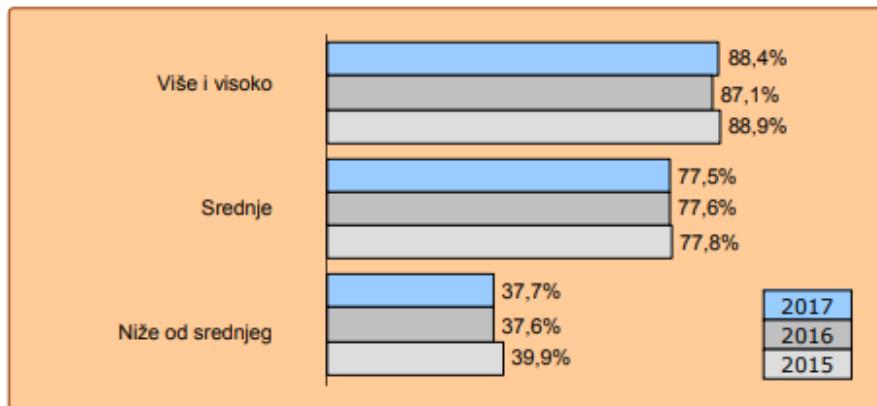
Jedan od najvećih problema u globalnom korišćenju IKT je da neki ljudi uopšte nemaju pristupa ovim tehnologijama. Razlozi za ovo su različiti, ali su najznačajniji:

- Nedostatak materijalnih sredstava za nabavku opreme
- Nedostatak obrazovanja iz oblasti IKT

Globalna statistika pokazuje da je korišćenje IKT u direktnoj vezi sa društvenim proizvodom po glavi stanovnika (GNI). U nerazvijenim zemljama i zemljama u razvoju stepen korišćenja IKT je vrlo nizak. Ovo će biti ilustrovano na primeru broja PC računara po glavi stanovnika.

Drugi važan faktor za pristup IT resursima je obrazovanje. Ljudi koji nemaju osnovno opšte obrazovanje i osnovno obrazovanje iz oblasti IKT ne mogu da koriste ove tehnologije. S druge strane, da bi se obrazovali potrebna su im materijalna sredstva, koja nemaju. Tako se ulazi u začarani krug koji ove ljudi sprečava da koriste IKT.

Graf. 1.14. Udeo korisnika računara (u poslednja tri meseca), prema nivou obrazovanja



Slika 1.1.3 - Udeo korisnika računara (u poslednja tri meseca), prema nivou obrazovanja

. <https://pod2.stat.gov.rs/ObjavljenePublikacije/G2017/pdf/G20176006.pdf> - Republički zavod za statistiku, 2017 god.

✓ 1.3 Digitalna podela

ŠTA JE TO DIGITALNA PODELA I KAKO JE NASTALA?

Uglavnom iz ekonomskih razloga neka društva nisu mogla da prate IKT razvoj

Razvoj informaciono-komunikacionih tehnologija (IKT) se u poslednjih 50 godina odvijao velikom brzinom. Uglavnom iz ekonomskih razloga neka društva nisu mogla da prate ovaj razvoj, pa je stepen primene IKT na nivou celog društva nizak. Pored toga, u okviru istog društva, pojedine osobe imaju različite mogućnosti i sklonosti da koriste IKT. Tako smo došli do situacije u kojoj se svet može podeliti na ljudе koji koriste IKT i ljudе koji ih ne koriste.

Termin digitalna podela opisuje činjenicu da se ljudi mogu podeliti na ljudе koji koriste IKT i ljudе koji nemaju pristupa ili nemaju mogućnosti da koriste moderne IKT. Kada se govori o digitalnoj podeli uglavnom se misli na korišćenje računara, telefona, televizije i Interneta. Izvorno, termin digitalna podela je nastao kada se shvatilo da ljudi koji ne koriste IKT ne mogu ravnopravno da učestvuju u modernom ekonomskom, političkom i socijalnom životu.

Ljudi koji koriste IKT imaju mogućnosti da korišćenjem ovih tehnologija ostvaruju veći profit i investiraju ga u nove tehnologije. Na taj način oni uspevaju da ostanu na vrhu tehnološkog talasa. S druge strane, ljudi koji ne koriste IKT uobičajeno ostvaruju manji profit i imaju manje prostora za investiranje radi početka korišćenja IKT. Na taj način se digitalni jaz koji postoji između ove dve grupe i dalje širi.

Faktori koji bitno utiču na digitalnu podelu su:

- ekonomska situacija
- obrazovanje
- radno mesto
- kognitivne sposobnosti
- socijalni faktori.

Pored ovih postoje i drugi faktori, kao što su geografska pozicija, nacionalnost, rasa, vera itd.

UTICAJ EKONOMSKE SITUACIJE NA DIGITALNU PODELU

Ekonomska situacija je jedan od primarnih faktora digitalne podele

Ekonomska situacija je jedan od primarnih faktora digitalne podele. Da bi se IKT koristile potrebno je posedovati opremu kao što je računar sa potrebnim softverom, telefon ili televizor. Pored toga potrebno je platiti i komunikacione usluge (telefon, Internet). U mnogim zemljama vlada siromaštvo tako da čak i zaposlene osobe ne mogu da prikupe sredstva za nabavku opreme. Iako je cena računarske i komunikacione opreme tokom godina stalno padala, primanja stanovništva u mnogim zemljama i dalje nisu na takvom nivou da se mogu odvojiti sredstva za nabavku ove opreme. U nekim zemljama su shvatili da investiranje u uvođenje IKT ne doprinosi samo smanjenju digitalnog jaza, nego posredno dovodi do popravljanje ekonomske situacije u društvu. Jedan od načina da se pomogne stanovništvu da prevaziđe digitalni jaz je da se država odrekne uvoznih carina i poreza na promet. Stanovništvo sa povećanjem svojih prihoda postaje sposobno da investira u nabavku IKT za korišćenje u domaćinstvu, tako da su i članovi porodice u prilici da ih koriste.

UTICAJ OBRAZOVANJA NA DIGITALNU PODELU

Osobe koje rade na radnim mestima na kojima se uvode IKT ulaze u programe obuke za korišćenje uređaja ili aplikacija.

Obrazovanje je takođe važan faktor digitalne podele. Jasno je da osobe bez ikakvog obrazovanja ne mogu da koriste IKT. Nepismene osobe ne mogu, na primer, ni da unose podatke ili čitaju informacije koje im računar prikazuje. Međutim, i osobe koje imaju obrazovanje često ne mogu da koriste IKT.

Najčešći razlozi za ovo su:

- **Obrazovanje nije obuhvatilo IKT.** Danas je u većini razvijenih zemalja normalno da se IKT uče još u osnovnoj školi. Međutim, mnogo je osoba koje su školovane u doba kada su IKT bile u povoju tako da im njihov sistem obrazovanja nije pružio znanja i veštine iz oblasti IKT. U razvijenim zemljama se radi uglavnom o osobama srednje i starije dobi, dok se u nerazvijenim zemljama ovo i danas odnosi na gotovo celokupno stanovništvo.
- **Jezička barijera.** Većina interfejsa čovek-mašina se radi na engleskom jeziku. Što se tiče računarske i telekomunikacione opreme proizvođači sve češće obezbeđuju interfejse i za masovno korišćene svetske jezike kao što su nemački, francuski, španski, portugalski,

arapski i kineski. Ako osoba ne poznaje jezik kojim se vrši interakcija sa IKT opremom onda neće ili će moći vrlo teško da je koristi. Osobe koje kao drugi jezik koriste jezik interfejsa koji oprema nudi takođe mogu da imaju poteškoće u interakciji. U slučajevima kada interfejs koristi termine žargona, čak i osobe kojima je to maternji jezik mogu da imaju probleme u interakciji.

- **Radno mesto** može da bude još jedan faktor koji utiče na digitalnu podelu. Osobe koje rade na radnim mestima na kojima se uvode IKT ulaze u programe obuke za korišćenje uređaja ili aplikacija. Na taj način ove osobe se osposobljavaju da koriste nove tehnologije čime se smanjuje digitalni jaz.

UTICAJ KONGITIVNIH SPOSOBNOSTI NA DIGITALNU PODELU

Osobe sa posebnim potrebama su u mnogim slučajevima sprečene da koriste neke od IKT tehnologija

Kognitivne sposobnosti osoba mogu bitno da utiču na ograničenje mogućnosti da se koriste IKT. Osobe sa posebnim potrebama su u mnogim slučajevima sprečene da koriste neke od ovih tehnologija. Na primer, slepe osobe ne mogu da gledaju televizijski program i vrlo teško mogu da koriste veb za informisanje. Stare osobe se vrlo teško prilagođavaju i teško prihvataju nove tehnologije. Tipičan primer je mobilna telefonija. Primećeno je da starije osobe teško shvataju način funkcionisanja mobilnog telefona i ne uspevaju da upotrebljavaju korisnički interfejs.

SOCIJALNI UTICAJ NA DIGITALNU PODELU

Prosperitet društva i pojedinaca u velikoj meri zavisi od korišćenja IKT

Socijalni faktor takođe može da utiče na digitalne podele. U nekim zemljama stanovništvo ili delovima stanovništva se uskraćuje pristup vebu. Razlozi mogu da budu politički, religiozni ili kulturno-istorijski.

Prosperitet društva i pojedinaca u velikoj meri zavisi od korišćenja IKT. Zbog toga društvo, država i pojedinci treba da rade na smanjenju digitalnog jaza. Svim članovima društva treba da bude omogućeno da koriste IKT. Prvi korak ka tome je razumevanje ekonomskih, socijalnih, kulturnih i psiholoških barijera koji doprinose digitalnoj podeli. Naredni korak je omogućiti svim članovima društva da pod istim uslovima mogu da imaju pristup IKT. I na kraju, treba omogućiti svim članovima društva da mogu i znaju da koriste IKT. Na taj način, prelaskom iz stanja digitalne nejednakosti u stanje digitalne jednakosti, može se prevazići digitalni jaz.

UTICAJ RAČUNARSTVA NA ZDRAVSTVO

Uticaj računarstva na zdravstvo počeo je uvođenjem digitalnih zdravstvenih kartona

Uticaj računarstva na **zdravstvo** počeo je uvođenjem digitalnih zdravstvenih kartona. Umesto čuvanja podataka o pacijentima u papirnom obliku prešlo se na čuvanje informacija

u bazama podataka. Ovo je dramatično povećalo efikasnost rada zdravstvenih ustanova. Umrežavanjem zdravstvenih institucija i pristupom bazama podataka sa udaljenih lokacija medicinsko osoblje je u prilici da uvek o pacijentu dobije relevantne podatke vezane za dijagnozu, istoriju bolesti i primenjene terapije. Pri ovome se koriste međunarodni standardi za razmenu zdravstvenih podataka tako da međudržavne granice ne predstavljaju barijeru.

Korišćenje veb tehnologija omogućeno je poboljšano informisanje stanovništva o zdravom životu i prevenciji bolesti. Pored toga, olakšano je upozoravanje, nadziranje i kontrolisanje širenje lako prenosivih bolesti.

Jedan od najočiglednijih uticaja IKT na medicinu je primer telemedicine, odnosno lečenja na daljinu. Na prenos zdravstvenih informacija ne utiče razdaljina između lekara i pacijenta, niti odrediše pružaoca usluge ili opreme. Zahvaljujući telemedicini udaljenom pacijentu moguće je pružiti čitav niz usluga kao što su:

- Teledijagnostika omogućuje da lekar koji je udaljen od pacijenta može da postavi dijagnozu.
- Telenadzor omogućava nadzor fizioloških parametara pacijenata izvan zdravstvene ustanove.
- Telekonzilijum omogućava da nekoliko medicinskih stručnjaka s različitih lokacija stupe u vezu putem konferencijskog poziva i razmene mišljenje o pacijentu.

Medicinskim radnicima su takođe na raspolaganju usluge telemedicine:

- Teleobrazovanje i teleobuka imaju za cilj obučavanje medicinskog osoblja.
- Telekonsultacije omogućavaju daljinski pristup informacijama koje se čuvaju u bazama znanja ili putem kontakata sa specijalistima.

▼ Poglavlje 2

Bioinformatika

ŠTA JE TO BIOINFORMATIKA?

U najširem smislu bioinformatika je nauka koja uz pomoć informacionih tehnologija, matematike i statistike rešava biološke probleme

U najširem smislu **bioinformatika** je nauka koja uz pomoć informacionih tehnologija, matematike i statistike rešava biološke probleme. Bioinformatika se može definisati i kao primena informacionih tehnologija u obradi i analizi bioloških podataka. Centralni zadatak bioinformatike jeste racionalizacija ogromnog broja podataka o biološkim sekvencama. Potreba da se ti podaci prevedu u biološki značajne informacije, i da se pročitaju strukturni, funkcionalni i evolucijski podaci kodirani u jeziku biološke sekvence postaje sve veća sa njihovim nagomilavanjem. Najveći problem bioinformatike je dešifrovanje potpuno novog jezika sastavljenog od svega 4 slova, ali u kome promena samo jednog slova može imati za posledicu razliku u informaciji što vodi sintezi neadekvatnog proteina.

Pre nego se pređe na opis aplikacija u oblasti bioinformatike daće se definicija osnovnih pojmoveva i skraćenica koji se koriste u bioinformatici.

OSNOVNI POJMOVI

DNK, mRNK, kodon, translacija

DNK - dezoksiribonukleinska kiselina.

mRNK - molekul informacione ribonukleinske kiseline.

Gen je određeni deo DNK koji ima svoj početak i kraj i koji nosi informaciju za sintezu tačno određenog proteina. Informacija za sintezu proteina zavisi od redosleda baza u jednom od lanaca DNK. Taj redosred se procesom transkripcije kopira na manji molekul ribonukleinske kiseline (mRNK) koji tu informaciju prenosi do mesta sinteze proteina.

Transkripcija je proces prepisivanja informacije sa DNK na manji molekul informacione ribonukleinske kiseline. Tako prenesena genetička informacija služi kao matrica za sintezu proteina.

Kodon je specifična kombinacija tri azotne baze (triplet) koja nosi kombinaciju za sintezu određene aminokiseline. Svaka specifična kombinacija od tri baze (triplet ili kodon) određuje pojedinu aminokiselinu u proteinu. Ovaj genetički kod isti je kod svih živih bića. Postoji 64 kodona od kojih svaki ima odgovarajuću funkciju. Od ovih 64 kodona 3 su stop kodoni (UAA, UAG, UGA) koji služe kao **stop-signal** i ukazuju protein-sintetizujućem kompleksu da

sintezu proteina treba završiti. Ovim kodonima ne odgovara ni jedna aminokiselina. Svi ostali kodoni (njih 61) kodiraju odgovarajuće aminokiseline, odnosno jednoj aminokiselini odgovara nekoliko kodona.

Translacija, odnosno **sinteza proteina** je prevođenje informacija sadržanih u kodonima u odgovarajuće aminokiseline. Ukoliko dođe do promene u redosledu baza u DNK (mutacija), dolazi do promene u mRNK i redosledu aminokiselina u odgovarajućem proteinu, što u velikom broju slučajeva utiče na njegovu aktivnost. Izmenjena aktivnost jednog jedinog proteina može dovesti do velikih poremećaja u organizmu. Smatra se da je za više od pet hiljada različitih oboljenja, uključujući i različite oblike kancera, odgovoran nedostatak ili promena u nekom od ćelijskih proteina. Sinteza proteina ili translacija se u ćeliji odvija na ribozomima i u njoj važnu ulogu imaju molekuli RNK.

Celokupna nasledna informacija jednog organizma sadržana je u genima, odnosno u molekulu DNK. Protok informacija kroz ćeliju je usmeren tako da se informacije sadržane u DNK preslikavaju posredstvom RNK u strukturu proteina, koji obavljaju skoro sve funkcije neophodne za održavanje ćelije u životu. Ovo se još naziva i centralnom dogmom molekularne biologije.



Slika 2.1.1 Centralna dogma bioinformatike

STRUKTURA DNK

Molekul DNK se sastoji od dva dugačka prepletena lanca sastavljena od fosfata i šećera dezoksiriboze

Struktura proteina određena je strukturom DNK, a jedan od osnovnih i početnih zadataka molekularne biologije bio je upravo da objasni na koji se način struktura DNK prevodi u strukturu proteina.

Molekul DNK se sastoji od dva dugačka prepletena lanca sastavljena od fosfata i šećera dezoksiriboze. Za svaki molekul šećera vezana je jedna od četiri azotne baze - **adenin (A)**, **guanin (G)**, **citozin (C)** ili **timin (T)**.

Preko tih baza vodoničnim vezama su povezana dva naspramna lanca, pri čemu se uvek vrši vezivanje adenina i timina odnosno citozina i guanina. Samim tim, redosled baza u jednom lancu u potpunosti zavisi od redosleda baza u drugom, komplementarnom lancu. Ovo svojstvo je od velikog značaja prilikom ćelijske deobe i prenošenja naslednjog materijala, pošto omogućava prenošenje identičnih kopija DNK u ćerke ćelije.

Pre deobe ćelije dolazi prvo do procesa replikacije DNK. Prilikom replikacije DNK dolazi do raskidanja vodoničnih veza i odvajanja komplementarnih lanaca. Svaki pojedinačni lanac onda služi kao kalup za sintezu dvolančane DNK u kojoj se opet nasuprot guanina ugradjuje

citozin, a nasuprot adenina timin (proces replikacije DNK) i vrši povezivanje vodoničnim vezama.

Na taj način nastaju dva identična molekula DNK, koji nose identične genetske informacije, od kojih svaki odlazi u drugu ćerku ćeliju. U jedru svake ljudske ćelije nalazi se molekul DNK gusto zbijen i podeljen u 23 para hromozoma, od kojih jedna polovina potiče od oca a druga od majke, čineći zajedno ljudski genom.



Slika 2.1.2 Replikacija DNK - adenin (A), guanin (G), citozin (C), timin (T)

VIDEO

Bioinformatics -- Understanding of living systems through information science

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ 2.1 Bioinformatičke aplikacije

BIOTEHNOLOGIJA

Rezultati biotehnologije se primenjuju u raznim oblastima od novih vakcina do metoda za dijagnostiku

Iz svega što je rečeno je očigledno da je biotehnologija izuzetno značajna grana nauke za čovečanstvo i da je izuzetno kompleksna. Rezultati biotehnologije se primenjuju za:

- Proizvodnju novih proteina
- Proizvodnju genetički modifikovane hrane
- Proizvodnju novih vakcina u veterini, kao i bioloških sredstava protiv korova i štetočina
- Dobijanje novih lekova i razvijanja novih terapija u skladu sa genetskim kodom pojedinca
- Za dijagnostiku
- Za otkrivanje naslednih poremećaja, sprečavanje nastanka bolesti i lečenje bolesti.

Zbog toga su razvijene mnoge aplikacije koje uglavnom imaju zadatak:

- Da omoguće unos podataka o novim biološkim sekvencama u bazu podataka
- Da pretražuju baze podataka sa biološkim sekvencama.

✓ 2.2 Medicinske aplikacije

PRIMER EKSPERTSKE APLIKACIJE U MEDICINI

Ovde će se prikazati samo jedna domaća aplikacija, Medicas, koja predstavlja opšti informacioni sistem namenjen zdravstvenim ustanovama

Medicina kao ogromna oblast koristi veliki broj aplikacija koje su namenjene automatizaciji pojedinih procesa rada. Ovde će se prikazati samo jedna domaća aplikacija, Medicas, koja predstavlja opšti informacioni sistem namenjen zdravstvenim ustanovama. Namena ovog sistema je da bude univerzalni alat za lekare i administraciju u zdravstvenim ustanovama. Pored toga Medicas na osnovu podataka skupljenih u svakodnevnom radu može da generiše različite izveštaje, za lekare, RZZO, ZZZZ, Ministarstvo zdravlja, Nacionalni i regionalne zdravstveno-informacione centre.



Slika 2.2.1 Segmenti ekspertske aplikacije

Projektovanje jednog ovakvog informacionog sistema je zahtevalo da se razmotre zahtevi i iz zdravstvene ustanove i okruženja. Zahvaljujući optimizaciji radnih tokova korišćenjem jednog ovakvog sistema se eliminišu prazni hodovi, a rukovodstvu olakšava rukovođenje ustanovom.

Ministarstvo zdravlja, kao i drugi republički organi finansiraju zdravstvene ustanove pa je neophodno ostvariti komunikaciju. U ovom slučaju zdravstvena ustanova izveštava Ministarstvo, odnosno Republičku zajednicu zdravstvenog osiguranja o pruženim uslugama pacijentima kako bi ih naplatilo. Takođe, zdravstvena ustanova će imati obavezu da izveštava i Nacionalne i Regionalne zdravstvenoinformacione centre (NZIC, RZIC). Baza podataka nad kojom radi Medicas ima veliki broj tabela u kojima se nalaze svi potrebni podaci o pacijentima, procesima i resursima zdravstvene ustanove. Ovde se navode samo neke, najvažnije tabele:

- Pacijenti
- Lekari i osoblje
- Ordinacije i odeljenja
- Registrovani lekovi (Jedinstvena knjiga lekova)
- Međunarodna klasifikacija bolesti (MKB-10)
- Zdravstvene usluge

FUNKCIONALNOSTI MEDICAS-A

Moduli, funkcije i izveštaji

Pored opšteg dela koji je zajednički za rad svih zdravstvenih ustanova, Medicas ima mogućnost dodavanja pojedinih modula koji služe za podršku radu specijalističkih službi. Na raspolaganju su sledeći moduli:

- Medicas GP - Opšta praksa
- Medicas Hosp - Stacionari
- Medicas Poli - Poliklinike
- Medicas Dent - Stomatolozi

Funkcionalnost programa je prilagođena standardnim poslovnim procesima u zdravstvenim ustanovama i pokriva sve aktivnosti kao što su prijema pacijenta, dijagnoza, terapija, otpust, praćenje itd. Pored toga program ima i neke napredne funkcije kao što su:

- Rezervacije poseta i smeštaja u ustanovama
- Izbor lekara
- Paketi osiguranja (RZZO, POZU, ostala osiguranja)
- Vakcinacije
- Opšti i posebni programi pacijenata
- Standardnu listu usluge uz ordinacije
- Praćenje bolničke ili priručne apoteke
- Bolnički postupci.

Svi podaci o pacijentu se čuvaju u bazi podataka na osnovu koje se mogu prikazati ili odštampati različiti **izveštaji o pacijentu** kao što su:

- Elektronski karton pacijenta (EHR)

- Medicinski status, istorija pacijenta
- Uložak zdravstvenog kartona – evidencioni list posete
- Istorija bolesti
- Etapna epikriza
- Otpusna lista sa epikrizom
- Uputi, recepti, nalozi, izveštaj specijaliste
- Laboratorijski izveštaji, medicinski snimci
- Dozname i potvrde.

DRUGI IZVEŠTAJI MEDICAS-A

Svi izveštaji su u standardnim formama koje su propisane medicinskim zakonima i uredbama.

Korišćenjem ovakvih izveštaja medicinsko osoblje radi i donosi odluke na osnovu kvalitetnih podataka, a pacijentu se pruža bolja usluga. Pored izveštaja o pacijentima, Medikas pruža i druge izveštaje kao što su:

- Statistika i izveštaji za potrebe lekara
- Statistika i izveštaji na nivo ustanove
- Lista morbiditeta, prijave zaraznih i hroničnih bolesti
- Izveštaji za RZZO i Ministarstvo zdravlja u XML formatu

Svi izveštaji su u standardnim formama koje su propisane medicinskim zakonima i uredbama.

✓ Poglavlje 3

E-Commerce

ŠTA PREDSTAVLJA ECOMMERCE?

Elektronska trgovina je oblast elektronskog poslovanja (e-Business) koja se, koristeći Internet kao infrastrukturu, bavi prodajom, distribucijom, kupovinom i marketiranjem proizvoda i usluga

Elektronska trgovina je oblast elektronskog poslovanja (e-Business) koja se, koristeći Internet kao infrastrukturu, bavi prodajom, distribucijom, kupovinom i marketiranjem proizvoda i usluga. Elektronska trgovina postaje sve značajniji kanal prodaje.

Elektronska trgovina obuhvata:

- Elektronsko plaćanje, odnosno transfer novca
- Upravljanje lancem snabdevanja
- Elektronski marketing
- Elektronsku razmenu podataka
- Sistem za praćenje zaliha
- Sistem za automatizovano prikupljanje podataka

Počeci elektronske trgovine datiraju iz kasnih sedamdesetih godina prošlog veka, kada je počela da se koristi tehnologija elektronske razmene podataka (engleski EDI – **Electronic Data Interchange**) za slanje poslovnih dokumenata kao što su zahtevi za ponudom i ponude.

Sa razvojem veba su se proširile mogućnosti elektronske trgovine pa se ona širi iz sfere **B2B** (**business to business**) i na sferu **B2C** (**business to customers**) i **C2C** (**customer to customer**). Ovo je bilo podržano bezbednosnim protokolima, kao što je HTTPS (**Hyper Text Transport Protocol Secure**), i sistemima za elektronsko plaćanje.

Osnova za elektronsku trgovinu je veb sajt koji predstavlja ulaznu tačku u proces trgovine. Razlikuju se više različitih osnovnih modela, mada su neki veb sajтовi mešavina različitih pristupa. Svaki osnovni model ima jedinstvene karakteristike i poslovni model koji ga razlikuje od drugih.

Sajt na principu brošure (**brochureware site**). Ovo je u principu sajt čija je osnovna namena marketing, pa kao takav samo predstavlja elektronski oblik podrške procesima kupovine i prodaje. Tipično ovakav sajt sadrži podatke o proizvodima ili uslugama, tehničku podršku, odgovore na često postavljana pitanja i informacije za kontakt. Sve transakcije se obavljaju

klasičnim putem, tako da strogo uzevši ovakav pristup ne pripada elektronskoj trgovini. Za formiranje ovakvog sajta koristi se tipičan content management system.

B2C

Online prodavnica je veb sajt na kome korisnici kupuju proizvode ili usluge

Online prodavnica je veb sajt na kome korisnici kupuju proizvode ili usluge. Ovakvi sajtovi se nazivaju **e-commerce** ili **B2C sajtovi**. Sadržaj koji se objavljuje na ovakovom sajtu je isti kao i na sajtu na principu brošure ali ima i detaljne informacije o proizvodima kao što su tehnička specifikacija, cena, rok isporuke, način transporta i mišljenje kupaca o pojedinim proizvodima. Pored toga stranice sadrže opis metoda i sam metod za online naručivanje. Stranice se dinamički generišu na osnovu informacija iz baze podataka, tako da su uvek ažurne. Poseban deo sajta čine stranice sa opisom metode plaćanja. Plaćanje ne mora uvek da se obavlja elektronskim putem. Moguće je, na primer, plaćanje prilikom dostave. Ukoliko se vrši elektronsko plaćanje postoje i posebne stranice sa formama za plaćanje iz kojih je ceo sistem za elektronsko plaćanje. Sistem za elektronsko plaćanje obezbeđuje bezbedan, pouzdan i jeftin način za autorizovano plaćanje i upravljanje transakcijama koje slede nakon toga. Najbolji sistemi su bazirani na tehnologijama **Secure Socket Layer (SSL)** i/ili **Secure Electronic Transactions (SET)** koje obezbeđuju kriptovanje podataka i nakon transakcije generišu i prikazuju korisniku potvrdu o plaćanju. Primer domaćeg B2C sajta je sajt kompanije gigatron koja se bavi prodajom računarske opreme (<http://www.gigatronshop.com/>).

Podaci o narudžbinama se smeštaju u bazu podataka kako bi bili na raspolaganju za dalju obradu u drugim aplikacijama, a mogu i automatski da se prosleđuju drugim odeljenjima kao što je, na primer, odeljenju za isporuku.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

B2B

B2B predstavlja sajt sa ponudom gde su posetnici umesto fizičkih, pravna lica koja imaju pregled ponude i mogućnost poručivanja proizvoda

B2B predstavlja sajt sa ponudom gde su posetnici umesto fizičkih, pravna lica koja imaju pregled ponude i mogućnost poručivanja proizvoda. B2B se povezuje sa informacionim sistemom iz koga dobija sve potrebne podatke o lageru, stanju, dokumentima, cenama, konto karticama itd. Prednost ovakvog načina poslovanja se ogleda u tome što će se sve promene prikazati u realnom vremenu kroz B2B portal i mreža poslovnih saradnika će moći u bilo kom trenutku da ima pristup ponudi. Jednom re;ju odnosi se na korišćenje Interneta i Web tehnologija za kupovinu, prodaju, jeftiniju i bržu saradnju polsovnih subjekata.

Prednosti B2B su:

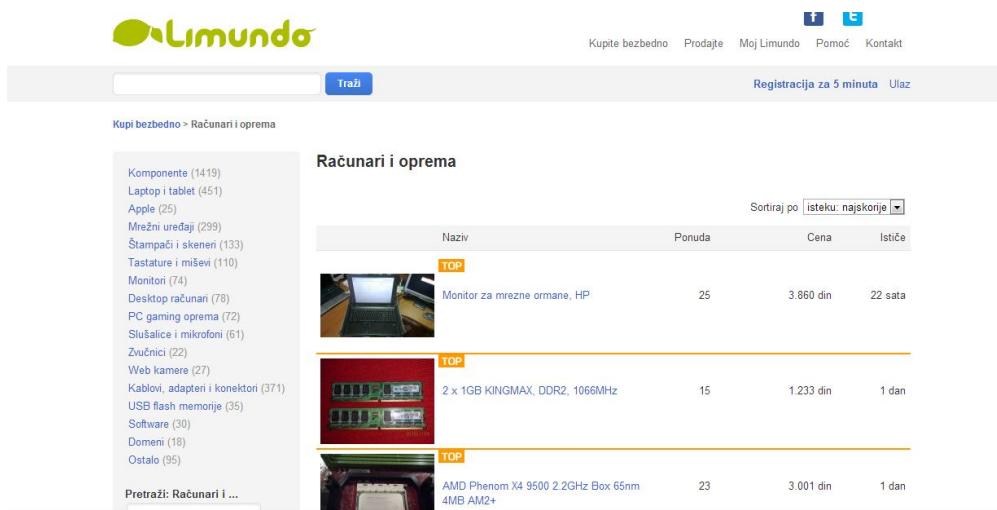
- Smanjenje troškova
- Povećavanje transparentnosti poslovanja
- Mogućnost pristupa novim tržištima
- Efikasnije i fleksibilnije transakcije.

Prve kompanije koje su koristile ovakav način su: Fedex, Cisko, Dell itd.

C2C

C2C (Customers to Customers) odnosi se na trgovanje potrošača sa drugim potrošačima

C2C (Customers to Customers) odnosi se na trgovanje potrošača sa drugim potrošačima. Ta usluga se obavlja preko web portala i uključen je veliki broj korisnika. Posrednici obično preko neki vid članarine naplaćuju svoju posredničku ulogu. Jedan od primera takvog sajta je limundo i kupindo (www.limundo.com, www.kupindo.com). Razlika između navedenih sajtova je što u prvi se kupovina izvršava uz pomoć aukcije, dok na kupindu se postaljaju produkti i moguće ih je odmah kupiti.



Naziv	Ponuda	Cena	Istiće
Monitor za mrežne ormane, HP	25	3.860 din	22 sata
2 x 1GB KINGMAX, DDR2, 1066MHz	15	1.233 din	1 dan
AMD Phenom X4 9500 2.2GHz Box 65nm 4MB AM2+	23	3.001 din	1 dan

Slika 3.1 Izgled limundo sajta

E-COMMERCE VIDEO

What is E-commerce? B2B and B2C

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 4

Bibliotečki sistem

NIBIS ARHITEKTURA I FUNKCIONALNOSTI

NIBIS je programski sistem namenjen automatizaciji poslovanja malih i velikih biblioteka

Bibliotečki informacioni sistemi imaju dve glavne funkcije. Sa aspekta korisnika, oni treba da obezbede računarsko korišćenje bibliotečkih resursa, a sa aspekta biblioteke, računarski podržano upravljanje resursima biblioteke. Treba imati na umu da bibliotečki informacioni sistem treba da podržava različite vrste građe. Pored naručištenih knjiga i periodičnih izdanja, biblioteke čuvaju i geografske karte, muzičke i video zapise, mikrofilmove, pisma, slike i fotografije. Zbog toga je bibliografski opis ovakve građe izuzetno složen i opisan različitim standardima kao što su UNIMARK, ISBD i drugi.

Ovde će se kao primer bibliotečkog informacionog sistema opisati program NIBIS. **NIBIS** je programski sistem namenjen automatizaciji poslovanja malih i velikih biblioteka. Svojom funkcionalnošću NIBIS zadovoljava sve potrebe kako administratora, tako i korisnika biblioteke. Pored toga, ovaj sistem omogućuje saradnju sa drugim bibliotečkim informacionim sistemima korišćenjem Interneta.



Slika 4.1 Izgled NIBIS-a

OSNOVNE OSOBINE NIBIS-A

Pregled osnovnih osobina NIBIS-a

Osnovne osobine NIBIS-a su:

- Jednostavan za obuku i korišćenje
- Ne zahteva skupu računarsku opremu
- Moguća je instalacija na jednom PC računaru ili na većem broju umreženih računara
- WEB orijentisana aplikacija
- Omogućen je pristup iz lokalne računarske mreže ili sa bilo koje tačke na Internetu
- Modularnost sistema omogućava krojena rešenja za male i velike biblioteke
- Potpuno podržava bibliotečke standarde kao što su UNIMARC i ISBD.

ADMINISTRACIJA KLIJENATA

Sistem administracije klijenata automatizuje jednu od najvažnijih lokalnih funkcija biblioteke, a to je iznajmljivanje bibliotečkog materijala korisnicima biblioteke

Sistem administracije klijenata automatizuje jednu od najvažnijih lokalnih funkcija biblioteke, a to je iznajmljivanje bibliotečkog materijala korisnicima biblioteke. Ovaj sistem omogućava:

1. Unos i ažuriranje podataka o članovima biblioteke

- Identifikacija korisnika se vrši na osnovu broja članske karte, koji se u sistem može slati ručno ili preko čitača za bar kod.

2. Zaduživanje i razduživanje bibliotečkog materijala

- Identifikacija knjiga se vrši na osnovu inventarskog broja, koji se u sistem može slati ručno ili preko čitača za bar kod

3. Pregled i administriranje prava i obaveza članova biblioteke (kazne, isteklo članstvo i sl.)

OJAVA GLAVNA STRANA KORISNICI BIBLIOGRAFIJA INVENTAR POMOC																																																																	
Profil klijenta: Mikan Stanković <table border="1"> <thead> <tr> <th colspan="6">Zaduženja klijenta</th> </tr> <tr> <th>Inv. broj (novi)</th> <th>Inv. broj (stari)</th> <th>Naslov</th> <th>Izdata</th> <th>Službenik</th> <th>Vraćena</th> </tr> </thead> <tbody> <tr> <td>1-11</td> <td>0</td> <td>Mitsko u poziji Vaska Pope</td> <td>03-04-2003</td> <td>1</td> <td><input type="checkbox"/></td> </tr> <tr> <td colspan="6"> <input type="button" value="Zaduživanje"/> <input type="button" value="Razduživanje"/> <input type="button" value="Pretraživanje"/> </td> </tr> </tbody> </table> <table border="1"> <thead> <tr> <th colspan="6">Korpa klijenta</th> </tr> <tr> <th>Inv. broj</th> <th>Signatura</th> <th>Naslov</th> <th>Autor</th> <th>Izbaci</th> <th>Zaduzi</th> </tr> </thead> <tbody> <tr> <td>1-5</td> <td>5</td> <td>Retorika</td> <td>Dobrivoje Stanojević</td> <td><input type="checkbox"/></td> <td><input type="checkbox" value="Zaduzi"/></td> </tr> <tr> <td>1-16</td> <td>16</td> <td>Neizvršena oporuka</td> <td>Pjer Boško</td> <td><input type="checkbox"/></td> <td><input type="checkbox" value="Zaduzi"/></td> </tr> <tr> <td>1-19</td> <td>19</td> <td>Romuliana alkana</td> <td>Radivoj Petrović</td> <td><input type="checkbox"/></td> <td><input type="checkbox" value="Zaduzi"/></td> </tr> <tr> <td colspan="6"> <input type="button" value="Isprazni korpu"/> <input type="button" value="Izbaci iz korpe"/> </td> </tr> </tbody> </table>						Zaduženja klijenta						Inv. broj (novi)	Inv. broj (stari)	Naslov	Izdata	Službenik	Vraćena	1-11	0	Mitsko u poziji Vaska Pope	03-04-2003	1	<input type="checkbox"/>	<input type="button" value="Zaduživanje"/> <input type="button" value="Razduživanje"/> <input type="button" value="Pretraživanje"/>						Korpa klijenta						Inv. broj	Signatura	Naslov	Autor	Izbaci	Zaduzi	1-5	5	Retorika	Dobrivoje Stanojević	<input type="checkbox"/>	<input type="checkbox" value="Zaduzi"/>	1-16	16	Neizvršena oporuka	Pjer Boško	<input type="checkbox"/>	<input type="checkbox" value="Zaduzi"/>	1-19	19	Romuliana alkana	Radivoj Petrović	<input type="checkbox"/>	<input type="checkbox" value="Zaduzi"/>	<input type="button" value="Isprazni korpu"/> <input type="button" value="Izbaci iz korpe"/>					
Zaduženja klijenta																																																																	
Inv. broj (novi)	Inv. broj (stari)	Naslov	Izdata	Službenik	Vraćena																																																												
1-11	0	Mitsko u poziji Vaska Pope	03-04-2003	1	<input type="checkbox"/>																																																												
<input type="button" value="Zaduživanje"/> <input type="button" value="Razduživanje"/> <input type="button" value="Pretraživanje"/>																																																																	
Korpa klijenta																																																																	
Inv. broj	Signatura	Naslov	Autor	Izbaci	Zaduzi																																																												
1-5	5	Retorika	Dobrivoje Stanojević	<input type="checkbox"/>	<input type="checkbox" value="Zaduzi"/>																																																												
1-16	16	Neizvršena oporuka	Pjer Boško	<input type="checkbox"/>	<input type="checkbox" value="Zaduzi"/>																																																												
1-19	19	Romuliana alkana	Radivoj Petrović	<input type="checkbox"/>	<input type="checkbox" value="Zaduzi"/>																																																												
<input type="button" value="Isprazni korpu"/> <input type="button" value="Izbaci iz korpe"/>																																																																	
PROFIL KLIJENTA <table border="1"> <tr> <td>Broj članske karte: 1</td> </tr> <tr> <td>Matični broj: 1234567890123</td> </tr> <tr> <td>Ime (*)</td> <td>Mikan</td> </tr> <tr> <td>Prezime (*)</td> <td>Stanković</td> </tr> <tr> <td>Broj lične karte (*)</td> <td>12345</td> </tr> <tr> <td>Izdata od (*)</td> <td>sed</td> </tr> <tr> <td>Vlasnik lične karte</td> <td>Da <input checked="" type="radio"/> Ne <input type="radio"/></td> </tr> <tr> <td>Ime roditelja</td> <td>Kg</td> </tr> <tr> <td>Datum rođenja (dan, mjesec, godina)</td> <td>12 Februar 1987</td> </tr> <tr> <td>Zanimanje (*)</td> <td>stolar</td> </tr> <tr> <td>Kategorija korisnika (*)</td> <td>pratek</td> </tr> <tr> <td colspan="2">Adresa stalnog boravka</td> </tr> <tr> <td>Poštanski broj</td> <td><input type="text"/></td> </tr> <tr> <td>Mesto</td> <td><input type="text"/></td> </tr> <tr> <td>Ulica i broj</td> <td><input type="text"/></td> </tr> <tr> <td colspan="2">Adresa privremenog boravka</td> </tr> </table>						Broj članske karte: 1	Matični broj: 1234567890123	Ime (*)	Mikan	Prezime (*)	Stanković	Broj lične karte (*)	12345	Izdata od (*)	sed	Vlasnik lične karte	Da <input checked="" type="radio"/> Ne <input type="radio"/>	Ime roditelja	Kg	Datum rođenja (dan, mjesec, godina)	12 Februar 1987	Zanimanje (*)	stolar	Kategorija korisnika (*)	pratek	Adresa stalnog boravka		Poštanski broj	<input type="text"/>	Mesto	<input type="text"/>	Ulica i broj	<input type="text"/>	Adresa privremenog boravka																															
Broj članske karte: 1																																																																	
Matični broj: 1234567890123																																																																	
Ime (*)	Mikan																																																																
Prezime (*)	Stanković																																																																
Broj lične karte (*)	12345																																																																
Izdata od (*)	sed																																																																
Vlasnik lične karte	Da <input checked="" type="radio"/> Ne <input type="radio"/>																																																																
Ime roditelja	Kg																																																																
Datum rođenja (dan, mjesec, godina)	12 Februar 1987																																																																
Zanimanje (*)	stolar																																																																
Kategorija korisnika (*)	pratek																																																																
Adresa stalnog boravka																																																																	
Poštanski broj	<input type="text"/>																																																																
Mesto	<input type="text"/>																																																																
Ulica i broj	<input type="text"/>																																																																
Adresa privremenog boravka																																																																	

Slika 4.2 Administracija klijenta u NIBIS-u

ADMINISTRACIJA INVENTARA

NIBIS bibliotekarima omogućava da u potpunosti automatizuju proces inventarisanja knjiga koje se nabavljaju.

NIBIS bibliotekarima omogućava da u potpunosti automatizuju proces inventarisanja knjiga koje se nabavljaju.

Posle zvaničnog prijema novih knjiga u biblioteku, sledi proces kataloške obrade istih. Sistem omogućava proveru da li su te knjige već nekada ranije kataloški obrađene, tako da nema dupliranja zapisa.

Na osnovu obrađenih kataloških jedinica se kasnije vrši inventarisanje pojedinih primeraka knjiga, njihova raspodela po odeljenjima biblioteke i sl.



Slika 4.3 NIBIS administracija inventara

BIOGRAFSKA OBRADA

NIBIS omogućava potpunu automatizaciju aktivnosti vezanih za katalogizaciju bibliotečkih jedinica

NIBIS omogućava potpunu automatizaciju aktivnosti vezanih za katalogizaciju bibliotečkih jedinica. Prilikom obrade se u potpunosti poštaju važeći standardi iz ove oblasti. Tu se pre svega misli na standard UNIMARC, koji je osnova iz koje se izvode svi danas korišćeni standardi u svetu vezani za katalogizaciju.

UNIMARC standard je opšti i sveobuhvatan, i nekim bibliotekama nisu potrebni svi podaci koje ovaj standard može da pokrije. Zbog toga je kataloška obrada bibliografskih jedinica u NIBIS-u tako organizovana da je moguć unos samo onih podataka koji su od interesa za određenu biblioteku. Unos se vrši putem sistema kartica (posebnih ulaznih maski) kojima se može pristupati proizvoljnim redosledom. Na karticama su grupisani srodni podaci. Biblioteka je ta koja odlučuje koji podaci, odnosno koje kartice će se koristiti. Na taj način je uspostavljena

prilagodljivost sistema (krojena rešenja), što znači da se podjednako dobro može primeniti u velikim, ali i malim bibliotekama.

Slika 4.4 Biografska obrada

KORISNIČKI SERVIS

Korisnici imaju mogućnost da pretražuju bazu podataka sa bibliotečkim materijalom.

Korisnici imaju mogućnost da pretražuju bazu podataka sa bibliotečkim materijalom. Moguće su dve vrste pretraživanja:

1. Pretraživanje od strane anonimnih korisnika

- Bilo koji korisnik može da pristupi serveru biblioteke (iz lokalne mreže ili sa Interneta) i da pogleda da li u toj biblioteci postoji knjiga koja ga zanima. Moguće su različite vrste pretraživanja (po autoru, naslovu, ključnim rečima, različitim odrednicama itd.) Rezultat pretraživanja su knjige koje zadovoljavaju zadate uslove. O svakoj knjizi se mogu dobiti detaljni bibliografski podaci.

2. Pretraživanje od strane prijavljenih korisnika

- Svaki član biblioteke prilikom upisa, pored članske karte, dobija svoje korisničko ime i lozinku. Pomoću tog imena i lozinke se kasnije u sistemu vrši njegova identifikacija. Postupak pretraživanja za prijavlenog korisnika je isti kao i za anonimnog. Razlika je u tome što korisnik koji je član biblioteke može da knjigu koju je pronašao smesti u svoju korpu. Ta korpa sadrži sve knjige za koje je on zainteresovan. Kasnije se iznajmljivanje knjiga može vršiti na bazi spiska knjiga koje se nalaze u njegovoj korpi.



Slika 4.5 Korisnički servis

ARHITEKTURA SISTEMA I RADNA PLATFORMA

Za rad sistema su potrebni Web server, server baze podataka i niz računara klijenata. Web server i server baze podataka se mogu fizički nalaziti na istom računaru

Arhitektura sistema

NIBIS funkcioniše kao Web aplikacija. Ovakve aplikacije obično imaju troslojnu arhitekturu, kakva postoji i u NIBIS-u. Prvi sloj služi za prezentaciju i to je u suštini korisnički interfejs, koji je u ovom slučaju realizovan u vidu veb strana, pisanih u HTML jeziku. Drugi nivo, nivo poslovne logike, je realizovan u programskom jeziku Java. Na ovom nivou su realizovane sve specifičnosti vezane za bibliotečko poslovanje. Treći nivo je nivo baze podataka u kojoj se nalaze sve informacije vezane za poslovanje biblioteke.

Radna platforma

Za rad sistema su potrebni Web server, server baze podataka i niz računara klijenata. Web server i server baze podataka se mogu fizički nalaziti na istom računaru. Broj mašina klijenata nije ograničen i zavisi isključivo od potreba i mogućnosti same biblioteke.

To što je NIBIS razvijen kao Web aplikacija omogućava njegovu nezavisnost od platforme i operativnog sistema koji administratori ili klijenti koriste. Konkretno to znači da NIBIS može da radi na operativnom sistemu Windows, koji je danas najčešći operativni sistem, ali se bez ikakvih problema može implementirati i na Linux (Unix) platformi. Mogućnost rada na Linuxu je posebno interesantna, jer je u pitanju besplatni operativni sistem, što može da značajno utiče na smanjenje troškova poslovanja same biblioteke.

Računari klijenti (koji se povezuju sa serverom i na kojima rade korisnici) treba da imaju bilo koji Web browser, tako da to dodatno pojeftinjuje implementaciju i upotrebu sistema.

Web aplikacije su po svojoj prirodi mrežne aplikacije. Za rad NIBIS-a je potrebno da postoji lokalna mreža (preko TCP/IP protokola), ili veza sa Internetom, ako se želi rad izvan same biblioteke. Struktura aplikacije je takva da ona ne zahteva nikakva prilagođavanja bilo da je reč o radu sa samo jednog računara, koji istovremeno služi i kao server i kao klijent, bilo

da je reč o intranet mreži neke biblioteke (interna mreža u samoj biblioteci) ili o radu preko Interneta

▼ Poglavlje 5

eUprava

FUNKCIONALNOST I DIZAJN EUPRAVE

Državna uprava ima složeni zadatak da ima evidenciju o fizičkim i pravnim licima u državi, da nadgleda sprovođenje zakona i da pruža različite usluge građanima, fizičkim licima i drugim or

Državna uprava ima složeni zadatak da ima evidenciju o fizičkim i pravnim licima u državi, da nadgleda sprovođenje zakona i da pruža različite usluge građanima, fizičkim licima i drugim organizacijama državne uprave. Ovaj zadatak je složen zato što su i podaci o građanima i fizičkim licima promenljivi tokom vremena, a izvori podataka geografski raspršeni. S druge strane, zakonska regulativa se stalno menja što izaziva promenu poslovne logike, pa čak i organizacije rada. Zbog svega ovoga, državna uprava je spora i ne efikasna.

Primena informaciono-komunikacionih tehnologija (IKT) u državnoj upravi ima uticaja na funkcionisanje uprave, a time i na celo društvo. Najvažnije posledice uvođenja računarstva u državnu upravu su:

- Automatizacija poslovnih procesa koji su definisani zakonima i uredbama
- Ubrzavanje rada uprave
- Povećanje kvaliteta usluga, transparentnosti, odgovornosti i efektivnosti rada uprave
- Automatizacija razmene informacija između organa uprave
- Bolji pristup uslugama uz manje troškove za građane i privredu
- Manji troškovi rada uprave
- Skraćenje administrativnih procedura što čini bolji ambijent za dalji ekonomski razvoj
- Širenje demokratije u društvu

USLUGE E-UPRAVE

Uvođenjem informaciono-komunikacionih tehnologija u državnu upravu predstavlja preduslov za organizovanje sistema e-uprave

Uvođenjem informaciono-komunikacionih tehnologija u državnu upravu predstavlja preduslov za organizovanje sistema e-uprave (engl. -Government). Ovakva uprava može svojim

građanima da pruži niz usluga bez potrebe da oni odlaze u jedan ili više organa uprave. Ovde se navode samo neke od usluga:

- Izvodi iz matičnih knjiga: rođenih, venčanih i umrlih
- Lična dokumenta: lična karta, pasoš i vozačka dozvola
- Obaveštenje o preseljenju, odnosno o promeni adrese
- Socijalno osiguranje
- Porez: poreska prijava, obaveštenje o proceni
- Servisi traženja zaposlenja pri biroima za rad
- Registracija automobila: novih, korišćenih i uvezenih
- Dobijanje građevinske dozvole
- Prijave policiji
- Javne biblioteke: dostupnost kataloga i alata za pretragu publikacija
- Prijave na konkurs za osnovno, srednje i više obrazovanje
- Usluge u vezi sa zdravstvom: interaktivni saveti u vezi sa dostupnošću usluga u različitim bolnicama; zakazivanje pregleda u bolnicama

Neke od usluga koje e-uprava može da pruži pravnim licima su:

- Regulisanje socijalnog doprinosa za zaposlene
- Porez na dobit preduzeća: poreska prijava, obaveštenje
- Porez na dodatu vrednost (PDV): poreska prijava, obaveštenje
- Registracija novog preduzeća
- Podnošenje podataka kancelarijama koje se bave statistikom
- Carinske deklaracije
- Dozvole u vezi sa životnom sredinom, uključujući i izveštavanje
- Javna nabavka itd.

Ovako širok spektar usluga koje uprava treba da interaktivno pruži građanima i pravnim licima zahteva fleksibilnu i otvorenu koncepciju sistema.

GLAVNE KARAKTERISTIKE KONCEPTA E-UPRAVE

Pregled glavnih karakteristika koncepta e-uprave

Glavne karakteristike koncepta e-uprave su:

- Postojanje više različitih komunikacionih kanala koji omogućavaju pristup javnim uslugama, a koji se biraju od strane građana i privrede shodno njihovoj podesnosti i pristupačnosti.
- Javne usluge su organizovane prema potrebama njihovih korisnika, tj. prema životnim i poslovnim situacijama građana i privrede, a ne prema internoj organizaciji državnih organa.
- Usluge koje državni organi pružaju su potpuno međusobno integrisane, umesto da predstavljaju izolovane celine.
- Zahtevi korisnika prihvaćeni na prijemnim mestima transparentno se obrađuju u pozadini, bez obzira na broj različitih organa koje učestvuju u obradi. U pružanju svojih usluga, neki organ se može oslanjati na usluge drugih organa.
- Građanima i privredi je potrebna minimalna dokumentacija da bi podneli zahtev i zadovoljili svoje potrebe. Sve druge potrebne informacije, ukoliko su u posedu nekog državnog organa, dobijaju se komunikacijom unutar državne uprave korišćenjem integrisanih usluga.

Izvor:

Nacionalna strategija za informaciono društvo u Srbiji, Nacrt, Ministarstvo nauke i zaštite životne sredine Republike Srbije, Beograd, 2005.

POSTEPENO UVODENJE E-UPRAVE

Razume se da je uvođenje ovakvog koncepta nemoguće odjednom zato što je potrebno vreme za njegov razvoj i zato što korisnike treba privići i prilagoditi za njegovo korišćenje.

Razume se da je uvođenje ovakvog koncepta nemoguće odjednom zato što je potrebno vreme za njegov razvoj i zato što korisnike treba privići i prilagoditi za njegovo korišćenje. Predstavljeni koncept e-uprave primenjuje se postepenim napredovanjem kroz različite nivoe razvoja. Sa aspekta kvaliteta usluga koji se nude javnosti, mogu se identifikovati tri takva nivoa:

- 1.)** Dostupne su samo informacije na Internetu. Korisnicima su dostupne samo statičke informacije na Internetu bez mogućnosti elektronske interakcije s državnom upravom.
- 2.)** Omogućeni su komunikaciono-interaktivni servisi. Građanima je omogućeno da s upravom komuniciraju elektronskim putem, tj. da razmenjuju elektronske poruke ili koriste jednostavne aplikacije za međusobnu komunikaciju.
- 3.)** Primenjeni su transakciono-integracioni servisi. U potpunosti se primenjuju složene i specijalizovane višestepene transakcije između različitih državnih organa sa ciljem zadovoljenja pojedinačnih korisničkih zahteva, kao što je obrada podnesaka građana.

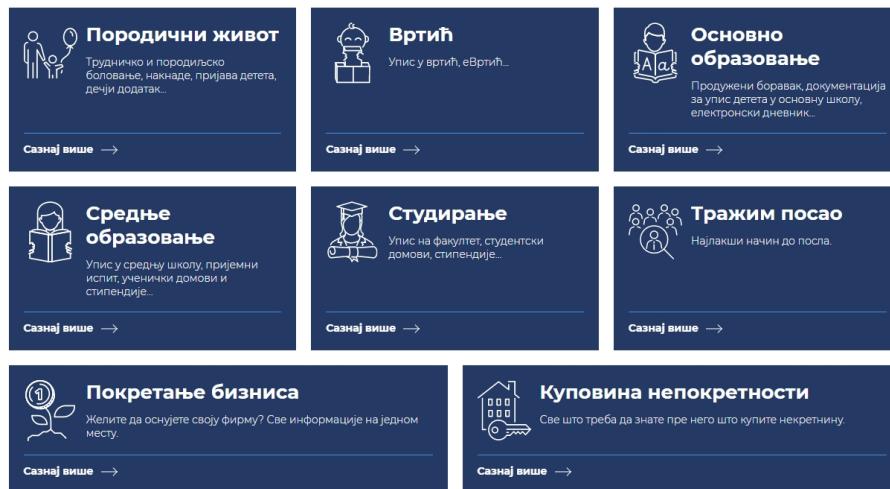
E-upravu treba realizovati poštujući sledeće osnovne principe:

- Pristup svima. Javni servisi moraju biti dostupni svim građanima. Glavni preduslov za ovo je jeftin i brz pristup Internetu.

- Sprečavanje digitalne podele. Građani sa malim prihodima i slabim tehničkim znanjem ne bi trebalo da budu diskriminisani, tj. mora se izbeći digitalni jaz među različitim socijalnim slojevima.
- Bezbednost i zaštita privatnosti. Javni servisi moraju biti bezbedni i moraju štititi privatnost građana.
- Otvoren sistem. E-uprava će se primenjivati u saglasnosti sa principima otvorenog sistema zasnovanog na otvorenim i međusobno funkcionalnim IKT rešenjima različitih proizvođača.
- Koherentnost i funkcionalna jedinstvenost. E-uprava predstavlja veoma složen, ali integriran informacioni sistem, koji funkcioniše kao jedan koherentan sistem, gde se jedinstvenost i zajedničko funkcionisanje različitih heterogenih delova postiže kroz standardizaciju i koordiniran razvoj.
- Autonomija u razvoju. Svaki državni organ ili javna organizacija može autonomno da se razvija i upravlja svojim podsistom prema prethodno dogovorenim standardima e-uprave i nacionalnom planu razvoja.
- Fleksibilna i moderna IKT rešenja. Primenjena IKT rešenja zasnivaće se na najnovijim metodološkim i tehnološkim dostignućima omogućujući produktivan razvoj i obezbeđujući fleksibilnost na buduće organizacione i tehnološke promene.

TRENUTNO STANJE E-UPRAVE

Vremenom e-Uprava dobija sve više i više aktivnih korisnika



Slika 5.1 - Prikaz dostupnih kategorija usluga



Slika 5.2 - Статистика e-Uправе

Kako vreme prolazi, sve se više i više korisnika prebacuje na online platformu kako bi realizovali usluge.

Pojavom pandemije virusa u martu 2020. godine jedan od uslova da bi se realizovale usluge je da se korisnici prijave putem e-Uprave (npr. kako bi korisnik izvadio novu ličnu, morao se prvo prijaviti putem e-Uprave kako bi rezervisao svoj termin).

Veliki broj upisa u škole se odvijao putem e-Uprave.

▼ Poglavlje 6

Uticaj ICT na grafički dizajn

PRIMENA RAČUNARA U DIZAJNU

U svim oblastima grafičkog dizajna koriste se računarski programi kao podrška kreativnom radu dizajnera

Poslovni ljudi su shvatili da iskustvo korisnika pri korišćenju nekog proizvoda u velikoj meri zavisi od dizajna. Zbog toga se danas velika pažnja posvećuje dizajnu. Tako su se razvili posebni pravci dizajna kao što su:

- grafički dizajn
- industrijski dizajn
- dizajn enterijera
- modni dizajn
- veb dizajn
- dizajn korisničkog interfejsa itd.

U svim ovim oblastima se koriste računarski programi kao podrška kreativnom radu dizajnera. Pored programa, dizajnerima su na raspolaganju i specijalni ulazno – izlazni uređaji koji im pomažu da lakše pretoče svoje ideje u umetničko delo. Posebno važnu primenu u grafičkom dizajnu imaju sledeći ulazno izlazni uređaji:

- digitalni foto aparati
- umetničke ploče (**Art pad**)
- skeneri (dvo ili trodimenzionalni)
- kolor štampači
- rezači folije (**Vinyl Cutter**)

PROGRAMSKA REŠENJA ZA GRAFIČKI DIZAJN

Photoshop je grafički editor kompanije Adobe Systems namenjen za kreiranje i grafičku obradu bitmapiranih (rasterskih) slika.

Grafički dizajn obuhvata izradu grafičkih rešenja čitav niz različitih proizvoda kao što su:

- vizit karte, memorandumi, koverte
- plakati, posteri, brošure, prospekti, katalozi, cenovnici, kalendari
- dnevne novine, časopisi, knjige, slikovnice
- pozivnice, zahvalnice, čestitke, diplome

- ambalaža
- promotivni materijal (npr. upaljči, olovke, majce) itd.

Najčešće korišćeni programi za rad u oblasti grafičkog dizajna su Photoshop, Corel Draw i Illustrator.

Photoshop je grafički editor kompanije Adobe Systems namenjen za kreiranje i grafičku obradu bitmapiranih (rasterskih) slika. Slika se sastoji od matrice tačaka koje se nazivaju pikseli (**picture elements**). Svaki piksel ima tačno definisanu boju, kao mešavinu tri osnovne boje, kao i neke druge atribute, kao što je na primer transparentnost. Photoshop se može koristiti za kreiranje potpuno novih slika ili za grafičku obradu digitalnih fotografija ili skeniranih slika.

Prva verzija Photoshop-a publikovana je 1990. godine i mogla se kupiti samo za Apple Macintosh računare. Od 1992. godine prodaje se i Photoshop verzija za Windows. Od 2004. godine program nosi ime Adobe Photoshop CS (**Creative Suite**).

Za kreiranje i editiranje slike Photoshop nudi veliki broj alata za crtanje i bojenje kao što su olovke, četkice, sprej, gumice itd. Svaki od ovih alata se može podešavati shodno potrebama korisnika. Unutar slika moguće je dodavati tekst čiji se izgled prilagođava korišćenjem stilova. Pored toga izgled slike se može menjati korišćenjem različitih filtera. Na taj način se mogu dobiti različiti vizuelni efekti. Filteri su nedestruktivni, što znači da se i posle promene izgleda slike ne uništavaju originalni podaci o pikselima.

Posebna snaga Photoshop-a leži u velikom broju modela boja. Program može da radi sa sledećim modelima: RGB, lab, CMYK, grayscale (slike u različitim nivoima sivog), binary bitmap (crno-bele slike) i duotone (slike u različitim nivoima neke boje, slično kao grayscale, ali u drugoj boji). Sliku je moguće prebaciti iz jednog u drugi model boje.

TIPOVI DATOTEKA

Photoshop može da čita i piše datoteke u svojim nativnim formatima

Photoshop može da čita i piše datoteke u svojim nativnim formatima:

- PSD (**Photoshop Document**) – koji čuva sve lejere sa maskama, prostorima boja, transparencijom, tekstrom itd.
- PSB (**Photoshop Big**) – koji je namenjen za velike datoteke (preko 2 GB) i
- PDD (**PhotoDeluxe Document**)

Slike je moguće zapisati i u drugim standardnim ili nativnim formatima kao što su jpeg, GIF, EPS, PNG, BMP, JPEG200, Cineon, OpenEXR, Targa i TIFF, ali se onda ne zapisuju svi atributi slike, nego se ona uglavnom zapisuje kao rasterska slika.

Photoshop čuva istoriju editiranja slike tako da je moguć povratak na neko ranije stanje. Ova istorija može biti zapisana zajedno sa slikom. Ukoliko se često rade isti zadaci postupak je moguće automatizovati pisanjem skripta baziranog na događajima.

CorelDraw je program koji je namenjen za izradu vektorski baziranih crteža. Namenjen je pre svega dizajnerima koji rade ilustracije. Prva verzija programa, objavljena 1989 godine, bila

je namenjena Windows platformi. Prilikom kreiranja crteža u CorelDraw-u koriste se osnovne grafičke primitive kao što su linije, poligoni, krugovi itd. Svaka primitiva je u programu opisana matematičkim modelom. Korišćenjem korisničkog interfejsa mogu se zadavati i menjati parametri primitive čime se dobijaju različite instance primitive. Lakoća rada sa ovim proizvodom posledica je intuitivnog korisničkog interfejsa koji dozvoljava da se krive koje čine geometrijske entitete edituju na nivou čvorova i tangenta.

Jaka strana CorelDraw-a oduvek je bio rad sa tekstrom. Tekst se posmatra kao specifičan objekat koji može biti obojen ili ispunjen različitom teksturom. Pored toga tekst je moguće postaviti po definisanoj putanji funkcijom fit text to path.



Slika 6.1 Primeri editovanja krivih linija u programu CorelDraw

MANIPULACIJA TEKSTOM

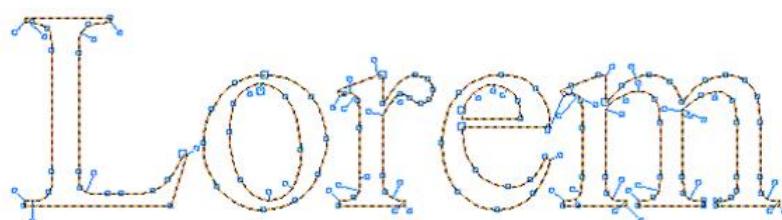
Sav tekst se može petvoriti i u krive linije i onda se može nastaviti njegovo editiranje na nivou čvorova i tangenata

Sav tekst se može petvoriti i u krive linije i onda se može nastaviti njegovo editiranje na nivou čvorova i tangenata.



Slika 6.2 Poravnavanje teksta sa prethodno zadatom putanjom u programu CorelDraw

Illustrator je Adobe-ovo rešenje za vektorsku grafiku. Program je po karakteristikama sličan program CorelDraw, ali omogućuje mnogo bolju integraciju sa PhotoShop-om i drugim Adobe-ovim proizvodima, kao što je na primer Adobe Flash.



Slika 6.3 Tekst pretvoren u krive

▼ Poglavlje 7

Pokazna vežba: EC, CMS i WordPress

E-COMMERCE

Pravljenje srednjih do većih aplikacija zahteva dodatnu integraciju sa postojećim informacionim sistemima, kao što su korporativne baze podataka

Predviđeno vreme pokazne vežbe je 105 minuta.

Iako dobro razvijena web lokacija ne daje dodatnu vrednost proizvodu ili usluzi, ona povećava vrednost kompanije. Zbog toga je važno da firma izabere pravilnu razvojnu strategiju u cilju većeg povratka uloženih resursa. Raznovrsnost e-biznis modela i aplikacija, koje variraju u veličini od malih prodavnica do globalnih razmara, zahteva različite razvojne metodologije i različite vrste pristupa. Na primer, mali sajtovi koji se bave prodajom mogu biti razvijeni sa HTML, JAVA ili nekim drugim jezikom. Takođe se mogu koristiti već gotovi komercijalni programi.

Veće ili specijalne **E-Commerce** (EC) aplikacije mogu biti pisane u okviru firme ili kompanije, a takođe se može angažovati profesionalni (specijalizovani) programer. Pravljenje srednjih do većih aplikacija zahteva dodatnu integraciju sa postojećim informacionim sistemima, kao što su korporativne baze podataka, intranet veze, **enterprise resource planning** (ERP), i sa drugim aplikativnim programima. Dakle, iako proces izgradnje EC sistema može da varira, u mnogim slučajevima ona teži da prati prilično standardni format. Tradicionalni sistemski životni ciklus (SDLC) sistematski vodi programere kroz šest faza analize i dizajna:

- identifikacija problema,
- analiza,
- logički dizajn,
- fizičko projektovanjem,
- implementacija,
- održavanje.

SDLC

Izbor web tehnologija pri izrađivanju web stranica je veliki

SDLC je osnova za razvoj većini tradicionalnih poslovnih sistema. Međutim, softverske i hardverske inovacije omogućavaju prelazak na više unapređeni pristup razvoju e-trgovine i

poslovanja. Izbor web tehnologija pri izrađivanju web stranica je veliki. Kasnije će biti opisane najzastupljenije tehnologije.

Iako je online trgovina prisutna u Srbiji, ona se uglavnom vezuje za veće kompanije koje mogu da izdvoje dovoljnu količinu resursa za izradu i održavanje svojih online prodavnica. Manja preduzeća uglavnom nisu spremna da ulože velika sredstva za razvijanje online biznisa, ali to ne predstavlja prepreku za otvaranje online prodavnice obzirom na to da postoji veliki broj besplatnih i kvalitetnih softverskih alata koji omogućuju kreiranje Web shop-a.

EC aplikacije, kao i svi ostali informacioni sistemi, obično su napravljeni da omoguće jedan ili više poslovnih procesa. Zbog toga, njihovo planiranje mora da bude usklađeno sa biznis planom organizacije kao i sa konkretnim uključenim procesima. Pored toga, svaki zahtev mora da bude pažljivo analiziran koristeći različite metode, da bi se osigurala potrebna funkcionalnost u skladu sa zahtevima poslovnih procesa i korisnika. Rezultat ovog koraka je da se nastavi sa specifičnom aplikacijom u okviru rasporeda, budžeta i dodeljenih odgovornosti. Prvi korak se uglavnom izvodi u okviru organizacije (sa konsultantima po potrebi). Drugi korak može se završiti u okviru organizacije ili angažovanjem spoljnog programera. Obe ove aktivnosti mogu biti kompleksne, ali su neophodne, posebno za sisteme koji zahtevaju velike investicije pri održavanju i radu.

EC ARHITEKTURA

EC arhitektura je plan za organizovanje osnovne infrastrukture i aplikacije za sajt

EC arhitektura je plan za organizovanje osnovne infrastrukture i aplikacije za sajt. Plan precizira sledeće :

- informacije i podaci potrebni za ispunjenje poslovnih ciljeva i vizija,
- moduli aplikacija koji će isporučiti i upravljati informacijama i podacima,
- specifični hardver i softver pomoću koga će aplikativni moduli raditi,
- potreba bezbednost, skalabilnost, i pouzdanost koju zahtevaju aplikacije,
- ljudski resursi i procedure za implementiranje arhitekture.

ALATI I METODOLOGIJE ZA RAZVIJANJE EC APLIKACIJA

Pošto je stvaranje arhitekture iterativan proces, metodologije saradnje, kao što je joint application development (JAD), su posebno korisne u identifikovanju i menjanju sistemskih zahteva

Raznovrsni IT alati i metodologije mogu biti iskorišćene za podršku u stvaranju aplikacije. Pošto je stvaranje arhitekture iterativan proces, metodologije saradnje, kao što je **joint application development** (JAD), su posebno korisne u identifikovanju i menjanju sistemskih zahteva.

EC aplikacije se mogu razvijati kroz nekoliko alternativnih pristupa:

- razvijati aplikaciju u okviru organizacije,
- razvoj od strane angažovane grupe (firma) profesionalaca koja će izgraditi sistem,
- kupiti postojeće aplikacije i instalirati, sa ili bez izmena, od strane administratora preko dobavljača,
- zakup softvera od provajdera servisa aplikacija (ASP), zakup kao servis,
- ući u partnerstvo ili savez koji će omogućiti kompaniji da koristi nečiju aplikaciju,
- pridružiti se nezavisnom E-tržištu, kao što je aukcijski sajt ili sajt razmene, koja pruža potrebne sposobnosti za učesnika,
- koristiti kolaboracioni pristup.

CMS

CMS je skraćenica od Content Management System i predstavlja sistem za upravljanje web sadržajem koji pruža široku lepezu alata za kreiranje, izmenu, pregled i postavljanje sadržaja

Tehnologije koje se koriste za izradu dinamičkih web sajtova su kombinacija PHP+CSS+HTML+JS, CGI, ASP, Java (JSP)... U zavisnosti od kompleksnosti sajta bira se i tehnologija koja će koristiti prilikom izrade istog.

Veliki broj CMS-ova koristi kombinaciju navedenih tehnologija (PHP, CSS, HTML, JS...). CMS je skraćenica od **Content Management System** i predstavlja sistem za upravljanje web sadržajem koji pruža široku lepezu alata za kreiranje, izmenu, pregled i postavljanje sadržaja kao što su tekst, slike, video materijali i audio materijali. Poznati CMS-ovi koji su namenjeni za kreiranje sajtova e-poslovanja su:

- Magento
- Joomla !CMS kombinovan sa Virtuemart komponentom ili Community Builder komponentom namenjenom za kreiranje socijalne mreže
- Wordpress CMS kombinovan sa WP e-Commerce pluginom ili sa WP Symposium pluginom za kreiranje socijalne mreže
- OSCommerce
- Drupal sa Ubercart komponentom

JOOMLA

Osnovna karakteristika Joomla! CMS je proširivost, odnosno instalacija dodataka koji proširuju njegovu funkcionalnost

Joomla! CMS je jedan od najpoznatijih CMS-ova i deli prvo mesto zajedno sa Wordpress-om što se tiče prisutnosti na Internetu. Joomla! je fonetički prepis reči swahili koja znači "svi zajedno" ili još "u jednom". Joomla! je CMS otvorenog koda. Sistem je više puta nagradivan, CMS može da se preuzme na adresi: <http://www.joomla.org>

Osnovna karakteristika Joomla! CMS je proširivost, odnosno instalacija dodataka koji proširuju njegovu funkcionalnost. Na sajtu Extensions.Joomla trenutno postoji 4293 dodataka, većim delom su besplatni. Dodaci se kod Joomla! CMS dele na: komponente, module i pluginove.

Virtuemart Joomla! komponenta je vrlo bitna komponenta za sve one koji žele da vrše online prodaju. Najčešće je korišćena u Joomla! svetu i jedna je od najkompleksnijih komponenti. Sama komponenta poseduje veliki broj dodataka.

Komponenta za kreiranje socijalne mreže je Community Builder. Komponenta je takođe besplatna i može se preuzeti na adresi: <https://extensions.joomla.org/extension/clients-a-communities/communities/community-builder/>

WORDPRESS

Vremenom WordPress je postao najpopularnija blog platforma za upravljanje sadržajem korišćen od strane desetine miliona sajtova

WordPress se prvi put pojavio 2003. godine i bio je jednostavan blogging sistem koji je omogućavamo da se na jednostavan i brz način pokrene i održava blog. Vremenom WordPress je postao najpopularnija blog platforma za upravljanje sadržajem korišćen od strane desetine miliona sajtova. Danas WordPress ne služi samo za kreiranje blogova. Neki od najpoznatijih korisnika ove blog platforme su: Nasa Istraživački Centar, The New York Times, CNN, Fox News. Prednost korišćenja WordPress platforme je jednostavnost pokretanja i korišćenja, mogućnost proširenja WordPress-a uz pomoć pluginova itd... WordPress se može preuzeti preko adrese <http://wordpress.org/>. CMS je besplatan.

Kako bi WordPress postao sajt namenjen e-poslovanju potrebno je dodati pluginove.

WP e-Commerce komponenta je besplatna komponenta za kreiranje Internet prodavnice. Komponenta se može preuzeti na adresi: <http://wordpress.org/extend/plugins/wp-e-commerce/>.

Komponenta Symposium od WordPress CMS-a kreira socijalnu mrežu. Više o komponenti je dato na sajtu: <http://www.wpsymposium.com/> i mogući je download. Komponenta je takođe besplatna.

MAGENTO

Uz pomoć Magenta moguće je istovremeno upravljanje sa više web sajtova

Magento je **open source** platforma na kojoj se kreira Internet prodavnica. Magento je razvijen i pušten na tržište kako bi bio zdrava konkurenca vodećoj e-commerce platformi OSCommerce. Neki od podataka vezanih za Magento:

- Preko 475 000 preuzimanja
- Više of 40 000 članova zajednice

- Hiljade članova zajednice koji rade na unapređenju Magento softvera
- e-Commerce web sajtovi koji imaju Magento će doprineti velikom pomaku za e-Commerce i internet poslovanja

Magento je lak za korišćenje, administratorski deo pogotovo. Uz pomoć Magenta moguće je istovremeno upravljanje sa više web sajtova. SEO optimizacija se obavlja automatski. Magento se može skinuti na adresi: <https://magento.com/>

OSCOMMERCE

Naziv OSCommerce softvera potiče od Open Source Commerce što ukazuje na to da je OSCommerce softver pod GPL (General Public Licence) licencom

OSCommerce predstavlja besplatan e-commerce softver za kreiranje online prodavnica. Naziv OSCommerce softvera potiče od **Open Source Commerce** što ukazuje na to da je OSCommerce softver pod **GPL (General Public Licence)** licencom. Jedan je od najpoznatijih e-Commerce alata.

OSCommerce kombinuje open-source rešenja koja pružaju besplatnu razvojnu platformu a koja uključuju PHP Web programski jezik, Apache Web server i MySQL bazu podataka. OSCommerce se može instalirati na bilo kom PHP3 ili PHP4 Web serveru.

Opšta funkcionalnost:

- Kompatibilan sa svim PHP 4 i MySQL 5 verzijama
- Objektno orijentisani backend
- Multijezička podrška

DRUPAL

Drupal CMS je sistem za upravljanje web sadržajem koji omogućava administratorima web sajta da organizuju i prikažu sadrža

Drupal CMS je sistem za upravljanje web sadržajem koji omogućava administratorima web sajta da organizuju i prikažu sadržaj, da prilagođavaju prikaze i obavljaju rutinske zadatke poput registracije korisničkih imena i lozinke. Jedna od ključnih karakteristika Drupala je da on u celini je otvorenog koda.

Drupal je pisan u PHP-u, programskom jeziku koji je poznat po upotrebi prilikom dizajniranja dinamičkih web sajtova. Drupal radi u mnogim operativnim sistemima (Windows, Mac OS X, Linux, i drugi). Međutim, potrebno mu je obezbediti bazu podataka, poput MySQL, za skladištenje sadržaja i podešavanja. Svako može da kreira modele za Drupal, a trenutno dostupni modeli obuhvataju svašta, od foto galerija do e-komerc sistema. Modulima se može čak promeniti i Drupalovo osnovno ponašanje kako bi se napravio bolji web sajt. Osim toga, možete pronaći i izdašne tutorijale i dokumentaciju koji su dostupni, zahvaljujući zajednici koja radi na razvoju Drupala. Iako su ga neki web dizajneri kritikovali, pripisujući mu da

je težak za učenje, Drupal je dobio brojna priznanja zbog svoje koristi i relativno brzog vremena za izgradnju. Poznat je i po svojoj dinamičkoj prirodi: web sajtovi koji su dizajnirani u Drupalovom okruženju se mogu vrlo brzo modifikovati, prostom izmenom modela. Jomla (Joomla!) predstavlja glavnu alternativuDrupalu. Drupal je začeo i napisaoDrajs Bajtert (Dries Buytaert), koji i dalje predvodi Drupal projekat. On je preveo nemačku reč "drupel", što znači "kapljica", kako bi kreirao reč "Drupal". Kako bi naglasio aspekt zajednice koja doprinosi projektu, upotrebio je nemačku reč "dorp", što znači "selo". CMS se nalazi na sajtu: <https://www.drupal.org/> i moguće ga je besplatno skinuti.

Ubercart je najpopularnija Drupal e-Commerce platforma koja omogućava prodaju preko interneta. Neki od sajtova rađeni uz pomoć ubercarta su: <http://www.adidas-club.sk/>, <http://www.ametrine.no/>, <http://frames.texasexes.org/> ... Komponenta je besplatna i moguće je skinuti sa sajta:

<https://www.drupal.org/project/ubercart/>

WORDPRESS

Osim za blogovanje WordPress se sve češće koristi za izradu i internet prodavnica, prezentacijskih sajtova, socijalnih mreža itd.

Ukoliko želimo da napišemo blog WordPress je jedan od najpopularnijih CMS-ova za izradu istog. Osim za blogovanje WordPress se sve češće koristi za izradu i internet prodavnica, prezentacijskih sajtova, socijalnih mreža itd... Svakodnevno se radi na razvoju dodataka (plug-in-ova) i tema za WordPress. Redovno se objavljaju nove zagrpe koje donose poboljšanja u vidu sigurnosti, funkcionalnosti, dizajna. Jedan je od najpopularnijih trenutno CMS-ova za izradu raznih vrsta sajtova. Sam WordPress je jednostavan, besplatan, ima mogućnost proširivanja (uz pomoć **plug-in-ova**), redovno se ažurira itd.

WordPress bloging platforma je najpopularniji program u svetu prema statistici.

WordPress se koristi na 15% od prvih milion sajtova.

22% novih sajtova koriste WordPress.

WordPress je trenutno najpopularniji CMS sistem na svetu.

Takođe je jedan od najlakših za rukovanje zbog fleksibilnosti i prilagođenosti prosečnom korisniku.

KREIRANJE WORDPRESS BLOGA

Ukoliko nemate hosting i domen, možete da zakupite ili da koristite neki besplatni

Da bi napravili blog u WordPress-u, prvo je potrebno da posudujete hosting i domen. Ukoliko nemate hosting i domen, možete da zakupite ili da koristite neki besplatni. Kao primer ovde će se iskoristiti besplatni hosting i besplatni domen za kreiranje bloga.

Na stranici www.wordpress.com imate mogućnosti da započnete kreiranje svog bloga klikom na dugme Sign Up Now. Čim vam se otvorи sedeća stranica potrebno je da unesete adresu bloga (**host name**). Ukoliko nemate hosting možete iskoristiti besplatni koji bi glasio ime.wordpress.com. U nastavku bice prikazano kako kreirati WordPress besplatan blog:

Potrebno je otići na stranu <https://wordpress.com/> i kliknuti na Create website

REGISTRACIJA

Pre svega da bi ste se registrovali morate imati aktivan mail

Potrebno je registrovati se. Pre svega da bi ste se registrovali morate imati aktivan mail ukoliko nemate potrebno je pre registracije otvoriti neki mail (gmail, hotmail, yahooMail).

The screenshot shows the WordPress.com registration interface. At the top, there's a blue header bar with the WordPress logo and navigation links: Themes, Support, News, Features, Sign Up, and Log In. Below the header, the main title is "Get started with WordPress.com". The registration form consists of four main input fields: "E-MAIL ADDRESS" containing "valentina.paunovic.517@gmail.com", "USER NAME" containing "valentina666", "PASSWORD" containing "*****", and "BLOG ADDRESS" containing "valentina.paunovic". To the right of each input field is a descriptive message or help text. For the email address, it says: "We'll send you an email to activate your account, so please triple-check that you've typed it correctly." For the username, it says: "Your username should be a minimum of four characters and can only include lowercase letters and numbers." For the password, it says: "Great passwords use upper and lower case characters, numbers, and symbols like !@#\$. Generate strong password". Finally, for the blog address, it says: "Choose an address for your blog. You can change the WordPress.com address later, if you don't want a blog you can [signup for just a username](#)".

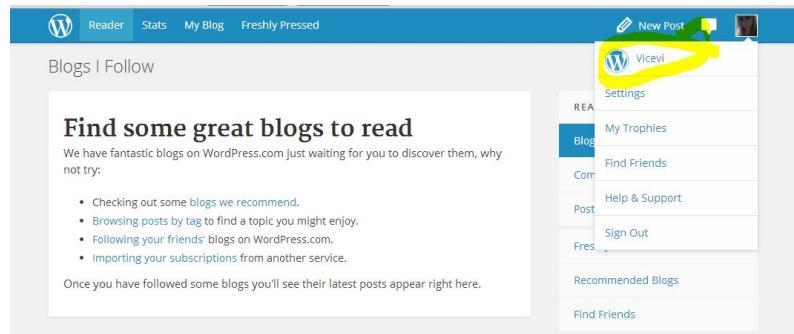
Slika 7.1 Registracija

Primer Forme za registraciju

ADMIN PANEL

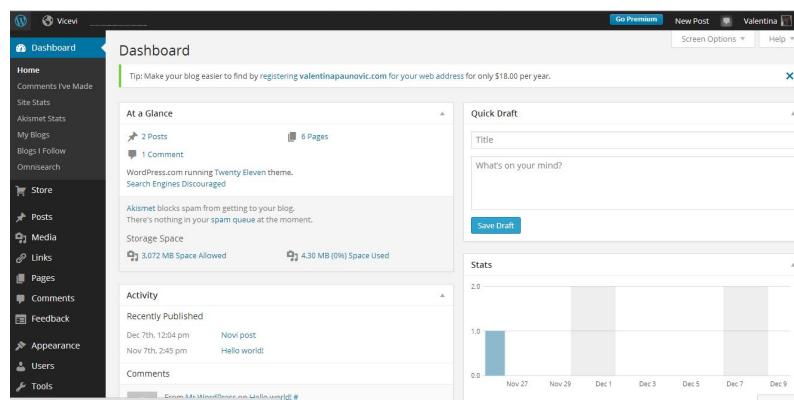
Svaki korisnik može imati više blogova

Nakon kreiranja **account-a** potrebno je logovati se. Svaki korisnik može imati više blogova. Kako bi ušli u admin panel našeg bloga sledimo sledeće korake.



Slika 7.2 Prikaz taba za ulazak na admin panel bloga

U ovom slučaju se blog zove Vicevi.

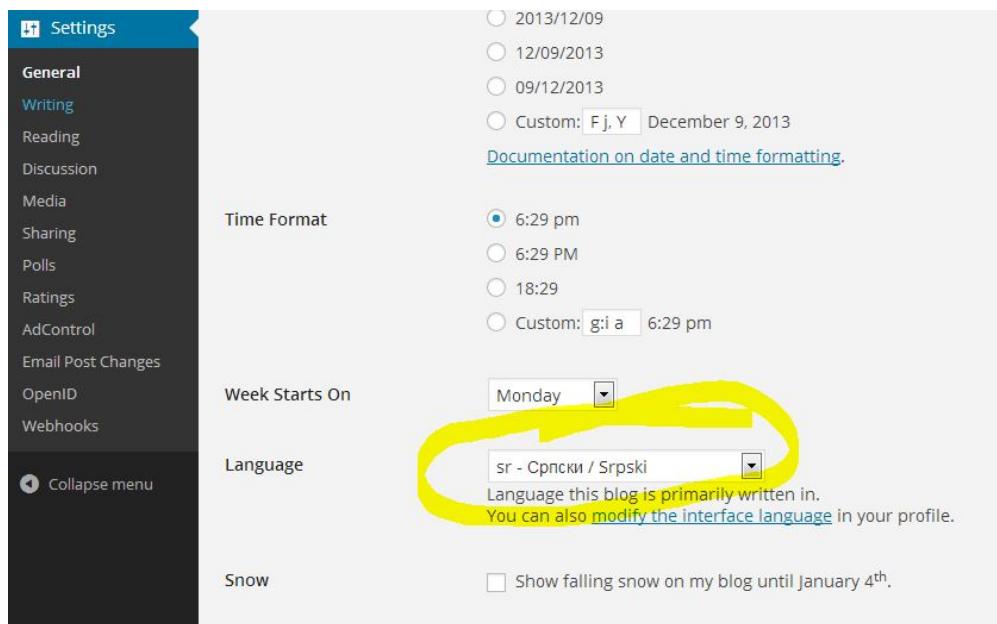


Slika 7.3 Admin panel bloga

PROMENA JEZIKA

Da bi promenili jezik, potrebno je u meniju odabratи opciju Settings.

Veoma važno je navesti da WordPress ima podršku za naš jezik što znači da celi deo za administriranje bloga može da bude na našem jeziku kao i sve informacije koje se automatski pojavljuju na stranici (npr: ko je objavio neki tekst itd). Da bi to primenili u ovom primeru, potrebno je u meniju odabratи opciju Settings.

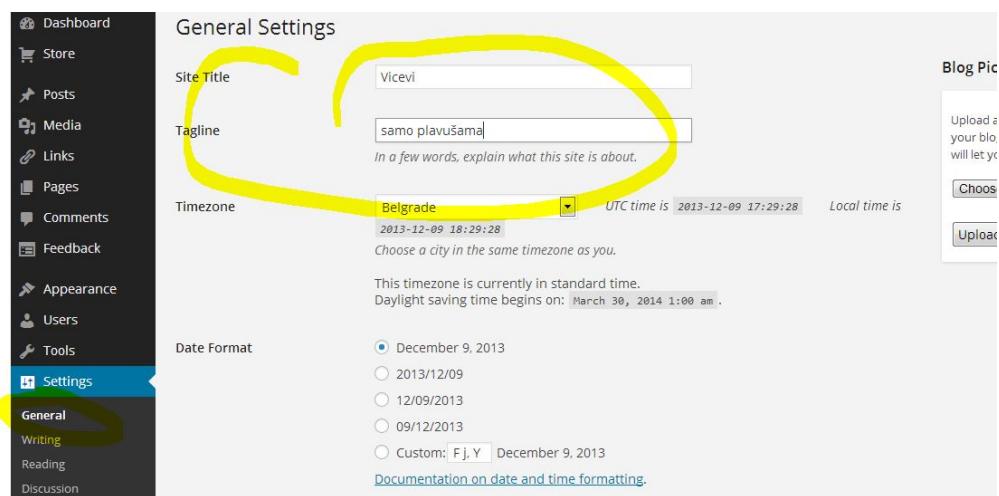


Slika 7.4 Prikaz promene jezika

IME I OPIS BLOGA

U opciji settings možemo promeniti ili dodati ime našeg bloga kao i opis bloga.

U opciji settings možemo promeniti ili dodati ime našeg bloga kao i opis bloga.



Slika 7.5 Prikaz imenovanja bloga

OSTALA PODEŠAVANJA

Deljenje(eng. Sharing) – povezivanje vašeg bloga na neku socijalnu mrežu itd.

Pored ovog opšteg (General) podešavanja možete vršiti još neka važnija podešavanja:

- Pisanja (engl. **Writing**) – koliko linija može da bude u polju za pisanje članaka, određena oblikovanja, koja je podrazumevana kategorija članka, podrazumevani oblik članka, podrazumevanu kategoriju itd.
- Čitanja (engl. **Reading**) – da li da se na početnu stranu prikazuju vaši skorašnji članci ili neka stranica koju odaberete, koliko najviše članaka da se prikazuje na jednoj stranici, da li da se prikazuje na stranici celi tekst ili samo sažetak itd.
- Diskusija (engl. **Discussion**) – dozvoliti ili ne pisanje komentara na novom članku, stizanje mail-a sa obaveštenjem kada neko komentariše članak, omogućiti slanje odgovora na komentare putem e-pošte itd.
- Sadržaja – kolika će biti veličina male, srednje i velike slike itd.
- Privatnosti – da li želite da vaš blog bude vidljiv svima uključujući i web pretraživače ili da se ne dozvoli pristup pretraživačima, ali da se dozvoli uobičajenim poetiocima ili da vaš blog bude privatni odnosno vidljiv samo korisnicima koje izaberete.
- Deljenje(engl. **Sharing**) – povezivanje vašeg bloga na neku socijalnu mrežu itd.

DODAVANJE KORISNIKA

U meniju Korisnici mogu se dodati korisnici bloga i dodeliti im se uloga

U meniju Korisnici mogu se dodati korisnici bloga i dodeliti im se uloga. Uloge mogu biti:

- Upravnik
- Urednik
- Autor
- Saradnik itd...

Dodavanje korisnika je jednostavno, upiše se njegova e-mail adresa i dodeli mu se uloga. Ukolik osoba koristi wordpress dodeliće se automatski. Ukoliko ne koristi javiće se poruka da ta e-mail adresa ne pripada ni jednom korisniku i da se korisnik poziva.

The screenshot shows the WordPress admin interface under the 'Users' section. A yellow circle highlights the 'Invite New' button at the top left of the page. The page displays a table with columns for Username, Name, E-mail, and Role. One row is visible, showing 'valentinapaunovic' as the Username, 'Valentina Paunovic' as the Name, 'valentina.paunovic.517@gmail.com' as the E-mail, and 'Administrator' as the Role. There are also 'Bulk Actions' and 'Apply' buttons at the bottom of the table.

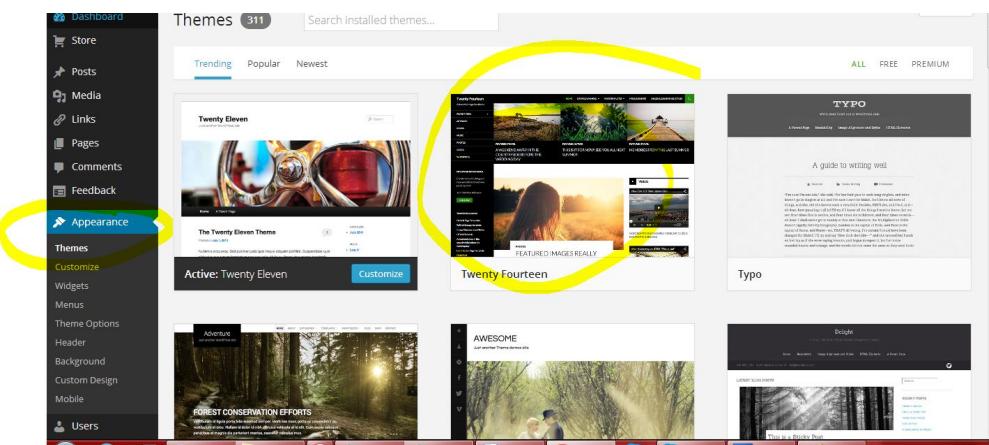
Slika 7.6 Prikaz opcije za dodavanje novog korisnika

PODEŠAVANJE IZGLEDA BLOGA

Nakon odabira tema možete ih modifikovati, dodavati header, dodavati background slike itd odabirom iz opcija apperance.

Veoma važan meni za uređenje bloga je Izgled (**Appearance**). U prvoj opciji ovog menija (**Teme - Themes**), možete da izaberete neku od teme i primenite je na svom blogu. U srednjem delu prozora date su vam ponuđene teme. Ispod slike svake teme imate njeno ime, kratki opis, mogućnost da pogledate kako izgleda ta tema na vašem blogu i mogućnost da, ukoliko vam se sviđi tema potrebno je uključiti je na dugme **Activate**. Postoje teme koje se naplaćuju i besplatne teme. Ukoliko ne želite da platite savet je da se držite tema koje ispod njihovog prikaza nemaju cenu.

Nakon odabira tema možete ih modifikovati, dodavati header, dodavati background slike itd. odabirom iz opcija appearance.

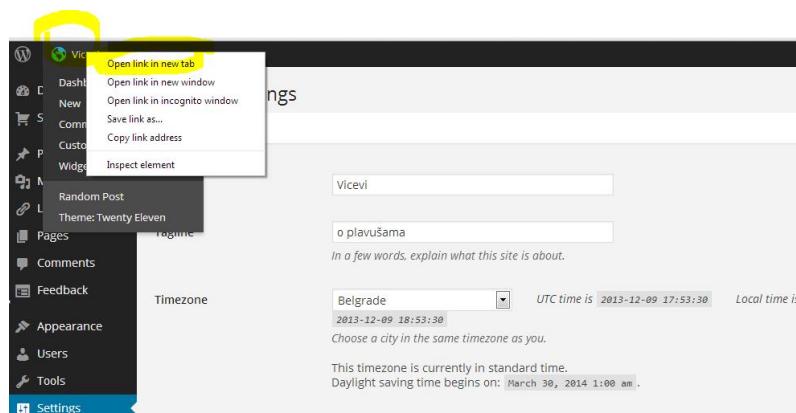


Slika 7.7 Odabir tema

PREVIEW

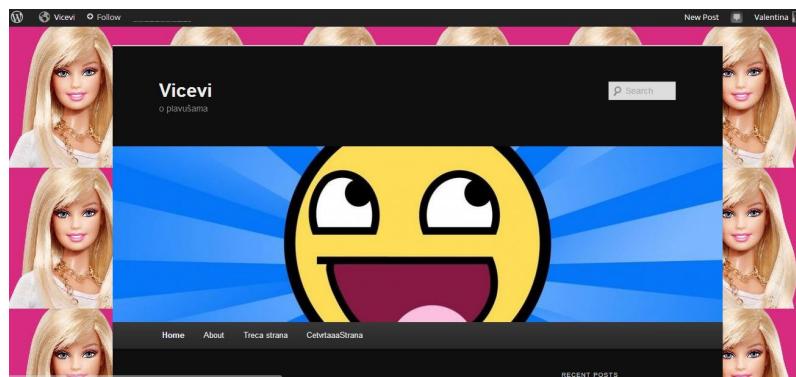
Ukoliko želimo da pogledamo blog koji smo kreirali potrebno je desni klik uraditi na ikonicu bloga u levom uglu.

Ukoliko želimo da pogledamo blog koji smo kreirali potrebno je desni klik uraditi na ikonicu bloga u levom uglu.



Slika 7.8 Pristup kreiranom blogu

Osim toga možete direktno odkucati adresu koju ste naveli za svoj blog. U ovom slučaju je valenitnapaunovic.wordpress.com.



Slika 7.9 prikaz trenutno kreiranog sajta

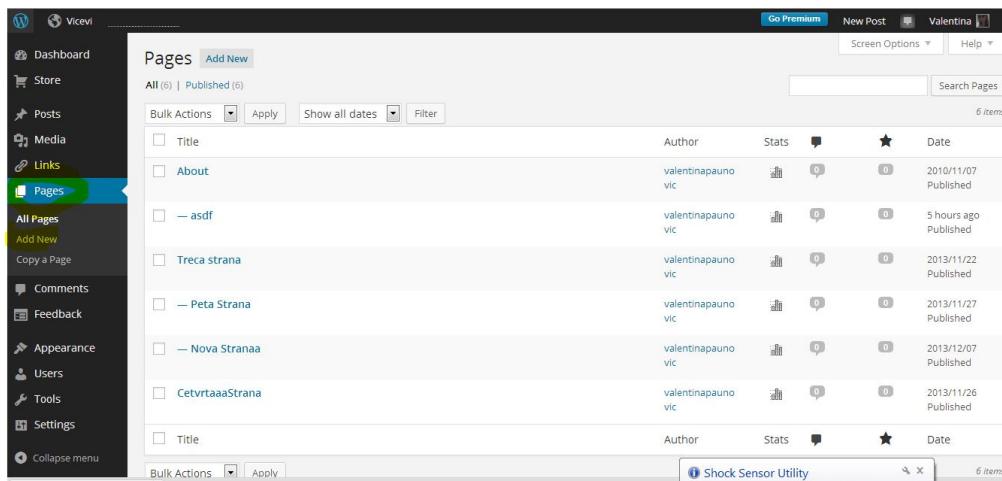
Slika-8 Pristup kreiranom blogu

VIDŽETI I STRANE

Primeri vidžeta su: Kategorije, Skorašnji članci, Kalendar, Pretraga, Veze

Vidžeti koje želite da se prikažu na vašoj stranici prenesite u desnom delu prozora u boksu Sidebar. Primeri vidžeta su: Kategorije, Skorašnji članci, Kalendar, Pretraga, Veze... Pored ovih vidžeta na raspolaganju su vam i mnogi drugi kao što su: Autori, Arhiva, Najbolje ocijenjeni članci, Najpopularniji članci i stranice, Statistika boga itd.

Ukoliko želimo da dodamo menije to radimo tako što u meniju Pages idemo na add new. Tu možemo da upravljamo i sa postojećim menijima.

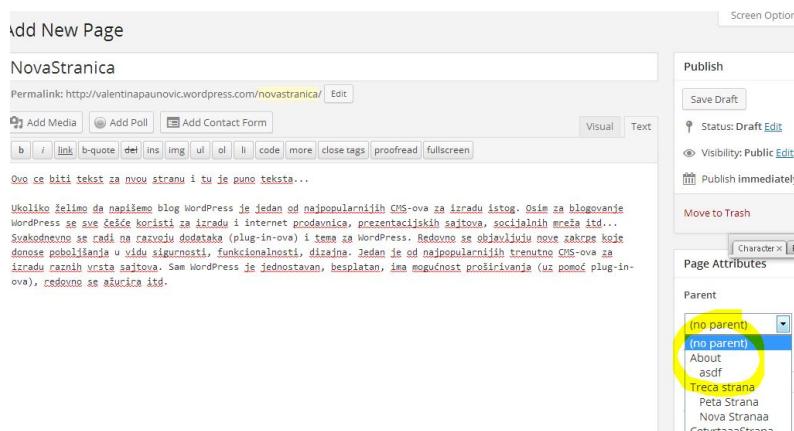


Slika 7.10 Prikaz opcije pages

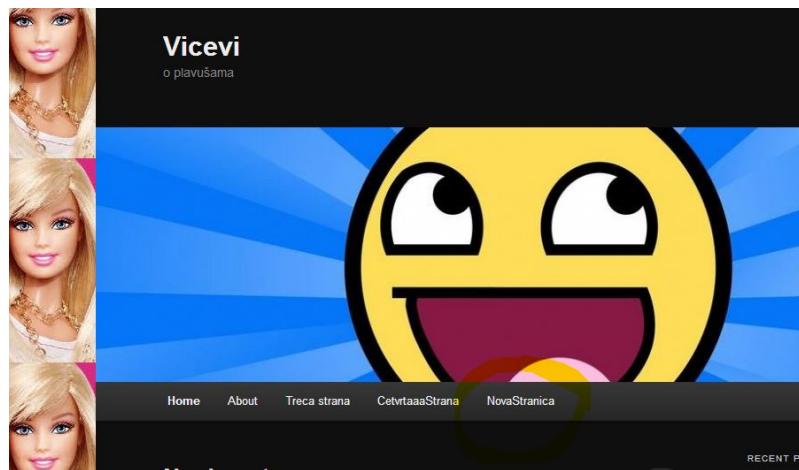
DODAVANJE STRANE

Ukoliko ima roditelja onda ce se prikazati kao padajući meni strane roditelja

Kada dodajemo novu stranu možemo izabrati da li strana da ima roditelje ili ne. Ukoliko ima roditelja onda ce se prikazati kao padajući meni strane roditelja. Ukoliko izaberemo da nema roditelje onda ce se samo prikazati u istom redu kao i ostali meniji. Nakon izrade kliknemo na publish.



Slika 7.11 Prikaz dodavanja novog menija - strane



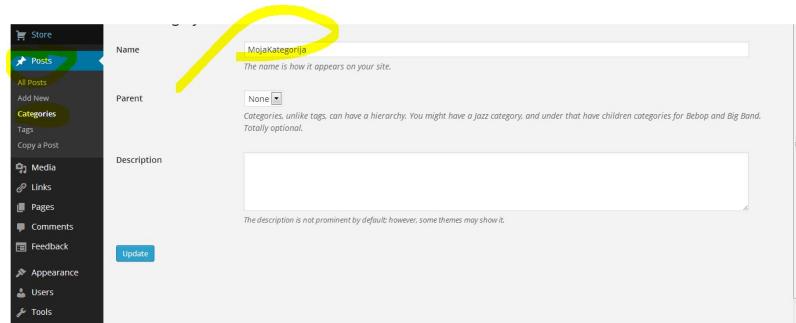
Slika 7.12 Prikaz dodate nove strane

DODAVANJE ČLANAKA

Ukoliko želimo da dodamo novi članak idemo na dodaj novi (add new)

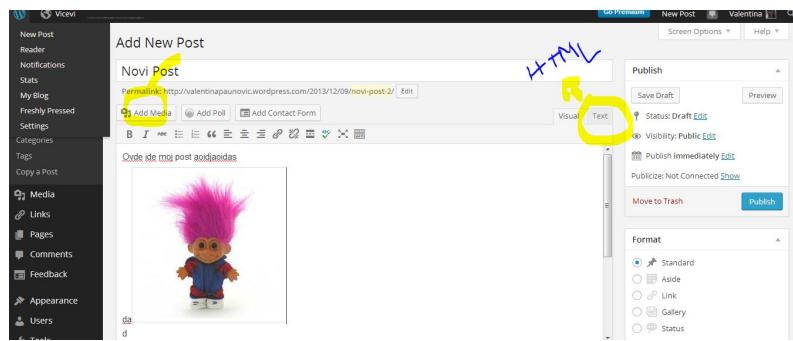
Sada smo stigli do dela gde se pišu članci (**posts**) na stranicama. Veoma je važno je da pre objavljivanja članaka kreirate kategorije u koje ćete svrstati članke koje budete objavljuvati. Prilikom kreiranja ovog bloga automatski je kreirana i jedna kategorija Uncategorized. Ukoliko ne želite ovu kategoriju na vašoj stranici, možete je izbrisati (ali pre toga morate da obrišete sve članke koji su u toj kategoriji) ili možete da joj promenite ime.

U meniju Članci imate podmeni Kategorije. Kada kliknete na taj podmeni, u desnom delu prozora nalaze se sve do sada kreirane kategorije. Da bi promjenili ime kategoriji Uncategorized potrebno je ispod njenog imena kliknuti na Uredi ili jednostavno kliknuti na samo ime kategorije.



Slika 7.13 Menjanje imena Uncategorized kategorije

Kada kliknemo na meni članci (**pages**), prikazaće se svi članci na stranici. Ukoliko želimo da dodamo novi članak idemo na dodaj novi (add new). Post se piše u WYSIWYG editoru koji konvertuje post u html. Moguće je pogledati i članak u HTML formatu i editovati ukoliko je potrebno.



Slika 7.14 Dodavanje članka

KRATAK PREGLED SVIH PLATFORMI, VIDEO

Choose the Best Website Builder for Making Your Website

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 8

Zadatak za samostalni rad: WordPress

ZADATAK ZA SAMOSTALNI RAD

WordPress blog

Predviđeno vreme za izradu zadatka je 30 minuta.

Kreirati WordPress blog koji će biti prezentacija vašeg hobija. Blog treba da ima bar tri strane.

✓ Poglavlje 9

DOMAĆI ZADATAK

OPIS DOMAĆEG ZADATKA - DZ15

Po uzoru na vežbu kreirati WordPress blog koji će biti Vaša lična prezentacija

Očekivano vreme izrade zadatka: 30 minuta

Opis domaćeg zadatka:

Po uzoru na vežbu kreirati WordPress blog koji će biti Vaša lična prezentacija. Word press blog mora sadržati minimalno 3 članka i 4 stranaPredlog šta bi blog mogao da sadrži:

1. Stranu o Vama
2. Stranu o Vašem hobiju
3. Stranu o omiljenom predmetu
4. Stranu koja će sadržati linkove do omiljenih web sajtova, kao i opise tih sajtova
5. Stranu kontaktiraj me

Predmetnom asistentu poslati web adresu do vašeg bloga.

▼ Zaključak

ZAKLJUČAK

U ovom predavanj dati su primeri kako informaciono komunikacione tehnologije mogu da unaprede rad i efikasnost u oblastima kao što su zdravstvo, bioinformatika, medicina, uprava, obrazovanje, elektronsko poslovanje i grafički dizajn.

Literatura

- [1] Helen Partridge, Establishing the Human Dimension of the Digital Divide, Information Security and Ethics: Social and Organizational Issues, IRM Press, 2005.
- [2] Lynette Kvasny, Fay Cobb Payton, Minorities and digital divide, Idea Group, 2005.
- [3] Miloš Simonović, mr Dragan Mišić, prof. dr Miroslav Trajanović, Olivera Tošić, Milan Zdravković, Osnovni aspekti korisničkog interfejsa NIBIS za bibliografsku obradu, YUINFO 2003, Kopaonik



IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

INFORMACIONE TEHNOLOGIJE I SRODNE DISCIPLINE

Lekcija 01

PRIRUČNIK ZA STUDENTE

IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

Lekcija 01

INFORMACIONE TEHNOLOGIJE I SRODNE DISCIPLINE

- ▼ INFORMACIONE TEHNOLOGIJE I SRODNE DISCIPLINE
- ▼ Poglavlje 1: Definicija oblasti računarstva
- ▼ Poglavlje 2: Računarske discipline
- ▼ Poglavlje 3: Informacione tehnologije
- ▼ Poglavlje 4: Računarske nauke
- ▼ Poglavlje 5: Softversko inženjerstvo
- ▼ Poglavlje 6: Informacioni sistemi
- ▼ Poglavlje 7: Računarsko inženjerstvo
- ▼ Poglavlje 8: Računarstvo i druge discipline
- ▼ Poglavlje 9: Pokazna vežba: Korišćenje matematičkih softvera
- ▼ Poglavlje 10: Domaći zadatak
- ▼ ZAKLJUČAK

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

✓ Uvod

UVOD

Cilj ove lekcije je da definiše čime se bave računarske discipline i da kategoriše svaku disciplinu ponaosob

Cilj ove lekcije je da definiše čime se bave računarske discipline i da kategoriše svaku disciplinu ponaosob. Često je teško razumeti razliku između pojedinačnih disciplina kao što su softversko inženjerstvo, računarske nauke, informacioni sistemi i informacione tehnologije. Da bi računarske discipline dospele na nivo napretka i saznanja na kojima su sada, bilo je neophodno da primene znanja i veštine i drugih nauka. Tako ćemo u ovom predavanju specijalnu pažnju posvetiti kognitivnim naukama i njihovom uticaju na računarske discipline.

▼ Poglavlje 1

Definicija oblasti računarstva

POJAM RAČUNARSTVA

Možemo definisati računarstvo kao ciljno orijentisanu aktivnost koja zahteva, koristi ili kreira računare

Pojam računarstva je vrlo širok i danas ga je veoma teško definisati, pre svega zbog toga što računari više nisu samo samostalni uređaji, već su delovi drugih uređaja i mašina. Danas, kad svaki telefon, automobil ili mašina za pranje ima ugrađen računar i kada se rad gotovo svakog preduzeća i ustanove prati ili se njim upravlja pomoću računarskih sistema teško je iscrtati granice koje će pokazati dokle se prostire računarstvo i odakle počinju druge discipline. Zbog toga se može prihvati definicija da je **računarstvo** bilo **koja aktivnost tehničke prirode koja uključuje računare** [1].

Uzmimo za primer jedan medicinski skener (CT - **Computed Tomography** ili MRI - **Magnetic Resonance Imaging**). Rendgenolog će reći da ove uređaje treba izučavati kroz medicinske discipline. Istovremeno, jedan mašinski inženjer, koji je projektovao i proizveo skener i upravljačku jedinicu, može da kaže da je reč o mašinskom uređaju. Budući da se kod ovakvih skenera obrada i prikaz slike vrše pomoću računara, slično pravo bi mogao da ima i inženjer informacionih tehnologija. Iz ovog primera se vidi da je teško definisati gde počinje, a gde se završava računarstvo.

U zajedničkom izveštaju [1] koji su napisale tri institucije: **Association for Computing Machinery** (ACM), **Association for Information Systems** (AIS) i **Computer Society** (IEEE-CS), pod nazivom **Computing Curricula 2005**, daje se nešto šira definicija računarstva:

„Uopšteno, možemo definisati računarstvo kao ciljno orijentisanu aktivnost koja zahteva, koristi ili kreira računare. Tako, računarstvo uključuje projektovanje i građenje hardverskih i softverskih sistema za široku oblast primene; obradu, strukturiranje i upravljanje različitim vrstama informacija; izradu naučnih studija uz upotrebu računara; stvaranje računarskih sistema koji se ponašaju inteligentno; kreiranje i upotrebu komunikacionih medija i medija za zabavu; traženje i dobijanje informacija iz bilo koje posebne oblasti itd. Ova lista je praktično beskrajna, a mogućnosti neizmerne.”

[1] The Joint Task Force for Computing Curricula 2005, A volume of the *Computing Curricula Series, Computing Curricula 2005, September 2005*.

▼ Poglavlje 2

Računarske discipline

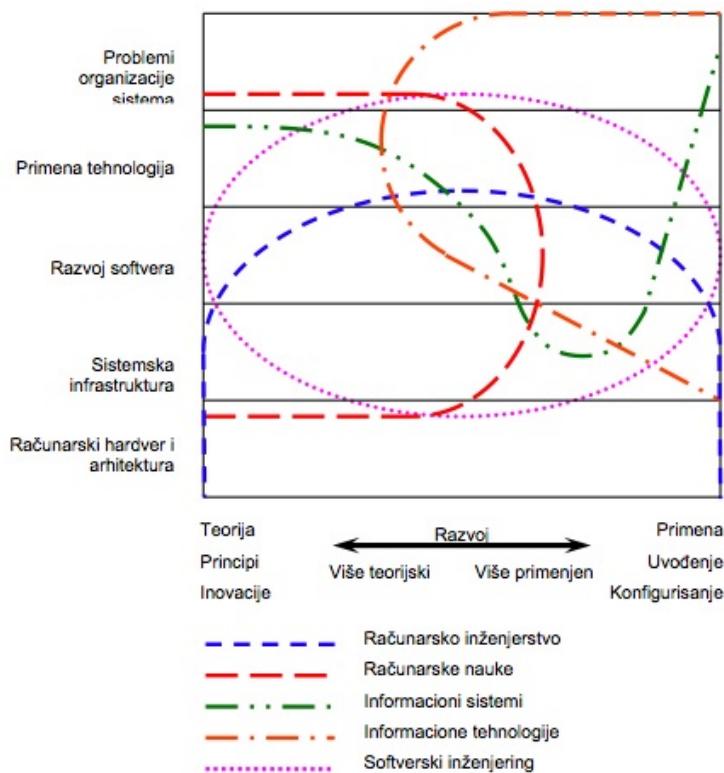
PET OSNOVNIH RAČUNARSKIH DISCIPLINA

Pet osnovnih računarskih disciplina su računarsko inženjerstvo, računarske nauke, informacioni sistemi, informacione tehnologije i softversko inženjerstvo

Do 1990. godine bio je opšti koncenzus da postoje tri računarske discipline: računarske nauke, elektronika i informacioni sistemi. Međutim, eksplozivni razvoj računarstva i Interneta 90-ih godina prošlog veka doveo je do nove kristalizacije disciplina. Može se reći da se danas računarstvo izučava preko pet disciplina:

- Računarsko inženjerstvo (engl. Computer Engineering)
- Računarske nauke (engl. Computer Science)
- Informacioni sistemi (engl. Information Systems)
- Informacione tehnologije (engl. Information Technology)
- Softversko inženjerstvo (engl. Software Engineering)

Kao što se vidi sa slike 1 pojedine oblasti ovih disciplina se preklapaju, ali se pristup njima razlikuje od discipline do discipline. U daljem tekstu se daje bliži opis pojedinih disciplina.



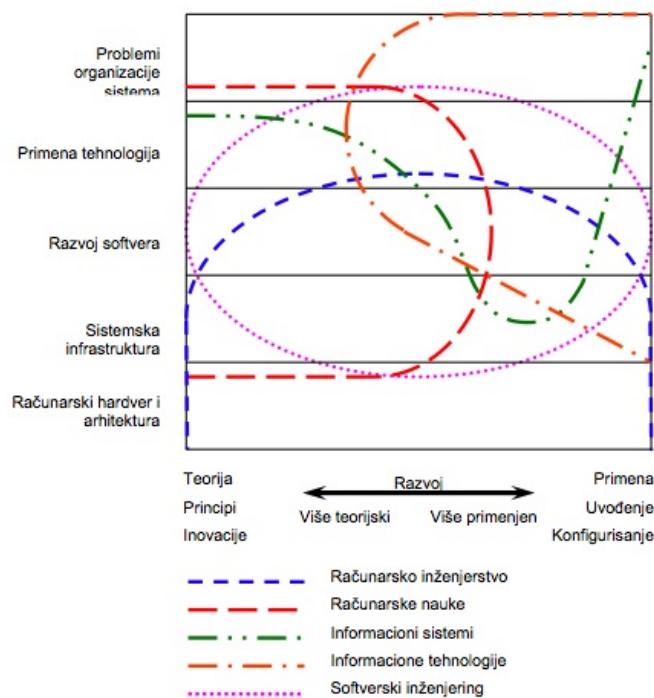
Slika 2.1.1 Oblasti računarskih disciplina

✓ 2.1 Razlike između računarskih disciplina

KOJE SU RAZLIKE IZMEĐU RAČUNARSKIH DISCIPLINA?

Informacione tehnologije poklapaju sa računarskim naukama u razvoju organizacije sistema i primenama softvera

Pogledajmo još jednom oblati računarskih disciplina i segmente u kojima su slični, ali i različiti.



Slika 2.2.1 Oblasti računarskih disciplina

Vidimo da se, na primer, **informacione tehnologije poklapaju sa računarskim naukama u razvoju organizacije sistema i primenama softvera, dok je njegova sistemska infrastruktura u pogledu primene, uvođenja i konfigurisanja bliža softverskom i računarskom inženjerstvu.**

Sa slike jasno vidimo da se **informacione tehnologije** više bave primenom, uvođenjem i konfigurisanjem tehnologija, razvoja softvera i organizacijom sistema, za razliku od softverskog inženjerstva koji se bavi svim fazama razvoja softvera. **Softversko inženjerstvo** se podjednako bavi i teorijom, principima i inovacijom, kao i primenom, uvođenjem i konfigurisanjem softvera. S druge strane, **softversko inženjerstvo** malo zalaže u probleme organizacija sistema i računarskog hardvera i arhitekture.

Informacioni sistemi se sa bave problemima organizacije sistema i iz teorijskog ugla, ali i ugla primene, dok se tehnologijama, softverom i sistemskom arhitekturom bave delimično u polju uvođenja i konfiguracije.

▼ Poglavlje 3

Informacione tehnologije

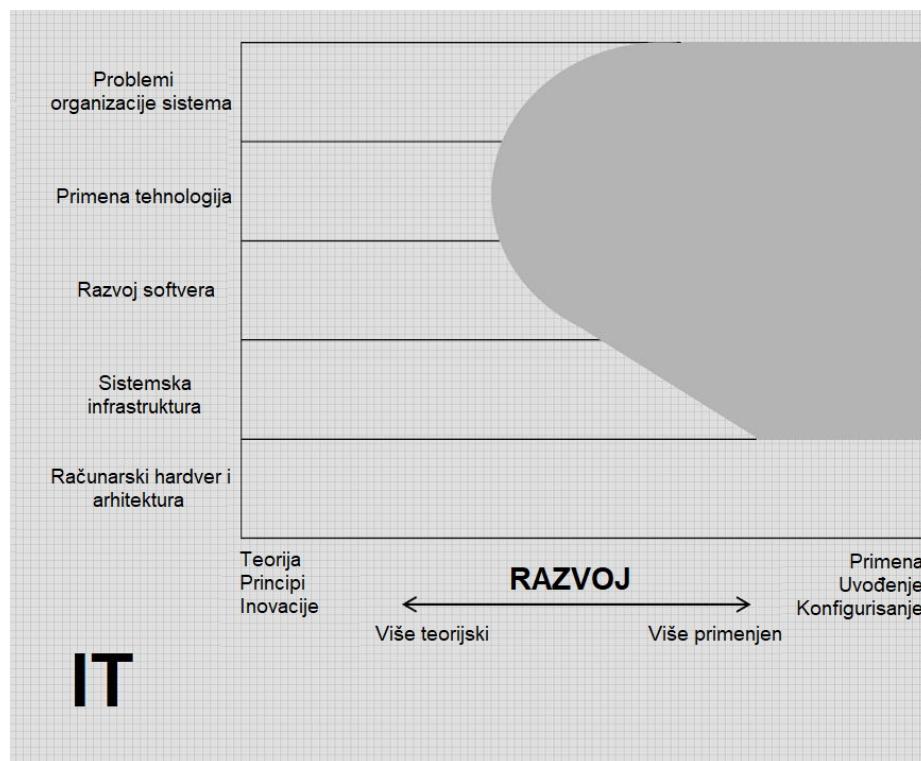
ČIME SE BAVI RAČUNARSKA DISCIPLINA INFORMACIONE TEHNOLOGIJE?

Težište informacionih tehnologija je na primeni tehnologija

Fokus informacionih tehnologija je na primeni tehnologija u oblasti računarstva. Tako da kažemo da je kod informacionih tehnologija težište na tehnologijama. Rad svakog informacionog sistema zahteva dobru tehnološku podršku koja se sastoji u izboru programskog i tehničkog dela računarskih sistema, u njegovom održavanju i unapređivanju. **Specijalisti iz ovog domena su, pored toga, zaduženi za instalaciju i održavanje mreža i mrežne opreme, komunikacionih komponenata, Internet i veb servisa, razvoj multimedijalnih resursa i slično.**

Informacione tehnologije (IT) u najširem smislu obuhvataju primenu svih aspekata računarskih tehnologija. IT, kao akademska disciplina, se bavi pitanjima vezanim za zalaganje za korisnike i njihove potrebe u okviru organizacionih i društvenih konteksta kroz selekciju, stvaranje, primenu, integraciju i administraciju računarskih tehnologija .

Kao posebno važan domen informacionih tehnologija ističe se pouzdanost i sigurnost informacionog sistema, kao i zaštita podataka. S tim u vezi je i planiranje i upravljanje životnim ciklusom tehnoloških resursa, što podrazumeva nabavku, održavanje, nadgradnju i zamenu pojedinih podistema sveukupnih računarskih resursa u organizaciji.



Slika 3.1 Informacione tehnologije

IEEE DEFINICIJA IT-A

CC2005: The Overview Report , page 14:

"Information technology is a label that has two meanings. In the broadest sense, the term information technology is often used to refer to all of computing. In academia, it refers to undergraduate degree programs that prepare students to meet the computer technology needs of business, government, healthcare, schools, and other kinds of organizations. In some nations, other names are used for such degree programs.

"In the previous section, we said that Information Systems focuses on the information aspects of information technology. Information Technology is the complement of that perspective: its emphasis is on the technology itself more than on the information it conveys. IT is a new and rapidly growing field that started as a grassroots response to the practical, everyday needs of business and other organizations. Today, organizations of every kind are dependent on information technology. They need to have appropriate systems in place. These systems must work properly, be secure, and be upgraded, maintained, and replaced as appropriate. Employees throughout an organization require support from IT staff who understand computer systems and their software and are committed to solving whatever computer-related problems they might have. Graduates of Information Technology programs address these needs.

"Degree programs in information technology arose because degree programs in the other computing disciplines were not producing an adequate supply of graduates capable of handling these very real needs. IT programs exist to produce graduates who possess the right combination of knowledge and practical, hands-on expertise to take care of both an

organization's information technology infrastructure and the people who use it. IT specialists assume responsibility for selecting hardware and software products appropriate for an organization, integrating those products with organizational needs and infrastructure, and installing, customizing, and maintaining those applications for the organization's computer users. **Examples of these responsibilities include the installation of networks; network administration and security; the design of web pages; the development of multimedia resources; the installation of communication components; the oversight of email systems; and the planning and management of the technology lifecycle by which an organization's technology is maintained, upgraded, and replaced."**

▼ Poglavlje 4

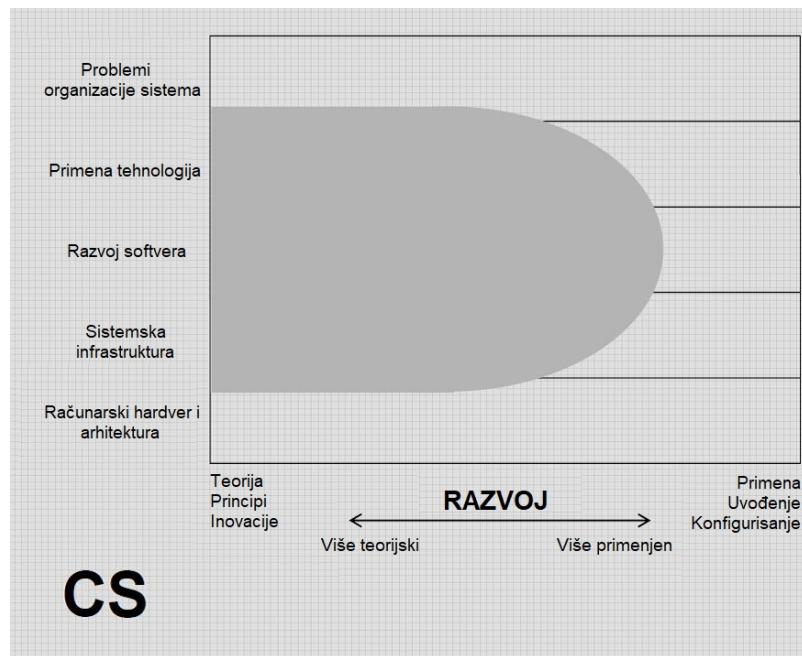
Računarske nauke

ČIME SE BAVI RAČUNARSKA DISCIPLINA RAČUNARSKE NAUKE?

Računarske nauke se bave nalaženjem efektivnih rešenja za računarske probleme, nalaženjem novih primena računarskih sistema, projektovanjem i razvojem novih programske rešenja

Računarske nauke obuhvataju široko polje baznih i razvojnih oblasti računarstva. Polaze od teorijskih osnova i baznih algoritama, pa idu do najsavremenijih aplikacija kao što su robotika, računarska vizija, inteligentni sistemi, bioinformatika i druge. Rad u ovoj oblasti može se podeliti u tri kategorije:

- **Nalaženje efektivnih načina za rešavanje računarskih problema.** Na primer, računarske nauke se bave nalaženjem najboljih načina za smeštanje informacija u baze podataka, slanje podataka kroz mreže ili prikazivanje složenih slika. Teoretska osnova koja se koristi za rešavanje ovakvih problema omogućuje nova, inventivna rešenja koja pomeraju granice mogućnosti korišćenja računarskih sistema.
- **Nalaženje novih primena računarskih sistema.** Ovo podrazumeva ne samo razvoj novih aplikacija nego i razvoj novih jezika višeg nivoa, novih metoda za projektovanje i razvoj aplikacija, nalaženje kvalitetnijih načina za korišćenje postojećih računarskih rešenja kao što su nove klase interfejsa čovek – sistem, pristup informacijama sa bilo kog mesta korišćenjem različitih uređaja i slično.
- **Projektovanje, razvoj i implementacija inovativnih programske rešenja.** Izrada programskih rešenja pre svega pripada oblasti softverskog inženjerstva. Međutim, i računarske nauke se bave ovom oblašću, ali ne rutinskim korišćenjem postojećih softverskih tehnologija, već više u smislu stvaranja inovativnih rešenja.



Slika 4.1 Računarske nauke

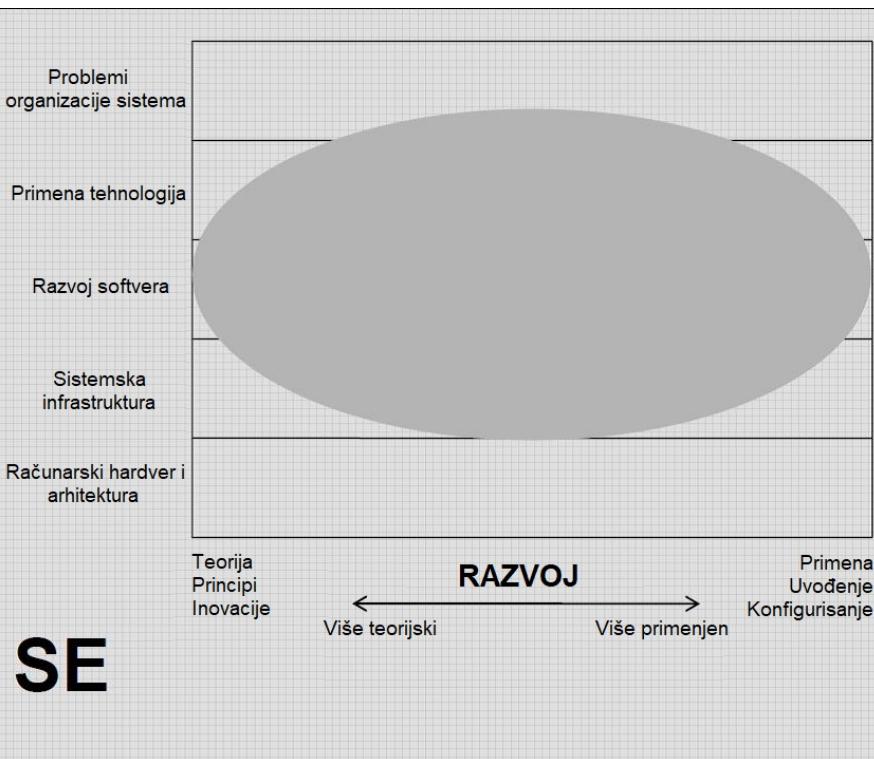
▼ Poglavlje 5

Softversko inženjerstvo

ČIME SE BAVI DISCIPLINA SOFTVERSKO INŽENJERSTVO?

Softversko inženjerstvo se bavi razvojem i održavanjem softverskih sistema, uključujući celokupan ciklus razvoja softvera

Softversko inženjerstvo se razvilo kao jedna od disciplina računarskih nauka, ali je vremenom počelo da se razvija kao zasebna disciplina. **Softversko inženjerstvo je disciplina koja se bavi razvojem i održavanjem softverskih sistema koji su pouzdani, efikasni i koji ispunjavaju sve postavljene zahteve.** Softversko inženjerstvo teži ka tome da integriše matematičke principe i principe računarskih nauka, koristeći inženjerski pristup. Softverski sistemi iz dana u dan rastu i postaju sve složeniji. Važna komponenta ove discipline je pouzdanost i efikasnost programskih sistema.



Slika 5.1 Softversko inženjerstvo

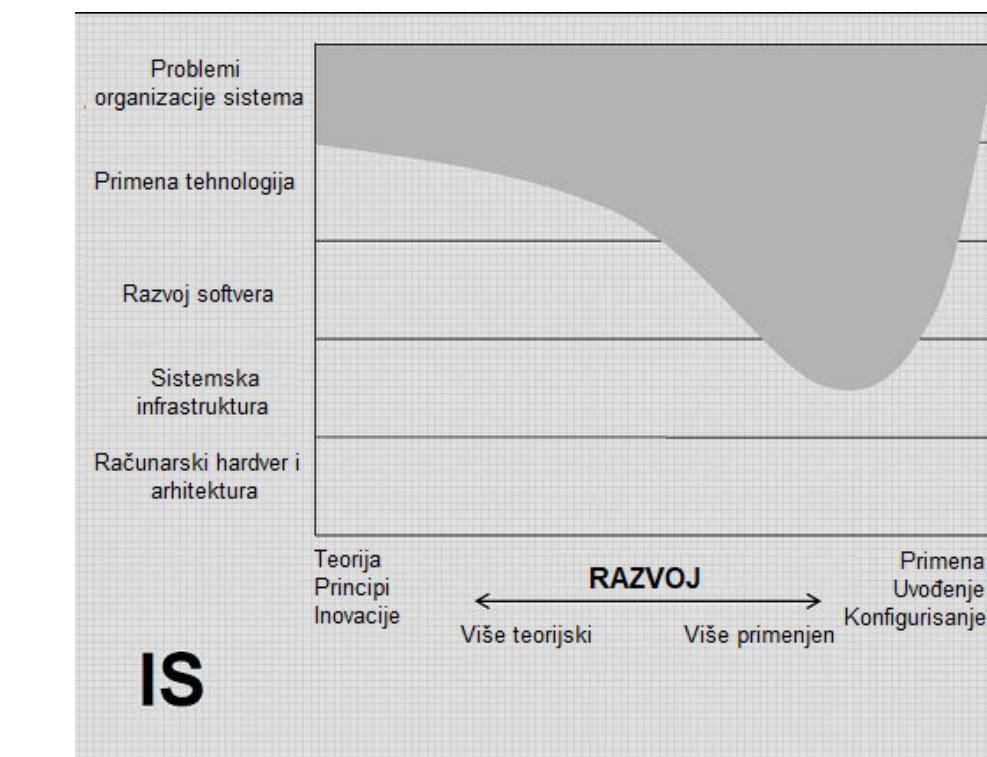
▼ Poglavlje 6

Informacioni sistemi

ČIME SE BAVI RAČUNARSKA DISCIPLINA INFORMACIONI SISTEMI?

Informacioni sistemi se bave integracijom informacionih tehnologija i poslovnih procesa

Informacioni sistemi su posebna disciplina računarstva koja se bavi integracijom informacionih tehnologija i poslovnih procesa kako bi se zadovoljile poslovne potrebe za informacijama i rad organizacija učinio efikasnim. Stručnjaci koji se bave informacionim sistemima treba dobro da poznaju savremene informacione tehnologije i poslovne procese da bi na najbolji način stvorili informacioni sistem koji će generisati, obrađivati i distribuirati potrebne informacije. Zbog njegove obimnosti i stalnog pomeranja tehnoloških mogućnosti i zahteva ni za jedan informacioni sistem se ne može reći da je završen. Informacioni sistemi uvek zahtevaju pored daljeg razvoja i održavanje i prilagođavanje novim poslovnim zahtevima i standardima. Nije moguć razvoj informacionog sistema bez odličnog poznavanja organizacije rada i poslovnih pravila i potreba organizacije. Zbog toga stručnjaci iz ove oblasti najčešće rade u organizaciji za koju razvijaju informacioni sistem i sarađuju sa stručnjacima iz drugih disciplina kako bi što brže implementirali nova rešenja. Njihova dodatna uloga je u upravljanju korišćenjem postojećih informacionih sistema.



Slika 6.1 Informacioni sistemi

✓ Poglavlje 7

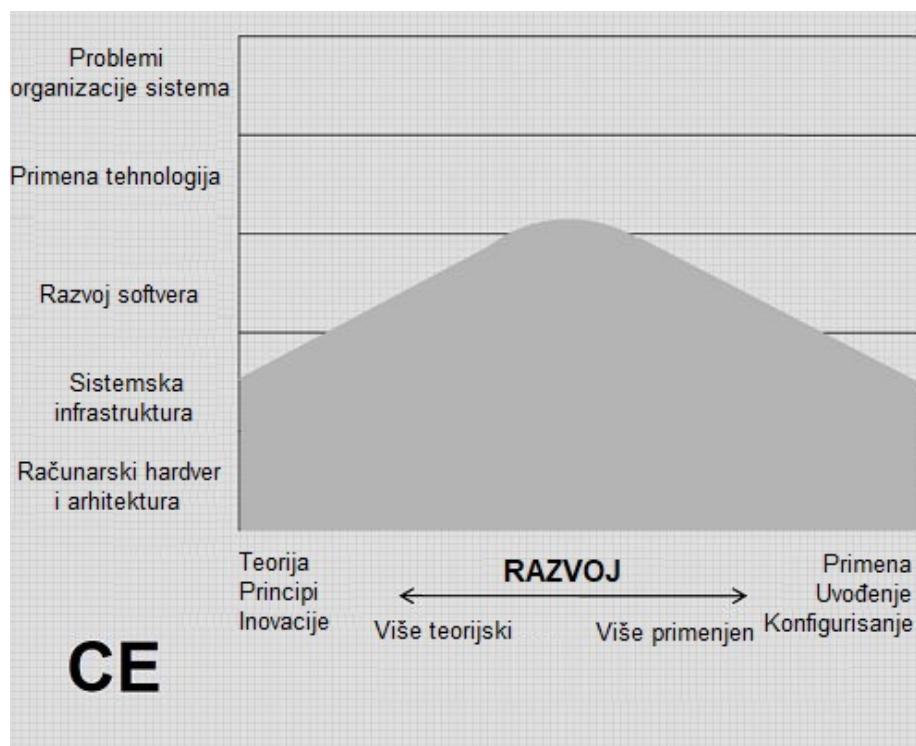
Računarsko inženjerstvo

ČIME SE BAVI RAČUNARSKA DISCIPLINA RAČUNARSKO INŽENJERSTVO?

Računarsko inženjerstvo se bavi dizajnom i razvojem računara i računarskih sistema

Računarsko inženjerstvo (engl. **Computing engineering**) se bavi projektovanjem i proizvodnjom računara, računarskih periferija, računarskih sistema i drugih uređaja koji imaju ugrađene računare. **Računarsko inženjerstvo se bavi dizajnom i razvojem računara i računarskih sistema.** Ova oblast izučava hardver, softver, komunikacije, kao i interakciju među njima. Glavni fokus ove discipline je da primeni principe elektrotehnike i matematike u rešavanju problema prilikom dizajna računara i računarskih uređaja.

Inženjeri koji su stručnjaci za računarsko inženjerstvo bave se projektovanjem procesora, matičnih ploča, memorija, ulazno-izlaznih uređaja i periferija koji se koriste u računarskim sistemima. Računari se sve češće ugrađuju u druge uređaje: automobile, klima uređaje, sisteme za grejanje, liftove, alarmne sisteme itd. Ovakvi računari se nazivaju ugnježdeni (engl. **embedded**) sistemi. Računarsko inženjerstvo se bavi i projektovanjem hardvera i softvera za ovakve sisteme.



Slika 7.1 Računarsko inženjerstvo

✓ Poglavlje 8

Računarstvo i druge discipline

POVEZANOST RAČUNARSTVA SA DRUGIM DISCIPLINAMA

Računarstvo je u dobroj meri zasnovano na mnogim disciplinama, a posebno na kognitivnim naukama, matematici, statistici

Iako se na prvi pogled čini da nema velike veze između računarstva i drugih disciplina, ozbiljnim proučavanjem računarstva moguće je uočiti da je ono u dobroj meri zasnovano na mnogim disciplinama, a posebno na:

- Kognitivnim naukama
- Matematici
- Statistici.



Slika 8.1.1 Baziranje računarstva na drugim naukama

✓ 8.1 Analogija rada ljudskog mozga i računara

„STRUKTURE PODATAKA“ I „PROCEDURE“ U LJUDSKOM MOZGU

Osnovna hipoteza kognitivnih nauka je da razmišljanje može najbolje da se razume preko reprezentacionih struktura u mozgu i računarskih procedura koje se izvode nad tim strukturama

Osnovna hipoteza kognitivnih nauka je da razmišljanje može najbolje da se razume preko reprezentacionih struktura u mozgu i računarskih procedura koje se izvode nad tim

strukturama. Nažalost, postoje velika neslaganja u naučnim krugovima u vezi sa prirodom reprezentacije i računarskim procedurama koje čine razmišljanje.

Većina radova kognitivnih naučnika tvrdi da um ima mentalnu reprezentaciju sličnu reprezentaciji podataka u računaru i računarske procedure slične računarskim algoritmima. Tako su kognitivni teoretičari u ljudskom mozgu prepoznali sledeće strukture podataka:

- *logičke teoreme*
- *pravila*
- *pojam*
- *slika*
- *analogija*

i sledeće mentalne procedure:

- *dedukcija*
- *traženje*
- *podudarnost*
- *rotacija*
- *istraživanje*

Jedna grupa kognitivnih naučnika, koja je nazvana konekcionisti, polazi od neurona, njihovih veza i načina rada kako bi definisala strukture podataka i algoritme. Alati, bazirani na ovim pretpostavkama, koji simuliraju rad ljudskog mozga nazivaju se neuronske mreže.

▼ 8.2 Kognitive nauke i psihologija

KOGNITIVNA PSIHOLOGIJA

Kognitivna psihologija pored eksperimenata takođe koristi i teorijski pristup i računarsko modeliranje

Iako kognitivna psihologija danas koristi teorijski pristup i računarsko modeliranje, njen osnovni metod su eksperimenti u kojima ljudi učestvuju. Ljudi, i to uglavnom studenti, dovode se u laboratorije kako bi se proučavali različiti načini razmišljanja pod kontrolisanim uslovima. Tako, na primer, psiholozi eksperimentalno istražuju:

- **vrste grešaka koje ljudi prave** pri deduktivnom zaključivanju
- **način kako ljudi formiraju i primenjuju principe**
- **brzinu ljudskog razmišljanja** sa mentalnim slikama
- **mogućnost ljudi da rešavaju probleme** korišćenjem analogija.

Kao i u drugim naukama, naučni rad psihologa se svodi na četiri cilja:

- **Opis ponašanja.** Istraživači se trude da nepristrasno opišu ponašanje. Kako to nije uvek moguće, oni pokušavaju da dokumentuju svaku predrasudu koja može da utiče na opis.

- **Predikcija (predviđanje) ponašanja.** Dužim studiranjem fenomena istraživač može da predvidi ponašanje. Ovakvo predviđanje je moguće čak i ako ne postoji razumevanje razloga koji su doveli do ponašanja. Ako se zasniva samo na posmatranju, predikcija ponašanja ne mora uvek da bude tačna.
- **Određivanje uzroka ponašanja.** Kada nisu jasni razlozi koji izazivaju neko ponašanje vrše se kontrolisani eksperimenti kojima se otkrivaju razlozi ponašanja. Kada su poznati razlozi ponašanja, moguće je dati tačnu predikciju ponašanja.
- **Objašnjenje ponašanja.** Poznavanje uzroka ponašanja ne objašnjava ponašanje. Da bi se objasnilo ponašanje potrebno je detaljno razumevanje mehanizma pomoću koga uzročni faktori izazivaju tačno određeno ponašanje.

VRSTE PSIHOLOŠKIH ISTRAŽIVANJA

Psihološka istraživanja mogu biti: kontrolisana, korelaciona i deskriptivna

Kao što je prethodno rečeno, **psihološka istraživanja se rade eksperimentisanjem sa subjektima - ljudima**. Da bi postigli navedene ciljeve istraživanja psiholozi vrše tri vrste istraživanja:

- **Kontrolisana istraživanja**
- **Korelaciona istraživanja**
- **Deskriptivna istraživanja**

Zaključak o tome kako radi ljudski mozak ne može se doneti samo na osnovu ovakvih posmatranja jer je njima nemoguće dokučiti prirodu mentalnih operacija. Ipak, u spremu sa saznanjima dobijenim od drugih kognitivnih nauka, rezultati ovih istraživanja postaju vrlo korisni.

KONTROLISANA ISTRAŽIVANJA

U kontrolisanim istraživanjima izabrani subjekti za eksperiment se po slučajnom principu dele na dve grupe, tako da obe grupe imaju slične karakteristike

U **kontrolisanim istraživanjima** izabrani subjekti za eksperiment se po slučajnom principu **dele na dve grupe, tako da obe grupe imaju slične karakteristike**: pol, prosečna starost, obrazovanje itd. Za obe grupe se obezbeđuju isti eksperimentalni uslovi. Razlog za ovo je da se izbegne da na rezultate eksperimenta utiču promenljive koje nisu važne za eksperiment. Ako se, na primer, vrši testiranje uticaja boje pozadine forme za unos podataka na tačnost unosa podataka, onda za obe grupe treba izabrati iste računare, istu aplikaciju, iste podatke za unos, iste radne uslove (temperatura, vlažnost) itd. **Tako će istraživači biti sigurni da razlike u rezultatima eksperimenta (ponašanje) potiču samo od razlike vrednosti nezavisne promenljive (ili promenljivih) koje se istražuju.** Rezultat eksperimenta, u ovom slučaju broj grešaka prilikom unosa podataka, naziva se zavisna promenljiva. Ako se, na primer, u jednoj grupi organizuje testiranje sa belom pozadinom forme za unos podataka, a u drugoj sa crnom

pozadinom forme, pri istim ostalim uslovima, i ako je broj grešaka u grupi koja radi sa belom pozadinom manji, istraživači mogu da izvuku zaključak da je bela pozadina pogodnija.

KORELACIONA ISTRAŽIVANJA

Koreaciona istraživanja se zasnivaju na šablonima povezanih događaja ili, drugim rečima, na korelaciji između događaja

U nekim slučajevima nije praktično ni etički vršiti slučajnu podelu subjekata u grupe. Na primer, ako se želi ispitati uticaj pušenja na pojavu raka, onda bi izabranu grupu zdravih nepušača trebalo podeliti na dve grupe, a onda jednu naterati da puši. Pošto je to neetički, jer pušenje možda može da izazove rak, ne primenjuje se kontrolisano istraživanje. **Koreaciona istraživanja se zasnivaju na šablonima povezanih događaja ili, drugim rečima, na korelaciji između događaja. I u ovom slučaju se subjekti istraživanja dele na dve grupe: eksperimentalnu i kontrolnu grupu.** Za članove kontrolne grupe se zna da nisu pod uticajem varijabile od interesa. U ovom primeru to su nepušači, a eksperimentalnu grupu čine pušači. Sada se ispituje frekvencija pojave raka kod jednih i kod drugih i na osnovu toga se izvlači zaključak. Nedostatak ove metode je što istraživači nikad nisu sigurni da ne postoji neka druga promenljiva, koju nisu uzeli u obzir, a koja možda značajno utiče na rezultate eksperimenta.

DESKRIPTIVNA ISTRAŽIVANJA

Deskriptivna istraživanja se bave opisom fenomena koji se istražuje

Ova istraživanja se ne zasnivaju na razlikama između ljudi ili grupa. Umesto toga, **deskriptivna istraživanja se bave opisom fenomena koji se istražuje.** I ovde se kao metod za definisanja zaključaka mogu koristiti statističke metode. Na primer, ako se želi ispitati koje vrste grešaka ljudi prave u radu sa računarcem, nema potrebe da se oni dele na grupe, već je potrebno meriti frekvenciju i opisati svaku vrstu greške.

▼ 8.3 Kognitivne nauke i veštačka inteligencija

POJAM INTELIGENCIJE I NJENA VEZA SA RADOM LJUDSKOG MOZGA

Da bi kompletirali psihološke eksperimente o zaključivanju, formiranju principa i rešavanju problema na bazi analogija, koriste se računarske metode koje simuliraju ljudske osobine

Da bi se rešila krucijalna pitanja vezana za rad ljudskog mozga, psihološki eksperimenti moraju da budu interpretirani teoretskim okvirima koji su bazirani na mentalnim reprezentacijama i procedurama.

Najbolji način za razvoj teoretske osnove je gradnja i testiranje računarskih modela koji su analogni mentalnim operacijama. Da bi kompletirali psihološke eksperimente o deduktivnom zaključivanju, formiranju principa, mentalnim slikama i rešavanju problema na bazi analogija, istraživači su razvili računarske metode koje simuliraju ove aspekte ljudskih osobina.

Da bi mogli adekvatno da definišemo termin "veštačka inteligencija", neophodno je da adekvatno formulisemo samo značenje i razumevanje termina "inteligencija". Međutim, kod definisanja ovog termina javljaju se sledeće nedoumice:

- Pojam inteligencije se vezuje za određene sposobnosti i osobine koje poseduju pojedini ljudi. **Da li se inteligencija može vezivati samo za pojedinačne sposobnosti i osobine, ili ona predstavlja skup povezanih sposobnosti? Da li inteligencija predstavlja skup nepovezanih osobina i mogućnosti?**
- Ljudski mozak je još uvek nepotpuno istražen. **Da li se pojam inteligencije uopšte može precizno definisati?**

Međutim, iako je cilj istraživanja i razvoja veštačke inteligencije da se približi logičkom radu ljudskog mozga, savremeni računari i najsavremenija dostignuća u informacionim tehnologijama nisu superiorna u odnosu na ljudski mozak, niti imaju za cilj da to dokazuju. Ona samo treba da ukažu da prirodna i veštačka inteligencija nisu isto.

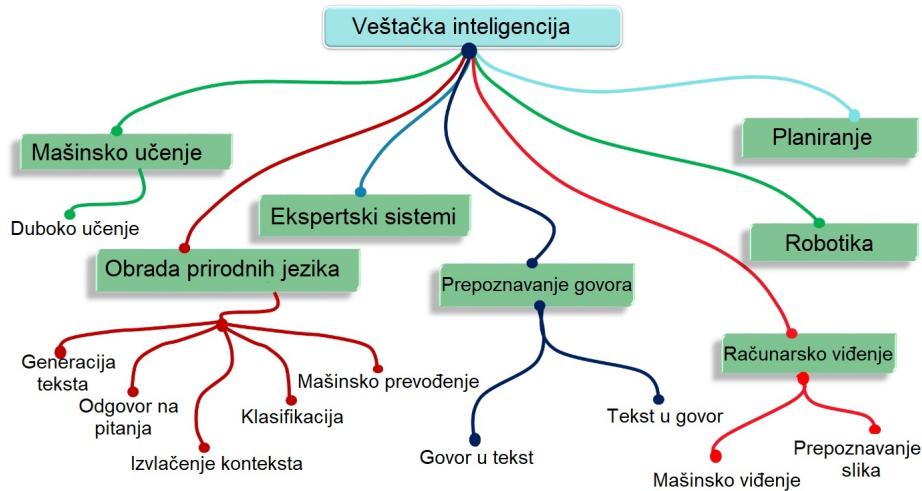
OSNOVNI CILJEVI VEŠTAČKE INTELIGENCIJE

Osnovni ciljevi veštačke inteligencije su projektovanje, gradnja i eksperimentisanje sa računarskim modelima rada ljudskog mozga

Međutim, iako je cilj istraživanja i razvoja veštačke inteligencije da se približi logičkom radu ljudskog mozga, savremeni računari i najsavremenija dostignuća u informacionim tehnologijama nisu superiorna u odnosu na ljudski mozak, niti imaju za cilj da to dokazuju. Ona samo treba da ukažu da prirodna i veštačka inteligencija nisu isto.

Veštačka inteligencija je grana računarstva koja se bavi intelligentnim sistemima. Osnovni ciljevi ove nauke su projektovanje, gradnja i eksperimentisanje sa računarskim modelima rada ljudskog mozga. Zbog toga je veštačka inteligencija idealni komplement psihološkim eksperimentima. Prvi istraživači u ovoj oblasti su bili John McCarthy, Marvin Minsky, Allen Newell i Herbert Simon.

Podoblasti veštačke inteligencije su prikazane na slici



Slika 8.2.1 Podoblasti veštače inteligencije

OBLASTI PRIMENE VEŠTAČKE INTELIGENCIJE

Veštačka inteligencija je našla primenu u igrama, ekspertskim sistemima, prepoznavanju govora itd.

Veštačka inteligencija je našla primenu u više različitih polja kao što su:

- Računarske igre – Veštačka inteligencija ima značajnu ulogu u strateškim igrama kao što su šah, poker, iks-oks itd., gde mašina može da napravi sledeći potez baziran na heurističkim znanjima.
 - Obrada prirodnih jezika - Postoje računari koji razumeju prirodne jezike koje ljudi govore.
 - Ekspertske sisteme - Postoje aplikacije koje integrišu maštine, softvere i specijalne informacije kako bi dale savete i razloge. One pružaju korisnicima objašnjenje i predloge.
 - Sistemi viđenja - Ovi sistemi razumeju, interpretiraju i spoznaju vizuelni ulaz preko računara. Na primer,
- Špijunski avioni prave fotografije koje se koriste da bi dobili prostorne informacije ili mape neke oblasti.
- Doktori koriste ekspertske sisteme da dijagnostikuju pacijente.
- Policija koristi softvere koji mogu da uporede i prepoznaju kriminalca sa portretom osobe kreirane od strane forenzičara.
- Prepoznavanje govora - Neki inteligentni sistemi su sposobni da čuju i prepoznaju jezik u vidu rečenica i njihovo značenje dok čovek govori. Može da razume različite akcente, žargone, promenu glasa čoveka ako je prehladen itd.
 - Prepoznavanje rukopisa - Softver za prepoznavanje rukopisa može da pročita tekst napisan na papiru ili na displejima. Prepoznaće oblik slova i konvertuje ih u elektronski izmenjiv tekst.
 - Pametni roboti - Roboti su sposobni da izvrše zadatke koji su im zadani od strane čoveka. Imaju senzore kako bi iščitavali podatke iz stvarnog sveta poput svetla, topote, temperature, pokreta, zvuka, udara i pritiska. Imaju efikasne procesore, više senzora i

veliku memoriju. Takođe, sposobni su da uče iz svojih grešaka i da se prilagode novim okruženjima.

KOGINITIVNO RAČUNARSTVO

Cognitive computing, Jerome Pesenti, TEDxBermuda

Pogledajte kratku prezentaciju Jerome Pesenti, glavnog programera IBM-ove igre Jeopardi, u kojoj kaže da je razvoju softvera, koji se oslanja na prirodnom jeziku i mašinskom učenju, potreban nov način prikupljanja i tretiranja podataka .

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

❖ 8.4 Kognitivne nauke i antropologija

KAKO ANTROPOLOGIJA UTIČE NA RAZVOJ RAČUNARSTVA?

Kognitivna antropologija proučava kako se mišljenje formira u različitim kulturološkim okruženjima

Kognitivna antropologija proučava kako se mišljenje formira u različitim kulturološkim okruženjima. Poznato je, naime, da se jedan način razmišljanja u jednoj kulturološkoj sredini može smatrati pozitivnim, dok se u drugoj smatra sasvim negativnim, a ponekad i zabranjenim. Razloge za ovo treba tražiti u nasleđenoj kulturi i tradiciji, kao i u veri. **Zato je za kognitivne nauke neobično važno da sagleda rad mozga u različitim psihološkim i socijalnim uslovima.**

Glavni metod kulturoloških antropologa je etnogeografija. Ova metoda zahteva od istraživača da dovoljno dugo žive u sredini koju ispituju kako bi se saživeli sa lokalnim stanovništvom i razumeli njihove običaje i način razmišljanja.

Znanja koja nudi kognitivna antropologija su posebno korisna pri projektovanju interakcije čovek - računar, za lokalizaciju i globalizaciju aplikacija i veb sajtova itd.

KAKO KULTURE I TEHNOLOGIJE STVARAJU JEDNA DRUGU?

How Culture and Technology Create One Another: Ramesh Srinivasan at TEDxUCLA

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

❖ 8.5 Kognitivne nauke i filozofija

KAKAV UTICAJ IMA FILOZOFIJA NA RAČUNARSTVO?

Filozofi se najčešće bave opštim pitanjima kao što je, na primer, odnos uma i tela ili metodološkim pitanjima, kao što je priroda objašnjenja pojmove nađenih u kognitivnim naukama

Osim retkih izuzetaka, filozofi ne rade sistematska empirijska posmatranja i ne prave računarske modele. Ali, oni su ipak važni za kognitivnu nauku jer se bave osnovnim problemima koji su baza za eksperimentalni i računarski pristup. **Apstraktna pitanja kao što su priroda reprezentacije ili računanja nisu problem kojim se psihologija ili veštačka inteligencija svakodnevno bavi, ali se i kod njih neminovno javljaju kada istraživači počnu da razmišljaju o tome šta oni zapravo rade.**

Filozofi se najčešće bave opštim pitanjima kao što je, na primer, odnos uma i tela ili metodološkim pitanjima, kao što je priroda objašnjenja pojmove nađenih u kognitivnim naukama. Pored toga, filozofi se bave i normativnim problemima kao što je kako ljudi treba da razmišljaju ili deskriptivnim problemima kao što je, na primer, kako ljudi razmišljaju.

❖ 8.6 Uticaj lingvistike na računarstvo

KOGNITIVNA LINGVISTIKA

Kognitivna lingvistika pokušava da opiše i objasni sistematičnost, strukturu i funkcije jezika i kako se te funkcije realizuju jezičkim sistemom

Mada lingvisti vrše psihološke eksperimente i razvijaju računarske metode, njihov glavni teoretski zadatak je da identifikuju gramatičke principe koji obezbeđuju osnovnu strukturu ljudskog jezika. Prve radove u ovoj oblasti publikovao je Čomski. Lingvisti pokušavaju da identifikuju, ponekad vrlo male, razlike između gramatički ispravnog i neispravnog načina izražavanja.

Kognitivna lingvistika pokušava da opiše i objasni sistematičnost, strukturu i funkcije jezika i kako se te funkcije realizuju jezičkim sistemom. Međutim, vrlo važan razlog zbog čega kognitivni lingvisti proučavaju jezik leži u činjenici da jezik reflektuje šablove razmišljanja.

OBLASTI PRIMENE LINGVISTIKE U RAČUNARSTVU

Primena lingvistike u računarstvu se može prepoznati u projektovanju novih računarskih jezika, računarskih podržanog prevodenja, prepoznavanju govora i dr.

Lingvistika je vrlo važna za računarstvo, posebno u oblastima kao što su:

- **Projektovanje novih računarskih jezika.** Svaki jezik, pa i računarski, ima semantiku i sintaksu i podleže određenim univerzalnim lingvističkim pravilima. Za projektovanje efikasnih računarskih jezika vrlo je važno poznavati kognitivnu lingvistiku.
- **Provera tačnosti pisanja.** Većina dobrih programa za obradu teksta ima ugrađene rutine za proveru ispravnosti pisanja reči na osnovu ugrađenog rečnika. Korisnik može da izabere rečnik za određeni jezik, a može i sam da doda nove reči ili izgradi novi rečnik. U slučajevima kada postoji više mogućih opcija ili nije moguće jednoznačno odrediti šta bi bilo ispravno rutina za proveru ispravnosti pisanja nudi korisniku moguće opcije.
- **Provera gramatičke ispravnosti.** Ovo je mnogo teži proces jer je pored sintakse u nekim slučajevima potrebno odrediti i semantiku rečenice.
- **Računarski podržano prevodenje.** Mašinsko prevodenje prirodnih jezika je davna težnja programera, ali je skopčana sa velikim problemima. Pored potrebe za rečnicima izvornog i odredišnog jezika, kao i u prethodnom slučaju, neophodna je ugrađena inteligencija koja će prepoznati semantiku teksta. Nažalost, trenutne mogućnosti i tehnike nisu na dovoljno visokom nivou da uvek omoguće ispravan prevod.
- **Prepoznavanja govora.** Postoji više razloga zašto je prepoznavanje govora važno za računarstvo. Korisnik sa ograničenim sposobnostima vida i koordinacije pokreta može koristiti računar zahvaljujući različitim metodama prepoznavanja govora. Pored izdavanja komandi operativnom sistemu i programima, prepoznavanje govora se koristi i za unos teksta i podataka. U ovoj oblasti postignuti su odlični rezultati i na tržištu se mogu naći dobri programi za pretvaranje govora u pisani tekst.

✓ 8.7 Matematika i informacione tehnologije

MATEMATIKA KAO OSNOVA ZA RAČUNARSKE DISCIPLINE

Budući da matematika uglavnom jednoznačno daje rešenja problema iz realnog sveta, ona predstavlja idealnu osnovu za pisanje računarskih algoritama za rešavanje tih problema.

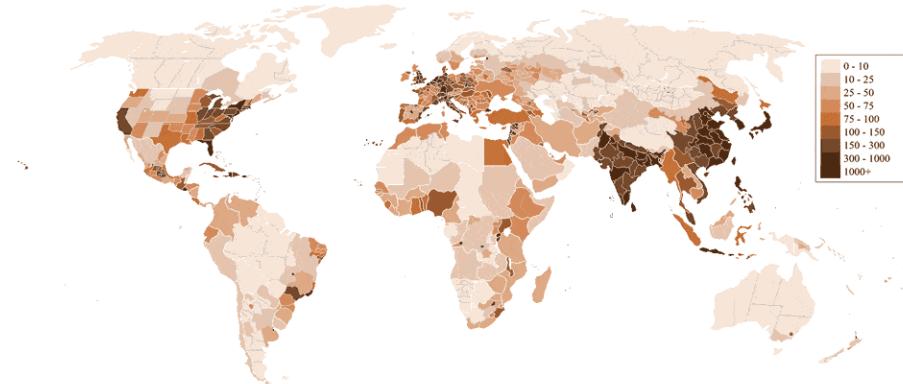
Matematika je apstraktna nauka koja se bavi proučavanjem količinskih odnosa i prostornim formama realnog sveta. Da bi dobila na opštosti, odnosno da bi njena upotreba mogla da bude univerzalna, matematika apstrahuje realnost i bavi se samo odnosima.

Budući da matematika uglavnom jednoznačno daje rešenja problema iz realnog sveta, ona predstavlja idealnu osnovu za pisanje računarskih algoritama za rešavanje tih problema.

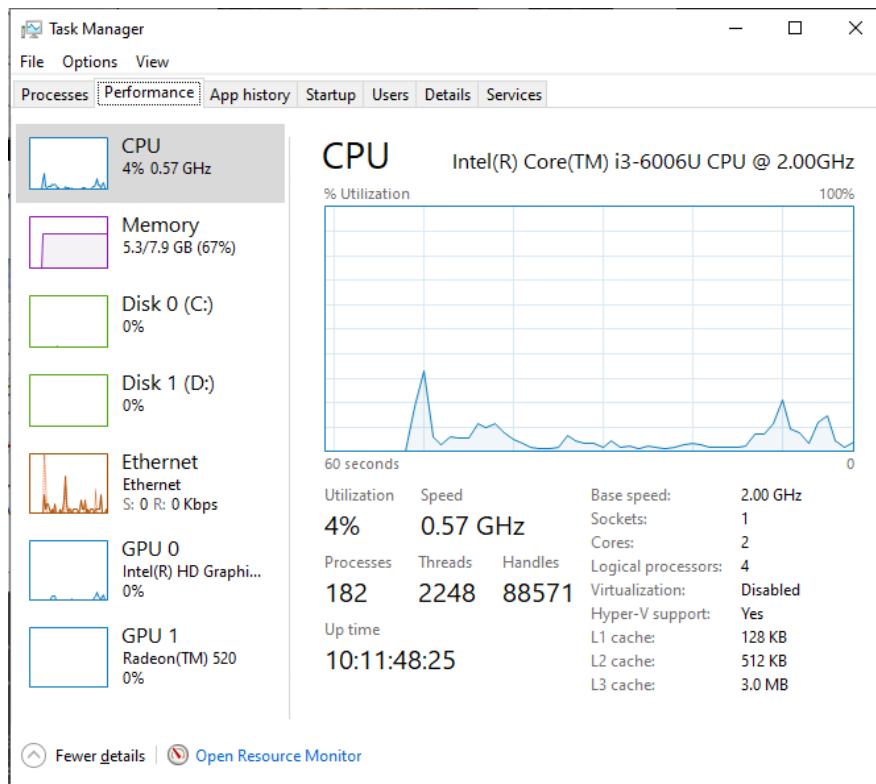
Prvi računari su bili zasnovani na osnovnim matematičkim principima. Konkretno se misli na principe **binarne matematike** u kojoj se koriste samo 0 i 1 za predstavljanje brojeva. U kontekstu binarne matematike, računari su mogli da čitaju **0 kao "off"** (isključeno) i **1 kao "on"** (uključeno), pa je samim tim niz preklopnika omogućio predstavljanje racionalnih brojeva. Kasnije su se javile i sofisticirane matematičke metode poput **teorije grafova i topologije**, a koje su primenjene u informacionim tehnologijama. Ove metode su omogućile brži razvoj računara i informacionih sistema. Kako je tehnološki razvoj informacionih tehnologija napredovao, svoju primenu i napredak u informacionim tehnologijama su doživele i druge grane matematike poput **numeričke analize, teorije aproksimacija, operaciono istraživanje i statistika**.

RAZLIČITE PRIMENE MATEMATIKE I STATISTIKE

Primene u softverskim alatima



Slika 8.3.1 Prikaz stanovništva na mapi



Slika 8.3.2 Prikaz Windows upravljača zadatacima (engl. Task manager)

KARIJERE U OBLASTI RAČUNARSTVA

Različite karijere u oblasti računarstva

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 9

Pokazna vežba: Korišćenje matematičkih softvera

OSNOVNI MATEMATIČKI SOFTVERI

Programski paketi poput Matlab, Rlab, Octave i Scilab predstavljaju pouzdana sredstva stručnjaka za rešavanje problema iz različitih oblasti

Predviđeno vreme pokazne vežbe je 30 minuta.

Uspeh u struci, u velikoj meri, zavisi od sposobnosti da se, u borbi sa izazovima profesije, pronađe rešenje. U toj borbi, kroz proces stvaranja, stručnjak koristi sve svoje intelektualne potencijale. Postizanje odgovarajućeg kvaliteta rada njegov je osnovni cilj, a za to su ravноправно važni i znanje i dobar alat. **U najvećoj meri osnovni alat predstavljaju upravo mentalne sposobnosti razvijene kroz proces učenja i saznavanja.** Međutim, neki od alata su fizičke prirode i tekovine su civilizacijskog i tehнološkog razvoja.

Računar je u potpunosti zamenio mnoge od fizičkih alata koje inženjer koristi. Njegovo korišćenje dovelo je do revolucije u nauci i tehnici. Bez upotrebe moćnih tehničkih i matematičkih programa za računare mnogi problemi, sa kojima se inženjer svakodnevno sreće, ostali bi nerešeni. Međutim, ovi programi sa svakom narednom verzijom postaju sve zahtevniji, toliko da prevazilaze sposobnosti prosečnog korisnika. Čest je slučaj korišćenja softvera čije se osnove i mogućnosti ne razumeju i koji zahteva dobru tehničku i matematičku osnovu.

Širom sveta, programski paketi Matlab, Rlab, Octave i Scilab predstavljaju pouzdana sredstva stručnjaka za rešavanje problema iz različitih oblasti. Koriste se u širokom spektru oblasti kao što su prirodne nauke, finansije i ekonomija, tehničke nauke.

Bilo da se radi o inženjerima posvećenim rešavanju praktičnih problema ili naučnicima okrenutim teorijskom radu, problemi sa kojima se oni sreću često se mogu brže rešiti uz pomoć ovih softverskih paketa nego uz pomoc standardnih jezika za programiranje (Fortran, indexFortran C, Java, C++, ...).

Primena ovih programskega paketa omogućava brzo donošenje zaključaka, smanjenje vremena potrebnog za analizu i razvoj, smanjenje troškova itd. Ovakav širok spektar mogućnosti Matlab, Rlab, Octave i Scilab čine idealnom osnovom za rešavanje mnogih praktičnih inženjerskih i istraživačkih problema.

MATLAB

Matlab pruža priliku inženjeru da definiše problem, modelira, pronađe rešenja i izvrši dalju analizu

Matlab je počeo da se razvija krajem 70-tih godina prošlog veka i još uvek, čak ni među dugogodišnjim korisnicima, ne postoji slaganje oko toga da li se radi o softverskom paketu ili programskom jeziku.

Matlab predstavlja najpopularnije okruženje. Matlab pruža priliku inženjeru da definiše problem, modelira, pronađe rešenja i izvrši dalju analizu. Matlab je postao de facto standard za dizajn i simulaciju DSP sistema. Međutim, Matlab ima bitan problem – skup je. Na svu sreću, postoje alternativna rešenja: Besplatni klonovi.

Glavni Matlab klonovi su Scilab, Octave, i Rlab. Nijedan od njih nije pravi "klon", zato što nijedan ne pruža 100% kompatibilnost sa Matlab "m-files". Ipak svi oni pružaju mogućnosti slične Matlab-u, za mnogo bolji odnos cene/performansi (zbog toga što su besplatni).

Pre nego što se krenulo sa njegovom širokom upotrebom, korišćen je uglavnom interno - u vojne svrhe. Tokom godina, postao je standardni alat u univerzitetskim centrima širom sveta. Paket je namenjen, pre svega, rešavanju problema predstavljenih u obliku vektora i matrica, te otuda i potiče njegovo ime: MATLAB (skraćenica od MATrix LABoratory). Međutim, primena MATLAB-a nije ograničena samo na oblast matematike. Svi koji se bave teorijskom i primjenjom fizikom, elektrotehnikom, hemijom, ekonomijom i/ili bilo kojom srodnom granom, naći će primenu ovom paketu u svojoj oblasti. Rezultati se mogu atraktivno prezentovati, počev od matrica, preko 2D i 3D grafikona, do modeliranja i simuliranja kompleksnih procesa. Moguće je čak i razvoj sopstvenih aplikacija, kao i prosleđivanje rezultata drugim softverskim paketima. Sa odgovarajućim dodacima MATLAB omogućava pristup podacima sa različitih instrumenata, iz drugih fajlova ili baza podataka.

RLAB & OCTAVE

Rlab (programske jezik) predstavlja jezik visokog nivoa sposoban da pruži brz razvoj programa i prototipova kao i laku vizuelizaciju podataka i procesiranja

Rlab

Rlab predstavlja interaktivni interpretiran program za numerička izračunavanja a jezgro njegovog programskega jezika je napisano od strane Ian Searle. Rlab (programske jezik) predstavlja jezik visokog nivoa sposoban da pruži brz razvoj programa i prototipova ka i laku vizuelizaciju podataka i procesiranja. Rlab nije dizajniran kao klon Matlab-a. Međutim, pošto je Rlab (program) sposoban da pruži dobro eksperimentalno okruženje za rad sa matricama, programski jezik poseduje slične koncepte i operacije tako da se može nazvati Matlab klonom. Rlab je pozajmio neke od najboljih osobina Matlab jezika ali ih je pružio kroz drugačiju sintaksu koje su modifikovane kako bi bile izražajnije a smanjila dvomislenost.

Za više informacija posetite linkove:

- Rlab Home Page
- Rlab Reference Manual
- Rlabplus

Octave

Octave je kompjuterski program za vršenje numeričkih proračuna. On je većim delom kompatibilan sa Matlab-om. Kao deo GNU Projekta, predstavlja besplatan softver pod uslovima GNU licence. Projekat je započet 1988. U početku je predviđeno da on bude sastavni deo kursa za razvoj hemijskih reaktora. Pravi razvoj je započet od strane John W. Eaton 1992. Prva alfa verzija je nastala 04 Januara 1993. Program je dobio ime po profesoru Octave Levenspiel-u.

Za više informacija:

- Octave Home Page
- Octave Documentation Page
- Octave Download Page

SCILAB

Scilab je naučni softverski paket za numerička izračunavanja koji pruža moćno okruženje za inženjerske i naučne aplikacije

Scilab je softverski paket za numeričke proračune razvijen početkom 90-tih od strane istraživača sa INRIA instituta. INRIA (engl. **National Institute for Research in Computer and Control Sciences**) je Francuski nacionalni institut za istraživanja sa glavnim fokusom na kompjuterske nukle, teoriju kontrole kao i primenjenu matematiku. Nakon osnivanja Scilab konzorcijuma u Maju 2003 on se održava i razvija od strane INRIA instituta. Scilab je najbolji Matlab klon. On ima dosta prednosti, uključujući i odličnu dokumentaciju kao i odličnu podršku (putem e-mail-a kao i newsgroup-a). Na sreću Windows korisnika, Scilab ne pruža samo source kod već i Windows binary. U svakom slučaju Scilab je najkompatibilniji sa Matlab-om. On čak poseduje i Matlab-to-Scilab prevodioce.

Scilab je naučni softverski paket za numerička izračunavanja koji pruža moćno okruženje za inženjerske i naučne aplikacije. Scilab je besplatan softver. Trenutno se koristi u obrazovnom i industrijskom okruženju širom sveta. Za Scilab je ogovoran Scilab konzorcijum koji je oformljen Maja 2003. Trenutno se Scilab konzorcijum sastoji od 18 članova. Scilab sadrži hiljade matematičkih funkcija sa mogućnošću interaktivnog dodavanja programa na različitim jezicima (FORTRAN, C, C++, JAVA...). On poseduje sofisticiranu strukturu podataka (uključujući liste, polinomijalne, racionalne funkcije, linearne sisteme...), interpreter i programske jezike visokog nivoa. Scilab je zamišljen da bude otvoren sistem gde korisnik može definisati nove tipove podataka i operacije nad ovim podacima.

Za više informacija posetite linkove:

- Scilab Home Page

- Scilab Documentation and Support Page
- Scilab Online Documentation
- Scilab Signal Processing Toolbox Functions
- Scilab Download Page

SCILAB ALATI

Scilab koristi brojne alate kako bi korisnicima obezbedio lakši i efikasniji rad

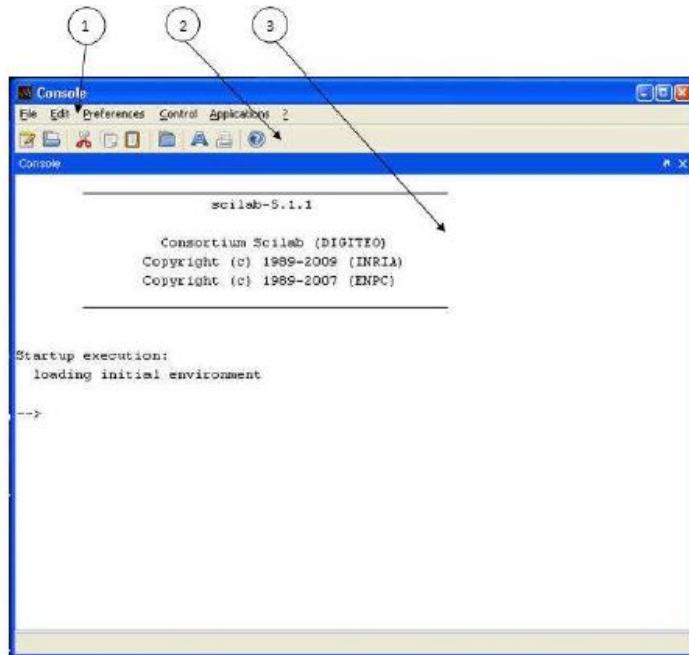
Postoje brojni alati koji su dostupni za SCILAB sistem:

- 2-D i 3-D grafika, animacija
- Linearna algebra, razuđene matrice
- Polinomijalne i racionalne funkcije
- Interpolacija i aproksimacija
- Simulacija: eksplicitni i implicitni sistemi za rešavanje diferencijalnih jednačina
- Scicos: alat za simulaciju i modelovanje hibridnih dinamičkih sistema
- Klasična i robustna kontrola, LMI optimizacija
- Diferentna i nediferentna optimizacija
- Procesiranje signala
- Grafovi i mreže
- Paralelni Scilab
- Statistika
- Interfejs za povezivanje sa Maple sistemom za kompjutersku algebru.

POKRETANJE SCILAB-A

U Scilab-u osnovne komande se mogu unositi u komandnom prozoru, takozvanoj konzoli (eng. Console)

Na vrhu prozora nalazi se linija sa menijima (broj 1.), a odmah ispod nje linija sa osnovnim alatima za rad (2.). Ispod osnovne linije sa alatima nalazi se deo radnog prostora - komandni prozor - **Console** (3.), rezervisan za unos komandi.



Slika 9.1.1 Pokretanje SciLAB-a

✓ 9.1 Vežba: Promenljive i osnovne operacije u SciLAB-u

PROMENLJIVE I OSNOVNE OPERACIJE

Promenljive u Scilab-u, za razliku od programskega jezika, nije potrebno deklarisati

Predviđeno vreme pokazne vežbe je 30 minuta.

Promenljive u Scilab-u, za razliku od programskega jezika, nije potrebno deklarisati. **Svaka nova kombinacija slova predstavlja novu promenljivu za koju Scilab rezerviše potrebnu memoriju.** Unošenje promenljive u radni prostor odvija se po sledećem obrascu:

promenljiva = vrednost ;

nakon čega se pritiska taster **Enter**. Na primer, promenljiva a, čija je vrednost 3, unosi se na sledeći način:

-->**a = 3;**

nakon čega se pritiska taster **Enter**.

Vrednost već korišćene promenljive dobija se unosom imena promenljive u komandni prozor i pritiskom na taster **Enter**. Za navedenu promenljivu a to izgleda ovako:

-->a

Na ekranu će se pojaviti:

a =

3.

nakon čega se iza prompta --> može uneti naredna komanda. Promenljiva se može uneti i tako da se izostavi znak „;“ na kraju komande. Tada Scilab, kao potvrdu u nastavku ispisuje vrednost promenljive:

--> a = 3

a =

3.

-->

DEFINISANJE FUNKCIJA U SCILAB-U

U programiranju pojам funkcije se koristi za posebnu celinu u programu koja ima za zadatак да користећи алгоритам трансформише одређене податке и time како излаз добије резултат

Pojam funkcije se može sresti u programiranju i u matematici. U programiranju pojам funkcije se koristi za posebnu celinu u programu koja ima za zadatak da koristeći algoritam transformiše određene podatke i time kao izlaz dobije rezultat. Svaka funkcija za ulaz koristi parametre kao svoje ulazne podatke, koje obrađuje i rezultat funkcije predstavlja izlazni podatak.

Funkcije se koriste kako bi se razdvojili programski kod u manje логичке celine kako bi se lakše upravljalо читавим programom. Deljenjem koda na više manjih celina omogućava njegovim lakšim upravljanjem, па самим tim i pronalaženjem i greška u kodu.

Kao primer data je prosta funkcija - konvertovanje valute eura (e) u dinare (d).

-->**function d=dinar(e,t); d=e*t; endfunction**

-->**dinar(200,120)** ans = 24000.

Veoma korisne numeričke funkcije su funkcije realnih varijabli. Na primer, dve funkcije f i g su definisane koristeći sledeće komande:

-->**function y=f(x); y=36/(8+exp(-x)); endfunction**

-->function y=g(x); y=4*x/9+4; endfunction

Definisane funkcije mogu biti korišćene za računanje vrednosti:

--> f(10) ans = 4.4999745

--> g(12.5) ans = 9.5555556

FUNKCIJE ZA RAD SA POLINOMIMA

Polinom definišemo na sledeći način $x = \text{poly}(0, 'x')$

Postoji veliki broj funkcija za rad sa polinomima, počev od osnovnih operacija kao što su množenje i deljenje sve do nalaženja nula polinoma, diferenciranja:

$p(x) = 2x^3 - 3x^2 + x + 9$

Unos polinoma u komandni prozor se započinje definisanjem nepoznate promenjive polinoma. To je u našem slučaju promenjiva x . Polinom definišemo na sledeći način:

--> x = poly(0, 'x')

x =

x

-->

Posle ispisa ove komande pritiskamo taster enter. Nakon toga pišemo polinom tako što između definisane nepoznate i koeficijenta stavljamo znak za množenje '*' a stepen polinoma označavamo sa '^' nakon čega pišemo veličinu stepena. Za gore navedeni polinom pisaćemo:

--> p=2*x^3-3*x^2+x+9;

Kraj komande označava se sa „;“. Obrazac unosa je sledeći:

polinom = [koeficijenti uz promenljive]*[promenljiva]^ [stepen];

Ako se polinom unosi bez znaka „;“ na kraju komande, Scilab automatski ispisuje zadate vrednosti. Na primer, polinom:

$p(x) = 2x^3 - 3x^2 + x + 9$

unosi se na sledeći način:

--> p=2*x^3-3*x^2+x+9

p =

$9 + x - 3x^2 + 2x^3$

-->

PRIMER

Uz pomoć funkcija izvršiti množenje i deljenje polinoma

Zadatak:

Uz pomoć funkcija izvršiti množenje i deljenje sledećih polinoma:

$$p(x) = 2x^3 - 3x^2 + x + 9$$

i

$$q(x) = 3x^3 + x^2 + x - 10$$

Rešenje:

Unos polinoma $p(x)$ i $q(x)$:

--> $p=2*x^3-3*x^2+x+9$

--> $q=3*x^3+x^2+x-10$

Za množenje se koristi znak *, pa je za unete polinome $p(x)$ i $q(x)$:

--> $r = p*q$

r=

2 3 4 5 6

-90 - x + 40x + 5x + 2x - 7x + 6x

-->

Slično se za deljenje polinoma koristi operacija **pdiv**:

--> $s = \text{pdiv}(p,q)$

s =

0.6666667

-->

Sa r i s , označeni su polinomi dobijeni množenjem, odnosno deljenjem $p(x)$ i $q(x)$, a dobijene vrednosti predstavljaju, takođe, odgovarajuće koeficijente.

DEFINISANJE NIZA

Operator ":" omogućava definisanje vektora brojeva

Operator ":" omogućava definisanje vektora brojeva čije su koordinate poznate aritmetički niz. Potrebno je zadati: **<<početnu vrednost: broj koraka : završnu vrednost>>**. Ponekad je moguće da je završna vrednost nije definisana. Ako broj koraka nije definisana, podrazumevana vrednost je 1.

Na primer, definisanje niza vektora brojeva koji se povećavaju za jedan, od tri do deset.

-->**3:10**

ans =3. 4. 5. 6. 7. 8. 9. 10.

Primer sledeći je povećavanje niza za 2. Početak niza je od 1 kraj niza je 10.

-->**1:2:10**

ans =3. 5. 7. 9.

Zadatak za samostalni rad

Koristeći SCILAB potrebno je rešiti sledeće zadatke:

- 1) Ispisati brojeve od 1 do 50 povećavajući za 7.
- 2) Definisati funkciju koja će kreirati niz od 50 do 20000 povećavajući za 50

UNOS MATRICA

Elementi vrste razdvajaju se razmacima ili zarezima, a kraj vrste označava se sa „;“

Unošenje matrica u radni prostor Scilab-a razlikuje se od unošenja polinoma. Elementi vrste razdvajaju se razmacima ili zarezima, a kraj vrste označava se sa „;“. Elementi sledeće vrste unose se u nastavku. Svi elementi matrice nalaze se između zagrada „[“ i „]“.

Obrazac na osnovu koga se matrica unosi u komandni prozor je :

ime_matrice = [elementi prve vrste ; elementi druge vrste; ...; elementi n-te vrste]
;

Kao i za promenljive i polinome, ako se izostavi znak „;“ na kraju komande, prikazuje odgovarajuću matricu.

PRIMER 2

Prikaz unošenja matrica

Zadatak:

Uneti u radni prostor sledeću matricu

$$A = \begin{bmatrix} 2 & 7 & -8 & 9 \\ 1 & 0 & 9 & 4 \\ 15 & 8 & 0 & 1 \\ 1 & 1 & 2 & 1 \end{bmatrix}$$

Rešenje:

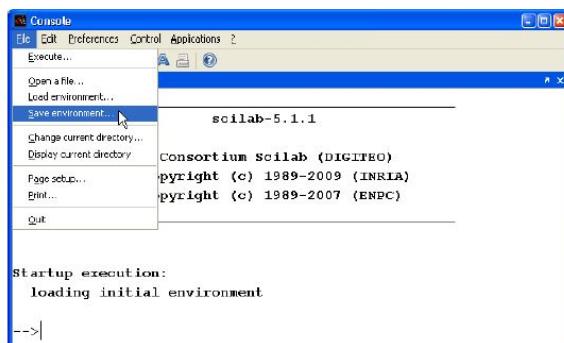
Matrica A se na osnovu datog obrasca unosi ovako

```
-->A=[2 7 -8 9;1 0 9 4;15 8 0 1;1 1 2 1]
A =
2.    7.   - 8.    9.
1.    0.    9.    4.
15.   8.    0.    1.
1.    1.    2.    1.

-->
```

Slika 9.2.1 Unos zadate matrice

Za snimanje unetih podataka koristi se opcija **File→Save Environment**. Radni prostor se snima u formatu **.sav**.



Slika 9.2.2 Snimanje unetih podataka

TABELA FUNKCIJA

Tabela često korišćenih funkcija

Funkcija	Svrha
help	pomoć za korišćenje određene naredbe
clc	brisanje sadržaja komandnog prozora
roots(p)	određivanje nula polinoma p
pdiv(p,q)	delenje polinoma p i q
detr(p)	izračunavanje determinante polinoma p
poly(A, prom)	nalaženje karakterističnog polinoma matrice A
size(A)	određivanje dimenzija matrice A
[k,l] = find(A)	nalaženje indeksa elemenata matrice A koji su različiti od nule
det(A)	izračunavanje determinante matrice A
rank(A)	određivanje ranga matrice A
inv(A)	određivanje inverzne matrice A
eye(n)	generisanje jedinične matrice dimenzija $n \times n$
ones(n)	generisanje matrice dimenzija $n \times n$ čiji su svi elementi jednaki jedinici

Slika 9.2.3 Tabela-1 Tabela funkcija

OPERATORI POREĐENJA

Kada želimo da uporedimo iskaze i znamo da li je upoređenje tačno, koristimo operatore poređenja

Kada želimo da uporedimo iskaze i znamo da li je upoređenje tačno, koristimo operatore poređenja. Operatori poređenja su:

- Jednako ==
- Različito <>
- Manje od <
- Veće od >
- Manje ili jednako <=
- Veće ili jednako >=

Takođe, možemo obaviti i logičke operacije sa operandima:

- Tačno %T
- Netačno %F
- I &
- Ili |
- Ne -

Ukoliko želimo da uporedimo dva vektora (ili dve matrice), „==“ i „<>“ će uporediti deo po deo, u ovom slučaju broj po broj iz oba niza.

Na primer:

-->**X=[1,2,5]; Y=[5,3,5];** -

-->X==Y ans =F F T

To znači da se upoređivalo da li je $1==5$, $2==3$, $5==5$.

Može se takođe koristiti i rezervisana reč za upoređivanje dva vektora ukoliko su jednaka i to: `isequal` i ukoliko nisu jednaka `-isequal`.

-->isequal(X,Y)ans =F

I da li nisu jednaka:

-->-isequal(X,Y) ans = T

▼ 9.2 Zadaci za samostalni rad: SciLab

ZADATACI ZA SAMOSTALNI RAD - MANIPULACIJA POLINOMA

Korišćenje SCILABA za manipulaciju polinomima

Predviđeno vreme izrade zadatka 1 i zadatka 2 je 10 minuta.

Zadatak 1

Definisati polinom koji ima sledeće korene: $x_1 = -1$ and $x_2 = 2$.

--> p=poly([-1 2],'x','r')

--> p =

-2 -x +x²

Rezultat je polinom: **p(x)=-2-x+x²**

Zadatak 2

Definisati dva polinoma, p_1 i p_2 i izvršiti njihovo sabiranje, oduzimanje, množenje i deljenje

p1=poly([-1 2],'x','r');

p2=poly([3 -3 -8 7],'x','c');

Sabiranje polinoma:

--> p1+p2

ans =

1 -4x -7x² +7x³

Oduzimanje polinoma

--> p1-p2

ans =

-5 +2x +9x² -7x³

Množenje polinoma

--> p1*p2

ans =

-6 +3x +22x² -9x³ -15x⁴ +7x⁵

Deljenje polinoma

--> **pdiv(p1,p2)**

ans =

0.

ZADATAK ZA SAMOSTALNI RAD - KORIŠĆENJE MATRICA

Korišćenje SCILABA za matrice

Predviđeno vreme izrade zadatka je 5 minuta.

Zadatak 1

Pomnožite dve matrice, A i B, gde je:

A=[1,2,3;4,5,6] i B=[1;1;2]

--> **A=[1,2,3;4,5,6]**

A = 1. 2. 3. 4. 5. 6.

-->**B=[1;1;2]**

B = 1. 1. 2.

-->**A*B**

ans =

9.

21.

✓ 9.3 Pokazna vežba: Programsко iscrtavanje - Scilab

PROGRAMSKO ISCRTAVANJE - SCILAB

Tabela prikazuje sedam primer koda za iscrtavanje neke funkcije po koracima

Predviđeno vreme pokazne vežbe je 20 minuta.

Proces konstrukcije grafika i potrebne stavke za njegov prikaz su opisane u sledećoj tabeli. Tabela prikazuje sedam primer koda za iscrtavanje neke funkcije po koracima.

Ako želimo da izvršimo samo analizu i pogledamo podatke na grafiku, biće nam potrebna samo prva tri koraka iz tabele. Međutim, ako kreiramo graf namenjen za prezentaciju i želimo da izvršimo lepo podešavanje pozicije grafa na strani, podesimo linijski stil i boju, dodamo anotaciju, moramo iskoristiti ostala stavke tabele.

Korak	Kod
1. Priprema vrednosti	x = 0:0.2:12; y1 = besselj(1,x); y2 = besselj(2,x); y3 = besselj(3,x);
2. Poziciranje regionala u prozoru gde će biti iscrtan grafik	hf = figure; subplot(2,2,1)
3. Poziv elementarne funkcije za iscrtavanje (2D)	h = plot(x,y1,x,y2,x,y3);
4. Selekcijska linija i davanje određenih karakteristika	set(h,'LineWidth',2,{ 'LineStyle'},{'--',':', '-'}) set(h,{ 'Marker'},{'none';'o';'x'}) set(h,{ 'Color'},{'r','g','b'})
5. Uređivanje ose i aktivacija grid linije	axis([0 12 -0.5 1]) grid on
6. Anotiranje grafa sa labelom na osi, legendom i tekstrom	xlabel('Vreme') ylabel('Amplituda') legend(h,'Prvi','Drugi','Treci') title('Bessel funkcija') [y,ix] = min(y1); text(x(ix),y,'Prvi Min \desnastrelica',... 'HorizontalnoPoravnanje','desno')
7. Ekstrakcija grafika	set(hf,'PaperPositionMode','auto') print -depsc -tiff -r200 myplot

Slika 9.3.1 Tabela-1 Osnovni koraci iscrtavanja plot funkcije

KREIRANJE LINIJA

Ako specificiramo dva vektora kao argumente funkcije plot(x,y), rezultat je graf y naspram x

Plot funkcija poseduje različite oblike koji zavise od ulaznih argumenata. Na primer, ako je y vektor, komanda plot(y) daje za rezultat linearni graf elemenata y-a. Ako specificiramo dva vektora kao argumente funkcije plot(x,y), rezultat je graf y naspram x.

Zadatak

Kreirati vektor sa vrednostima u rangu [0, 2 pi] sa inkrementom pi/100, a potom iskoristiti vektor da ograničimo sinusnu funkciju po tom rangu. MATLAB iscrtava vektor po x-osi kao i vrednost sinusne funkcije na y-osi.

Rešenje:

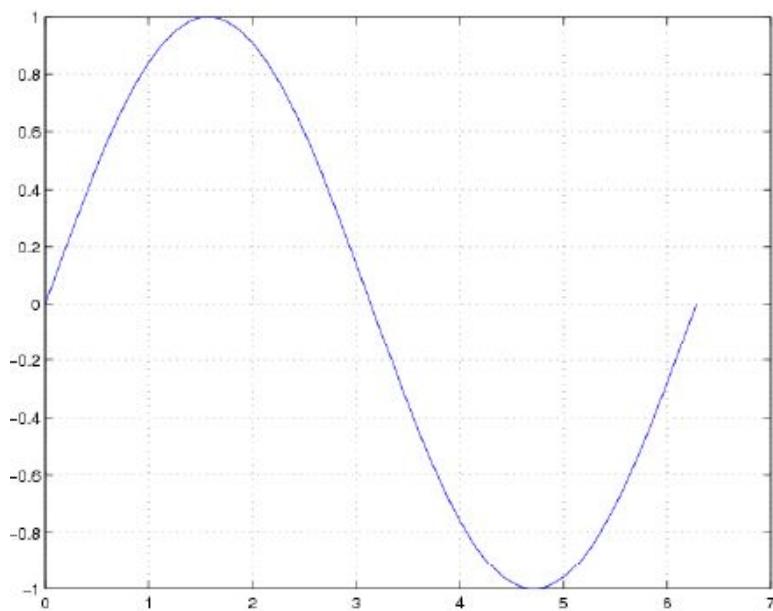
t = 0:pi/100:2*pi;

y = sin(t);

plot(t,y)

grid on % Uključuje grid linije za iscrtavanje

Automatski su adekvatno izbaždarene ose pri prikazu rezultata funkcije:



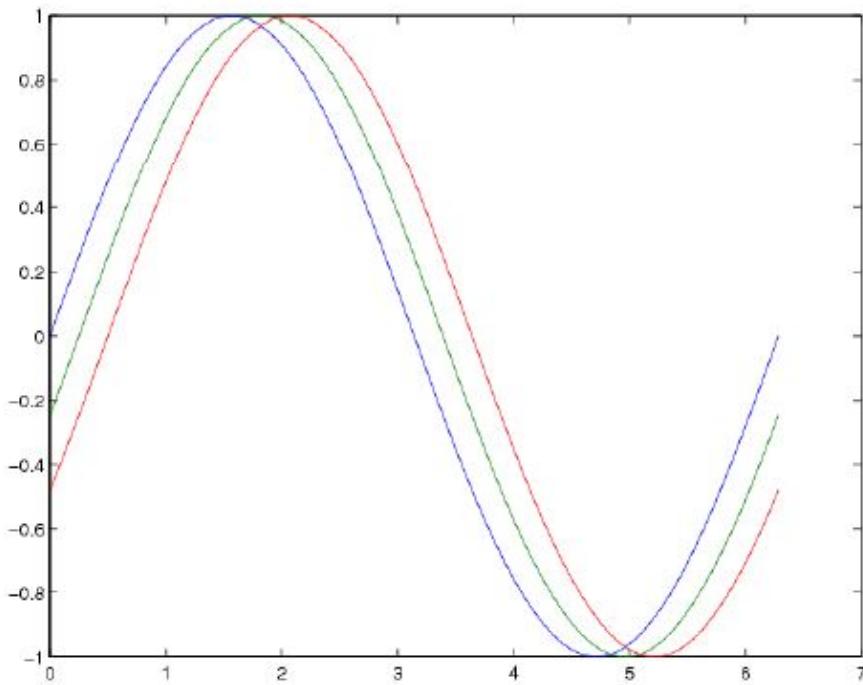
Slika 9.3.2 Sinusna funkcija $y = \sin(t)$

ISCRTAVANJE VIŠE GRAFOVA

Više grafova se može iscrtati funkcijom $\text{plot}(t,y,t,y2,t,y3)$

Možemo iscrtati više grafova pozivom komande “`plot`” i korišćenjem x-y parova. Program automatski prolazi kroz predefinisanu listu boja (odredjenu `ColorOrder` podešavanjem) i time postiže različit vizuelni doživljaj menu određenim skupovima podataka. Isrtavanje tri krive kao funkcije promenljive t .

```
y = sin(t);  
y2 = sin(t-0.25);  
y3 = sin(t-0.5);  
plot(t,y,t,y2,t,y3)
```



Slika 9.3.3 Sinusne funkcije: $\sin(t)$, $\sin(t-0.25)$, $\sin(t-0.5)$

SPECIFIKACIJA STILIZOVANIH LINIJA

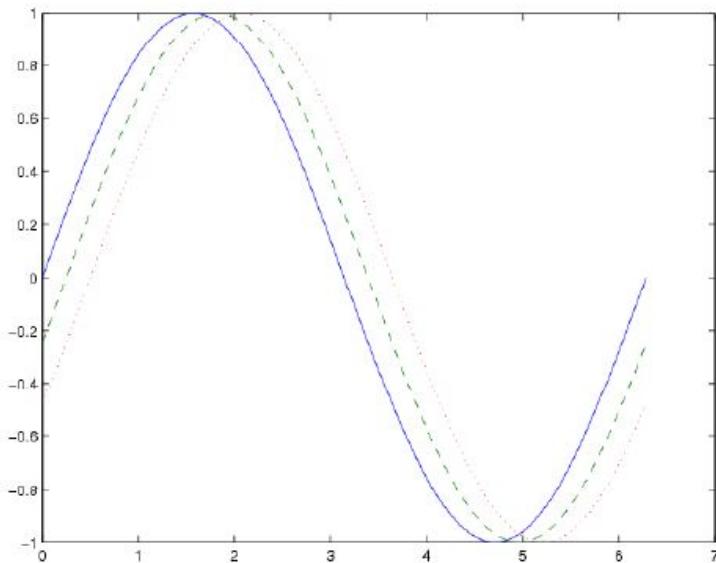
*Puna linija se definiše sa '-' , isprekidana sa '--' i tačkasta sa ':' -
`plot(t,y,'-',t,y2,'--',t,y3,:')`*

Pri iscrtavanju možemo naznačiti iscrtavanje sa različitim stilom linija za određene skupove podataka sa punim, ispresecanim ili tačkastim linijama.

Na primer:

```
t = 0:pi/100:2*pi;  
y = sin(t);  
y2 = sin(t-0.25);  
y3 = sin(t-0.5);  
plot(t,y,'-',t,y2,'--',t,y3,:')
```

Grafik prikazuje tri linije različitih boja i stila linija sinusne funkcije sa malom fazom pomeraja menu funkcijama, predočene kao y , y_2 i y_3 . Linije su definisane kao plava puna, zelena ispresecana i crvena tačkasta.



Slika 9.3.4 Isrtavanje sinusnih funkcija uz pomoć stilizovanih krivih

✓ 9.4 Zadaci za samostalni rad: SciLab iscrtavanje

ZADACI ZA SAMOSTALNI RAD - SCILAB

Iscrtavanje funkcija

Predviđeno vreme za izradu zadatka je 10 minuta.

Zadatak

Definišite dve funkcije f i g u intervalu $[-2, 5]$:

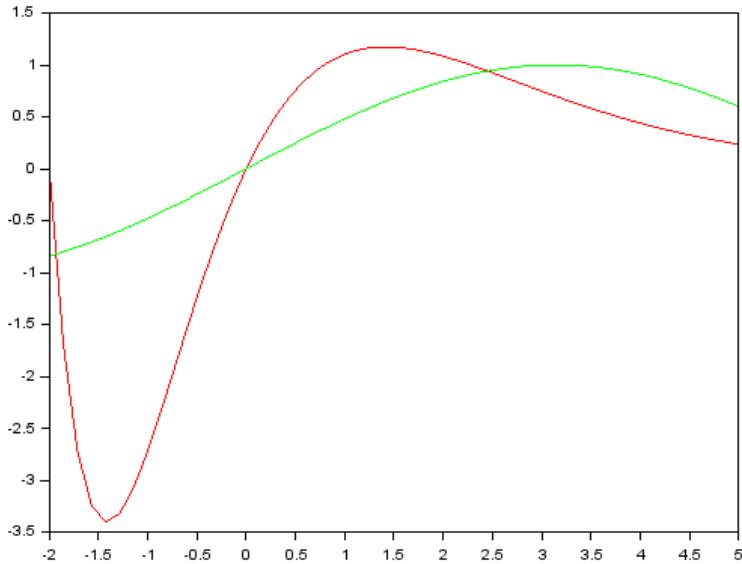
$f(x)=(x^2 + 2x)e^{-x}$ i $g(x)=\sin(x/2)$ i nacrtajte njihove grafike u različitim bojama.

```
-->function y=f(x)
> y=(x^2+2*x)*exp(-x)
> endfunction

--> x=linspace(-2,5,50);
--> function y=g(x)
> y=sin(x/2)
> endfunction

--> x=linspace(-2,5,50);

--> clf
--> plot(x,f,"r",x,g,"g")
```



Slika 9.4.1 IsCRTavanje funkcije iz zadatka za samostalni rad

▼ 9.5 Pokazna vežba: Programi za rad sa tabelama

PROGRAMI ZA RAD SA TABELAMA (SPREADSHEETS)

Rad sa tabelama postaje jedan od značajnih segmenata svakodnevnog korišćenja računara.

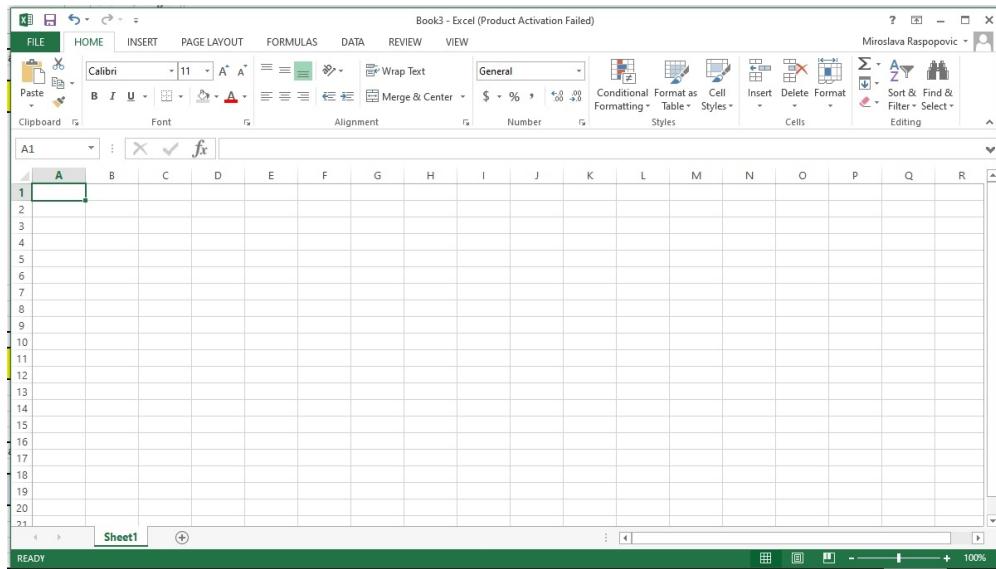
Predviđeno vreme pokazne vežbe je 20 minuta.

Rad sa tabelama postaje jedan od značajnih segmenata svakodnevnog korišćenja računara. Pojedini zadaci koji su do sada bili isključivo namenjeni programima za obradu teksta, počinju da se na jednostavan i efikasan način rešavaju uz pomoć tabela. Dva programa koja će biti obrađena u okviru ovih vežbi su MS Excel i OpenOffice.org Calc. Microsoft Excel je program za rad sa tzv. radnim tabelama (engl. [spreadsheet](#)) koji pruža mogućnosti različitih proračuna, crtanja grafika i sl. Zahvaljujući agresivnom marketingu, ovaj program je postao jedan od najpopularnijih programa današnjice. OpenOffice.org Calc je program sličan Excel-u. On nudi ekvivalentne opcije, ali i neke koje ne postoje kod Excel-a.

EXCEL

Osnovni elementi excel radnog prostora su naslovna linija, linija sa menijima i linija sa standardnim alatima

Excel se pokreće kao i sve ostale aplikacije Office paketa. Nakon pokretanja pojavljuje se prozor koji sadrži osnovne elemente: naslovnu liniju, liniju sa menijima, liniju sa standardnim alatima itd.



Slika 9.5.1 Prikaz Excel prozora

ELEMENTI EXCEL PROZORA

Osim uobičajenih, Excel prozor sadrži i neke elemente karakteristične za Excel:

Osim uobičajenih, Excel prozor sadrži i neke elemente karakteristične za Excel:

Element prozora	Svrha
Linija formula	Unošenje teksta u ćelije, naknadno uređenje teksta, kao i unošenje i menjanje formula.
Adresna linija	Opisuje položaj ćelije.
Zaglavljivo kolona	Za obeležavanje kolona za šta se koriste velika slova na vrhu.
Zaglavljivo vrsta	Za obeležavanje vrsta za šta se koriste brojevi sa leve strane prozora.
Označivač	Predstavlja okvir aktivne ćelije.

Slika 9.5.2 Tabela-1 Elementi Excel prozora

RADNE SVESKE I ĆELIJE

Svaki radni list čine ćelije razmeštene u mrežu koju čine vrste obeležene brojevima i kolone obeležene slovima

Ovde se za radni dokument koristi termin radna sveska. Prilikom pokretanja Excel-a može se otvoriti potpuno nova radna sveska ili koristiti neki iz skupa obrazaca, na sličan način kao u Wordu, prikazan u okviru druge vežbe. Inicijalno nova radna sveska ima tri radna lista označena u dnu prozora kao Sheet1, Sheet2 i Sheet3, a njihov broj se može povećavati ili smanjivati u skladu sa potrebama. Pokretanjem Excel-a dobija se radna sveska koja ima radne listove. Svaki radni list čine ćelije razmeštene u mrežu koju čine vrste obeležene brojevima i kolone obeležene slovima. Svaka ćelija ima adresu koju čine naziv kolone i vrste u čijem preseku se ćelija nalazi. Na primer, prva ćelija se nalazi u preseku kolone A i vrste 1, pa je njena adresa A1. Aktivna ćelija je ona oko koje se nalazi označivač i adresa te ćelije je navedena u adresnoj liniji.

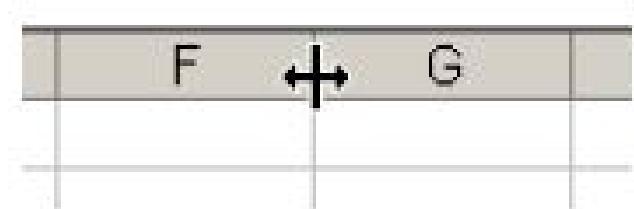
	A	B	C	D	E
1					
2					
3					
4					
5					
6					
7					

Slika 9.5.3 Prikaz aktivne ćelije

KORIŠĆENJE ĆELIJA

Za pomeranje označivača po radnom listu mogu se koristiti standardni načini uz pomoć miša tako što se klikne na željenu ćeliju ili strelica na tastaturi.

Za pomeranje označivača po radnom listu mogu se koristiti standardni načini uz pomoć miša tako što se klikne na željenu ćeliju ili strelica na tastaturi. Sadržaj se u ćeliju najčešće unosi uz pomoć tastature i to tako što se na željenu ćelije prvo klikne i odmah se počinje sa kucanjem. Druga mogućnost je da se klikne na datu ćeliju, potom da se klikne na liniju formula i da se na taj način unosi tekst. Nakon toga pritisne se taster Enter, pri čemu je tekst unet u datu ćeliju, a označivač sada obeležava ćeliju ispod. Ako je potrebno da se pređe u ćeliju sa desne strane koristi se taster Tab. Pravilno unošenje brojeva podrazumeva korišćenje brojnih oznaka od 0 do 9, kao i specijalne znake kao što su: + - , . \$ % itd. To znači da broj može uključivati i decimalne zareze, oznake valuta, procente i sl. Prilikom unošenja brojeva postoji mogućnost da ćelija nije dovoljno široka da bi broj bio prikazan pa se pojavljuju znaci #####. U tom slučaju širina kolone ili vrste se može podešiti tako što se cursor postavi na granici i držeći levi taster miša podešava veličina.



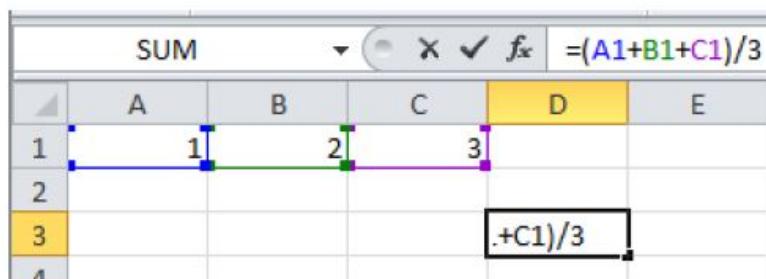
Slika 9.5.4 Podešavanje širine ćelije

FORMULE I FUNKCIJE

Formule se u Excel-u koriste za različita izračunavanja od vrlo jednostavnih do komplikovanih

Formule se u Excel-u koriste za različita izračunavanja od vrlo jednostavnih do komplikovanih. Na primer, ako je potrebno odrediti prosek vrednosti koje se nalaze u ćelijama A1, B1 i C1, obeležava se ćelija u kojoj treba da se nalazi rezultat i unosi sledeća formula: =(A1+B1+C1)/3

Potrebno je da svaka formula počne znakom jednakosti, jer u suprotnom Excel neće razumeti da se želi izračunavanje, već da se radi o npr. tekstu.



Slika 9.5.5 Unos formule

STANDARDNE OPERACIJE

U formulama se mogu koristiti aritmetičke operacije

Standardne operacije koje se mogu koristiti u formulama su:

Operator	Operacija	Primer	Rezultat
$^$	Eksponencija	=A1 $^$ 2	Datoj ćeliji dodeljuje vrednost koja je jednaka drugom stepenu vrednosti u ćeliji A1
$+$	Zbir	=A1+A2	Datoj ćeliji dodeljuje vrednost koja je jednaka zbiru vrednosti iz ćelija A1 i A2
$-$	Razlika	=A1-A2	Datoj ćeliji dodeljuje vrednost koja je jednaka razlici vrednosti iz ćelija A1 i A2
$*$	Množenje	=2*A2	Datoj ćeliji dodeljuje vrednost koja je jednaka dvostrukoj vrednosti iz ćelije A2.
$/$	Deljenje	=A1/10	Datoj ćeliji dodeljuje vrednost koja je jednaka deljenju vrednosti iz ćelija A1 sa 10

Slika 9.5.6 Tabela-2 Standardne operacije

REDOSEDLED IZVRŠAVANJA OPERACIJA

Eksponencijalne operacije imaju najveći prioritet

Operacije se izvode po sledećem redosledu:

1. eksponencijalne i relacije u zagradama,
2. množenje i deljenje, i
3. sabiranje i oduzimanje.

Na primer u izrazu: =C2+B8*4, prvo se izračunava vrednost B8*4, a zatim se rezultat dodaje vrednosti iz ćelije C2. Formule se unose na više načina. Prvi podrazumeva da se prvo klikne na ćeliju u koju će biti upisan rezultat. Potom se jednostavno otkuca znak =. U nastavku se unosi formula, koja se pojavljuje u liniji formula. Na kraju se pritisne Enter i Excel izračunava rezultat.

UNOS FORMULA

U ćeliji se nakon unosa formule prikazuje rezulat

Drugi način unosa formule podrazumeva izbor ćelija koje su deo formule uz pomoć miša. Prvo se obeleži ćelija gde će biti smešten rezultat. Upisuje se znak =, a potom se klikne na prvu ćeliju čija je adresa deo formule. Njena adresa se pojavljuje u liniji formule. Potom se unosi odgovarajući matematički operator koji se takođe pojavljuje u liniji. Dalje se kombinacijom unosa odgovarajućih operatora i klikom na željene adrese kreira potrebna formula. Na kraju se pritisne taster Enter.

Formula se, nakon unošenja u ćeliju, ne prikazuje. Umesto toga prikazan je rezultat. Ako je potrebno prikazati formulu neke ćelije, potrebno je kliknuti na datu ćeliju i formula će se prikazati u liniji formule. Za primer proseka vrednosti koje se nalaze u ćelijama A1, B1 i C1,

D1	A	B	C	D	E
1	1	2	3	2	
2					

Slika 9.5.7 Ćelija sa rezultatom

FUNKCIJE

Excel ima i složene gotove formule koje izvode niz operacija

Excel ima i složene gotove formule koje izvode niz operacija. Na primer za određivanje sume brojeva u ćelijama sa adresama od A1 do H1 može se koristiti funkcija: =SUM(A1:H1), umesto: =A1+B1+...+H1. Generalno može se reći da svaka funkcija ima tri osnovna dela:

1. znak =,
2. naziv funkcije, i
3. argumente, koji se navode između zagrade.

Neke često korišćene funkcije:

Funkcija	Primer	Rezultat
AVERAGE	AVERAGE(D4:D9)	Izračunava srednju vrednost brojeva koji se nalaze u ćelijama sa adresama od D4 do D9.
IF	=IF(A3>=10, A3*3, A3/2)	Ako je A3 veće ili jednako 10, vrednost u ćeliji se računa po obrascu A3*3, u suprotnom koriste se formula A3/2.
MAX	=MAX(C1:C10)	Ispisuje se najveća vrednost iz ćelija u rasponu od C1 do C10
MIN	=MIN(B1:B10)	Određuje se najmanja vrednost iz ćelija u rasponu od B1 do B10.

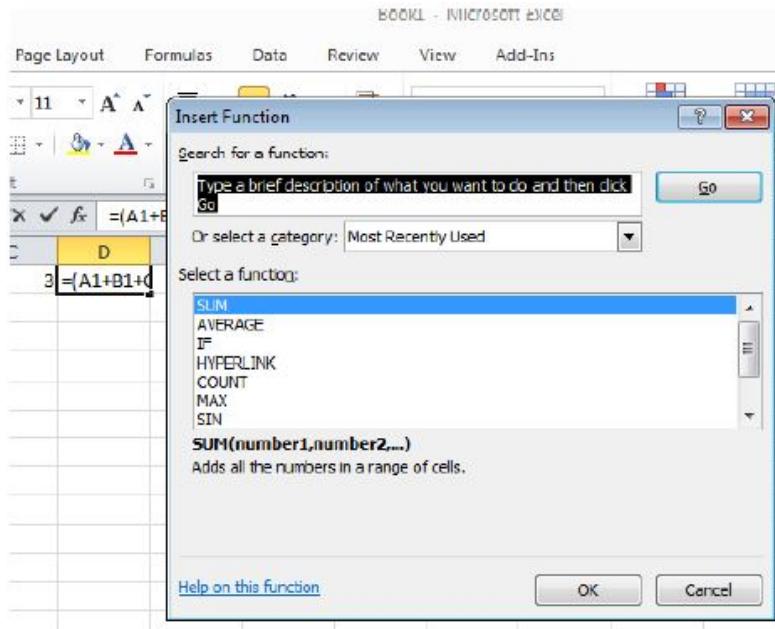
Slika 9.5.8 Tabela-3 Često korišćene funkcije

UNOS FUNKCIJA POMOĆU ČAROBNIJAKA

Osim direktnog upisivanja, funkcija se u ćeliju može uneti i uz pomoć čarobnjaka

Osim direktnog upisivanja, funkcija se u ćeliju može uneti i uz pomoć čarobnjaka, tačnije čarobnjaka funkcija (Function Wizard). Za korišćenje čarobnjaka potrebno je, pre svega, obeležiti ćeliju u koju je potrebno upisati rezultat. Zatim se klikne dugme Insert Function, koje se nalazi odmah pored linije formule i obeleženo je sa fx. Pojavljuje se novi prozor i iz dela: Select a function, bira željena funkcija. Ukoliko ta funkcija nije trenutno na spisku može se koristiti opcija za traženje Search for a function ili opcija traženja po kategorijama u polju: Or

select a category . U data polja može se uneti i raspon ćelija. Na kraju se klikne na dugme OK. Excel će datu funkciju uneti u označenu ćeliju i prikazati rezultat.



Slika 9.5.9 Unos funkcije pomoću wizarda

PRIMER

Radna tabela treba da omogući studentu da nakon unetog broja tačnih odgovora, dobije informaciju o broju tačnih odgovora, kao i automatsko računanje ostvarenih bodova tokom seme

Koristeći mogućnosti MS Excel ili OO.o Calc kreirati radnu tabelu za praćenje rezultata postignutih na testovima iz predmeta IT101, za svaku nedelju semestra. Smatra se da je student test položio ako je odgovorio tačno na najmanje 10 od ukupno 20 pitanja i na osnovu toga ostvaruje 1 poen na kraju semestra. Radna tabela treba da omogući studentu da nakon unetog broja tačnih odgovora, dobije informaciju o broju tačnih odgovora, kao i automatsko računanje ostvarenih bodova tokom semestra.

PRILAGOĐAVANJE ĆELIJA

Da formulu ne bi unosili u svaku ćeliju, dovoljno je kopirati sadržaj ćelije

Nakon pokretanja, u novom praznom listu, u prvu vrstu od ćelije B1 do P1, unose se vrednosti od 1 do 15 koje označavaju nedelje. U prvu kolonu se u ćeliji A1 unosi naziv predmeta: IT101). U ćeliju A2 unosi se tekst: „Tačnih odgovora“, a A3 tekst: „Netačnih odgovora“. U ćelije od B2 do P2 unosiće se vrednosti koje odgovaraju broju ostvarenih tačnih odgovora na testu svake nedelje. U ćelijama od B3 do P3, treba da se automatski generiše informacija o broju netačnih odgovora. Za unošenje odgovarajuće formule klikne se na ćeliju B3 i u liniju formula unese izraz: =20-B2, a potom klikne na Enter. Da formulu ne bi unosili u svaku ćeliju, dovoljno

je kopirati sadržaj ćelije B3. To se radi tako što se ponovo klikne na ćeliju B3, tako da se sada oko nje nalazi označivač. U njegovom donjem desnom uglu nalazi se mali crni kvadrat na koji je potrebno postaviti cursor i držeći levi taster miša povlačiti do ćelije P3. U svim ćelijama od B3 do P3 pojaviće se broj 20, a ako se klikne na bilo koju od njih u liniji formule pojaviće se odgovarajuća formula. Tako, ako se klikne na ćeliju I3 videćemo da se u njoj nalazi izraz: =20-I2 . To znači da je Excel automatski prilagodio izraze u svim ćelijama.

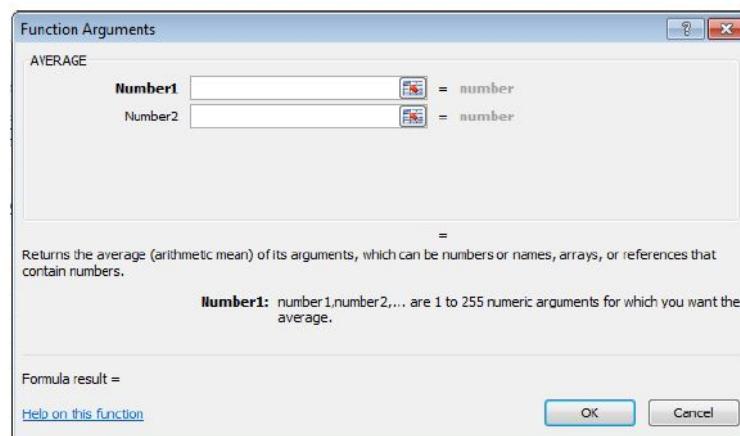
	A1	B1	C1	D1	E1	F1	G1	H1	I1	J1	K1	L1	M1	N1	O1	P1	IT101
1	IT101	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
2	Tačnih odgovora																
3	Netačnih odgovora	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
4																	
5																	

Slika 9.5.10 Primer prilagodjavanja ćelija

RAČUNANJE PROSEKA

Za računanje proseka koristi se funkcija AVERAGE

Na primer, ako želimo da pratimo prosečan učinak ostvarenih tačnih i netačnih odgovora možemo u ćeliju Q2 da unesemo funkciju koja automatski izračunava prosek za vrednosti od ćelija B2 do P2. Prvo kliknemo na Q2, a potom na opciju unosa funkcije. U prozoru koji će se pojaviti, u donjem delu, biramo funkciju AVERAGE i kliknemo na OK. U sledećem prozoru u polja treba uneti vrednosti za koje se traži srednja vrednost. Međutim, jednostavniji način je da se cursor, držeći levi taster miša prevuče od ćelije B2 do P2. U prozoru pojaviće se: B2:P2, a date ćelije biće uokvirene.



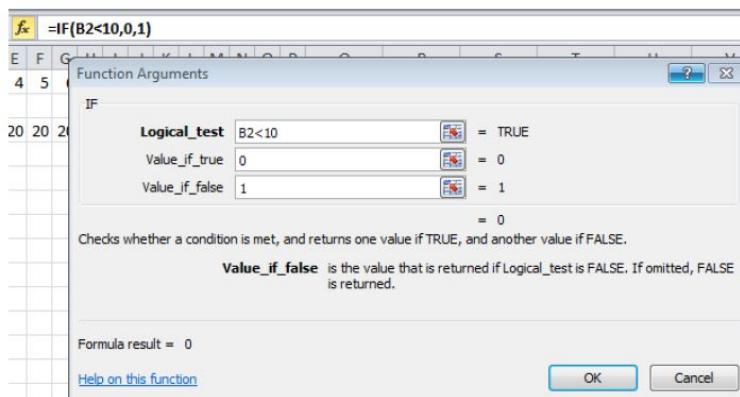
Slika 9.5.11 Argumenti funkcije

KORIŠĆENJE FUNKCIJA IF I SUM

Za računanje sume vrednosti u nizu koristi se funkcija SUM

U celiji Q2 pojaviće se rezultat: #DIV/0!, ili nešto slično, jer su u svim celijama datog opsega vrednosti 0. Da bi automatski računali i prosečan broj netačnih odgovora, dovoljno je da kopiramo vrednost celije Q2 na Q3, na već prikazani način. U Q3 će se pojaviti vrednost 20, jer je u svim celijama od B3 do P3 uneta ta vrednost. Zadatkom se traži da za svaki položen test student dobija po 1 poen na kraju semestra. Za automatsko praćenje broja poena, prvo ćemo u celiju A4 uneti tekst: „Poena“, a potom tasterom TAB preći u celiju sa desne strane B4. U ovom slučaju koristićemo funkciju IF koja će ovoj celiji automatski dodeliti vrednost 1, ako je test položen, odnosno vrednost 0 u suprotnom slučaju. Ponovo kliknemo na opciju za unos funkcije i biramo funkciju IF. U ovom slučaju novi prozor ima tri polja. U prvo unosimo logički test: B2<10, što odgovara slučaju kada je ostvaren manji broj poena od potrebnog za polaganje. U drugo polje unosi se vrednost koja će se dodeliti celiji ako je uslov iz prvog polja zadovoljen, odnosno kada test nije položen. U našem slučaju to je vrednost 0. U treće polje se zato unosi vrednost 1, za slučaj da uslov nije ispunjen, odnosno kada je test položen. Potom se klikne na OK.

Nakon unošenja funkcije, u polju B4, pojaviće se vrednost 0. Kopiranjem ove funkcije u sve celije do P4, na već prikazani način, ova funkcija se prilagođava vrednostima u drugim celijama. Na kraju, da bi student mogao da prati ostvareni učinak, u polje Q4 unosi se zbir svih vrednosti od B4 do P4. Za to se koristi funkcija SUM. A način njenog korišćenja odgovara korišćenju funkcija koje su prikazane u prethodnom delu. Rezultat je radni list koji student može da koristi za praćenje svojih predispitnih obaveza.



Slika 9.5.12 Unos argumenata funkcije

▼ 9.6 Zadaci za samostalni rad: Excel

ZADATAK ZA SAMOSTALNI RAD - EXCEL

Kreiranje osnovnih tabela u Excel-u

Predviđeno vreme izrade zadatka je 10 minuta.

Zadatak 1

Napraviti tabelu kao na slici, pri čemu se kolone vrednost i vrednost u Eurima izračunavaju automatski. Za vrednost Eura uneti srednji kurs NBS

1 Euro = 120.00 RSD

Naziv proizvoda	Cena	Količina	Vrednost	Vrednost u Eurima
Šećer	88	2	176.00 RSD	1.47 €
Brašno	40	5	200.00 RSD	1.67 €
Kafa	2	100	200.00 RSD	1.67 €
Mleko	3	80	240.00 RSD	2.00 €
Jogurt	4	90	360.00 RSD	3.00 €
Ukupno			1,176.00 RSD	9.80 €

Slika 9.6.1 Tabela-1 Tabela za zadatak za samostalni rad

✓ Poglavlje 10

Domaći zadatak

OPIS DOMAĆEG ZADATKA - DZ01

Uputstvo za izradu i slanje zadatka

Predviđeno vreme izrade domaćeg zadatka je 60 minuta.

- Radni prostor snimite kao **IT101-DZ01-Ime_Prezime_broIndexa**
- U radnom prostoru je potrebno da pošaljete dokumente (.doc ili .docx formatu) sa opisom i rešenjem svakog zadatka, kao i SCILAB i EXCEL dokumente za svaki zadatak (radni prostor, grafike i sl.)
- Napomena: Grafik ekstraktovati sa nazivom „**IT101-DZ01-Z<broj zadatka>-Ime_Prezime_broIndexa**“
- Sve datoteke je potrebno arhivirati u ZIP fajlu. Naziv arhiviranog fajla treba da bude **IT101-dz01_ime_prezime_brojIndexa.zip**
- **Domaći zadatak pošaljite predmetnom asistentu na e-mail,** a u subject-u mejla napisati **IT101 - DZ01**

ZADACI 1 - 3

Cilj ovih zadataka je da se provežba rad u SciLab-u

Zadatak 1.

Uz pomoć funkcija SCILAB-a izvršiti množenje i deljenje sledećih polinoma:

$$p(x) = 5x^3 - ax^2 + bx + c$$

i

$$q(x) = x^3 + x^2 + 2x - 1.$$

gde **koeficijente a, b i c određujete na osnovu broja indeksa**. Na primer, ako student ima broj indeksa 123, tada su vrednosti ovih koeficijenata a = 1, b = 2 i c = 3.

Zadatak 2.

- Koristeći SCILAB odrediti inverznu matricu matrice:

$$A = \begin{bmatrix} a & 1 & 1 \\ 3 & b & 1 \\ 1 & 1 & c \end{bmatrix}$$

gde su elementi a , b i c određeni na isti način kao i u delu Zadatka 1 ovog vežbanja.

Zadatak 3.

- Na osnovu objašnjenja u vežbanju o iscrtavanju funkcija, kreirati vektor sa vrednostima u rangu $[0, 2 \pi]$ sa inkrementom $\pi/100$, a potom iskoristiti vektor da ograničimo sinusnu funkciju po tom rangu. IsCRTati funkciju $y = \sin(t-1)$ gde je t gore pomenuti vektor. Urediti grafik i izvršiti ekstrakciju grafika.

ZADACI 4 - 6

Cilj ovih zadataka je da se provežba računanje polinoma i inverznih matrica u SciLabu

Zadatak 4.

Uz pomoć funkcija SCILAB-a izvršiti množenje i deljenje sledećih polinoma:

$$p(x) = 11x^3 - ax^2 + bx + c$$

i

$$q(x) = x^3 + 2x^2 + 3x - 15$$

gde koeficijente a , b i c određujete na osnovu broja indeksa. Na primer, ako student ima broj indeksa 123, tada su vrednosti ovih koeficijenata $a = 1$, $b = 2$ i $c = 3$.

Zadatak 5.

Odrediti inverznu matricu matrice:

$$A = \begin{bmatrix} a & 0 & 7 \\ 2 & b & 4 \\ 6 & 1 & c \end{bmatrix}$$

gde su elementi a , b i c određeni na isti način kao i u prethodnom zadatku.

Zadatak 6.

1) Ispisati brojeve od 1 do broja $<\text{abc}>$ povećavajući brojeve za $<\text{d+1}>$. gde a , b , c i d određujete na osnovu broja indeksa.

ZADATAK 7

Cilj ovog zadatka je da se provežba rad u Excel-u

Napraviti Excel tabelu sa podacima o radnim satima za mesec oktobar. Uneti redovne sate kao i prekovremene. Izračunati ukupne radne sate za dan kao i za svaku nedelju. Na kraju izračunati ukupne radne sate, redovne i prekovremene za ceo mesec oktobar. Na slici je prikazan primer tabele.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1		Nedelja 1			Nedelja 2			Nedelja 3			Nedelja 4			Nedelja 5		
2	Oktobar	Redovni sati	Prekovremeni sati	Ukupno sati	Redovni sati	Prekovremeni sati	Ukupno sati	Redovni sati	Prekovremeni sati	Ukupno sati	Redovni sati	Prekovremeni sati	Ukupno sati	Redovni sati	Prekovremeni sati	Ukupno sati
3	Ponedeljak															
4	Utorak															
5	Sreda															
6	Četvrtak															
7	Petak															
8	Subota															
9	Nedelja															
10	Ukupno	Oktobar ukupno redovnih sati:			0			Oktobar ukupno prekovremeno:			Oktobar ukupno prekovremeno:			0		
11																

Slika 10.1 Primer tabele

▼ ZAKLJUČAK

ZAKLJUČAK

U ovom predavanju smo definisali računarske discipline, dali smo definiciju pojma računarstva, sa specijalnim osvrtom na računarske discipline: softversko inženjerstvo, informacione tehnologije, informacioni sistemi, računarsko inženjerstvo i računarske nauke. Kako bi se razumelo zašto su određene tehnologije razvijene na specifičan način, treba razumeti način na koji ljudski mozak i ljudska percepcija funkcionišu, te smo razmotrili kognitivne nauke i uticaj drugih oblasti na informacione tehnologije.

Literatura

1. The Joint Task Force for Computing Curricula 2005, A volume of the *Computing Curricula Series, Computing Curricula 2005, September 2005*.
2. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, **Introduction to Algorithms, Second Edition**, The MIT Press, 2001.
3. Russell Shackelford i drugi članovi Joint Task Force for Computing Curricula 2005, **Computing Curricula 2005, Overview Report**, A cooperative project of The Association for Computing (ACM), The Association for Information Systems (AIS), The Computer Society (IEEE-CS), 2006.
4. Benjamin Martin Bly, David E. Rumelhart, **Cognitive Science**, Handbook of Perception and Cognition, Academic Press, 1999.
5. Paul Thagard, **Mind, Introduction to Cognitive Science**, A Bradford Book, The MIT Press, 2005.
6. Cognitive Science Websites, University of Waterloo, Canada. Web. 20 Nov 2009.



IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

ŠEME KODIRANJA I BROJNI SISTEMI

Lekcija 02

PRIRUČNIK ZA STUDENTE

IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

Lekcija 02

ŠEME KODIRANJA I BROJNI SISTEMI

- ✓ ŠEME KODIRANJA I BROJNI SISTEMI
- ✓ Poglavlje 1: Klase podataka
- ✓ Poglavlje 2: Brojni sistemi
- ✓ Poglavlje 3: Kodiranje podataka
- ✓ Poglavlje 4: Reprezentacija celih brojeva
- ✓ Poglavlje 5: Osnovne veličine u računarstvu
- ✓ Poglavlje 6: Logičke operacije
- ✓ Poglavlje 7: Pokazna vežba: Konverzija brojeva
- ✓ Poglavlje 8: Zadaci za samostalni rad: Konverzije
- ✓ Poglavlje 9: Pokazna vežba: Tablice istinitosti i komb. kola
- ✓ Poglavlje 10: Zadaci za samostalni rad: Logička kola
- ✓ Poglavlje 11: Pokazna vežba: Opšte i logičke funkcije u Excel-u
- ✓ Poglavlje 12: Zadaci za samostalni rad: Excel
- ✓ Poglavlje 13: Domaći zadatak
- ✓ ZAKLJUČAK

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

✓ Uvod

UVOD

Cilj ovog predavanja je da se obrade šeme kodiranja, brojni sistemi i osnovne logičke operacije

Cilj ovog predavanja je da:

- Predstavi klase podataka i brojne sisteme
- Objasni način na koji se podaci kodiraju i smeštaju u računaru
- Predstavi osnovne logičke operacije koje su važne u radu računara i programiranju

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

✓ Poglavlje 1

Klase podataka

KLASE PODATAKA U RAČUNARSKOM SISTEMU

Sa aspekta ljudske percepcije postoji potreba da računari predstave informacije koristeći bogatije oblike od simbola, kao što su rasterske slike, vektorski crteži, zvuk i film

Računari su danas sposobni da kreiraju i predstavljaju informacije u različitim oblicima. Informacije se najlakše predstavljaju simbolima. Pored slova i brojeva koji se najčešće koriste, upotrebljavaju se i drugi simboli kao što su, na primer, note, matematički simboli, matematički i interpunkcijski znaci i drugi specijalizovani simboli.

Međutim, sa aspekta mogućnosti ljudske percepcije informacija, simboli nisu uvek najbolji način za predstavljanje informacija. Tako je, na primer, čoveku mnogo lakše da primi informaciju o nekom muzičkom delu ako čuje njegov zvučni zapis nego ako gleda njegov notni zapis.

Zbog toga postoji potreba da računari predstave informacije u mnogo bogatijim oblicima kao što su rasterske slike, vektorski crteži, zvuk i film, koji se može smatrati i kombinacijom rasterskih slika i zvuka. Kako ovaj oblik informacija nije podesan za čuvanje i obradu unutar samog računarskog sistema, on se interno takođe predstavlja nekom kombinacijom simbola. Pri tom se koriste različiti matematički algoritmi za transformaciju informacija iz oblika pogodnog za čuvanje i obradu unutar računarskog sistema i oblika pogodnog za prikazivanje, izdavanje ili emitovanje.

Slika 1.1 Klase podataka koje se koriste u računarskim sistemima

POTREBA ZA PREZENTACIJOM

Sistem može smeštati podatke kao numeričke ili alfanumeričke podatke

Ako bi učenik trebalo da napiše koliko strana ima trougao, on bi napisao 3. Učenik iz perioda Rimskog carstva bi napisao III, a kineski učenik ≡. U sva tri slučajeva upotrebljen je neki simbol (3, III, ≡) koji reprezentuje vrednost tri. Različite kulture i nacije su tokom istorije čovečanstva razvile različite simbole za reprezentaciju realnih i apstraktnih pojmoveva.

Razume se da je nemoguće za svaki pojam koristiti poseban simbol jer bi ih bilo beskonačno mnogo. Tipičan primer je kineski alfabet koji ima preko 5000 simbola. Samo

mali broj Kineza zna značenje svih ovih simbola. Smisao reprezentacije je da sa što manjim brojem simbola reprezentuje što više pojmljiva.

Rad računarskih sistema je zasnovan na instrukcijama i podacima (Slika 2) . Broj instrukcija koje jedan računar može da izvršava nije mali. Takođe je potrebno u računar smestiti različite podatke. Oni mogu biti alfanumerički (na primer tekst) ili čisto numerički. Alfanumerički podaci se sastoje od slova (velikih i malih), brojeva i različitih simbola (znakovi interpunkcije, matematički simboli itd.). Kada se govori o slovima, treba imati na umu da postoje različiti jezici koji koriste različita pisma sa različitim brojem slovnih karaktera. Numerički podaci mogu biti pozitivni ili negativni, celi ili realni brojevi.

Sve instrukcije i podatke je potrebno na neki način efikasno interno reprezentovati unutar računarskog sistema. Budući da je rad računara zasnovan na bistabilnim elementima koji mogu da prepozna samo dva stanja, razvijeni su različiti metodi za internu reprezentaciju instrukcija i podataka.

Slika 1.2 Instrukcije i podaci i njihove moguće vrednosti

▼ Poglavlje 2

Brojni sistemi

TIPOVI BROJNIH SISTEMA

Postoje dve grupe brojnih sistema: pozicioni i nepozicioni

U računarstvu se često koriste različiti pojmovi za čije je razumevanje potrebno znanje o brojnim sistemima. Brojni sistem je skup pravila kojim se definiše način izražavanja kvantitativnih svojstava. Postoje dve grupe brojnih sistema:

- **Pozicioni**, kod kojih vrednost broja zavisi samo od pozicije cifara.
- **Nepozicioni**, kod kojih se vrednost broja određuje na osnovu vrednosti simbola od kojih se broj sastoji. Tipičan predstavnik nepozicionih brojnih sistema je rimski brojni sistem.

U bilo kom brojnom sistemu broj različitih cifara je jednak osnovi brojnog sistema. **Brojni sistem je definisan svojom osnovom.** Tako je osnova binarnog brojnog sistema 2, trinarnog 3, oktalnog 8, a heksadecimalnog 16. **Cifre brojnog sistema se kreću u intervalu od nula (0) do broja koji je za jedan manji od osnove brojnog sistema.** U tabeli 1 su date osnove i cifre nekih brojnih sistema.

Slika 2.1.1 Tabela-1 Osnove i cifre nekih brojnih sistema

U bilo kom pozicionom brojnom sistemu vrednost nekog celog broja se može prikazati kao

$$V = \sum_{i=1}^n a_i b^{i-1}$$

gde je:

V – vrednost broja

i - pozicija cifre brojano zdesna uлево

n - ukupan broj cifara u broju

b - osnova brojnog sistema

a_i - cifra na i-toj poziciji, $a \in (0, b-1)$.

✓ 2.1 Decimalni brojni sistem

CELI BROJEVI U DECIMALNOM BROJNOM SISTEMU

Baza ili osnova decimalnog brojnog sistema je 10, zato što se koristi deset simbola i to simboli 0-9

Čovečanstvo je tokom svoje istorije prihvatiло i naučilo da broji i računa u decimalnom brojnom sistemu. Osnova ili baza ovog brojnog sistema je deset i verovatno potiče od broja prstiju na rukama. U decimalnom brojnom sistemu se koristi deset simbola i to simboli 0-9. Broj u decimalnom brojnom sistemu se zapisuje kao niz cifara pri čemu, po konvenciji, krajnja desna cifra ima najmanju težinu, a krajnja leva najveću. Značaj neke cifre zavisi od njenog položaja u nizu, ali i od položaja decimalnog znaka (tačka ili zarez) ako je reč o decimalnom broju. Dakle, decimalni brojni sistem spada u pozicione.

Neka se, na primer, posmatra broj 2579 iz decimalnog brojnog sistema. On se može napisati i kao suma proizvoda cifara i težina mesta na kojima se cifre nalaze:

$$2579 = 2 \times 1000 + 5 \times 100 + 7 \times 10 + 9 \times 1$$

Bazirani na poziciji cifara u broju, broj 2459 ima devet jedinica (9×1), pet desetica (5×10), četiri stotine (4×100) i dve hiljade (2×1000). Ovaj primer je grafički prikazan na sledećoj slici, prikazuje kako su ove vrednosti u odnosu na poziciju izračunate, počevši sa cifrom sa najmanjom težinom i idući ka cifri sa najmanjom težinom. Svaka pozicija je označena od 0 pa naviše ($10^0, 10^1, 10^2$, itd.).

Dakle, vrednost nekog celog broja u decimalnom brojnom sistemu je data izrazom:

$$V = \sum_{i=1}^n a_i 10^{i-1}$$

gde je:

V - vrednost broja

i - pozicija cifre brojano zdesna uлево

n - ukupan broj cifara u broju

a_i - cifra na i-toj poziciji.

Slika 2.2.1 Primer reprezentacije decimalnog broja 2579

REALNI BROJEVI U DECIMALNOM BROJNOM SISTEMU

Realni brojevi se sastoje iz celobrojnog dela kao i razlomljenog dela.

Realni brojevi u decimalnom brojnom sistemu se sastoje od celobrojnog dela, koji se nalazi sa leve strane decimalnog separatora (zareza) i razlomljenog dela koji se nalazi sa desne strane decimalnog separatora. Položaj cifara se računa od decimalnog separatora, gde se za celobrojni deo težine povećavaju sa desna na levo, a za razlomljeni deo se povećavaju sa leva na desno. Realni brojevi mogu biti predstavljeni sa sledećim izrazom:

$$V = \sum_{i=1}^n a_i 10^{i-1} + \sum_{i=1}^m a_j 10^{-j}$$

gde je:

V - vrednost broja

i - pozicija cifre levo od decimalnog znaka, **j** - pozicija cifre desno od decimalnog znaka

n - broj cifara celobrojnog dela broja, **m** - broj cifara decimalnog dela broja

a_i - cifra na i-toj poziciji ulevo od decimalnog znaka

a_j - cifra na j-toj poziciji udesno od decimalnog znaka

Na primer, broj 165,423 se može napisati kao:

$$165,423 = 1 \times 10^2 + 6 \times 10^1 + 5 \times 10^0 + 4 \times 10^{-1} + 2 \times 10^{-2} + 3 \times 10^{-3}$$

Slika 2.2.2 Primer reprezentacije realnog decimalnog broja 165,423

✓ 2.2 Binarni brojni sistem

OPŠTI IZRAZ ZA BINARNI BROJNI SISTEM

Binarni brojni sistem koristi osnovu 2 i dve cifre: 0 i 1

Ma koliko ljudima izgledao podesan, decimalni brojni sistem je krajnje neprikladan sa aspekta računarske tehnike. Ovo je, pre svega, zbog toga što ne postoje procesi sa 10 jasno odvojenih(diskretnih) stanja. Međutim, postoji niz procesa koji se može definisati sa dva stanja: na primer, kroz provodnik teče struja(stanje 1) ili ne teče (stanje 0) ili je elektromagnetna sklopka uključena ili isključena. Ovakvi procesi se mogu opisati binarnim brojnim sistemom koji ima osnovu **b=2** i samo dve cifre: **0 i 1**. S obzirom na to da je rad računara zasnovan na logičkim, bistabilnim elementima, ovaj brojni sistem se koristi za internu reprezentaciju brojeva u računaru.

Opšti izraz za vrednost broja u binarnom brojnom sistemu je:

$$V = \sum_{i=1}^n a_i 2^{i-1}$$

Tako, na primer, broj 2579 decimalnog brojnog sistema u binarnom sistemu ima oblik:

$$\begin{array}{rcl} 101000010011 & = \\ 1 \times 2^{11} + 0 \times 2^{10} + 1 \times 2^9 + 0 \times 2^8 + 0 \times 2^7 + 0 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 \end{array}$$

Može se reći da je broj 101000010011 binarni ekvivalent decimalnog broja 2579, što se obično piše kao:

$$101000010011_2 = 2579_{10}$$

PRIMER KONVERZIJE IZ BINARNOG U DECIMALNI BROJNI SISTEM

1101 binarno = 13 decimalno

Primer na slici 1 prikazuje reprezentaciju binarnog broja 1101 čiji je ekvivalent u decimalnom brojnom sistemu broj 13. Sistem dodavanja težine poziciji je isti kao i decimalnom brojnom sistemu, tako da težine rastu gledajući s desna u levo. Iz ovog primera se vidi da je za zapis broja u binarnom brojnom sistemu potreban veći broj cifara nego u decimalnom sistemu. Uočava se, takođe, da je zbog čovekove naviknutosti na decimalni sistem binarni zapis neprikladan za čoveka. Ovo je posebno očigledno kada se radi sa velikim brojevima. Zbog toga se u računarskoj tehnici, a naročito u programiranju, češće koristi heksadecimalni brojni sistem.

Slika 2.3.1 Primer konverzije binarnog broja u decimalni

VIDEO - KONVERZIJA BINARNOG BROJA U DECIMALNI

Primer konverzije binarnog broja 1011 u decimalni broj

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

PRIMER KONVERZIJE BINARNOG BROJA U DECIMALNI - VIDEO

Konverzija binarnog broja u decimalni

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

PRINCIP KONVERZIJE DECIMALNOG U BINARNI BROJ

konverzija decimalnog u binarni brojni sistem se zasniva na deljenju.

Kako se konverzija iz binarnog u decimalni brojni sistem zasniva na množenju, gde smo binarnu cifru množili stepenom osnove, tako možemo zaključiti da se konverzija decimalnog u binarni brojni sistem se zasniva na deljenju.

Postupak:

- Decimalni broj delimo sa 2
- Pišemo ostatak na mestu najmanje težine, odnosno najmanje značajnosti
- Celobrojni deo dalje delimo sa 2
- Postupak se završava kada se u deljenju dođe do nule .

Prilikom izrade primera ćete primetiti da kada je broj paran, ostatak je uvek 0, a kad je neparan ostatak je uvek 1.

Primer

Konvertujemo broj $(156)_{10}$ u binarni broj.

$$156 / 2 = 78 \text{ - ostatak } 0 < \dots \text{ bit najmanje značajnosti}$$

$$78 / 2 = 39 \text{ - ostatak } 0$$

$$39 / 2 = 19 \text{ - ostatak } 1$$

$$19 / 2 = 9 \text{ - ostatak } 1$$

$$9 / 2 = 4 \text{ - ostatak } 1$$

$$4 / 2 = 2 \text{ - ostatak } 0$$

$$2 / 2 = 1 \text{ - ostatak } 0$$

$$1 / 2 = 0 \text{ - ostatak } 1 < \dots \text{ bit najveće značajnosti}$$

Tako dobijamo $(156)_{10} = (10011100)_2$

PRIMER KONVERZIJE DECIMALNOG U BINARNI BROJ

Konverzija decimalnog u binarni brojni sistem se zasniva na deljenju sa 2 i ostatku koji ostaje pri tom deljenju

Prikazaćemo kako da prevedemo broj 68 iz dekadnog u binarni broj. Konverzija se zasniva na deljenju sa osnovom binarnog brojnog sistema, odnosno sa 2.

Kada broj podelimo sa 2, važno je da vodimo računa da li pri deljenju ima ostatka ili ne. Ako postoji ostatak, tada se piše cifra 1, a ako ostastka nema, onda se piše cifra 0.

Dakle, kada podelimo željeni broj, koji je u ovom slučaju 68 sa 2, s obzirom da je 68 paran broj, ostatka nema i on je 0.

68:2=34 ostatak: 0

Prvi dobijeni ostatak, u ovom slučaju 0, biće cifra najmanje težine binarnog broja (LSB) i upisujemo je na poziciju najmanje težine.

Zatim, proces nastavljamo dalje i delimo sa 2, svaki sledeći dobijeni broj.

34:2=17 ostatak: 0

17:2=8 ostatak: 1

8:2=4 ostatak: 0

4:2=2 ostatak: 0

2:2=1 ostatak: 0

1:2=0 ostatak: 1

Postupak se završava kada se u deljenju dođe do nule (1:2=0, ostatak 1). Dobijeni binarni broj je: **1000100**

$$(68)_{10} = (1000100)_2$$

VIDEO - KONVERZIJA DECIMALNOG BROJA U BINARNI

Primer konverzije decimalnog broja 156 u binarni broj

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

REALNI BROJEVI U BINARNOM BROJNOM SISTEMU

Cilj ove sekcije je da student shvati kako se dobijaju realni brojevi u binarnom sistemu

Realni brojevi se u binarnom brojnom sistemu mogu prikazati na sličan način. Vrednost nekog realnog binarnog broja može se izračunati po izrazu

$$V = \sum_{i=1}^n a_i 2^{i-1} + \sum_{j=1}^m a_j 2^{-j}$$

gde je:

i - pozicija cifre levo od decimalnog znaka

j - pozicija cifre desno od decimalnog znaka

n - broj cifara celobrojnog dela broja

m - broj cifara decimalnog dela broja

a_i - cifra na i-toj poziciji u levo od decimalnog znaka

a_j - cifra na j-toj poziciji udesno od decimalnog znaka.

Treba primetiti da težine cifara levo od decimalnog znaka imaju vrednost kao što je prikazano u tabeli 1.

Na primer, broj 2349,62510 u binarnom brojnom sistemu izgleda

100100101101.101₂

KONVERZIJA DECIMALNOG REALNOG BROJA U BINARNI

$$0,2875 \text{ decimalno (10)} = 0,010010011 \text{ binarno (2)}$$

Konverzija decimalnog realnog broja u binarni se vrši tako što se posebno konvertuju celobrojni i decimalni deo. Celobrojni deo broja se konvertuje u binarni broj metodom koji se normalno koristi za cele brojeve.

Decimalni deo, koji ima opšti oblik 0,abcde..., se konvertuje prema sledećem algoritmu:

1. Decimalni broj 0,abcde se množi sa 2 i dobija se rezultat oblika p,qrstu
2. Cifra p postaje prva decimalna cifra binarne reprezentacije i ona se otklanja iz broja
3. Ostatak 0,qrstu se ponovo množi sa 2 kako bi se dobila sledeća decimalna cifra binarnog broja
4. Koraci 2 i 3 se ponavljaju sve dok ostatak ne bude nula ili dok se ne postigne željena preciznost.

Radi ilustracije ovog algoritma, prikazuje se postupak konverzije broja 0,2875 u binarni oblik (Tabela 1). Postupak konverzije je prekinut posle 9 koraka iako je ostatak bio 0,2 a ne 0. Ovo znači da vrednost binarnog broja 0,010010011₂ približno aproksimira vrednost decimalnog broja 0,2875₁₀. Ako se sada izvrši konverzija broja 0,010010011₂ u decimalni oblik, videće se da on ima vrednost 0,287109375₁₀. Dakle, razlika između vrednosti broja 0,2875₁₀ i njegove binarne reprezentacije sa 9 značajnih cifara 0,010010011₂ je 0,000390625. Ovo pokazuje da se u internoj reprezentaciji vrednosti realnih brojeva najčešće aproksimiraju.

Slika 2.3.2 Tabela-1 Postupak konverzije razlomljenog decimalnog broja u binarni

NORMALIZOVANI OBLIK

Cilj ove sekcije je objasniti normalizovani obliku binarnom sistemu

Normalizovani oblik nekog broja u naučnoj notaciji se može na sličan način predstaviti i u binarnom brojnom sistemu. U tom slučaju opšti oblik broja je:

$$\pm M * 2^E$$

Pri čemu celobrojni deo mantise C, koji je u ovom slučaju binarna cifra, može da uzme samo vrednost C=1, jer zbog izvršene normalizacije prva cifra ne može da bude nula. Kao što će kasnije biti pokazano, ova činjenica je iskorišćena da se sa istim brojem bitova dobije binarna reprezentacija veće preciznosti.

Na primer, broj $2349,625_{10}$ u binarnom brojnom sistemu pre normalizacije izgleda $100100101101,101_2$. Posle normalizacije ovaj broj dobija oblik $1,00100101101101_2 * 2^{1011}$. Decimalni zarez je pomeren za 11 mesta, pa je vrednost eksponenta $E = 11_{10} = 1011_2$.

Slično tome broj $0,2875_{10}$ koji u binarnom obliku pre normalizacije izgleda $0,010010011_2$, nakon normalizacije dobija oblik: $1,0010011_2 * 2^{-10}$, jer je zarez pomeren za 2 mesta udesno ($E = -2_{10} = 10_2$).

Iz oba primera se vidi da je jedina cifra celobrojnog dela 1, što je slučaj sa svim normalizovanim binarnim brojevima.

PRIMER KONVERZIJE DECIMALNOG REALNOG BROJA

Primer konverzije u binarni brojni sistem

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ 2.3 Heksadecimalni brojni sistem

OPŠTI IZRAZ ZA BINARNI BROJNI SISTEM

Heksadecimalni brojni sistem koristi osnovu 16 i simbole 0-9, A,B,C,D,E,F

Vrlo važan brojni sistem u računarskoj tehnici je i heksadecimalni brojni sistem, čija je osnova 16 i koji koristi simbole 0-9,A,B,C,D,E,F. Decimalna vrednost karaktera A je 10, B je 11, C je 12, D je 13, E je 14 i F je 15. Ovaj brojni sistem je pogodan za upotrebu u računarima zbog toga što suvrednosti kompaktnije i lakše za čitanje od brojeva u binarnoj reprezentaciji. S druge strane, konverzija između binarnih i heksadecimalnih brojeva je vrlo efikasna.

Vrednost broja u heksadecimalnom sistemu je data izrazom:

$$V = \sum_{i=1}^n a_i 16^{i-1}$$

Slika 1 ilustruje konverziju heksadecimalnog broja A15 u njegov decimalni ekvivalent 2581. Iz ovog primera se može videti da je heksadecimalni broj najkraći u poređenju sa reprezentacijom ovog broja u decimalnom i binarnom brojnom sistemu.

Slika 2.4.1 Reprezentacija heksadecimalnog broja A15

Broj 2579 iz decimalnog brojnog sistema ima u heksadecimalnom brojnom sistemu oblik

$$A13 = A \times 16^2 + 1 \times 16^1 + 3 \times 16^0$$

pa se može zapisati

$$101000010011_2 = 2579_{10} = A13_{16}$$

iz čega se vidi da je heksadecimalni zapis najkraći, dakle i najekonomičniji od sva tri opisana brojna sistema.

PRIMERI KONVERZIJE HEKSADECIMALNOG BROJA U DECIMALNI

Konverzija heksadecimalnog broja FA8 u decimalni

Na slici 2 dat je primer konverzije heksadecimalnog broja FA8 u decimalni, gde je prikazan postupak konverzije i konačno rešenje.

Slika 2.4.2 Primer konverzije heksadecimalnog broja FA8 u decimalni

Slika 3 daje prikaz primer konverzije heksadecimalnog broja 7988 u decimalni, gde je prikazan postupak konverzije i konačno rešenje.

Slika 2.4.3 Primer konverzije heksa decimalnog broja 7988 u decimalni

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 3

Kodiranje podataka

PREDSTAVLJANJE PODATAKA U RAČUNARU

Svaki simbol ima svoju internu binarnu reprezentaciju

Računar interno koristi binarni brojni sistem, jer njegova logika poznaje samo dva moguća stanja: uključeno (1) i isključeno (0). Cifra u binarnom brojnom sistemu se naziva bit (skraćenica od BInary digiT). Različiti simboli imaju svoje interne reprezentacije u računaru, što znači da svi oni imaju svoje predstavnike u binarnom obliku. To je razlog zašto je neophodno da se definišu različiti skupovi karaktera. Primer može biti skup karaktera engleskog alfabeta, srpske ćirilice, matematičkih simbola ili muzičkih simbola. Predstave alfanumeričkih podataka u računaru znači da je određeni skup znakova definisan i da je svakom znaku dodeljen broj ili ime, kao i njegova binarna reprezentacija (predstavljena nizom predefinisanih bitova).

Da bi se obezbedila jednoznačnost, uvode se pravila kodiranja podataka u binarni sistem. Binarno kodiranje je postupak kojim se simbol zamjenjuje nizom bitova po strogo propisanom sistemu kodiranja. Postoje različiti sistemi kodiranja i za različite namene, kao što su sistemi kodiranja numeričkih i alfanumeričkih simbola i podataka.

REPREZENTACIJA ALFANUMERIČKIH PODATAKA

Reprezentacija alfanumeričkih podataka u računaru podrazumeva da se najpre definije određeni set karaktera, a zatim da se svakom karakteru dodeli broj i binarna reprezentacija

Računarska praksa zahteva da se vrši obrada i vizualizacija različitih setova karaktera. Primer može da bude set karaktera engleskog alfabeta, srpske ćirilice, matematičkih simbola ili muzičkih simbola. Reprezentacija alfanumeričkih podataka u računaru podrazumeva da se najpre definije određeni set karaktera, a zatim da se svakom karakteru dodeli broj (ili ime) i binarna reprezentacija (kombinacija bitova).

Interna reprezentacija alfanumeričkih podataka je zasnovana na tri povezana koncepta:

1. Set karaktera ili repertoar je neuređena kolekcija karaktera koja može biti predstavljena numeričkim vrednostima.
2. Kodirani set karaktera preslikava karaktere iz seta karaktera u numeričke vrednosti.
3. Šema kodiranja karaktera definije reprezentaciju numeričkih vrednosti iz jednog ili više kodiranog seta karaktera u binarnom obliku.

3.1 ASCII kod

ASCII

ASCII kod omogućuje binarno i heksadecimalno kodiranje malih i velikih slova latinice, cifara, znakova i upravljačkih (komandnih) znakova

ASCII kod (American Standard Code for Information Interchange) opšte je prihvaćen kod i susreće se gotovo kod svih računara. Ovo je 7-bitni kod, što znači da se njime može prikazati 128 različitih karaktera. **ASCII kod omogućuje binarno i heksadecimalno kodiranje malih i velikih slova latinice, cifara, znakova i upravljačkih (komandnih) znakova.** Moguće je koristiti i osmi bit u ASCII kodu, ali samo za proveru pariteta. **U tom slučaju osmi bit se postavlja tako da ukupan broj jedinica u bajtu bude paran ili neparan za celu tabelu.** U tabeli 1 je prikazan ASCII kod.

ASCII set je inicijalno projektovan za prenos podataka sa udaljenih ulaznih terminala (engl. **teletypes**), tako da sadrži neke karaktere koji su danas nepotrebni i čudni za savremene korisnike. U doba kad je projektovan ASCII kod, osmi bit se koristio kao bit parnosti za proveru pariteta, kako bi se izbegle slučajne greške pri čitanju i prenosu podataka. **Kasnije je osmi bit iskorišćen za definisanje proširenog ASCII seta koji ima dodatnih 128 karaktera (dakle ukupno 256).** Ovi karakteri se koriste za kodiranje neengleskih slova, za grafičke i matematičke simbole. Ovaj set karaktera je definisan ISO standardima ISO 8859-1 do 8859-15.

Slika 3.1.1 Tabela-1 ASCII kod

U tabeli 1 su navedeni i specijalni znaci koji se koriste u procesu prenosa podataka. Najčešće korišćeni znaci su: **EOT-End Of Transmission**, **ACK-Acknowledge**, **BS-Backspace**, **HT-Horizontal Tabulation**, **LF-Line Feed**, **CR-Carriage Return**.

ANALIZA ASCII TABELE

Razlika između heksadecimalnih kodova istog malog i velikog slova je 2016, pa je vrlo jednostavno napisati program za konverziju velikih u mala slova i obratno

Analizom tabele ASCII mogu se uočiti neke osobine ovog seta koje mogu biti od praktične koristi prilikom pisanja programa za obradu teksta:

- Mala slova a do z su u intervalu od 9710 do 12210. Velika slova A do Z su u intervalu od 6510 do 9010. Na ovaj način se lako može detektovati da li je karakter malo ili veliko slovo.
- Razlika između heksadecimalnih kodova istog malog i velikog slova je 2016, pa je vrlo jednostavno napisati program za konverziju velikih u mala slova i obratno.

- Cifre 0 do 9 zauzimaju kodove 4810 do 5710. Na osnovu ovoga se lako može odrediti da li je uneti karakter cifra.

EBCDIC ([Extended Binary Coded Decimal Interchange Cod](#)) kod je 8-bitni kod, pa omogućava kodiranje šireg skupa znakova, ali zbog korišćenja osmog bita nije moguće vršiti proveru pariteta. Ovaj kod je uglavnom koristio IBM na svojim mainframe računarima.

▼ 3.2 Unicode

UNICODE STANDARD

Unicode standard je projektovan tako da bude univerzalan, efikasan i nedvosmislen

ASCII kod, koji je u osnovi osmobitni, ne omogućuje kodiranje više od 256 karaktera. Ovaj broj je nedovoljan za mnoga pisma (na primer za kineski Han). Zbog toga je 1988. pokrenut **Unicode** projekat čiji je cilj bio definisanje novog internacionalnog standarda za računarsku reprezentaciju karaktera koja će omogućiti jednostavno procesiranje teksta. Unicode standard je projektovan tako da bude:

- univerzalan, kako bi obuhvatio sve internacionalne, nacionalne i industrijske setove karaktera
- efikasan, tj. da omogući efikasno sortiranje, pretraživanje, editiranje i prikazivanje
- nedvosmislen, dakle jedan kod predstavlja samo jedan tačno određeni karakter.

Unicode je potpuno kompatibilan sa međunarodnim standardom ISO/IEC 10646 iz 2003. godine. Unicod standard je baziran na elementima teksta. Pri tom se kao element teksta uzima karakter. Karakter je najmanji tekstualni element pisma. On je apstraktni koncept kojim se predstavlja recimo slovo B, znak pitanja ili neka cifra. Svakom karakteru koji je definisan Unicode standardom se dodeljuje broj koji se naziva kodna tačka (engl. [code point](#)).

Kodni prostor predviđen Unicode standardom se prostire od 0 do 10FFFF16, što znači da je 1.114.112 kodnih tačaka na raspolaganju za dodeljivanje apstraktним karakterima. Označavanje neke kodne tačke u Unicode standardu se vrši navođenjem njene numeričke vrednosti u heksadecimalnom obliku sa prefiksom "U+". Referenciranje na neki kodirani karakter se vrši preko kodne tačke, ali se zbog nedvosmislenosti dodaje i zvanično Unicode ime karaktera. Na primer:

- U+0061 LATIN SMALL LETTER A
- U+10330 GOTHICS LETTER AHSA

Postoji sedam osnovnih tipova kodnih tačaka.

OSNOVNI TIPOVI KODNIH TAČAKA

Postoji sedam osnovnih tipova kodnih tačaka: grafičke, formatske, upravljačke, za privatnu upotrebu, surogati, nekarakteri i rezervisane

Postoji sedam osnovnih tipova kodnih tačaka: grafičke, formatske, upravljačke, za privatnu upotrebu, surogati, nekarakteri i rezervisane. Grafičke (engl. **graphics**) kodne tačke namenjene su za kodiranje slova, znakova, brojeva, znakova interpunkcije, različitih simbola i blanko (engl. **space**) karaktera. Svi karakteri koji se koriste u različitim pismima (takozvani osnovni karakteri) definisani su ovim kodnim tačkama. Kodne tačke tipa **format** se koriste za formatiranje okolnih karaktera, a kontrolnog (engl. **control**) tipa se koriste za upravljanje, ali nisu definisane Unicode standardom već su iste kao u ASCII kodu i kompatibilne su sa ISO/ECU 2022 okvirom. Slična situacija je i sa kodnim tačkama tipa **private-use**.

Kodne tačke tipa **surrogate**, **noncharacter** i **reserved**, za razliku od prethodnih, nisu dodeljene apstraktnim karakterima, pri čemu reserved nisu uopšte definisane, nego su ostavljene kao rezerva za neku buduću upotrebu. Ove kodne tačke se ne koriste prilikom razmene podataka ili je njihovo korišćenje ograničeno. Noncharacter kodne tačke su namenjene samo za privatnu upotrebu unutar aplikacije. Važnu funkciju imaju kodne tačke tipa **surrogate**. Za njih je odvojeno 2048 kodnih tačaka koje se koriste samo u UTF – 16 kodnom obliku, kako bi se prevazišao limit od 65535 kodnih tačaka koji je moguć sa 16 bitova.

One se koriste za reprezentaciju apstraktnih karaktera pomoću surogatnih parova, tj. sekvence od dve 16-bitne kodne jedinice. Na taj način je moguće reprezentovati dodatna 1.048.544 karaktera.

Slika 3.2.1 Tabela-1 Osnovni tipovi kodnih tačaka

UTF

Unicode forme kodiranja koriste kodne jedinice veličine 8, 16 i 32 bita, pa su i njihove oznake analogne tome: UTF-8, UTF-16 i UTF-32

Celi brojevi se u računaru reprezentuju specifičnim kodnim jedinicama koje mogu da budu 8, 16 ili 32 bita. Unicode standard koristi tri forme kodiranja koje specificiraju kako će neka kodna tačka za neki karakter biti reprezentovana nizom od jedne ili više kodnih jedinica. Unicode forme kodiranja koriste kodne jedinice veličine 8, 16 i 32 bita, pa su i njihove oznake analogne tome:

- UTF – 8
- UTF – 16
- UTF – 32

UTF je skraćenica od **Unicode Transformation Format**.

Na slici 2 je na primeru prikazan način reprezentacije karaktera u različitim formama kodiranja. Primenom UTF – 32 moguće je bilo koji karakter reprezentovati jednom 32-bitnom kodnom jedinicom. Ona ima istu vrednost kao i kodna tačka za taj karakter. U slučaju UTF –

16, većina karaktera se može izraziti jednom 16-bitnom kodnom jedinicom čija je vrednost ista kao i kodna tačka karaktera. Međutim, za karaktere čije kodne tačke imaju vrednost veću od FFFF potreban je par surogatnih 16-bitnih kodnih jedinica. Kada se koristi UTF - 8, karakter može biti reprezentovan sa jednim, dva, tri ili četiri bajta, a odnos između vrednosti ovih bajtova i vrednosti kodnih tačaka je mnogo složeniji.

Slika 3.2.2 Uporedni prikaz reprezentacije nekih karaktera formama kodiranja UTF-32, UTF-16 i UTF-8

UTF-8,16,32

UTF - 32 reprezentacijom je dozvoljeno reprezentovati kodne tačke u oblasti 0 do 10FFFF

UTF - 32 je najjednostavniji oblik kodiranja karaktera, jer je svaka kodna tačka predstavljena jednom 32-bitnom kodnom jedinicom iste vrednosti. Zbog toga je lako odrediti Unicode karakter iz njegove UTF - 32 reprezentacije. Njime je dozvoljeno reprezentovati kodne tačke u oblasti 0 do 10FFFF. Reprezentacije svih karaktera su 32 bita, što smanjuje probleme u izradi aplikacija. UTF - 32 je idealna forma kodiranja kada memoriski prostor nije ograničavajući faktor.

UTF - 16 je 16-bitni kod, pa je njime moguće jednom 16-bitnom kodnom jedinicom prezentovati 65535 različitih kodnih tačaka u intervalu U+0000 do U+FFFF. Kodne tačke u oblasti U+10000 do U+10FFFF se reprezentuju parom surogatnih 16-bitnih kodnih jedinica. Pošto se velika većina najčešće korišćenih karaktera modernih jezika nalazi u oblasti U+0000 do U+FFFF, pa ih je moguće reprezentovati jednom 16-bitnom kodnom jedinicom, u tim slučajevima UTF - 16 predstavlja efikasnije rešenje nego UTF - 32.

UTF - 8 je projektovan da se održi kompatibilnost sa sistemima koji koriste ASCII kod. Ova forma kodiranja ima varijabilnu dužinu, jer se neki karakteri predstavljaju jednom, neki sa dve, tri ili četiri 8-bitnih kodnih jedinica. Karakteri u opsegu U+0000 do U+007F predstavljaju se jednim bajtom.

BIG-ENDIAN I LITTLE-ENDIAN

„Big-endian“ prvo upisuje bajt sa najvišim značajem, a „little-endian“ sa najmanjim

Prethodna diskusija o formama kodiranja Unicode-a se odnosila na internu reprezentaciju kodnih jedinica u računaru. Svaka kodna jedinica u računaru reprezentovana je numeričkim tipom podataka. **Postoji, međutim, problem razmene podataka između računarskih sistema koji mogu biti i sa različitom arhitekturom.** Zbog toga je potrebno razmotriti tačan redosled bitova i bajtova u numeričkoj reprezentaciji kodiranog teksta. Da bi se obavila razmena neke tekstualne datoteke vrši se njena serijalizacija u tačno definisani niz bitova. Proces serijalizacije bajtova dozvoljava svim aplikacijama da pri čitanju datoteke tačno rekonstruišu numeričke vrednosti, a time i karaktere. U Unicode standardu se različite metode serijalizacije bitova nazivaju Unicode šeme kodiranja.

Postojeći računarski sistemi na različite načine internu smeštaju velike numeričke podatke. Razlika je pre svega u redosledu bajtova, odnosno u tome da li se u internoj prezentaciji prvo stavlja **bajt sa najmanjim značajem** ili **bajt sa najvišim značajem**. Ovi redosledi se nazivaju “**big-endian**” (prvo se upisuje najznačajniji bajt) ili “**little-endian**” (prvo se upisuje bajt najmanjeg značaja). Unicode oblici kodiranja UTF – 16 i UTF – 32 prilikom specifikacije serijalizacije bajtova uzimaju u obzir big-endian ili little-endian arhitekturu sistema na kojima će podaci biti reprezentovani.

Prema tome, šema kodiranja karaktera se sastoji od specifikacije forme kodiranja karaktera i specifikacije serijalizacije. Unicode standard, takođe, specificira upotrebu inicijalnog znaka redosleda bajtova (BOM – **byte order mark**) da bi se napravila eksplicitna razlika između big-endian i little-endian redosleda. Tabele 2 i 3 prikazuju moguće šeme kodiranja koje su propisane Unicode standardom.

Slika 3.2.3 Tabela-2 Šeme kodiranja u Unicode-u

Slika 3.2.4 Tabela-3 Heksadecimalna reprezentacija karaktera Ω i A u različitim šemama kodiranja Unicode standarda

▼ Poglavlje 4

Reprezentacija celih brojeva

KODIRANJE CELIH BROJAVA SA ZNAKOM

Celi brojevi sa znakom se mogu kodirati koristeći metode: magnituda sa znakom, komplement jedinice, komplement dvojke, excess-N

Celi brojevi (engl. **integers**) mogu biti sa ili bez znaka. **Celi brojevi bez znaka** su po definiciji pozitivni i nazivaju se prirodni brojevi. Koriste se za definisanje broja komada nekog entiteta u skupu ili kao redni brojevi. **Celi brojevi bez znaka** su ne-negativni brojevi i njihova interna reprezentacija odgovara binarnoj reprezentaciji broja.

Celi brojevi sa znakom mogu biti pozitivni ili negativni. Ovde je pored broja potrebno kodirati i znak. Postoje tri metode kodiranja celih brojeva sa znakom. Sve tri metode koriste najznačajniji bit (prvi sa leva) za kodiranje znaka. Po konvenciji **bit znaka** (engl. **sign bit**) je 0 ako je broj pozitivan i 1 ako je broj negativan. Metode za kodiranje celih brojeva sa znakom su:

- Magnituda sa znakom,
- Komplement jedinice,
- Komplement dvojke,
- Excess-N.

Prva dva metoda su se koristila u ranim danima računarske tehnike, dok se danas gotovo isključivo koristi metod komplementa dvojke.

MAGNITUDA SA ZNAKOM

Reprezentacija broja magnitudo sa znakom se sastoji od znaka i magnitudo boja

Magnituda sa znakom je najjednostavniji metod kodiranja celih brojeva sa znakom. Reprezentacija broja se sastoji od dva dela: znaka i magnitudo broja. Za znak je odvojen jedan bit, a ostali bitovi predstavljaju magnitudu u binarnoj reprezentaciji. Na primer broj $+56_{10}$ u ovoj reprezentaciji ima oblik 00111000, a negativni broj -56_{10} oblik 10111000. Najveći pozitivan broj koji može da se memoriše u jednom bajtu je $+127_{10} = 01111111_2$, a najveći negativan broj je $-127_{10} = 11111111_2$.

Iako je jednostavan ovaj metod se reko koristi jer ima neke nedostatke. Najpre, javljaju se dva moguća koda za nulu, +0 (00000000) i -0 (10000000), što znatno komplikuje logiku centralne procesorske jedinice. Takođe, sabiranje pozitivnog i negativnog broja nije moguće izvesti,

nego se operacija mora preuređiti na oduzimanje magnitude negativnog broja od pozitivnog. Ovo takođe komplikuje logiku rada procesora.

U slučaju reprezentacije celih brojeva sa znakom postoji samo jedan kod za 010 čiji je kod 00000000, pa se jednim bajtom mogu reprezentovati brojevi od -128 do +127.

KOMPLEMENT JEDINICE

Komplement jedinice nekog binarnog broja se dobija tako što svi bitovi koji su bili 0 postaju 1, a svi koji su bili 1 postaju 0.

Komplement jedinice nekog binarnog broja se dobija tako što svi bitovi koji su bili 0 postaju 1, a svi koji su bili 1 postaju 0. Na primer komplement jedinice binarnog broja 1001 je 0110. I u slučaju reprezentacije metodom komplementa jedinice prvi bajt predstavlja znak broja. Magnituda pozitivnih brojeva odgovara binarnoj reprezentaciji magnitude. Magnituda negativnih brojeva se dobija kao komplement jedinice binarne reprezentacije magnitude. Na primer, koristeći kodiranje komplementa jedinice broj $+51_{10}$ ima reprezentaciju 00110011, a broj -51_{10} 11001100. Kodiranje negativnih celih brojeva komplementom jedinice opisano je izrazom :

$$-X = \text{not}(X)$$

Treba primetiti da se procedurom komplementiranja automatski menja i bit znaka broja. Metod komplementa jedinice ima iste nedostatke kao i metod magnitude sa znakom.

KOMPLEMENT DVOJKE

Komplement dvoje binarnog broja se određuje prvo komplementom jedinice, nakon čega se rezultatu dodaje jedinica

Komplement dvojke nekog binarnog broja nalazi se po sledećoj proceduri:

Odredi se komplement jedinice datog binarnog broja

Doda se jedinica rezultatu iz prvog koraka.

Procedura nalaženja komplementa dvojke će biti prikazana na primeru broja -56_{10} .

Binarni broj **00111000**

Komplement jedinice **11000111**

Sabiranje sa 1 **+1**

Komplement dvojke **11001000**

Reprezentacija negativnih brojeva se po ovoj reprezentaciji dobija tako što se nađe komplement dvojke datog pozitivnog broja. U ovom slučaju je broj 11001000 reprezentacija broja -56_{10} . Ako se izvrše dve uzastopne negacije i po ovom metodu će se dobiti početni broj. Kodiranje negativnih celih brojeva komplementom dvojke opisano je izrazom:

-X=not(X)+1

KOMPLEMENT DVOJKE - PRIMER

Postupak određivanja komplementa dvojke za broj -17

Pronaći komplement dvojke za decimalni broj -17.

Prvo je potrebno izvršiti konverziju broja u binarni:

$$17/2 = 1$$

$$8/2 = 0$$

$$4/2 = 0$$

$$2/2 = 0$$

$$1/2 = 1$$

Dakle, čitajući ostatke deljenja odozgo na dole dobija se:

$$17_{10} = 0001\ 0001_2$$

Potom se određuje komplement jedinice dobijenog broja tako što se nule menjaju u jedinice, a jedinice u nula, te se dobija:

$$1110\ 1110$$

Komplement dvojke se potom dobija dodavanjem **1** na prethodni rezultat, pa je komplement dvojke broja -17:

$$1110\ 1111_2$$

PRIMER - KOMPLEMENT DVOJKE

Određivanje komplementa dvojke broja -30

Razmotrimo decimalni broj **-30**, koji želimo da prikažemo komplementom dvojke. Prvo određujemo njegovu **binarnu reprezentaciju**:

$$0000\ 0000\ 0000\ 0000\ 0000\ 0001\ 1110$$

Uradimo njegovu negaciju, odnosno odredimo **komplement jedinice**:

$$1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1110\ 0001$$

Dodamo 1 i dobijamo **komplement dvojke broja -30**:

$$1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1110\ 0010.$$

Ako želimo da konverujemo ovaj broj u heksadecimalni broj dobijamo 0xFFFFFE2.

Takođe ako iskoristimo ovaj kod, treba da dobijemo izlaz -30:

```
#include <stdio.h>
int main() {
int myInt;
myInt = 0xFFFFFE2;
printf("%d\n",myInt);
return 0;
}
```

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

BCD

Kodiranje se vrši tako što se svaka cifra decimalnog broja predstavlja sa četiri bita

Pored prethodna tri, za reprezentaciju celih brojeva se koristi i metod binarnog kodiranja decimalnih brojeva ili BCD (Binary Coded Decimal) kodiranje. Najpoznatiji BCD kodovi su: 8421 BCD, Excess-3 i Gray-ov kod. U ovom slučaju je potrebno kodirati decimalne cifre od nula do devet. Kodiranje se vrši tako što se svaka cifra decimalnog broja predstavlja sa četiri bita. Koja će kombinacija bitova predstavljati neku decimalnu cifru zavisi od izabrane šeme kodiranja. U Tabeli 1 su uporedno prikazani neki od kodova za binarno kodiranje decimalnih brojeva.

Slika 4.1 Tabela-1 Uporedni prikaz nekih kodova za binarno kodiranje decimalnih brojeva

8241 BCD KOD

U 8241 BCD kodu svaka cifra decimalnog broja je predstavljena odvojenim nizom bitova

Najčešće se koristi standardni 8421 BCD kod, koji je istovetan sa binarnom prezentacijom cifara. Tako, na primer, decimalni broj 2579 u 8421 BCD kodu ima oblik

2 5 7 9 0010 0101 0111 1001

gde svaku cifru decimalnog broja predstavljamo odvojenim nizom bitova.

Ovaj kod je, kao i ostali BCD kodovi, manje ekonomičan od binarne reprezentacije broja. Na primer, sa 8 bita je u binarnom zapisu moguće predstaviti cele nenegativne brojeve u intervalu od 0 do 255, a u BCD kodu samo od 0 do 99. Razlog za njegovo korišćenje leži u činjenici da je olakšana komunikacija između čoveka i računara, jer se pojedine cifre mogu lakše dekodirati nego ceo broj. U radu sa 8421 BCD kodom se javljaju problemi pri operacijama sabiranja i oduzimanja, pa se zbog toga često koristi Excess-3 kod. Grey-ov kod se uglavnom koristi kod računara i merne opreme koja je namenjena akviziciji podataka. Ovaj kod se u industrijskoj praksi pokazao najpouzdaniji zbog toga što se prilikom promene vrednosti broja za jedan uvek menja samo po jedan bit u kodu.

▼ Poglavlje 5

Osnovne veličine u računarstvu

MERE I IZVEDENE MERE U RAČUNARSTVU

Mera brzine prenosa digitalnih signala koristi se broj bitova u sekundi (b/s)

Bit je binarna cifra (**Binary Digit**) u nekom broju binarnog brojnog sistema. Može da ima vrednost 0 ili 1. Oznaka za bit je „**b**“.

Bajt (**Byte = BinarY TablE**) je grupa od 8 bita koja se tretira jedinstveno. Koristi se kao osnovna mera memorijskog kapaciteta. Oznaka za bajt u računarskoj praksi je „**B**“. Izvedene mere su dobijene stepenovanjem osnove 2 sa 10, 20, kao što je prikazano u Tabeli 1. U računarskoj praksi se ustalilo korišćenje simbola za izvedene mere: KB, MB itd. Ovi simboli su u suprotnosti sa SI (**International System of Units**), standardom koji prefiks "K" tretira kao 1000 , a ne 1024. Slično je sa prefiksom M koji označava 1.000.000, a ne 1.048.576. Razlika između ovih veličina je 4,85%. Zbog toga je organizacija IEC 1998. godine usvojila nove nazine i simbole za izvedene mere memorijskog kapaciteta, koji su takođe prikazani u tabeli 2. 6 u koloni IEC SIMBOL.

Reč je niz bitova koji predstavljaju kompletan podatak. Broj bitova u reči zavisi od arhitekture procesora. Radi pojednostavljenja, računari rade uvek sa podacima jednake dužine. Manji računari koriste reč od 8 ili 16 bita, a veći od 32 ili 64 bita.

Kao mera brzine prenosa digitalnih signala koristi se broj bitova u sekundi (b/s). U praksi se koriste izvedene mere Kb/s (kilobita u sekundi), Mb/s (megabita u sekundi) i Gb/s (gigabita u sekundi). Jedan Kb/s = 1.000 b/s.

FLOPS (**Floating-point Operations Per Second**) mera je brzine rada (procesne snage) računara, bazirana na operaciji sa pokretnim zarezom. Jedan FLOPS odgovara izvršenju jedne operacije sa pokretnim zarezom u sekundi. Izvedene mere su: **KFLOPS = 1000 FLOPS, MFLOPS=1000 KFLOPS, GFLOPS=1000 MFLOPS i TFLOPS=1000 GFLOPS.**

IPS (**Instruction Per Second**) druga je mera brzine rada računara, ali ona više govori o brzini rada centralne procesorske jedinice nego o brzini izvršenja nekog zadatka. Jedan IPS odgovara izvršenju jedne procesorske naredbe u sekundi. Zbog velikih brzina procesora danas se upotrebljava izvedena mera MIPS (**Mega Instruction Per Second**).

Slika 5.1 Tabela-1 Osnovna mera i izvedene mere za memorijski kapacitet (binarni multiplikatori)

DODATNI MATERIJAL - KONVERZIJA IZMEĐU BITOVA I BAJTOVA

Converting Between Bits and Bytes - "Ladder" Analogy - General Maths

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

DODATNI MATERIJAL - PRIMER KONVERZIJE BITOVA I BAJTOVA

Converting Between Bits and Bytes - Practice Problems - General Maths

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 6

Logičke operacije

OSNOVNE LOGIČKE OPERACIJE

Tri osnovne logičke operacije su I, ILI, NE

Logičke operacije su operacije koje se izvode nad promenljivim (operandima) koje mogu da imaju samo dve vrednosti. Uobičajeno, ove vrednosti su tačno (engl. **true**) i netačno (engl. **false**). U računarskoj tehnici simbol za tačno je 1, a za netačno 0. Po svojoj prirodi ovo su, dakle, binarne promenljive. Engleski naučnik Džordž Bul (George Bool) je razvijajući binarnu algebru sredinom 19. veka predvideo **tri osnovne logičke operacije**:

- **I** (engl. **AND**)
- **ILI** (engl. **OR**)
- **NE** (engl. **NOT**).

Pored ovih osnovnih, postoje još dve izvedene logičke operacije **NI** i **NIL**.

Na osnovu rada Džordža Bula formirana je posebna grana matematike koja se naziva **logička** ili **Bulova algebra**, a koja proučava **logičke operacije nad binarnim promenljivim veličinama**. Bulova algebra se koristi u automatiči za projektovanje i analizu prekidačkih mreža. Ove mreže mogu biti realizovane različitim klasama komponenata, kao što su pneumatske, hidraulične, električne ili elektronske. U računarskim sistemima se koriste elektronske digitalne komponente koje su realizovane kao logička kola sa tranzistorima. One predstavljaju osnovni gradivni element računara.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ 6.1 Logička operacija „I“

ISTINITOSNE VREDNOSTI LOGIČKE OPERACIJE „I“

Logička operacija „I“ je tačna samo kada su svi operandi tačni

Logička operacija „I“ (engl. **AND**) se izvodi nad dva ili više **operanda**. **Rezultat operacije je jednak 1 („tačno“) ako su svi operandi jedinice**. Ako je bar jedan operand jednak 0, rezultat ove logičke operacije je 0.

U tabeli 1 je dat primer korišćenja logičke operacije **I**. Operacija je izvršena nad dva operanda X_1 i X_2 . Rezultat operacije je Y . Ova tabela naziva se i tabela stanja.

Slika 6.1.1 Tabela-1 Tablica istinitosnih vrednosti logičke operacije I za dva operanda X_1 i X_2

Logička operacija **I** se može jednostavno digitalno realizovati pomoću dva redno vezana prekidača. Po konvenciji, stanje isključenog prekidača se označava sa 0, a uključenog prekidača sa 1. Na slici 2 je prikazana realizacija. Očigledno je da će signal Y moći da postoji, tj. da ima vrednost 1 samo ako su i prekidač X_1 i X_2 uključeni, što odgovara vrednosti promenljivih $X_1 = 1$ i $X_2 = 1$.

Slika 6.1.2 Realizacija logičke operacije I pomoću dva prekidača X_1 i X_2

LOGIČKA OPERACIJA „I“ SA DVA I VIŠE OPERANADA

Logička operacija I sa tri operanda X_1 , X_2 , i X_3 će imati vrednost 1 samo ako i X_1 i X_2 i X_3 imaju vrednost 1

Logička operacija **I** sa dva operanda X_1 i X_2 se opisuje sledećom logičkom ili Bulovom jednačinom:

$$Y = X_1 * X_2$$

$$Y = X_1 \wedge X_2$$

Logička tačka između X_1 i X_2 u gornjoj jednačini nema nikakve veze sa množenjem, već predstavlja simbol za logičku **I** operaciju. Logičke operacije **I** mogu da se izvode na najmanje dva operanda, ali i na tri, četiri i više. Logička operacija **I** sa tri operanda X_1 , X_2 , i X_3 bila bi opisana sledećim izrazom:

$$Y = X_1 \wedge X_2 \wedge X_3$$

Vrednost Y će imati vrednost 1 samo ako i X_1 i X_2 i X_3 imaju vrednost 1. Za prikaz logičke operacije I u šemama logičkih sklopova koristi se simbol prikazan na slici 3.

Slika 6.1.3 Simbolički prikaz logičke operacije I a) sa dva operanda, b) sa tri operanda

✓ 6.2 Logička operacija “ILI”

ISTINITOSNE VREDNOSTI LOGIČKE OPERACIJE „ILI“

Rezultat operacije ILI je 1 ako bar jedan operand ima vrednost 1

Logička operacija **ILI** (engl. **OR**) se izvodi nad dva ili više operanda. **Rezultat operacije ILI je 1 ako bar jedan operand ima vrednost 1**. Ako svi operandi imaju vrednost 0, rezultat operacije je 0. Logička operacija **ILI** odgovara paralelnoj vezi dva ili više prekidača.

U tabeli 1 prikazuje stanje logičke operacije **ILI** sa dva operanda X_1 i X_2 . Rezultat operacije je Y

Slika 6.2.1 Tabela-1 Tabela stanja logičke operacije ILI sa dva operanda X_1 i X_2 .

Logička operacija **ILI** sa dva operanda X_1 i X_2 se opisuje sledećom logičkom ili Bulovom jednačinom:

$$Y = X_1 + X_2$$

$$Y = X_1 \vee X_2$$

Za prikaz logičke operacije **ILI** u šemama logičkih sklopova koristi se simbol prikazan na slici 2. U ovom slučaju operandi su X_1 i X_2 , a vrednost operacije **ILI** je Y .

Slika 6.2.2 Simbolički prikaz logičke operacije ILI

✓ 6.3 Logička operacije „NE“

ISTINITOSNE VREDNOSTI LOGIČKE OPERACIJE „NE“

Ako je vrednost operanda $X_1 = 1$ i ako se nad njim izvrši logička operacija NE, rezultat operacije će biti 0

Logička operacija "NE" (engl. NOT) se izvodi samo nad jednim operandom. **Rezultat operacije je promena binarnog stanja operanda.** Na primer, ako je vrednost operanda $X_1 = 1$ i ako se nad njim izvrši logička operacija **NE**, rezultat operacije će biti 0. Zbog toga se logički sklop koji izvodi ovu operaciju naziva invertor.

U tabeli 1 je prikazano stanje logičke operacije **NE**. Operacija je izvršena nad logičkom promenljivom X_1 , a rezultat operacije je Y .

Slika 6.3.1 Tabela-1 Tabela stanja logičke operacije NE nad operandom X_1

Logička operacija **NE** nad operandom x_1 se opisuje sledećom logičkom ili Bulovom jednačinom:

$$Y = -X_1 = X_1'$$

Za prikaz logičke operacije **NE** u šemama logičkih sklopova koristi se simbol prikazan na slici 2 . U ovom slučaju operand je X_1 a rezultat logičke operacije **NE** je Y . Mali kružić na vrhu trougla predstavlja inverziju. Takav kružić na ulazu ili izlazu bilo kog sklopa takođe predstavlja inverziju.

Slika 6.3.2 Simbolički prikaz logičke operacije NE

✓ 6.4 Logička operacija „NI“

ISTINITOSNE VREDNOSTI LOGIČKE OPERACIJE „NI“

Logička operacija NI ima vrednost 0 samo kada svi operandi imaju vrednost 1

Logička operacija “**NI**” (engl. **NAND**) je izvedena od osnovnih logičkih operacija “**I**” i “**NE**.“ Logička operacija “**NI**” je ustvari negacija rezultata logičke operacije “**I**.“ Tablica istinitosti logičke operacije “**NI**” je prikazana u Tabeli 1 , a logički simbol na slici 2 .

Slika 6.4.1 Tabela-1 Istinitosna tablica logičke operacije “ NI”

Slika 6.4.2 Simbolički prikaz logičke operacije „ NI“

✓ 6.5 Logička operacija “NILI”

ISTINITOSNE VREDNOSTI LOGIČKE OPERACIJE „NILI“

Logička operacija NILI ima vrednost 1 samo kada svi operandi imaju vrednost 0

Logička operacija “**NILI**” (engl. **NOR**) je izvedena od osnovnih logičkih operacija “**IL**” i “**NE**.“ Logička operacija “**NILI**” je ustvari negacija rezultata logičke operacije “**IL**.“ Tablica istinitosti logičke operacije “**NILI**” je prikazana u Tabeli 1 , a logički simbol na slici 2 .

Slika 6.5.1 Tabela-1 Istinitosna tablica logičke operacije NILI

Slika 6.5.2 Simbolički prikaz logičke operacije NILI

✓ 6.6 Logička operacija “isključivo ILI”

ISTINITOSNE VREDNOSTI LOGIČKE OPERACIJE „IKSLJUČIVO ILI“

ISKLJUČIVO ILI ima vrednost 1 samo ako je jedan od operanada 1

Za razliku od logičke operacije **IL** gde je rezultat jednak 1 ako je bar jedan operanada jednak jedinici, logička operacija **isključivo (ekskluzivno) ILI** (engl. **XOR**) kao rezultat operacije

daje vrednost 1 samo ako je jedan od operanada 1. Ako su dva ili više operanda 1, rezultat operacije je 0. Zbog toga što ova operacija isključuje slučaj kada su dva ili više operanda jednaka 1 naziva se **isključivo ILI** (**exclusive OR**). Kao i kod operacije ILI vrednost operacije će biti 0 ako su svi operandi jednaki nuli. Logička operacija isključivo **IL**I sa dva operanda X1 i X2 se opisuje sledećom logičkom jednačinom:

$$y = x_1 \oplus x_2$$

Za prikaz logičke operacije isključivo ILI u šemama logičkih sklopova koristi se simbol prikazan na slici 2.

Slika 6.6.1 Tabela-1 Istinitosna tablica logičke operacije “ Isključivo ILI ” (XOR)

Slika 6.6.2 Simbolički prikaz logičke operacije “ Isključivo ILI ” (XOR)

✓ 6.7 Bulova algebra i digitalna logička kola

PREKIDAČI

Kada je prekidač zatvoren, struja može da teče od jednog terminala do drugog; kada je otvoren, struja ne može da teče

Slika 1 prikazuje izgled dva položaja jednostavnog prekidača.

Kada je prekidač zatvoren, struja može da teče od jednog terminala do drugog; kada je otvoren, struja ne može da teče.

Zamislite da je takav prekidač deo kola prikazan na slici 2. Sijalica se pali ako, i samo ako, struja teče kroz njega. A to će se desiti ako, i samo ako, je zatvoren prekidač.

Slika 6.7.1 Prikaz jednostavnog prekidača u položajima “otvoreno” i “zatvoreno”

Slika 6.7.2 Prikaz jednostavnog električnog kola sa prekidačem, baterijom i sijalicom

PRIKAZ PARALELNO I REDNO POSTAVLJENIH PREKIDAČA

Sijalica se pali ako, i samo ako, struja teče kroz njega. A to će se desiti ako, i samo ako, je zatvoren prekidač.

U kolu na slici 3 (a), struja teče i sijalica se pali ako, i samo ako, su oba prekidača P i Q zatvoreni.

Za prekidače u ovom kolu se kaže da su u nizu.

U kolu na slici 3 (b), struja teče i sijalica se pali ako, i samo ako, je barem jedan od prekidača P ili Q zatvoren.

Za prekidače u ovom kolu se kaže da su paralelni.

Svi mogući izlazi ovih kola su opisani u tabelama 1 i 2.

Slika 6.7.3 Prikaz prekidača u nizu (a) i paralelnih prekidača (b)

Slika 6.7.4 Tabela-1 Prikaz svih mogućnosti izlaza za redno postavljene prekidače

Obratite pažnju da ako se reči zatvoren i otvoren zamene sa T i F, Tabela 1, postaje istinosna tabela za logički "i", a Tabela 2 postaje isto za logičko "ili".

Slika 6.7.5 Tabela-2 Prikaz svih mogućnosti izlaza za paralelnopostavljene prekidače

Shodno tome, prekidačko kolo sa slike 3(a) odgovara logičkom iskazu $P \wedge Q$, a kolo na slici 3 (b) iskazu $P \vee Q$. Komplikovanija kola imaju komplikovanije logičke iskaze.

VEZA LOGIČKIH KOLA I LOGIČKIH TABLICA

Osnovne elektronske komponente digitalnih sistema se zovu digitalna logička kola.

1940-ih i 1950-ih, prekidači su zamenjeni elektronskim uređajima, sa fizičkim stanjima "zatvoren" i "otvoren" koji su odgovarali elektronskim stanjima visoki i nizak napon. Nova elektronska tehnologija je dovela do razvoja modernih digitalnih sistema kao što su elektronski računari, elektronske telefonske centrale, elektronski kalkulatori i kontrolni mehanizmi koji se koriste u drugim vrstama elektronske opreme.

Osnovne elektronske komponente digitalnih sistema se zovu digitalna logička kola. Reč "logika" ukazuje na značajnu ulogu logike u projektovanju takvih kola, a reč "digitalna" ukazuje na to da kola procesuiraju diskretne ili odvojene signale (za razliku od kontinuiranih). Elektro-inženjeri i dalje koriste jezik logike kada se pozivaju na vrednosti signala koje proizvodi elektronski prekidač kao "tačno" ili "netačno." Ali oni uglavnom koriste simbole 1 i 0, pre nego T i F za označavanje ovih vrednosti. Simboli 0 i 1 se zovu bitovi, skraćeno od binary digits. Ova terminologija je uveden 1946. godine od strane statističara John Tukey.

KOMBINATORNO KOLO

Kombinaciono kolo predstavlja skup međusobno povezanih prekidačkih kola čiji je izlaz funkcija njegovih izlaza

Kombinaciono kolo predstavlja skup međusobno povezanih prekidačkih kola čiji je izlaz funkcija njegovih izlaza.

Bilo koju Bulovu funkciju je moguće implementirati u elektronskom obliku kao mrežu sastavljenu od prekidačkih kola.

Primer 1

Odredite izlaze za prikazano kolo

Slika 6.7.6 Kolo za primer 1

Kako pristupiti rešenju:

Krenite s leva na desno kroz dijagram, prateći funkcionalnosti svakog kola. Pogledajmo primer kada je $P = 0$ i $Q = 1$. NE-kolo menja $P = 0$ u 1, tako da su oba ulaza I-kola 1. Stoga je izlaz $R = 1$. Ovo je ilustrovano anotirajući dijagram kao što je prikazano u nastavku.

Slika 6.7.7 Jedno od mogućih rešenja za kolo iz primera 1

Međutim, ovo nam ne daje prikaz svih ulaza i njima odgovarajućih izlaza. Ovo možemo prikazati kroz tablicu istinosti. Iskaz za izlaz R je

$$R = P' \wedge Q$$

A njegova tablica istinosti je

Slika 6.7.8 Tabela-3 Tablica istinitosti za kolo iz primera 1

KOMBINATORNO KOLO - PRIMER 2

Odredite izlaze za prikazano kolo

Primer 2

Odredite izlaze za prikazano kolo

Slika 6.7.9 Kolo za primer 2

Kako pristupi rešenju:

Pogledajmo kolo za vrednosti $P=1$ $Q=0$ $R=1$

Slika 6.7.10 Jedno od mogućih rešenja za kolo iz primera 2

Logički iskaz za S je $(P \vee Q)' \wedge R$

A njegova tablica istinosti je

Slika 6.7.11 Tabela-4 Tablica istinitosti za kolo iz primera 2

KOMBINATORNO KOLO - PRIMER 3

Odredite izlaze za prikazano kolo na slici

Primer 3

Konstruisati tablicu istinosnih vrednosti za ulaze i izlaze sledećeg kola

Slika 6.7.12 Kolo za primer 3

Iskaz za R

$Q' \vee P$

Rešenje:

Slika 6.7.13 Tabela-5 Tablica istinitosti za kolo iz primera 3

KOMBINATORNO KOLO - PRIMER 4

Odredite izlaze za kolo sa slike

Primer 4

Konstruisati tablicu istinosnih vrednosti za ulaze i izlaze sledećeg kola

Slika 6.7.14 Kolo za primer 4

Iskaz za primer 4 je:

$A^B \vee (A^C)'$

Rešenje:

Slika 6.7.15 Tabela-6 Tablica istinitosti za kolo iz primera 4

KOMBINATORNO KOLO - PRIMER 5

Odredite izlaze za kolo prikazano na slici

Primer 5

Konstruisati tablicu istinosnih vrednosti za ulaze i izlaze sledećeg kola

Slika 6.7.16 Kolo za primer 5

Iskaz za primer 5 je:

$(A^B) \vee ((B \vee C) \wedge (B^C))$

Rešenje:

Slika 6.7.17 Tabela-7 Tablica istinitosti za kolo iz primera 5

✓ Poglavlje 7

Pokazna vežba: Konverzija brojeva

PODSETNIK - KONVERTOVANJE HEX U BIN

Video prikaz konvertovanja heksadecimalnog broja u binarni

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

PRETVARANJE BINARNOG BROJA U DEKADNI

Prikaz pretvaranja binarnog broja u dekadni

Predviđeno vreme pokazne vežbe je 45 minuta.

Ukoliko je lakše, možete stepene automatski izračunati kao što je urađeno u prvom primeru.

Predviđeno vreme izrade sledećeg zadatka je 3 minuta.

1.) Konvertovati binarni broj 1010001 u decimalni broj

Rešenje:

$$(1 \times 2^0) + (0 \times 2^1) + (0 \times 2^2) + (0 \times 2^3) + (1 \times 2^4) + (0 \times 2^5) + (1 \times 2^6) = 1 + 0 + 0 + 0 + 16 + 0 + 64 \\ = 81$$

Predviđeno vreme izrade sledećeg zadatka je 2 minuta.

2.) Konvertovati 1001 binarni u decimalni broj:

Rešenje:

$$1001 = (1 \times 2^3) + (0 \times 2^2) + (0 \times 2^1) + (1 \times 2^0) = (1 \times 8) + (0 \times 4) + (0 \times 2) + (1 \times 1) = 8 + 0 + 0 + 1 = 9$$

Predviđeno vreme izrade sledećeg zadatka je 3 minuta.

3.) Konvertovati 01011101 binarni u decimalni broj:

Rešenje:

$$01011101 = (0 \times 2^7) + (1 \times 2^6) + (0 \times 2^5) + (1 \times 2^4) + (1 \times 2^3) + (1 \times 2^2) + (0 \times 2^1) + (1 \times 2^0) \\ = (0 \times 128) + (1 \times 64) + (0 \times 32) + (1 \times 16) + (1 \times 8) + (1 \times 4) + (0 \times 2) + (1 \times 1)$$

$$= 0 + 64 + 0 + 16 + 8 + 4 + 0 + 1$$

Rešenje:

$$1010110010 = (1 \times 29) + (0 \times 28) + (1 \times 27) + (0 \times 26) + (1 \times 25) + (1 \times 24) + (0 \times 23) + (0 \times 22) + (1 \times 21) + (0 \times 20)$$

$$\begin{aligned} &= (1 \times 512) + (0 \times 256) + (1 \times 128) + (0 \times 64) + (1 \times 32) + (1 \times 16) + (0 \times 8) + (0 \times 4) + (1 \times 2) + (0 \times 1) \\ &= 512 + 0 + 128 + 0 + 32 + 16 + 0 + 0 + 2 + 0 = 690 \end{aligned}$$

PRETVARANJE DEKADNOG U BINARNI SA POKRETNIM ZAREZOM

Prikaz pretvaranja dekadnog broja u binarni

Predviđeno vreme izrade sledećeg zadatka je 5 minuta.

1. Dekadni broj $x_{(10)} = 240,375_{(10)}$ pretvoriti u binarni, sa 3 decimale tačnosti $x_{(10)} \rightarrow x_{(2)}$.

Rešenje:

$$240 : 2 = 120\ 0$$

$$0.375 * 2 = 0.75\ 0$$

$$120 : 2 = 60\ 0$$

$$0.75 * 2 = 1.5\ 1$$

$$60 : 2 = 30\ 0$$

$$0.5 * 2 = 1.0\ 1$$

$$30 : 2 = 15\ 0$$

$$15 : 2 = 7\ 1$$

$$7 : 2 = 3\ 1$$

$$3 : 2 = 1\ 1$$

$$1 : 2 = 0\ 1$$

$$240,375_{(10)} \rightarrow 11110000,011_{(2)}$$

PRETVARANJE BINARNOG BROJA U HEKSADECIMALNI BROJ

Prikaz pretvaranja binarnog broja u heksadecimalni

Predviđeno vreme izrade sledećeg zadatka je 2 minuta.

1. Binarni broj $x_{(2)} = 1011_{(2)}$ pretvoriti u heksadecimalni.

Rešenje:

$$1*2^3 + 0*2^2 + 1*2^1 + 1*2^0 =$$

$$8 + 0 + 2 + 1 = 11 = B$$

$$1011_{(2)} = B_{(16)}$$

Predviđeno vreme izrade sledećeg zadatka je 2 minuta.

2. Binarni broj $x_{(2)} = 01101101_{(2)}$ pretvoriti u heksadecimalni.

Rešenje:

$$0*2^3 + 1*2^2 + 1*2^1 + 0*2^0 = 6$$

$$1*2^3 + 1*2^2 + 0*2^1 + 1*2^0 = 13 = D$$

$$01101101_{(2)} = 6D_{(16)}$$

Predviđeno vreme izrade sledećeg zadatka je 2 minuta.

3. Binarni broj $x_{(2)} = 101011100_{(2)}$ pretvoriti u heksadecimalni.

Rešenje:

$$1 \ 0101 \ 1100$$

$$\mathbf{0101} : 0*2^3 + 1*2^2 + 0*2^1 + 1*2^0 = 5$$

$$\mathbf{1100} : 1*2^3 + 1*2^2 + 0*2^1 + 0*2^0 = 12 = C$$

$$101011100_{(2)} = 15C_{(16)}$$

PRETVARANJE OKTALNOG BROJA U BINARNI

Prikaz pretvaranja oktalnog broja u binarni

Predviđeno vreme izrade sledećeg zadatka je 3 minuta.

1. Pretvoriti oktalni broj $x_{(8)} = 32_{(8)}$ u binarni.

Rešenje:

Da bi se oktalni broj pretvorio u binarni, potrebno je pretvoriti svaku cifru posebno. Dakle, prvo ćemo pretvoriti 3 u binarni broj, a zatim 2 i spojićemo dobijene binarne vrednosti.

$$3_{(8)} = 011_{(2)}$$

$$2_{(8)} = 010_{(2)}$$

$$\mathbf{32}_{(8)} = \mathbf{011010}_{(2)}$$

Predviđeno vreme izrade sledećeg zadatka je 3 minuta.

2. Pretvoriti oktalni broj $x_{(8)} = 74$ u binarni

Rešenje:

$$7_{(8)} = 111_{(2)}$$

$$4 \text{ (8)} = 100 \text{ (2)}$$

$$\mathbf{74 \text{ (8)} = 111100 \text{ (2)}}$$

ZADACI ZA VEŽBU SA REŠENJIMA

Primer konverzije brojeva u različite brojne sisteme

Zadatak 1 (Predviđeno vreme izrade: 2 minuta)

Koji je dekadni ekvivalent binarnog broja 1011011?

$$(1011011)_2 = 1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 64 + 16 + 8 + 2 + 1 \\ = (91)_{10}$$

Zadatak 2 (Predviđeno vreme izrade: 2 minuta)

Prevesti brojeve 0,1101 i 1,01 iz binarnog u dekadni brojni sistem.

$$(0,1101)_2 = 0 \cdot 2^0 + 1 \cdot 2^{-1} + 1 \cdot 2^{-2} + 0 \cdot 2^{-3} + 1 \cdot 2^{-4} = (0,6875)_{10}$$

$$(1,01)_2 = 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} = (1,25)_{10}$$

Zadatak 3 (Predviđeno vreme izrade: 2 minuta)

Koja je binarna reprezentacija broja 26,75?

$$(26,75)_{10} = (11010,11)_2$$

Zadatak 4 (Predviđeno vreme izrade: 2 minuta)

Prevesti broj (110100110,111001)₂ u oktalni i heksadecimalni zapis.

$$(110100110,111001)_2 = 110|100|110|,111|001 = (646,71)_8$$

$$(110100110,111001)_2 = 0001|1010|0110|,1110|0100 = (1A6,E4)_{16}$$

Zadatak 5 (Predviđeno vreme izrade: 2 minuta)

Odrediti komplement dvojke decimalnog broja 55

Rešenje:

$$(55)_{10} = (0110111)_2$$

Te je komplement dvojke onda

$$1001001$$

Zadatak 6 (Predviđeno vreme izrade: 10 minuta)

Koristeći istinitosne tablice rešiti sledeće zadatke:

a. $(P \vee Q) \Rightarrow (P \wedge R)'$

b. $(P' \vee Q')'$

▼ Poglavlje 8

Zadaci za samostalni rad: Konverzije

ZADATAK ZA SAMOSTALNI RAD - KONVERZIJE

Rešiti sledeće zadatke

Predviđeno vreme za izradu sledećih zadataka je 20 minuta, vreme izrade pojedinačnog zadatka je 2 minuta.

1. Koji je dekadni ekvivalent binarnog broja 1001010101?
2. Koji je dekadni ekvivalent binarnog broja 10111111?
3. Koji je dekadni ekvivalent heksadecimalnog broja 1010?
4. Koji je dekadni ekvivalent heksadecimalnog broja A0EF?
5. $(A0EF)_{16} = (x)_2$?
6. $(FFF)_{16} = (x)_2$?
7. $(BC12)_{16} = (x)_{10}$?
8. $(27)_8 = (x)_2$?
9. $(4307)_8 = (x)_{10}$?
10. $(100011)_2 = (x)_8$?

▼ Poglavlje 9

Pokazna vežba: Tablice istinitosti i komb. kola

ZADATAK 1

Vežbanje određivanja istinitosnih tablica

Predviđeno vreme pokazne vežbe je 25 minuta.

Predviđeno vreme izrade sledećeg zadatka je 15 minuta.

Odrediti tablicu istinitosti za sledeća logička kola:

Slika 9.1 Logičko kolo

Primetiti da i jednom i drugom logičkom kolu odgovara ista istinitosna tablica. **Navedena logička kola su ekvivalentna, tj. Za iste kombinacije ulaznih vrednosti daju jednak rezultat.**

Slika 9.2 Tabela-1 Tablica istinitosti

ZADATAK 2

Odredite izlaze za navedeno kolo

Predviđeno vreme izrade sledećeg zadatka je 10 minuta.

Odredite izlaze za prikazano kolo

Slika 9.3 Logičko kolo

Kako pristupi rešenju: Pogledajmo kolo za vrednosti $P=1$ $Q=1$

Slika 9.4 Logičko kolo sa određenim vrednostima P i Q

Logički iskaz za S je

$$P \vee Q'$$

A njegova tablica istinosti je

Slika 9.5 Tabela-2 Tablica istinitosti

✓ Poglavlje 10

Zadaci za samostalni rad: Logička kola

ZADATAK ZA SAMOSTALNI RAD - LOGIČKA KOLA

Rešiti sledeće zadatke za samostalni rad

Predviđeno vreme za izradu sledećih zadataka je 5 minuta po zadatku, ukupno 20 minuta.

1. Nacrtati logičko kolo koje odgovara izrazu $(AB+C)D$
2. Nacrtati logičko kolo koje odgovara izrazu $(AB)' + (CD)'$
3. Napraviti istinitosnu tablicu za logičko kolo sa slike 1
4. Napraviti istinitosnu tablicu za logičko kolo sa slike 2

Slika 10.1 Zadatak 3

Slika 10.2 Zadatak 4

▼ Poglavlje 11

Pokazna vežba: Opšte i logičke funkcije u Excel-u

AND LOGIČKA FUNKCIJA

AND daje TRUE ukoliko svi argumenti imaju vrednost TRUE

Predviđeno vreme pokazne vežbe je 15 minuta.

AND funkcija:

Daje TRUE ukoliko svi argumenti imaju vrednost TRUE; daje FALSE ukoliko neki argumenti imaju vrednost FALSE.

Upotrebljava se u kako bi se povećala korisnost drugih funkcija kojima se vrše logička testiranja. Na primer, funkcija IF izvršava logičko testiranje, a zatim daje jednu vrednost ako testiranje rezultira vrednošću TRUE i drugu vrednost ako testiranje rezultira vrednošću FALSE. Upotrebom funkcije AND kao argumenta logical_test funkcije IF možete testirati mnogo različitih uslova umesto samo jednog. Predstavlja se:

AND(logical1, [logical2], ...)

Sintaksa funkcije AND ima sledeće argumente:

logical1 Obavezno. Prvi uslov koji želite da testirate i koji može da rezultira vrednošću TRUE ili FALSE.

logical2, ... Opcionalno. Dodatni uslovi (najviše 255 uslova) koje želite da testirate i koji mogu da rezultiraju vrednošću TRUE ili FALSE.

Slika 11.1 Primer AND funkcije

OR LOGIČKA FUNKCIJA

OR daje TRUE ako je bilo koji argument TRUE

OR funkcija:

Daje TRUE ako je bilo koji argument TRUE, a FALSE ako su svi argumenti FALSE.

Sintaksa:

OR(logičko1, [logičko2], ...)

Logičko1 je obavezno, naredne logičke vrednosti su opcionalne. Od 1 do 255 uslova koje želite da proverite, a koji mogu imati vrednost TRUE ili FALSE.

Slika 11.2 Primer OR funkcije

NOT LOGIČKA FUNKCIJA

NOT funkcija daje obrnutu vrednost svog argumenta

NOT funkcija:

Daje obrnutu vrednost svog argumenta. Upotrebite funkciju NOT kada želite da budete sigurni da vrednost nije jednaka određenoj vrednosti.

Sintaksa:

NOT(logičko)

Logičko - vrednost ili izraz koji može da ima vrednost TRUE ili FALSE. Ako je vrednost argumenta logičko FALSE, funkcija NOT daje TRUE; ako je vrednost argumenta logičko TRUE, funkcija NOT daje FALSE.

Slika 11.3 Primer NOT funkcije

XOR LOGIČKA FUNKCIJA

Daje logičku isključivu OR operaciju svih argumenata

Sinaksa glasi:

XOR(L1, [L2], ...)

Sintaksa funkcije XOR ima sledeće argumente:

L1, L2, ... Logičko1 je uvek obavezno, a naredne logičke vrednosti su opcionalne.

Postoje 1 do 254 uslova za proveru. I mogu imati vrednost TRUE ili FALSE i mogu da budu logičke vrednosti, nizovi ili reference.

Slika 11.4 Primer XOR logičke funkcije

FUNKCIJA FACT

Daje faktorijel broja

FACT je matematička funkcija i daje faktorijel broja. Faktorijal se izračunava : $1*2*3*...*$ broja
Sinaksa glasi: **FACT(broj)** pri čemu je broj pozitivan broj za koji se želi izračunati faktorijel.
Ako broj nije ceo broj, zaokružuje se.

Slika 11.5 Prikaz FACT funkcije

FUNKCIJA ABS

Daje absolutnu vrednost broja.

Funkcija **ABS** je matematička funkcija i daje absolutnu vrednost broja tj. bez znaka.

Sintaksa glasi:

ABS(broj) - pri čemu je broj obavezan

Slika 11.6 Prikaz ABS funkcije

FUNKCIJA TODAY

Vraća današnji datum

Funkcija **TODAY** vraća broj današnjeg datuma.

Prikazani broj je za datum ili vreme koji Excel koristi za računanje datuma i vremena. Ako je pre unošenja funkcije format ćelije bilo podešeno na Opšti, Excel će format ćelije promeniti u Datum. Ukoliko želite da prikažete redni broj, format ćelije morate promeniti u Opšti ili Broj.

Funkcija se može koristiti kada je potrebno da na radnom listu bude prikazan tekući datum bez obzira na to kada otvorite radnu svesku. Ona je takođe korisna za izračunavanje intervala.

Na primer, ako znate da je neko rođen 1989. godine, sledeću formulu možete da upotrebite da biste izračunali koliko ta osoba ima godina računajući od te godine rođenja:

= YEAR(TODAY())-1989

Ova formula koristi funkciju TODAY kao argument za funkciju YEAR da bi se dobila tekuća godina od koje se oduzima 1989 i tako se dobija podatak koliko ta osoba ima godina.

Slika 11.7 Tabela-1 Prikaz nekih od funkcija TODAY funkcije

FUNKCIJA LOWER

Konvertuje velika slova iz teksta u mala slova.

Pretvara sva velika slova u mala slova.

Sintaksa glasi **LOWER(tekst)**

Sintaksa funkcije **LOWER(Tekst)** ima sledeće argumente:

Tekst - Tekst koji želite da konvertujete u velika slova. **LOWER** ne pretvara znakove u tekstu ako nisu u pitanju slova.

Slika 11.8 Korišćenje LOWER funkcije

Slika 11.9 Zaokruživanje polja sa tekstrom koji želimo da konvertujemo

Slika 11.10 Konvertovani tekst

FUNKCIJA UPPER

Konvertuje mala slova iz teksta u velika slova.

Pretvara sva mala slova u velika slova.

Sintaksa glasi **UPPER(tekst)**

Sintaksa funkcije **UPPER(Tekst)** ima sledeće argumente:

Tekst - Tekst koji želite da konvertujete u mala slova. UPPER ne pretvara znakove u tekstu ako nisu u pitanju slova.

Slika 11.11 Korišćenje UPPER funkcije

Slika 11.12 Zaokruživanje polja sa tekstrom koji želimo da konvertujemo

Slika 11.13 Konvertovani tekst

✓ Poglavlje 12

Zadaci za samostalni rad: Excel

ZADATAK ZA SAMOSTALNI RAD - EXCEL

Prikaz zadatka za samostalni rad

Predviđeno vreme za izradu sledećih zadataka je 10 minuta.

Zadatak

Koristeći Excel:

1. Konvertovati capslock tekst: STUDENT SAM METROPOLITAN UNIVERZITETA
2. Konvertovati tekst iz mala slova u velika: student sam metropolitan univerziteta
3. Pronaći koji će datum biti za 160 dana koristeći funkciju TODAY

✓ Poglavlje 13

Domaći zadatak

OPIS DOMAĆEG ZADATKA - DZ02

Uputstvo za izradu i slanje zadatka

Predviđeno vreme za izradu domaćeg zadatka je 45 minuta.

- Radni prostor snimite kao **IT101-DZ02-Ime_Prezime_broIndexa**
- U radnom prostoru je potrebno da pošaljete dokumente (.doc ili .docx formatu) sa opisom i rešenjem svakog zadatka
- Sve datoteke je potrebno arhivirati u ZIP fajlu. Naziv arhiviranog fajla treba da bude **IT101-dz02_ime_prezime_brojIndexa.zip**
- **Domaći zadatak pošaljite predmetnom asistentu na e-mail,** a u subject-u mejla napisati **IT101 - DZ02**

OPIS ZADATAKA

Zadaci 1-3

Zadatak 1.

Konvertovati sledeće brojeve:

$$(156)_{10} = (?)_{16}$$

$$(100110011010)_2 = (?)_{10}$$

$$(17C)_{16} = (?)_{10}$$

$$(\text{"broj indeksa"})_8 = (?)_{10}$$

$$(\text{"broj indeksa"})_{16} = (?)_{10}$$

$$(\text{"broj indeksa"})_{10} = (?)_2$$

Napomena, ako u broju indeksa imate brojeve koji nisu u brojnom sistemu, stavite proizvoljni broj umesto njih.

Zadatak 2.

U Excel-u napraviti tabelu u kojoj će biti upisani svi poeni iz predmeta IT101 (domaći, projekat, aktivnost, prisustvo). Poeni za aktivnost se dobijaju ako je student prisustvovao na 70% nastave, u suprotnom student ne može dobiti poene za aktivnost. Student može izaći na ispit ako ima više od 50% poena na projektnom zadatku i ako ukupno ima više od 35 poena.

Napraviti kolone koja računaju ukupno poene za svaku stavku, kolonu koja računa koliko ukupno predispitnih poena student ima, kao i kolonu u kojoj se ispisuje da li student ima uslov da izade na ispit ili ne.

Zadatak 3.

Odrediti tablicu istinosti za sledeće kolo

Slika 13.1 Zadatak 3

▼ ZAKLJUČAK

ZAKLJUČAK

Važno je da znamo na koji način se podaci smeštaju i kodiraju u računaru. Zbog toga smo u ovom predavanju izučili brojne sisteme koji se koriste prilikom ovog procesa, kao i načine kodiranja podataka pomoću šema kao što su ASCII i Unicode. Pored toga, neophodno je da IT stručnjak bude upoznat sa osnovnim veličinama u računarstvu i osnovnim logičkim operacijama, koji su bili predstavljeni u ovom predavanju.

Literatura

1. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, *Introduction to Algorithms*, Second Edition, The MIT Press, 2001.
2. Pantić Ž., Verovatnoća i statistika, GraĐevinski fakultet, Niš, 1988.
3. Gary B. Shelly, Misty E. Vermaat, *Discovering Computers 2011-Introductory: Living in a Digital World*, Cengage Learning 2010.



IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

KOMPONENTE RAČUNARSKIH SISTEMA

Lekcija 03

PRIRUČNIK ZA STUDENTE

IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

Lekcija 03

KOMPONENTE RAČUNARSKIH SISTEMA

- ▼ KOMPONENTE RAČUNARSKIH SISTEMA
- ▼ Poglavlje 1: Računarski sistem
- ▼ Poglavlje 2: Hardver računara
- ▼ Poglavlje 3: Memorija
- ▼ Poglavlje 4: Magistrale
- ▼ Poglavlje 5: Primeri magistrala
- ▼ Poglavlje 6: Vežba: Korišćenje Excel statističkih funkcija
- ▼ Poglavlje 7: Zadaci za samostalni rad: Excel funkcije
- ▼ Poglavlje 8: Domaći zadatak
- ▼ ZAKLJUČAK

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

✓ Uvod

UVOD

Cilj ovog predavanja je da objasni osnovne funkcionalnosti rada računarskog sistema, kroz osnovne funkcije operativnog sistema, uslužnih i aplikativnih softvera, kao i samog hardvera

Cilj ovog predavanja je da objasni rad računarskih sistema, kroz svakog od njegovih pojedinačnih elemenata od osnovnih funkcija operativnog sistema, uslužnih i aplikativnih softvera, do hardvera računara. Cilj je da se objasni rad glavnih računarskih komponenti kao što su centralna procesorska jedinica, razne vrste memorija kroz njihovu klasifikaciju i strukture, kao i funkcionalnosti magistrale u računaru.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 1

Računarski sistem

DELOVI RAČUNARSKOG SISTEMA

Računarski sistem se sastoji od hardvera i softvera

Računarski sistem (engl. **computer system**), nezavisno od svoje veličine, se sastoji od **hardvera** i **softvera**. Grubo rečeno, može se reći da se hardver sastoji od:

- sistemske jedinice (engl. **unit system**),
- ulaznih uređaja (engl. **input devices**)
- izlaznih uređaja (engl. **output devices**).

Svi računarski sistemi imaju sistemske jedinice, koje takođe nazivamo i šasijama, a koje mogu biti napravljene od plastike ili metala. Njihova glavna uloga je da zaštiti računare od oštećenja. Na slici 1 je prikazano da desktop računari, laptop računari, mp3 plejeri, mobilni telefoni, svi imaju sistemsku jedinicu ili šasiju.

Komponente kao što su kamera, tastatura, miš, zvučnici, skener, USB eksterni disk, su eksterne komponente koje se nalaze van sistemske jedinice (Slika 1) . Personalni desktop računari koji imaju integrisan monitor u svojim sistemima su izuzetak pošto imaju monitore u svojim sistemskim jedinicama. Sličan slučaj je i sa laptop računarima, pošto su njihovi monitori i tastature integrisane u sistemskoj jedinici.



Slika 1.1.1 Sistemska jedinica u različitim računarskim sistemima

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

STRUKTURA RAČUNARSKOG SOFTVERA

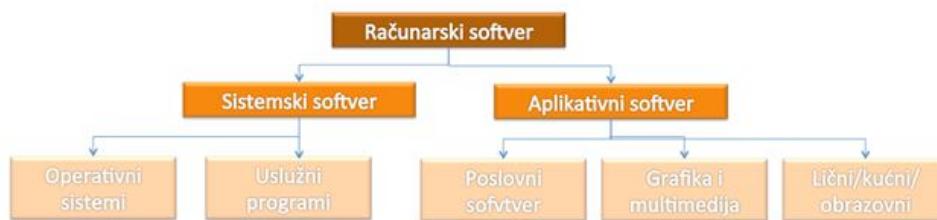
Računarsi softver se sastoji od sistemskog i aplikativnog softvera

Računarski sistem je sistem koji može da primi i obradi instrukcije. Ove instrukcije se primaju preko **perifernih jedinica**, koje nazivamo **ulaznim uredajima**. Kada se ove instrukcije procesuju, one se mogu prikazati korisniku ili se proslediti drugom delu sistema koji je zahtevao tu informaciju. Takođe, ova informacija koju se može procesovati i koristiti, ili sačuvati za kasniju upotrebu.

Softver računarskog sistema se može podeliti u sledeće grupe:

- **sistemski softver**
- **aplikativni softver.**

Struktura računarskog softvera i njegova podela po kategorijama je prikazana na slici 2.



Slika 1.1.2 Kategorije softvera računarskog sistema

✓ 1.1 Sistemske softver

ULOGA SISTEMSKOG SOFTVERA

Sistemske softver omogućuje efikasno korišćenje računarskog sistema, za čije je pisanje potrebno odlično poznavanje hardvera

Sistemske softver (engl. **system software**) se obično isporučuje od strane proizvođača računarskog sistema ili nezavisnih programske kuće i imaju zadatak da omoguće efikasno korišćenje računarskog sistema. Uloga sistemskog softvera je da upravlja računarskim hardverom i aplikativnim softverom. Ovo su vrlo složeni programi za čije pisanje je potrebno odlično poznavanje hardvera. Pošto bitno zavise od arhitekture hardvera, neki sistemske programi pisani za jedan hardver se ne mogu bez izmena koristiti za neki drugi, zasnovan na drugom procesoru ili drugim komponentama. Postoje dve grupe sistemskih programa: **operativni sistem** i **uslužni programi** (engl. **utility program**). Pošto nema precizne definicije operativnog sistema, ponekad je teško odrediti granicu i reći gde se završava operativni sistem, a gde počinju uslužni programi.

Primeri sistemskog softvera, pored već navedenih, su sledeći:

- **BIOS** (basic input/output system) koji pokreće računar nakon što ga korisnik uključi i upravlja tokom podataka između operativnog sistema i priključenih uređaja kao što su hard disk, video adapter, miš, tastatura, štampač i sl.
- **boot** program koji unosi operativni sistem u glavnu memoriju računara ili RAM memoriju
- **asembler** (engl. assembler) koji konvertuje računarske instrukcije u niz bitova koji procesor računara može da koristi

✓ 1.2 Operativni sistem

OSNOVNE ULOGE OPERATIVNOG SISTEMA

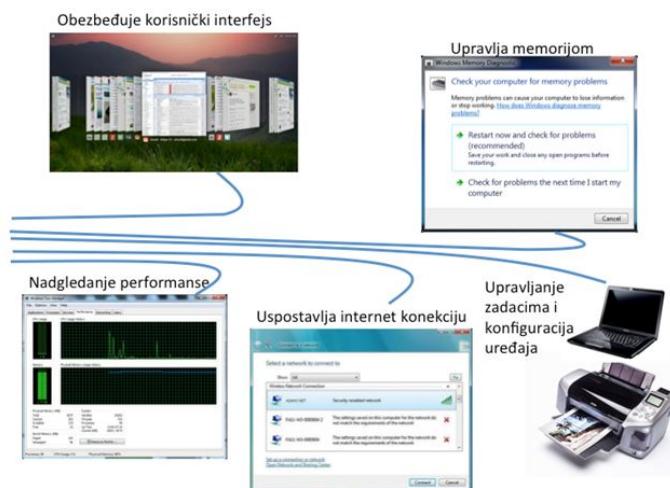
Operativni sistem upravlja radom računarskog sistema, nadgleda ulazno/izlazne operacije i operacije obrade podataka

Operativni sistem (engl. **operating system**(OS)) se sastoji od niza specifičnih programa čija je uloga da upravljuju radom samog računarskog sistema i usklade svu komunikaciju koja se obavlja između hardvera. OS, takođe, nadgleda sve ulazno/izlazne operacije sistema i operacije obrade podataka i upravlja njima. Zbog toga je rad računarskog sistema nemoguć bez operativnog sistema. **Operativni sistem predstavlja posrednika između korisnika, odnosno aplikativnih programa koje on koristi i hardvera, jer obezbeđuje izvršenje korisnikovih komandi i aplikativnih programa.** Programi operativnog sistema se mogu svrstati u četiri kategorije.

To su programi koji imaju sledeće funkcionalnosti (slika 1) :

- Pokretanje računara
- Gašenje računara
- Upravljanje memorijom (engl. **memory management**)
- Upravljanje zadacija (engl. **task management**)
- Nadgledanje performanse (engl. **performance monitoring**)
- Upravljanje datotekama (engl. **file management**)
- Automatsko ažuriranje (engl. **automatic updates**)
- Konfiguracija uređaja
- Obezbeđuje korisnički interfejs
- Uspostavljanje internet konekcije i sl.

Računari različitih veličina obično imaju različite operativne sisteme. Na primer, “**mainframe**” računari neće koristiti operativne sisteme koji se koristi na **personalnim računarima** (engl. **personal computers**(PC)). Čak ni sami PC računar ne koriste iste operativne sisteme. Sa druge strane, neki računari imaju mogućnost da rade sa nekoliko operativnih sistema. **Aplikativni softver je prilagođen za različite operativne sisteme.** Na primer, kada kupite Microsoft Office, treba imati u vidu da postoje posebne verzije za operativne sisteme Windows OS i Mac OS. Operativni sistemi nekada nazivamo i platformama.



Slika 1.2.1 Funkcionalnosti operativnog sistema

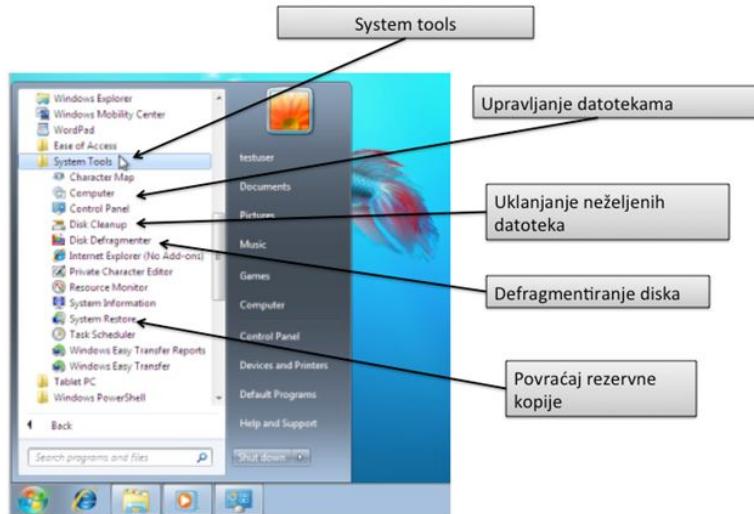
▼ 1.3 Uslužni programi

ULOГA I PRIMERI USLUŽNIH PROGRAMA

Uslužni programi su deo sistemskog softvera i predstavljaju nadogradnju operativnog sistema

Uslužni programi (engl. **utility programs**) su deo sistemskog softvera. Uslužni program se često naziva **“utility”**. Računar može da funkcioniše bez uslužnih programa, međutim, računar u tom slučaju ne bi bio u mogućnosti da u potpunosti koristi svoje resurse. U neku ruku, može se reći da uslužni programi predstavljaju nadogradnju operativnog sistema. Uslužni programi su obično napisane od strane specijalizovane softverske kuće. Korisnik može sam da odredi koje će uslužne programe da instalira kod sebe na računaru. Ovaj izbor se obično pravi na osnovu toga koje su funkcionalnosti neophodne da se obavljaju. Postoje razni uslužni programi. Većina operativnih sistema ima ugrađene uslužne programe kao što su

- **Systems tools**
- Alat za uravljanje datotekama (engl. **managing files**)
- Alat za uklanjanje neželjenih datoteka
- Alat za defragmentaciju diska
- Alat za povraćaj rezervne kopije datoteka (engl. **restoring backed up files**)
- Podešavanje **“screen saver”**
- Obezbeđenje računara od neovlašćenog pristupa
- Zaštita od virusa
- Filtriranje internet sadržaja
- Rezanje CD-a i sl.



Slika 1.3.1 Ugrađeni uslužni programi u Windows OS

▼ 1.4 Aplikativni softver

KATEGORIJE APLIKATIVNOG SOFTVERA

Aplikativni softver može biti: poslovni softver, softver za grafiku i multimedije, softver za ličnu upotrebu i softver za komunikaciju

Aplikativni softver se može podeliti prema njihovoj nameni i korišćenju u četiri kategorije:

1. **Poslovni softver** – koristi se da unapredi poslovne procese i da ih učini efikasnijim
2. **Softver za grafiku i multimedije** – koristi se da pomogne pri izradi projekata koji sadrže grafiku i multimedije
3. **Softver za kućnu, ličnu i obrazovnu upotrebu**
4. **Softver za komunikaciju**

Softver nije uvek striktno podeljen u ove četiri kategorije, niti jedan softver treba isključivo da pripada jednoj od ovih grupa. Na primer, jedan softver se može koristiti i u poslovne ali i obrazovne svrhe.

POSLOVNI SOFTVER I SOFTVER ZA GRAFIKU I MULTIMEDIJE

Namena poslovnog softvera je da poveća efikasnost u poslovanju.

Poslovni softver je aplikativni softver koji se koristi za povećanje efikasnosti u poslovanju. Poslovni softver može da sadrži programe kao što su programi za baze podataka (engl. **database**), tabele (engl. **spreadsheet**), obradu teksta (engl. **word processing**), prezentacije, upravljanje projektima (engl. **project management**), računovodstvo i sl. Primer poslovnog softvera prikazani su u tabeli 1.

Namena aplikativnog softvera	Proizvođač	Ime programa
Obrada teksta	Microsoft	Word
	Apple	Pages
Tabele	Microsoft	Excel
	Apple	Numbers
Presentation	Microsoft	Power point
	Apple	Keynote
Upravljanje projektima	Microsoft	Project
	Oracle	Project Manager

Slika 1.4.1 Tabela-1 Primeri poslovnog softvera

Pored poslovnog softvera, ljudi takođe koriste softver koji je specifičan za njihovu struku. Inženjeri, arhitekte, dizajneri i ostali često rade sa softverom koji im omogućava da lako manipulišu i kreiraju grafiku i multimedijalne sadržaje. **Softver za grafiku i dizajn multimedija** se koristi za kreiranje i obradu slika, video i audio montažu, kreiranje veb stranica, itd. Tabela 2 navodi neke primere softvera za dizajn grafike i multimedija.

Namena aplikativnog softvera	Proizvođač	Ime programa
Desktop publishing	Microsoft	Visio
	AutoDesk	AutoCAD
Obrada slika	Adobe	Illustrator
	Microsoft	Expression Design
Video montaža	Adobe	Soundbooth
	Sony	ACID Pro
Kreiranje veb stranica	Adobe	Dreamweaver
	Microsoft	Sharepoint

Slika 1.4.2 Tabela-2 Primeri softvera za grafiku i dizajn multimedija

SOFTVER ZA LIČNU UPOTREBU I SOFTVER ZA KOMUNIKACIJU

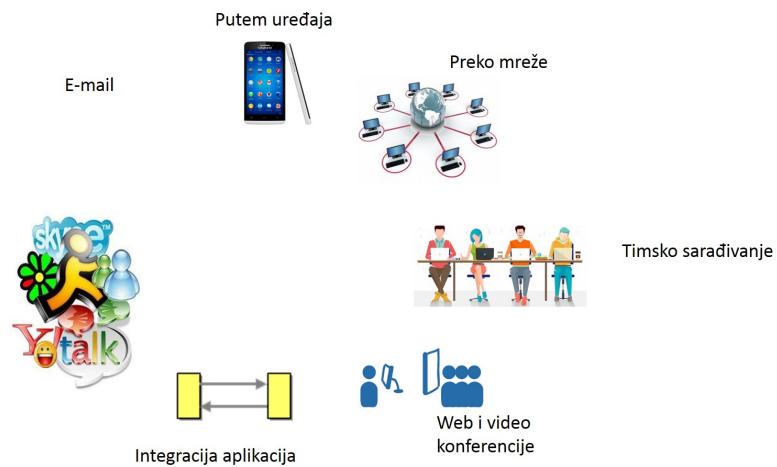
Softver za komunikaciju se koristi da poveže dva udaljena sistema

Veliki broj aplikativnog softvera je razvijen **za kućnu i ličnu upotrebu**. Osim toga, za razliku od poslovnog softvera, softver za ličnu upotrebu je obično jeftiniji, a često je i besplatan. Tabela 3 prikazuje listu softvera koji je prevashodno namenjen za ličnu upotrebu.

Namena aplikativnog softvera	Proizvođač	Ime programa
Priprema poreza	H&R Block	TaxCut
	Intuit	TurboTax
Lične finansije	IGG Software	iBank
	Intuit	Quicken
Desktop publishing	Microsoft	Publisher
	BorderBund	PrintMaster

Slika 1.4.3 Tabela-3 Primeri softvera za ličnu upotrebu

Softver za komunikaciju se koristi da se povežu dva udaljena sistema, kako bi se obavila razmena datoteka u različitim formatima (audio, video, tekst i sl.). Ovakav softver može podrazumevati ne samo mesindžere (engl. **messengers**) nego i softver za transfer datoteka (**file transfer protocol** (FTP)) i email. Primeri softvera za komunikaciju mogu se videti na slici 4



Slika 1.4.4 Primeri softvera za komunikaciju

✓ Poglavlje 2

Hardver računara

ZADATAK HARDVERA

Hardver računara ima zadatak da omogući brzo i tačno izvršavanje programa programskog sistema

Hardver računara ima zadatak da omogući brzo i tačno izvršavanje programa programskog sistema. Hardver računara, zavisno od svoje namene i tehnološkog nivoa, imaju različitu arhitekturu. Međutim, bez obzira na konfiguraciju hardvera, osnovna struktura većine računarskih sistema je ista. Hardver se sastoji od tri dela:

- **računara**
- **ulaznih uređaja**
- **izlaznih uređaja.**

Računar se sastoji od:

- centralne procesorske jedinice (engl. **Central processing unit(CPU)**)
- memorije
- magistrale
- napajanja
- portova i konektora.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

✓ 2.1 Centralna procesorka jedinica

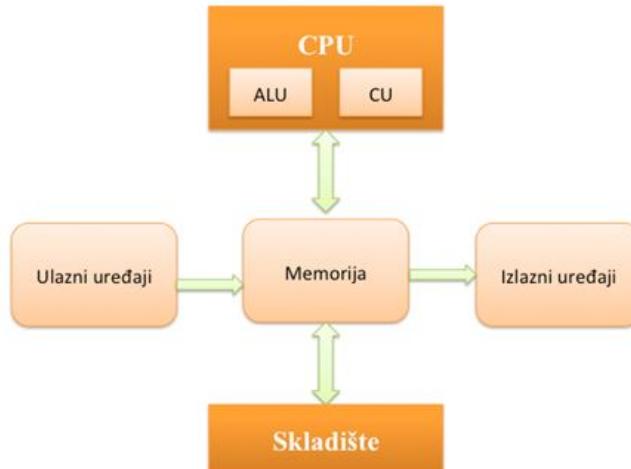
ZADATAK CENTRALNE PROCESORSKE JEDINICE

Zadatak centralne procesorske jedinice je da obradi informacije i upravlja radom računarskog sistema

Centralna procesorska jedinica (engl. **Central processing unit (CPU)**) ima zadatak da interpretira i obradi osnovne instrukcije računara. **Zadatak centralne procesorske jedinice je da obradi informacije i upravlja radom računarskog sistema.** Kako bi obavila ove zadatke, CPU se sastoji iz dva dela. CPU sadrži:

- Aritmetičko-logičku jedinicu (engl. Arithmetic logic unit(ALU))
- Upravljačku jedinicu (engl. Control unit(CU))

Uredaji koji su povezani na računar najčešće komuniciraju sa CPU kako bi obavili zahtevane operacije. Na primer, kada korisnik pokrene program koristeći miša kao ulazni uređaj, instrukcije programa se kopiraju iz skladišta u memoriju (Slika 1) .



Slika 2.1.1 Komunikacija sa centralnom procesorskom jedinicom

ARITMETIČKO-LOGIČKA JEDINICA

Aritmetičko-logička jedinica izvršava elementarne aritmetičke operacije nad binarnim brojevima, logičke operacije, operacije upoređivanja vrednosti i druge mašinske operacije

Aritmetičko-logička jedinica (ALU) izvršava elementarne aritmetičke operacije nad binarnim brojevima, logičke operacije, operacije upoređivanja vrednosti i druge mašinske operacije. ALU obavlja aritmetičke i logičke operacije prema informacijama dobijenim od kontrolne jedinice. Sve operacije se izvršavaju pomoću binarnih brojeva. **ALU obavlja aritmetičke operacije** kao što su sabiranje, oduzimanje, deljenje i množenje. Složenije matematičke operacije, kao što je kvadratni koren, se obavljaju preko navedenih osnovnih aritmetičkih operacija. Aritmetička jedinica može da obradi samo dva operanda, što znači da može da obavlja unarne i binarne operacije. U unarne operacije spadaju komplementi (engl. **complements**) i pomeranja (engl. **shifting**), a u **binarne operacije spadaju logičke operacije (I, ILI, ekskluzivno ILI) , sabiranje i oduzimanje.**

UPRAVLJAČKA JEDINICA

Upravljačka jedinica upravlja i sinhronizuje rad svih elemenata računarskog sistema

Upravljačka jedinica upravlja i sinhronizuje rad svih elemenata računarskog sistema tako što pravovremeno, na osnovu programa koji se trenutno izvršava, generiše upravljačke signale

ostalim komponentama računarskog sistema. Da bi to postigla, upravljačka jedinica najpre učitava jednu mašinsku naredbu iz operativne memorije i izvršava je, a zatim određuje sledeću naredbu za izvršenje. Upravljačka jedinica može da primi i obradi eksterne signale koji potiču od korisnika ili nekog drugog ulaznog uređaja. Pored toga, upravljačka jedinica sinhronizuje različite delove računara, upravlja aritmetičko-logičkom jedinicom i šalje potrebne kontrolne signale izlaznim uređajima.

INTEL X86 PROCESOR

Intel CPU-ovi su postali de facto standard za mnoge kompjuterske arhitekture

Nakon velikog uspeha arhitekture IBM PC-a, Intel CPU-ovi su postali de facto standard za mnoge kompjuterske arhitekture. Originalni PC koristi 4,77 MHz 16-bitni 8088 CPU. Sledeći model IBM PC / AT koristio je napredniji 16-bitni 80286. Nekoliko godina kasnije, Intel je proizveo 32-bitni 80386 i 80486 procesore. Pošto se ova imena završila brojem 86, generička arhitektura je nazvana x86. Kasnije su Intel procesori dobili imena koja su bila zvučnija potrošačima, pa su tako procesori dobili ime kao što je Pentium (uglavnom zato što Intel nije mogao dobiti patentirane brojeve kao ime), ali je arhitektura i dalje bila zasnovana na prvobitnom dizajnu x86. To je omogućilo kompatibilnost softvera sa prethodnim verzijama, pa je samim tim softver koji je bio napisan za 8088 mogao i dalje radi na kasnijim CPU modelima bez promene. U 2017. godini, najnoviji Intel x86 model je 22-jezgarni E5-2699A Xeon procesor, koji radi na 2,4 GHz.

AMD X86 PROCESOR

AMD je 1982. godine potpisao ugovor sa kompanijom Intel, postajući licencirani drugi proizvođač 8086 i 8088 procesora za IBM

Advanced Micro Devices, Inc. (AMD) bio je, i još uvek je, najveći konkurent Intelu. AMD je drugi najveći globalni snabdevač mikroprocesora zasnovan na x86 arhitekturi. AMD je 1982. godine potpisao ugovor sa kompanijom Intel, postajući licencirani drugi proizvođač 8086 i 8088 procesora za IBM. IBM je želeo da koristi Intel 8088 na svom IBM računaru, ali je IBM-ova politika u to vreme zahtevala najmanje dva izvora za svoje čipove. Intel je otkazao ugovor o licenciranju 1986. godine kako bi sprečio AMD-u da proizvede klon svog veoma uspešnog 80386 procesora. AMD je bio prisiljen da analizira arhitekturu 80386 i da kreira novi kompatibilni čip iz početka. AMD je 1991. godine objavio AM386, svoj klon procesora Intel 80386. Pošto je ovaj procesor bio veoma uspešan, AMD je primenio isti process i na 80486 i Pentium procesore. Analiza i proizvodnja već postojećeg čipa je veoma dugotrajan i skupan rad. AMD još uvek proizvodi x86 kompatibilne CPU-e, na taj način terajući Intel da konstantno uvodi inovacije i da održava cene relativno niskim. U 2017. godini, najnoviji model je 16-jezgarni AMD Opteron 6386 SE CPU, koji radi na 2,8 GHz.

ITANIUM I X86-64 PROCESORI

Danas se x86-64 arhitektura koristi u svim Intel i AMD procesorima.

Linija procesora Itanium bila je porodica 64-bitnih “**high-end**” procesora namenjenih visokokvalitetnim serverima i radnim stanicama. Arhitektura Itaniuma nije zasnovana na x86 arhitekturi. Da bi mogao da pokreće x86 softver (na primer Windows), Itanium je sadržao x86 emulator. Arhitekturu su zajednički razvili HP i Intel. Intel se prvo bitno nudio da će Itaniumova arhitektura (IA-64) zameniti x86 arhitekturu, ali HP je bila jedina kompanija koja je aktivno proizvodila sisteme zasnovane na Itaniumu, većinom koristeći HP-UX, a neke OpenVMS. Iako je inicijalno Windows mogao da radi i na sistemima zasnovanim na Itanium-u, Microsoft je zbog prekida proizvodnje prodao Windows podršku 2010. godine. AMD je 2005. izdao K8 procesorsku arhitekturu kao odgovor na Intelovu Itanium arhitekturu. K8 je uključio 64-bitno proširenje na skup x86 instrukcija. Kasnije je Intel prihvatio skup instrukcija AMS-ovog procesora kao produžetak za k86 procesora. Prvi Intel procesor koji je u potpunosti implementirao x86-64 bio je Xeon procesor. x86 16-bitni i 32-bitni skupovi instrukcija ostali implementirani na hardveru. Danas se x86-64 arhitektura koristi u svim Intel i AMD procesorima.

▼ 2.2 Ulagano-izlazni uređaji

ULOGA ULAZNO/IZLAZNIH UREĐAJA

Ulagano/izlazni uređaji služe za unos informacija i instrukcija, koji se dalje čuvaju i obrađuju, ali se koriste i za prikaz obrađenih podataka

Ulagano-izlazni uređaji ili, kako se često nazivaju, periferije služe za ostvarivanje komunikacije između računara i njegovog okruženja, kao i za trajno čuvanje informacija. **Ulagano/izlazni uređaji služe za unos informacija i instrukcija, koji se dalje čuvaju i obrađuju, ali se koriste i za prikaz obrađenih podataka.** Ulagano-izlazni uređaji mogu biti striktno ulazni ili striktno izlazni, ali postoje i uređaji koji imaju i ulazne i izlazne funkcije. Striktno ulazni uređaji su na primer tastatura i miš, dok su striktno izlani uređaji zvučnici. Uređaji koji mogu imati obe funkcionalnosti su ekran osjetljivi na dodir (engl. **touch screen**), zbog toga što im ekran služe kao izlazni uređaj da prikažu traženu informaciju, ali istovremeno se mogu izdati naredbe računaru preko njegovog ekrana koji je osjetljiv na dodir.

Specijalni tip ulazno/izlaznih uređaja su **spoljne memorije**. Karakteristika ovih uređaja je da omogućuju učitavanje (ulaz) informacija sa nekog memorijskog medija (nosilca informacija) u računar i/ili zapisivanje (izlaz) informacija iz računara na memorijski medijum. Ovi memorijski medijumi imaju mogućnost trajnog čuvanja zapisanih podataka. Pritstup podacima na spoljnim memorijama je mnogo sporiji nego pristup podacima u operativnoj memoriji.

✓ Poglavlje 3

Memorija

MEMORIJA U RAČUNARU

Memorija predstavlja skup elektronskih komponenti koji sadrži podatke i instrukcije koje su potrebne računarskom sistemu

Memorija predstavlja skup elektronskih komponenti koje su dizajnirane da skladište instrukcije koje čekaju procesorsku jedinicu da ih izvrši, da skladište podatke koje treba obraditi po tim instrukcijama i da skladište informacije koje su rezultat obrađenih podataka na kraju ovog ciklusa.

Operativni sistemi zajedno sa aplikativnim softverom postaju sve složeniji. Kompleksni softver povećava potrebu za većim memorijskim kapacitetom. U slučajevima kada postoji nedostatak potrebne memorije, tada obično dolazi do usporavanja izvršenja programa, a u ekstremnim slučajevima može se desiti da program prestane u potpunosti da se izvršava.

Memorija može da skladišti:

- operativne sisteme i sistemski softver
- aplikativne programe
- podatke koje obrađuju korisnički programi i dobijene informacije

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

✓ 3.1 Klasifikacija memorija

NAČINI KLASIFIKACIJE MEMORIJA

Memorije možemo klasifikovati prema načinu pristupa, mogućnosti izmene sadržaja, trajnosti zapisa, načinu pretraživanja, hijerarhijskim nivoima

Memorije možemo klasifikovati prema:

- načinu pristupa
- mogućnosti izmene sadržaja
- trajnosti zapisa

- načinu pretraživanja
- hijerarhijskim nivoima.

Prema načinu pristupa razlikujemo memorije sa sekvensijalnim pristupom (npr. magnetne trake), cikličnim pristupom (npr. CD), sa proizvoljnim pristupom (npr. RAM). Prema mogućnosti izmene sadržaja razlikujemo promenljive memorije za privremeno memorisanje, nepromenljive (ROM) i polupromenljive koje dozvoljavaju jedan upis sadržaja (PROM). Kada se ugasi računar, on gubi sve podatke iz promenljive memorije, dok nepromenljiva memorija ne gubi svoj sadržaj. Promenljiva memorija se koristi za skladištenje aktivnih programa i podataka koji su potrebni da se ovi programi izvršavaju. Pošto im je kapacitet ograničen, promenljive memorije sadrže samo podatke programa koji su trenutno aktivni na sistemu, a koji su neophodni za njihovo izvršenje. Preliminarni rezultati i izlazni rezultati se takođe mogu privremeno smeštati u memoriju.

Prema trajnosti zapisa memorije možemo podeliti na statičke koje zahtevaju stalno napajanje i dinamičke koje zahtevaju periodično obnavljanje sadržaja tokom rada. Prema načinu pretraživanja memorije delimo na adresne (sadržaju se pristupa uz pomoć jedinstvene adrese) i bezadresne memorije (npr. asocijativni pristup). Prema hijerarhijskom nivou memoriju možemo podeliti u primarnu (operativna memorija, RAM), sekundarnu (eksterna) i brzu memoriju (keš).

Najvažnije je memorija koja služi potrebama procesora. Na ovaj način, procesor može brzo da pristupi podacima i programima koji trenutno rade i privremeno skladišti obrađene rezultate u ovoj memoriji, a koji se mogu kasnije koristiti.

▼ 3.2 Strukture memorija

ELEMENTI MEMORIJE

Memorijska ćelija je elementarni deo memorije koji može da memoriše jedan bit. Skup memorijskih ćelija kojima se pristupa jedinstveno predstavlja memorijski registar

Bez obzira na vrstu, memorije se sastoje od:

- upravljačkog sistema memorije
- memorijskog medija.

Upravljački sistem memorije ima zadatak da, na osnovu zahteva procesora, odredi vrstu pristupa (ukoliko memorija ima mogućnost upisivanja), odredi položaj podataka u memoriji i vrši upis i/ili čitanje podataka.

Memorijski medijum ima zadatak da obezbedi čuvanje upisanih podataka sve dok postoji potreba za njihovim čitanjem. Memorijski medijum je po nekom pravilu podeljen na memorijske registre. Memorijski registar je skup memorijskih ćelija kojima se pristupa jedinstveno. Memorijska ćelija je elementarni deo memorije koji može da memoriše jednu

binarnu cifru ([bit](#)). Pristup samo jednoj memorijskoj ćeliji nije moguć. Ako se želi čitanje ili upisivanje u samo jednu memorijsku ćeliju, mora se pristupiti memorijskom registru u kome se ona nalazi.

Podatak koji se smešta u jedan memorijski registar naziva se [memorijska reč](#). Broj ćelija memorijskog registra definiše dužinu memorijske reči. Tako su poznate memorije čija je memorijska reč 8-bitna, 16-bitna, 32-bitna i sl.

Ulazni i izlazni kanali moraju da imaju onoliko veza i priključaka koliko je duga memorijska reč.

[Kapacitet memorije](#) je broj memorijskih registara celokupne memorije pomnožen dužinom memorijske reči. Ova veličina predstavlja broj memorijskih ćelija celokupne memorije (dakle broj bita). Pošto bi zbog kapaciteta današnjih memorija ovo bio neprikladno veliki broj, u praksi se kapacitet memorije izražava u bajtovima ili u memorijskim rečima, ali se pri tom obavezno kaže koja je dužina memorijske reči. Na primer, ako memorija ima 1024 memorijska registra dužine 16 bita, može se reći da je kapacitet memorije 2 KB ili 1 K 16-bitnih reči.

Kada se želi upis ili čitanje više memorijskih registara potrebno je obaviti više [ciklusa pristupa](#). [Vremenski interval između dva uzastopna pristupa memoriji se naziva memoriski ciklus](#). Vreme pristupa je vremenski interval koji protekne od trenutka kada procesor izda naredbu čitanja ili pisanja pa do završetka te radnje. Memorijski ciklus i vreme pristupa mogu da se razlikuju zbog tzv. mrtvog vremena.

Memorijski ciklus ne definiše potpuno brzinu upisa ili čitanja, jer ne uzima u obzir dužinu memorijske reči. Zato se za definisanje brzine memorije koristi druga mera - [širina opsega memorije](#), koja predstavlja odnos dužine memorijske reči i memorijskog ciklusa.

VRSTE PRISTUPA MEMORIJI

Podacima u memoriji se može pristupiti korišćenjem adresnog ili asocijativnog pristupnog metoda

Da bi se pristupilo nekom memorijskom registru potrebno je precizno locirati njegov položaj u memoriji. Ovo se može postići korišćenjem metoda:

- [adresnog pristupa](#)
- [asocijativnog pristupa](#).

[Adresni pristup](#) podrazumeva da svaki memorijski registar ima svoju adresu. Lociranje željenog memorijskog registra se postiže dovođenjem koda adrese na ulazni kanal upravljačkog sistema memorije. U ovom slučaju procesor određuje adresu memorijskog registra kome se želi pristupiti.

U slučaju [asocijativnog pristupa](#) memorijski registar se sastoji iz dva dela. U prvom delu se nalazi ključ (ili ime podatka) a u drugom sam podatak. Lociranje željenog memorijskog registra se vrši tako što procesor definiše ključ željenog registra, a zatim se vrši njegovo upoređivanje sa upisanim ključevima u memoriji. Ovo upoređivanje traje samo jedan [memorijski ciklus](#).

✓ 3.3 RAM

ŠTA JE RAM MEMORIJA?

RAM predstavlja privremenu memoriju koja ne skladišti podatke trajno, tako da se njen sadržaj uklanja

Najčešći tip promenljive memorije je RAM memorija. RAM je skraćenica za random access memory. RAM memorija predstavlja čip koji može da se koristi za unos i čitanje podataka iz memorije. RAM može da koristi procesor ili bilo koji drugi uređaj. Kao promenljiva memorija, RAM predstavlja privremenu memoriju koja ne skladišti podatke trajno, tako da se njen sadržaj uklanja. Kada se računar uključi, određene sistemske datoteke koje koristi operativni sistem se učitavaju u RAM sa hard diska. Pošto je RAM privremena memorija, ove datoteke će ostati u RAM-u dokle god je računar uključen. Postoje tri vrste RAM memorije:

- Dinamički RAM
- Statički RAM
- Magnetno-otporni RAM



Slika 3.1.1 RAM memorija

ŠTA JE RAM? (VIDEO)

What is RAM? Ultimate Guide to Computer RAM

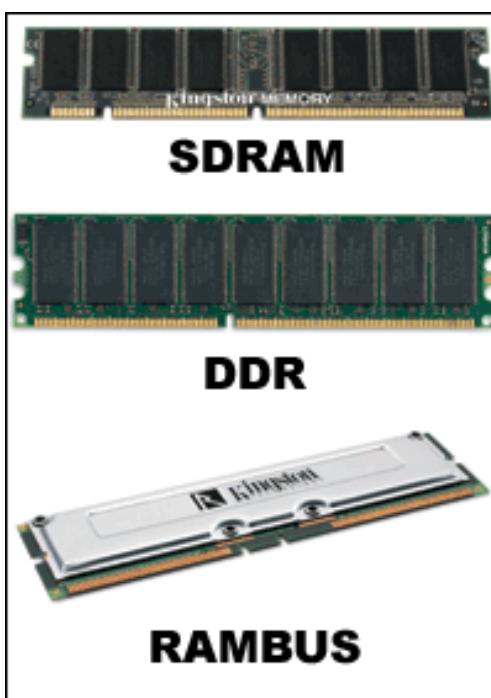
Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

❖ 3.4 DRAM

ŠTA JE DRAM MEMORIJA?

DRAM je memorija koja čuva svaki bit u kondenzatoru integrisanih kola

Dinamička memorija sa proizvoljnim pristupom **DRAM** je skraćenica za **Dynamic random access memory**. DRAM je memorija koja čuva svaki bit u kondenzatoru integrisanih kola. Pošto se električni napon gubi tokom vremena, obično je potrebno da se sadržaj memorije osveži svake milisekunde. Većina računara danas koristi ili **SDRAM** (**Synchronous dynamic random-access memory**) ili **RDRAM** (**Rambus DRAM**).



Slika 3.2.1 SDRAM, DDR, Rambus

PRODAJA DRAM MEMORIJE DOSTIGLA \$100 MILIJARDI U 2018-OJ GODINI

Cene DRAM memorije su visoke zbog nedovoljnog snabdevanja

"Anyone on the market for a new build, or for an upgrade, knows full well that DRAM prices are sky high due to "tight supply." The cycle that began two years ago is now coming to an end however, and if all predictions come true, 2019 will see a substantial break in DRAM pricing and a lot less burn in our pockets.

DRAM market analysts TrendForce and IC Insights have just released reports claiming it'll be a bumper year for DRAM makers as revenues will top \$100 billion by the end of 2018, with a 39% market growth.

Worldwide, in the second quarter of 2018, DRAM revenues grew 11.3% quarter-on-quarter, with Samsung, SK Hynix and Micron taking a mind-boggling 95% of the share. While market share has come and gone, the top three manufacturers show increased revenues across the board. SK Hynix, in particular, has clawed back market share and is showing considerable revenue growth of nearly 20%.

Another noticeable trend is that DRAM is now generating twice as much revenue as CPUs, according to IC Insights. And that is huge. For many years, CPUs (and similar processing units) have driven IC revenues, but it seems that is no longer the case. If you're on the market for a new build right now, you'll see that 32GB of top-tier DDR4 will set you back as much as the CPU. This is due to a number of reasons. DRAM has seen pressure from a number of markets, including smartphones, with production being diverted to mobile, low-power ICs, graphics cards and NAND, due to diverted production capacity, as seen in the case of Samsung's V-NAND/DRAM fabs.

Right now, the top three memory makers are all in varying stages of increasing capacity at their fabs, say the reports. While Samsung is building more fabs, Micron is migrating to more modern process nodes at their Taiwan operations, and SK Hynix is very busy improving yields on current-gen DRAM.

Despite the doom and gloom predictions of ever-increasing DRAM prices until year's end, all is not lost, say the analysts."

Izvor i ceo tekst na : <https://www.techspot.com/news/75949-dram-revenues-top-100-billion-2018.html>

3.5 SRAM

ŠTA JE SRAM MEMORIJA?

SRAM ne mora da osvežava memoriju da bi sačuvao sadržaj

Statička memorija sa proizvoljnim pristupom **SRAM** je skraćenica za **Static random access memory**. SRAM je poluprovodnička memorija koja koristi bistabilna električna kola. Za razliku od DRAM, SRAM ne mora da osvežava memoriju da bi sačuvao sadržaj i zbog toga su obično brži i pouzdaniji. S druge strane SRAM je obično skuplji od DRAM. Keš memorija koristi SRAM. Svaki bit u SRAM se čuva u tranzistoru i predstavlja memorijsku ćeliju. Ove ćelije imaju dva stanja, a obeležene su kao 0 ili 1. Obično SRAM koristi šest MOSFET-a za skladištenje.

Kako je SRAM mnogo skuplji od DRAM, ali i brži, DRAM se koristi kao radna memorija za obrade, a SRAM kao brza priručna memorija (**cache**) koja služi mikroprocesoru za interne radnje kao privremeno skladište podataka dok se ne omogući pristup sporijem DRAM-u, a koristi se i kao međumemorija u komunikaciji sa stalnom memorijom.

SRAM memorija se proizvodi na sličan način kao i procesori: visoko integrисани uzorci raspoređeni tranzistora se foto-graviraju u silicijumu. Svaki bit SRAM memorije se sastoji od

četiri do šest tranzistora, što je razlog zašto SRAM zauzima mnogo više prostora u poređenju sa DRAM memorijom, koja koristi samo jedan tranzistor po bitu (plus kondenzator). Ovo, kao i činjenica da je SRAM memorija nekoliko puta skuplja od DRAM memorije, objašnjava zašto se ona ne koristi u većoj meri u PC sistemima.

▼ 3.6 MRAM

ŠTA JE MRAM MEMORIJA?

MRAM koristi magnetske elemente koji su postavljeni na podlozi od silicijuma. Ova memorija ima neograničen broj ciklusa upisivanja, a brzina čitanja i pisanja su dosta ubrzane.

Magnetno-otporni RAM (Magnetoresistive RAM (MRAM)) koja koristi magnetske elemente koji su postavljeni na podlozi od silicijuma. Ova memorija ima neograničen broj ciklusa upisivanja, a brzina čitanja i pisanja su dosta ubrzane. Vreme koje je potrebno da se upiše prvi bit informacije u MRAM čip je oko milion puta kraće od vremena potrebnog da se izvrši upis u fleš memoriju. MRAM ne gubi svoj sadržaj kada se računar isključi. Kako se cena MRAM bude smanjivala vremenom, očekuje se da će MRAM zameniti MRAM i SRAM u masovnoj upotrebi.

Samsung i IBM zajedno razvijaju novi tip memorije MRAM

Dovoljno je reći da su Samsung i IBM sklopili partnerstvo da bi kompletna industrija počela osluškivati o čemu se radi. Ipak je riječ o tehnološkim gigantima, a ovaj put surađuju na razvoju MRAM (Magnetoresistive random-access memory) memorije, kojom se „zabavljala“ Toshiba 2012. godine, ali nije napravila pomak. Ove dvije kompanije bi mogle, a kad bi uspjеле, to bi značilo mnogo za industriju. Jer, MRAM troši manje energije od svega što danas postoji po pitanju memorijskih rješenja i samim time je idealan za Internet stvari (IoT). Koristi se magnetskim sustavom pod nazivom okret-transfer-moment (Spin-Transfer-Torque), čime se postže "čitanje" od samo deset nanosekundi za jedan bit, što je stotinu puta brže od jedne mikrosekunde za jedan bit, koliko za "čitanje" treba NAND flash. Dakle, u teoriji sve ovo zvuči više nego idealno i odlično, sad je samo pitanje mogu li Samsung i IBM postići sve što su naumili pa da ova priča dobije komercijalni smisao, postane standard u svijetu i pomakne tehnologiju još pokoji korak prema naprijed. Naravno, ove najave dodatno potiču konkurenčiju kao Qualcomm ili MediaTek da i sami nešto naprave po ovom pitanju pa bismo napredak trebali dočekati još i prije nego što mislimo. Bilo bi to sve korisno i u finansijskom smislu jer podaci prodaje dinamičke radne memorije (DRAM) za Samsung nisu baš na najboljim mogućim razinama. Cijene opasno padaju i trenutno se prosječna za jednu DRAM "pločicu" kreće oko 1,25 američkih dolara, što je utjecalo na Samsung i ostvarili su u drugom kvartalu ove godine 3,36 milijuna dolara, što je na godišnjoj razini pad od 16,6 posto. Jasno je, promjene su potrebne i na ovaj način se ništa ne postiže, što u Samsungu znaju pa stalno nešto rade, na svim područjima. Ponekad promaše, ali ne žele stagnirati i gledati što drugi rade pa se tome prilagođavati. Upravo zato suradnja s IBM-om može donijeti samo korist objema stranama.

Izvor: <http://www.ictbusiness.info/poslovna-rjesenja/samsung-i-ibm-zajedno-razvijaju-novi-tip-memorije-mram>

✓ 3.7 ROM

PERMANENTNE MEMORIJE

Permanentne memorije su one memorije čiji se sadržaj upisuje nekim posebnim postupkom i ne može se izmeniti naredbama računara

Permanentne memorije su one memorije čiji se sadržaj upisuje nekim posebnim postupkom i ne može se izmeniti naredbama računara. Drugim rečima, sadržaj ovih memorija se može samo čitati. Poznate su kao **ROM** (Read Only Memory) memorije. Sadržaj fiksnih memorija se ne gubi sa prekidom napajanja tako da se može reći da su one postojane. Permanentne memorije se koriste za one programe i podatke koji se ne menjaju (ili se menjaju vrlo retko) tokom eksploatacije računara. Tako, na primer, neki računarski sistemi imaju kompletan operativni sistem smešten u fiksnoj memoriji.

Permanentne memorije spadaju među memorije sa proizvoljnim pristupom. Uobičajeno se rade kao poluprovodničke memorije. Zavisno od načina upisa podataka, permanentne memorije se mogu podeliti na:

- **Permanentne ROM** (read-only memory)
- **Programabilne read-only memorije PROM** (Programmable read-only memory) - koristi mikro instrukcije da programira PROM čip; ne može se izbrisati ili menjati pošto se mogu programirati samo jedanom
- **Programabilno izbrisive read-only memorije EPROM** (Erasable programmable read-only memory) - mogu se brisati ultraljubičastim svetlom
- **Električno izbrisive programabilne read-only memorije EEPROM** (Electrically erasable programmable read-only memory) - omogućava brisanje i ponovno pisanje sadržaja električnim signalom.

Specijalni tipovi EEPROM su fleš memorije i EARM (Electrically Alterable Read Only Memory). EARM omogućuje da se menja jedan bit u jednom trenutku sa visokim naponom, što očigledno predstavlja veoma spor proces. Koristi se za sisteme koji ne zahtevaju česte promene, dok je fleš memorija dizajnirana za veliku izdržljivost i često se mogu podaci u nju ponovo upisivati.

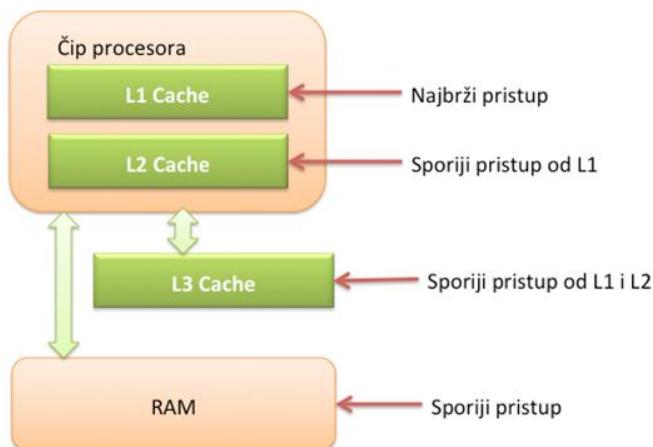
3.8 Keš memorija

PRISTUP KEŠ MEMORIJI

Keš memorije se nalaze na samom procesoru, pa je vreme potrebno za pristup podacima u njima malo

Većina računara ubrzava svoj rad korišćenjem keš memorije (engl. **cache**). Osnovni problem leži u činjenici da zapisivanje i čitanje podataka iz memorija velikog kapaciteta zahteva mnogo više vremena nego pri radu sa memorijama malog kapaciteta. Kod memorija velikog kapaciteta razlika između vremena pristupa registru i vremena pristupa memoriji je vrlo veliko. Zbog toga računarski sistemi obično imaju više vrsta memorija koje su organizovane hijerarhijski. Na vrhu su najbrže memorije malog kapaciteta. One se nalaze ili u samom procesoru ili vrlo blizu njega, pa je vreme potrebno za pristup podacima veoma malo.

Većina PC računara imaju L1 i L2 keš, dok neki imaju i L3 keš. L1 se naziva primarni keš, a L2 se naziva sekundarni keš. Primarni keš je najbrži tip memorije. On je ugrađen u čipu procesora (engl. **processor chip**) i često ima mali kapacitet. On se koristi da privremeno skladišti često korišćene instrukcije i podatke. Implementiran je kao SRAM memorija. L2 keš je nešto sporiji od L1. L2 je takođe ugrađen u čip procesora, ali sa većim kapacitetom od L1. L3 keš je procesor na matičnoj ploči i potpuno je odvojen od čipa procesora. Instrukcije se pretražuju u keš memoriji sledećim redosledom: L1, L2, L3 i onda RAM (Slika 1) .



Slika 3.3.1 Pristup različitim nivoima keš memorije

KEŠ MEMORIJA

Kako funkcioniše keš memorija

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 4

Magistrale

ULOGA MAGISTRALE U RAČUNARU

Magistrala predstavlja snop provodnika koji omogućuje prenošenje podataka između pojedinih komponenti računara

Centralna procesorska jedinica komunicira sa drugim delovima računara preko magistrale (u domaćoj literaturi se susreće i izraz sabirnica). Generalno, **magistrala** (engl. **bus**) se može posmatrati kao snop provodnika koji omogućuje prenošenje podataka između pojedinih podsistema (komponenata) računarskog sistema.

Svaki provodnik u nekom trenutku može da prenosi jedan bit. Magistrale se uobičajeno izvode sa 16, 32 ili 64 provodnika, pa se tako govori o 16-bitnim, 32-bitnim i 64-bitnim magistralama. Pošto različiti delovi računarskog sistema imaju potrebu da koriste magistralu, postoji opasnost od "sudara" podataka. Da se to ne bi dogodilo, samo jedna komponenta računara može da koristi magistralu u toku nekog vremenskog intervala. Ostale komponente, koje u tom trenutku imaju potrebu za magistralom, moraju da sačekaju da taj interval prođe, kako bi magistrala bila dodeljena njima. Budući da je vreme prenosa podataka kroz magistralu direktno proporcionalno njenoj dužini, teži se skraćivanju rastojanja između komponenata koje intenzivno komuniciraju, kako bi se skratilo vreme intervala. Osim toga, na brzinu rada računara direktno utiče i broj komponenata koje koriste istu magistralu. Veći broj komponenata povećava vreme čekanja neke komponente da joj se dodeli magistrala. Pored provodnika magistrala mora da ima i uređaj koji upravlja radom magistrale (dodeljuje pristup), koji se naziva kontroler magistrale.

Osnovni tipovi magistrala su FSB (engl. **front side bus**), BSB (engl. **backside bus**) i magistrala za proširenje (engl. **expansion bus**). FSB je deo matične ploče i povezuje procesor sa glavnom memorijom. BSB povezuje procesor sa kešom, a magistrala za proširenje omogućava procesoru da komunicira sa perifernim uređajima. Magistrale za proširenje koje se obično nalaze u PC računarima su PCI, PCI Express, AGP, USB i PC Card.

Magistrale mogu biti izgrađene po principu "od tačke do tačke" ili kao zajedničke. Magistrale tipa od tačke-do-tačke (**point-to-point**) spajaju samo dve komponente, na primer upravljačku i aritmetičko-logičku jedinicu. Koriste se u slučajevima kada je potrebno obezbediti maksimalno brzu komunikaciju između komponenata ili kada je razmena podataka toliko intenzivna da bi, zbog stalnog čekanja drugih komponenata, usporila rad celog računarskog sistema. U jednom savremenom računarskom sistemu ima više različitih magistrala, mada postoje i računari sa samo jednom magistralom. Svaka od magistrala ima svoj *protokol*, koji se sastoji od niza pravila za zajedničko korišćenje magistrale. Zavisno od toga kako protokol reguliše pravo za dodeljivanje magistrale na korišćenje, razlikuju se sinhrone i asinhrone magistrale.

ARHITEKTURA MAGISTRALE

Tri podsklopa magistrale su: adresna magistrala, magistrala za prenos podataka i kontrolna magistrala

Svaka magistrala se obično sastoji od 50 do 100 različitih fizičkih linija, podijeljenih u tri podsklopa:

- **Adresna magistrala** (engl. **address bus**), koja se ponekad naziva i memorijska magistrala (engl. **memory bus**), prenosi memorijske adrese kojima procesor želi da pristupi, kako bi čitao ili pisao podatke. Adresna magistrala je jednosmerna magistrala.
- **Magistrala za prenos podataka** (engl. **data bus**) prenosi instrukcije koja dolaze od ili se šalju u procesor. Ova magistrala je dvosmerna
- **Kontrolna magistrala** (engl. **control bus**) transportuje zahteve i sinhronizacijske signale koji dolaze iz kontrolne jedinice i putuju do svih ostalih hardverskih komponenti. Kontrolna magistrala je dvosmerna.

SINHRONE I ASINHRONE MAGISTRALE

Kod sinhronih magistrala prenos podataka mora da se obavi unutar jednog ili više ciklusa magistrale. Na asinhronu magistralu se može povezati veći broj komponenata različitih brzina

Sinhrone magistrale (engl. **synchronous bus**) rade po taktu čije je trajanje poznato. Jedno od najvažnijih pravila protokola sinhronih magistrala je da neki prenos podataka mora da se obavi unutar jednog ili više ciklusa magistrale. Ciklus magistrale je vremenski period između dva otkucaja internog sata magistrale. Ako, na primer, interni sat magistrale radi na taktu od 200 MHz, onda je trajanje jednog ciklusa $1/200.000.000$ ili 5 ns. Trajanje jednog ciklusa ne može biti kraće od vremena potrebnog da se izvrši prenos podataka sa jednog na drugi kraj magistrale.

Asinhronе magistrale (engl. **asynchronous bus**) imaju upravljačke linije za koordinaciju pristupa pojedinih komponenata magistrali i koriste mehanizam usaglašavanja (**handshaking protocol**). Po ovom protokolu vreme korišćenja magistrale od strane neke komponente nije ograničeno ciklusom magistrale, tako da se na istu magistralu može povezati veći broj komponenata različitih brzina. Svaka komponenta koja ima potrebe za magistralom preko upravljačke linije šalje upravljačkoj jedinici magistrale zahtev. Kada se zahtev odobri i izvrši transfer podataka, komponenta je dužna da potvrdi prijem, čime se magistrala oslobađa za druge komponente.

Sistemska magistrala prenosi podatka velikom brzinom i velikim propusnim opsegom i one predstavljaju sinhrone magistrale.

ARHITEKTURA MAGISTRALA

CompTIA A+

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 5

Primeri magistrala

PCI I PCI EXPRESS

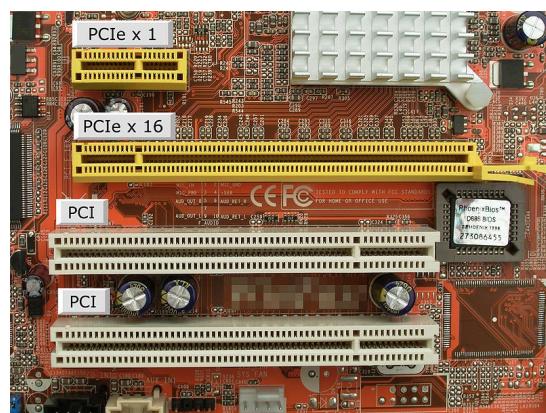
PCI magistrala se koristi da se povežu uređaji sa visokim brzinama sa centralnom procesorskom jedinicom

PCI

Peripheral Component Interconnect (**PCI**) magistrala se koristi da se povežu uređaji sa visokim brzinama sa centralnom procesorskom jedinicom. Projektovao ju je Intel i koristi se od 1993. godine. PCI podržava 32-bitni i 64-bitni paralelni protok podataka, pa je praktično istisnula ISA i EISA magistrale. PCI magistrala spada u klasu sinhronih magistrala. U praksi se češće koristi 32-bitna verzija magistrale. 32-bit PCI magistrala koristi 62 pina, pri čemu se 32 pina koriste za multipleksirani prenos adresa i podataka. U 64-bitnoj varijanti se koriste 94 pina. Vrste kartica koje se mogu ubaciti u PCI slot uključuju video kartice, zvučne kartice, SCSI kartice i mrežne kartice. Primer slota za PCI na matičnoj ploči se može videti na slici 1.

PCI Express

PCI Express je objavio PCI Special Interest Group4 (PCI-SIG) 2002. godine. Razlog zašto je kreirana ova magistrala je taj što je PCI magistrala fleksibilna, skalabilna i da može da podrži velike komponente. Međutim, paralelni način prenosa bitova duž magistrale (32 bita istovremeno) je stvarao probleme. Zbog kašnjenja u prenosu svi signali ne dolaze u isto vreme na drugi kraj magistrale. PCI Express je dvosmerna serijska magistrala od tačke do tačke. Ovakav dvosmerni bitni prenos se naziva i magistralom koja funkcioniše u punom dupleksu. Ovaj režim rada obezbeđuje mnogo veću frekvenciju, veći propusni opseg i smanjuje kašnjenje koje je veoma bitno za video prenos.

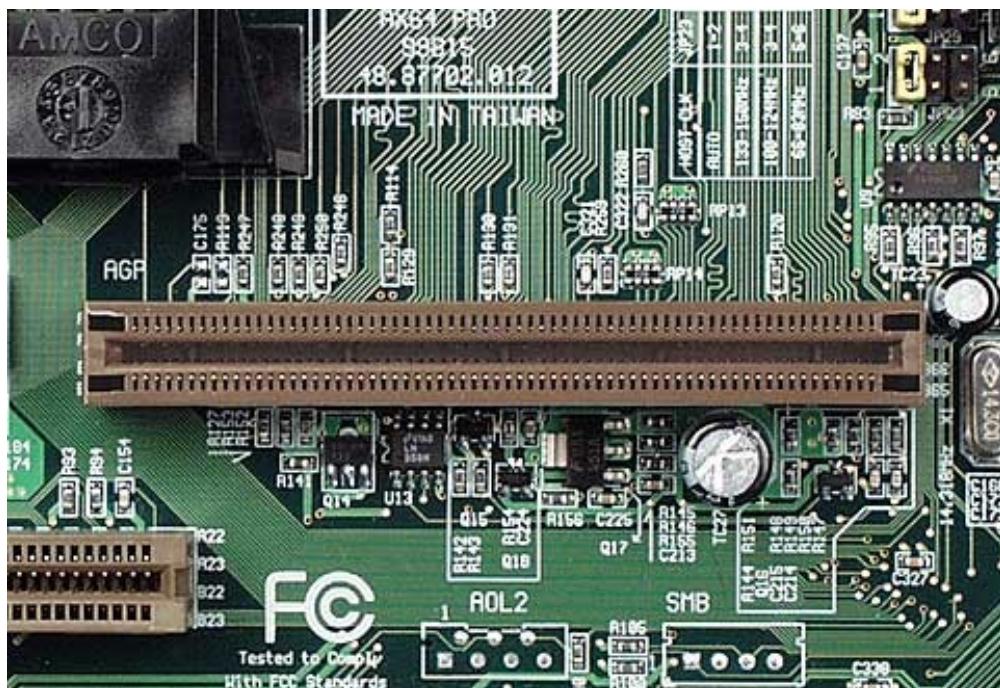


Slika 5.1 PCI slot

AGP

AGP je sofisticirana PCI magistrala specijalno prilagođena za video

AGP (Accelerated Graphics Port) je sofisticirana PCI magistrala specijalno prilagođena za video. Pojavom grafičkih okruženja, kao što su WindoWs i OS X, porasla je potreba za velikim protokom podataka od procesora do grafičke kartice. Dakle, PCI magistrala je postajala sve opterećenija. Rešenje je nađeno u "point to point" magistrali. Prva AGP 1.0 specifikacija je razvijena od strane kompanije Intel 1996. godine. Dve godine kasnije, 1998., nova specifikacija je objavljen za AGP 2.0, koja je definisala i AGP-4X magistralu. Specifikacije objavljene 2000-te godine je napisana kako bi definisala AGP 8X magistralu. Ovu magistralu je dizajnirao Intel kako bi poboljšao brzine prenosa 3D grafike i videa. Primer slota za AGP na matičnoj ploči se može videti na slici 2.



Slika 5.2 AGP slot

PCI EXPRESS (PCIE) 3.0

PCle komatibilnost i performanse

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

USB

USB uređaj se konektuje van sistemske jedinice pomoću kabla koji se povezuje na USB port

USB (Universal serial bus) je magistrala koja ne mora da instalira kartice u magistrali za proširenje. USB uređaj se konektuje van sistemske jedinice pomoću kabla koji se povezuje na USB port. Međutim, USB port je povezan sa PCI magistralom na matičnoj ploči. Primer slota za USB se može videti na slici 3.



Slika 5.3 USB slot

USB danas koriste mnogi izlazni i ulazni uređaji kao što su tastature, pokazivački uređaji, digitalne kamere, štampači, portabilni medija plejeri, disk uređaji, mrežni adapteri, smart mobilni telefoni, PDA i konzole za igre. USB standardi koji do danas postoje su 1.1, 2.0, 3.0 i 3.1. Treba napomenuti da USB koristi više vrsta konektora, koje možemo svrstati u tip A i tip B konektore. Ovakav dizajn se koristi kako bi se izbegla naponska preopterećenja, pa samim tim i oštećenje opreme, pošto je tip A konektor povezan sa električnim izvorom.



Slika 5.4 USB tipovi konektora

▼ Poglavlje 6

Vežba: Korišćenje Excel statističkih funkcija

AVERAGEIF FUNKCIJA

AVERAGE IF funkcija se koristi da bi pronašli prosečnu vrednost u skupu vrednosti ćelija koje zadovoljavaju određeni kriterijum

Predviđeno vreme pokazne vežbe je 90 minuta.

AVERAGE IF funkcija se koristi da bi pronašli prosečnu vrednost u skupu vrednosti ćelija koje zadovoljavaju određeni kriterijum. Opšta formula za **AVERAGE IF** funkciju je:
=AVERAGEIF(Range, Criteria, Sum Range)

Range - predstavlja grupu ćelija koju funkcija treba da pretraži.

Criteria - određuje da li ćelija treba da se računa ili ne.

Average_range - opseg podataka za koji se računa prosek ako prvi opseg zadovoljava zadate kriterijume. Ako se ovaj opseg izostavi umesto njega za prosek se računa prvi opseg

PRIMER AVERAGEIF FUNKCIJE

Odgovor 25 bi trebalo da se pojavi u ćeliji F7

1. Unesite sledeće vrednosti u ćelije E1 do E6: 114,165,178,143,130,165.
2. Unesite sledeće vrednosti u ćelije F1 do F6: 10, 20, 30, 10, 20, 30.
3. Kliknite na ćeliju F7 – mesto gde će rezultat biti prikazan.
4. Kliknite na tab Formulas
5. Izaberite More Functions > Statistical da bi otvorili listu funkcija.
6. Kliknite na AVERAGEIF u listi da bi ste otvorili dialog box za funkciju.
7. U dialog box-u, kliknite na Range liniju.
proseka. Prosek vrednosti za 20 i 30 je 25.
8. Označite ćelije E1 to E6 u radnoj tabeli.
9. U liniji Criteria u dialog box-u napišite "165".

10. Kliknite na Average_range liniju.
11. Označite ćelije F1 do F6 u radnoj tabeli.
12. Klinkite na OK.
13. Odgovor 25 bi trebalo da se pojavi u ćeliji F7. Pošto kriterijum jednakosti sa 165 zadovoljavaju samo dve ćelije - E2 i E6, samo njihove odgovarajuće ćelije - F2 i F6 ulaze u računanje

		fx	=AVERAGEIF(E1:E6,165,F1:F6)
D	E	F	
	114	10	
	165	20	
	178	30	
	143	10	
	130	20	
	165	30	
		25	

Slika 6.1 AVERAGEIF

MEDIAN FUNKCIJA

MEDIAN funkcija, jedna od Excelovih statističkih funkcija prikazuje srednju vrednost u listi brojeva.

MEDIAN funkcija, jedna od Excelovih statističkih funkcija prikazuje srednju vrednost u listi brojeva.

Srednja vrednost, u ovom slučaju, se odnosi na aritmetičku veličinu a ne na položaj brojeva u listi. Ako postoji paran skup brojeva, njihova medijana je srednja vrednost od sredine dve vrednosti.

Opšta formula za MEDIAN funkciju je:

=MEDIAN(number1, number2, ...number255)

Do 255 brojeva može biti uneto u funkciju.

PRIMER MEDIAN FUNKCIJE

Odgovor 24 bi trebalo da se pojavi u ćeliji E1

Unesite sledeće podatke u ćelije D1 do D5: 4,12,49,24,65.

2. Kliknite na ćeliju E1 – mesto gde će rezultat biti prikazan.

3. Kliknite na Formulas tab.
4. Izaberite More Functions > Statistical da bi ste otvorili listu sa funkcijama.
5. Kliknite na MEDIAN u listi kako bi ste otvorili dialog box funkcije.
6. Označite ćelije D1 do D5 u radnoj tabeli kako bi ste uneli opseg u dialog box.
7. Kliknite na OK.
8. Odgovor 24 bi trebalo da se pojavi u ćeliji E1 pošto su dva broja veća (49 i 65) a dva broja su manja (4 i 12) od njega u listi.
9. Kompletna funkcija **=MEDIAN(D1:D5)** se vidi u formula bar-u iznad radnog lista kada kliknete na ćeliju F1.

	f x	=MEDIAN(D1:D5)
D	E	
	4	24
	12	
	49	
	24	
	65	

Slika 6.2 MEDIAN

MODE FUNKCIJA

MODE funkcija, jedna od Excel-ovih statističkih funkcija nam pokazuje koja vrednost se najčešće pojavljuje u listi brojeva

MODE funkcija, jedna od Excel-ovih statističkih funkcija nam pokazuje koja vrednost se najčešće pojavljuje u listi brojeva.

Opšta formula funkcije je:

=MODE(number1, number2, ...number255)

Do 255 brojeva može biti uneto u funkciju.

1. Unesite sledeće podatke u ćelije D1 to D6: 98,135,147,135,98,135.
2. Kliknite na ćeliju E1 – mesto gde će biti prikazan rezultat.
3. Kliknite na Formulas tab.
4. Izaberite More Functions > Statistical da bi ste otvorili listu sa funkcijama.

5. Kliknite na Mode u listi kako bi ste otvorili dialog box funkcije.
6. Označite ćelije D1 do D6 u radnoj tabeli kako bi ste uneli opseg u dialog box.
7. Kliknite na OK.
8. Odgovor 135 bi trebalo da se pojavi u ćeliji E1 pošto se ovaj broj pojavljuje najveći broj puta (3 puta) u listi podataka.
9. Kompletan funkciju **=MODE(D1:D6)** se vidi u formula bar-u iznad radnog lista kada kliknete na ćeliju E1.

D	E	F
98	135	
135		
147		
135		
98		
135		

Slika 6.3 MODE

RANK FUNKCIJA

RANK funkcija rangira veličine brojeva u poređenju sa drugim brojevima u listi podataka.

RANK funkcija, jedna od Excel-ovih statističkih funkcija, rangira veličine brojeva u poređenju sa drugim brojevima u listi podataka.

Opšta formula RANK funkcije je:

=RANK(Number, Ref, Order)

Number – referenca na ćeliju čiji broj zelimo da rangiramo.

Ref – Opseg ćelija koje će biti korišćene u rangiranju Number parametra.

Order – Određuje da li će broj biti rangiran u opadajućem ili rastućem poretku.

Upišite "0" (nula) da bi ste rangirali u opadajućem poretku (od najvećeg do najmanjeg). Upišite "1" da bi ste rangirali u rastućem redosledu (od najmanjeg do najvećeg).

PRIMER RANK FUNKCIJE

Broj 3 bi trebalo da se pojavi u ćeliji E1 jer je broj 135 treći najveći broj

1. Unesite sledeće podatke u ćelije D1 do D6: 123,135,147,130,98,187.
2. Kliknite na ćeliju E1 – mesto gde će rezultat biti prikazan.
3. Kliknite na Formulas tab.
4. Izaberite More Functions > Statistical da bi ste otvorili listu sa funkcijama.
5. Kliknite na Rank u listi kako bi ste otvorili dialog box funkcije.
6. Kliknite na ćeliju D2 kako bio ste izabrali broj koji želimo da rangiramo (135).
7. Kliknite na "Ref" liniju u dialog box-u.
8. Označite ćelije D1 do D6 na spreadsheet-u kako bi ste uneli opseg u dialog box.
9. Kliknite na "Order" liniju u dialog box-u.
10. Upišite nula u ovoj liniji kako bi ste brojeve rangirali u opadajućem poretku.
11. Kliknite na OK.
12. Broj 3 bi trebalo da se pojavi u ćeliji E1 jer je broj 135 treći najveći broj.
13. Kompletna funkcija **=RANK(D2, D1:D6, 0)** se vidi u formula bar-u iznad radnog lista kada kliknete na ćeliju E1.

D	E	F
123	3	
135		
147		
130		
98		
187		

Slika 6.4 RANK

SMALL FUNKCIJA

SMALL funkcija može se koristiti za pronalaženje podataka u odnosu na relativnu veličinu

SMALL funkcija, jedna od Excel-ovih statističkih funkcija može se koristiti za pronalaženje podataka u odnosu na relativnu veličinu.

Opšta formula za SMALL funkciju je:

=SMALL(Array, K)

Array – Niz ili opseg ćelija koje sadrže podatke koji će biti korišćeni u funkciji.

K – K-ta najmanja vrednost, kao što je npr. treća najmanja vrednost.

PRIMER SMALL FUNKCIJE

Broj 130 bi trebalo da se prikaže u ćeliji E1 posto je treći najmanji broj

1. Unesite sledeće podatke u ćelije D1 to D6: 123,135,147,130,98,187.
2. Kliknite na ćeliju E1 – mesto gde će rezultat biti prikazan.
3. Kliknite na Formulas tab.
4. Izaberite More Functions > Statistical da bi ste otvorili listu sa funkcijama.
5. Kliknite na Small u listi kako bi ste otvorili dialog box funkcije.
6. Kliknite na Array liniju u dialog box-u.
7. Označite ćelije D1 do D6 na spreadsheet-u kako bi ste uneli opseg u dialog box.
8. Kliknite na K liniju u dialog box-u.
9. Otkucajte 3 (tri) na ovoj liniji kako bio ste pronašli treću najmanju vrednost u izabranom opsegu vrednosti.
10. Kliknite na OK.
11. Broj 130 bi trebalo da se prikaže u ćeliji E1 posto je treći najmanji broj (brojevi 123 i 98 su manji po veličini).
12. Kompletna funkcija =**SMALL(D1:D6, 3)** se vidi u formula bar-u iznad radnog lista kada kliknete na ćeliju E1.

	D	E	F
	123	130	
	135		
	147		
	130		
	98		
	187		

Slika 6.5 SMALL

LARGE FUNKCIJA

LARGE funkcija se može koristiti za pronalaženje podataka u odnosu na relativnu veličinu.

LARGE Funkcija:

LARGE funkcija, jedna od Excel-ovih statističkih funkcija koja se može koristiti za pronalaženje podataka u odnosu na relativnu veličinu.

Opšta formula LARGE funkcije je:

=LARGE(Array, K)

Array - Niz ili opseg ćelija koje sadrže podatke koji će biti korišćeni u funkciji.

K - K-ta najveća vrednost, kao što je npr. treća najveća vrednost.

PRIMER LARGE FUNKCIJE

Broj 135 bi trebalo da se pojavi u ćeliji E1 pošto je treći najveći broj

1. Unesite sledeće podatke u ćelije D1 do D6: 123,135,147,130,98,187.
2. Kliknite na ćeliju E1 – mesto gde će rezultat biti prikazan.
3. Kliknite na Formulas tab.
4. Izaberite More Functions > Statistical da bi ste otvorili listu sa funkcijama.
5. Kliknite na Large u listi kako bi ste otvorili dialog box funkcije.
6. Kliknite na Array liniju u dialog box-u.
7. Označite ćelije D1 do D6 u radnoj tabeli kako bi ste uneli opseg u dialog box.
8. Kliknite na K liniju u dialog box-u.

9. Otkucajte 3 (tri) na ovoj liniji kako bio ste pronašli treću najveću vrednost u izabranom opsegu vrednosti.

10. Kliknite OK.

11. Broj 135 bi trebalo da se pojavi u ćeliji E1 pošto je treći najveći broj (brojevi 187 i 147 u listi su

veći po veličini).

12. Kompletna funkcija =**LARGE(D1:D6, 3)** se vidi u formula bar-u iznad worksheet-a kada kliknete na ćeliju E1.

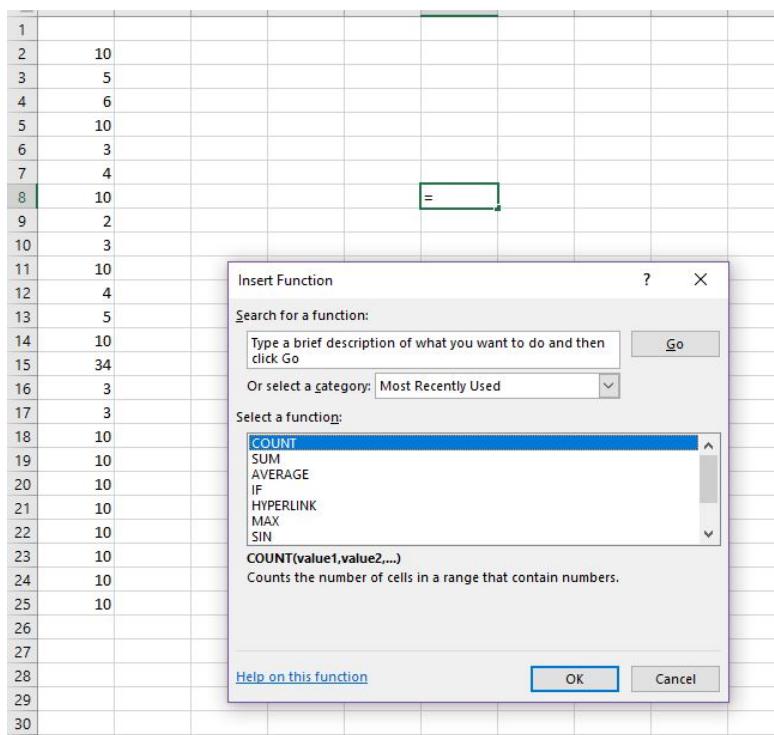
	f _x	=LARGE(D1:D6,3)
D	E	F
123	135	
135		
147		
130		
98		
187		

Slika 6.6 LARGE

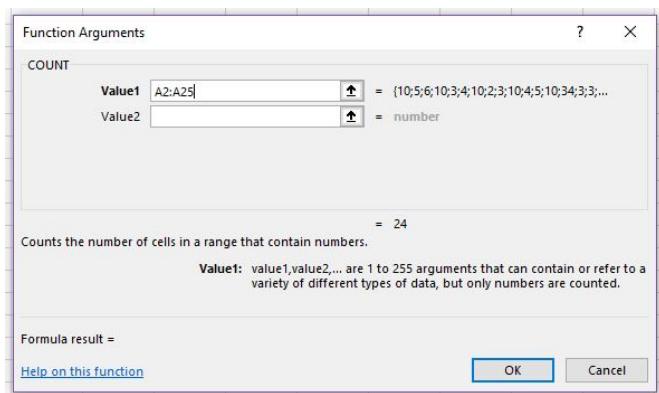
COUNT FUNKCIJA

Prikaz COUNT funkcije

COUNT funkcija broji ćelije koje sadrže brojeve, kao i brojeve na listi argumenata. Ukoliko se upotrebi funkcija COUNT dobija se broj stavki u polju za broj koje se nalazi u opsegu ili nizu brojeva. Na primer, možete uneti sledeću formulu da biste prebrojali brojeve u opsegu A2:A25: =**COUNT(A2:A25)**.



Slika 6.7 Odabir COUNT funkcije



Slika 6.8 Prikaz odabira polja

U datom primeru ukoliko ako 24 ćelija iz tog opsega sadrži brojeve, rezultat će biti 24.

5	
6	
10	
3	
4	
10	
2	
3	
10	
4	
5	
10	
34	
	24

Slika 6.9 Rezultat COUNT funkcije

MIN FUNKCIJA

Statistička funkcija MIN daje minimalnu vrednost iz skupa vrednosti.

Opšta formula glasi:

=MIN(broj1, [broj2], ...)

Broj1, broj2, ... Argument broj1 je opcionalan, a naredni brojevi su opcionalni i to od 1, do 255 brojeva za koje želite da pronađete minimalnu vrednost.

Šta mogu biti argumenti: brojevi, imena, nizovi ili reference koje sadrže brojeve.

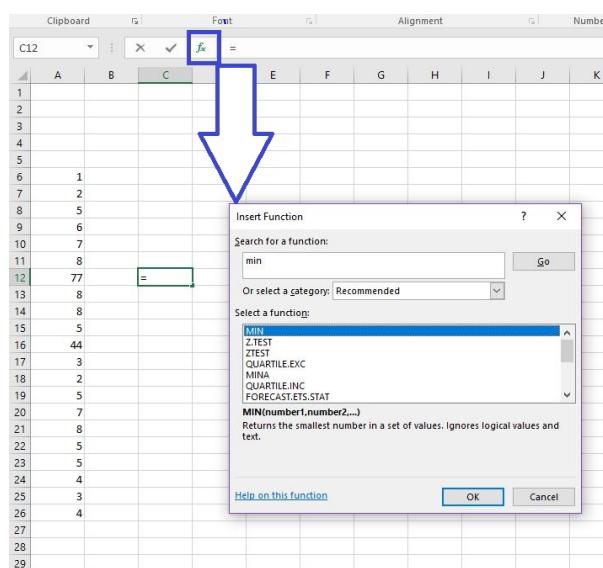
Ukoliko je argument niz ili referencia, koriste se samo brojevi u tom nizu ili referenci. Prazne celije, logičke vrednosti ili tekst u nizu ili referenci se zanemaruju.

Ako argumenti ne sadrže brojeve, MIN daje 0.

PRIMER MIN FUNKCIJE

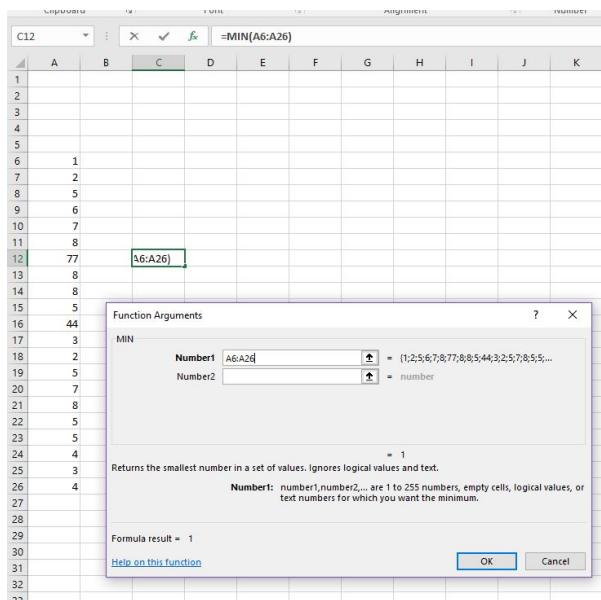
Prikaz MIN funkcije na primeru

Na početku smo uneli niz brojeva. Kako bi koristili funkciju MIN odabrali smo je u tabu funkcija:



Slika 6.10 Odabir funkcije Min

Zatim odabiramo koja polja uključujemo u funkciju - Između kojih polja biramo MIN.



Slika 6.11 Odabir polja funkcije

Na kraju je prikazan rezultat koji je dala funkcija.

C12	A	B	C	D	E	F
1						
2						
3						
4						
5						
6	1					
7	2					
8	5					
9	6					
10	7					
11	8					
12	77		1			
13	8					
14	8					
15	5					
16	44					
17	3					
18	2					
19	5					
20	7					
21	8					
22	5					
23	5					
24	4					
25	3					
26	4					
27						

Slika 6.12 Prikaz rešenja MIN funkcije

MAX FUNCKIJA

Statistička funkcija MAX daje najveći broj iz skupa vrednosti.

Opšta formula glasi:

=MAX(broj1, [broj2], ...)

Broj1, broj2, ... Argument broj1 je opcionalan, a naredni brojevi su opcionalni i to od 1, do 255 brojeva za koje se želi pronaći maksimalna vrednost.

Šta mogu biti argumenti: brojevi, imena, nizovi ili reference koje sadrže brojeve.

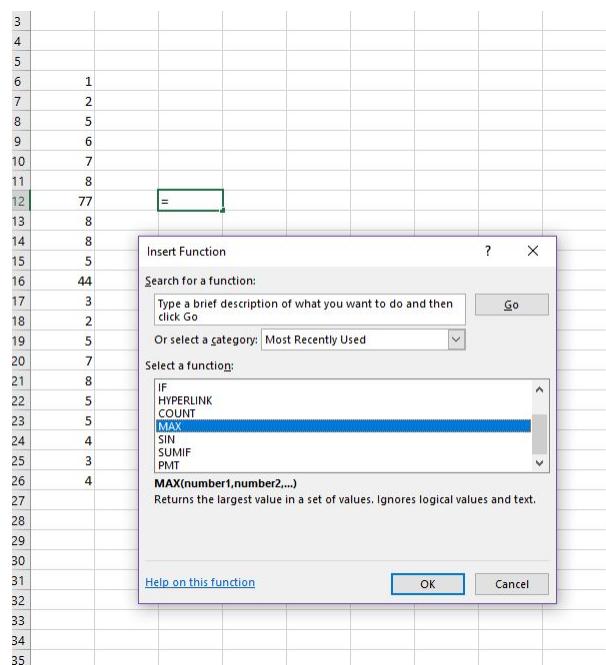
Ukoliko je argument niz ili referenca, koriste se samo brojevi u tom nizu ili referenci. Prazne celije, logičke vrednosti ili tekst u nizu ili referenci se zanemaruju.

Ako argumenti ne sadrže brojeve, MIN daje 0.

PRIMER MAX FUNKCIJE

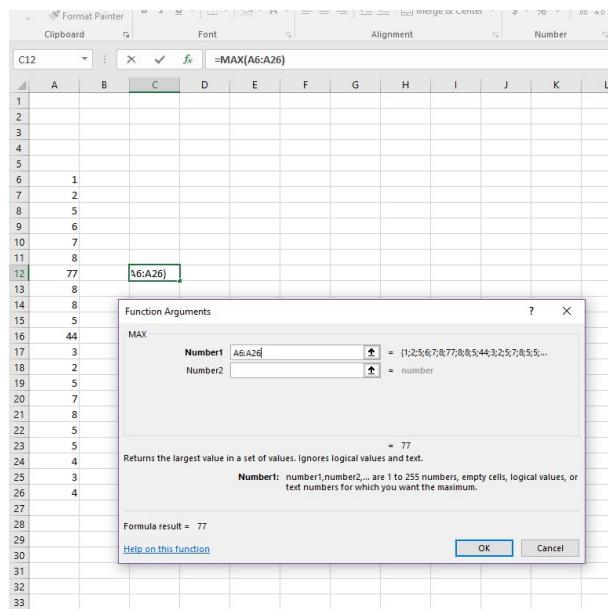
Prikaz MAX funkcije na primeru

Na početku smo uneli niz brojeva. Kako bi pronašli maksimalnu vrednost, koristili smo funkciju MAX i odabrali smo je u tabu funkcija:



Slika 6.13 Odabir funkcije MAX

Zatim odabiramo koja polja uključujemo u funkciju - Između kojih polja biramo MAX.



Slika 6.14 Odabir polja funkcije

Na kraju je prikazan rezultat koji je dala funkcija.

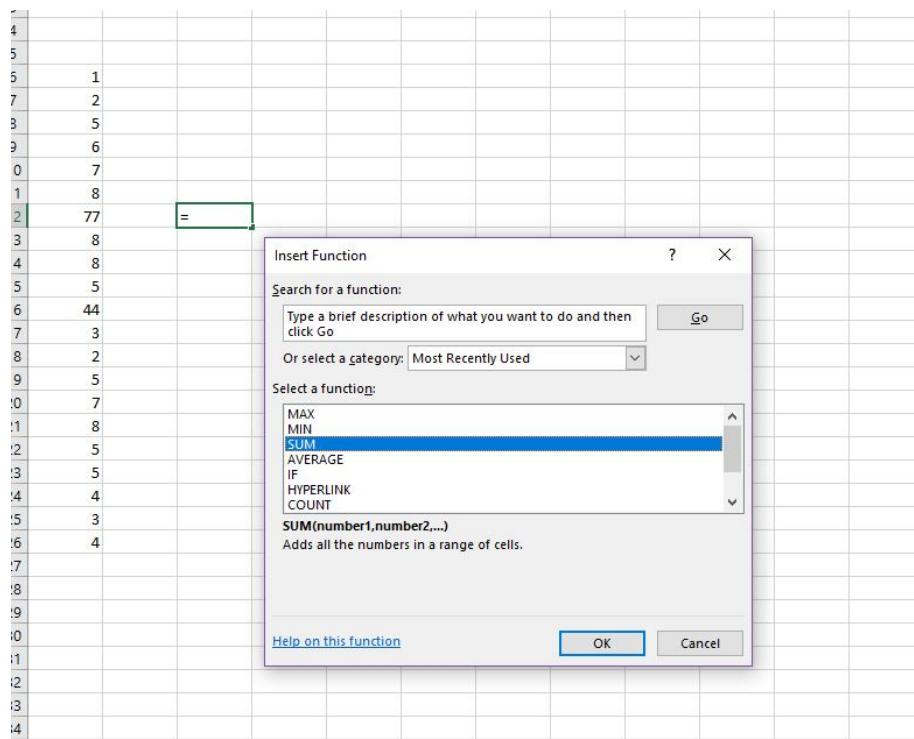
Clipboard						Font					
C12						=MAX(A6:A26+R14)					
A	B	C	D	E	F	A	B	C	D	E	F
1											
2											
3											
4											
5		1									
6		2									
7		5									
8		6									
9		7									
10		8									
11		8									
12		77						77			
13		8									
14		8									
15		5									
16		44									
17		3									
18		2									
19		5									
20		7									
21		8									
22		5									
23		5									
24		4									
25		3									
26		4									
27											
28											
29											
30											
31											
32											
33											

Slika 6.15 Prikaz rešenja MAX funkcije

SUM FUNKCIJA - OBNAVLJANJE

Prikaz SUM funkcije

Kao što je već obrađeno u prethodnoj lekciji, SUM funkcija služi kako bi se sumirali broevi. Kao što je već rečeno u prethodnoj lekciji, matematička funkcija SUM sumira-sabira vrednosti. Možete dodati pojedinačne vrednosti, reference ili opsege celija ili kombinaciju sva tri.



Slika 6.16 Korišćenje SUM funkcije

Clipboard						
C12	A	B	C	D	E	F
1						
2						
3						
4						
5						
6	1					
7	2					
8	5					
9	6					
10	7					
11	8					
12	77		217			
13	8					
14	8					
15	5					
16	44					
17	3					
18	2					
19	5					
20	7					
21	8					
22	5					
23	5					
24	4					
25	3					
26	4					
27						
28						

Slika 6.17 Prikaz izračunate sume

POKAZNI PRIMER SA REŠENJEM

Prikaz zadataka sa rešenjem

Predviđeno vreme izrade pokaznog primera je 30 minuta.

Data je excel tabela koja je dobijena iz banke kao dugovanja kreditne kartice. Na osnovu tabele potrebno je izracunati:

1. Koliko je ukupno dugovanje izraženo u RSD? (neizvršene transakcije)
2. Koliko je ukupno dugovanje izraženo u EUR? (neizvršene transakcije)
3. Koliko je iznosila minimalna transakcija u EUR? (neizvršene i izvršene)
4. Koliko je iznosila maksimalna transakcija u RSD? (neizvršene i izvršene)
5. Koliko je ukupno bilo transakcija?
6. Koliko je ukupno bilo transakcija u EUR?
7. Koliko je ukupno bilo transakcija u RSD?

Rešenje:

Kako bi se rešili sledeći zadaci potrebno je koristiti funkcije: SUM, MAX, MIN, COUNT.

Iznos	Valuta	Mesto koriscenja	Status
1.43	EUR	AliExpress.com	Neizvršen
4.31	EUR	AliExpress.com	Neizvršen
2285	RSD	Caffe Teatar	Neizvršen
54.04	EUR	RyanAir JXR0	Neizvršen
569	RSD	AliExpress.com	Neizvršen
1.47	EUR	Plotun DOO Krusevac	Neizvršen
1129	RSD	AliExpress.com	Neizvršen
1.01	EUR	AliExpress.com	Neizvršen
14.33	EUR	AliExpress.com	Neizvršen
3.12	EUR	AliExpress.com	Neizvršen
1.07	EUR	AliExpress.com	Neizvršen
2.04	EUR	AliExpress.com	Neizvršen
3.55	EUR	AliExpress.com	Neizvršen
0.74	EUR	AliExpress.com	Neizvršen
1111.5	RSD	dm filijala 37	Neizvršen
1.53	EUR	AliExpress.com	Neizvršen
0.74	EUR	AliExpress.com	Neizvršen
1.24	EUR	AliExpress.com	Neizvršen
1.24	EUR	AliExpress.com	Neizvršen
1.45	EUR	AliExpress.com	Neizvršen
437.98	RSD	Mercator-S DOO	Izvršen
10.19	EUR	RyanAir UST0	Izvršen
69.99	EUR	Wizz Air HU EH2DHB	Izvršen
325	RSD	Good Coffee DOO	Izvršen

Slika 6.18 Prikaz tabele dobijene iz banke

REŠENJE POKAZNOG PRIMERA

Na narednoj slici prikazano je rešenje pokaznog primera.

Iznos	Valuta	Mesto koriscenja	Status			Drugi nacin
1.43	EUR	AliExpress.com	Neizvrsen	1	5094.5	5094.5
4.31	EUR	AliExpress.com	Neizvrsen	2	93.31	93.31
2285	RSD	Caffe Teatar	Neizvrsen	3	0.74	
54.04	EUR	RyanAir JXR0	Neizvrsen			
569	RSD	AliExpress.com	Neizvrsen	4	2285	
1.47	EUR	Plotun DOO Krusevac	Neizvrsen			
1129	RSD	AliExpress.com	Neizvrsen	5	24	
1.01	EUR	AliExpress.com	Neizvrsen			
14.33	EUR	AliExpress.com	Neizvrsen	6	18	18
3.12	EUR	AliExpress.com	Neizvrsen			
1.07	EUR	AliExpress.com	Neizvrsen	7	6	6
2.04	EUR	AliExpress.com	Neizvrsen			
3.55	EUR	AliExpress.com	Neizvrsen			
0.74	EUR	AliExpress.com	Neizvrsen			
1111.5	RSD	dm filijala 37	Neizvrsen			
1.53	EUR	AliExpress.com	Neizvrsen			
0.74	EUR	AliExpress.com	Neizvrsen			
1.24	EUR	AliExpress.com	Neizvrsen			
1.24	EUR	AliExpress.com	Neizvrsen			
1.45	EUR	AliExpress.com	Neizvrsen			
437.98	RSD	Mercator-S D000	Izvrsen			
10.19	EUR	RyanAir UST0	Izvrsen			
69.99	EUR	Wizz Air HUEH2DHB	Izvrsen			
325	RSD	Good Coffee D000	Izvrsen			

Slika 6.19 Prikaz rešenja

▼ Poglavlje 7

Zadaci za samostalni rad: Excel funkcije

ZADATAK ZA SAMOSTALNI RAD

Excel tabela

Vreme predviđeno za izradu sledećih zadataka je 30 minuta.

- Napraviti Excel tabelu za računanje zarade knjižare i održavanje robe na lageru
- U tabeli iz dela (a) napraviti signalizaciju koja proverava da li je dnevni pazar u plusu ili minusu, kao i da obaveštava kada je potrebno naručiti neki artikal pošto više nije na lageru

KONFIGURACIJA RAČUNARA

Prikaz pronalaženja konfiguracije računara

Vreme predviđeno za izradu sledećih zadataka je 15 minuta.

Zadatak:

Pronaći na Internetu laptop računar do 70 000 RSD sa što boljim komponentama. Potrebno je kupiti računar u Srbiji, pa se pretraga ograničava na domaće sajtove.

Navesti link do sajta kao i cenu i proizvođača i oznaku laptop računara. Opisati šta ste uspeli da ispunite, a šta ne ne prelazeći budžet.

Potrebe su:

1. SSD HDD (koliko maksimalno može da priušti ne prelazeći budžet), ukoliko ne, onda 1TB SATA disk.
2. Što bolji Intelov procesor (pokušati pronaći i7, ukoliko ne i5 pa tek zatim i3)
3. Rezolucija ekrana mora biti 1920 x 1080
4. Što više RAM DDR4 memorije
5. Težiti ka tome da poseduje dodatnu grafičku pored integrisane.

PRIKAZ JEDNOG REŠENJA

Prikaz jednog od rešenja

U datom trenutku, laptop koji je pronađen i iskorišćavao je sve karakteristike je laptop:
https://gigatronshop.com/laptop_racunari/hp_15bs047nm_2kg97ea-108441

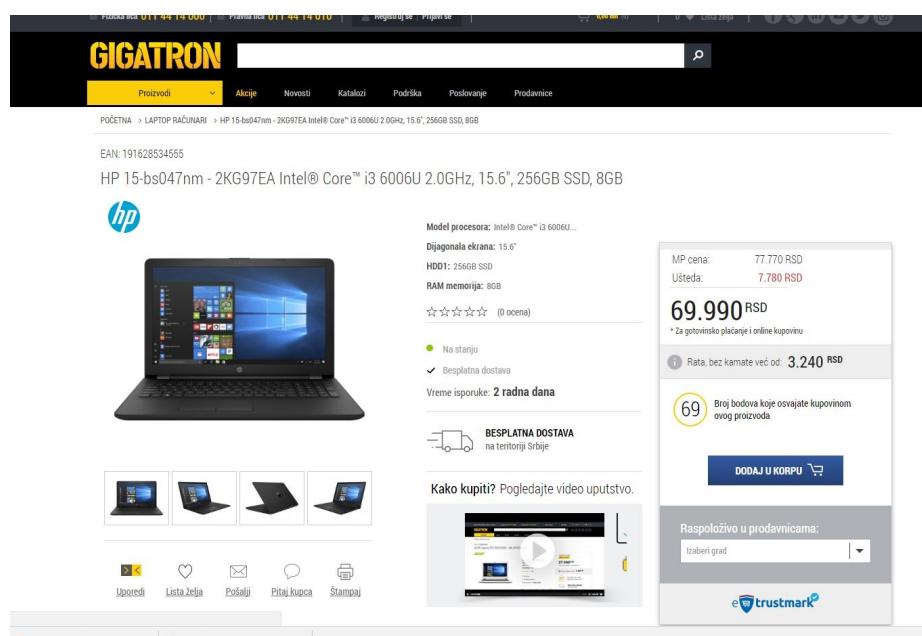
Konfiguracija:

Prednost je:

- HDD je 256 SSD
- Postoji dodatna grafička: AMD Radeon 520 sa 2GB DDR3 sopstvene memorije
- RAM je 8 GB

Mane su:

- Rezolucija ekrana je 1366 x 768 i nije koja je navedena
- Procesor je i3



Slika 7.1 Prikaz računara

✓ Poglavlje 8

Domaći zadatak

OPIS DOMAĆEG ZADATKA - DZ03

Uputstvo za izradu i slanje zadatka

Predviđeno vreme za izradu domaćeg zadatka je 30 minuta.

- Radni prostor snimite kao **IT101-DZ03-Ime_Prezime_brojIndexa**
- U radnom prostoru je potrebno da pošaljete excel datoteke
- Sve datoteke je potrebno arhivirati u ZIP fajlu. Naziv arhiviranog fajla treba da bude **IT101-dz03_ime_prezime_brojIndexa.zip**
- **Domaći zadatak pošaljite predmetnom asistentu na e-mail,** a u subject-u mejla napisati **IT101 - DZ03**

ZADATAK 1

Koristeći MS Excel ili OpenOffice Calc, napraviti tabelu sa predispitnim poenima predmeta 1. godine koje pratite. Primeniti sve prikazane excel funkcije.

ZADATAK 2

Uz pomoć internet pretraživača pronaći podatke o cenama komponenata i imenama komponenata (potrebno je imati sve vrste komponenata iz vežbanja). Koristeći MS Excel ili OpenOffice Calc potrebno je osmislići i kreirati tabelu koja će sadržati sve komponente za jedan funkcionalni računar. Za svaku komponentu je potrebno navesti cenu (npr procesor Intel q9300 core2quad - 170e) i izračunati komponentu koja je najskuplja, najjeftinija, prosečna cena svih komponenti, kao i koliko će koštati kreirana konfiguracija Izbrojati koliko ukupno komponenti kupujete.. (Za realizaciju zadatka iskoristiti funkcije MIN, MAX, SUM, AVERAGE, COUNT).

Dopunske instrukcije:

Uz tabelu kreirajte zaseban dokument u kom ćete navesti zašto ste izabrali željenu konfiguraciju, kao i odakle ste preuzeли i informacije cena, zajedno sa naglašenim informacijama u sumarnoj tabeli za najskuplju i najjeftiniji komponentu, prosečne cene svih komponenti i konačnu cenu celog računara. Ovaj dokument sačuvati pod nazivom IT101-dz03_ime_prezime_brojIndexa.doc

▼ ZAKLJUČAK

ZAKLJUČAK

U ovom predavanju smo objasnili rad računara kroz njegove osnovne hardverske i softverske komponente. Hardver smo grubo podelili na sistemske jedinice, ulazne i izlazne uređaje. Dok smo računarski softver podelili u dve grupe sistemski softver i aplikativni softver. Specijalni fokus smo dali na memorijama, njihovoj klasifikaciji i strukturi.

Literatura

1. Gary B. Shelly, Misty E. Vermaat, Discovering Computers 2011-Introductory: Living in a Digital World, Cengage Learning 2010.
2. W3, The Global Structure of HTML Document, <http://www.w3.org/TR/1999/PR-html40-19990824/struct/global.html> (17.07.2014).



IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

ULAZNI I IZLAZNI UREĐAJI

Lekcija 04

PRIRUČNIK ZA STUDENTE

IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

Lekcija 04

ULAZNI I IZLAZNI UREĐAJI

- ▼ ULAZNI I IZLAZNI UREĐAJI
- ▼ Poglavlje 1: Podaci i informacije
- ▼ Poglavlje 2: Ulazni uređaji
- ▼ Poglavlje 3: Izlazni uređaji
- ▼ Poglavlje 4: Štampači
- ▼ Poglavlje 5: Displeji
- ▼ Poglavlje 6: Pokazna vežba: Programi za obradu rasterskih slika
- ▼ Poglavlje 7: Pokazne vežbe: Kreiranje Power Point prezentacije
- ▼ Poglavlje 8: Zadaci za samostalni rad: PowerPoint prezentacija
- ▼ Poglavlje 9: Domaći zadatak
- ▼ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

❖ Uvod

UVOD

Cilj ovog predavanja je da da pregled osnovnih ulaznih i izlaznih uređaja i da objasni način njihovog funkcionisanja

Kako smo već razmatrali, informacione tehnologije se zasivaju na radu ljudskog mozga i ljudske percepcije. Samim tim, način na koji mi obradujemo neke informacije su takođe važne za rad računara. Kako bi mogli na adekvatan način da izdamo instrukcije računaru, ali istovremeno i da nam računar da povratnu informaciju na način na koji je nama najlakše razumljiv, potrebno je da imamo adekvatne uređaje koji će nam to omogućiti. Cilj ovog predavanja je da da pregled osnovnih ulaznih i izlaznih uređaja i da objasni način njihovog funkcionisanja.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 1

Podaci i informacije

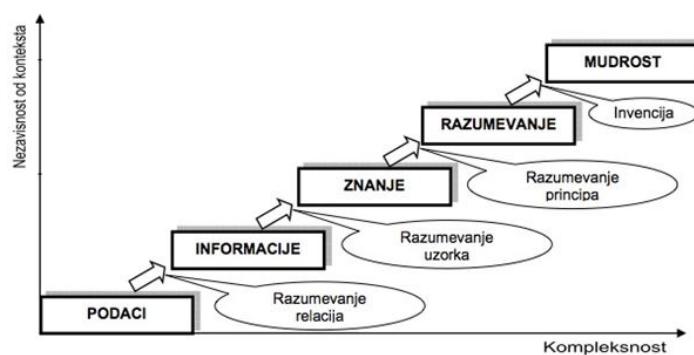
ODNOS IZMEĐU PODATAKA, INFORMACIJA, ZNANJA, RAZUMEVANJA I MUDROSTI

Podaci predstavljaju činjenice, informacije obrađene podatke, znanje predstavlja primenu podataka i informacija, razumevanje shvatanje razloga, a mudrost izvedeno razumevanje

Poznato je da sadržaj ljudskog mozga može da bude podeljen na pet kategorija, koje se najkraće mogu definisati na sledeći način:

- **Podaci**: simboli, sirove činjenice.
- **Informacije**: obrađeni podaci koji pružaju odgovore na pitanja ko, šta, gde i kada.
- **Znanje**: primena podataka i informacija radi odgovora na pitanje kako.
- **Razumevanje**: shvatanje razloga (odgovor na pitanje zašto). Razumevanje je interpolativan i probabilistički proces. Na osnovu prethodnih znanja njime se može generisati novo znanje.
- **Mudrost**: izvedeno razumevanje. Prema nekim autorima mudrost je ekstrapolativan i neprobabilistički proces koji na osnovu postojećeg razumevanja može da generiše novo razumevanje koje je do tada bilo nepoznato. Mudrost može da se definiše i kao veština korišćenja postojećeg znanja.

Odnos između ovih kategorija je prikazan na slici 1. Informacioni sistemi barataju sa prve četiri kategorije. Za sada se ne očekuje da mudrost postane svojstvena informacionim sistemima.



Slika 1.1 Odnos između podataka, informacija, znanja, razumevanja i mudrosti

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

VEZA IZMEĐU PODATKA I INFORMACIJE

Informacije se dobijaju izborom potrebnih podataka iz skupa raspoloživih podataka i njihovom obradom

Pojmovi informacija i podatak ni u informacionim tehnologijama nemaju isto značenje. U praksi se, međutim, ovi pojmovi vrlo često pogrešno upotrebljavaju, pa će se zato ovde odvojiti više prostora za njihovo razjašnjenje.

Podatak je sirova, nestruktuirana, zabeležena činjenica. Značenje samog podatka zavisi od konteksta, a njegovim tumačenjem u nekom kontekstu se dobija informacija. Računari obrađuju podatke bez razumevanja njihovog značenja. Podaci kao što su na primer: **12, crveno, ne, A2, video zapis, zvuk**, ne govore ništa onome ko ih prima ako ne shvata njihovo značenje. Ako nije poznat smisao podataka (kontekst u kome se koriste), onda oni samo predstavljaju činjenicu bez značenja. Svaki podatak karakteriše: simbol, opis, kontekst.

Informacija je potreban podatak čije je značenje poznato. Reč informacija potiče od latinske reči informare što znači informisanje ili obaveštavanje. Informacija se može definisati kao protumačeni podatak u nekom kontekstu. Informacija, pored značenja, obavezno uključuje i podatak. Ona predstavlja odgovor na neko pitanje. Ukoliko primaocu značenje podatka nije poznato ili nema potrebe za njim, on nema nikakvu informaciju. Informacije se dobijaju izborom potrebnih podataka iz skupa raspoloživih podataka i njihovom obradom. Obrada podataka se može definisati kao skup aktivnosti kojim se podaci transformišu u informacije.

Primer 1

Kontekst: Ispit iz predmeta IT101 se održava januarskom ispitnom roku u učionici U1

Opis podataka: Vreme, Datum, Učionica, Predmet

Podaci: 10-12, 26.01.2017, U1, IT101

Primer 2

Red letenja nekog avio-prevoznika je primer skupa podataka. Ako je red letenja štampan na jeziku koji korisnik ne razume, ovi podaci su potpuno nekorisni, jer recimo korisnik ne može da sazna kada sredom ima let za London i na kom izlazu se vrši prijem putnika. Ukoliko korisnik razume jezik, potrebno je da izvrši obradu podataka da bi došao do informacije. U ovom slučaju je potrebno da među svim stranama pronađe stranu koja se odnosi na letove za London i da pročita vreme poletanja sredom, kao i broj izlaza za prijem putnika.

PRIMER PODATAKA I INFORMACIJA

Prezime i godine predstavljaju podatak. Odgovor na pitanje „ko su zaposleni mlađi od 30 godina“ predstavlja informaciju

Primer 3

U tabeli 1 je prikazan primer koji treba da pojasni razliku između podataka i informacija i objasni proces transformacije podataka u informacije. Leva tabela predstavlja podatke. Oni se sastoje od liste prezimena zaposlenih i liste brojeva koji predstavljaju godine. Ako se uvede relacija između liste prezimena i liste sa godinama tako da se svakom prezimenu dodeli odgovarajući broj godina, onda je moguće dobiti informaciju o broju godina svih zaposlenih.

Međutim, ako postoji potreba da se sazna ko su zaposleni mlađi od 30 godina, potrebno je izvršiti određenu obradu podataka. U ovom slučaju potrebno je odbaciti one zaposlene koji su stariji od 30 godina. Na taj način se dolazi do desne tabele koja sadrži samo zaposlene mlađe od 30 godina.

Budući da ova tabela sadrži samo potrebne podatke, ona predstavlja informaciju. Nepotrebni podaci (o zaposlenima starijim od 30 godina) su odbačeni. U nekim slučajevima podatke je potrebno i dodatno obraditi. U ovom primeru je izvršeno i sortiranje zaposlenih prema broju godina u rastućem redosledu.

Podaci	
Prezime	Godine
Petrović	23
Ivanić	40
Simanić	29
Ivković	38
Stojković	39
Zarić	27

Slika 1.2 Tabela-1 Podaci (primer)

Informacija	
Prezime	Godine
Petrović	23
Simanić	29
Zarić	27

Slika 1.3 Tabela-2 Informacije (primer)

INFORMACIJA je rezultat obrade podataka i pojavljuje se kao značajna za one koji je dobiju u specifičnom domenu, odnosno datom kontekstu. Informacija zavisi od: konteksta i predznanja primaoca informacija.

KADA SU INFORMACIJE VREDNE ZA KORISNIKA?

Informacije su vredne za korisnika kada su tačne, kompletne, pouzdane, relevantne, ažurne, ekonomične i perceptivne

Da bi bile vredne za korisnike informacije treba da zadovoljavaju niz karakteristika:

1. **Tačnost.** Pogrešne informacije se mogu dobiti **ako su podaci koji su obradivani pogrešni ili ako se podacima dodeli pogrešan smisao.** **Ako se podatak broj godina odnosi na godine radnog staža, a ne starosti, onda su dobijene informacije netačne.**
2. **Kompletност.** **Kompletna informacija sadrži sve potrebne podatke.** Vrednost informacije vrlo brzo opada ako neki podaci nedostaju, a u većini slučajeva informacija postaje bezvredna. **Informacija o poletanju aviona je bezvredna ako se zna samo vreme poletanja, a ne zna se sa kog izlaza.** Kompletne informacije se mogu dobiti samo na osnovu svih potrebnih relevantnih podataka.
3. **Pouzdanost.** **Pouzdanost informacija se zasniva na pouzdanosti podataka.** Pouzdanost podataka zavisi od metode prikupljanja podataka i pouzdanosti izvora. Ukoliko postoji sumnja u pouzdanost izvora podataka, potrebno je obezbediti alternativne izvore radi verifikacije podataka.
4. **Relevantnost.** **Informacija treba da bude relevantna za korisnika, jer u suprotnom ona postaje bezvredna.** Na primer, **putniku za London je irrelevantno gde je ulaz za putnike za Pariz.** Irrelevantne informacije opterećuju korisnika i mogu da izazovu zabunu.
5. **Ažurnost.** **Informacije treba da budu dostupne korisniku u trenutku kad su mu potrebne.** Informacije koje kasne postaju bezvredne, a mogu i da nanesu štetu (**avion je odleteo**). Da bi informacije bile dostupne na vreme, potrebno je unapred obezbediti relevantne podatke. Osim toga, vreme potrebno za obradu podataka ne treba da ugrozi ažurnost (**koga interesuje prognoza vremena za juče**).
6. **Ekonomičnost.** **U nekim slučajevima obrada podataka radi dobijanja informacija može da zahteva skupe računarske resurse i/ili da zahteva vrlo veliko vreme za obradu.** Tada je potrebno odmeriti da li vrednost informacija opravdava potrebna finansijska ulaganja.
7. **Perceptivnost.** **Informacije treba da budu prilagođene percepционим osobinama korisnika.** Korisnici mogu da koriste različite organe za prijem informacija koje takođe mogu da budu u različitom obliku. Tako su, na primer, **odštampane informacije bezvredne za slepe korisnike.** **Korisniku koji vidi fotografiju neke osobe ta fotografija je mnogo vrednija informacija nego njen opis** (crna kosa, plave oči, isturene jagodice). Takođe će fotografija u boji predstavljati vredniju informaciju nego crno-bela fotografija.

VIDEO - KARAKTERISTIKE INFORMACIJA

Prodiskutujmo karakteristike koje je potrebno da imaju informacije da bi bile korisne za korisnika

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

AKTIVNOSTI NAD PODACIMA

Nad podacima se može izvršiti više aktivnosti kao što su: klasifikacija, sortiranje, odabir, agregacija, računanje

Nad podacima se može izvršiti više aktivnosti kao što su:

- klasifikacija - podrazumeva svrstavanje u grupe ili kase na osnovu određene karakteristike, odnosno kriterijuma
- sortiranje - podrazumeva uređivanje redosleda podataka na osnovu određenog kriterijuma
- odabir (selekcija) - podrazumeva izdvajanje podataka na osnovu postavljenog kriterijuma
- agregacija - podrazumeva spajanje srodnih podataka ili spajanje podataka po zadatom kriterijumu
- računanje - podrazumeva manipulaciju podataka nad numeričkim podacima
- kodiranje - transformisanje podataka prema zadatom kriterijumu

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

ZNANJE

Znanje predstavlja informaciju koja ima neki kontekst, koja je relevantna i na osnovu čijeg sadržaja se mogu pokrenuti određene aktivnosti.

Znanje predstavlja informaciju koja ima neki kontekst, koja je relevantna i na osnovu čijeg sadržaja se mogu pokrenuti određene aktivnosti. Shodno tome, znanje je dinamično i ustvari predstavlja informaciju u akciji. Tokom vremena, sa iskustvom, znanje evoluira.

Za pojedince i organizacije je veoma važno da unapređuju svoje znanje, kako bi bile konkurentne. U oblasti IT-a ovo je naročito važno, kako bi kompanije mogle da održe prednost i kako bi unapredile sistem za upravlje znanjem i intelektualnim kapitalom.

Znanje je Intelektualni kapital i ima finansijsku vrednost.

Znanje može biti objektivno i subjektivno.

Objektivno znanje je dokumentovano u formi koja se može distribuirati drugima ili transformisati u proces, dok je subjektivno znanje teško dokumentovati, pošto je nestruktuirano, rasuto i ugrađeno znanje.

▼ Poglavlje 2

Ulazni uređaji

OPŠTI POJMOVI I KLASIFIKACIJA

Ulazni uređaji omogućavaju unošenje podataka od strane izvora informacija i njihov transport u operativnu memoriju računara

Tokom rada računarskog sistema javlja se potreba da se određene informacije (podaci, signali, datoteke i sl.) unesu u operativnu memoriju računara. Izvor ovih podataka može biti:

- čovek
- proces
- spoljne memorije
- drugi računarski sistemi.

Ulazni uređaji omogućavaju unošenje podataka od strane izvora informacija i njihov transport u operativnu memoriju računara. Da bi se procesor računara osloboodio obaveze da čeka na unos podataka, obično se ugrađuje ulazno/izlazni procesor koji, između ostalog, prihvata podatke od ulaznih uređaja i šalje ih na raspolažanje glavnom procesoru koji ih smešta u operativnu memoriju radi dalje obrade.

Neki ulazni uređaji istovremeno mogu da budu i izlazni. Ipak, uvek je moguće odvojiti sklop koji vrši funkciju ulaza od sklopa koji vrši funkciju izlaza. Na primer, alfa-numerički terminal se obično smatra jednim uređajem koji ima ulazno/izlaznu funkciju, ali se on može podeliti na ulazni sklop-tastatura i izlazni sklop-displej.

Ulazni uređaji se prema **klasi podataka** koje primaju mogu podeliti na:

- tastature
- grafički ulazne uređaje
- video ulazne uređaje
- audio ulazne uređaje
- senzore – davače.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

2.1 Tastature

OSNOVNI ELEMENTI TASTATURE

Svi tasteri na tasturi mogu se podeliti u nekoliko grupa, razlikuju se: alfa-numerički, numerički, funkcijski i specijalni tasteri

Tastature su najčešće upotrebljavani ulazni uređaji. Osnovni element tastature je taster koji može biti označeno dugme, prekidač ili poluga. Pritiskom na taster generiše se binarni kod (ASCII ili EBCIDC) koji odgovara tom tasteru i koji se kasnije šalje računaru.

Uređeni niz tastera naziva se tastatura. Svi tasteri na tasturi mogu se podeliti u nekoliko grupa, razlikuju se:

- alfa-numerički tasteri
- numerički tasteri
- funkcijski tasteri
- specijalni tasteri

Alfa-numerički tastaturi omogućavaju unošenje malih i velikih slova, brojeva i znakova. Numerički tasteri koji se nalaze sa desne strane PC tastature se koriste za unos brojeva kada je aktiviran taster NumLock, ili za pomeranje kursora kada je NumLock deaktiviran. Funkcijski tasteri su najčešće označeni sa F1 do F12, a njihova funkcionalnost zavisi od programa u kojem se radi.

Pored tastera koji služe za unošenje karaktera tastatura ima i specijalne upravljačke tastere kao što su:

- **DELETE** - brisanje karaktera
- **CAPS LOCK** - izbor slova (velika ili mala)
- **TAB** - pomeranje za tabulacioni razmak
- **SHIFT** - alternativni znak
- **SPACE** - blanko karakter
- **RETURN** - prenos informacija i prelazak na novi red.

Na mnogim savremenim tastaturama postoji i taster sa Windows-ovim logom (Slika 1) i koristi se za automatsko otvaranje Windows-ovog menija.

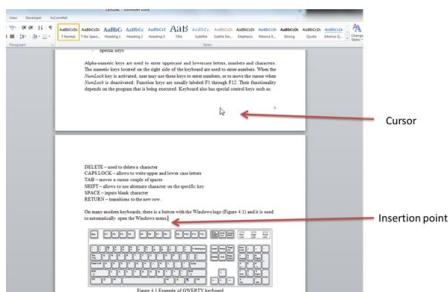


Slika 2.1.1 Raspored tastera na latiničnoj tastaturi

DODATNE FUNKCIONALNOSTI TASTATURE

Dodatne funkcionalnosti tastature podrazumevaju prebacivanja između dva stanja, kontrolisanje medija i priključavanje USB uređaja

Dodatna funkcionalnost, koju pruža tastatura, je prebacivanje između dva stanja. Za to, korisnik može koristiti tastera za prebacivanje (engl. **toggle keys**). Neke tastature imaju funkciju da kontrolisu medije, kao što je Media Player program. Takva tastatura ima posebne tastera za kontrolu jačine zvuka, kao i „**play**“ i „**pause**“ dugmad. „Tačka umetanja“ (engl. **insertion point**) se ponekad naziva i kurzor (engl. **cursor**). Kurzor je u nekim programima predstavljen vertikalnom treptućom crtom (Slika 2) . Ova treperuća crta označava gde će sledeći znak da se ubaci kada ga korisnik otkuca na tastaturi. Tastature se takođe koriste i kao među-mediji za USB uređaje. Neke tastature imaju USB portove, tako da korisnik može priključiti bilo koji USB uređaj na njih. To su uređaju kao što su USB miš, zvučnici, ili flash disk.



Slika 2.1.2 Prikazi „insertion point“ i „cursor“-a

Tastature su obično pravougaonog oblika, a tasteri su smešteni ravno u redovima. Korisnici koji provode dosta vremena koristeći ove tastature često razviju pojavu bola u rukama. Tastature koji su dizajnirane da umanju šansu ove vrste povrede se zovu ergonomski tastature. Ergonomski tastature su dizajnirane na takav način da bi se omogućio odmor zglobova i dlanova (Slika 3) .

Ponekad je potrebno da se tastatura integriše u sistemsku jedinicu. To je slučaj sa laptopovima, gde je tastatura deo sistemske jedinice (šasije). Tipična tastatura na laptopu ima oko 85 tastera. Pošto je važno da se održi ista funkcionalnost koju ima korisnik na desktopu, pojedini tasteri na laptop tastaturi imaju dve ili tri funkcije.



Slika 2.1.3 Ergonomski tastatura

TASTATURE MOBILNIH UREĐAJA

Većina smart telefona i tableta ima QWERTY tastaturu ugrađenu u sistemsku jedinicu

Mobilni uređaji kao što su tableti i pametni mobilni telefoni takođe imaju tastature, ali zbog njihove veličine, ključne funkcije su ograničene. Na primer, na telefonu taster "2" takođe prikazuje znakove a, b, c. Na mnogim telefonima korisnik treba da prođe kroz ciklus svih simbola koji su integrirani na jednom tasteru, dok ne stignu do karaktera koji je tražen. Da bi se kompenzovalo vreme trajanja pisanja, neki telefoni koristite algoritam za intuitivni način unosa teksta, koji pokuša da predviđi šta korisnik pokušava da otkuca na osnovu ukucanog karaktera. Većina smart telefona i tableta ima QWERTY tastaturu, ali kao što je prethodno pomenuto, broj tastera je smanjen u odnosu na desktop tastature, zbog ograničene veličine ekrana.



Slika 2.1.4 Telefoni sa QWERTY tastaturom

▼ 2.2 Pokazivački uređaji

NAČIN FUNKCIONISANJA MIŠA

Pokazivački uređaji služe za interaktivno unošenje podataka

Zajednička karakteristika pokazivačkih uređaja (engl. pointer devices) je da služe za interaktivno unošenje podataka. Obično rade u kombinaciji sa grafičkim korisničkim interfejsom.

Obično jedan pokazivački uređaj omogućuje unošenje različitih vrsta informacija i definiše poziciju na grafičkom terminalu. Najčešće korišćeni pokazivački ulazni uređaj je miš. On može biti mehanički ili optički. Mehanički miš ima dva rotaciona enkodera čije su ose postavljene pod pravim uglom. Kuglica koja se okreće prilikom pokretanja tela miša prenosi kretanje

na ove enkodere tako da je moguće odrediti pomeranje miša u X i Y pravcu (Slika 1) . Optički miš koristi svetlosne senzore da bi osetio pokretanje miša. Neki koriste optički, dok drugi koriste laserske senzore, zbog čega se ovi uređaji često nazivaju laserski miševi. Novije verzije kombinuju optički i laserski senzor, omogućavajući korisnicima da rade na različitim vrstama podloga.



Slika 2.2.1 Konstrukcija mehaničkog miša

Tipičan miš ima više dugmića kojima mogu biti dodeljene različite mogućnosti. Ove funkcionalnosti služe za obavljanje levih i desnih klikova. Za brže pomeranje gore i dole na jednoj strani, postoji dodatna mogućnost za miša, pod nazivom točak (Slika 2) .



Slika 2.2.2 Funkcionalnosti dugmića na optičkom mišu

DIGITALNA OLOVKA

Miš nije pogodan ulazni uređaj u situacijama kada je preciznost važna, na primer kod crtanja ili potpisivanja, pa se u tu svrhu koristi digitalna olovka

Miš kao grafički uređaj nije pogodan za crtanje. Njime je vrlo teško pratiti potez ruke umetnika. U tu svrhu se koristi umetnička ploča. Ona ima dve funkcije. Najpre, ona može vrlo precizno da snimi kretanje pisaljke po svojoj površini. Digitalna olovka (engl. **digital pen**) može biti koristan ulazni uređaj za sisteme kao što su smart telefoni, sistemi koji snimaju digitalni potpis i grafički tablet (Slika 3) .



Slika 2.2.3 Korišćenje digitalne olovke

TRACKBALL

Trackball je pokazivački uređaj sličan mišu i izgleda kao okrenuti miš

Trackball je pokazivački uređaj koji ima ulogu isto kao i miš, ali je njegov način funkcionisanja različit. On se sastoji od loptice, koja se nalazi u ležištu koji sadrži senzore. Ovi senzori detektuju pokrete loptice u dve ose, slično kao i klasičan miš. U suštini, ovaj uređaj možemo gledati kao obrnuti miš. Korisnik rola loptu, a kao odgovor na to vidimo kurzor na ekranu koji se pomera. Prednost trackball-a u odnosu na klasični miš, je što ponekad miša moramo da podignemo i postavimo ga u incijalnu poziciju, pošto odguramo miša do mesta na stolu gde nam više ne odgovara, dok trackball ostaje na svom mestu, a korisnik samo rola prstima loptu.

Iako se ova vrsta miša pojavila nakon drugog svetskog rata i imala različite primene, od video konzola do radarskih konzola, danas se često koriste kao ergonomski uređaji kao pomoćna tehnologija. Trackball se može koristiti dok se koristi laptop u krevetu, ili bežično iz fotelje. On je takođe koristan na brodovima ili drugim nestabilnim platformama, koji otežavaju korišćenje klasičnog miša.



Slika 2.2.4 Trackball

GAMEPAD

Gamepad služi kao ulazni uređaj za igranje igara

Gamepad ili Joypad je ulazni uređaj koji držimo sa dve ruke i prvenstveno služi za igranje igara. Ova vrsta kontrolera za igru (engl. **game controller**) se uobičajeno koristi na konzolama kao što su Nintendo, Playstation, X-box i slično, ali se može koristiti i na personalnim računarima jer je neke vrste igara mnogo lakše igrati sa njim nego npr. sa mišem, tastaturom a ponekad i joystick-om.

Na gamepad-u se najčešće sa desne strane nalaze tasteri za akciju, a sa leve tasteri za kretanje (d-pad). Moderni kontroleri često imaju i dodatne analogne kontrole (engl. **stick**). Postoji više načina spajanja gamepad-a, ali najčešći su preko PS2, USB priključaka i bežično (IRDA, Bluetooth).



Slika 2.2.5 Gamepad kontroler

✓ 2.3 Ulazni uređaji za video i sliku

ULAZNI UREĐAJI ZA SLIKU I VIDEO

Ulazni uređaji za snimanje videa i slike su digitalni fotoaparati, digitalne video kamere, skeneri

Snimanje video zapisa predstavlja zabeležavanje kontinualnog pokreta, odnosno niza slika, koji se čuvaju u računaru. Neki video snimci se snimaju kao analogni signal. Međutim, računari mogu samo da obrade **digitalni signal**. Dakle, da bi se video sačuvao u računaru, neophodno je da se analogni video konvertuje u digitalni. Danas, mnogi uređaji imaju mogućnost da video zapis snime u digitalnom formatu bez potrebe za konvertovanjem i na taj način se lako skladište u memoriju računara. Jedan od uređaja koji snimaju digitalni video je **digitalna video (DV) kamera**. Prebacivanje video zapisa sa DV kamere na računar se obično vrši preko USB kabla. Većina uređaja koji može da snima digitalni video može da snimi i digitalne fotografije. Sadržaj slike može da varira i može biti tehnički crtež, fotografija, tekst, snimci, scene, ili neke kombinacije. Ulazni uređaji za snimanje videa i slike su digitalni fotoaparati, digitalne video kamere, skeneri, itd.

Prilikom snimanja slike, kvalitet slike će zavisiti od rezolucije koju ima ulazni uređaj. **Rezolucija** predstavlja broj vertikalnih i horizontalnih piksela. Uređaj ima mogućnost da snimi sliku sa boljim kvalitetom ukoliko ima veću rezoluciju. Pixel predstavlja najmanji element elektronske slike. Svaki pixel ima svoju poziciju i boju. Svaka boja je definisana jednim brojem.

Jedan bajt može da predstavi maksimalno 256 boja, što nije dovoljno za sliku visoke rezolucije. Kvalitet je definisan brojem piksela po jedinici dužine. Digitalna kamera može imati rezoluciju od 4MP do više od 16MP. Opšte pravilo je da se 4MP slike koriste za veb i email distribuciju. Međutim, da bi mogla da se odštampa slika sa dobrom kvalitetom, potrebno je da ima rezoluciju preko 5MP.

Skener predstavlja ulazni uređaj pomoću koga je moguće učitati tekst ili sliku u formatu koji je prepoznatljiv za računar. Skenere možemo podeliti na:

- Sheetfed skenere
- Flatbed skenere
- Foto skenere
- Portabilni skenere



Slika 2.3.1 Vrste skenera

BAR KOD ČITAČ

Bar kod čitač je uređaj koji čita bar kodox i pretvara ih u električne impulse, koje računar obrađuje

Bar kod čitač je uređaj koji čita bar kodox i pretvara ih u električne impulse, koje računar obrađuje. Bar kod predstavlja kodirane podatke, koji mogu biti linijski ili 2D bar kodovi. Linijski bar kod je u obliku svetlih i tamnih linija. Linearni bar kod obično sadrži neku vrstu tekstualnih informacija. Nasuprot tome, 2D bar kod je složeniji i može uključiti više informacija u kodu: cenu, količinu, veb adresu ili sliku. Skener linijskog barkoda ne može pročitati 2D bar kod. 2D bar kodovi zahtevaju upotrebu skenera za čitanje informacija ugrađenih u 2D barkodu. 2D barkod nazivamo i QR kod



Slika 2.3.2 Linijski bar kod



Slika 2.3.3 QR kod

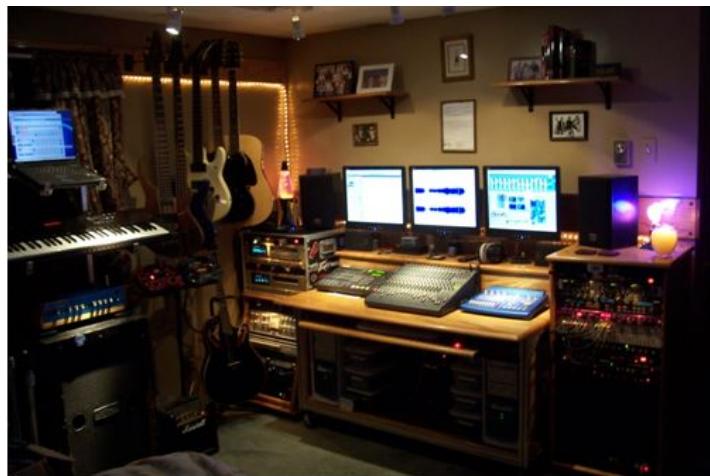
Većina bar kod čitača se sastoji iz tri dela: sistem za osvetljavanje, senzor i dekoder. Uopšteno gledano, bar kod čitač skenira crne i bele elemente bar koda sa crvenim svetlom, koji se nakon toga konvertuje u odgovarajući tekst.. Preciznije, senzor koji se nalazi u barkodnom skeneru detektuje reflektovano svetlo iz sistema za osvetljavanje (crveno svetlo) i generiše analogni signal koji se šalje dekoderu. Dekoder interpretira taj signal i potvrđuje barkodu pomoću kontrolne cifre i konvertuje ga u tekst.

▼ 2.4 Audio ulazni uređaji

ULOGA ULAZNIH UREĐAJA ZA AUDIO

Audio ulazni uređaji omogućavaju da se preko mikrofona unese i memoriše neki zvuk

Audio ulazni uređaji omogućavaju da se preko mikrofona unese i memoriše neki zvuk. Ovaj zvuk može biti govor, muzika, ambijentalni zvuk ili bilo koji drugi zvuk. Slično kao i video, audio mora da se sačuva u računaru kao digitalni signal. Korisnici mogu uneti audio u računar koristeći mikrofon, CD/DVD plejere, radio ili bilo koji drugi uređaj koji se priključuje preko zvučne kartice. Na primer, neki muzički instrumenti mogu da se priključe na računar preko adaptera koji su povezani sa zvučnom karticom, tako da se muzika može da direktno snimiti u računar dok muzičar svira instrument (Slika 1) .



Slika 2.4.1 Muzički studio koji povezuje instrumente direktno sa računarama kao njegovim ulaznim uređajima

Mikrofon je jedan od najčešće korišćenih ulaznih uređaja za snimanje glasa. Većina mikrofona funkcioniše na jednom od sledećih principa:

- Elektromagnetna indukcija (dinamički mikrofon)
- Promena kapacitivnosti (kondenzatorski mikrofon)
- Piezoelectrični mikrofon
- Modulacija svetlosti.

U svim slučajevima se pomeranje membrane mikrofona pretvara u električne signale koji se nakon toga digitalizuju i memorišu. Kvalitet mikrofona zavisi od frekventnog opsega koji može da obuhvati.

Glasovni unos u računar se koristi za više aplikacija kao što je razgovor preko interneta, prepoznavanja glasa (engl. **voice recognition**) i sl. Prepoznavanje glasa je razvijen za potrebe kao što su razlikovanje izgovorenih reči radi njihovog konvertovanja u tekst ili neki tip komande koja treba da se izvrši na računaru.

▼ Poglavlje 3

Izlazni uređaji

OPŠTI POJMOVI I KLASIFIKACIJA

Izlazni uređaji su oni delovi računarskog sistema kojima se šalju izlazni podaci radi izdavanja ili prikaza podataka

Izlazni uređaji su oni delovi računarskog sistema kojima se šalju izlazni podaci radi izdavanja ili prikaza podataka. Izlazne informacije mogu biti u različitim oblicima i njihov oblik zavisi od zahteva koji je upućen kada su te informacije tražene. Korisnik može da pošalje zahtev za fotografijom koja se nalazi na računaru, tako da se ona prikaže na monitoru računara. Međutim, ne mora uvek korisnik da traži informaciju koja je potrebna da se prikaže na monitoru, to isto mogu činiti i programi i uređaji. Zavisno od forme koju imaju, mogu se razlikovati dve vrste izlaznih podataka:

- mašinski čitljivi
- ljudski čitljivi.

Mašinski čitljivi izlazni podaci se izdaju spoljnim memorijama, drugim računarskim sistemima i procesima. Ljudski čitljivi izlazni podaci su namenjeni korisniku i u takvom su obliku da ih ljudi mogu direktno prepoznati. Ovde će se govoriti o izlaznim uređajima za ljudski čitljive izlazne podatke. Izlazni uređaji koji pružaju ljudski čitljivu formu izlaznih podataka se mogu podeliti na izlazne uređaje za:

- izdavanje podataka
- prikaz podataka
- objavljivanje podataka.

Izlazni podaci mogu biti u obliku teksta, slike, videa ili drugoj traženoj formi.

PIKSEL, REZOLUCIJA, BRZINA OSVEŽAVANJA

Osnovna terminologija: piksel, rezolucija i brzina osvežavanja

Piksel (engl. *pixel*) predstavlja tačku, koja je deo jedne slike. Pikseli su organizovani u redove i kolone. Slika se može sastojati od 480 000 do 2 304 000 (16:10, Full HD) i više piksela.

Rezolucija predstavlja broj piksela na monitoru. Rezolucija predstavlja broj piksela u horizontalnim i vertikalnim linijama monitora. Veća rezolucija podrazumeva veći broj piksela. Što je rezolucija veća, pikseli su bliže. Tako na primer, $1920 \times 1200 = 2\ 304\ 000$ piksela.

Brzina osvežavanja (engl. **refresh rate**) predstavlja brzinu osvežavanja slike na monitoru, odnosno brzinu ponovnog očrtavanja slike na monitoru u sekundi. Brzina osvežavanja se meri u hercima, pa je tako na primer, 75 iscrt/ sec ustvari 75Hz. Što je veća frekvencija, to je slika stabilnija, a manja ona je nestabilnija i može biti napornija za ljudsko oko.

▼ Poglavlje 4

Štampači

ŠTA JE ŠTAMPAČ

Štampač je izlazni uređaj koji izdaje tražene informacije u formi kao što su tekst ili slika a fizičkom medijumu koji je obično papir.

Štampač je izlazni uređaj koji izdaje tražene informacije u formi kao što su tekst ili slika na fizičkom medijumu koji je obično papir. Štampači su zasnovani na kvalitetu štampe koja je potrebna, kao i kvantitetu. Na primer, kućni štampači se ne koriste često i oni mogu da štampaju do stotinu strana nedeljno. Malo preduzeće može da odštampa oko stotinak stranica dnevno, dok je u jednoj velikoj kompaniji potrebno štampati na hiljade stranica po satu. Danas, većina štampača su jedan od sledećih tipova: ink-jet štampači, laserski štampači, termički štampači, štampači etiketa, poštanski štampači, ploteri, štampači velikog formata i dr.



Slika 4.1 Primeri štampača

MATRIČNI ŠTAMPAČI

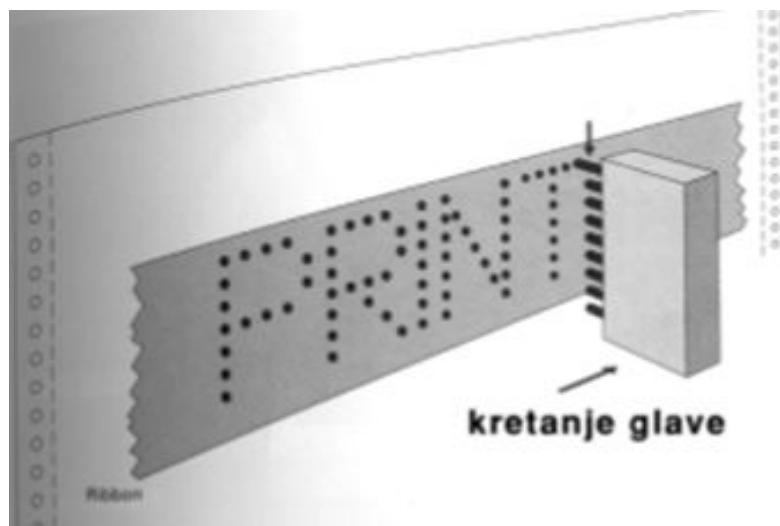
Matrični štampači štampaju pomoću niza iglica, koje se aktiviraju za određene znakove, a njihov udar na traku ostavlja otisak na papiru

Matrični štampači (engl. **dot matrix printers**) su bili nekada najrasprostranjeniji štampači udarnog, a danas se sve manje koriste. Znak koji se štampa formira se od niza iglica, prečnika 0.2 mm, postavljenih u obliku pravougaone matrice. Broj iglica u matrici se kreće od 5x7 do 48x35. Svaka iglica, kao pogon, ima svoj elektromagnet. Kada se štampa određeni

znak, aktiviraju se samo određeni elektromagneti, koji odgovaraju datom znaku. Ovo izaziva pomeranje odabralih iglica i njihovim udarom na traku ostavlja se otisak na papiru. Na slici 2 je prikazan raspored aktiviranih iglica pri formiranju nekih slova, kao i vrste izvođenja matrice.

Zbog toga što je moguće pojedinačno upravljati svaku iglicu mogu se štampati različita pisma, ali i slike. Većina matričnih štampača ima takozvani grafički režim rada, koji omogućuje da se slika od štampa u rasterskoj formi. Kvalitet slike zavisi od toga koliko je velika gustina tačaka po jedinici površine. Kvalitetni štampači omogućuju rezoluciju do 360x360 tačaka po inču kvadratnom.

Matrični štampači se obično izrađuju tako da prime beskonačni papir sa rubnom perforacijom veličine A4 formata. Profesionalni printeri primaju i papir širine do 370 mm. Povlačenje papira vrši specijalni mehanizam (traktor), mada se kod nekih modela ovo postiže frikcijom, kada je moguće umetati i pojedinačne listove bez perforacije. Papir sa rubnom perforacijom se proizvodi i sa do tri samokopirajuća sloja papira, tako da se jednim štampanjem dobija više kopija.



Slika 4.2 Formiranje otiska na matričnim štampačima

TERMIČKI ŠTAMPAČI

Termički štampači rade na principu nagorevanja onih delova papira koji odgovaraju tamnim mestima izlaznog dokumenta

Termički štampači rade na principu nagorevanja onih delova papira koji odgovaraju tamnim mestima izlaznog dokumenta. Zbog toga što zahtevaju specijalni termo papir i što je trajnost dokumenta ograničena gotovo su istisnuti iz upotrebe. Ova rešenja se još susreću kod faksa i i kasa za štampanje fiskalnih računa.



Slika 4.3 Termički štampač

. My POS Printer, <http://www.myposprinter.com/images/SRP-350G.gif> (09.09.2015)

INK-JET ŠTAMPAČI

Ink-jet štampači koriste mlaznice koje pod dejstvom struje brzo otvore i skupe istiskujući pri tom kap boje

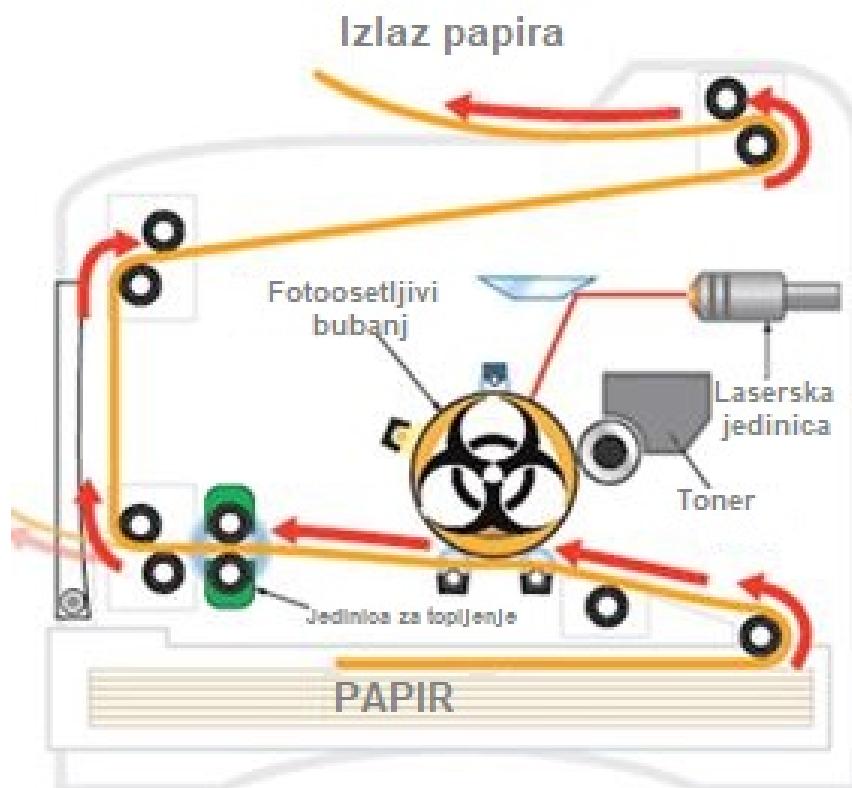
Štampači sa mlazom mastila ili brizgajući štampači (engl. **ink-jet printers**) rade na principu brizganja sitnih kapljica boja na papir. U glavi za štampanje se nalaze kertridži (engl. **cartridge**) sa crvenom, zelenom, plavom i crnom bojom. Svaka boja ima svoju mlaznicu. Mlaznice mogu da rade na termičkom ili piezoelektričnom principu. Mlaznice se pod dejstvom struje brzo otvore i skupe istiskujući pri tom kap boje. Mešanjem boja moguće je dobiti kvalitetnu kolor sliku. Nedostatak mlaznica je što se mogu zapušti i onemogućiti precizno štampanje. Ono što olakšava korišćenja ovog štampača je da kada se potroši mastilo, kertridž koji sadrži tu boju se lako može zameniti. Broj strana koji se može odštampati jednim kertridžom zavisi od proizvođača štampača. Crni kertridž može da odštampa od 200 do 800 strana, a kertridž u boji od 100 do 450 strana.

LASERSKI ŠTAMPAČI

Laserski štampači za štampanje se koristi laserski snop i crni prah koji se zove toner

Laserski štampači su mnogo brži i kvalitetnije štampaju od drugih štampača. Laserski štampači obično mogu da odštampaju 15-60 strana u minuti kada štampaju u crno-beloj varijanti, dok laserski štampači u boji mogu da odštampaju oko 40 strana u minuti. Profesionalni laserski štampači mogu da štampaju i više od 150 strana u minuti. Laserski štampači skladište čitavu stranu u memoriji pre štampanja. Brzina štampanja zavisi od veličine skladišta. Poslovni laserski štampači imaju memoriju i do 1GB i hard disk od 80GB.

Laserki štampač funkcioniše po sličnom principu kao i mašina za kopiranje. Za štampanje se koristi laserski snop i crni prah koji se zove toner. Osnovni deo laserskog štampača je valjak presvučen materijalom koji omogućava zadržavanje elektrostatičkog naboja. Laserski snop skenira duž celog valjka i osvetljava ona mesta na kojima treba da se ostavi otisak. Tačke na kojima treba ostaviti otisak su negativno nanelektrisane. Laserski snop je usmeren prema centru valjka. Toner laserskog štampača predstavlja veoma fini crni prah koji je pozitivno nanelektrisan, tako da prah privlače negativno nanelektrisane tačke na površini valjka. Papir zatim prolazi kroz sistem za sušenje koji zagreva na temperaturi od 200 stepeni Celzijusa i trajno ostavlja toner na papiru.



Slika 4.4 Laserski štampač

PLOTERI

Ploteri koriste inkjet tehnologiju za štampanje velikih formata

Ploteri su sofisticirani štampači koji se koriste u specijalizovanim oblastima. Ploteri su veoma skupi. Trenutni ploteri koriste "styli." "Styli" predstavlja red nanelektrisanih žica koji iscrtavaju elektrostatičke šare na papiru, a zatim toner ostaje na papiru. Drugim rečima, ploteri koriste inkjet tehnologiju, ali u mnogo većem obimu. Ploteri i štampači velikog formata mogu obično da štampanju na formatima veličine do 98 inča širine. Oni se koriste za nacrte, mape, postere, itd.

LASERSKI, INK-JET, TERMIČKI I MATRIČNI ŠTAMPAČI - VIDEO

Laserski, ink-jet, termički i matrični štampači

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

3D ŠTAMPAČI

Za razliku od subtraktivnih mašina (strugovi, bušilice, testere), koje od velikog komada materijala oduzimaju višak, 3D štampači su aditivni: oni polaze „ni od čega“ i proizvod stvaraju

Iako ne rade svi **3D štampači** na isti način, jedno im je zajedničko: za razliku od subtraktivnih mašina (strugovi, bušilice, testere), koje od velikog komada materijala oduzimaju višak, 3D štampači su aditivni: oni polaze „ni od čega“ i proizvod stvaraju tako što dodaju materijal tačku po tačku, ili barem liniju po liniju. Taj materijal može da bude rastopljena plastična masa, koja se pomoću ekstrudera selektivno polaže po X osi. Kad se jedna linija završi, glava dispenzera se vraća, pomera se za jedan voksel po Y osi, pa se postupak ponavlja u krug. Nakon završetka jednog horizontalnog sloja, započeti objekat se spušta za jedan voksel i sve se ponavlja sa narednom Z koordinatom.

Objekat se, dakle, gradi od slojeva rastopljene plastične mase, koja se odmah hlađi i očvščava. Obično se koristi ABS ili biorazgradiva PLA plastika u obliku niti koja se mehanički dovodi do mesta topljenja u pokretnom ekstruderu 3D štampača. Zajedno s materijalom za gradnju objekta dovodi se i materijal za privremenu popunu praznih delova objekta, koji se po završetku posla odstranjuje hemijski, potapanjem u hlorovodoniku kiselinu. Postoje i džinovski 3D štampači koji grade kuće od specijalnog betona.

Naziv ove tehnologije je FDM (**Fused Deposition Modelling**, modeliranje fuzionim taloženjem). To je najjednostavnija kategorija 3D štampača, a cena im se kreće od nekoliko stotina do nekoliko hiljada evra. Skuplji modeli su oklopljeni komorom u kojoj se održava konstantna temperatura, čime se postiže željena brzina hlađenja što neutrališe mehanička naprezanja, pa je proces štampanja precizniji a objekat čvršći. Veličina jednog voksela je u opsegu od 75 mikrometara, pa do više od jednog milimetra.

Kvalitetniji rezultati se postižu printerima koji rade na principu lepljenja praškastog materijala, ali njihove cene iznose i po nekoliko desetina hiljada evra. I ovde postoji širok izbor materijala, a najčešći je kompozitni materijal koji liči na gips i koji se vezuje lepkom raspršenim iz dispenzera, kao kod ink-jet štampača. Ovaj postupak nosi naziv binder jetting (ne postoji dobar prevod, ali je najpričutniji vezivanje prskanjem). Neki modeli imaju glavu sa pet brizgaljki, pa se kroz jednu raspršuje lepak, a kroz preostale četiri boje iz CMYK modela, čime se dobija obojen 3D proizvod, u rezoluciji 300-600 dpi. Čitav proces se odvija u komori praškastog materijala koji se suksesivno doprema dozatorom.

Izvor: <https://pcpress.rs/zlatne-ruke-3d-stampaca/>

BIO PRINTING

Bioprinting ne podrazumeva izgradnju živih ćelija, nego samo njihovo uzimanje s nekog drugog mesta, eventualno razmnožavanje van organizma i pravljenje organa za zamenu.

Mada tehnologija 3D štampanja živog tkiva nije za široko tržište, zanimljivo je znati kakve su mogućnosti i dokle se danas stiglo. Treba odmah reći da bioprinting ne podrazumeva izgradnju živih ćelija, nego samo njihovo uzimanje s nekog drugog mesta, eventualno razmnožavanje van organizma i pravljenje organa za zamenu. Pošto se uzorci ćelija za razmnožavanje uzimaju sa iste osobe koja će biti primalac, velike su šanse da će organizam prihvati takav organ. Ćelije našeg organizma su približno iste veličine kao kapljice ink-jet štampača, pa se u bioprintingu koriste obične glave komercijalnih ink-jet-ova.

Kod štampanja živih organa, dovoljno je grubo postaviti ćeliju na željeno mesto, a ostalo će učiniti priroda i ćelija će sama naći svoje preciznu poziciju i „upoznati se“ sa okolnim ćelijama. Kod pravljenja krvnih sudova, recimo, meke mišićne ćelije se same kreću ka centru zida krvnog suda, fibroblasti zauzimaju poziciju na spoljnoj strani, a endotelijalne ćelije već „znaju“ da im je mesto na unutrašnjoj. Slično vezivnom materijalu u opisanom FDM postupku, ovde se koristi kolagen ili neki sličan materijal, a posle „rasta“ organa, tkivo ga samo eliminiše ili se ukloni na neki drugi način.

Na nekoliko mesta u svetu se radi na ovakvom „štampaju“ organa i već postoje značajni uspesi u eksperimentima na životinjama, a nedavno je i prvi čovek dobio svoj odštampan krvni sud. Očekuje se da prvi odštampani organ bude bubreg, jer je najjednostavniji. Smatra se da će najveći napredak biti ostvaren kad bude odštampano i uspešno zamenjeno ljudsko srce.

Štampanje ljudskih organa za zamenu nije jednini cilj bioprintinga. Velika su očekivanja i od pravljenja živih ljudskih tkiva za eksperimentisanje s novim lekovima, jer taj način ne dolazi u sukob sa etičkim i pravnim principima, a nudi mnogo bolji način ispitivanja nego na dobrovoljcima ili životinjama.

Izvor: <https://pcpress.rs/zlatne-ruke-3d-stampaca/>

PROBLEMI I NEDOSTACI

Jedan nedostatak je taj da je posle svakog štampanja potrebno ručno doraditi 3D objekat, jer su ivice neravne i uvek ima sitnih grešaka

3D printeri nisu nova tema, ali je u razvoju digitalne tehnologije tek sada nastao pravi trenutak za ubrzani razvoj i širu primenu. Postoji, ipak, nekoliko nerešenih problema koji će usporiti širenje ove tehnologije i odložiti trenutak u kome će hiperprodukcija oboriti cene dovoljno da svaki pojedinac može da ima 3D printer u svom domu. Prvi nerešen problem je brzina rada. Vreme potrebno za štampanje malog objekta meri se satima, a objekta prosečnih dimenzija (dvadesetak centimetara) čak i desetinama sati, pa i dana. To svakako umanjuje početno oduševljenje koje nas je navelo na to da pomislimo kako ćemo odmah moći da

odštampamo sve što nam zatreba. Drugi problem je neophodnost upotrebe kvalitetnih fajlova u kojima su definisani 3D objekti. Svakako će ih vremenom biti sve više na Internetu, ali je pitanje koliko će biti skupi oni koji su kvalitetni, i šta ćemo moći da dobijemo od besplatnih. Naravno da će biti i programa koji će nam omogućiti da pravimo i svoje objekte, ali isto važi i za muziku, likovna i video dela, pa opet nemamo mnogo ljudi spremnih da uposle svoju kreativnost. Postojeće CNC mašine nude mogućnosti kao 3D printeri i mnogi su pohitali da ih kupe, a onda su se suočili s nedostatkom ideja, pa većina novih vlasnika nije odmakla dalje od ideje da prave i prodaju crkvene ikone.

Još jedan nedostatak je taj da je posle svakog štampanja potrebno ručno doraditi 3D objekat, jer su ivice neravne i uvek ima sitnih grešaka, uzrokovanih prirodnim čudima sintetičkih materijala. Pojava koju smo pod nazivom aliasing već upoznali kod matričnih monitora (a danas su praktično svi matrični) i koja se pokazuje kao „stopenice“ na kosim linijama, ovde je posebno primetna, jer se pod prstima oseća čak i najmanja neravnina i nesavršenost. Kod video-jedinica ona je rešena (ili barem ublažena) anti-aliasing postupkom, u kome se neki kritični pikseli dozirano „zamućuju“ kombinacijom boja dva susedna piksela, ali kod 3D objekata nema zamućenja – voksel ili postoji ili ne postoji. Prvi opisani postupak (FDM), kojim se tope ABS ili PLA niti, za rezultat ima vidljive „stopenice“ čak i na horizontalnim i vertikalnim površinama, a ne samo na kosim. To nam pomaže da lako odredimo da je 3D objekat štampan FDM tehnologijom, jer će površina „zapevati“ kad prevučemo noktom preko nje. Protiv ovoga je jedini lek ručna obrada šmirglom i eventualno turpijom. Drugi postupak, sa praškastim materijalom (**binder jetting**) daje ravnije površine, ali je i ovde potrebna završna obrada. U tu svrhu se obično koriste komore za peskiranje, koje rešavaju problem neravnina. Neki od savršenijih 3D štampača stvaraju objekte sa veoma finim površinama, ali su njihove cene takve da ćemo verovatno još dugo čekati na njihovu masovnu primenu. To ne važi samo za uređaje nego i za repromaterijal, čije učešće u troškovima nije nimalo zanemarljivo. Ipak, hiperprodukcija nije još uvek počela i treba je očekivati uskoro, a kakav je uticaj hiperprodukcije na cenu proizvoda videli smo u poslednjih nekoliko decenija, tokom prodora kompjutera i prateće opreme na široko tržište.

▼ Poglavlje 5

Displeji

IZLAZNI UREĐAJI ZA PRIKAZ PODATAKA

Displeji su izlazni uređaji koji vizuelno prikazuju tražene informacije u obliku teksta, grafike i videa

Izlazni uređaji za prikaz podataka omogućavaju privremeno prikazivanje izlaznih podataka na nekoj vrsti displeja (engl. **display**). Displeji su izlazni uređaji koji vizuelno prikazuju tražene informacije u obliku teksta, grafike i videa. **Displeji** se sastoje od ekrana i komponenti koje omogućavaju da se informacije prikažu na ekranu. Tradicionalni desktop računari imaju monitore koji se koriste kao odvojene periferne jedinice, dok laptop računari i mobilni uređaji imaju ekran integrisan unutar sistemske jedinice. Danas je većina ekranova u boji. Međutim neki ekranovi su još uvek monohromatski. Monohromatski ekran znači da se podaci prikazuju samo u jednoj boji. Na primer, starija generacija mobilnih telefona koristi samo monohromatske ekranove, kako bi sačuvali bateriju i dužinu njenog rada.

Monitori se izrađuju uglavnom koristeći dve tehnologije:

- na bazi katodne cevi (engl. **cathode ray tube(CRT)**)
- na bazi tečnih kristala (engl. **liquid crystal display(LCD)**)

CRT MONITORI

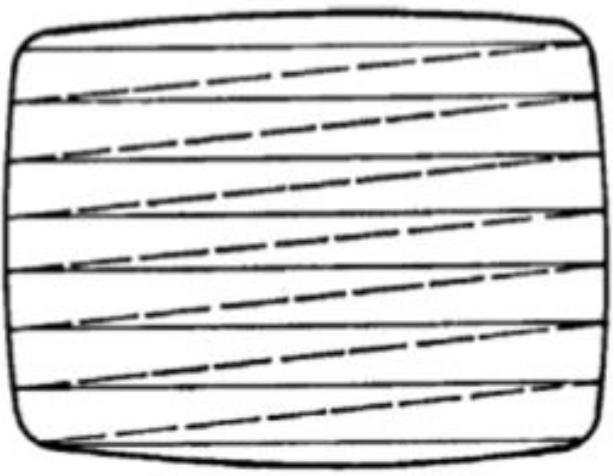
Kada elektronski zrak pogodi zrno fosfora, koje se nalazi u unutrašnjosti katodne cevi, ono zasvetli

Kod uređaja za prikaz podataka baziranih na **katodnoj cevi** koristi se tehnologija rasterskog skeniranja koja je gotovo istovetna tehnologiji televizora. Unutrašnjost šire strane katodne cevi je obložena zrncima fosfora. Kada elektronski zrak pogodi zrno fosfora, ono zasvetli. Upravljanjem energije elektronskog zraka može se regulisati intenzitet osvetljenja neke tačke. Pomoću otklonskih jedinica moguće je upravljati pomeranje elektronskog zraka po ekranu tako da se, kontrolom intenziteta osvetljenja, na ekranu dobija slika.

Međutim, tokom vremena intenzitet te svetlosti opada. Pod istrajnošću fosfora se definiše vreme koje protekne od osvetljavanja nekog zrna pa do trenutka kada intenzitet svetlosti opadne na vrednost 10% od početnog intenziteta. Istrajnost različitih vrsta fosfora se kreće u intervalu od 10 ms do 1h. Ako se želi brza promena oblika slike na ekranu istrajnost fosfora ne bi trebala da bude veća od 100 ms. Da bi se slika održala na ekranu postupak osvetljavanja tačaka se mora ponoviti. Ovaj proces se naziva osvežavanje ekrana. Kako čovekovo oko može da primi 15 slika u sekundi, proces osvetljavanja treba da se vrši više od 20 puta u sekundi.

Na taj način je izbegnuta pojava treperenja slike. U praksi se ovo osvežavanje vrši 60 do 100 puta u sekundi.

Rasterski displeji (engl. *raster-scan display*) sa osvežavanjem imaju fiksnu putanju kretanja elektronskog zraka. Kod rasterskih displeja vektorski zrak skenira ceo ekran liniju po liniju, kao što je to prikazano na slici. To znači da je pri svakom osvežavanju ekrana dužina putanje zraka, pa time i vreme skeniranja, ista.



Slika 5.1 Rastersko skeniranje

NAČIN FUNKCIJONISANJA CRT MONITORA

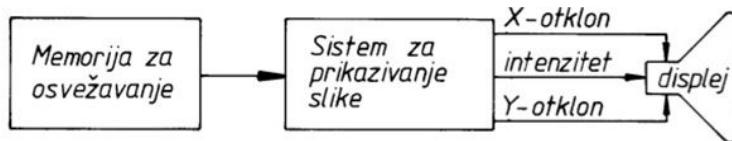
Podaci o sliци se čuvaju u memoriji za osvežavanje, sistem za prikazivanje slike određuje adresu piksela, zrak dalje prolazi kroz vertikalnu i horizont

Rasterski displej je radi skeniranja podeljen na linije. Broj linija zavisi od veličine displeja i kreće se od 600 linija pa naviše. Svaka linija je podeljena na tačke. Ove tačke predstavljaju osnovni element slike koji se naziva piksel (pixel je skraćenica od picture element). Broj piksela u liniji takođe zavisi od veličine displeja i kreće se od 800 piksela pa naviše. Tako se, na primer, na displeju čija je dijagonala 15 inča može obrazovati matrica piksela, odnosno raster od 1024x1280 tačaka.

Podaci o slići se čuvaju u **memoriji za osvežavanje** (engl. *refresh memory*) koja je organizovana na sličan način, tj. kao dvodimenzionalno polje. Svaki element memorijskog polja (dva ili više bajta) odgovara jednom pikselu na ekranu. U njima su memorisani podaci o osvetljenju slike i/ili podaci o boji. Podatke o slići ob razaje računar i uvek kada dođe do neke promene na slići me morija za osvežavanje se ažurira.

Sistem za prikazivanje slike najpre odredi adresu piksela koji će biti procesiran. Ova adresa se koristi kako za nalaženje odgovarajućih podataka u memoriji za osvežavanje tako i za upravljanje otklonskom jedinicom. Obično se počinje od memorijskog polja koje odgovara pikselu u gornjem levom uglu. Ovi podaci se onda iskoriste za osvetljavanje te tačke. Zatim se čitaju podaci za sledeći piksel udesno i tako redom do kraja linije. Svetlosni zrak se onda

spusti na narednu nižu liniju i obrađuje je. Ovaj postupak se ponavlja za sve linije do dna ekrana, a zatim počinje novi proces skeniranja. Izmena podataka u memoriji za osvežavanje se vrši između dva ciklusa skeniranja. Kako je proces skeniranja vrlo čest, brzina promene slike zavisi samo od brzine rada računara koji daje podatke o slici. Z bog relativno niske cene i dobrih karakteristika rasterski displeji su najčešće upotrebljavani uređaji za prikaz podataka.

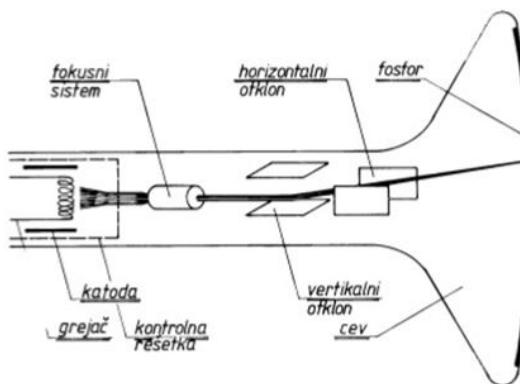


Slika 5.2 Način funkcionisanja CRT monitora

KAKO SNOP SVETLA OSVETLJAVA FOSFORNE ĆELIJE?

Snop elektrona prolazi kroz fokusni sistem, fokusira se i pogoda samo jednu tačku na široj strani cevi

Izlazni uređaji za prikaz podataka sa osvežavanjem koriste katodnu cev (CRT) za prikaz podataka. Katodna cev za crno-belu tehniku ima oblik prikazan na slici 3. Na užem kraju katodne cevi grejač zagreva katodu, koja onda emituje elektrone. Metalna kontrolna rešetka obavlja katodu sa svih strana usmeravajući snop elektrona ka široj strani katodne cevi. Snop elektrona prolazi kroz fokusni sistem, fokusira se i pogoda samo jednu tačku na široj strani cevi. Ovaj zrak dalje prolazi kroz vertikalnu i horizontalnu otklonsku jedinicu čiji su zadaci da upravljuju kretanjem snopa po unapred zadatoj rasterskoj putanji.



Slika 5.3 Katodna cev za crno-belu tehniku

Kolor displeji sa osvežavanjem najčešće koriste tehniku svetlosne maske (**shadow-mask**), koja se upotrebljava i kod kolor televizora. Ovakvi displeji imaju po unutrašnjoj površini ekrana raspoređene trijade fosfornih tačaka. Pojedina zrna u trijadi emituju crvenu, zelenu ili plavu svetlost. Mešavinom ovih triju boja moguće je dobiti bilo koju boju. Ove fosforne tačke su zbog toga postavljene toliko blizu da se stapaju u jednu tačku kada se gledaju sa dovoljne udaljenosti. Podešavanje boje se postiže različitim intenzitetima zraka koji pogoda pojedine tačke. Zbog toga ovakvi displeji imaju tri posebna elektronska topa: za crvenu, zelenu i plavu boju. Zbog širine zraka se može dogoditi da jedan zrak pobudi dva ili sva tri zrna fosfora, čime

bi se dobila neželjena boja. Da bi se to sprečilo neposredno ispred površine sa slojem fosfora je postavljena svetlosna maska po kojoj su izbušeni vrlo precizni otvori. Svaki otvor odgovara jednoj trijadi tačaka. Pošto su zraci iz topa upućeni pod različitim uglom. Ako se pretpostavi da svaki zrak može da ima 256 nivoa intenziteta, to znači da za svaki piksel treba obezbediti tri bajta ili 24 bita u memoriji za osvežavanje. Ovakav sistem bi omogućio prikazivanje 2 na 24 = 16.777.216 različitih boja. Kada je tokom ciklusa skeniranja potrebno osvetliti neki piksel, onda se iz memorije za osvežavanje pročita broj boje, a zatim se za taj broj iz tabele boja pročitaju intenziteti pojedinih elektronskih zrakova.

NEDOSTATAK KATODNIH CEVI

Velika potrošnja struke, elektromagnetna radijacija, velike dimenzije

Displeji bazirani na katodnim cevima imaju niz nedostataka kao što su:

- velika potrošnja struje
- velike dimenzije po dubini i velika težina
- nejednaka oštrina i boje na različitim delovima ekrana, a pogotovo u uglovima
- jako magnetno polje izaziva neželjenu elektromagnetsku radijaciju.

CRT MONITORI - VIDEO

CRT monitori - kako rade?

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

LCD MONITORI

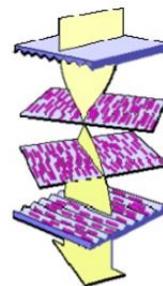
Svetlost koja prolazi kroz tečni kristal prati orijentaciju molekula od kojih je kristal sastavljen

Zbog svega ovoga bivaju potiskivani sa tržišta **displejima baziranim na tečnim kristalima**. Tečni kristali su gotovo potpuno transparentne materije koje imaju svojstva i čvrstih i tečnih materija. Svetlost koja prolazi kroz tečni kristal prati orijentaciju molekula od kojih je kristal sastavljen. Godine 1960. je otkriveno da prilikom propuštanja elektriciteta kroz tečni kristal dolazi do promene orijentacije molekula, a time i do promene putanje kojom svetlost prolazi kroz njih.

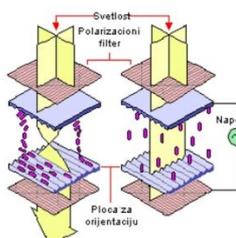
Većina tečnih kristala ima šipkasti oblik molekula i u prirodnom stanju oni su paralelni. Moguće je precizno orijentisati položaj molekula ako se tečni kristal posipa po površini u kojoj su izrezani vrlo precizni kanali. Orientacija molekula onda prati kanale, pa ako su kanali paralelni onda su i molekuli paralelno orijentisani. Displeji sa tečnim kristalom imaju dve ploče između kojih se nalaze tečni kristali. Kanali na jednoj ploči su izrezani pod uglom od 90 stepeni u odnosu na kanale na drugoj ploči. Ako su molekuli na jednoj ploči orijentisani,

na primer, u pravcu sever – jug, onda su molekuli na drugoj ploči orijentisani u pravcu istok – zapad. Molekuli između ovih ploča su prisiljeni da se postupno uviju (engleski *twist*) za 90 stepeni. Pošto svetlost prati molekule, onda se i ona uvija za 90 stepeni prilikom prolaska kroz tečni kristal (slika 4). Oni mogu da pobude samo odgovarajuće zrno fosfora.

Pored ploča za orijentaciju molekula, displeji sa tečnim kristalom imaju i dva polarizaciona filtera. Filter je ploča na kojoj su narezane tanke paralelne linije. Ovaj filter blokira prolazak svih svetlosnih talasa osim onih koji su orijentisani paralelno sa linijama. Jedan polarizacioni filter ima linije koje su pod uglom od 90 stepeni u odnosu na drugi, tako da je prolazak svetlosti totalno blokiran. Svetlost će proći samo ako se i sama uvije za 90 stepeni prolazeći put od jednog do drugog polarizacionog filtera.



Slika 5.4 Uvijanje svetlosti



Slika 5.5 Prolazak svetlosti kroz kristal bez napona (levo) i pod naponom (desno)

KARAKTERISTIKE LCD MONITORA

Karaktersitike LCD monitora su prirodna rezolucija, brzina odziva, ugao gledanja, osvetljenje i kontrast, širina i visina slike

- **Rezolucija** - LCD monitori su napravljeni da najbolje rade na rezoluciji koja prevashodno zavisi od veličine ekrana. Moguće je postaviti rezoluciju koja nije prirodna za neki LCD monitor, ali onda opada kvaliteta slike i gubi se pravilan geometrijski oblik slike)
- **Brzina odziva** (engl. *response rate*) - meri se u milisekundama i predstavlja brzinu kojom piksel menja boju. Veća brzina je bolja i znači da slika neće kasniti i da neće biti pojave anomalija kao što je "*ghosting*", koja se javlja kada je promena boja česta.
- **Ugao gledanja** - za razliku od CRT monitora, LCD monitori nemaju isti kvalitet slike ako se u monitor gleda iz različitih uglova. U slučaju da se slika gleda sa strane, slika gubi na kvalitetu, često je zamračena i nečitljiva.
- **Osvetljenje i kontrast** - predstavlja jačinu svetla koju monitor može da proizvede. Osvetljenje se izražava u kandelima po metru kvadratnom - cd/m^2 , i obično ima vrednost

300cd/m² ili više. Kontrast može biti statički i dinamički. Kontrast predstavlja mogućnost LCD monitora da prikaže bele i tamne tonove. Obično današnji monitori imaju odnos 450:1 pa do 6000:1.

- **Širina i visina slike** - standardne proporcije širine i visine slike na monitorima su 4:3, ali neki novi monitori imaju širi format: 16:9 ili 16:10 i dizajnirani su za gledanje filmova ili HDTV-a u širokom formatu, i nazivaju se **widescreen**.

TEHNOLOGIJE LCD EKRANA

Pasivna matrična i aktivna matrična tehnologije su dve osnovne tehnologije u proizvodnji displeja na bazi tečnih kristala

U praksi se koriste dve tehnologije proizvodnje displeja na bazi tečnih kristala:

- Pasivna matrična tehnologija (engl. **passive-matrix technology**)
- Aktivna matrična tehnologija (engl. **active-matrix technology**) poznata i kao "**thin film transistor**" (TFT) tehnologija.

TFT LCD (Thin film transistor liquid crystal display) displeji koriste odvoje tranzistore da primene nanelektrisanje na svaki kristal koji omogućava prikazivanje boja visokog kvaliteta iz svih uglova. Novija verzija TFT je organski LED (OLED) koji koristi organske molekule da bi obezbedio bolji kvalitet od standardnog TFT ekrana. OLED ekran nišu toliko skupi, mogu biti tanki i fleksibilni, a pri tom troše manje energije .

Pasivna matrična tehnologija koristi manje tranzistora od aktivne matrične tehnologije, međutim takvi ekran često nemaju intenzitet kao displeji koji koriste aktivnu matričnu tehnologiju.

Rezolucija u LCD monitorima rezolucije se definiše kao broj horizontalnih i vertikalnih piksela ekrana. Monitor sa većom rezolucijom prikazuje veći broj piksela i daje bolji kvalitet slike. S druge strane, rezolucija je obično proporcionalna veličini samog uređaja. Osvetljenje se meri u nitima (jedinica intenziteta vidljive svetlosti je jednaka jednom kandelu po kvadratnom metru). LCD monitori imaju osvetljenje od 250 do 550 nita.

LCD TEHNOLOGIJA

LCD Technology: How You're Seeing this Video

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

EKRANI KOJI RADE NA DODIR

Kada korisnik dodirne ekran, na tom mestu dolazi do kontakta dva sloja.

Ekran na dodir postaju sve više rasprostranjeni, obzirom na stalni pad cene proteklih godina. Postoje tri osnovna sistema koji se koriste da prepoznađaju dodir:

- Otporni (engl. **Resistive**)
- Kapacitivni (engl. **Capacitive**)
- Površinski talasi (engl. **Surface acoustic wave**)
- Near field imaging

Otporni sistemi se sastoje od normalnog panela stakla koji je prekriven sa provodnim i otpornim metalnim slojem. Ova dva sloja su razdvojena separatorima, a preko njih je postavljen sloj otporan na ogrebotine. Električno kolo prolazi kroz ova dva sloja dok je ekran uključen. Kada korisnik dodirne ekran, na tom mestu dolazi do kontakta dva sloja. Beleži se promena električnog polja i koordinate te tačke. Kada računar izračuna koordinate, specijalni drajver prevodi taj dodir u nešto što operativni sistem može da razume, slično kao što drajver računarskog miša prevodi kretanje miša u klik ili pomeranje po monitoru.

OPIS RADA TIPOVA EKRANA NA DODIR

Otporni sistemi se sastoje od normalnog panela stakla koji je prekriven sa provodnim i otpornim metalnim slojem.

Otporni sistem

Otporni sistemi za ekrane na dodir rada slično kao "transparentna tastatura" koja je postavljena preko ekranu. Postoji fleksibilan gornji sloj od provodne poliesterske plastike koji je vezan na kruti donji sloj provodnog stakla i razdvojen izolacionom membranom. Kada se dodirne ekran, gornji sloj dodiruje stakleni sloj i zatvara kolo - kao i pritiskanje dugmeta na tastaturi. Čip unutar ekranu izračunava koordinate mesta dodira.

Kapacitivni

Ovi ekrani su napravljeni od nekoliko slojeva stakla. Unutrašnji sloj provodi struju kao i spoljni sloj, te se ekan ponaša kao dva električna provodnika koje razdvaja izolator - drugim rečima, kondenzator. Kada stavite prst na ekran, dolazi do promene električnog polja za određenu količinu koja može da varira u zavisnosti gde se ruka nalazi. Ekrani koji koriste ovaj sistem se mogu dodirnuti na više mesta od jednom. Za razliku od većine ekrana na dodir, ovi ekrani ne reaguju ako se dodiruju plastičnom iglom (iz razloga što je plastika izolator i sprečava ruku da utiče na električno polje).

Površinski talasi

Ova tehnologija ekrana na dodir detektuje prste koristeći zvuk, a ne svetlost. Ultrazvučni talasi (visoke frekvencije da bi ljudi mogli da ih čuju) se generišu na ivicama ekrana i reflektuju napred-nazad preko površine. Kada se dodirne ekran, dolazi do prekida zvučnih talasa i apsorbuje se deo njihove energije, a kontrolni čip ekrana izračunava mesto gde je ekran bio dodirnut.

Near-field obrada

Da li ste primetili da neki stari radio uređaji počnu da zuje i bruje ako primaknete ruku ka njemu? To se dešava zato što ljudsko telo utiče na elektromagnetsko polje koje radio stvara u i oko antene. Što je telo bliže, dolazi do većeg efekta. **Near field imaging** (NFI) ekran i rade na sličan način. Kako se prst približava ekranu, menja se električno polje na staklu, što trenutno registruje dodir. Iako su robusniji od drugih, ovakvi ekran i su pogodni za gruba okruženje (vojna upotreba). Takođe, za razliku od većine drugih tehnologija podržava dodir olovke, igli ili ruku sa rukavicama.

EKRANI KOJI RADE NA DODIR - VIDEO

How Touchscreen Works

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 6

Pokazna vežba: Programi za obradu rasterskih slika

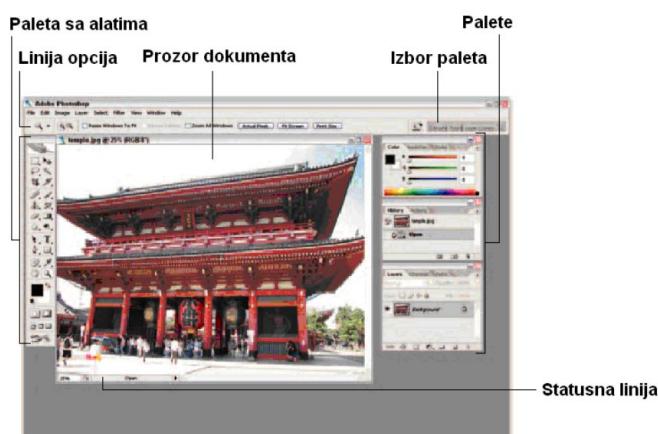
PROGRAMI ZA OBRADU RASTERSKIH SLIK - ADOBE PHOTOSHOP

Osnovne Adobe Photoshop opcije

Predviđeno vreme pokaznih vežbi je 90 minuta.

Adobe Photoshop predstavlja standard u industriji softvera za obradu slika. Ono što ga krasiti jeste jednostavna interfejs i veoma moćne komande i alati.

Photoshop se pokreće kao i svaki drugi program u Windows okruženju. Nakon pokretanja na ekranu će se pojaviti standardno Photoshop radno okruženje. Unutar glavnog Photoshop prozora nalaze se drugi prozori, kao npr. Image document window koji omogućava pregled i izmenu slike. Glavni prozor sadrži i druge standardne elemente koji se mogu videti kod drugih Windows programa: naslovna linija na vrhu prozora i meniji za izvršavanje komandi i dobijanje važnih informacija o slici. Osnovne kontrolne opcije su svrstane u palete (**palettes**).



Slika 6.1 Adobe Photoshop osnovni prozor

Svaka radna površina, pa tako i ona „prava“ – fizička, često umeđu da bude pretrpana. Međutim, Adobe je izradio neke posebne elemente, smeštene na liniji opcija, koji dozvoljavaju organizovanje radnog prostora.

Svaki dokument na kojem se radi pojavljuje se u granicama prozora dokumenta i ne izlazi iz tog okvira. Oko tog prozora mogu se pomerati druge komponente, kao što su različite palete i opcije.

Linija sa menijima omogućava pristup opcijama kojih ima više stotina, koje se češće ili ređe upotrebljavaju, ali u svakom slučaju pružaju mogućnost menjanja slike.

PRIPREMA SLIKE

Prikaz pripreme slike

Da bi se slika pripremila za obradu u radnom prozoru, u meniju File nalazi se obilje različitih opcija za rad sa datotekama, od otvaranja novih ili već snimljenih slika do zatvaranja i snimanja. Osim toga, tu su i standardne opcije za podešavanje strane, pregled slike, njeni štampanje itd. Meni Edit sadrži alate koji omogućavaju sečenje, kopiranje ili umetanje selektovanog dela slike. Selektovani deo se može popuniti ili se njegove ivice mogu izglađiti. Uz pomoć ovog menija slika se može rotirati, izobličiti, osim toga moguće je menjati veličinu ili se mogu izvršiti druge transformacije.

Meni Image omogućava opcije za promenu veličine, boja ili kontrasta i to ne samo za čitavu sliku, već i za pojedine slojeve ili selektovane delove. Na primer Mode opcija dozvoljava promenu slike od obojene do crno-bele. Opcije poput Image Size..., Canvas Size..., Rotate canvas, Crop ili Trim..., svaka na svoj način utiču na čitav dokument. Sa druge strane, promene se mogu izvršiti i iz Adjustments podmenija za celu sliku ukoliko dokument ima samo pozadinu i nema slojeve. Ukoliko dokument ima više od jednog sloja, podešavanja kao što su Color Balance, Hue/Saturation ili Levels, se odnose samo na jedan sloj ili neki njegov selektovani deo. Selektovanje dozvoljava rad sa samo delom slike. Može se izabrati čitav sloj (layer) ili više njih. Meni Select dozvoljava nekoliko komandi za modifikovanje selektovanog dela. Meni Select, iako veoma kratak, omogućava veliki broj opcija za promenu slike.



Slika 6.2 Menu Select

SLOJEVI

Prikaz slojeva u Photoshop-u

Slojevi dozvoljavaju slaganje delova slike jedan na drugi, tako da možete raditi sa pojedinim elementima nezavisno od drugih. Tada, ako ste zadovoljni učinjenim promenama, postoji mogućnost kombinovanja promenjenih delova u jedinstvenu konačnu verziju slike ili se mogu ostaviti u obliku sloja sa maksimalnom mogućnošću daljeg menjenja.

Deo Photoshopa koji se odnosi na rad sa slojevima dozvoljava kreiranje novih, njihovo brisanje, menjanje osobina ili dodavanje posebnih elemenata kao što su senke ili zaobljene ivice za objekte sloja. Postoji i mogućnost kreiranja posebnih slojeva u svrhe podešavanja ili prikrivanja delova slike. Poseban meni dozvoljava promenu redosleda slojeva ili njihovog grupisanja. Na sledećoj slici je prikazan primer sa tri sloja. Prvi predstavlja sliku u pozadini sa simfonijskim orkestrom, drugi je instrument, a treći se odnosi na slova.



Slika 6.3 Primer slike sa tri sloja

Slojevi se mogu spajati, kombinovati sa drugim slojevima ili postaviti, ili kako se još kaže poravnati (flatten), u jedinstven sloj slike. Iako konsolidacija slojeva čini datoteku manjom, poravnanje je nepovratan proces nakon zatvaranja datoteke.

Pamćenje datoteke dok je još u nepovratnom obliku je uvek preproučljivo, ako je potrebno kasnije činiti neke promene.

Filter je efekat koji menja čitav sloj ili selektovani deo. Meni Filter se gotovo u potpunosti sastoji od različitih kategorija poređanih u kaskade koji omogućavaju transformaciju slike. Često korisnici redom primenjuju sve opcije da bi pronašli onu koja odgovara potrebama određene slike ili selektovanog dela. Opcije iz Filter galerije se mogu kombinovati simulatano sa ciljem da se ostvare odgovarajući efekti.

U meniju Filter se nalaze i opcije poput Extract..., Liquify..., Pattern Maker... ili Vanishing Point... za koje se može reći da više pripadaju u domen mini-programa nego filtera.

Pojedini filteri se primenjuju u samo jednom koraku i mogu imati velikog uticaja na sliku. U tu grupu spadaju filteri poput Blur, Facet ili Clouds. Dovoljno je samo kliknuti na njih. Filteri sa dijalogom (prepoznajemo ih jer u nazivu sadrže tri tačke) dozvoljavaju promenu obilja opcija. Ovi filteri pružaju mogućnost kratkog pregleda pre primene ili dodatne opcije za npr. distorziju, pretvaranje u tačku, primenu tekstura, izoštravanja...

OPTIONS

Prikaz linije Options

Linija Options prikazana na slici, eliminiše potrebu da se pristupa pojedinim paletama za svaki od alata. Linija je u svakom trenutku dostupna, odmah ispod linije sa menijima i opcije se lako menjaju. Ova linija menja svoj izgled u zavisnosti od aktivnog alata, pa je nemoguće u nekom kratkom roku opisati sve komponente koje se tu mogu naći. Međutim, sve linije opcija imaju neke zajedničke karakteristike:

- **Opcije alata** (Tool options). Ovo polje prikazuje ikonu trenutno aktivnog alata i može uključivati i neke opcije za taj alat.

- **Pop-up meni opcija** (Options pop-up menu). Linija Options može imati popup meni koji uključuje alate za crtanje ili brisanje, tzv. leteće (flyout-type) opcije koje dozvoljavaju korišćenje unapred podešenih i snimljenih podešavanja za različite alate, dodatne opcije, kao što su npr. veličina ikona korišćenih da predstave brush tip (četkice). Osim toga, postoji mogućnost vraćanja osnovnih vrednosti za određene opcije.
- **Opcije same linije** (Bar options) kao što su mod, prozirnost, stilovi tipova i fontovi su, takođe, smešteni na liniji Options.
- **Adobe bridge** dugme koje dozvoljava prelazak na Bridge aplikaciju jednim klikom.
- **Palette Well** (skup paleta) koja se obično pojavljuje sa desne strane linije Options. Palete se iz različitih grupa mogu prevući do Palette Well gde će se pojaviti odgovarajući tab (jezičak). Kada se klikne na taj jezičak pojaviće se paleta spremna za korišćenje. Kada ponovo kliknete na dokument paleta se vraća u jezičak. Palette Well je odličan alat za čuvanje paleta koje se često koriste.



Slika 6.4 Prikaz jednog vida linije Options

GLEDANJE I NAVIGACIJA

Funkcije dostupne preko View menija

Ogroman broj funkcija dostupan je preko View menija i verovatno je većini prosečnih korisnika poznat način korišćenja za zumiranje slike. Osim toga moguć je izbor moda koji npr. dozvoljava da se slika prikaže preko celog ekrana sa menijima i paletama ili npr. samo sa paletama.

U meniju View Show može se izvršiti izbor dodatnih elemenata koji će biti prikazani u prozoru:

- **Layer Edges** – prikazuje plavo prugasto polje koji predstavlja granicu sadržaja selektovanog layera,
- **Selection Edges** – pomerajuće linije koje definišu granicu selektovane oblasti,
- **Target Path** – linije i krive koje definišu oblik ili se koriste za selektovanje dela slike,
- **Grids and Guides** – linije koje se koriste za poravnanje selektovanih delova, objekata ili drugih komponenti,
- **Smart Guides** – omogućava precizno pozicioniranje i poravnanje sadržaja sloja, a pojavljuje se samo kada je potrebno,
- **Slices** – pravougaoni delovi slike koji se mogu optimizirati ili se primeniti neki Web alati,
- **Annotations** – beleške na ekranu ili audio napomene koje se mogu dodati imenjati...

PALETE

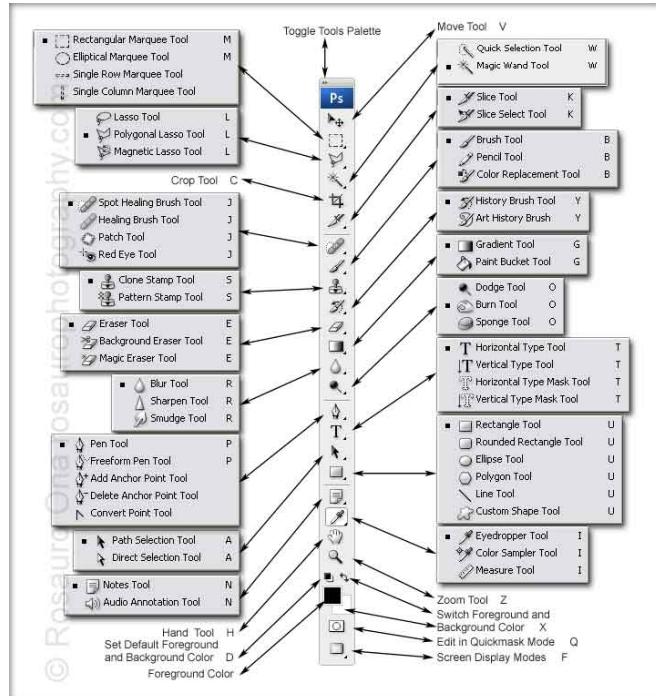
Prikaz palete u PS-u

Paleta Tools se otvara uz pomoć Window Tools. Neke od značajnijih opcija su:

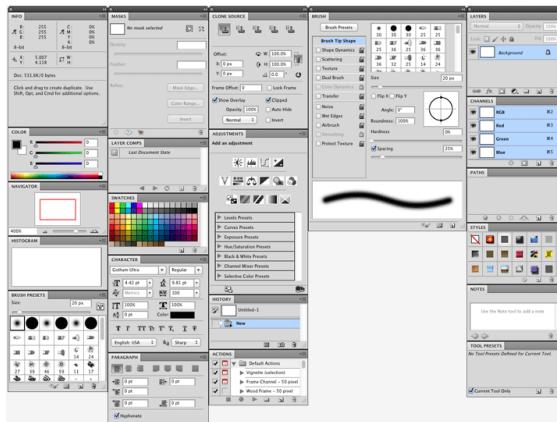
- za brzo sakrivanje i prikazivanje palete Tools dovoljno je pristisnuti taster Tab,

- za zatvaranje palete, potrebno je dva puta kratko kliknuti na naslovnu liniju iznad ikone sa perom na vrhu linije,
- za pomeranje palete Tools dovoljno je prevući naslovnu liniju

Za izbor alata, dovoljno je kliknuti na Tools paletu. Mali crni trougao u donjem desnom uglu svakog alata označava da je više alata sakriveno iza tog alata u flyout meniju. Na slikama prikazane su palete u različitim verzijama Photoshop-a (CS3 i CS5).



Slika 6.5 Paleta Photoshop CS3



Slika 6.6 Paleta Photoshop CS5

PALETA TOOLS

Prikaz paleta tools

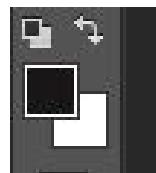
Osim toga alatima se može pristupiti i uz pomoć tastature.

Za većinu alata, alatu se može pristupiti pritiskom na Shift taster i tasterom sa slovom koje

se nalazi kod vidljivog alata. Na primer, za izbor olovke (Pencil), koja je deo flyout menija sa Brush alatom, dovoljno je pritisnuti Shift+B.

Paleta Tools je podeljena na tri osnovne sekcije: alati, za promenu boja i ikone za modove maskiranja i opcije gledanja. U nastavku je data lista detalja vezanih za ovu paletu:

- **Foreground Color i Background Color** prikazuju trenutnu prednju površinu i boje pozadine. Kada se koristi neki od alata, kao što su četkica ili olovka, može se koristi bilo koja od ovih boja. Malo crno-belo dugme služi za izbor osnovnih boja. Klikom na Default Colors, ikonu u donjem levom uglu ove opcije, boje se vraćaju na osnovne. Kliknite na ikonu sa zakrivljenom strelicom da biste se prebacili sa boje prednje površine na boju pozadine



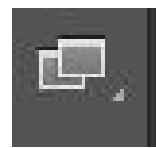
Slika 6.7 Boja prednje površine i boja pozadine

- **Editovanje u Standard i Quick Mask** modu izvodi se klikom na jednu od dve opcije prikazane na sledećoj slici. One dozvoljavaju rad ili u Standard (ikona s leve strane) ili Quick Mask (ikona sa desne strane) modu koji omogućava pregled, menjanje i selektovanje.



Slika 6.8 Standard i Quick Mask ikone

- **Screen Modes:** Standardni mod prikaza predstavljen je ikonom sa krajnje leve strane i znači podrazumevanu vrednost (deafault). Ovaj mod rada omogućava da se vidi cela Photoshop radna površina. Postoji i Full Screen mod sa linijom sa menijima koja sakriva pozadinu radne površine i druge otvorene slike. Full Screen mod, sa krajnje desne strane služi za sakrivanje radne površine pozadine, svih otvorenih slika i linije sa menijima



Slika 6.9 Screen Mode ikone

ALATI ZA SELEKTOVANJE

Prikaz alata za selektovanje

Alati za selektovanje predstavljaju pokretačku snagu Photoshop aplikacije. Oni dozvoljavaju zahvatanje i izolovanje nekih piksela tako da se može menjati ili manipulisati samo delom slike. Marquee alat dozvoljava zahvatanje piksela u obliku pravougaonika, elipsi ili pojediničnih kolona ili vrsta. Slika prikazuje primer selektovanja u obliku elipse. Alata u

obliku lasa dozvoljava selektovanje slobodnih oblika. Magic Wand tool omogućava selektovanje piksela slične boje. Move i Crop alati omogućavaju pomeranje i „skraćivanje“ slike.

Alat Path kreira i modifikuje putanje (**paths**) koji su elementi koji se sastoje od pravih i zakrivljenih segmenata kao i tačaka sidra (**anchor**). Ove putanje mogu biti baza za selektovanje ili definisanje oblika. Zbog svoje preciznosti korišćenje Path alata omogućava kreiranja složenih selekcija, što u krajnjoj liniji daje bolje rezultate od onih koji se mogu postići Selections alatima. Path Selection alat selektuje putanje i njihove komponente nakon što ih nacrtate.



Slika 6.10 Primer korišćenja Pen i Selection alata

Alati za crtanje, uopšteno, dozvoljavaju primenu boja na određene piksele ili njihovo brisanje. Kada se koristi Gradient alat, mogu se koristiti višestruke boje simultano, a uz pomoć Art History Brush alata izvode se stilizovani efekti. Color Replacement alat dozvoljava zamenu boje. Naredna slika prikazuje korišćenje Custom Shape alata, obojenog uz pomoć posebnog efekta četkicom, a na desnoj slici prikazan je slučaj kada je deo izbrisana uz pomoć Eraser alata.



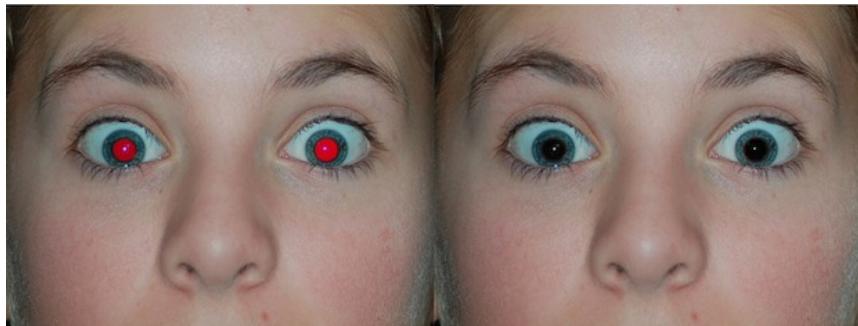
Slika 6.11 Rezultat primene Custom Shape i Erase alata

KORIŠĆENJE ALATA ZA KLONIRANJE I OPORAVAK

Prikaz korišćenja alata za kloniranje i oporavak

Alati za kloniranje i retuširanje su snažni alati koji se koriste kada je potrebno popraviti neku sliku. Ovi alati dozvoljavaju da dupliranjem delova slike, bojenje uz pomoć obrazaca, sređivanje ogrebotina bez šavova, neravnina ili drugih nedostataka.

Spot Healing Brush brzo uklanja male nedostatke svih vrsta. Na primer, Red Eye alat uklanja pojavu crvene refleksije na fotografijama kod očiju.



Slika 6.12 Uklanjanje „crvenih očiju“

Fokus i alati za toniranje dozvoljavaju menjanje slika i to menjanjem piksela na različite načine. Oni se mogu posvetljivati, potamnjivati, zamagliti ili izoštravati određenim delovima. Ovi alati odlične rezultate postižu kod malih površina, bolje nego za čitave slike.

PREGLED, NAVIGACIJA, UZORKOVANJE I KOMENTARI

Prikaz pregleda, navigacije, uzrokovana i komentara

Photoshop ima dosta opcija koje pomažu prilikom pregleda ili navigacije slike u prozoru. Ovi alati dozvoljavaju da se vrši zumiranje, pomera slika u oviru prozora, mere rastojanja i uglovi itd. Eyedropper i Color Sampler alati dozvoljavaju izbor i probanje izabarene boje u nekom kontekstu. Notes i Audio Annotation alati omogućavaju kreiranje pisanih i audio beleški koji se mogu ostaviti u prozoru sa slikom. To je naročito korisno kada više osoba radi na jednoj slici ili jednostavno kao podsetnik.

Osim svega navedenog postoji i mogućnost podešavanja datih opcija:

Redosled je sledeći:

1. Bira se alat za koji se žele izvršiti podešavanja. Ako neki alat ne dozvoljava tu mogućnost kao što je Measure Tool tada je dugme Tool Preset Picker obojeno sivom bojom.
2. Bira se opcija za izabrani alat na liniji opcija. Na primer, ako se izabere Rectangular Marquee alat, moguće je izabrati Fixed Size iz Style pop-up menija i uneti željenu visinu i širinu u Height i Width polju.
3. Kliknuti na Tool Preset Picker dugme na Option liniji. Druga mogućnost je da se izabere Window-->Tool Presets za rad uz pomoć palete Tool Presets.
4. Kliknuti na Create New Tool Preset dugme ili ako se koristi i Tool Presets paleta, izabere se New Tool Preset iz menija paleta.



Slika 6.13 Tool Preset Picker dugme

5. Dati naziv podešavanju i kliknuti na OK. Tada je podešavanje spremno za korišćenje.
6. Za izbor podešavanja potrebno je uraditi jednu od sledećih stvari:
 - kliknuti na Tool Preset Picker dugme i izabrati podešavanje iz Picker pop-up menija,
 - Izabrati podešavanje u Tool Presets paleti, ili
 - izabrati podešavanje u Preset Manager, izborom Tools iz pop-up menija, a potom odgovarajućeg podešavanja.

POKAZNI PRIMER U ADOBE PHOTOSHOP-U

Primer za vežbu

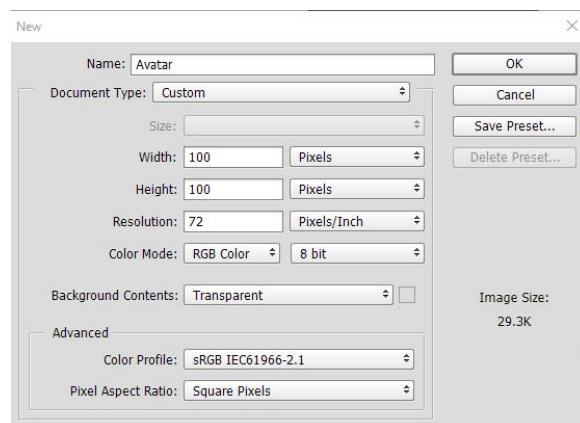
Predviđeno vreme za pokazni primer je 10 minuta.

Primer:

Koristeći Adobe Photoshop kreirati avatar dimenzija 100x100 piksela. Za kreiranje avatara koristiti sliku deda_mraz.jpg koju možete preuzeti sa e-learning sistema.

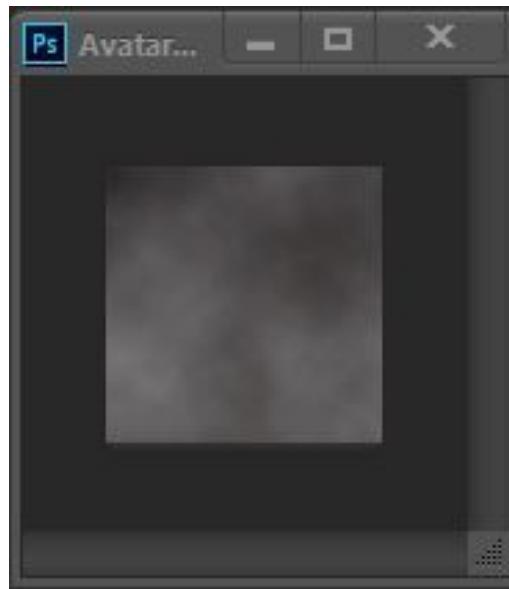
Rešenje:

Na početku otvorimo novi dokument sa FileNew i unosimo podatke koji su za dati problem.



Slika 6.14 Određivanje dimenzija slike

Trenutni raspored boja možete vratiti u početno stanje tako što otkucate veliko slovo - „D“. U narednom koraku dodajemo pozadinu uz pomoć npr. opcije Filter->Render->Clouds. Pokušajte da koristite i druge ponuđene opcije za izgled pozadine.



Slika 6.15 Filter->Render->Clouds

Sa File->Open otvaramo sliku koja će nam poslužiti za kreiranje avatara. Kao što je u zadatku rečeno, u ovom primeru koristićemo sliku deda_mraz.jpg. Uz pomoć alata Lasso na paleti sa alatima obeležimo željeni deo slike.

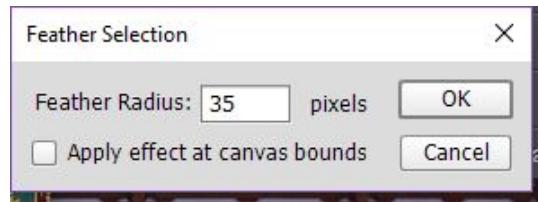
POKAZNI PRIMER U ADOBE PHOTOSHOP-U - NASTAVAK

Primer za vežbu - nastavak



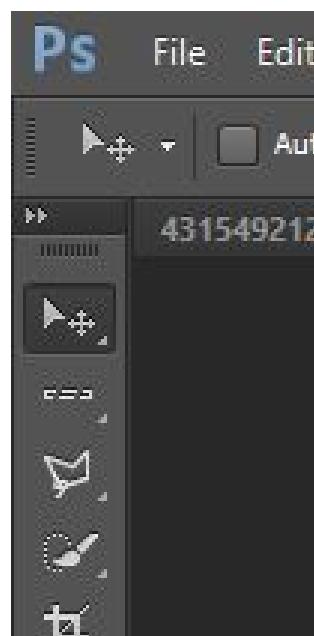
Slika 6.16 Prikaz selektovane slike

Nakon toga kliknemo desni taster miša na obeleženu oblast i biramo opciju Feather. U prozor koji će pojaviti, unosimo npr. vrednost 35 piksela



Slika 6.17 Opcija Feather

Potom pritiskamo kombinaciju tastera CTRL+C i kopiramo izabrani deo slike. Potom kliknemo na prozor sa pripremljenom pozadinom i pritisnemo kombinaciju tastera CTRL+V. Zatim na liniji sa alatima kliknemo na alat za pomeranje i pomjerimo sliku tako da se lik nalazi u centru.



Slika 6.18 Alata za pomeranje



Slika 6.19 Centrirana slika

Budući da je slika veća od zadatog okvira, pritisnemo CTRL+T i potom desnim klikom biramo opciju Scale. Potom pomerajući jednu od tačaka koja se nalazi na uglu definisane oblasti, pri čemu držimo pritisнуте tastere CTRL+SHIFT+ALT da bi odnos visine i širine ostao isti., možemo da dobijemo rezultat koji je prikazan na slici.

Naravno, vaš rezultat može izgledati i drugačije. Nakon toga klikom na znak za štikliranje, koji se nalazi u gornjem desnom uglu prozora, potvrđujemo promenu.

KORIŠĆENJE OPEN SOURCE ALATA

U ovoj sekciji biće prikazano korišćenje open source alata GIMP

GIMP je besplatan alat za obradu slika. Ovaj alat podržava većinu opcija koje ima i Photoshop. Može se preuzeti na sajtu www.gimp.org. Kada se program preuzme započinje se instalacija. Početak je prikazan na slici 20.



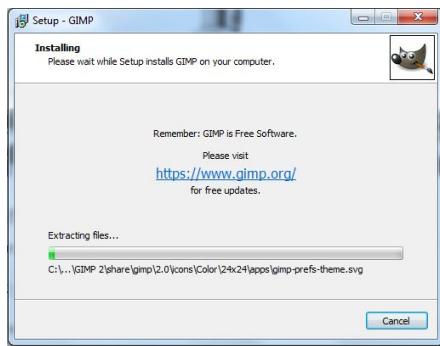
Slika 6.20 Odabir jezika

Sledeći korak je započinjanje instalacije što je prikazano na slici 21.



Slika 6.21 Početak instalacije

Kada se klikne dugme "Install" započinje se proces instalacije prikazan na slici 22.



Slika 6.22 Proces instalacije

Instalacija traje kratko i poslednji korak je završetak instalacije prikazan na slici 23.

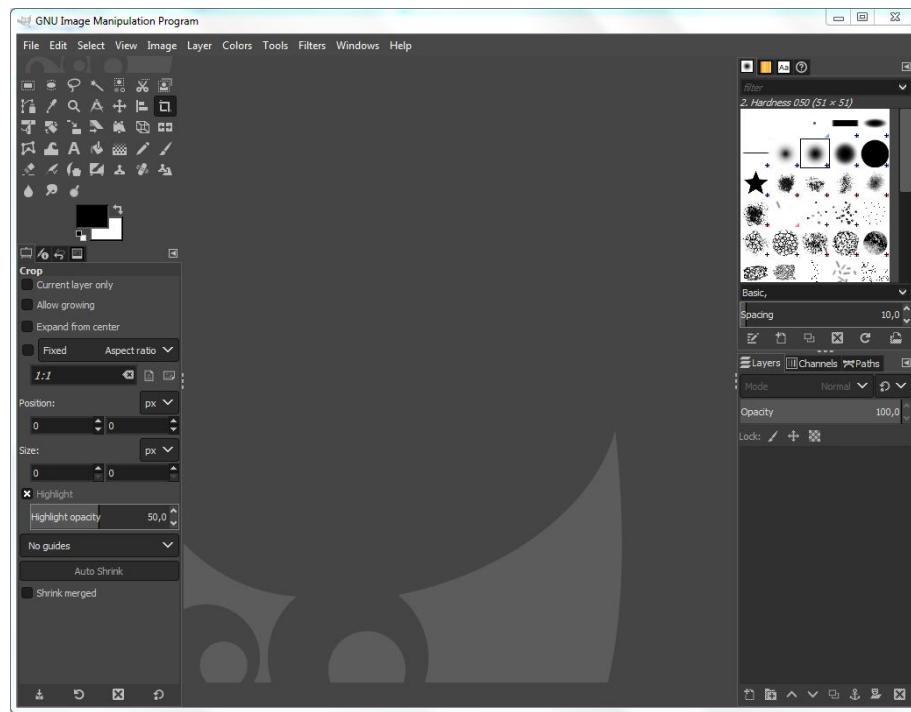


Slika 6.23 Završetak instalacije

IZGLED GIMP-A

U ovoj sekciji biće prikazan izgled alata

Na slici 24 prikazan je izgled alata GIMP pri pokretanju.



Slika 6.24 Izgled GIMP alata

Može se primetiti da je raspored elemenata sličan Photoshopu. U sredini se nalazi radna površina u kojoj se otvaraju slike. Sa leve i desne strane se nalaze različite opcije za obradu slika.

Osim ovih opcija, u liniji menija se mogu pronaći još mnoge opcije koje se takođe mogu postaviti i na aktivnu radnu površinu.

OTVARANJE SLIKE

U ovoj sekciji biće prikazano kako se otvara slika u GIMP-u

Slika se otvara klikom na opciju menija File -> Open i zatim se izabere slika koju želimo da otvorimo. Na slici 25 je prikazano kako izgleda otvorena slika na radnoj površini.

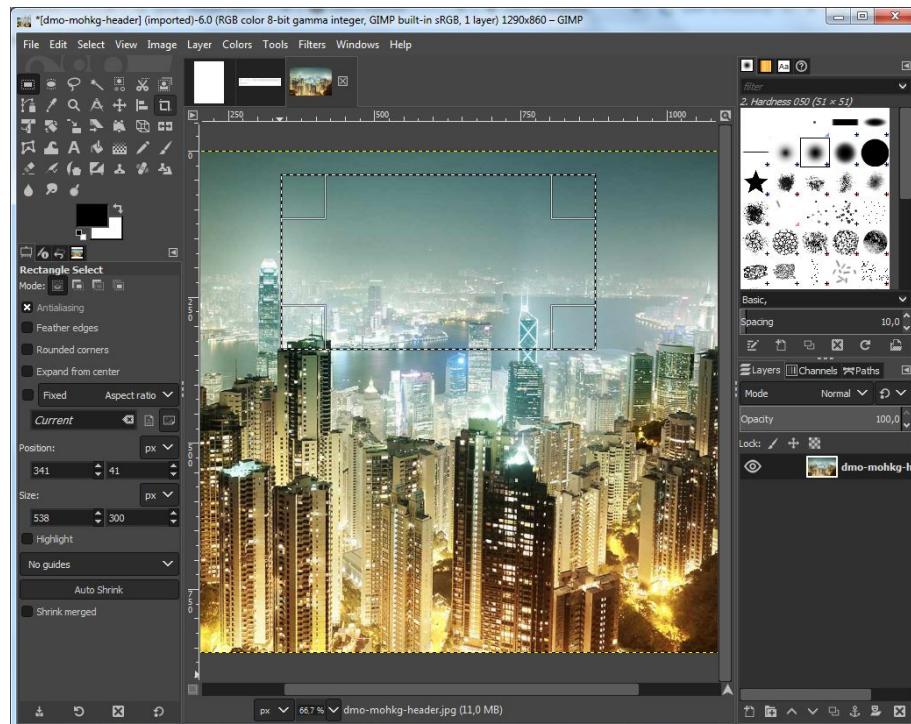


Slika 6.25 Otvorena slika

SELEKTOVANJE DELA SLIKE

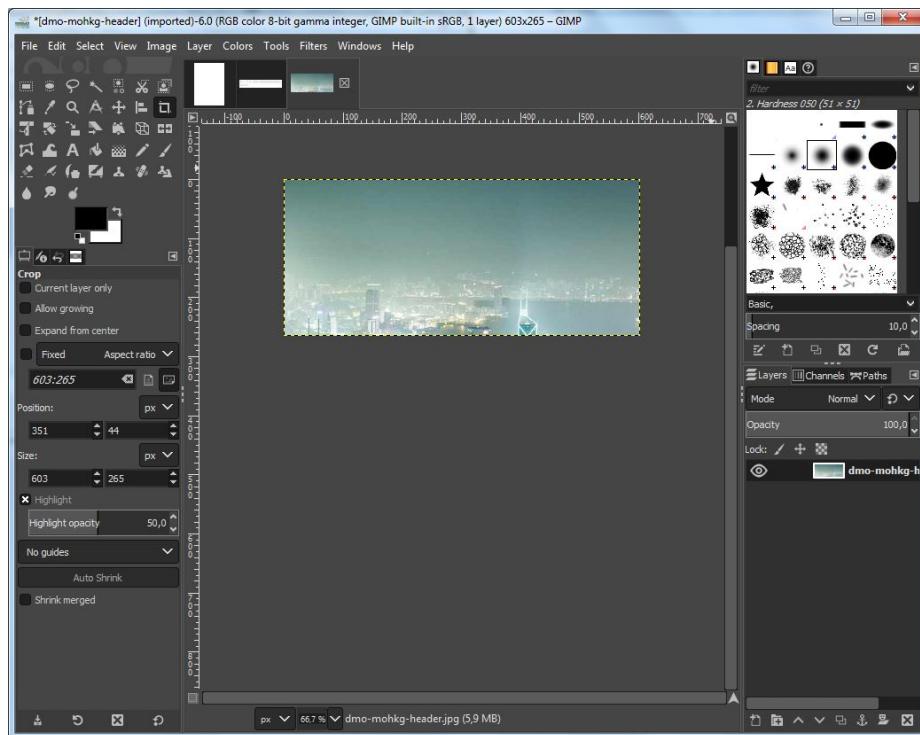
Prikaz alata za selektovanje u GIMP-u

U gornjem levom uglu nalazi se alat za pravougaono selektovanje dela slike. Kada se odabere alat, jednostavno se prevuče preko željenog dela slike (Slika 26).



Slika 6.26 Selektovani deo slike

Deo slike se takođe može i iseći koristeći alat Crop koji se takođe može naći u gornjem levom uglu među alatima. Slika se selektuje kao i sa prethodno prikazanom opcijom i zatim se klikne na selektovani deo kako bi se on isekao. Rezultat je prikazan na slici 27.

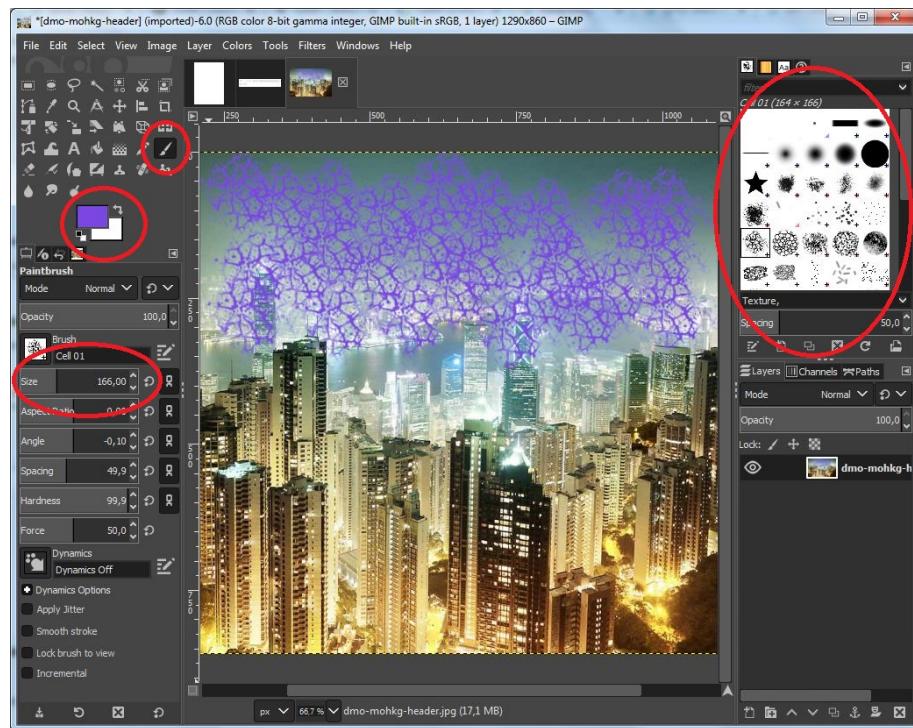


Slika 6.27 Isecanje dela slike

PAINTBRUSH ALAT

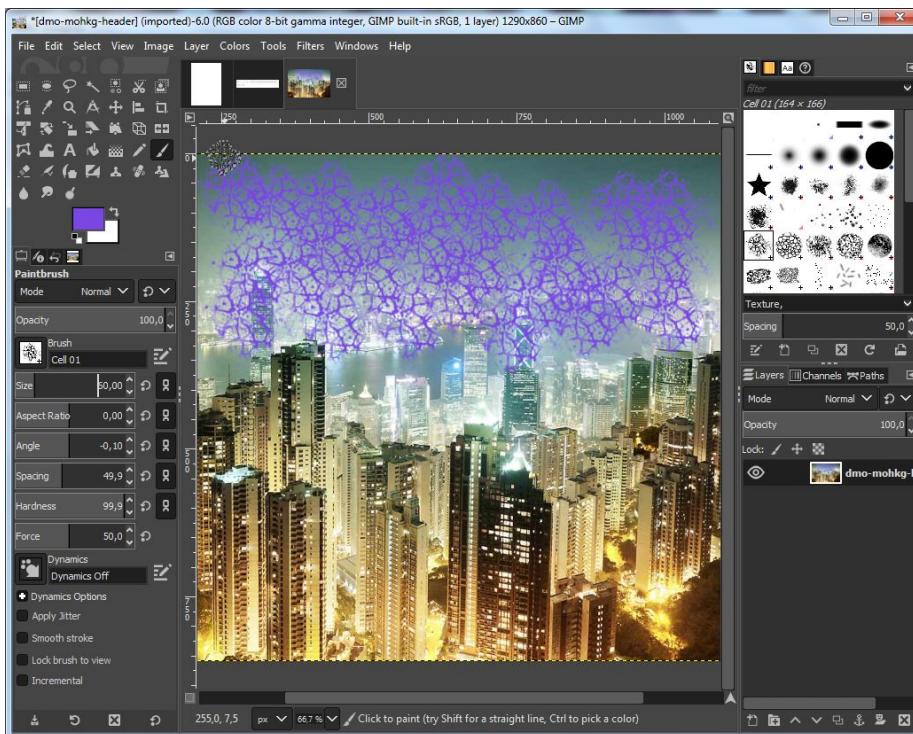
Prikaz paintbrush alata

Paintbrush alat se može koristiti za crtanje po sliци. Kada se selektuje alat, sa desne strane se može izabrati tekstura odnosno oblik četkice. Sa leve strane se nalazi opcija za promenu boje kao i veličine četkice. Ove opcije obeležene su na slici 28.



Slika 6.28 Opcije za četkicu

Za ovaj primer izabrana je ljubičasta boja kao i jedna od tekstura za četkicu i rezultat se može videti na slici 29.

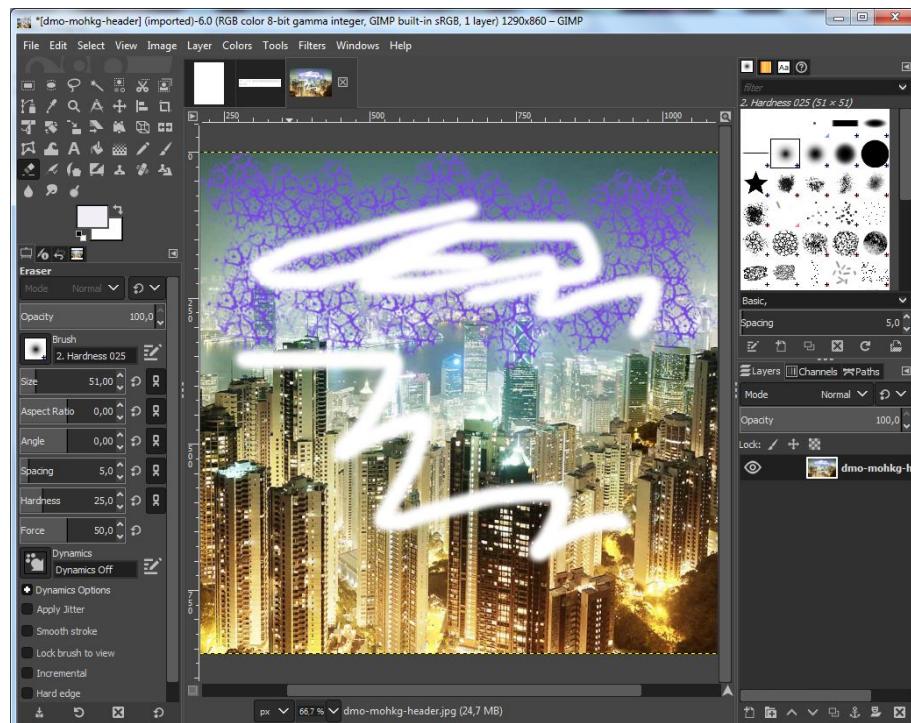


Slika 6.29 Prikaz četkice

BRISANJE SLIKE

Prikaz alata za brisanje u GIMP-u

Alat za brisanje se može naći u levom gornjem uglu među ostalim alatima. Njemu se takođe može podešiti veličina kao i oblik, kao i sa prethodnim alatom. Rezultat brisanja slike je prikazan na slici 30.



Slika 6.30 Brisanje slike

Na sajtu GIMP alata mogu se naći razni korisni tutoriali za korišćenje mnogih opcija koje alat nudi.

Tutorijali se mogu naći na zvaničnom sajtu.

▼ Poglavlje 7

Pokazne vežbe: Kreiranje Power Point prezentacije

POWER POINT

PowerPoint je softver za kreiranje prezentacija koji je deo Microsoft Office paketa

Predviđeno vreme pokaznih vežbi je 30 minuta.

Sposobnost prezentovanja je jedna od ključnih osobina savremenog profesionalca, bez obzira na oblast. Iskusni stručnjaci znaju da i malim unapređenjima u ovoj oblasti mogu da postignu odlične rezultate. **PowerPoint** je softver za kreiranje prezentacija koji je deo Microsoft Office paketa. Ovaj alat koristi grafički pristup za kreiranje prezentacija u formi slajdova, koji imaju zadatak da daju podršku oralnoj prezentaciji. Ovaj program se dosta koristi u poslovnom svetu, a poseban značaj ima za potrebe različitih treninga i edukacija. Ono što posebno izdvaja ovaj program je jednostavnost korišćenja, pa ga je u tom smislu lako i naučiti. Prezentacije se snimaju sa ekstenzijom .ppt. Program koji je deo OpenOffice.org paketa je Impress.

Veoma sličan po opcijama PowerPoint-u ovaj alat ima i mogućnost izvoženja prezentacija u SWF ili PDF format. Softver je razvio Sun Microsystems, a pruža i podršku za rad sa .ppt datotekama. Inače, format u kojem se snimaju Impress prezentacije je .odp. Jedno od „velikih“ pravila za kreiranje prezentacije kaže: „Na početku recite šta ćete predstaviti, potom predstavite i na kraju recite šta ste predstavili“. Drugim rečima, svaka prezentacija ima tri dela: uvod, telo i zaključak. Svrha uvoda nije samo da se predstave teme, već i da se zainteresuju oni koji prate.

Pažnja slušaoca se stiče tako što se uključe u vašu temu. Ovo se može postići postavljanjem pitanja na samom početku prezentacije, predstavljanjem nekih zanimljivosti koristeći neki citat ili pričanjem neke kratke zanimljive priče. Na primer, može se na početku postaviti pitanje „Koliko ljudi ovde ima kompjuter?“, što je mnogo bolje nego prezentaciju početi izjavom: „Danas ću vam pričati o Internetu“. Sadržaj prezentacije se može predstaviti kratkom „konturom“ diskusije o kojoj će biti reči. Jedna od mogućnosti je da se na početku objasne motivi za obrađivanje teme. Telo prezentacije treba da podrži uvodni deo nuđenjem činjenica, mišljenja i razloga zbog kojih se ta tema obrađuje. Zaključak treba da ponovi glavne tačke bez navođenja primera. Razmišljajte o tome kao o sumarnom delu koji ističe sve ono što želite da slušaoci zapamte. Možete završiti sa preporukama, ličnim stavom, zapažanjem ili pitanjem. Vaša završna izjava treba da objedini čitavu prezentaciju.

Veoma je važno uočiti da prezentaciju predstavlja čovek, a ne računar. U nastavku će uz pomoć jednog primera biti opisana procedura u 11 koraka za kreiranje uspešne prezentacije

korišćenjem PowerPoint-a. Osnovni principi korišćenja OOo Impress su slični, a razlike koje postoje biće posebno opisane.

IDENTIFIKOVANJE PUBLIKE ZA PREZENTACIJU I SVRHA PREZENTACIJE

Pažljivo razmislite i definišite publiku i svrhu u jednoj rečenici

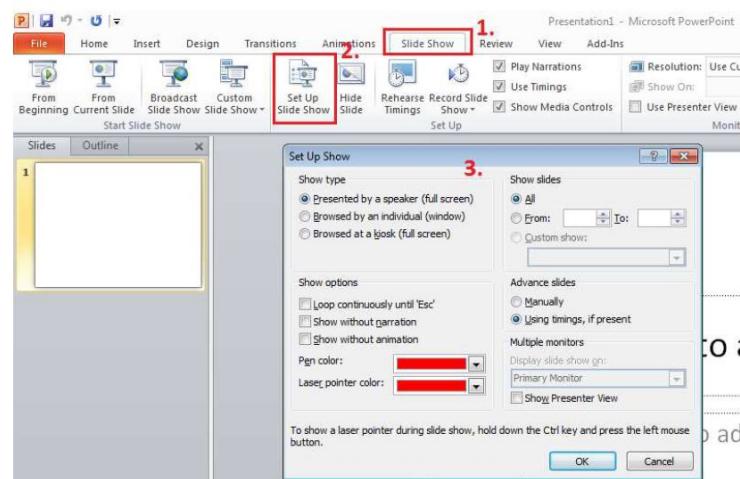
Različita publika različito reaguje na istu prezentaciju. Na primer, ako predstavljate novi proizvod prezentacija će biti potpuno drugačija od one kojom nešto predstavljate svojim kolegama na poslu. Pažljivo razmislite i definišite publiku i svrhu u jednoj rečenici. To može da izgleda ovako: Pripremam prezentaciju za 100 radnika neke fabrike kojima predstavljam privatni penzioni fond. Pripremam prezentaciju za malu grupu potencijalnih korisnika mojih Internet servisa sa željom da im objasnim osnovne funkcije. Pripremam prezentaciju za profesora i želim da mu predstavim urađen seminarски rad na najbolji mogući način. Naravno, u ovom primeru podrazumevaćemo da se radi o poslednjem slučaju.

ODABIR NAČINA PREZENTACIJE

Generalno postoje tri načina za predstavljanje prezentacije

Generalno postoje tri načina za predstavljanje prezentacije. Prvi predstavlja tradicionalan način gde onaj koji prezentira stoji pred publikom i izlaže prezentaciju. Drugi način je samostalna prezentacija. Za razliku od prvog slučaja ovde je neophodno da sve informacije budu predstavljene na slajdovima jer ne postoji dodatna podrška predavača. Konačno, postoje i prezentacije koje imaju mogućnost interakcije sa korisnikom. U ovom slučaju postoji jedna osoba koja prati prezentaciju i ima mogućnost unosa i izbora informacija.

Opcije koje omogućavaju izbor odgovarajućeg načina prezentacije kod PowerPoint-a nalaze se u delu Slide Show→Set Up Slide Show. Nakon aktiviranja ove opcije, na ekranu će se pojaviti prozor, gde u gornjem levom uglu, obeleženom sa Show type, birate način prezentovanja.



Slika 7.1 Set Up Show prozor

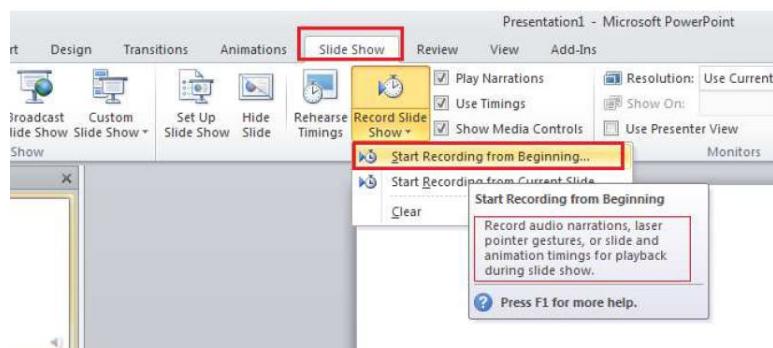
ODABIR NAČINA PREDSTAVLJANJA

Može se reći da izbor metoda prezentacije predstavlja i izbor načina komunikacije sa slušaocima

Može se reći da izbor metoda prezentacije predstavlja i izbor načina komunikacije sa slušaocima. Budući da je komunikacija često dvosmerna koristi se i termin interakcija. U tom smislu PowerPoint ima puno opcija: predstavljanje slajdova uz pomoć video projektor-a, snimanje u formatu za predstavljanje na Web-u, priprema slajdova u formatu od 35 mm za prikazivanje na starim projektorima, priprema folija za predstavljanje uz pomoć grafskoga i, konačno, postoji mogućnost štampanja na papiru koji se mogu distribuirati slušaocima prezentacije.

Za naš primer izabrani način predstavljanja je uz pomoć projektor-a. Ukoliko studirate preko Interneta možete prezentaciju pripremiti i tako što ćete snimiti svoj glas uz pomoć mikrofona. Izaberite opciju Slide Show→Record Slide Show...

U okviru prozora postoji više opcija za podešavanje snimanja. Nakon što kliknete na OK počinje snimanje zvuka pri čemu narator kontroliše tok i dužinu prezentacije.

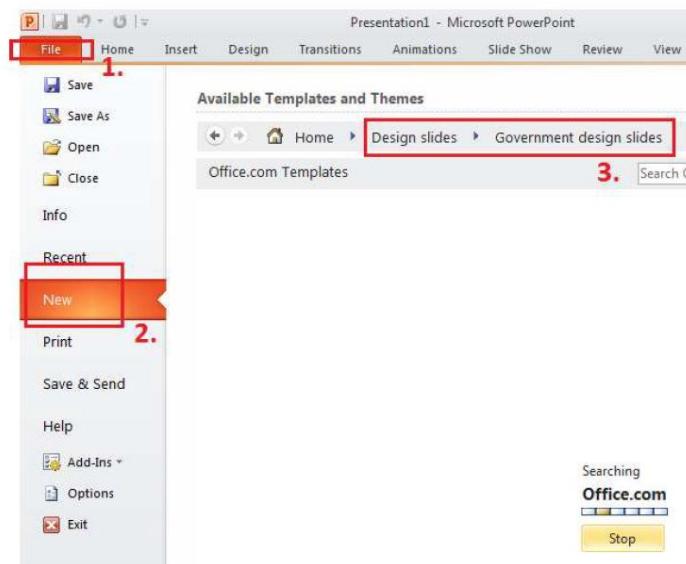


Slika 7.2 Record Slide prozor

IZBOR ŠABLONA

PowerPoint ima dve vrste šablona: šablon za prezentacije i dizajn šablone

PowerPoint ima puno šablona koji se mogu koristiti u zavisnosti od potreba. Za te potrebe mogu se koristiti dostupe teme i templejti kojih ima jako puno. PowerPoint ima dve vrste šablona: šablon za prezentacije i dizajn šablone. Šablone prezentacija čine čitave prezentacije sa slajdovima koji su pripremljeni za određene potrebe, na primer za pripremu biznis plana ili prezentaciju određenog proizvoda ili usluge. Sa druge strane, šablon dizajna predstavlja kombinaciju izbora slova, boja, grafičkih elemenata... Za izbor odgovarajućeg šablona potrebno je kliknuti na File→New. U zavisnosti od verzije PowerPoint-a pojaviće se Task Pane sa desne strane prozora pa je potrebno da kliknete na željenu opciju ili će se odmah pojaviti prozor koji vam omogućava izbor



Slika 7.3 Task Pane prozor

KORIŠĆENJE ŠABLONA

Blank prezentacija pokreće rad na novoj prezentaciji bez korišćenja podloge

Design Template dozvoljava da se podese pozadina i šema boja iz skupa podloga, pre nego se nastavi rad na prezentaciji. Blank prezentacija pokreće rad na novoj prezentaciji bez korišćenja podloge. Poslednja opcija u datom prozoru daje spisak već kreiranih PowerPoint prezentacija koje se mogu izabrati i nastaviti dalji rad na njima. Za potrebe našeg primera izabraćemo opciju Blank presentation. Čest je slučaj da se na početku kreira Blank prezentacija, a da se podloga odredi kasnije tako da bude u kontekstu onoga što se prezentira. Kod OOo Impress proces izbora prezentacije je nešto drugačiji i omogućava kreiranje osnove prezentacije u tri koraka. Prvi je izbor između prazne prezentacije (engl. **Empty presentation**), postojećeg šablona (engl. **From template**) i već pripremljene prezentacije (engl. **Open existing presentation**).



Slika 7.4 Presentation Wizard prozor

Treba reći da je izbor dizajna slajdova veoma siromašan u odnosu na PowerPoint, ali zato korisnik ima puno slobode da sam definiše kako će prezentacija izgledati i na taj način postigne značajne efekte u cilju da na efektan način predstavi željeni sadržaj. Ponekad je bolje

da prezentacija nema nikakav dizajn, nego neki koji će u potpunosti biti van konteksta materije koja se izlaže.

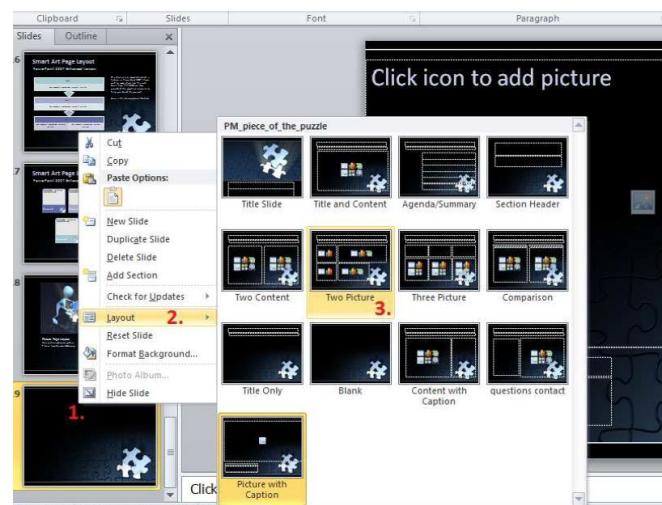


Slika 7.5 Odabir dizajna i efekata

KREIRANJE SLAJDova

Nakon izbora u prethodnom koraku, kreiraju se slajdovi i njihov sadržaj.

Nakon izbora u prethodnom koraku, kreiraju se slajdovi i njihov sadržaj. Za svaki slajd priprema se tekst uz pomoć tekstualnih polja (Placeholder). Kada se otvori nova prezentacija, PowerPoint prikazuje dijalog New Slide koji sadrži nekoliko AutoLayout skica za slajdove različitih namena. Ove skice omogućavaju konzistentnost predstavljanja kroz prezentaciju.



Slika 7.6 Slide Layout prozor

Tako je u primeru na slici obeležen Two Picture slajd koji se odnosi na dve slike na slajdu. Dovoljno je kliknuti na OK, odnosno na samu sliku slajda u Task Pane-u, i pojaviće se novi slajd u prezentaciji.

PowerPoint prozor obično ima razvojno okruženje koje se sastoji od tri oblasti:

- okvir sa kratkim pregledom svih slajdova (engl. **Outline Pane**),
- okvir slajda (engl. **Slide Pane**) prikazuje svaki tekst koji se unosi u odgovarajuća polja za tekst na slajdu ili objekte koji su smešteni na slajdu, i
- okvir beležaka (engl. **Notes Pane**) gde se mogu unosti beleške vezane za prezentaciju.



Slika 7.7 Tri oblasti prozora

PLACEHOLDERI

*Većina već pripremljenih skica ima tzv. „nosioce položaja“ (engl. **Placeholders**), tačnije tekstualna polja u koja se unosi odgovarajući tekst*

Većina već pripremljenih skica ima tzv. „nosioce položaja“ (engl. **Placeholders**), tačnije tekstualna polja u koja se unosi odgovarajući tekst. Veoma je važno unositi odgovarajuće informacije u polja koja su predviđena za tu vrstu teksta, jer to pomaže onome ko prezentira da se lakše orijentiše prilikom prezentacije. Na primer, u polje za naslov, obeleženo na slici sa Click to add title, treba uneti naslov čitave prezentacije. U našem slučaju to: „Skype kao sredstvo komunikacije“. U donje polje unosimo podnaslov i podatke o autoru. U ovom primeru to će biti: „IT101 – Seminarski rad“.

DODAVANJE UVODNOG I ZAVRŠNOG SLAJDA

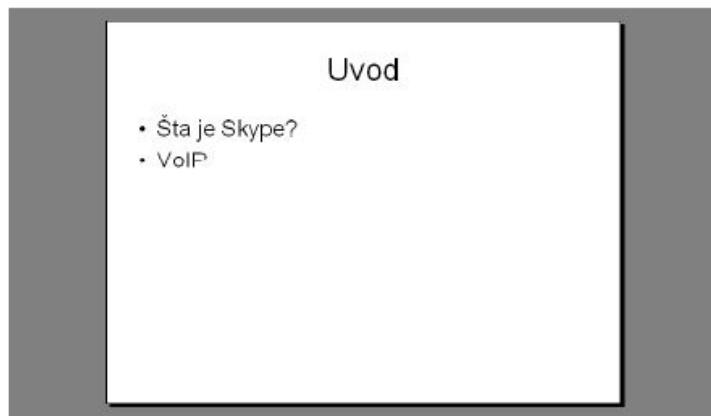
Za ubacivanje sledećeg slajda koristi se opcija Home→New Slide na ribonu

Slajd koji je napravljen obično se zove naslovnim slajdom, jer sadrži naslov i ime autora. Za ubacivanje sledećeg slajda koristi se opcija Home→New Slide na ribonu. Za potrebe ovog primera dodaćemo slajd koji treba da nam omogući uvodno izlaganje. Najbolje je da to bude slajd Title and Text. U naslovni deo unosimo tekst: „Uvod“, a u nastavku kratko objašnjenje o čemu će biti reči u prezentaciji.



Slika 7.8 Početni slajd

Na sličan način dodajemo glavni i završni slajd:



Slika 7.9 Uvodni slajd

KREIRANJE VIZUELNOG IZGLEDA

Treba imati na umu da slušaoci vole konzistentnost

Termin vizuelni izgleda se odnosi na ukupni utisak koji publika dobija prilikom gledanja prezentacije. Treba imati na umu da slušaoci vole konzistentnost. Na primer, poželjno je da ako pripremate prezentaciju neke kompanije na svakom slajdu stoji logo te kompanije. Zatim, poželjno je dosledno pojavljivanje naslova prezentacije uvek na istom mestu na slajdu i sl. Za ubacivanje slike koristi se opcija Insert→Picture. Slika se može pomerati na standardan način, kao i u svim ostalim Office aplikacijama. Tekst se formatira na standardni način uz pomoć opcija na liniji sa alatima za formatiranje. Uvek kada je potrebno menjati format teksta, potrebno je pre svega obeležiti tekst koji se menja. Sve opcije koje se koriste od tog trenutka imaju efekta samo na obeleženi deo. Opcije na liniji Insert > Shapes je namenjena za iscrtavanje različitih oblika, linija, elemenata dijagrama...

KREIRANJE BELEŠKI

Beleške mogu biti potpune kopije slajdova ili samo skraćene verzije

Ovaj korak uglavnom se odnosi na prezentacije koje vodi prezenter-predavač. Najčešće je publika prisutna tokom prezentacije pa je potrebno obezbediti beleške uz pomoć kojih će pratiti izlaganje predavača. Beleške mogu biti potpune kopije slajdova ili samo skraćene verzije.

PRIPREMA PREZENTACIJE

Bez obzira koji tip prezentacije se priprema, potrebna je vežba

Bez obzira koji tip prezentacije se priprema, potrebna je vežba. Priprema zavisi od načina prezentovanja. Ako imate prezentaciju „uživo“, preporuka je da se prvo provere slajdovi. Pre svega, da li su kompletni, precizni, poređani po željenom redosledu. Kod samostalnih prezentacija potrebno je obratiti pažnju da se nikakve dodatne korekcije ne mogu vršiti. Kod predstavlja gde postoji predavač nedostaci na slajdovima se mogu kompenzirati odgovarajućim objašnjenjima. Međutim, kod samostalnih prezentacija to nije slučaj. Potrebno je uzeti u obzir vreme slušaoca da pročita tekst pri čemu parametar treba da predstavlja brzina čitanja najsposrijed slušaoca. Jedna od najvažnijih stvari prilikom pripreme prezentacije je procena vremena potrebnog publici da pročita dati tekst na svakom slajdu. Kod interaktivnih prezentacija potrebno je potencijalnim slušaocima obezbediti odgovarajuću dugmad uz pomoć kojih će kontrolisati tok prezentacije, pa u tom slučaju vreme nije neophodan parametar.

PREDSTAVLJANJE

Ako tokom prezentacije predavač želi slušaocima da skrene posebnu pažnju na neke delove prezentacije može da koristi i alate poput olovke

Kada je pripremljena prezentacija se pokreće pritiskom na taster F5 ili klikom na Slide Show > From Beginning. Ako tokom prezentacije predavač želi slušaocima da skrene posebnu pažnju na neke delove prezentacije može da koristi i alate poput olovke. Dovoljno je da klikne desni taster miša i u kontekstnom meniju izabere Pointer Options, a potom odgovarajući oblik i boju olovke. Kod prezentacije koju predstavlja predavač, potrebno je da on bude spremjan na ponovno objašnjavanje, dodatne sugestije, odgovaranje na pitanja slušaoca i sl.

PROCENA OSTVARENIH REZULTATA I DOPUNA

PowerPoint slajdovi treba da sadrže kratke, koncizne, opisne faze, koje će onom ko prezentira pomoći da se seti detalja prezentacije

Vrlo je verovatno da ćete, makar tokom studiranja na Univerzitetu Metropolitan, biti u prilici da pripremate više prezentacija. Zato, imajte na umu ove preporuke. Razmotrite sledeća pitanja: - izbor boje i dizajn slajdova, - mogućnost da svi slušaoci pravilno pročitaju dati tekst, - dužina trajanja izlaganja, - da li ima slajdova koje je bolje izbaciti i sl. PowerPoint je aplikacija koja dozvoljava kreiranje, štampanje i predstavljanje prezentacija. Jedna od najvažnijih preporuka prilikom kreiranja prezentacije je da ne treba razumišljati da se na slajdove postave sve informacije koje treba prezentovati. PowerPoint slajdovi treba da sadrže kratke, koncizne, opisne faze, koje će onom ko prezentira pomoći da se seti detalja prezentacije.

✓ Poglavlje 8

Zadaci za samostalni rad: PowerPoint prezentacija

ZADATAK ZA SAMOSTALNI RAD

Kreiranje power point prezentacije uz primenu obrade slika

Predviđeno vreme izrade zadatka je 30 minuta.

Zadatak

Napraviti Power point prezentaciju koja će imati tri slajda i predefenisanu pozadinu koja je ista na svim slajdovima (Opcija “Format background”):

- Prvi slajd je uvodni i na njemu treba da piše naslov prezentacije (npr. IT101-Power point prezentacija), Vaše ime i prezime, broj indeksa i godina studija
- Drugi slajd treba da sadrži grafikon sa naslovom “Prodaja za period Januar-Mart 2017.godine” u kome je grafički prikazana prodaja tri artikla za navedeni period (Opcija: Insert->Chart).

	Januar	Februar	Mart	Ukupno
Artikal 1	156	235	125	516
Artikal 2	222	333	444	999
Artikal 3	321	522	111	954
Total	699	1090	680	2469

Slika 8.1 Tabela-1 Prikaz tabele koju je potrebno uneti

- Treći slajd je završni na kome treba da piše “Hvala na pažnji” i na njemu treba da se nalazi i logo Metropolitan Univerziteta koji je grafički i vizuelno obrađen korišćenjem opcije “Format Picture”

✓ Poglavlje 9

Domaći zadatak

OPIS DOMAĆEG ZADATKA - DZ04

Kreirati PowerPoint prezentaciju

Vreme predviđeno za izradu domaćeg zadatka je 45 minuta.

Koristeći PowerPoint ili OO Express kreirati prezentaciju o neophodnim komponentama za jedan računar. U prezentaciju ubaciti slike tih komponenti i obraditi ih koristeći Internet i Adobe Photoshop (ili GIMP). Potrebno je da prezentacija sadrži minimum 20 slajdova.

Dokument snimite kao

IT101-DZ04-Ime_Prezime_brojIndeksa.zip gde su Ime, Prezime i brojIndeksa vaši podaci.

Prilikom slanja maila predmetnom profesoru ili asistentu koristiti pravila poslovne komunikacije. Ovo pravilo je potrebno usvojiti i koristiti do kraja školovanja.

▼ Zaključak

ZAKLJUČAK

U ovom predavanju smo definisali rad nekih od osnovnih ulaznih i izlaznih uređaja kao što su tastature, štampači, audio ulazni i izlazni uređaji, video ulazni i izlazni uređaji, i sl.

U sledećem video snimku pogledajte šta se koristi kao ulazno-izlazni uređaji

Top 10 Most Anticipated Tech Products of 2016

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Literatura

1. Gary B. Shelly, Misty E. Vermaat, Discovering Computers 2011-Introductory: Living in a Digital World, Cengage Learning 2010.
2. Alan Freedman, Computer Desktop Enciklopedija, McGraw Hill, 2001.
3. Gary B. Shelly, Misty E. Vermaat, Discovering Computers 2011-Introductory: Living in a Digital World, Cengage Learning 2010.



IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

OPERATIVNI SISTEMI

Lekcija 05

PRIRUČNIK ZA STUDENTE

IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

Lekcija 05

OPERATIVNI SISTEMI

- ▼ OPERATIVNI SISTEMI
- ▼ Poglavlje 1: Uloga operativnog sistema
- ▼ Poglavlje 2: Vrste operativnih sistema
- ▼ Poglavlje 3: Operativni sistemi personalnih računara
- ▼ Poglavlje 4: Operativni sistemi servera
- ▼ Poglavlje 5: Real-time operativni sistemi
- ▼ Poglavlje 6: "Embedded" operativni sistemi
- ▼ Poglavlje 7: Zastupljenost operativnih sistema na tržištu
- ▼ Poglavlje 8: Pokazna vežba: Osnove UNIX shell-a
- ▼ Poglavlje 9: Zadaci za samostalni rad: UNIX Shell skripte
- ▼ Poglavlje 10: DOMAĆI ZADATAK
- ▼ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

▼ Uvod

UVOD

Cilj ovog predavanja je da da pregled operativnih sistema, njihove osnovne funkcionalnosti, kao i kriterijume na osnovu kojih se mogu podeliti operativni sistemi

Cilj ovog predavanja je da da pregled operativnih sistema, njihove osnovne funkcionalnosti, kao i da objasni kriterijume na osnovu kojih se mogu podeliti operativni sistemi. Za svaku od kategorija biće navedeni primeri često korišćenih operativnih sistema. Kao osnovne funkcionalnosti biće obrađeno upravljanje memorijom, upravljanje uređajima, koordinacija zadataka i sl.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

✓ Poglavlje 1

Uloga operativnog sistema

KOJA JE OSNOVNA ULOGA OPERATIVNOG SISTEMA?

Operativni sistem predstavlja skup programa čija je uloga da koordinira sve zadatke računara i da upravlja aktivnostima između svih hardverskih resursa

Operativni sistem (OS) predstavlja skup programa čija je uloga da koordinira sve zadatke računara i da upravlja aktivnostima između svih hardverskih resursa. Operativni sistem ima sledeće zadatke:

- Uključivanje računara
- Isključivanje računara
- Upravljanje programima
- Upravljanje memorijom
- Upravljanje uređajima
- Koordinacija zadataka
- Praćenje performansi rada.

Kada se računar uključi ili ponovo pokrene, taj proces se zove "booting" ili pokretanje. **Pokretanje može da bude "hladno" ili "toplo" pokretanje.** Kada korisnik uključi računar koji je ugašen, to se zove hladno pokretanje (engl. *cold booting*). S druge strane, kada korisnik koristi operativni sistem da ponovo pokrene računar, to se naziva toplo pokretanje (engl. *hot booting*). Toplo pokretanje omogućava da se svi procesi pravilno zatvore, međutim to ne omogućava da se sačuvaju nesačuvane informacije. Osim toga, toplo pokretanje omogućava da se računar ponovo pokrene, a da se pri tom ne ošteti bilo koji od procesa koji trenutno rade.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

✓ 1.1 Upravljanje memorijom

ULOGA OPERATIVNOG SISTEMA U UPRAVLJANJU MEMORIJOM

Neophodno je da operativni sistem upravlja memorijom kako bi nasumični pristup memoriji mogao da bude optimizovan

Neophodno je da operativni sistem upravlja memorijom kako bi nasumični pristup memoriji (engl. **random access memory**) mogao da bude optimizovan. RAM se sastoji od jednog ili više čipova koji se nalaze na matičnoj ploči i koji sadrže instrukcije koje procesor izvršava. Zadatak operativnog sistema je da dodeli deo memorije za podatke i instrukcije koje treba izvršiti.

Upravljanje memorijom je funkcionalnost operativnog sistema koji upravlja primarnom memorijom i pomera procese između glavne memorije i diska tokom izvršenja. Upravljanje memorijom prati svaku memorijsku lokaciju, bez obzira da li je dodeljena nekim procesima ili je slobodna. Proverava koliko se memorije dodeljuje procesima. Odlučuje koji proces će dobiti memoriju i kada. Upravljanje memorijom prati kada se deo memorije oslobodi ili ne dodeli, a samim tim i ažurira status.

Kada se više programa izvršava istovremeno na sistemu, moguće je da ponestane slobodne RAM memorije. Na primer, ako operativni sistem zahteva 512MB RAM-a, antivirusni program zahteva 256MB RAM-a, veb čitač 128MB RAM-a, program za ažuriranje fotografija 512MB RAM-a, ukupna potrebna RAM memorija je 1408MB. Ako računar ima samo 1GB RAM-a, tada će operativni sistem najverovatnije koristiti virtuelnu memoriju. U takvom slučaju operativni sistem najčešće alocira deo skladišta (engl. **storage**), kao što je hard disk, da funkcioniše kao dodatni RAM. Kada se to desi, korisnik obično primeti da je računar usporen. To je zbog činjenice da je virtuelna memorija sporija od RAM-a. Deo hard diska koji se koristi kao virtuelna memorija naziva se "swap file" zbog toga što se ova funkcionalnost odnosi na razmenu informacija između memorije i skladištenja, tako da se može efikasno koristiti kao ekstenzija RAM-a. Kada se podaci razmenjuju između memorije i skladišta to nazivamo "paging". "Paging" je dug proces koji oduzima dosta vremena. Kada računar provodi puno vremena u "paging"-u to se naziva "thrashing".

▼ 1.2 Upravljanje zadacima

ULOGA OPERATIVNOG SISTEMA U UPRAVLJANJU ZADACIMA

Uloga operativnog sistema je da odredi redosled po kojem će se obavljati zadaci

Uloga operativnog sistema je da odredi redosled po kojem će se obavljati zadaci, pa je potrebno da on tim zadacima upravlja (engl. **task coordination**). Ovi zadaci mogu da obuhvataju primanja ili slanja podataka, obradu instrukcija, slanje informacija ka izlaznim uređajima, prenos podataka iz memorije i skladišta itd. Zadaci u operativnom sistemu se mogu obrađivati po principu "ko prvi dođe, prvi biva uslužen" (engl. **first-come first-served**). Međutim, ponekad kada je operativni sistem ustvari višekorisnički sistem, neki korisnici mogu imati veće prioritete i njihov zadatak će biti izvršen prvi. To se postiže podešavanjem rasporeda zadataka (engl. **schedule of tasks**).

✓ 1.3 Upravljanje uređajima

ULOGA OPERATIVNOG SISTEMA U UPRAVLJANJU UREĐAJIMA

Upravljački programi ili drajveri uređaja omogućavaju operativnom sistemu da komunicira sa određenim uređajem pomoću skupa određenih komandi

Operativni sistem komunicira sa uređajima po unapred određenom skupu pravila. Upravljački programi ili drajveri uređaja (engl. *device drivers*) omogućavaju operativnom sistemu da komunicira sa određenim uređajem pomoću skupa određenih komandi. Svaki uređaj uključujući štampač, miš, tastaturu ili monitor, ima određeni skup komandi koje se koriste da komuniciraju sa njim. **Bez ispravnih upravljačkih programa, ovi uređaji ne bi mogli pravilno da rade.** Kada se novi uređaj poveže sa računarom, njen upravljački program treba da bude instaliran pre nego što se taj uređaj može koristiti. Danas većina uređaja nudi "plug and play" opciju, što znači da operativni sistem automatski konfiguriše nove uređaje kada se oni priključe u računar. Drugim rečima, oni se automatski instaliraju na određeni uređaj.

✓ 1.4 Deljenje resursa

ULOGA OPERATIVNOG SISTEMA U DELJENJU RESURSA

Operativni sistem može biti dizajniran da podrži više korisnika na mreži dok deli neki resurs

Operativni sistem može biti dizajniran da podrži više korisnika na mreži dok deli neki resurs. Ovaj operativni sistem se naziva serverski operativni sistem. **Resursi koji se mogu deliti su hardver, softver, štampači, pristup internetu, datoteke itd.** Operativni sistem servera je odvojen od operativnih sistema na računarama klijenata koji pristupaju mreži. Klijent računari obično koriste svoje operativne sisteme posebno kada se radi van mreže. Međutim kada oni koriste resurse na mreži, oni mogu da preuzmu neke od funkcionalnosti od serverskog operativnog sistema.

✓ Poglavlje 2

Vrste operativnih sistema

KRITERIJUMI ZA PODELU OPERATIVNIH SISTEMA

Tri osnovne kategorije operativnih sistema koji su trenutno zastupljeni su personalni računari, serveri i ugnježdeni računari

Mnogi operativni sistemi su prvobitno dizajnirani da zavise od platforme za koju su pravljeni. To znači da su operativni sistemi dizajnirani samo za određeni hardver i za određenu marku računara. Danas međutim postoji tendencija da se razvijaju operativni sistemi koji su nezavisni od hardverske platforme, tako da se mogu koristiti na mnogim računarima koje prave različiti proizvođači.

Tri osnovne kategorije operativnih sistema koji su trenutno zastupljeni su:

- Personalni računari
- Server
- Ugnježdeni ili ugrađeni (engl. **embedded**)

Tabela 1 predstavlja neke od često korišćenih operativnih sistema u svakoj od kategorija.

Međutim, operativne sisteme možemo **podeliti i prema drugim kriterijumima** kao što su:

- broj procesora
- broj programa
- centralizovanost.

Kategorija	OS
Personalni računari	DOS
	Windows
	Mac OS X
	Unix
	Linux
Server	Windows Server
	Unix
	Linux
	Solaris
Ugrađeni (embedded)	Windows Embedded CE
	Windows Mobile
	Palm OS
	iPhone OS
	BlackBerry
	Google Android
	Embedded Linux
	Symbian OS

Slika 2.1.1 Tabela-1 Često korišćeni operativni sistemi

VIDEO - PODELA OPERATIVNIH SISTEMA

Prodiskutujmo kriterijume na osnovu kojih možemo podeliti operativne sisteme i navedimo tipične predstavnike za svaki kriterijum

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

✓ 2.1 Broj procesora

JEDNOPROCESORSKI I VIŠEPROCESORSKI OPERATIVNI SISTEMI

Uobičajeno je da računarski sistemi imaju jedan procesor, ali mnogi serveri i superračunari imaju dva, četiri ili više procesora

Operativni sistemi mogu biti napisani za računarske sisteme koji imaju jedan procesor ili više procesora. Uobičajeno je da računarski sistemi imaju jedan procesor, ali mnogi serveri i superračunari imaju dva, četiri ili više procesora. U tom slučaju se poslovi mogu dekomponovati na više manjih zadataka koji se izvršavaju paralelno, čime se skraćuje ukupno vreme izvršavanja posla. Zavisno od broja procesora kojim mogu da upravljaju, operativni sistemi se dele na:

- jednoprocesorske
- višeprocesorske.

Jedan od načina da se poveća brzina računara je da se napravi računarski sistem sa više procesora. U zavisnosti od toga kako su procesori povezani i šta dele, da li funkcionišu kao različiti paralelni računari, multiračunari (gde svaki procesor ima svoju memoriju) ili multiprocesori, oni zahtevaju posebne operativne sisteme koji su najčešće predstavljanju napredne serverske operativne sisteme koji obezbeđuju komunikaciju i povezivanje. Tipični predstavnici ovih operativnih sistema su Linux, Solaris, and Windows.

✓ 2.2 Broj programa

JEDNOPROGRAMSKI I VIŠEPROGRAMSKI OPERATIVNI SISTEMI

Većina današnjih operativnih sistema omogućuje da se više programa paralelno izvršava.

Prvi operativni sistemi su omogućavali izvršavanje samo jednog programa. Kada se završi izvršavanje jednog programa, počinjalo bi izvršavanje narednog. Međutim, većina današnjih operativnih sistema omogućuje da se više programa paralelno izvršava. Ali kako jedan procesor u jednom trenutku može da izvršava samo jedan program, svim programima koji se izvršavaju po određenoj proceduri dodeljuje se pravo na korišćenje računarskih resursa na određeno vreme. Zatim se, iako se prethodni program nije završio, resursi dodeljuju nekom drugom programu. Zavisno od broja programa koji paralelno mogu da se izvršavaju, operativni sistemi se dele na:

- jednoprogramske
- višeprogramske.

▼ 2.3 Centralizovanost

MEJNFREJM OPERATIVNI SISTEMI

Kod mejnfrejm računara obrada podataka je bila centralizovana, a ulaz i izlaz su distribuisani

Mejnfrejm (engl. **mainframe**) računari se razlikuju od drugih računara pre svega po velikom broju ulazno-izlaznih uređaja koje mogu paralelno da opslužuju. Ovakvi računari su uglavnom bili glavni računari velikih korporacija, kao što su banke, pošte, osiguravajuća društva, velike trgovачke kompanije, i opsluživali su na hiljade zaposlenih. Obrada podataka je bila centralizovana, a ulaz i izlaz su distribuisani po celoj korporaciji. Korisnici su na radnom mestu koristili terminale koji se sastoje od displeja i tastature. Time je omogućen ulaz i izlaz podataka od korisnika i ka njemu. Sva obrada podataka i njihovo čuvanje vrši se na mejnfrejm računaru. I danas se ovi računari koriste kao veb serveri najvećih e-commerce kompanija.

Operativni sistem mejnfrejm računara je orijentisan ka paralelnoj obradi više poslova odjednom, pri čemu većinu poslova predstavlja ogroman broj ulazno-izlaznih operacija. Ovakvi operativni sistemi tipično nude tri režima rada pri izvršavanju programa:

- paketnu obradu (engl. **batch processing**)
- transakcionu obradu (engl. **transactional processing**)
- deljenje vremena (engl. **time sharing**).

Paketna obrada je režim rada računara pri kome se izvršava samo jedan program koji nije prekidan drugim programima ili obradom ulaza-izlaza od strane korisnika sve dok se izvršavanje programa ne završi. Paketni režim rada, koji je bio svojstven računarima 2. i 3. generacije, se uobičajeno koristio kada je bilo potrebno izvršiti veliku obradu podataka, kao što je na primer izrada godišnjeg izveštaja neke velike kompanije. Ovakva obrada se često radila noću kada korisnici ne koriste sistem za unos podataka.

Transakcionalna obrada se koristi u slučajevima kada je potrebno obraditi ogroman broj malih zahteva kao što su provera stanja na računu ili rezervacija avionskih karata. Svaki od ovih zadataka je mali, ali je potrebno obraditi ih na hiljade u sekundi.

Režim rada sa deljenjem vremena omogućuje da više udaljenih korisnika istovremeno izvršava svoje aplikacije na istom računaru, uz lažno uverenje da je računar dodeljen samo njima. Realno procesor sukcesivno posvećuje svakom aktivnom programu delić procesorskog vremena i tako deleći ukupno vreme istovremeno zadovoljava potrebe svih aktivnih korisnika.

Tipični predstavnici ovih operativnih sistema su OS/360 i njegov naslednik OS/390.

MEJNFREJM I KLAUD

Način mejnfrejm rada podseća na rad klaud tehnologije

Mejnfrejm računari se više ne koriste, međutim, neophodno je da se obradi ovaj koncept, kako on predstavlja osnovu za klaud. Ideja je da mnogo korisnika može da radi na jeftinijim stanicama koji istovremeno štede na resursima i koriste resurse jednog "većeg računara." Klaud se smatra novom idejom, ali on proističe iz ideje mejnfrejm računara, iako njegova tehnologija i način rada nisu identični.

Operativni sistem klad računara je tip operativnog sistema koji je napravljen da radi u okviru klad računara i virtualizovanog okruženja. Operativni sistem klad računara upravlja operacijama, izvršavanjem procesa na virtuelnoj mašini, virtuelnim serverima i virtuelnoj infrastrukturi, kao i da upravlja njegovim hardverskim i softverskim resursima. Ovaj operativni sistem se može nazvati i virtuelni operativni sistem.

Neki od open source rešenja za operativne sisteme klad računara su:

- JoliCloud - dostupan je na GitHub i tvrdi da je trenutno pokrenuti na više od 2 miliona uređaja. Može se instalirati i na samom računaru, tako da se može koristiti i kad računar nije povezan na internet
- Silve OS - odaje utisak da se koristi Windows OS, po svom izgledu. Trenutno sadrži aplikacije kao što su File Explorer, Internet Browser, Video Player, Rich Text editor, RSS čitač, Twitter, Flickr, YouTube, IM i neke igre. Omogućava da se instaliraju eksterni softveri, ali podrazumeva da su rađeni u Silverlight

▼ Poglavlje 3

Operativni sistemi personalnih računara

SAMOSTALNI OPERATIVNI SISTEM

Samostalni operativni sistem je operativni sistem koji radi na desktop računarima, laptopovima i mobilnim računarskim uređajima

Samostalni operativni sistem je operativni sistem koji radi na desktop računarima, laptopovima i mobilnim računarskim uređajima. Samostalni operativni sistemi se takođe zovu i **klijent operativni sistemi** ili **operativni sistemi personalnih računara**. Klijentski operativni sistemi mogu da rade sa ili bez mreže. Neki od primera ovih operativnih sistema koji se najčešće koriste dati su u Tabeli 1. Tipični predstavnici ovih OS koji se danas koriste su Windows, Mac OS X, UNIX i Linux.

Personalni računari	DOS Windows (Windows 95, Windows 98, Windows 2000, Windows ME, Windows NT, Windows XP, Windows Vista, Windows 7, Windows 8, Windows 10)
	Mac OS X (10.0 Cheetah, 10.1 Puma, 10.2 Jaguar, 10.2 Panther, 10.4 Tiger, 10.5 Leopard, 10.6 Snow Leopard, 10.7 Lion, 10.8 Mountain Lion, 10.9 Mavericks, 10.10 Yosemite, 10.11 El Capitan)
	Unix
	Linux

Slika 3.1 Tabela-1 Primer operativnih sistema personalnih računara

SERIJA WINDOWS OPERATIVNIH SISTEMA (1980-2004)

Najuspešnija verzija Windows OS-a u svom vremenu je bio Windows 95 sa plug and play tehnologijom i podrškom za umrežavanje

Sredinom 1980-tih Microsoft je razvio prvu verziju **Windows operativnog sistema**. Prve dve verzije operativnog sistema Windows nisu bile popularne. Njihov naslednik Windows 3.0 je imao primjenjen je grafički korisnički interfejs (GUI), a kao takav je objavljen 1990. Tri godine kasnije, Windows NT 3.1 je pušten u korišćenje, a njegova glavna funkcija je predstavljala da radi kao operativni sistem na strani klijenta tako da može da se poveže sa Windows NT Server OS. GUI Windows NT 3.1 je ostao sličan GUI-u Windows 3.0. Godine

1995. objavljen je Windows 95. Ova verzija Windows-a je bio najuspešniji operativni sistem u to vreme. Windows 95 je prešao iz 16-bitne na 32-bitnu arhitekturu, implementirao je „Plug and Play“ tehnologiju, obezbedio podršku za umrežavanje, i bio pravi multitasking operativni sistem. Nadogradnja Windows 95 je objavljen 1998. godine kao Windows 98. Ova verzija operativnog sistema Windows koristila je aplikacije kao što su Internet Explorer, Outlook Express, Windows Address Book, FrontPage Express, Microsoft Chat, Personal Web Server, itd. Osim toga poboljšanja koje je uveo Windows 98 je bio bolji sistem upravljanja datotekama, podrška za multimedijalne tehnologije (DVD i slično), kao i USB povezanost. Nadogradnja Windows 98 je bio Windows Millennium Edition (ME), koji je dizajniran za kućne korisnike koji su želeli da budu u stanju da puštaju muziku, kreiraju i modifikuju video zapise, što je bilo omogućeno sa softverom kao što je Windows Media Player 7 i Windows Movie Maker. To je bio poslednji operativni sistem Windows serije 9.x.

U istoj godini kada je Windows ME objavljen, inicijativa za Windows 2000 Professional je pokrenuta. Četiri verzije operativnog sistema Windows 2000 Professional su objavljene: Professional, Server, Advanced Server i Datacenter Server. Dok je svako od ovih verzija bilo namenjeno drugom tržištu, sva četiri su imala iste osnovne funkcionalnosti kao što su sertifikovani drajveri, brže performanse, adaptivni Start meni, proširene funkcionalnosti za mobilne korisnike, itd. Nadogradnja na Windows Millenium je bio Windows XP Home Edition, a nadogradnja za Windows 2000 Professional je Windows XP Professional. Windows XP je objavljen 2001-e, dok je Service Pack 2 za Windows XP je objavljen 2004. Service Pack 3 je pušten na tržište godine 2008. U svim izdanjima XP-a, interfejs i performanse su poboljšani, ugrađena je bolja bezbednost, poboljšan je firewall, obezbeđeno je automatsko blokiranje pop-up reklama, i omogućena je kompatibilnost sa više aplikativnog softvera.

SERIJA WINDOWS OPERATIVNIH SISTEMA (2007-2015)

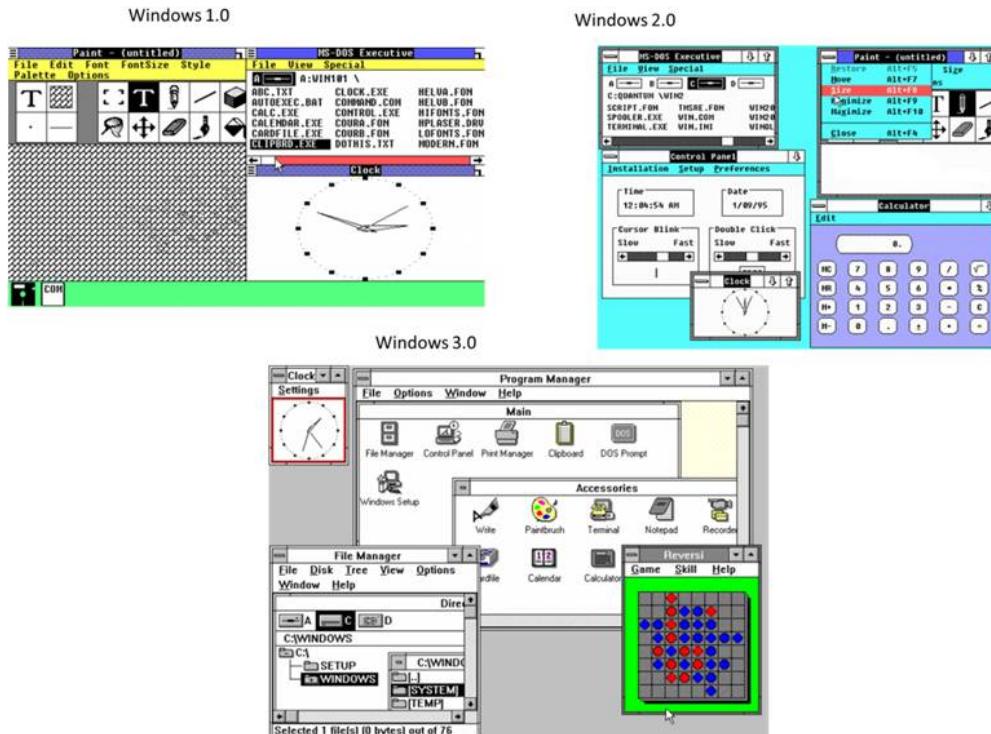
Istorijski prikaz objavljivanje Windows OS od Windows Viste do Windows 10

Godine 2007-me Windows Vista je objavljen širom sveta, dok je njegov naslednik Windows 7 objavljen dve godine kasnije. Windows Vista je pojačan sa mnogim novim karakteristikama i funkcionalnostima, kao što su inovirani GUI i vizuelni stil, multimedijalni alati za Windows DVD Maker, pojednostavljene tehnike za prilagođavanje, poboljšana administracija za korisničke naloge, poboljšane mogućnosti za pretraživanje, poboljšana podrška za uređaje i drajvere, redizajnirano umrežavanje, audio, štampanje i prikaz podsistema. Windows 7 je namenjen za upotrebu na ličnim računarima. Neki od poboljšanja koja su uključena u Windows 7 su poboljšanja prepoznavanja dodira i rukopisa, podrška za virtuelne hard diskove, poboljšani kernel, podrška za sisteme koji koriste višestruke grafičke kartice od različitih proizvođača, poboljšanje medijskih funkcionalnosti, pristup programima kroz Windows Live Veb sajt i sl. Kontrolna tabla (**Control Panel**) je poboljšana sa dodatnim stavkama kao što su ClearType Text Tuner, Credential Manager, Location Sensor, Display Color Calibration Wizard, itd. Sledeća verzija u nizu Windows operativnih sistema je Windows 8. Razvoj Windowsa 8 počelo pre razvoja Windowsa 7. Windows 8 je objavljen 2012 godine, a predstavlja deo porodice Windows NT operativnih sistema. Windows 8 je napravljen u nadi da će unaprediti korisničko iskustvo u radu na tablet računarima, te je tako ušao u trku sa Androidom i iOS-om. Iako je

ovaj OS prihvaćen pozitivno u pravcu unapređenja performansi, naročito za „*touch screen*“, sam GUI operativnog sistema je primio više kritika u pravcu teškog učenja i prilagođavanja korisnika na njegov novi izgled. Godinu dana nakon objave Windows 8 operativnog sistema, Microsoft objavljuje njegovo unapređenje pod nazivom Windows 8.1. Iako bolje prihvaćen nego njegov prethodnik, ovaj operativni sistem i dalje prima kritike za njegovu lošu integraciju korisničkog interfejsa. U julu 2015, Windows 8.1 je imao najveću zastupljenost na tržištu od perioda publikovanja Windows 7 operativnog sistema, sa zastupljenosti od 13,1%. Windows 10 operativni sistem je pušten u beta testiranje 2014 godine. Kako bi popularizovao njegovo korišćenje Microsoft je najavio da svi korisnici koji imaju legalno kupljenu verziju Windows 7 i 8.1, u prvih godinu dana mogu dobiti besplatno Windows 10 kopiju operativnog sistema. Microsoft je opisao ovaj operativni sistem kao „*operating system as a service*“.

GUI WINDOWS OPERATIVNIH SISTEMA – 1, 2 i 3

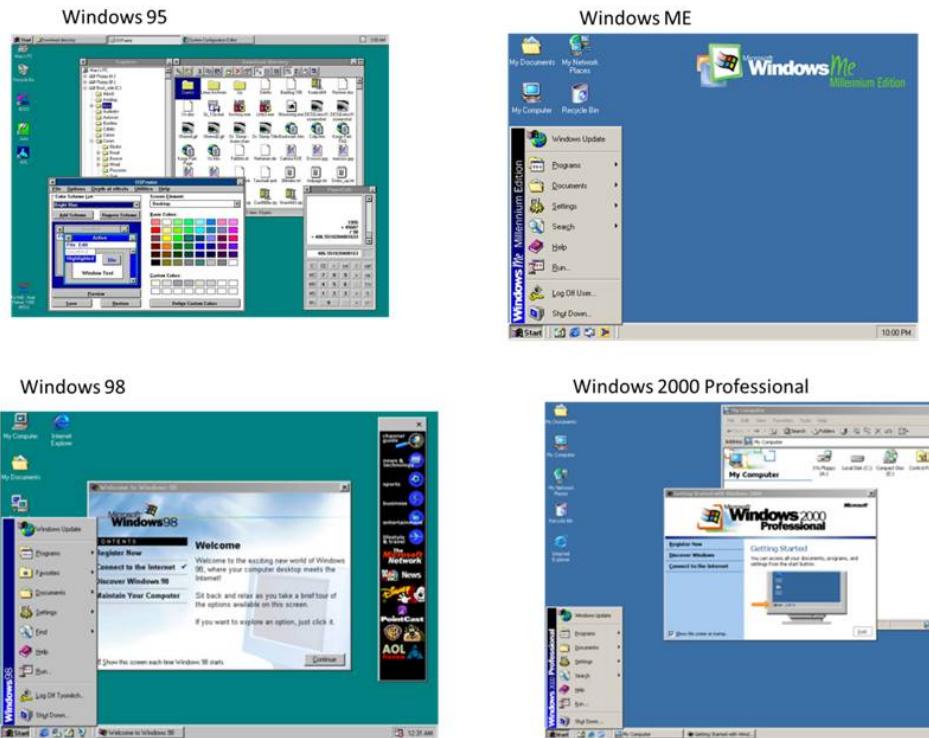
Prikaz GUI Windows 1.0, Windows 2.0, Windows 3.0



Slika 3.2 Primeri GUI Windows 1, 2 i 3

GUI WINDOWS OPERATIVNIH SISTEMA – 95, 98, ME i 2000 PRO

Prikaz GUI-a Windows 95, Windows ME, Windows 98 i Windows 2000 Professional



Slika 3.3 Primer GUI-a Windows 95, Windows ME, Windows 98 i Windows 2000 Professional

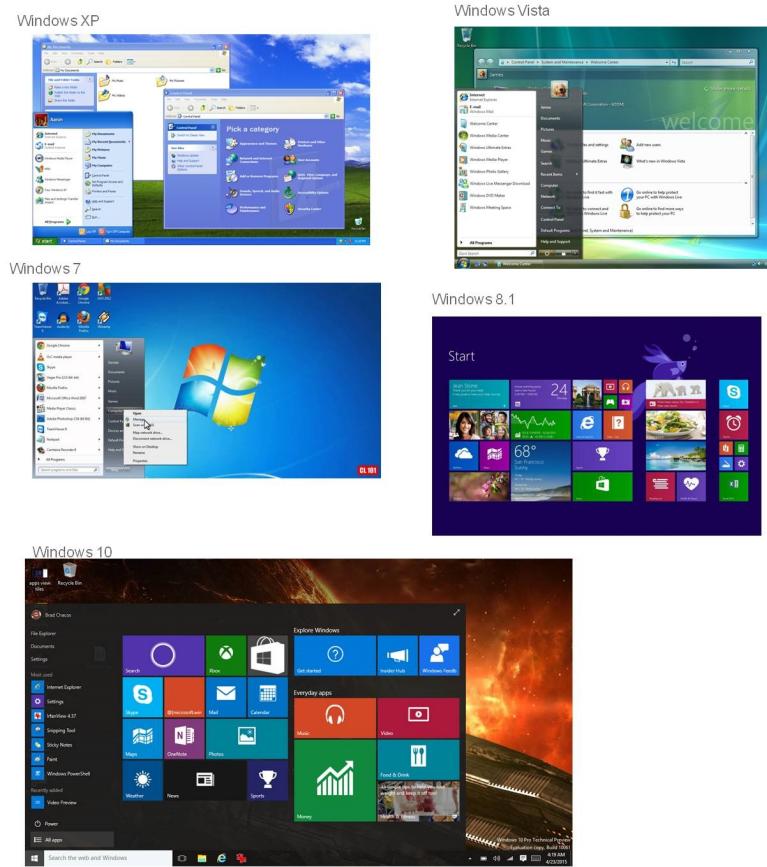
ANEGDOTA PRILIKOM PROMOCIJE WINDOWS 98

Bill Gates, Windows 98, Blue Screen of Death

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

**GUI WINDOWS OPERATIVNIH SISTEMA -XP , VISTA,
7, 8.1, 10**

*Prikaz GUI-a Windows XP, Windows Vista, Windows 7, Windows 8.1,
Windows 10*



Slika 3.4 Primer GUI-a Windows XP, Windows Vista, Windows 7, Windows 8.1, Windows 10

DODATNI MATERIJALI

Dodatni materijali o istoriji razvoja Windows operativnih sistema

Za dodatne informacije o istoriji Windows operativnog sistema, pogledajte istorijat koji je prikazan na Microsoft-ovom sajtu

<https://www.businessapac.com/history-of-windows-from-ms-dos-windows-1-to-windows-10/>

Kao i video ***The History of Microsoft Windows***

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

MAC OS X

Mac OS X je operativni sistem koji je zasnovan na UNIX-u

Macintosh operativni sistemi su postavili standarde za sve operativne sisteme od objavljuvanja Macintosh računara 1984-te godine. Macintosh operativni sistem je bio prvi operativni sistem koji je koristio grafički korisnički interfejs i bio je uzor za druge operativne sisteme od tada. Najnovija verzija operativnog sistema Macintosh Mac OS X je multitasking

operativni sistem i koristi se samo za računare koji su proizvedeni od strane Apple-a. Osim toga, Mac OS X je instaliran na svim računarima koje proizvodi Apple. Mac OS X je operativni sistem koji je zasnovan na UNIX-u. Kada su proizvedeni Intel Mac računari, specijalizovana verzija operativnog sistema pod nazivom Mac OS X 10.4 "Tigar" je razvijena za njih. Kasnija verzija Mac OS X 10.5 "Leopard" je dizajnirana da podrži oba PowerPC i Intel Mac računare. Sledеća verzija Mac OS X 10.6 „Snow Leopard“ je dizajnirana da podrži 32-bitne procesore i da radi na 64-bitnim Intel procesorima. Mac OS X 10.7 „Lion“ objavljen 2011, Mac OS X 10.8 „Mountain Lion“ objavljen 2012, sledeći u nizu planirani Mac OS X je bio 10.9 „Maverick“, a za njim i 10.10 „Yosemite“ i 10.11 „El Capitan.“ Neke od karakteristika Mac OS X su funkcije koje su zadržale od prethodnih verzija Mac OS, kao što su kvalitet fotografija, podrška za umrežavanja, poboljšano prepoznavanje govora, napredne multimedijalne mogućnosti, itd. Neke od karakteristika su dodatno razvijene za OS X u cilju poboljšanja performanse i iskustva korisnika: lakše navigacije, bolji GUI, nova tehnologija pretraživanja, poboljšani utilities programi, poboljšani softver za multimedijalne sadržaje (posmatranje i montaža), 3D video konferencije, itd.

MAC OS X LEOPARD - PRIMER

Prikaz GUI-a Mac OS X Leopard



Slika 3.5 Mac OS X Leopard

ISTORIJSKI RAZVOJ MAC OS-A

Pogledajte video koji daje prikaz istorijskog razvoja Mac OS-a

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

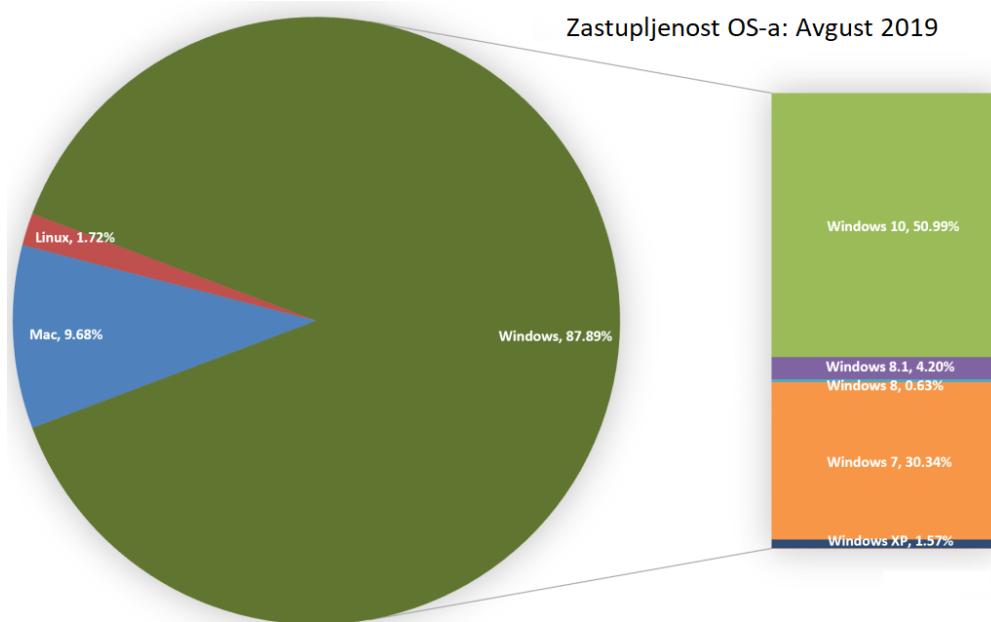
UNIX I LINUX

Sisteme koji su zasnovani na UNIX operativnom sistemu nazivamo Unix-like operativni sistemi

UNIX je prvo bitno razvijen 1969-te godine od strane Bell Labs. Prvobitno, UNIX je bio najzastupljeniji u akademskim krugovima, ali sada se UNIX koristi za servere, radne stanice i mobilne uređaje. Neke verzije UNIX-a imaju samo terminal sa komandnom linijom, dok novije verzije UNIX-a imaju OS sa GUI-em. Još jedna popularan operativni sistem koji je zasnovan na UNIX-u je **Linux**. Linux je open-source operativni sistem. Najznačajnija komponenta Linux-a je njegovo jezgro. Većina Linux verzija podržava mnoge programske jezike. Razvojni alati koji se koriste za izgradnju programa i operativnih sistema su sadržani u GNU toolchains, kao što je **GNU Compiler Collection** (GCC) i **GNU Build System**. Linux se najčešće koristi za desktop i server računare, ali postoji i specijalizovana verzija za „embedded“ sisteme.

ZASTUPLJENOST OPERATIVNIH SISTEMA

Zastupljenost OS-a - avgust 2019. godine



Slika 3.6 Zastupljenost operativnih sistema 2019. godina

▼ Poglavlje 4

Operativni sistemi servera

OPERATIVNI SISTEMI NAMENJENI ZA SERVERE

Tipični operativni sistemi servera su Windows Server, UNIX, Linux and Solaris

Serveri su računari vezani na mrežu s ciljem opsluživanja više korisnika paralelno. Uobičajeno, udaljeni korisnici imaju svoj personalni računar, koji je takođe vezan na mrežu, preko koga pristupaju serveru.

Serveri su specijalizovani da nude jednu ili, ređe, više usluga ostalim korisnicima u mreži koji imaju pravo korišćenja tih usluga. Tipični serveri su:

- Application server
- Database server
- File server
- Intranet server
- Mail server
- Network access server
- Print Server
- Proxy server
- Remote access server
- Web server

Tipični operativni sistemi servera su Windows Server, UNIX, Linux and Solaris.

Server	Windows Server (Windows NT Server, Windows Server 2000, Windows Server 2003, Windows Server 2008, Windows Server 2012, Windows Server 2012 R2) Unix Linux Solaris
--------	--

Slika 4.1 Tabela-1 Primer operativnih sistema servera

PRIMERI OPERATIVNIH SISTEMA ZA KOMPANIJE

Primeri OS za poslovanje su MS Windows server, Ubuntu, CentOS, Red Hat Enterprise Linux

Trenutno postoji više stotina Linux distribucija. Svi operativni sistemi ovih servera nude povećanu sigurnost i brze **cloud-ready** alate što ih čini idealnim za poslovne servere.

Microsoft Windows Server - Microsoft i dalje ima jedan od najboljih operativnih sistema za poslovne i lične servere. Microsoft Windows Server je razvijen iz Windows 10 OS-a i namenski je dizajniran tako da ponudi sve što je potrebno za kreiranje servera. Kao i većina njihovih proizvoda, lako se instalira. Microsoft Windows Server je, takođe, **cloud-ready** operativni sistem, što ih čini lakšim za korišćenje nego većina servera baziranih na Linux-u. Lako je lakši za korišćenje od Linux servera, i dalje postoji prostora za učenje, većina je već upoznata sa Windows sistemima, ali nisu potpuno upoznati sa Windows Server sistemima.

Ubuntu Server - Ubuntu server je **user-friendly**. Siguran je, brz i ekonomičan. Takođe, podržava većinu postojećeg softvera i hardvera. Obzirom na reputaciju, Ubuntu serveri su pouzdan izbor za poslovne servere.

CentOS Server - Još jedna poznata distribucija je CentOS. Kao i Fedora, baziran je na Red Hat-u, a za razliku od Fedore, kreiran je više za korisnike koji koriste openSUSE. Budući da koristi istu instalacijski program kao Fedora, lako se instalira i postoji dobar izbor aplikacija. Kao i većina distribucija, CentOS je besplatan i predstavlja zajednicu sa grupom saradnika i korisnika koji rade na razvoju novih aplikacija i projekata.

Red Hat Enterprise Linux Server - Red Hat Enterprise Linux je jedna od vodećih svetskih softverskih kompanija u Americi i poznata je po dobro poznatoj Linux distribuciji. Pruža **open source** softverska rešenja i proizvode za različite industrije. Posebno dizajniran za biznise, RHEL se koristi za servere ili desktop računare. Redhat i Fedora su najviše korišćene Linux distribucije.

Unix Server - Ovi serveri daju prednosti za **e-commerce** biznise. Unix Server OS i Unix Dedicated serveri su pouzdani i sigurni. Mogu se koristiti kao multitasking OS, time-sharing OS za više korisnika, enkripciju fajlova i šifri i drugo. Kao i Linux sistemi, može da se prilagodi specifikacijama korisnika.

▼ Poglavlje 5

Real-time operativni sistemi

OSNOVNE KARAKTERISTIKE REAL-TIME OPERATIVNIH SISTEMA

Glavna karakteristika dobrog real-time sistema je da isporučuje odgovor na dati zahtev "blagovremeno", a da pri tom održe konzistenciju funkcionalnosti i efikasnosti

Postoje mnoge aplikacije koje se izvršavaju na računarskim sistemima koje ne dozvoljavaju kašnjenje odziva računara u odnosu na neki događaj, dakle zahtevaju odziv u eksplisitno definisanom ograničenom realnom vremenu (engl. **real time**). **Glavna karakteristika dobrog real-time sistema je da isporučuje odgovor na dati zahtev "blagovremeno", a da pri tom održe konzistenciju funkcionalnosti i efikasnosti.** Neophodno je da vreme odziva sistema uvek bude u dozvoljenom vremenskom intervalu, tako da se ne čeka uvek različito na odgovor. U zavisnosti od primene, odgovor na zahtev je možda neophodan odmah. Primer za ovakav slučaj je kada hirurg obavlja operaciju na pacijentu koristeći neki uređaj telemedicine. Ako akcija apsolutno mora da se dogodi u nekom vremenskom intervalu, govori se o **čvrstim real-time operativnim sistemima**. Tipičan primer je upravljanje robotima. Ako akcija treba da se izvrši u realnom vremenu, ali je prihvatljivo i malo zakašnjenje, govorimo o **mekim real-time operativnim sistemima**. Dakle, kod ovakvih sistema performanse sistema mogu da budu degradirane, al ne i ugrožene odzivom koji je duži od zadatog. Ovo je slučaj sa sistemima koji podržavaju audio ili multimedijalne aplikacije.

Tipični predstavnici real-time operativnih sistema su LynxOS, OSE, Enea OSE (**Operating System Embedded**), QNX, RTLinux (**hard real-time OS**), VxWorks i Windows CE.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

PRIMENA REAL-TIME OPERATIVNIH SISTEMA

Real-time operativni sistemi se koriste u različitim oblastima

Real-time sistemi se koriste:

- rezervacionim sistemima za avio karte,
- kontrolu avio saobraćaja,
- internet telefoniji,
- u svim sistemima koji pružaju najnovije informacije o cenama akcija,

- srčani pejsmejker,
- ABS sistem

▼ Poglavlje 6

“Embedded” operativni sistemi

UGRAĐENI OPERATIVNI SISTEM

Operativni sistemi koji se koriste na mobilnim uređajima ili bilo kojim drugim računarskim uređajima se zove ugrađeni operativni sistem

Operativni sistemi koji se koriste na mobilnim uređajima ili bilo kojim drugim računarskim uređajima (Slika 1) se zove ugrađeni operativni sistem (engl. **embedded operating system**). „Embedded“ operativni sistemi imaju neke karakteristike real-time operativnih sistema, ali se odnose na specifičnosti kapaciteta, snage, korisničkog interfejsa i još mnogo toga.

Ugrađeni (ugnježdeni) sistem je sistem sa posebnom namenom, koji je u potpunosti određen za namenu uređaja koji kontroliše. Za razliku od opšte namene računara, kao što su personalni računari, ugnježdeni sistem obavlja prethodno definisane zadatke, obično sa vrlo specifičnim zahtevima. Budući da je sistem posvećen određenom zadatku, projektanti ovih sistema mogu da optimizuju, kao i da smanjuju veličinu i cenu proizvoda. Ugrađeni sistemi se često masovno proizvode, tako da ušteda može biti i u rangu nekoliko miliona.

Neki primeri ugrađenih sistema su bankomati, mobilni telefoni, štampači, termostati, kalkulatori i konzole za video igre. Ručni računari (engl. **handheld computers**) ili PDA se takođe smatraju da spadaju u grupu ugrađenih uređaja zbog prirode dizajna svog hardvera, iako se više mogu proširiti u softverskom smislu. Kako se proširuje ovaj spisak uređaja, tako i nestaje jasna granica šta su tačno ugrađeni sistemi, s obzirom da su moderniji uređaji pravljeni kao višenamenski.



Slika 6.1 Popularni "embedded" OS

Oblast istraživanja koja se bavi ugrađenim sistemima je bogata sa potencijalom jer kombinuje dva faktora. Prvo, dizajner sistema obično ima kontrolu nad dizajnom i hardvera i softvera, za razliku u slučaju računara koji su opšte namene. Drugo, ugrađeni sistemi se grade na širokom spektru disciplina, uključujući i arhitekturu računara (arhitektura procesora i mikroarhitekture, dizajn memorije sistema), kompjuler, scheduler operativnog sistema i real-time sistema. Kombinacija ova dva faktora omogućava synergiju između više oblasti i dovodi do optimizacije proizvoda.

WINDOWS EMBEDDED CE

Windows Embedded CE može da se koristi za uređaje kao što su kućni aparati, konzole za igre, smart telefoni, industrijsko upravljanje, šivaće mašine

Windows Embedded CE operativni sistem je smanjena verzija Windows operativnog sistema, tako da može radi na uređajima koji imaju ograničene mogućnosti. Grafički korisnički interfejs za Windows Embedded CE podržava boje, multimedije, e-mail, Internet, pretraživanje Interneta, omogućava pregledanje datoteka za Word, Excel, Power Point, itd. Windows Embedded CE može da se koristi za uređaje kao što su kućni aparati, konzole za igre, smart telefoni, industrijsko upravljanje, šivaće mašine (Slika 2), itd.



Slika 6.2 Primer šivaće mašine sa Windows CE

WINDOWS MOBILE, IOS, ANDROID

Android je zasnovan Linux-u i razvijen je od strane Open Handset Alliance, koji predvodi Google

Windows Mobile operativni sistem zasnovan je na karakteristikama sistema Windows Embedded CE operativnog sistema. Windows Mobile je specijalno dizajniran da radi na određenim mobilnim uređajima i PDA-evima koji eksplisitno koriste „**touchscreen**“ tehnologiju. Mobilni uređaji koji ne koriste „**touchscreen**“, koriste Windows Mobile Standard. Windows Mobile ima više funkcionalnosti nego Windows Embedded CE i može da se sinhronizuje sa računarom, bilo žično ili bežično.

iOS operativni sistem je razvijen od strane Apple-a kako bi se koristio na Apple-ovim iPhone, iPad i iPod touch uređajima. Ovi uređaji prepoznaju više dodirnih tačaka na svom interfejsu tako da se nazivaju „**multi-touch**“ uređaji. Ova funkcionalnost omogućava korišćenje ručne gesture koja je efikasnija u poboljšanju interakcije između čoveka i računara. Najčešće se koriste gesturi poput dodirivanja tastera na ekranu, pomeranja prsta preko ekrana kako bi se prešlo sa jedne strane na drugu, ili jedne aplikacije na drugu, ili čak prevlačenje objekata. Takođe postoje specijalne gesturi za zumiranje i umanjivanje.

Android je operativni sistem koji postaje sve popularniji na tržištu. Android OS se instalira na smart telefone i mobilne uređaje kao što su tablet računari. Android je zasnovan Linux-u i razvijen je od strane Open Handset Alliance, koji predvodi Google. Google je objavio Android kao open source, pod Apache licencom. Pošto je open source, zajednica programera koja kreira mobilne aplikacije za Android platforme postaje sve veća. Neke od karakteristika koje podržava Android su skladištenje, razmena poruka, pretraživanje interneta, Java podrška, **multi-touch**, video pozivi, **multitasking**, omogućava dodatni hardver, itd.

ANEGDOTA PRILIKOM DEMOA IPHONE-A

CNET News: Steve Jobs' demo fail

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

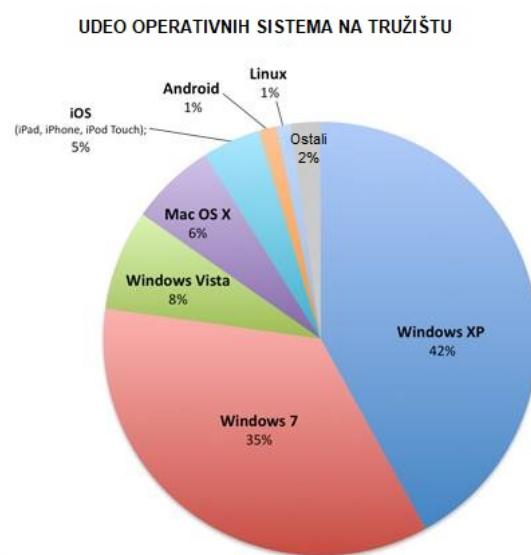
▼ Poglavlje 7

Zastupljenost operativnih sistema na tržištu

ZASTUPLJENOST OS-A 2011 I 2012 GODINE

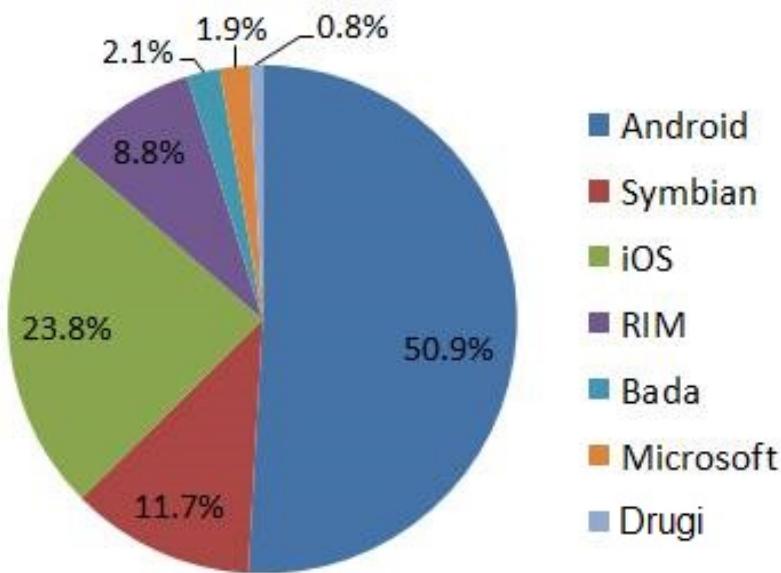
2012-te godine i dalje dominira Windows XP za personalne računare, a 2011-te za smart telefone primat ima Android OS

Prema podacima od februara 2012 Net Market Share-a, Windows je i dalje najzastupljeniji operativni sistem. Slika 1 pokazuje udeo na tržištu za određene operativne sisteme. Može se videti da Windows XP, Windows Vista i Windows 7 obuhvataju preko 85% tržišta. Iza Windows-a su Mac OS X, iOS, Android i Linux.



Slika 7.1 Udeo operativnih sistema na tržištu 2012-te

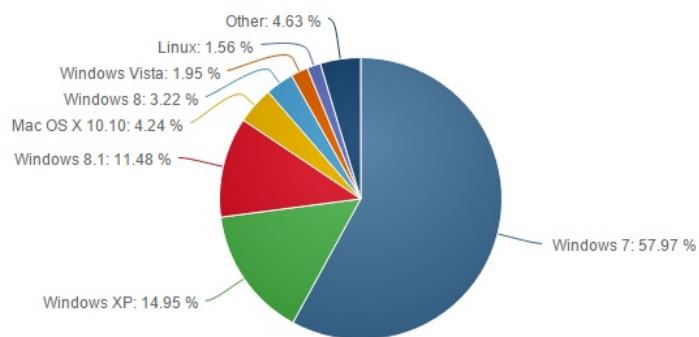
S druge strane, Gartnerova analiza iz 2011 pokazuje da u globalnom korišćenju smart telefona, Android zauzima veći deo tržišta, dok iOS i Simbian zaostaju (Slika 2).



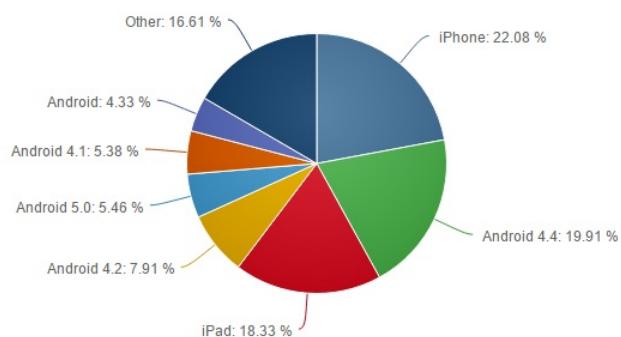
Slika 7.2 Zastupljenost operativnih sistema telefona na tržištu 2011-te

ZASTUPLJENOST OS-A U 2015 GODINI

Prikaz NetMarket Share statistike za period januar-septembar 2015



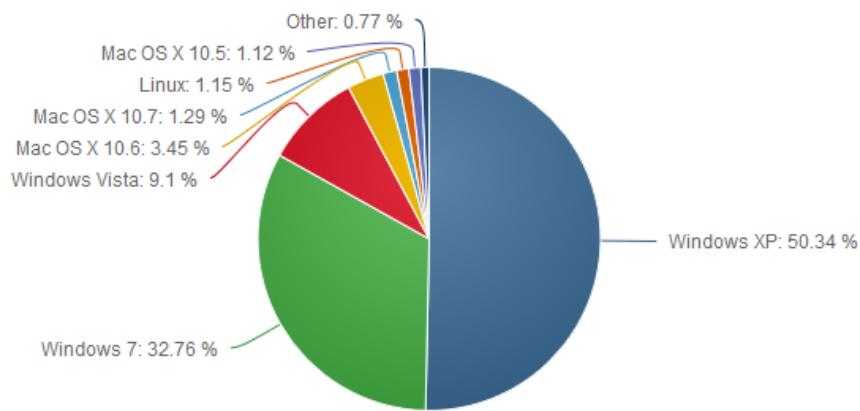
Slika 7.3 Zastupljenost OS za Desktop računare u 2015. godini



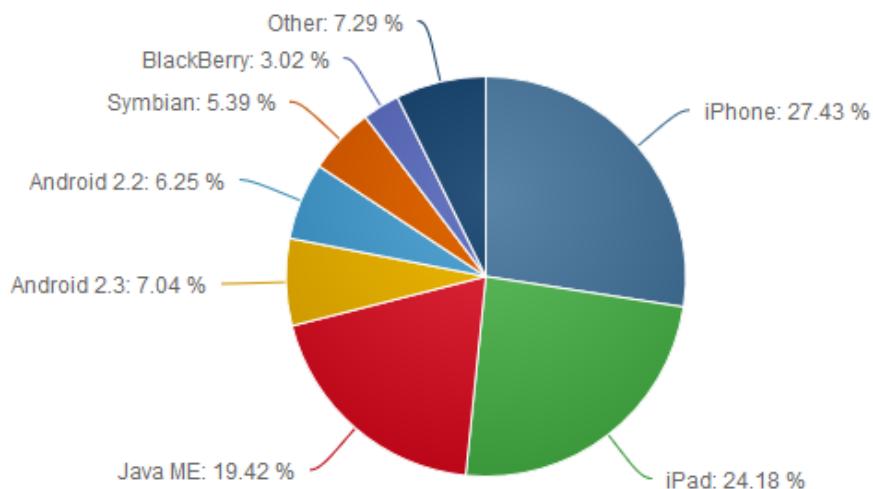
Slika 7.4 Zastupljenost OS-a za mobilne uređaje u 2015. godini

ZASTUPLJENOST OS-A U 2016 GODINI

Prikaz NetMarket Share statistike za period januar-avgust 2016



Slika 7.5 Zastupljenost OS za Desktop računare u 2016. godini



Slika 7.6 Zastupljenost OS-a za mobilne uređaje i tablete u 2016. godini

▼ Poglavlje 8

Pokazna vežba: Osnove UNIX shell-a

INSTALACIJA CYGWIN OKRUŽENJA

Koraci instalacije Cygwin okruženja

Predviđeno vreme pokaznih vežbi je 100 minuta.

Cygwin je softver koji simulira Linux terminala na Windows operativnom sistemu. Instalacija se može preuzeti sa sledeće adrese: <https://cygwin.com/install.html>.

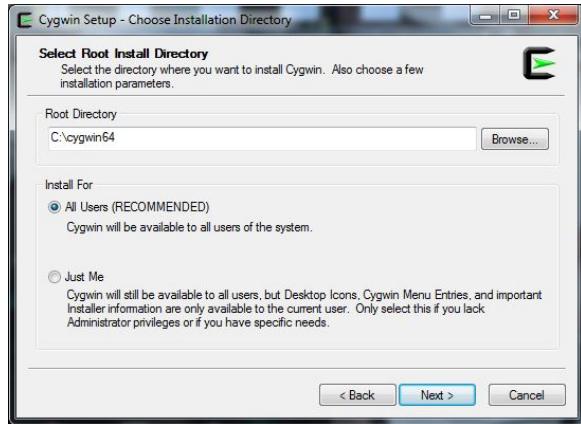
Nakon preuzimanja .exe fajla, instalacija se pokreće na uobičajeni način. Na slici 1 je prikazan početak instalacije.



Slika 8.1 Početak instalacije

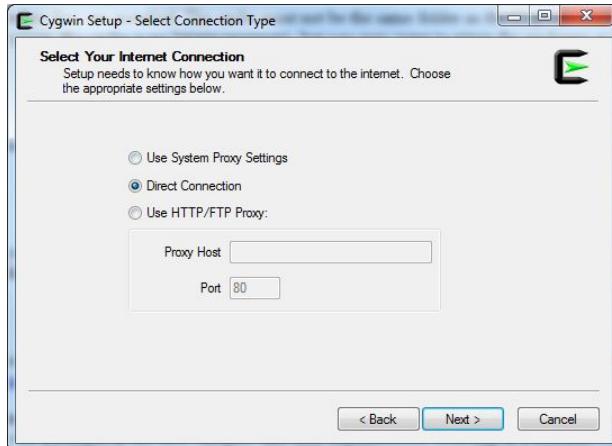


Slika 8.2 Instalacija sa interneta



Slika 8.3 Odabir lokacije instalacije

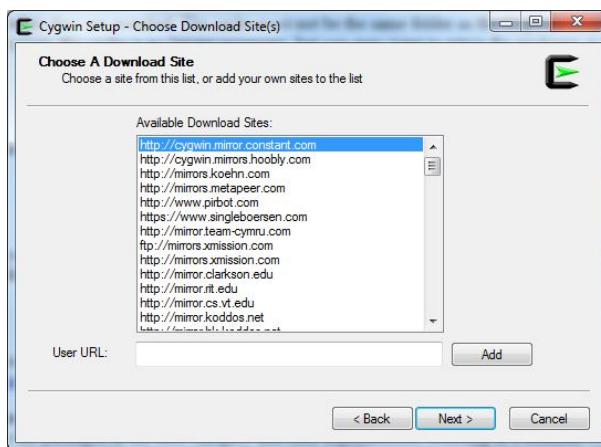
Opcija Direct connect će se automatski povezati na internet. Ukoliko je sistem podešen da koristi proxy server, može se odabrati opcija Use System Proxy Settings, a može se i ručno podesiti opcijom Use HTTP/FTP Proxy.



Slika 8.4 Odabir direktne konekcije

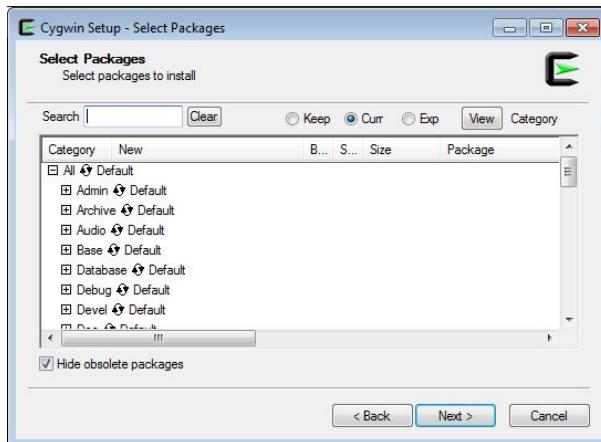
ZAVRŠETAK INSTALACIJE CYGWIN OKRUŽENJA

Odabir paketa za instalaciju i završavanje instalacije



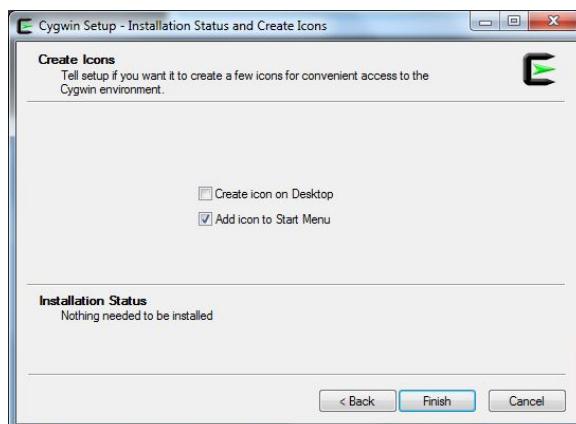
Slika 8.5 Odabir sajta za skidanje paketa

Od paketa bi se mogao čekirati nano editor, Editors->nano, kako bi kasnije tokom rada mogao da se koristi za pisanje skripti.



Slika 8.6 Odabir paketa za instalaciju

Sada se program može pokrenuti sa desktop-a ili iz startnog menija.



Slika 8.7 Kraj instalacije

SHELL

Shell je interfejs preko koga korisnici mogu da pristupe servisima operativnog sistema, tj. Kernela

Shell je interfejs preko koga korisnici mogu da pristupe servisima operativnog sistema, tj. Kernela.

Windows grafički shell ima prepoznatljiv taskbar i start meni. Za Linux OS postoje brojni grafički shell-ovi kao što su **GNOME**, **KDE** itd.

Command line interface ([CLI](#)) je korisnički interfejs u kome korisnik zadaje komande programu u formi teksta. Ovo je bio jedini način rada sa operativnim sistemima u prošlosti, dok danas većina operativnih sistema za opštu upotrebu ima grafičke interfejse. Međutim, [CLI](#) često daje znatno veću fleksibilnost i brzinu rada u odnosu na grafičke interfejse, i zato je poznavanje korišćenja ovakvog interfejsa značajno za napredne korisnike kao što su administratori sistema, programeri itd.

UNIX SHELL

Danas je najrasprostranjeniji Bourne Again Shell (bash)

Unix shell radi kao interpreter komandi. Postoji nekoliko varijanti Unix shell-a kao što su **Bourne shell**, **Korn shell**, **C shell**, a danas je najrasprostranjeniji **Bourne Again Shell ([bash](#))**. Zajedničke funkcije većine linux shell-ova su piping, variable, supstitucija komandi, strukture za kontrolu toka (grananje i petlje) itd.

Najčešći način korišćenja shell-a je kao [REPL](#) (read-eval-print-loop): korisnik unosi komandu, shell interpretira komandu i ispisuje rezultat, korisnik čita rezultat i zadaje sledeću komandu itd.

Često korišćene sekvene komandi je moguće napisati u obliku skripte, koja se može izvršiti sa ili bez ulaza od korisnika.

Skripte su veoma korisne, i obično se koriste za automatizaciju administrativnih operacija.

TERMINAL

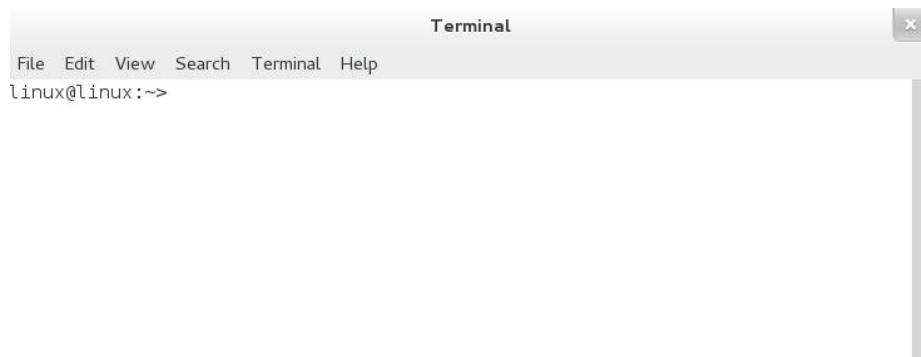
U prošlosti su terminali bili uređaji sa ekranom i tastaturom, dok se danas najčešće koriste terminal emulatori.

Komande se u shell unose pomoću terminala. U prošlosti su terminali bili uređaji sa ekranom i tastaturom, dok se danas najčešće koriste terminal emulatori.



Slika 8.8 DEC VT220 terminal

Terminal emulator je program koji emulira rad video terminala. Neki od terminal emulatora za Linux OS su xterm, gterm, konsole itd.



Slika 8.9 Terminal emulator

SHELL KOMANDE

Broj i oblik argumenata zavisi od komande tj. programa koji se izvršava.

Shell interpretira komande liniju po liniju. Komande se unose kao reči sa razmakom (**whitespace** karakter). Prva reč koja se uneće u novoj liniji se interpretira kao ime programa ili interna komanda, a ostale reči se prosleđuju kao argumenti pokrenutom programu.

Kada shell pročita naziv komande, pokušaće da je izvrši kao internu komandu, tj. komandu implementiranu u samom shell programu. Ako takva komanda ne postoji, shell će pokušati da nađe program sa unesenim imenom na fajlsistemu i pokrene ga. Shell će tražiti program u direktorijumima navedenim u PATH podešavanjima.

Opšti oblik komande je:

naziv_komande argument1 argument2 argument3 argument4 ...

Broj i oblik argumenata zavisi od komande tj. programa koji se izvršava. Po konvenciji, argumenti koji predstavljaju opcije koje modifikuju rad programa počinju sa znakom `--`. Ovakvi argumenti se nazivaju *switch*.

FAJLSISTEM PUTANJE

Fajlsistem predstavlja stablo, sa korijenom u / (root) direktorijumu.

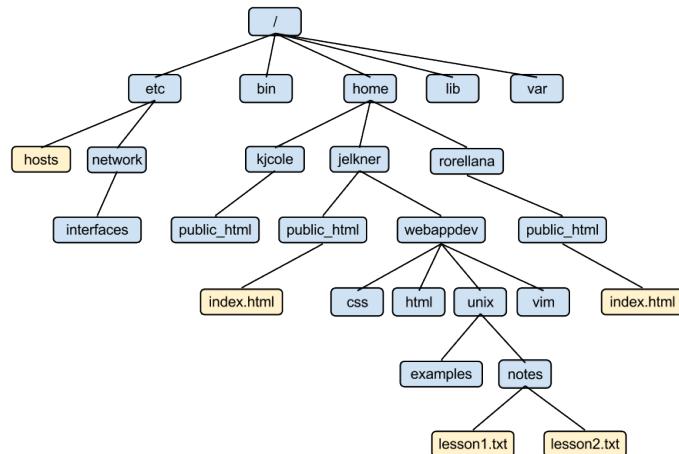
Fajlsistem putanje su hijerarhijske i sledećeg oblika `/direktorijum1/direktorijum2/direktorijum3`. Mogu biti relativne i absolutne. Apsolutne putanje ukazuju na datoteku od korjena fajlsistema, dok relativne putanje ukazuju na putanju u odnosu na radni direktorijum procesa. Svaki proces ima jedan radni direktorijum, i to je direktorijum iz koga je proces pokrenut.

Fajlsistem predstavlja stablo, sa korijenom u **(root)** direktorijumu.

Apsolutne putanje počinju sa znakom `/`.

Na primer, do direktorijuma **interfaces** sa prethodne slike moguće je doći na sledeće načine:

- Apsolutnom putanjom `/etc/network/interfaces`
- Relativnom putanjom `network/interfaces` iz `/etc` direktorijuma
- Relativnom putanjem `interfaces` iz `/etc/network` direktorijuma.



Slika 8.10 Fajlsistem stablo

HELP OPCIJA

Po konvenciji argumenti `-h` ili `-help` prikazuju uputstvo za korišćenje komande.

Po konvenciji argumenti **-h** ili **-help** prikazuju uputstvo za korišćenje komande. U uputstvu su obično označni mogući argumenti funkcije.

Komanda **man ime_funkcije**, prikazuje stranicu Unix priručnika gde je detaljno opisano ponašanje komande sa navedenim imenom.

PROMENA RADNOG DIREKTORIJUMA

Komanda cd vrši promjenu radnog direktorijuma

Komanda **pwd** ispisuje trenutni radni direktorijum.

Komanda **cd** vrši promenu radnog direktorijuma. Prima jedan argument koji predstavlja putanju novog radnog direktorijuma.

1. **cd /home/student**
2. **cd Documents**

Prva komanda vrši promenu radnog direktorijuma na **/home/student**.

Druga komanda vrši promenu radnog direktorijuma na direktorijum **Documents** koji se nalazi u trenutnom radnom direktorijumu. Ovo je relativna putanja.

LISTANJE SADRŽAJA DIREKTORIJUMA

Komanda ls izlistava sadržaj direktorijuma

Komanda **ls** bez argumenta izlistava sadržaj trenutnog direktorijuma, a može da primi putanju do direktorijuma kao argument. Najčešće korišćeni **switch** argumenti su **-l** i **-a**. **Switch -l** prikazuje dugi listing sa dodatnim informacijama o fajlovima kao što su dozvole, vlasništvo, veličina itd. **Switch -a** prikazuje i sakrivene fajlove.

ls /home/student

ls -al /etc

Switch -l prikazuje sadržaj direktorijumima u detaljnem (long) formatu. U kolonama su prikazani sledeći podaci: vrsta čvora (direktorijum ili fajl), dozvole pristupa, korisnik vlasnik, grupa vlasnik, veličina fajla, vreme poslednje modifikacije i ime fajla ili direktorijuma.

```
jpp@jpp:/boot
jpp@jpp:/boot$ ls -la
total 39132
drwxr-xr-x  3 root root    4096 2011-05-13 08:52 .
drwxr-xr-x 23 root root    4096 2011-05-04 09:27 ..
-rw-r--r--  1 root root 700761 2011-03-18 16:33 abi-2.6.35-28-generic
-rw-r--r--  1 root root 730039 2011-04-11 01:24 abi-2.6.38-8-generic
-rw-r--r--  1 root root 122616 2011-03-18 16:33 config-2.6.35-28-generic
-rw-r--r--  1 root root 130313 2011-04-11 01:24 config-2.6.38-8-generic
drwxr-xr-x  3 root root   12288 2011-05-04 09:32 grub
-rw-r--r--  1 root root 11008098 2011-04-15 08:58 initrd.img-2.6.35-28-generic
-rw-r--r--  1 root root 13134896 2011-05-13 08:52 initrd.img-2.6.38-8-generic
-rw-r--r--  1 root root 160988 2010-10-22 09:08 memtest86+.bin
-rw-r--r--  1 root root 163168 2010-10-22 09:08 memtest86+ multiboot.bin
-rw-r--r--  1 root root 2344143 2011-03-18 16:33 System.map-2.6.35-28-generic
-rw-r----- 1 root root 2654256 2011-04-11 01:24 System.map-2.6.38-8-generic
-rw-r--r--  1 root root 1336 2011-03-18 16:35 vmcoreinfo-2.6.35-28-generic
-rw-r--r--  1 root root 1368 2011-04-11 01:26 vmcoreinfo-2.6.38-8-generic
-rw-r--r--  1 root root 4342384 2011-03-18 16:33 vmlinuz-2.6.35-28-generic
-rw----- 1 root root 4523936 2011-04-11 01:24 vmlinuz-2.6.38-8-generic
jpp@jpp:/boot$
```

Slika 8.11 Izlaz ls -l komande

UNIX SISTEM DOZVOLA

Svaki direktorijum ili datoteka u UNIX sistemu je u vlasništvu jednog sistemskog korisnika i jedne grupe korisnika

U UNIX sistemu, identitet autentikovanog korisnika i koncept vlasništva su centralni elementi sistema bezbjednosti.

Svaki direktorijum ili datoteka u UNIX sistemu je u vlasništvu jednog sistemskog korisnika i jedne grupe korisnika. Na osnovu tih podataka se donose odluke o autorizaciji akcija korisnika nad tom datotekom ili direktorijumom.

Akcije koje korisnik može da izvrši nad datotekama su **read**, **write** i **execute**. Dakle, svaka datoteka ima skup od tri grupe po tri bita koji označavaju prava pristupa korisnika.

Na primjer, datoteka može imati sledeće permisije:

rwx rw- r-

Prva grupa bitova označava prava koja ima vlasnik datoteke. U ovom slučaju, vlasnik ima pravo čitanja, pisanja i izvršavanja datoteke.

Druga grupa bitova označava prava grupe koja je vlasnik datoteke. U ovom slučaju, korisnici iz grupe imaju pravo čitanja i pisanja u datoteku.

Treća grupa bitova označava prava pristupa koji imaju svi ostali korisnici.

Postoji specijalni korisnik root, koji ima potpunu kontrolu nad sistemom.

KOPIRANJE FAJLA

Komanda cp kopira fajlove i direktorijume.

Switch -r označava rekurzivno kopiranje čitavog direktorijuma.

cp /home/fajl1 /home/Documents

cp /home/fajl1 /home/fajl2 /home/Documents

cp -r Documents /var/backup

Prva komanda kopira fajl1 u **/home/Documents**, dok druga komanda kopira dva fajla u isti direktorijum. Treći primer je kopiranje kompletног direktorija.

BRISANJE FAJLOVA

Komanda rm briše fajl na putanji koja je data u argumentu

Komanda **rm** briše fajl na putanji koja je data u argumentu. Ako je putanja u argumentu putanja do direktorijuma koji nije prazan, brisanje se neće uspešno izvršiti. Za brisanje direktorijuma i svih fajlova u njemu koristi se **switch -r**.

rm fajl1

rm -r /home/Documents

ARHIVIRANJE FAJLOVA

Tar je format za arhiviranje fajlova

Tar je format za arhiviranje fajlova. Komanda **tar** upravlja tar arhivama. **Switch -f** govori da su ulazi u komandu putanje do fajlova. **Switch -t** ispisuje sadržaj arhive, **-x** ekstrahuje sadržaj arhive, a **-c** kreira arhivu.

Da bismo, na primer, arhivirali dva fajla u arhivu **ar.tar** koristimo sledeću komandu:

tar -c ar.tar fajl1 fajl2

Sadržaj kreirane arhive se štampa komandom:

tar -tf ar.tar

Sledeća komanda vrši ekstrakciju sadržaja arhive:

tar -xf ar.tar

KOMBINOVANJE PROGRAMA

Unix filozofija programa nalaže da svaki program ima samo jednu funkciju

Unix filozofija programa nalaže da svaki program ima samo jednu funkciju, i da obavlja svoju funkciju dobro. Unix daje mehanizme za kombinovanje funkcija više mali programa. Ovakav pristup je veoma fleksibilan, i daje veliki stepen kontrole korisniku.

Osim preko argumenata, programi mogu dobiti ulazne podatke direktno preko tastature, dok je proces programa aktivan. Dok se program izvršava, on preuzima sav ulaz sa termimala.

Pokretanje programa u pozadini vrši se sa operatorom &.

REDIREKCIJA ULAZA I IZLAZA

Operator > vrši redirekciju standardnog izlaza u fajl

Svi procesi dobijaju tri kanala za komunikaciju (**stream**): standardni ulaz, standardni izlaz i standardni izlaz greške (**standard error**). Ulaz je obično sa tastature, a izlaz na ekran. Sva tri kanala mogu se redirektovati.

Operator **>** vrši redirekciju standardnog izlaza u fajl. Ovo znači da izlaz koji bi program inače ispisao na ekran ide direktno u navedeni fajl. Ako fajl već postoji, ova operacija će pobrisati njegov sadržaj i upisati redirektovani izlaz. Ako želimo da sačuvamo trenutni sadržaj fajla i na njega samo dodamo izlaz, koristi se **>>**.

ls > lista.txt

U ovom primeru, izlaz komande **ls** (**lista fajlova**) se upisuje u fajl **lista.txt**.

PIPING

Unix shell takođe može redirektovati izlaz jednog programa u ulaz drugog programa

Unix shell takođe može redirektovati izlaz jednog programa u ulaz drugog programa. Ovo se zove **piping**. Neki programi su posebno napravljeni da bi se koristili na ovaj način, i nazivaju se filter programi. Za ovu vrstu redirekcije koristi se pipe operator |. Opšti oblik ove operacije je:

komadna1 | komanda2

U ovom slučaju, izlaz prve komande će biti redirektovan u ulaz druge komande.

Program **cat** ispisuje sadržaj fajla iz argumenta na standardni izlaz.

Program **grep** ispisuje samo linije sa standardnog ulaza koje odgovaraju šablonu zadatom u argumentu.

Na primer, moguće je koristiti ove dve komande zajedno na sledeći način:

cat file1.txt | grep linux

Ova komanda će ispisati sve linije iz fajla **file1.txt** koje sadrže reč **linux**.

HEAD I TAIL KOMANDE

Komande head i tail ispisuju dio sadržaja fajla od početka ili kraja, respektivno

Komande **head** i **tail** ispisuju dio sadržaja fajla od početka ili kraja, respektivno. Ove komande se mogu koristiti na isti način kao i cat komanda, s tim što ispisuju samo deo fajla.

Na primer,

tail -n 10 file.txt ispisuje poslednjih deset linija fajla file.txt.

head -c 10 file.txt ispisuje prvih deset bajtova iz fajla file.txt.

ODREDJIVANJE VELIČINE FAJLA

Komanda wc sa opcijama -c, -l, i -m ispisuje broj bajtova, linija i karaktera teksta sa standardnog ulaza.

Komanda **wc** sa opcijama **-c**, **-l**, i **-m** ispisuje broj bajtova, linija i karaktera teksta sa standardnog ulaza.

cat file1.txt | grep linux | wc -l

Ova komanda će ispisati broj linija iz fajla file1.txt koje sadrže riječ linux.

Izlaz iz prve komande **cat** je sadržaj fajla. Izlaz cat komande se prosledjuje direktno na ulaz **grep** komande, a izlaz iz **grep** komande su samo linije koje sadrže reč linux. Izlaz iz **grep** komande se prosledjuje na ulaz komande **wc**, koja ispisuje broj linija teksta na standardni izlaz. Pošto standardni izlaz ove komande nije redirektovan, broj linija se ispisuje na ekran.

SORT KOMANDA

SORT komanda se koristi za sortiranje sadržaja.

SORT komanda se koristi za sortiranje sadržaja, odnosno raspoređivanje zapisa u određenom redosledu. Po **default**-u, pravila sortiranja su sledeća:

- Linije počinju sa brojem koji se nalazi na početku svakog reda
- Linije su poređane po alfabetu
- Linije koje počinju malom slovom biće ispred onih koji počinju istim slovom ali počinju velikim slovom

Ova pravila se mogu promeniti različitim opcijama koje pruža sort komanda. Neke od opcija prikazane su u sledećoj tabeli:

-b	Ignoriše razmake na početku reda
-r	Sortira po obrnutom redosledu
-o	Ispisuje rezultat u navedeni fajl
-n	Sortira po brojevima
-R	Sortira po nasumičnom redosledu
-k, --key=POS1[,POS2]	Počinje od ključa POS1 i završava sa POS2
--help	Prikazuje pomoćnu poruku
-f	Ignoriše mala i velika početna slova

Slika 8.12 Tabela-1 Opcije komande SORT

Primer: U fajlu **data.txt** nalazi se sledeći sadržaj:

```
apples
oranges
pears
kiwis
bananas
```

Korišćenjem komande **sort data.txt**, bez dodatnih opcija dobija se sledeći rezultat:

```
apples
bananas
kiwis
oranges
pears
```

Ukoliko bi želeli da sačuvamo ovaj rezultat morala bi da se koristi komanda za redirekciju izlaza:

sort data.txt > output.txt

Ova komanda će sačuvati rezultat u novi fajl **output.txt**.

OPCIJE SORT KOMANDE

Primeri opcija SORT komande

SORT komanda takođe ima i opciju za preusmeravanje rezultata u fajl:

sort -o output.txt data.txt

Primer: Sortiranje po obrnutom redosledu može se izvršiti korišćenjem opcije -r: **sort -r data.txt** što će imati sledeći rezultat:

```
pears
oranges
kiwis
```

bananas
apples

Koristeći opciju -c može se proveriti da li je fajl već sortiran **sort -c data.txt**.

Primer: Sortiranje i uklanjanje istih linija, odnosno ispisivanje samo onih koje se ne ponavljaju. Ukoliko fajl data.txt pre sortiranja ima sledeće podatke:

kiwis
oranges
apples
oranges
bananas
mango
kiwis
apples

Nakon primene komande **sort -u data.txt** rezultat će biti sledeći:

kiwis
oranges
apples
bananas
mango

SKRIPTE

*Poznavanje shell skriptinga je neophodna vještina za sistem administratore i programere koji rade na *nix sistemima.*

Često je potrebno izvršiti operacije koje se sastoje od više koraka, tj. koriste više shell komandi. Ovakvi procesi se mogu automatizovati korišćenjem shell skripti. Poznavanje shell skriptinga je neophodna vještina za sistem administratore i programere koji rade na *nix sistemima.

Shell skripte su tekstualni fajlovi sa skupom komandi koje shell treba da izvrši. Obično koriste nastavak **.sh**. Da b se shell skripte izvršile, potrebno je da fajl skripte ima dozvolu za izvršavanje. Ova dozvola se dodaje komandom chmod.

chmod +x file

Znak **#** označava komentar u skripti. Linije koje počinju sa **#** se ignorišu.

Prva linija skripte počinje sa znakom **#** i navodi putanju do interpretera. U ovom slučaju koristimo bash, pa će prva linija skripte uvijek biti:

#/bin/bash

Sledeća skripta arhivira sadržaj direktorijuma Documents, i kopira rezultujuću arhivu u **/home/user/backup** direktorijum.

#/bin/bash

```
tar -cf backup.tar /home/user/Documents  
cp backup.tar /home/user/backup
```

PRIMERI SHELL SKRIPTI

Zadaci sa rešenjem

Predviđeno vreme izrade sledećeg primera je 5 minuta.

1. Primer

Kreiranje jednostavne shell scripte koja ispisuje „Zdravo svete!“

```
#!/bin/sh  
echo "Zdravo svete!"
```

Predviđeno vreme izrade sledećeg primera je 5 minuta.

2. Primer

Kreiranje shell skripte sa argumentima:

```
#!/bin/bash  
  
# primer korišćenja argumenata u skriptama  
echo "Moje ime je $1"  
echo "Moje prezime je $2"  
echo "Ukupan broj argumenata je $"
```

Zapamtite skriptu kao ime.sh, omogućite dozvole za fajl naredbom chmod a+x ime.sh i nakon toga izvršite fajl pokretanjem ./ime.sh.
chmod a+xime.sh ./ime.sh Goran Stamenovic

Rešenje:

Moje ime je Goran
Moje prezime je Stamenovic
Ukupan broj argumenata je 2

▼ Poglavlje 9

Zadaci za samostalni rad: UNIX Shell skripte

ZADATAK ZA SAMOSTALNI RAD

Napisati shell skriptu

Predviđeno vreme za izradu sledećih zadataka je 35 minuta.

- a) Napisati shell skriptu koja će ispisati trenutni datum (Dan-Mesec-Godina) i vreme (Sat:Minuta:Sekunda) u navedenim formatima. Za izradu ovog zadatka koristiti *date* komandu. (Vreme izrade: 15 minuta)
- b) Napisati shell skriptu koja treba da proveri da li fajl sa navedenim imenom (ime fajla treba da se prosledi kao argument) postoji ili ne. Ukoliko ne postoji kreirati fajl sa tim imenom. (Vreme izrade: 20 minuta)

✓ Poglavlje 10

DOMAĆI ZADATAK

OPIS DOMAĆEG ZADATKA - DZ05

Kreirati bash skriptu koja će koristiti komande iz vežbi

Predviđeno vreme izrade domaćeg zadatka je 50 minuta.

Kreirajte DZ05.txt datoteku koja će imati više od 50 linija i u kojoj se pojavljuje više puta reč programiranje. Ovo može i biti neki od vaših prethodnih domaćih zadataka.

***Zameniti slova abcd sa ciframa broja indeksa. Ukoliko se prilikom deljenja sa 2 ne dobije ceo broj zaokružiti dobijeni.**

Kreirati bash skriptu koja će izvršiti sledeće zadatke:

1. Napisati komandu koja će **ispisati broj karaktera prvih $ab/2$ linije datoteke /dz05.txt**. Koristiti komandu **head** u kombinaciji sa ostalim potrebnim filterima. (Za broj indeksa 1234, zadatak bi glasio ispisati broj karaktera prvih 6 linija)
2. Napisati komandu koja će **ispisati broj karaktera zadnjih $cd/2$ linija datoteke /dz05.txt**. Koristiti linux komandu **tail** u kombinaciji sa ostalim potrebnim filterima. (Za broj indeksa 1234, zadatak bi glasio ispisati broj karaktera prvih 17 linija)
3. **Napisati komandu koja će naći ukupan broj bajtova svih linija iz fajla /dz05.txt** koji sadrže reč "programiranje". Broj bajtova upisati u datoteku **/bytes.txt**. Koristiti redirekciju izlaza.
4. **Napisati komandu koja će ekstrakovati arhivu /archive.tar** u direktorijum **/extracted**, a potom ispisati sadržaj direktorijuma **/extracted**.
5. Napisati komandu koja će iz fajla **/dz05.txt** izvući prvih **a** i poslednjih **d** linija i upisati ih u novi fajl. (Za broj indeksa 1234, zadatak bi glasio izvući prvih 1 i poslednjih 4 linija i upisati u novi fajl)

Napomena:

- **Domaći zadatak pošaljite predmetnom asistentu na e-mail, a u subject-u mejla napisati IT101 - DZ05**
- Zadatak dostaviti kao **IT101-DZ05-Ime_Prezime_BrojIndeksa.sh**
- Pored bash skripte dostavite i dokument koji opisuje kako ste svaku od navedenih koraka rešili, šta komande koje ste koristili postižu, kao i "rešenja" za svaki od zadataka. Na primer, koliki je broj karaktera u prvih deset linije datoteke koji je tražen. Ovaj dokument dostaviti kao **IT101-DZ05-Ime_Prezime_BrojIndeksa.doc**

▼ Zaključak

ZAKLJUČAK

Na ovom predavanju je bilo reči o ulogama operativnih sistema, kao i podeli operativnih sistema na osnovu različitih kriterijuma. Od osnovnih funkcionalnosti operativnih sistema obrađeni su upravljanje memorijom, upravljanje uređajima i koordinacija zadataka. Operativne sisteme smo podelili prema njihovoj ulozi, načinu rada i arhitekturi. Dati su primeri za operativne sisteme personalnih računara, servera, ugnježdenih, real-time sistema, međinfrejm i dr. Ne treba zaboraviti da se zastupljenost različitih operativnih sistema menja iz godine u godinu, a ovde smo prikazali samo zastupljenost koja je bila 2012-te godine.

Literatura

1. Gary B. Shelly, Misty E. Vermaat, Discovering Computers 2011-Introductory: Living in a Digital World, Cengage Learning 2010.



IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

RAČUNARSKE MREŽE I KOMUNIKACIJE

Lekcija 06

PRIRUČNIK ZA STUDENTE

IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

Lekcija 06

RAČUNARSKE MREŽE I KOMUNIKACIJE

- ▼ RAČUNARSKE MREŽE I KOMUNIKACIJE
 - ▼ Poglavlje 1: Računarska komunikacija
 - ▼ Poglavlje 2: Računarska mreža
 - ▼ Poglavlje 3: Prenosni medijum
 - ▼ Poglavlje 4: Mrežni uređaji
 - ▼ Poglavlje 5: Mrežni protokoli i standardi
 - ▼ Poglavlje 6: Pokazna vežba: Cisco Packet Tracer
 - ▼ Poglavlje 7: Zadatak za samostalni rad: Kreiranje mreže
 - ▼ Poglavlje 8: Domaći zadatak
 - ▼ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

✓ Uvod

UVOD

Cilj ovog predavanja je da predstavi osnovne karakteristike žičnih i bežičnih mreža i njihovih hardverskih i softverskih komponenti

U ovom predavanju će biti obrađene sledeće teme:

- značaj i prednosti umrežavanja
- prenosni medijumi (žični i bežični)
- mrežna oprema
- mrežni protokoli

Cilj ovog predavanja je da predstavi osnovne karakteristike žičnih i bežičnih mreža i njihovih hardverskih i softverskih komponenti. Poseban fokus je na definiciji šta čini računarsku mrežu i kako se odvija efikasna komunikacija u njoj.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 1

Računarska komunikacija

ŠTA PREDSTAVLJA RAČUNARSKU KOMUNIKACIJU?

Računarska komunikacija predstavlja dva ili više uređaja koji su povezani tako da mogu da šalju i primaju podatke u formi instrukcija ili informacija

Računarska komunikacija predstavlja dva ili više uređaja koji su povezani tako da mogu da šalju i primaju podatke u formi instrukcija ili informacija. Komunikacioni sistem može da bude računar, telefon, satelit, navigacioni sistem, mejnfrejm računar i sl. Ova komunikacija može da se obavi žično ili bežično. Računarske komunikacije podrazumevaju da postoji uređaj koji ima ulogu pošiljaoca, uređaj koji ima ulogu primaoca i da se komunikacija obavlja preko komunikacionog kanala. Pošiljalac inicira prenos podataka, dok primalac prima ove podatke, koje je primio preko komunikacionog kanala (prenosnog medijuma).

▼ Poglavlje 2

Računarska mreža

ŠTA PREDSTAVLJA RAČUNARSKU MREŽU?

Za realizaciju jedne računarske mreže, pored samih računarskih sistema, potrebno je obezbiti mrežnu infrastrukturu i definisati protokole po kojima mreža radi

Od svog nastanka pa do danas računarske mreže su se razvijale eksplozivno. To je dovelo do toga da su računarske komunikacije postale jedan od najvažnijih delova IT infrastrukture. Danas je gotovo nezamislivo da su u nekoj organizaciji, pa i u kući, računarski sistemi neumreženi.

Računarska mreža je skup međusobno povezanih autonomnih računarskih sistema ili drugih uređaja koji se mogu povezati na mrežu, a koji se uopšteno nazivaju čvorovi. Pod pojmom međusobno povezani ne misli se da su uvek povezani nekim provodnikom, jer postoje i bežične mreže, nego da su računarski sistemi sposobni da komuniciraju. Svaki računar u mreži je autonoman, što znači da drugi računari ne mogu da upravljaju njime. Mreže mogu da budu sastavljene i od samo dva računarska sistema koji se nalaze u istoj prostoriji, a mogu da sadrže na hiljade računara lociranih po celoj zemaljskoj kugli.

Za realizaciju jedne računarske mreže, pored samih računarskih sistema, potrebno je obezbiti mrežnu infrastrukturu i definisati protokole po kojima mreža radi. Mrežna infrastruktura je hardver koji povezuje računare tako da oni mogu da razmenjuju signale. Protokoli specificiraju usluge koje obezbeđuje mreža. Protokoli omogućuju da se hardver iskoristi pomoću aplikativnih programa za komunikaciju.

Tako možemo reći da se računarska mreža sastoji od:

- mrežnih čvorova
- prenosnog medijuma
- protokola.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

✓ 2.1 Prednosti umrežavanja

RAZLOZI ZA UMREŽAVANJE

Generalno gledano, može se reći da računarske mreže omogućuju ekonomičniju primenu računara i pružaju nove usluge

Postoje različiti razlozi zašto se računari umrežavaju i oni su se, istorijski gledano, menjali. U doba kada su se mreže pojavile, primarni razlog za umrežavanje je bio zajedničko korišćenje resursa IT sistema. Devedesetih godina prošlog veka primarni razlog umrežavanja je postao komunikacija između računarskih sistema. **Generalno gledano, može se reći da računarske mreže omogućuju ekonomičniju primenu računara i pružaju nove usluge.** Neke od prednosti umrežavanja računarskih sistema su:

- Deljenje hardvera
- Deljenje podataka i informacija
- Deljenje softvera
- Komunikacije
- Povećanje pouzdanosti
- Transfer novca
- Smanjenje troškova

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

DELJENJE HARDVERA

U poslovnom okruženju je neekonomično da se svakom zaposlenom kupi po štampač, pa se zbog toga često deli štampač na više zaposlenih

U doba kada su hardverski resursi bili skupi postojao je veliki interes da se omogući da više računara koristi iste zajedničke resurse. Deljenje hardvera na mreži omogućava da se uštedi novac, ne samo u poslovnim nego i kućnim mrežama. U poslovnom okruženju je neekonomično da se svakom zaposlenom kupi po štampač, pa se zbog toga često deli štampač na više zaposlenih. Ovo se pre svega odnosilo na štampače, plotere, diskove i druge ulazno-izlazne uređaje. I danas, po pravilu, više računara u istoj prostoriji koristi zajednički štampač.

DELJENJE PODATAKA I INFORMACIJA

Vrlo je neracionalno da se te informacije nalaze u svakom računarskom sistemu, kako sa aspekta zauzetosti memorijskih resursa, tako i sa aspekta održavanja integriteta informacija

U svakoj organizaciji postoji masa informacija za koju su zainteresovani svi ili bar većina korisnika. Postoji mnogo primera kada se računarska mreža koristi za deljenje informacija kao što su datoteke, baze podataka, digitalna biblioteka, web i drugo. Vrlo je neracionalno da se te informacije nalaze u svakom računarskom sistemu, kako sa aspekta zauzetosti memorijskih resursa, tako i sa aspekta održavanja integriteta informacija. Zbog toga se takve informacije smeštaju u jedan računarski sistem, a svim autorizovanim korisnicima se daje pravo pristupa. Na primer, **velika kompanija ima bazu podataka sa podacima o svim svojim mušterijama, kao i o robi koju prodaju.** Autorizovana osoba sa pristupom mreži, treba da ima pristup svim ovim informacijama. Poznati standardi za deljenje podataka su:

- **EDI** (Electronic Data Interchange) za rukovanje i distribuciju kataloga, ponuda, narudžbi, faktura, naplatu, obaveštenja o poštarinama i transportu, procesovanju uplata
- **XML** (Extensible Markup Language) za omogućavanje različitim uređajima da prikažu sadržaj

S druge strane, dok je deljenje podataka i informacija jedna od glavnih prednosti umrežavanja, sa povećanom razmenom podataka, bilo poslovnih bilo privatnih, pojavio se i problem. Prilikom prenosa podataka putem Interneta pojavila se povećana potreba za pouzdanošću, bezbednosti, ali i nizom mehanizama koji treba da obezbede očuvanje privatnosti podataka.

DELJENJE SOFTVERA

Instaliranje iste aplikacije na svim računarskim sistemima u mreži nije racionalno kako sa aspekta zauzetosti memorijskog prostora, tako i sa aspekta održavanja aplikacija

Slično kao i sa deljenjem podataka i informacija, ni **instaliranje iste aplikacije na svim računarskim sistemima u mreži nije racionalno kako sa aspekta zauzetosti memorijskog prostora, tako i sa aspekta održavanja aplikacija.** Osim toga, politika licenciranja pojedinih softverskih kuća je takva da nameće rešenje kojim se aplikacija instalira samo na jednom računarskom sistemu, uobičajeno aplikacionom serveru. Svi autorizovani korisnici mogu da koriste instancu ove aplikacije na svom računarskom sistemu. **Time je umnogome olakšana administracija aplikacija.** Na primer, **prilikom instaliranja nove verzije programa, instalacija se vrši samo na aplikacionom serveru, a ne na svim računarima u mreži.**

Kada su računari umreženi, to značajno pojednostavljuje i njihovu podršku. Za jednu organizaciju je daleko efikasnije kada tehničko osoblje održava jedan operativni sistem i kada su svi računari identično podešeni prema konkretnim potrebama te organizacije.

KOMUNIKACIJE

Prednost računarskih mreža je što istom infrastrukturom omogućavaju sinhrone i asinhrone komunikacije, kao i prenos podataka, govora, zvuka i videa

Komunikacije danas postaju dominantni razlog umrežavanja računara, pošto omogućavaju da ljudi komuniciraju efikasnije. **Prednost računarskih mreža je što istom infrastrukturom omogućavaju sinhrone i asinhrone komunikacije, kao i prenos podataka, govora, zvuka i videa.** Danas se u ove svrhe masovno koriste mnogi servisi kao što su: **transfer datoteka, e-mail, chat, diskusioni forumi, VoIP, telekonferencije** i drugi.

S druge strane, iako su računarske mreže olakšale komunikaciju, što se jednako odnosi i na privatnu, ali i poslovnu komunikaciju, pojavio se i niz nus pojava zbog iste. Naime, sa sve većim brojem različitih aplikacija koje putem interneta omogućavaju komunikaciju, ljudi sve više pribegavaju komuniciranju putem interneta, pri čemu se dosta gubi ljudski kontakt, odnosno ljudski karakter komunikacije među ljudima. "Internet komunikacija" je čak dovela do novog kodeksa onlajn ponašanja, pa čak i samog komuniciranja.

POVEĆANJE POUZDANOSTI

Korišćenjem mreže umnogome je olakšana izrada rezervnih kopija važnih datoteka i baza podataka, što takođe povećava pouzdanost sistema

Iako računarske mreže donose mnoge opasnosti po sigurnost računarskih sistema, posmatrano u celini one doprinose pouzdanosti. Korišćenjem mreža moguće je da se jedan uređaj koji nije u funkciji zameni istovremeno istim ili sličnim uređajem koji je vezan za mrežu. **Korišćenjem mreže umnogome je olakšana izrada rezervnih kopija važnih datoteka i baza podataka, što takođe povećava pouzdanost sistema.**

TRANSFER NOVCA I SMANJENJE TROŠKOVA

Korišćenjem navedenih mrežnih servisa u nekoj organizaciji moguće je iste poslovne ciljeve ostvariti sa manje ulaganja u hardversku i softversku infrastrukturu

Transfer novca

Elektronski transfer novca omogućuje korisnicima koji su umreženi da prebacuju novac iz jedne banke u drugu. Ceo proces je pojednostavljen i ubrzan, naročito kada se uporedi sa čitavom papirologijom koju je potrebno popuniti da se ovaj transfer ne obavlja elektronski.

Smanjenje troškova

Korišćenjem navedenih mrežnih servisa u nekoj organizaciji moguće je iste poslovne ciljeve ostvariti sa manje ulaganja u hardversku i softversku infrastrukturu. Indirektno, omogućava da se smanje troškovi kada radnici imaju mogućnost da na efikasan način pristupe podacima i informacijama u njihovom svakodnevnom poslovanju.

✓ Poglavlje 3

Prenosni medijum

TIPOVI PRENOSNOG MEDIJUMA

Prenosni medijum može biti fizički ili bežični

Posmatrajući najniži nivo računarske komunikacije, komunikacija se obavlja preko prenosnog medijuma. Prenosni medijum može biti metal, staklo, etar i sl. Dakle prenosni medijum, praktično, koristimo kao komunikacioni kanala, preko koga prenosimo podatke. U opštem slučaju prenosnim medijumom se prenosi signal, odnosno njegova energija. Energija može biti električna, svetlosna, elektromagnetna, a svaka od njih ima različite osobine. Shodno tome, govorimo o različitim tipovima prenosnih medijuma.

Prenosni medijum može biti **fizički** ili **bežični**. Fizički prenosni medijum može da koristi kablove kao što su upredene parice, optički kabl ili koaksijalni kabl. Svaki od ovih prenosnih medijuma se koristi u različitim mrežama i shodno tome imaju i različite karakteristike (Tabela 1).

Brzine prenosa za fizički prenosni medijum	
Tip kabla	Maksimalna brzina prenosa
Upredene parice	
10Base-T (Ethernet)	10Mbps
100Base-T (Fast Ethernet)	100Mbps
1000Base-T (Gigabit Ethernet)	1Gbps
Koaksijalni kabl	
10Base2 (ThinWire Ethernet)	10 Mbps
10Base2 (ThickWire Ethernet)	10 Mbps
Optički kabl	
10Base-F (Ethernet)	10Mbps
10Base-FX (Fast Ethernet)	100Mbps
Gigabit Ethernet	1Gbps
10- Gigabit Ethernet	10Gbps
40- Gigabit Ethernet	40Gbps
100- Gigabit Ethernet	100Gbps

Slika 3.1.1 Tabela-1 Brzine prenosa za različite fizički prenosne medijume

❖ 3.1 Bakarni provodnici

ZAŠTO SE KORISTE BAKARNI PROVODNICI

Bakarni provodnici se koriste kao mediji u računarskim mrežama zato što su jednostavnji za instaliranje, imaju malu električnu otpornost i jeftini su

Bakarni provodnici se koriste kao mediji u računarskim mrežama zato što su jednostavnji za instaliranje, imaju malu električnu otpornost i jeftini su. Oblik provodnika je različit i uslovjen je zahtevom da se minimiziraju smetnje. Smetnje nastaju zato što prolaskom električnih signala kroz provodnik, on deluje kao minijaturni radio-predajnik jer emituje elektromagnetnu energiju, koja u drugim paralelno postavljenim provodnicima koji su na maloj udaljenosti generiše električnu struju.

Pošto prijemnici ne mogu da detektuju koji su signali poslati od izvorišnog sistema, a koji su generisani slučajno, dolazi do smetnji, odnosno do prekida normalne komunikacije. Da bi se minimizirale ove smetnje, bakarni provodnici se za mreže koriste u dva oblika, i to kao:

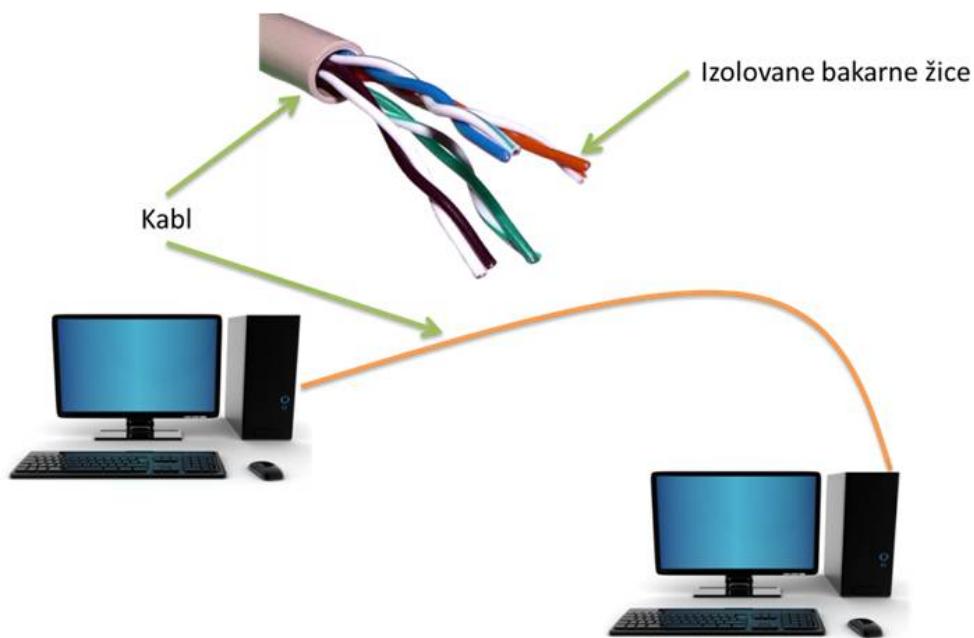
- upredene parice
- koaksijalni kabl

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

UPREDENE PARICE

Upredene parice se sastoje od jednog ili više pari žica koje su upredene zajedno

Upredene parice se uglavnom koriste za **mrežno umrežavanje i telefonski saobraćaj**. One se sastoje od jednog ili više pari žica koje su upredene zajedno. Sve upredene parice se sastoje od dve uvijene izolovane bakarne žice (Slika 1). Žice su upredene kako bi smanjile elektromagnetne smetnje, odnosno, preslušavanje sa drugih žica.



Slika 3.2.1 Upredene parice

KOAKSIJALNI KABL

Koaksijalni kabl se sastoji od bakarnog jezgra (žice) koji je obmotan unutrašnjim dijalektričnim izolatorom, metalnom mrežicom i plastičnim spoljnjim izolatorom

Koaksijalni kabl! se sastoji od bakarnog jezgra (žice) koji je obmotan unutrašnjim dijalektričnim izolatorom (engl. **dielectric insulator**), metalnom mrežicom i plastičnim spoljnjim izolatorom. Komunikacija se obavlja kroz bakarnu žicu. Svi izolacioni slojevi smanjuju curenje elektrničnog i magnetnog polja. Koaksijalni kabl se uglavnom koristi za **kablovsku televiziju** pošto može da prenosi podatke na veće distance nego upredene parice, zbog smanjenih smetnji. U jednom trenutku ovo su bili najrasprostranjeniji mrežni kablovi, i to iz više razloga: relativno su jeftini, laki, fleksibilni i jednostavnii za rad. Kablovi koji imaju jedan sloj izolacije i jedan sloj od upletenog metala zovu se i kablovi sa dvostrukom zaštitom. Postoje, takođe, i kablovi sa četvorostrukou zaštitom (dva sloja izolacije i dva sloja metalne mrežice), koji se primenjuju u sredinama sa jakim elektromagnetskim smetnjama. Bakarni provodnik i metalna mrežica ne smeju biti u kontaktu. Kada eventualno dođe do kontakta u kablu, nastaje kratak spoj, a šum ili zalutali signali sa mreže prelaze u provodnik. Na ovaj način dolazi do pojave skretanja podataka u neželjenom smeru. Koaksijalni kabl je mnogo otporniji na pojave mešanja i slabljenja signala od kabla sa upredenim paricama.



Slika 3.2.2 Koaksijalni kabl

3.2 Optički kablovi

STAKLENA VLAKNA

Predajnik koristi svetlosnu diodu ili laser da pošalje svetlosne impulse duž vlakna

Pored bakarnih provodnika, računarske mreže koriste savitljiva staklena vlakna za prenos podataka. Ova vrsta prenosnog medija je poznatija kao optički fiber (engl. optical fiber). Vrlo tanko stakleno vlakno je oklopljeno plastičnom izolacijom. Predajnik, na jednoj strani staklenog vlakna, koristi svetlosnu diodu (engl. light emitting diode(LED)) ili laser da pošalje svetlosne impulse duž vlakna. Prijemnik, na drugoj strani vlakna, koristi svetlosno osetljivi tranzistor da pročita svetlosne impulse i pretvori ih u električne signale.

Svetlosna vlakna imaju nekoliko prednosti u odnosu na bakarne provodnike:

- Svetlosna vlakna ne emituju niti su osetljiva na elektromagnetske smetnje.
- Dužina mrežnog segmenta izrađenog od staklenog vlakna može da bude mnogo veća nego kod bakarnog provodnika zato što gotovo da nema gubitka energije tokom prenosa.
- Važna prednost staklenog vlakna je da se signali prenose ogromnom brzinom.
- Bolja bezbednost tokom prenosa.

Glavni nedostatak optičkih kablova u poređenju sa bakarnim provodnicima je njegova visoka cena, kao i kompleksnije instaliranje u odnosu na koaksijalni kabl.

Za razliku od ostalih kablova, optički kabl ima sposobnost da prenosi veliki broj informacija velikom brzinom, budući da se informacije prenose putem svetlosti. Ovi kablovi su izuzetno tanki i osetljivi, ali zahvaljujući savremenim tehnologijama, pakuju se u snopove sa više vlakana (2, 4, 8 pa sve do 512) i vrlo ih je lako postaviti u zemlju, pod vodu, na stubove i dalekovode.

Fiber optički kabl omogućava izuzetno brzo prenošenje velike količine podataka između geografski udaljenih lokacija. Zahvaljujući činjenici da se lako instaliraju pod vodom, ne čudi

činjenica da su kontinenti povezani ovim kablovima, te tehnički internet kakav pozajemo postoji zahvaljujući optičkim kablovima.

KAKO SE PRAVE STAKLENA VLAKNA?

How It's Made, Fiber Optics

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ 3.3 Radio talasi

RADIO FREKVENCIJA

Elektromagnetski radio talasi se koriste za prenos podataka bežičnim putem koristeći takozvanu radio frekvenciju

Jedan od načina da se podaci prenesu od izvorišnog do odredišnog sistema je korišćenje elektromagnetnih radio-signalata koji rade na frekvenciji koja se koristi za prenos radio i TV signala, odnosno na takozvanoj **radio-frekvenciji (RF)**. U ovom slučaju ne postoji fizička veza izvedena nekom vrstom kablova, ali predajnik i prijemnik moraju da imaju neku vrstu antena koje emituju i primaju elektromagnetne talase u delu spektra od nekoliko MHz do otprilike 3 GHz. Nedostatak prenosa radio-signalima zemaljskim antenama je što se zahteva **optička vidljivost** između antena (na primer zgrade, zidovi, planine i dr.), što može da se reši primenom satelitskih antena. Neke od bežičnih tehnologija koje koriste radio talase za svoj prenos su **BlueTooth, WiFi, WiMAX, i Ultra Wideband (UWB)**.

MIKROTALASI

Mikrotalasi obuhvataju spektar elektromagnetnih talasa čija je frekvencija između 300MHz i 300GHz

Mikrotalasi obuhvataju spektar elektromagnetnih talasa čija je frekvencija između 300MHz i 300GHz. Zbog toga što radi na višim frekvencijama, prenos mikrotalasima **ima veći propusni opseg u odnosu na RF talase**, ali je osetljiv na smetnje, vremenske prilike i prisluškivanje. Mikrotalasni prenos se često naziva i fiksnim bežičnim prenosom, a odnosi se na prenos između dve mikrotalasne stanice. Mikrotalasna stanica je uređaj sa antenom i transiverom. Pošto je podložan smetnjama, mikrotalasni prenos zahteva da pošiljalac i primalac budu jedan drugom u direktnom vidokrugu. Takođe, ima svoje prednosti kada se koristi u zabačenim oblastima gde fizički prenosni medijum nije dostupan.

INFRACRVENI TALASI

Za prenos infracrvenih talasa je potrebna direktna vidljivost između prijemnika i predajnika

Infracrveni talasi (engl. **infrared (IR)**) rade na frekvencijama koji su odmah ispod donje granice vidljivog spektra. Za prenos infracrvenih talasa je potrebna direktna vidljivost između prijemnika i predajnika, ali postoje tehnologije koje omogućuju i difuzni prenos korišćenjem talasa odbijenih od okolnih predmeta. Ova vrsta prenosa se koristi za prenos signala unutar jedne prostorije, jer talasi imaju ograničen domet i ne mogu da prođu kroz zidove. Mobilni računari, štampači, telefoni i miševi često koriste IR za prenos podataka sa jednog uređaja na drugi.

Princip rada infracrvenih talasa se bazira na tome da uvek imate infracrveni emiter (emituje infracrveni talas) i infracrveni receiver (detektuje infracrveni talas). U mobilnoj robotici se isti sistem koristi za potrebe zaobilaženja prepreka, kako bi mobilni robot izgledao pametno, jer mnogi pomisle: kako zna da je ovde zid ili bilo kakva prepreka, zapravo ima infracrveni talas koji odašilje infracrveni senzor a zatim isti detektuje odbijeni signal. U suštini, infracrveni sistem uvek ima predajnu stranu i prijemnu stranu, senzori za mobilne robote imaju isto integrисано u okviru istog package-a, dok infracrveni senzor za televiziju ima odvojeno, emiter se nalazi u okviru daljinskog upravljača dok je prijemna strana integrисана u okviru televizora. Bezbednosni sistemi video nadzora imaju integrисани sistem, gde se infracrveni emiter nalazi u istom pakovanju kao i infracrveni detektor. Što se tiče složenosti, bezbednosni sistem prednjači jer isti daje video izlaz određene rezolucije dok, sistemi mobilnih roboata se baziraju isključivo na detekciji signala kao i televizijski infracrveni sistem.

Ljudsko oko ne može detektovati IR talas, ali ne može jasno da vidi ni u mraku, bezbednosni sistemi video nadzora samim tim koriste infracrvene sisteme kako bi snimali šta se dešava tokom noći. IR sistem video nadzora koristi isti princip kao i gore navedeni senzori. Postoje IR emiter koji generišu IR talas, IR talas se odbija od svim predmeta tj. od okruženja, tako odbijeni IR signal se vraća nazad ka sistemu video nadzora do IR detektoru koji je dosta složeniji od gore navedenih, ali zapravo radi isto. IR talase demoduliše u električne signale koji je nosio video snimka koji se zatim zapisuje tj. snima na HDD (harddisk bezbednosnog sistema). Znači kao i u gore navedenim slučajevima: imamo IR emitere, set IR LED dioda koje emituju IR talas, IR talas se odbija od prostorije, od svih predmeta koji se u istoj nalaze, tako dobijeni signal detektuje IR detektor, koji IR talas demoduliše u električni signal tj. video snimaj koji se zatim snima na HDD. Šta je poenta takođe u ovoj priči, upravo IR talas, isti nije vidljiv golim okom, provalnici, ili oni koji se bave pretanjama (recimo) često nisu sigurni da li je bezbednosni sistem video nadzora tj. kamera prava ili ne, **Night Vision** (noćno snimanje) funkcioniše na principu IR talasa ali isti nije vidljiv ljudskom oku.

▼ Poglavlje 4

Mrežni uređaji

ŠTA JE MREŽNI UREĐAJ?

Mrežni uređaj je deo mrežne opreme koji ima mogućnost da prenosi podatke i instrukcije između pošiljaoca i primaoca

Mrežni uređaj je deo mrežne opreme koji ima mogućnost da prenosi podatke i instrukcije između pošiljaoca i primaoca. Neki od uređaja koji će biti obrađeni su dial-up modem, digitalni model, bežični modem, mrežna kartica, bežična pristupna tačka, ruter, hab, svič, repetitor i mrežni most.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

MREŽNA KARTICA

Svi mrežni paketi se šalju i primaju preko mrežne kartice

Mrežna kartica (engl. **network interface card (NIC)**) predstavlja komunikacioni uređaj koji omogućava računaru ili nekom drugom uređaju da pristupi mreži. Svi mrežni paketi se šalju i primaju preko mrežne kartice. Mrežna kartica ima port za kablovsko povezivanje (Slika 1). Mrežne kartice za mobilne računare su obično USB mrežni adapteri ili ExpressCard moduli (Slika 1).

NIC sadrži elektronsko kolo koje je neophodno za žičnu (na primer, Ethernet) ili bežičnu konekciju (na primer, WiFi).

Mrežna kartica se još naziva i mrežni kontroler (eng. **network interface controller**), mrežni adapter (eng. **network adapter**) ili **Local Area Network (LAN)** adapter.

Većina novih računara imaju ili Ethernet mogućnosti integrisane u matičnu ploču ili koriste neki jeftini namenski Ethernet čip povezan preko PCI ili PCI Express magistrale. Poseban NIC obično više nije potreban. Ako kartica ili kontroler nije integrisana u matičnu ploču, ona može da bude integrisana komponenta u ruteru, štampaču ili USB. Obično postoji LED pored konektora koji obaveštava korisnika da li je mreža aktivna i da li se podaci prenose. U zavisnosti od kartice ili matične ploče, brzina prenosa podataka može biti 10, 100 ili 1000 megabita u sekundi.



Slika 4.1 Mrežna karica i Express card modul

DIAL-UP MODEM

Dial-up modem pretvara digitalne signale u analogni, i analogni u digitalni, kako bi mogli da prenose i primaju signale iz telefonskih linija.

Dial-up modem pretvara digitalne signale u analogni, i analogni u digitalni, kako bi mogli da prenose i primaju signale iz telefonskih linija. Konverzija je neophodna zato što računar može da obradi podatke i informacije samo ako su u digitalnom obliku, a signal se preko telefonskih linija šalje kao analogni signal. Dial-up modem se obično umeće na matičnu ploču računara i povezan je preko telefonskog kabla na telefonski utičnik.

DIGITALNI MODEMI

Digitalni modemi su uređaji kao što su ISDN, DSL i kablovski modem

Digitalni modemi mogu da prime i šalju digitalni signal ka i od digitalnog kanala. Digitalni modemi su uređaji kao što su ISDN, DSL i kablovski modem. Integrated Services Digital Network (ISDN) je dizajniran da omogući digitalni prenos preko običnih telefonskih bakarnih žica. ISDN omogućava da se prenos obavlja preko više digitalnih kanala istovremeno. Prenos se obavlja preko regularnih telefonskih linija koji se koriste za analogni prenos, međutim, telefonske kompanije sa modernom opremom mogu da obrade i podrže digitalnu konekciju. Tako da, iako se koriste telefonske linije, prenos se obavlja digitalnim, a ne analognim signalom. Samim tim, ovakav prenos omogućava veće brzine prenosa podataka.

Digital Subscriber Line (DSL) je takođe digitalni prenosni sistem koji koristi telefonske bakarne žice. Razlika između DSL i ISDN je u brzini, ceni i tehnologiji. ISDN je nešto skupljii od DSL, dok DSL ima veće brzine prenosa. Najveća razlika između ISDN i DSL je tehnologija koja se koristi. ISDN koristi dve verzije, jednu za kućnu upotrebu, a drugu za poslovne mreže. Kućna verzija ili Basic Rate Interface(BRI) koristi tri kanala, dok poslovna verzija Primary Rate Interface (PRI) koristi 24 kanala. Svaki od ISDN kanala se može koristiti zasebno, tako da svaki kanal ima svoj opseg (npr. prenosnu brzinu podataka). DSL koristi samo jedan kanal da prenosi glas, podatke i video. Takođe, DSL ima dva prenosna kapaciteta koje nazivamo "upload" i "download", dok ISDN radi u punom dupleksu gde se podaci u oba smera prenose u istom kapacitetu.

REPETITORI

Repetitor prima signal sa jednog segmenta, čisti ga i pojačava, a zatim ga šalje na drugi segment

Repetitor (engl. **repeater**) je uređaj koji se koristi kada je potrebna dužina kabla između dva čvora mreže veća od dozvoljene dužine. Zbog toga što signal koji se kreće kroz provodnik na dugim segmentima mreže oslabi neophodno ga je pojačati. **Repetitor prima signal sa jednog segmenta, čisti ga i pojačava, a zatim ga šalje na drugi segment.**

HAB

Hab prima pakete iz mreže i šalje ih svim uređajima koji su povezani na njega

Hab ili **razvodna kutija** (engl. **hub**) je Ethernet uređaj koji povezuje više računara u jednu mrežnu komponentu. Svi uređaji koji su povezani na hab mogu da komuniciraju između sebe. **Hab prima pakete iz mreže i šalje ih svim uređajima koji su povezani na njega.** Uloga haba nije da čita i obrađuje podatke koji prolaze kroz njega, tako da on nije "svestan" kome su ti paketi zaista namenjeni kada ih prosledi. Hab ima više portova na koji su računari povezani. Manji hab ima 4-5 porta, dok veći imaju 8, 12, 16 i 24 portova. Hab može biti podeljen u tri grupe:

- aktivni
- pasivni
- inteligentni

Pasivni hab ne pojačava signal pre nego što ga prenese, on samo difuzno emituje pakete svima, dok aktivni hab služi i kao repetitor, tako da pojačava signal. Inteligentni hab ima jednu dodatnu funkcionalnost u odnosu na aktivni hab. Oni omogućavaju udaljeno upravljanje koristeći upravljački protokol **SNMP i virtualni LAN (VLAN)**.

SVIČ

Za razliku od haba, komutatori su „svesni“ izvora i odredišta koji šalju i primaju pakete tako da tačno znaju na koju destinaciju da prenesu primljene pakete

Svič ili **komutator** (engl. **switch**) je uređaj koji povezuje više računara unutar lokalne mreže (**LAN**). Za razliku od haba, komutatori su „svesni“ izvora i odredišta koji šalju i primaju pakete, tako da tačno znaju na koju destinaciju da prenesu primljene pakete. Paketi se prenose samo do odredišta kome su paketi namenjeni, za razliku od haba koji svoje pakete difuzno šalju svima koji su priključeni na njega. Većina komutatora za kućnu upotrebu ima 4 ili 8 konekcija za Ethernet uređaje.

Koja je razlika između haba i sviča? Pogledajte kratak video koji ti objašnjava:

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

RUTER

Ruter ima zadatku da obezbedi putanju od čvora u jednoj mreži do odredišnog čvora u drugoj mreži

Ruter (engl. router) ima zadatku da obezbedi putanju od čvora u jednoj mreži do odredišnog čvora u drugoj mreži. Pri tom odredišni čvor može da bude udaljen nekoliko mreža. Ruter se još naziva i usmerivač. Ruter može da povezuje računare, druge ruter, uređaje kao što su štampači ili bežične pristupne tačke. Ruteri se mogu koristiti u mrežama različitih veličina. U manjim mrežama, ruteri se koriste da povežu računare, štampače i druge resurse na Internet. U većim mrežama, ruteri se koriste da prenose podatke najbržom mogućom putanjom. Ruteri mogu biti povezani sa drugim ruterima u mreži. Većina rutera je obezbeđena od neautorizovanog pristupa sa "firewall"-om. Neki ruteri imaju ugrađenu antenu i mogu prenositi bežičnim putem, te se tako eliminše potreba za korišćenjem bežične pristupne tačke kada se koristi bežični ruter.



Slika 4.2 Mrežni uređaji mogu da se povežu na Internet preko rutera

MREŽNI MOST

Mrežni most je uređaj koji se koristi da poveže dve mreže, pri čemu vrši ulogu i medijuma i filtera

Mrežni most (engl. network bridge) je uređaj koji se koristi da poveže dve mreže, pri čemu vrši ulogu i medijuma i filtera. On omogućava jednoj mreži da pošalje pakete od jednog mrežnog čvora do drugog mrežnog čvora, ako su MAC adrese oba čvora poznate. MAC adrese

svakog uređaja su jedinstvene u celom svetu, tako da je jednostavno da se prepoznaaju uređaji u mrežama koje su povezane mrežnim mostom. Takođe, mrežni most može da služi i kao filter za neke pakete. Za razliku od repetitora, koji samo pojačava signal, most može da se konfiguriše tako da može da određene pakete prosleđuje, dok druge odbacuje.

BEŽIČNA PRISTUPNA TAČKA

Bežična pristupna tačka omogućava bežičnim uređajima da se povežu na žičnu mrežu pomoću WiFi, Bluetooth ili slične bežične tehnologije

Bežična pristupna tačka (engl. wireless access point (**WAP**)) omogućava bežičnim uređajima da se povežu na žičnu mrežu pomoću WiFi, Bluetooth ili slične bežične tehnologije. Bežični uređaji imaju antene koje omogućavaju prijem i prenos bežičnih signala. Korišćenjem bežične pristupne tačke, korišćenje žica je minimizirano, a pristup mreži je omogućen na lokacijama gde je nemoguće razvuci kablove ili je neefikasno i neekonomično. Ovakva mesta su često javna mesta kao što su biblioteke, kafići, aerodromi ili slična mesta gde ljudi imaju potrebu da budu povezani na Internet konstantno dok su u pokretu.



Slika 4.3 Bežična pristupna tačka

▼ Poglavlje 5

Mrežni protokoli i standardi

ŠTA JE PROTOCOL?

Protokol je skup unapred definisanih pravila koja određuju kako dva ili više procesa treba da komuniciraju i stupaju u interakcije da bi razmenjivali podatke

Kao što hardver ne može da radi bez softvera, tako ni mreže ne mogu da funkcionišu bez protokola. **Protokol je skup unapred definisanih pravila koja određuju kako dva ili više procesa treba da komuniciraju i stupaju u interakcije da bi razmenjivali podatke.**

Osim toga, kako bi se osiguralo da različiti uređaji sa različitim hardverskim i softverskim komponentama mogu da komuniciraju i da se prevaziđe nekompatibilnost, različite organizacije kao što su **ANSI** i **IEEE** su predložile i razvile određene mrežne standarde. Postoje mnogi protokoli i standardi koji se koriste u računarskim mrežama. Neke od često korišćenih standarda su **Ethernet**, **TCP/IP**, **WiFi**, **Bluetooth**, **UWB**, **RFID**, **iMAX** i **WAP**. Neki od ovih standarda definišu na koji način mreža treba da bude dizajnirana i fizički organizovana, dok drugi određuju kako se formiraju paketi i na koji način se šalju kroz mrežu. U mnogim slučajevima kada se paket prenosi kroz mrežu potrebno je da se koristi više od jednog protokola.

ETHERNET

Ethernet je najčešće primenjivan standard za LAN mreže, pošto je lak za implementaciju i održavanje

Ethernet je komercijalno uveden 1980-ih i standardizovan je kao **IEEE802.3 standard**. Prvobitno Ethernet je koristio koaksijalni kabl za prenos, ali sada može da koristi upredene parice i optičke kablove. Prema ovom standardu svaki čvor može da pokuša da prenese podatke kad god je to potrebno. Međutim, kada se podaci prenose u isto vreme u zajedničkom domenu kolizije, tada može doći do kolizije dva paketa. Kada dođe do kolizije dva paketa, paketi se ponovo šalju od strane uređaja koji ih je prvobitno poslao. Drugim rečima, ovde ne postoji centralni računar ili uređaj koji kontroliše pristup prenosnim medijumima. Ethernet standard definiše fizičku konfiguraciju mreže (tj. kablove i mrežne kartice). **Ethernet je najčešće primenjivan standard za LAN mreže, pošto je lak za implementaciju i održavanje.** Noviji Ethernet standardi su Fast Ethernet i Gigabit Ethernet.

TCP/IP

TCP/IP je protokol koji definiše kako podaci treba da budu formatirani i prenešeni, tako da podaci budu primljeni i protumačeni pravilno

TCP/IP (Transmission Control Protocol / Internet Protocol) je protokol koji definiše kako podaci treba da budu formatirani i prenešeni, tako da podaci budu primljeni i protumačeni pravilno. Ovaj protokol ima sledeću ulogu:

- definiše kako se podaci fragmentišu i formatiraju u pakete,
- omogućuje da je svaki paket pravilno označen sa adresama izvorišnog i odredišnog uređaja,
- obezbeđuje da su podaci ispravno primljeni tako što proverava svaki paket za greške,
- reguliše protok saobraćaja duž mreže.

TCP/IP je usvojen kao standard za komunikaciju na Internet.

WIFI

WiFi se odnosi na sve proizvode koji koriste IEEE802.11 porodicu standarda

WiFi (Wireless Fidelity) omogućava računarima i drugim elektronskim uređajima da prenose podatake bežično. Uređaji koji koriste WiFi se povezuju na internet preko bežične pristupne tačke, a to su uređaji kao što su računari, smart telefoni, konzole za video igre, digitalni audio plejeri i sl. WiFi se odnosi na sve proizvode koji koriste IEEE802.11 porodicu standarda. IEEE802.11 porodica standard uključuje IEEE802.11 a/b/g/n. WiFi mreže rade u kombinaciji sa TCP/IP standardom. WiFi se često koristi u oblastima gde nije praktično ili je teško da se sprovedu kablovi. **Primer takvih mesta su lokacije gde ima mnogo različitih i privremenih korisnika, biblioteke, aerodromi, kafići i sl.**

Porodica standarda 802.11	
Standard	Brzine prenosa
802.11a	up to 54Mbps
802.11b	up to 11Mbps
802.11g	up to 54Mbps
802.11n	up to 150Mbps

Slika 5.1 Tabela-1 Brzine prenosa za IEEE802.11 standard

BLUETOOTH

Bluetooth je protokol koji se koristi za bežični prenos kratkog dometa.

Bluetooth je protokol koji se koristi za bežični prenos kratkog dometa. Podaci se prenose sa jednog uređaja na drugi sa brzinom prenosa od 3Mbps. Da bi uređaji mogli da komuniciraju preko Bluetooth-a, neophodno je da oni budu na rastojanju do 10 metara. To rastojanje nekada može biti i do 100 metara, ali uz korišćenje dodatne opreme. Takođe, uređaj koji je ima Bluetooth može da komunicira samo sa drugim uređajima koji takođe imaju Bluetooth. Takvi uređaji su obično **notebook računari, pametni telefoni, slušalice, miševi, tastature, digitalne kamere, GPS, i štampači**.

Ranije je Bluetooth bio namenjen samo za mobilnu telefoniju, a sada i za povezivanje računara i drugih digitalnih uređaja. Frekvencijski opseg ovog standarda je od 2.4 GHz do 2.5 GHz, domet je do 100m, a maksimalna brzina iznosi 720 Kb/s. Kada govorimo o prenosu podataka, danas se prenos ostvaruje umesto malog jeftinog radija, uz pomoć mikročipa koji se ugrađuje u uređaje poput tastature, slušalice, mobilnih telefona i sl.

Tehnika koju Bluetooth uređaji koriste se zove zamena paketa. Na taj način, podaci su izdeljeni u manje grupe, odnosno pakete. Tako se više paketa može preneti preko različitih frekvencija i u različitom redosledu. Tek kada se svi paketi prime, celina se formira na strani primaoca. Sa druge strane, kada pričamo o prenosu glasa, tada se koristi druga tehnika pod nazivom tehnika prekidanja. U ovom slučaju nema deljenja na pakete, već dolazi do uspostavljanja kanala za vreme trajanja transmisije. Ono što bi još bilo važno dodati jeste da Bluetooth uređaji mogu u isto vreme koristiti zamenu podataka i tehniku prekidanja, i na taj način istovremeno imaju mogućnost da prenose podatke i glas.

Pored ovih spomenutih tenika, postoji još jedna tehnika koju Bluetooth koristi a zove se tehnika širenja frekvencijskog opsega. Ova tehnika omogućava da se signal prebacuje sa jedne na drugu frekvenciju posle svake transmisije. U suštini, Bluetooth stalno menja frekvencije unutar ISM

opsega nakon primanja i slanja svakog paketa podataka. Na taj način se izbegava interferencija sa drugim uređajima, kao i smetnje od strane drugih uređaja koji koriste isti opseg.

RFID I WIMAX

Najvažnija karakteristika WiMAX-a je to što omogućava širokopojasni pristup udaljenim Internet uređajima

RFID

RFID (Radio Frequency Identification) je protokol koji definiše kako se radio signali koriste u komunikaciji sa uređajima koji imaju RFID tag. RFID tagovi se mogu ubaciti u objekte, životinje, pa čak i ljudi. RFID tagovi se nazivaju transponderima (engl. **transponders**), a sastoje se od antene i memorijskog čipa. RFID čitač (engl. **transceiver**) čita prenute informacije i prosleđuje obrađene informacije dalje uređaju (najčešće računaru).

Kada je u blizini odgovarajućeg čitača, RFID (Radio Frequency Identification) čip reaguje i beskontaktno, bez baterija, javlja svoj jedinstveni kod tj. broj. Čip se najčešće ugrađuje u neki mali predmet, privezak ili karticu, čime svaki takav predmet postaje jedinstven. Ograničenja u pogledu primene ove tehnologije ogledaju se u vidu količine energije koja dopire do nalepnice od čitača i u pogledu osetljivosti čitača na povratni signal.

WiMAX

WiMAX (Worldwide Interoperability for Microvawe Access) je standard koji je razvijen od strane IEEE, a zove se **IEEE802.16**. WiMAX uređaji komuniciraju pomoću radio talasa. Najvažnija karakteristika WiMAX-a je to što omogućava širokopojasni pristup udaljenim Internet uređajima, uključujući seoske i udaljene oblasti.

Jedna od osnovnih grešaka koja se potkrada ljudima jeste da WiMAX poistovećuju sa WiFi-jem. U svojoj osnovi, WiFi (802.11b) je zamišljen kao bežični način povezivanja malih lokalnih mreža (WLAN) na malom rastojanju.

NFC

NFC (Near Field Communication) je kratko-dometna tehnologija bežične komunikacije

NFC (Near Field Communication) je kratko-dometna tehnologija bežične komunikacije. O NFC se može razmišljati na isti način kao i o Bluetooth-u, WiFi-u, 3G ili 4G mreži, ali treba imati u vidu da NFC koristi drugu frekvenciju, drugi nivo električnog napajanja i komunikacioni protokol koji drugačije izvršava slanje i primanje informacija.

NFC čipovi mogu biti aktivni i pasivni. Aktivni NFC čipovi se aktiviraju, odnosno šalju informaciju, samo onda kada su stimulisani od strane NFC čitača. Aktivni NFC čipovi mogu da šalju i primaju informacije nakon neke komande. Pasivni NFC čipovi šalju informaciju koja je prethodno sačuvana negde na samom čipu, dok aktivni NFC čipovi šalju informacije koje se nalaze sačuvane na nekoj memoriji van samog NFC čipseta.

Kako su pasivni NFC čipovi stimulisani samo od strane NFC čitača, tako oni ne zahtevaju da imaju eksterno napajanje, već oni sami generišu energiju kroz elektromagnetnu indukciju koja nastaje putem radio talasa generisanih od samog NFC čitača. Pasivni NFC čipovi mogu samo da šalju podatke. Onog trenutka kada se čip aktivira i kada se obavi autentifikacija, procesor i već unete programske komande daju instrukcije NFC čipu da može da pošalje potrebne podatke.

Za razliku od pasivnih, aktivni NFC čipovi moraju da imaju eksterno napajanje, koji su tipično već integrисани u uređaj, kao što je smart telefon. Aktivni čipovi šalju informacije samo kada su za to dobili instrukciju od nekog eksternog programa.

Osim načina rada, ovi čipovi se razlikuju po svojoj veličini. Pasivni čipovi su dosta manji, a aktivni veći i obično su deo nekog većeg sistema koji ima procesor, memoriju, komunikacione kontrolere, antenu itd.

NFC koristi prenosnu frekvenciju 13,56 MHz, dok mu je prenosna brzina od 100 do 450 Kbs. Iako ova brzina nije dovoljna da se možda prenese HD video, sasvim je dovoljna da se prenesu manji podaci, kao što je na primer lista kontakata. Domet NFC je oko 20 cm.

Primer NFC primene je beskontaktno plaćanje.

▼ Poglavlje 6

Pokazna vežba: Cisco Packet Tracer

CISCO PACKET TRACER

Packet Tracer omogućava simulaciju kompleksnih mreža koje sadrže razne mrežne komponente kao što su hubovi, routeri itd.

Predviđeno vreme pokaznih vežbi je 110 minuta.

Cisco Packet Tracer je alat za simulaciju računarskih mreža. Packet Tracer omogućava simulaciju kompleksnih mreža koje sadrže razne mrežne komponente kao što su hubovi, routeri itd. Podržana je simulacija postojećih Cisco uređaja, sa funkcionalnim firmware-om, komandnim shell-om i fizičkim komponentama koje se mogu priključiti na date uređaje.

Packet Tracer podržava simulaciju najvažnijih protokola aplikacionog sloja, kao i protokola i algoritama rutiranja.

OSNOVNI POJMOVI

Prvi deo IP adrese označava mrežu a drugi deo označava poseban uređaj na mreži.

IP adresa je niz od 32 bita koji dodeljen svakom uređaju na mreži koja koristi IP (Internet Protokol). IP adresa se sastoji od dva dela. Prvi deo adrese označava mrežu a drugi deo označava poseban uređaj na mreži. Adrese se obično pišu u formatu sa brojevima odvojenim tačkama, npr. 192.168.1.2. Ovakav tip adresa su IPv4 adrese.

Postoji i nova iteracija IP standarda, IPv6 koja koristi adrese od 128 bita.

Subnet Mask je niz bitova koji govori koji se deo IPv4 adrese odnosi na podmrežu, a koji na poseban host. Npr. subnet maska 255.255.255.0 govori da se prva 24 bita odnose na mrežu, a ostali bitovi na adresu hosta.

MAC adresa je adresa fizičkog sloja. Svaki mrežni interfejs ima jedinstvenu MAC adresu.

DHCP je protokol za dinamičko konfigurisanje hostova. Pomoću ovog protokola DHCP server može dodeliti IP adrese, DNS i gateway podešavanja dinamički svim hostovima na mreži. Ovo se koristi kao česta alternativa statičkom podešavanju IP adresa.

POVEZIVANJE UREĐAJA

Posle postavljanja uređaja na radni prostor, potrebno je spojiti uređaje u mrežu koristeći mrežne kablove

Glavni deo radnog prostora u Packet Traceru je prikaz simulirane mreže. Postoje dva taba koja prikazuju logički i fizički raspored mreže. U ovoj vežbi, koristićemo logički prikaz mreže.

Donji deo radnog prostora zauzima paleta uređaja. Uređaj se u mrežu dodaje izborom uređaja iz palete, i klikom na glavni deo radnog prostora. Uređaji su grupisani u sledeće grupe po funkciji: routers, hubes, switches, end devices, wireless devices itd.

Dupli klik na uređaj u mreži otvara prozor sa detaljima uređaja. Ovde je moguće promeniti podešavanja uređaja, fizičku konfiguraciju, pristupiti komandnoj liniji itd.

Fizička konfiguracija uređaja predstavlja skup mrežnih interfejsa, tj. mrežnih adaptera. Uređaji, zavisno od tipa dolaze sa već ugrađenim mrežnim interfejsima, a dodatni delovi se mogu dodati prevlačenjem adaptera iz liste na slobodan slot na grafičkom prikazu uređaja. Pre ove akcije, potrebno je isključiti uređaj klikom na power dugme.

Pod tabom Config, moguće je podesiti IP i MAC adrese.

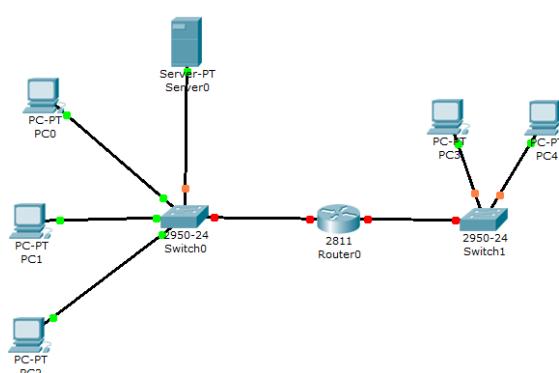
Posle postavljanja uređaja na radni prostor, potrebno je spojiti uređaje u mrežu koristeći mrežne kablove. Potrebno je izabrati željeni tip mrežnog kabla (serial, coaxial, cross over itd.) iz grupe Connections, a potom kliknuti na dva uređaja koja želimo spojiti. Moguće je koristiti prvu opciju iz liste da se automatski izabere tip kabla.

PODEŠAVANJE RUTERA

Ruter ima dva interfejsa, i potrebno je prvo podesiti IP adresu za oba.

U ovom primeru, simuliraćemo dve male mreže sa nekoliko uređaja.

Na slici je prikazana mreža koju simuliramo.



Slika 6.1 Simulirana mreža

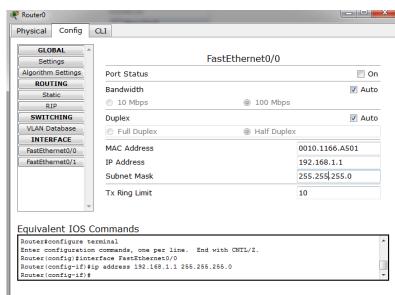
Mreža je podeljena u dve podmreže, sa leve i desne strane 2811 ruter. Ruter je na oba interfejsa spojen sa 2950-24 switchem, na koji su spojeni krajnji uređaji.

Posle postavljanja i spajanja uređaja kao na slici, potrebno je podešiti uređaje.

Ruter ima dva interfejsa, i potrebno je prvo podešiti IP adresu za oba.

Levi deo mreže ćemo označiti sa 192.168.1.0 a desni sa 192.168.2.0, a interfejs ruteru sa 192.168.1.1 i 192.168.2.1.

Na slici 2 se vidi podešavanje adrese interfejsa ruteru.



Slika 6.2 Podešavanje adrese interfejsa ruteru

PODEŠAVANJE TABELE RUTIRANJA

Posle pokretanja simulacije, ruter će koristeći RIP obavestiti druge ruteru o mrežama u koje on može da rutira pakete.

Sledeće što je potrebno podešiti je tabela rutiranja. U ovom slučaju koristimo dinamičko rutiranje pomoću RIP protokola.

Pod tabom Routing -> Rip, potrebno je dodati mreže u koje ovaj ruter može da rutira pakete. U ovom slučaju to su mreže 192.168.1.0 i 192.168.2.0.

Posle pokretanja simulacije, ruter će koristeći RIP obavestiti druge ruteru o mrežama u koje on može da rutira pakete. Ostali ruteri će na osnovu tih obaveštenja ažurirati svoju tabelu rutiranja.

DHCP SERVER

Posle podešavanja DHCP servera, moguće je podešiti ostale hostove na podmreži da koriste DHCP za automatsku konfiguraciju.

Server u mreži 192.168.1.0 će imati ulogu DHCP servera. Prvo je potrebno podešiti server sa statičkom IP adresom, klikom Config-> Fast Ethernet. Serveru ćemo dodeliti adresu 192.168.1.2, i subnet masku 255.255.255.0. Sledеće, pod stavkom DHCP potrebno je podešiti DHCP servis. Defaultnu pool adresu ćemo podešiti tako da je početna adresa 192.168.2.3 i subnet mask 255.255.255.0. Default gateway će biti adresa našeg ruteru, tj. 192.168.1.1.

Posle podešavanja DHCP servera, moguće je podesiti ostale hostove na podmreži da koriste DHCP za automatsku konfiguraciju. Ovo se postiže klikom na svaki host i promenom podešavanja u Config -> Fast Ethernet prozoru. Potrebno je postaviti sa statičke na DHCP.

Možemo proveriti da li su hostovi dobili ispravne adrese klikom na Desktop -> IP Configuration stavku.

IP adrese dva hosta u mreži 192.168.2.0 podesiti na statičke adrese 192.168.2.2 i 192.168.2.3 respektivno.

POKRETANJE SIMULACIJE MREŽE

Da bismo generisali saobraćaj i testirali mrežu koristićemo alat ping

Sada je moguće pokrenuti simulaciju mreže. Da bismo generisali saobraćaj i testirali mrežu koristićemo alat ping. Klikom na Desktop -> Command prompt otvara se komandni shell hosta. Komanda ping ip_adresa će poslati ICMP ping na datu adresu. Testirati rad mreže pingovanjem hostova iz jedne podmreže sa hostovima iz druge podmreže.

CISCO PACKETTRACER TUTORIJALI

Dopunski materijal

Kako bi se bolje upoznali sa simulatorom pogledajte uvodne tutorijale

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

DHCP tutorijal

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

DNS+DHCP+HTTP

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 7

Zadatak za samostalni rad: Kreiranje mreže

ZADATAK ZA SAMOSTALNI RAD

PacketTracer

Predviđeno vreme izrade zadatka je 25 minuta.

Koristeći PacketTracer napraviti prezentaciju mreže po izboru. Mreža mora da sadrži bar tri podmreže i 6 računara. Na jednoj podmreži konfigurisati DHCP server.

✓ Poglavlje 8

Domaći zadatak

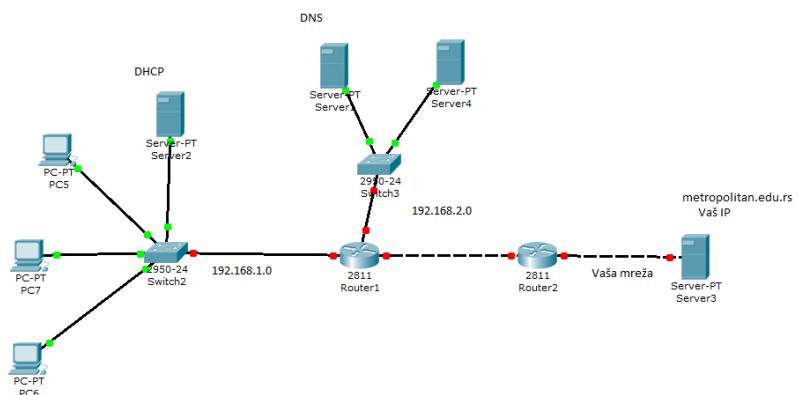
OPIS DOMAĆEG ZADATKA - DZ06

U Packet Tracer-u kreirati mrežu kao na slici 1

Očekivano vreme izrade zadatka: 45min.

Kreirati mrežu kao na slici 1. Mreža se sastoji od tri podmreže: 192.168.X.Y (X predstavlja treći broj Vašeg indeksa, dok Y predstavlja četvrti broj Vašeg indeksa), 192.168.2.0 i mreža sa proizvoljnom adresom podmreže (vaša mreža). Server u prvoj mreži radi kao DHCP server sa statičkom adresom 192.168.X.2. Zadati adresu DNS servera u DHCP podešavanjima.

Ruteri koriste RIP protokol.



Slika 8.1 Mreža

- **Domaći zadatak pošaljite predmetnom asistentu na e-mail, a u subject-u mejla napisati IT101 - DZ06**
- Zadatak dostaviti kao **IT101-DZ06-Ime_Prezime_BrojIndeksa**
- Potrebno je dostaviti i dokument koji opisuje kako ste svaku od navedenih koraka rešili, kako ste podešili sve čvorove u mreži. Ovaj dokument dostaviti kao **IT101-DZ06-Ime_Prezime_BrojIndeksa.doc**

▼ Zaključak

ZAKLJUČAK

U ovom predavanju smo dali osnovnu definiciju računarskih mreža i sistema. Naveli smo primere kada je korisni biti umrežen, odnosno koje su generalne prednosti umrežavanja.Zatim, definisali smo šta podrazumevamo kada kažemo da su dva sistema umrežena, odnosno kada komuniciraju.Kako bi dva mrežna čvora mogla da komuniciraju potrebno je da se podaci prenesu preko nekog prenosnog medijuma (na primer kablovi ili bežični prenos) i da se poštuju određena pravila koja su određena protokolima i standardima.U ovom predavanju je bilo reči o žičnim i bežičnim prenosnim medijumima, mrežnoj opremi i često korišćenim mrežnim standardima i tehnologijama.

Literatura

1. Gary B. Shelly, Misty E. Vermaat, Discovering Computers 2011-Introductory: Living in a Digital World, Cengage Learning 2010.



IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

RAZVOJ VEB SAJTOVA

Lekcija 07

PRIRUČNIK ZA STUDENTE

IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

Lekcija 07

RAZVOJ VEB SAJTOVA

- ▼ RAZVOJ VEB SAJTOVA
- ▼ Poglavlje 1: Razvoj veb sajtova
- ▼ Poglavlje 2: Tipovi veb sajtova
- ▼ Poglavlje 3: Multimedija na vebu
- ▼ Poglavlje 4: HTML
- ▼ Poglavlje 5: Pokazna vežba: Izrada HTML dokumenta
- ▼ Poglavlje 6: Pokazna vežba: Povezivanje HTML dokumenata
- ▼ Poglavlje 7: Pokazna vežba: Stilovi u HTML dokumentu
- ▼ Poglavlje 8: Zadatak za samostalni rad: HTML
- ▼ Poglavlje 9: Domaći zadatak
- ▼ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

✓ Uvod

UVOD

Cilj ovog predavanja je da da pregled osnovnih veb standarda i tehnologija

U ovom predavanju će biti obrađene sledeće teme:

- Razvoj veb sajtova
- Tipovi veb sajtova
- Multimedije na vebu
- Osnove kreiranja HTML dokumenta

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

✓ Poglavlje 1

Razvoj veb sajtova

OSNOVNI KORACI U RAZVOJU VEB SAJTOVA

Najvažniji deo razvoja veb sajta je projektovanje sajta, i to kako sa aspekta usluga i sadržaja koji će se publikovati tako i sa aspekta njegovog izgleda

Razvoj veb sajta se ne sastoji samo od izbora hardvera, operativnog sistema i veb servera. Najvažniji deo posla je projektovanje sajta, i to kako sa aspekta usluga i sadržaja koji će se publikovati tako i sa aspekta njegovog izgleda. Većina današnjih sajtova liči na brošure sa sadržajem koji je neretko neažuran. Dinamičko formiranje veb stranica pruža mnoge nove mogućnosti, ali i zahteva mnogo ozbiljniji pristup razvoju veb sajta i održavanju njegove ažurnosti.

Razvoj veb sajtova ima logički sled koraka koji će nadalje biti opisani, a to su:

- **Definisanje ciljeva i problema**
- **Korisnici**
- **Definisanje zahteva**
- **Izrada prototipa**
- **Pisanje koda**
- **Validacija**
- **Testiranje**
- **Održavanje sajta**

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

DEFINISANJE CILJEVA I PROBLEMA

Na početku je potrebno jasno definisati koji su ciljevi sajta

Različite organizacije se bave različitim oblastima poslovanja pa su i **ciljevi razvoja veb sajta** različiti. U nekim slučajevima je to samo bolje informisanje klijenata, ali u nekim slučajevima sajtovi predstavljaju sredstvo preko koga se obavljaju poslovne transakcije. **Zato je na početku potrebno jasno definisati koji su ciljevi sajta.** Definišite jasno koji je cilj veb sajta kako bi vam kasniji koraci u procesu razvoja sajta bili lakši. Dakle, ako na primer razvijate sajt za prodaju dečiji igračaka, onda treba jasno definisati da li je cilj da se uradi promocija novih igračaka ili da se privuku novi kupci (deca ili roditelji). Naravno, nije

isključeno da ćete možda imati i oba cilja u ovom slučaju na svom spisku. U svakom slučaju od tog cilja će dosta zavisiti kasnije dizajn i organizacija podataka.

U svakom slučaju sajt ne može nikada da pruži sve usluge za sve korisnike, iako je na žalost ovo česta greška kod početnika koji tek počinju da se bave razvojem veb sajtova. Preglomazan veb sajt na kome se ne mogu naći potrebne informacije ili servisi neće koristiti nikom. Zbog toga treba jasno ograničiti njegov delokrug i, ukoliko je to potrebno, napraviti više sajtova za istu organizaciju.

KORISNICI

U procesu razvoja veb sajta važno je definisati ciljnu grupu koja će koristiti taj sajt, odnosno koji je profil njenih korisnika

Sledeće važno pitanje je ko su korisnici sajta i kakav je njihov profil. Da li su to zaposleni u organizaciji, klijenti ili slučajni posetioci. Važna pitanja su i koliko su oni stari, kog su pola, kakva je njihova IT kultura, koji jezik govore, kakvu konekciju sa Internetom koriste. Pored toga treba napraviti istraživanje o tome koje veb čitače korisnici upotrebljavaju. Većini veb sajtova se pristupa sa desktop računara koji imaju velike displeje, ali postoje i sajtovi kojima se uglavnom pristupa sa mobilnih uređaja, koji po pravilu imaju vrlo male displeje. Na kraju, ali ne i manje važna su pitanja zbog čega će korisnici koristiti sajt, kako će ga naći i koliko često će mu se vraćati. Dobra osnova za elaboraciju ove teme može da bude statistika korišćenja prethodne verzije veb sajta ako je postojao. Za poslovne sajtove je uputno razgovarati sa potencijalnim korisnicima kako bi se snimile njihove želje i potrebe.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

DEFINISANJE ZAHTEVA I IZRADA PROTOTIPA

Na osnovu definisanih ciljeva i korisnika za sajt, sledeći korak podrazumeva definisanje zahteva

Definisanje zahteva

Na osnovu saznanja iz prethodna dva koraka treba formulisati zahteve koje treba da ispunи veb sajt. Ovaj dokument treba da odgovori na sledeća pitanja:

- koji će sadržaj biti publikovan i na koji način
- kako će sadržaj biti organizovan, u smislu koji će sadržaji biti prikazivani na pojedinim stranama
- kako će se organizovati navigacija unutar sajta
- ko će pripremati sadržaj i ko će održavati veb server
- kakav će biti dizajn, odnosno izgled sajta
- koje će usluge nuditi sajt
- koliko i kojih servera treba instalirati u hardverskom i softverskom smislu i gde će se oni nalaziti

- kakav je vremenski plan realizacije
- koliko će koštati postavljanje i održavanje sajta
- kako će se evidentirati posetnici sajta itd.

Izrada prototipa

Na osnovu prethodno definisanih zahteva potrebno je razviti tehnički i vizuelni prototip sajta. U isto vreme treba pripremiti što više sadržaja, jer je testiranje sajta moguće jedino na realnom sadržaju.

PISANJE KODA

Za kreiranje HTML sadržaja se mogu koristiti različite tehnike koje nimalo ne automatizuju, malo ili više automatizuju pisanje HTML koda

Proizvodnji HTML stranica se mora pristupiti vrlo pažljivo jer to treba da bude stabilna osnova na kojoj će se nadgraditi prezentacija i interaktivnost. Za kreiranje HTML sadržaja se mogu koristiti različite tehnike koje nimalo ne automatizuju, malo ili više automatizuju pisanje HTML koda. U tabeli 1 dat je pregled mogućnosti za pisanje HTML stranica. Na raspolaganju su četiri različite tehnike, ali nijedna nije bez nedostataka. U različitim prilikama svaka od tehnika može biti najpodesnija, tako da ih treba kombinovati.

Metod	Alat	Prednost	Nedostatak
Ručno	Notepad	Potpuna kontrola u pisanju HTML koda	Spor razvoj Podložan greškama Zahteva odlično znanje HTML Nema vizuelne prezentacije izmenjenog sadržaja u istom momentu kada se izmeni nešto
Prevodenje	Save HTML file from Word	Brza i jednostavna konverzija postojećih dokumenata	Dobijena stranica je obično puna grešaka Zahteva ručnu doradu kako bi se dobio traženi izgled
HTML editori	HomeSite	Potpuna kontrola nad pisanjem HTML koda Brži od ručnog metoda	Razvoj HTML koda može biti spor Zahteva odlično znanje HTML-a
WYSIWYG editori	FrontPage	Veb stranice se kreiraju u vizuelnom okruženju Ne zahteva znanje HTML-a	Često generiše netačan kod

Slika 1.1 Tabela-1 Pregled mogućnosti pisanja HTML stranica

VALIDACIJA, TESTIRANJE I ODRŽAVANJE SAJTA

Testiranje treba da potvrди da sajt ima sadržaj, funkcionalnost i izgled kakav je zadat specifikacijom, kompatibilnost sa različitim veb čitačima i prihvatljivu brzinu odziva

Validacija

Bez obzira na to kako su kreirani HTML dokumenti, uvek postoji opasnost od postojanja grešaka kao što su izostavljanje tagova, neodgovarajući atributi, loši linkovi i slično. Proces validacije ne podrazumeva samo nalaženje i otklanjanje grešaka kodiranja nego i proveru specificiranog dizajna, funkcionalnosti i dostupnosti.

Testiranje

Nakon instalacije hardverskih i softverskih komponenti veb sajta i nakon potpune pripreme celokupnog sadržaja treba pristupiti testiranju sajta. Testiranje treba da potvrdi da sajt ima sadržaj, funkcionalnost i izgled kakav je zadat specifikacijom, kompatibilnost sa različitim veb čitačima i prihvatljivu brzinu odziva. Međutim, testiranje sajta treba da pokaže i u kom stepenu su potencijalni korisnici sajta zadovoljni njime. Ovo treba uraditi pre puštanja sajta u javnost kako bi se na vreme otklonili svi nedostaci.

Održavanje sajta

Dobar veb sajt nikad nije završen. Korisnici će retko posećivati sajt čiji je sadržaj nepromenljiv. Zato je potrebno stalno raditi na inoviranju sadržaja, poboljšanju funkcionalnosti i dizajna.

PLANIRANJE WEB SAJTA - PRIMER

Primer lego.com - cilj, korisnici i sadržaj

Prva faza u izradi sajta je planiranje. Potrebno je jasno definisati koji problem sajt treba da reši, tj. šta je svrha sajta. Često ni sami poručiocci sajta nisu sigurni šta tačno žele, pa zahtevaju da sajt rešava više problema, tj. ima mnogo različitih funkcija. Najčešća teškoća je kada su ti zahtevi veoma apstraktni, pošto ni sami naručiocci nisu sigurni kako i da li nešto može tehnički da se izvede.

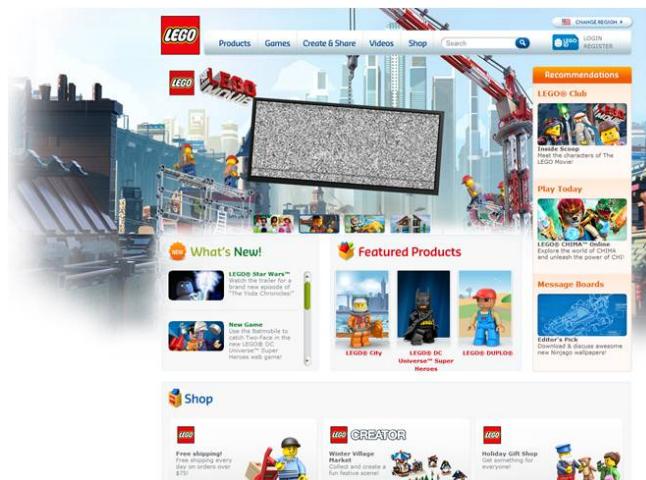
S druge strane, preglomazni sajтови mogu samo zbumnjivati korisnike. Previše sadržaja na jednoj stranici može da ima i kontra efekat. Kako bi izbegli konfuziju korisnika sajta, u ovom slučaju, potrebno je izraditi više stranica, a ponekad i više sajtova. Velike kompanije obično imaju posebne sajtove za različite vrste usluga koje pružaju, a nije retkost ni da se izradi novi sajt za svaki proizvod koji kompanija ima.

Posle definisanja problema koji sajt rešava, potrebno je odrediti ko će biti korisnici sajta. Potrebno je odrediti starosnu grupu korisnika, njihov pol, koji jezik govore itd.

Izrađuje se dokument zahteva kojim se definiše sadržaj sajta, dizajn sajta, vrste navigacije itd.

Kao primer uzećemo sajt lego.com.

Cilj ovog sajta je prezentovanje i prodaja proizvoda kompanije Lego. Korisnici sajta su deca i roditelji. Glavni sajt je na engleskom jeziku i namjenjen je tržištu SAD, a postoji i nekoliko lokalizacija sajta. Sadržaj sajta čine strane sa opisima Lego proizvoda i igre koje su organizovane po temama, npr. City, Star Wars, LOTR itd. Svaki od proizvoda ima nekoliko jednostavnih mini igara koje se izvršavaju direktno u web browseru. Opisi proizvoda i likova iz svake kategorije su očigledno pisani za decu.



Slika 1.2 Lego.com

PRIMER LEGO.COM – NAVIGACIJA I IZGLED

Lego.com – navigacija, izgled i diskusija

Navigacija na glavnom sajtu je realizovana pomoću tabova koji vode na odgovarajuće delove sajta. Navigacija tabovima je dobro rešenje kada broj linkova nije veliki. Neke stranice imaju i drugi nivo tabova. Na primer, strana sa igrama ima dodatne tabove za podelu po žanrovima.

Deo sajta sa web prodavnicom ima znatno kompleksniju šemu navigacije. Postoje dva nivoa tabova sa padajućim menijima, filteri po temama, kategorijama, ceni, uzrastu itd. Ovaj deo sajta je namjenjen odraslima.

Na sajtu dominiraju svetle, drečave boje, sa mnogo stilizovanih elemenata i animiranih likova. Pozadina sajta je prilagodjena kategoriji igračaka koja se prikazuje.

Pitanja za diskusiju vezana za sajt lego.com:

1. Koji je cilj sajta?
2. Kojim korisnicima je sajt prevashodno namenjen? Da li su sve stranice ovog sajta namenjene istim korisnicima?
3. Na osnovu prethodna dva odgovora, pokušajte da sastavite zahteve koji su bili postavljeni za ovaj sajt u fazi definisanja zahteva:
4. koji će sadržaj biti publikovan i na koji način
5. kako će sadržaj biti organizovan, u smislu koji će sadržaji biti prikazivani na pojedinim stranama
6. kako će se organizovati navigacija unutar sajta

7. ko će pripremati sadržaj i ko će održavati veb server
8. kakav će biti dizajn, odnosno izgled sajta
9. koje će usluge nuditi sajt
10. koliko i kojih servera treba instalirati u hardverskom i softverskom smislu i gde će se oni nalaziti
11. kakav je vremenski plan realizacije
12. koliko će koštati postavljanje i održavanje sajta
13. kako će se evidentirati posetioci sajta itd.
14. Na koji način bi organizovali testiranje ovog sajta pre nego što je pušten u javno korišćenje?

ANALIZA SAJTA KUPIME.COM

Pitanja za diskusiju sajta kupime.com

Pitanja za diskusiju za sajt kupime.com

1. Koji je cilj sajta?
2. Kojim korisnicima je sajt prevashodno namenjen? Starost? Pol? Da li su sve stranice ovog sajta namenjene istim korisnicima?
3. Na osnovu prethodna dva odgovora, pokušajte da sastavite zahteve koji su bili postavljeni za ovaj sajt u fazi definisanja zahteva:
4. koji će sadržaj biti publikovan i na koji način
5. kako će sadržaj biti organizovan, u smislu koji će sadržaji biti prikazivani na pojedinim stranama
6. kako će se organizovati navigacija unutar sajta
7. ko će pripremati sadržaj i ko će održavati veb server
8. kakav će biti dizajn, odnosno izgled sajta
9. koje će usluge nuditi sajt
10. koliko i kojih servera treba instalirati u hardverskom i softverskom smislu i gde će se oni nalaziti
11. kakav je vremenski plan realizacije
12. koliko će koštati postavljanje i održavanje sajta
13. kako će se evidentirati posetioci sajta itd.
14. Na koji način bi organizovali testiranje ovog sajta pre nego što je pušten u javno korišćenje?
15. Koje su specifični tehnički zahtevi za ovaj sajt?

ISPLANIRAJTE SVOJ SAJT

Dajte primer i analizu svog sajta

Isplanirajte svoj sajt. Tema sajta treba da plasira neki proizvod ili uslugu koju bi vi isplanirali kao budući IT profesionalac. Dakle, naglašavamo ovde da sajt ne treba da bude lična promocija. Isplanirajte sajt po sledećim fazama:

- definisanje ciljeva i problema
- ko su korisnici i koje su njihove potrebe
- definišite zahteve sajta
- napravite prototip (bez kodiranja)
- napravite plan testiranja i održavanja sajta

Napomena: ovde preskačemo deo izrade sajta.

VIDEO

Starting A Serious Web Development Project

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

✓ Poglavlje 2

Tipovi veb sajtova

VRSTE SAJTOVA - PORTAL, VESTI, INFORMACIONI SAJTOVI

Primeri portala, vesti i informacionih sajtova

Sajtovi mogu da budu kreirani za različite namene i za različite ciljne grupe. Neke vrste sajtova su:

- Portal – veb sajt koji pruža raznovrstan sadržaj i servisa kao što su pretraživanje, vesti, sport, vreme, kupovina, email, akcije na berzi i dr. Popularni portali su Yahoo, MSN, Lycos, Netvibes i sl.



Slika 2.1 Portal Yahoo

- Vesti – veb sajt koji sadrži članke, odnosno vesti, o trenutnim događajima, životu, sportu, vremenu i sl. Neke novine objavljaju kraće verzije svojih članaka u odnosu na štampano izdanje. Neki to čine besplatno, a neki za mesečnu pretplatu.



Slika 2.2 Sajt CNN-a

- **Informacioni sajtovi** – sadrži informacije o određenoj temi, ili informacije koje se odnose na određenu oblast interesovanja. Na primer, sajt može da pruži informacije o svim podacima koji se odnose na poreze, pravdanje poreza, formulare koji su neophodni za prijavu poreza, poreske kodove itd. Takođe, primer može biti sajt koji sadrži sve informacije o javnom prevozu u gradu uključujući red vožnje, cene karata, informacije za odeljenje za žalbe i dr.



Slika 2.3 Vefsajt "Internal Revenue Service" koji sadrži informacije o porezima u Americi

VRSTE SAJTOVA – POSLOVNI SAJT, BLOG, OBRAZOVNI SAJT

Primeri poslovnog sajta, bloga i obrazovnog sajta

- **Poslovni sajtovi** – sadrži materijal koji promoviše određene proizvode ili proizvode i usluge koje nudi određena firma. Skoro svaka firma ima svoj sajt koji su informativnog karaktera, koji služe za marketing, a ponekad i za prodaju svojih proizvoda i online servisa.



Slika 2.4 Vebajt Apple-a

- **Blog** – predstavlja neformalni sajt na kome su postavljeni članci. Ovi članci su napisani o različitim temama i napisani su u različitim formama, tako da se razlikuju od članaka koji se objavljaju u časopisima. Ovi članci su obično hronološki izloženi u zavisnosti od vremena kada su objavljeni. Blogovi mogu biti lični, ali i kompanije mogu da kreiraju blogove o aktuelnim događajima, temama od interesa za svoje zaposlene, vestima o proizvodima itd.



Slika 2.5 Blog Microsofta

- **Obrazovni sajt** – sadrži materijale za formalno ili neformalno učenje. Teme mogu da se kreću od "kako da popravim svoj automobil" do online obuke za zaposlene. Ovi sajtovi i njihovi sadržaji mogu biti napisani kao tutorijali, ili kao lekcije kada se radi o studentima. Često predavači koriste ovakve sajtove da postave svoje materijale koji se koristi kao osnovni ili kao dopunsku materijal za učenje kako bi unapredili nastavu u učionici.



Slika 2.6 Oracle-ov Java tutorijal

VRSTE SAJTOVA - ZA ZABAVU, PERSONALNI SAJTOVI, SOCIJALNE MREŽE

Primer socijalne mreže Second Life

- Sajtovi za zabavu – zabavni sajтови обично садрже интерактивни садржај као што је музика, игре, видео, чат и сл.
- Personalni sajtorvi – свако може да има свог личног сајта, на коме може да ради своју промоцију, на пример за тражење посла, или једноставно да користи тај сајт да подели своја животна искуства са целим светом на интернету.
- Socijalne mreže – представља онлайн јединицу која подстиче своје чланове да деле своје интересе, идеје, фотографије, музику, видео записе и друге садржаје на својим личним страницама. Већина друштвених мрежа као што је Facebook омогућује својим корисницима да комуницирају путем чата, да имaju преглед догађаја својих пријатеља у форми вести, као и друге услуге за комуникацију. У неким друштвеним мрежама корисници воде виртуелне животе и имају интеракцију са другим корисницима у том истом виртуелном окружењу. Пример таквог сајта је Second Life.



Slika 2.7 Prikaz Second Life-a

VRSTE SAJTOVA - ECOMMERCE, PORTFOLIO

Čisto korporativni veb sajt bez e-trgovine још увек може индиректно подстаки кориснике да нешто купују.

eCommerce

Veb локација за електронску трговину, иначе позната као Internet продавница, омогућава вам плаќање путем Interneta за производе или услуге. Prodavnice могу функционисати као самосталне veb странице или се комбинују са blogом или корпоративним veb страницама. Veb локација за e-trgovinu је свака veb локација која директно продава производ или услугу. На основу тога, помоћу veb локације за e-trgovinu можете додати производе или услуге у своју корпу и платити производе путем veb локације.

Na primer, čisto korporativna veb lokacija bez funkcije e-trgovine i dalje može indirektno da podstakne korisnike da nešto kupe, ali ne mogu prihvati nijedno plaćanje.

Portfolio

Poput fizičkog portfelja, ova vrsta veb lokacija se koristi za prikazivanje i promovisanje primera prethodnog rada. Prvenstveno ga koriste oni iz kreativne industrije, portfelj veb stranica može se koristiti kao CV, pokazujući vaše veštine da biste impresionirali klijente, kupce ili buduće poslodavce.

Da biste izgradili impresivan portfelj koji će vam pomoći da se istaknete, potreban vam je vodeći graditelj veb lokacija za dizajn, [Squarespace](#). Ova platforma ima najbolje alate za dizajn bilo kog graditelja veb stranica, koji vam omogućavaju da napravite veb lokaciju profesionalnog izgleda u nekoliko sati.

Crowdfunding

Grupno finansiranje je praksa finansiranja projekta ili poduhvata prikupljanjem malih količina novca od mnogo različitih ljudi. Ove vrste veb lokacija postaju glavni izvor za nove start-up poslove.

U prošlosti, jedini način da se finansira novi poslovni poduhvat bio je traženje velikih ulaganja od samo nekoliko ljudi (tzv. [Dragon's Den](#)). Ali ovih dana možete sa lakoćom kreirati [crowdfunding](#) stranicu - trebate samo da napravite video spot za svoj projekat, a zatim da odredite ciljni iznos i rok.

Korisnici Interneta koji veruju u ono na čemu radite uložiće vašu količinu novca. Takođe možete ponuditi podsticaje u zamenu za donacije, poput sniženih proizvoda ili VIP iskustava.

VRSTE SAJTOVA - VIDEO STRIMING, NEPROFITNI

Na stranicama za striming video snimaka poslednjih godina je opala njihova popularnost.

TV ili video striming

[Netflix](#) je zajedno sa sličnim sajтовima kao što je [NowTV](#) revolucionirao način na koji svet gleda televiziju. Ove veb lokacije za striming video snimaka poslednjih godina beleže veliku popularnost, pri čemu su veb lokacije kao [BBC iPlayer](#) i [All 4](#) predstavljale tradicionalnije primere ove posebne veb lokacije.



Slika 2.8 Netflix

Neprofitni

Neprofitne organizacije se ne mogu klasifikovati kao preduzeća na isti način, ali im i dalje treba veb lokacija. To će obično biti prilično jednostavno, podvlačeći šta se odnosi na neprofitnu organizaciju i pokazaće posetiocima kako mogu da se uključe, doniraju ili podrže na neki drugi način. Veb stranice sa neprofitnim organizacijama često imaju funkciju prikupljanja adresa e-pošte i stvaranje baze podataka o ljudima koji su zainteresovani da budu u kontaktu sa organizacijom putem biltena (engl. **newsletters**).

VIDEO

Understanding Video Streaming

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 3

Multimedija na vebu

ŠTA SU MULTIMEDIJE?

Multimedija je sve što možete čuti ili videti: tekst, knjige, slike, muzika, zvuk, CD, video snimci, DVD, audio, filmovi i drugo

Multimedija je sve što možete čuti ili videti: tekst, knjige, slike, muzika, zvuk, CD, video snimci, DVD, audio, filmovi, VRML i drugo.

Multimediji mogu biti u različitim formatima. Na Internetu se mogu pronaći mnogi od ovih oblika koji su ugrađeni u veb stranicu. Današnji čitači imaju podršku za mnogobrojne multimedijalne formate.

Multimedija se može koristiti za efikasno prenošenje informacija ljudima. Multimedije su donele fundamentalne promene u način na koji ljudi uče, igraju se i pronalaze informacije.

Kada ispitujete različite multimedijalne proizvode i dostupne platforme, važno je da ispitate njihovu efikasnost. Treba uzeti u obzir sledeće:

- **Jednostavnost pristupa i navigacije** - kada ste korisnik koji prvi put koristi multimedijalni proizvod, važno je da proverite koliko je taj proizvod pristupačan? Često dizajneri multimedija naprave proizvod koji odlično izgleda, ali je to isto tako i proizvod koji je suviše težak za rad.
- **Ciljna grupa i njihove ICT veštine** - proizvod treba da bude prilagođen krajnjem korisniku. Tako na primer, ako je proizvod namenjen deci, onda on mora da bude krajnje jednostavan za korišćenje
- **Adekvatnost sadržaja** - trebalo bi da na odgovarajući način angažuje korisnika, a istovremeno da sadržaj bude informativan i relevantan.
- **Korišćenje boja** - boja treba da doda vrednost proizvodu, a ne da odvraća pažnju korisnika.
- **Balans i mešavina komponenti** - trebalo bi da postoji dobra ravnoteža između teksta, grafike, zvuka, videa i animacije
- **Interaktivnost** - interaktivni elementi moraju biti dobro osmišljeni

Multimedijalni proizvodi su sada uobičajeni i mogu se koristiti iz raznih razloga, uključujući: e-učenje (obrazovanje), zabava, promotivne i reklamne svrhe, e-publikacije, modeliranje i simulacija, javne informacije.

Generalno, multimedije su pogodne da se koriste u većini situacija kada je potrebno da se podeli informacije elektronskim putem. Ne samo da multimedije mogu da obogate sadržaj, već pomažu i oko toga da čovek bolje zapamti neku informaciju. Čovek je u stanju da zapamti

oko 20% podataka ako ih je sam čuo, 40% ako ih je i čuo i video, a 75% ako ih je video, čuo i aktivno koristio.

MULTIMEDIJA NA VEBU – AUDIO, VIDEO, ANIMACIJA I SL.

Pomoću multimedija se obogaćuje sadržaj stranice, tako da je stranica interesantnija korisnicima

Većina veb sajtova pored hipertekta i hiperlinkova sadrži i multimedije kao što su grafike, audio, video, animacije i sl. Pomoću multimedija se obogaćuje sadržaj stranice, tako da je stranica interesantnija korisnicima, a samim tim im i Internet postaje interesantniji. Za preuzimanje multimedija je obično potrebno više vremena za preuzimanje, pošto njihov sadržaj čini datoteke većim.

U kontekstu veb stranica multimedija se koristi da poboljša izgled stranice, kao i da se poboljša korisničko iskustvo. Veb sadrži veliku količinu slika koje se koriste u komercijalne i nekomercijalne svrhe. Nezavisno od sadržaja, grafika koja se koristi za Veb je obično u JPEG, GIF i PNG formatu. JPEG je format koji kompresuje grafiku kako bi smanjio veličinu datoteke. Smanjenje datoteke omogućava da se koristi manje prostora u skladištenju. Važno je da veb sadržaj bude u manjem formatu kako bi se proces preuzimanja grafike na vebu ubrzao. S druge strane, kada se dizajnira stranica važno je da se veličina datoteke ne smanji previše pošto to obično smanji i kvalitet slike. Za izradu GIF datoteka se koriste tehnike za komprimovanje, tako da su one pogodne za grafiku koje jedino imaju nekoliko različitih boja. S druge strane, PNG poboljšava GIF format. Drugi formati koji se koriste za veb grafiku su BMP i TIFF.

Zvučna datoteka (audio) može da sadrži govor, muziku ili bilo koji drugi zvuk. Audio se može slušati na vebu, može se preuzeti na računar ili se može slušati putem "streaming"-a. "Streaming" omogućava korisnicima da slušaju audio u isto vreme kada se sam sadržaj isporučuje od strane provajdera. Ovakav način prenosa se koristi kada je potrebno da se sadržaj reprodukuje u kontinuitetu, a kada nije potrebno da se sačeka da se celu datoteku preuzme, na primer slušanja radio stanice u realnom vremenu preko Interneta. Kao i grafika, audio datoteke se takođe komprimuju za prenos preko veba. Ovo je neophodno kako se rad ne bi usporio. Na primer, audio datoteka u MP3 formatu će biti smanjena do 10 puta, a pri čemu se može očuvati kvalitet zvuka. Da bi se sluša audio datoteke na računaru, potrebno je imati adekvatan aplikativni softver koji se naziva "player". Neki od popularnih audio plejera su Windows Media Players, iTunes i RealPlayer.

DIGITALNI VIDEO

Niz digitalnih slika ili frejmova, koji se prikazuju određenom brzinom, čine digitalni video.

Digitalni video se sastoji od niza digitalnih slika ili frejmova, koji se prikazuju određenom brzinom. Iskustvo je pokazalo da se pri prikazivanju 25 i više slika u sekundi, sva kretanja na

filmu čine kontinualna. Zbog ovako velikog potrebnog broja slika u sekundi veličina digitalnih video zapisa je ogromna, pa se zapisi komprimuju.

Moving Pictures Expert Group (MPEG) je formirana 1988. kao telo ISO/IEC tehničkog komiteta za informacione tehnologije sa zadatkom da razvije standarde za kompresovanu reprezentaciju digitalnog videa i pridruženog zvuka. Cilj je bio da se razviju tri standarda:

- MPEG-1 za kodiranje videa VHS kvaliteta u rezoluciji 360 x 280 piksela i sa 30 slika u sekundi, pri brzini od 1,5 Mb/s, što je u to doba odgovaralo brzini čitanja CD-ROM-ova.
- MPEG-2 za kodiranje videa u kvalitetu digitalne televizije u rezoluciji 720 x 480 piksela i sa 30 slika u sekundi, pri brzini od 2 do 10 Mb/s.
- MPEG-3 za kodiranje videa u kvalitetu HDTV (**high definition TV**) pri brzini oko 40 Mb/s. Rad na ovom standardu je prekinut 1992. jer je ustanovljeno da se ciljevi mogu postići MPEG-2 tehnologijom.

Tokom rada na standardima MPEG -1 i 2 su se uočile dodatne potrebe za komprimovanje videa, pa je radna grupa predložila izradu novog standarda **MPEG-4** koji omogućuje efikasno memorisanje, prenos i manipulaciju multimedijalnim podacima. Glavna osobina MPEG-4 je da on omogućuje rad sa audio-video objektima (AVO) tretirajući ih nezavisno i dozvoljavajući njihovu kombinaciju na nekoj sceni. Drugi standardi koji se dosta koriste su AVI (**Audio Video Interleaved**), QuickTime, Windows Media Format, Real Video i Adobe Flash.

▼ Poglavlje 4

HTML

ČEMU SLUŽI HTML?

HTML služi za publikovanje dokumenata na World Wide Web-u a skraćenicu za hipertekstualni markap jezik, odnosno Hyper Text Markup Language

HTML predstavlja skraćenicu za hipertekstualni markap jezik, odnosno Hyper Text Markup Language. Iz samog naziva HTML-a se vidi da on nije programski jezik, već markap jezik (engl. markup language). HTML služi za publikovanje dokumenata na World Wide Web-u. HTML koristi tagove ili elemente, koji ustvari predstavljaju skraćenice i simbole koji ukazuju na to kako će stranica biti prikazana i kako je njen sadržaj organizovan. Veb strane mogu da se sastoje od teksta, tabela, lista, slike i elemenata za unos podataka. Svaki dokument može da sadrži i metainformacije koje ukazuju na sadržaj dokumenta. Unutar dokumenta se uobičajeno nalaze hiperlinkovi koji pomoći URI-ja daju vezu sa drugim veb stranicama. Dokument može da sadrži i ugnježdene skripte, aplete i/ili objekte.

Koristeći ove mogućnosti HTML-a, autori veb strana su u mogućnosti da:

- Publikuju veb dokumente koji imaju naslove, tekst, tabele, liste, slike i druge ugnezđene objekte kao što su matematički, zvučni ili video objekti
- Omoguće čitaocima veb strana da pristupe drugim informacijama koje su referencirane hiperlinkovima
- Uključe u veb strane forme, komandnu dugmad, izborne liste, opcione grupe i druge elemente koji služe za unos podataka i komandi.

VERZIJE HTML-A

Najnovija verzija HTML-a je HTML 5.0. HTML 5.0 dopunjuje HTML sa sintaktičkim svojstvima koje uključuju elemente <video>, <audio>, <header> i <nav>

Prva verzija HTML se zvala "HTML Tags". Međutim, prva verzija HTML koja je bila po IETF standardu je HTML 2.0 koja je objavljena 1995-te. Sledеća verzija HTML-a, HTML 3.2, je objavljen u skladu sa W3C preporukama. HTML verzija koja je bila u skladu sa standardima SGML DTD je bila verzija HTML 4.0, a koja je objavljena 1998-e godine. Njegova revidirana verzija bila je verzija HTML 4.01, koja je objavljena godinu dana kasnije. HTML verzija 4 je objavljena 1998. godine, a revidirana verzija HTML 4.01 je iz 1999. godine. Ova verzija HTML-a omogućuje kreiranje dokumenta na bilo kom jeziku i korišćenje različitih pisama. Time se

postiže efektivno indeksiranje dokumenata za mašine za pretraživanje, kvalitetnija tipografija i bolja konverzija iz teksta u govor. Posebna pažnja pri razvoju HTML-a je posvećena poboljšanju dostupnosti sadržaja osobama sa fizičkim ograničenjima.

Radi pojednostavljenja opisa veb strana, HTML 4.0 uvodi veću razliku između strukture dokumenta i prezentacije, ohrabrujući tako upotrebu stilova (engl. **style sheets**) umesto HTML prezentacionih elemenata i atributa. Informacije o stilu mogu biti specificirane unutar HTML dokumenta ili u eksternim datotekama i mogu se odnositi na individualne elemente ili na grupe elemenata. Mechanizam za definisanje stilova ne zavisi od jezika kojim se opisuje stil. U nameri da se nametne korišćenje stilova WWW konzorcijum planira da u narednim verzijama ukine neke prezentacione elemente i atribute. Takvi elementi (tagovi) su u verziji 4.01 označeni za izbegavanje (engl. **deprecated**).

Najnovija verzija HTML-a je HTML 5.0. HTML 5.0 dopunjuje HTML sa sintaktičkim svojstvima koje uključuju elemente `<video>`, `<audio>`, `<header>` i `<canvas>`. Ove osobine su dodate kako bi omogućile lakše upravljanje multimedijalnim i grafičkim sadržajima bez dodatne potrebe za plugin-ovima i API. HTML5 takođe omogućava obogaćenje i semantičkog zadržaja dokumenta sa atributima kao što su `<section>`, `<article>`, `<header>` i `<nav>`. Iz istog razloga su dodati i neki novi atributi, dok su određeni elementi i atributi odbačeni u odnosu na prethodnu verziju.

▼ 4.1 Struktura HTML dokumenta

PRIMER HTML DOKUMENTA

HTML dokument se sastoji iz više elemenata grupisanih u verziju, zaglavje i telo dokumenta

HTML dokument se sastoji iz više različitih elemenata od kojih se većina može kategorizati u tri grupe:

- Verzija dokumenta
- Deklarativno zaglavje dokumenta (engl. **header**)
- Telo dokumenta (engl. **body**)

Primer jednostavnog HTML dokumenta:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Naslov stranice</title>
  </head>
  <body>

    <h1>Moje prvo zaglavje</h1>
    <p>Moj prvi paragraf.</p>
```

```
</body>  
</html>
```

Objašnjenje primera:

- **<!DOCTYPE html>** deklaracija definiše ovaj dokument kao HTML5
- **<html>** element je koreni element HTML stranice
- **<head>** element sadrži meta informacije dokumenta
- **<title>** element određuje naslov dokumenta
- **<body>** element sadrži delove stranice koji će biti prikazani na njoj
- **<h1>** element definiše veliki naslov
- **<p>** element definiše novi paragraf

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

✓ 4.2 Struktura HTML jezika

HTML KONSTRUKCIJE

HTML jezik koristi sledeće konstrukcije: elemente, atribute, karakter reference i komentare

HTML datoteka, koja opisuje jednu stranicu, sastavljena je od teksta i tagova. Tag je osnova svake HTML strane i kazuje veb čitaču kako da prikaže Veb stranu. HTML tag predstavlja jednu reč ili skraćenicu koja se nalazi u trouglastim zagradama. Na primer, tag može biti napisan kao

<html>

HTML datoteka mora da ima ekstenziju tipa **.html** , **.htm**, ali se može kreirati u bilo kom tekst editoru ili pomoću nekog alata za kreiranje HTML strana.

HTML jezik koristi sledeće konstrukcije:

- elemente
- atribute
- karakter reference
- komentare.



Slika 4.1.1 Primer HTML dokumenta sa ilustrovanim primerima za elemente i attribute

Primer elemenata i atributa je ilustrovan na slici 1. Navedeni elementi su:

- <HTML>
- <HEAD>
- <TITLE>
- <BODY>
- <P>

HTML ELEMENTI

Svaka deklaracija nekog tipa elementa opisuje tri dela: početni tag, sadržaj, završni tag.

HTML dokumenti su definisani pomoću njihovih elemenata. Tako HTML jezik ima tipove elemenata kojim se predstavljaju naslovi, pasusi, hipertekstualni linkovi, liste, tabele, slike itd. Svaka deklaracija nekog tipa elementa opisuje tri dela:

- početni tag
- sadržaj
- završni tag.

Ime elementa se pojavljuje i u startnom i u završnom tagu, s tim što se u završnom tagu ispred imena elementa upisuje znak „/“ kako bi se znalo da je u pitanju završni tag. Imena tagova se nalaze između zagrade „<“ i „>“.

U sledećem primeru je prikazano korišćenje tipa elementa *TITLE*. Između startnog taga i završnog taga nalazi se sadržaj „Naziv HTML dokumenta“.



Slika 4.1.2 Primer korišćenja TITLE elementa

Neki HTML tipovi elemenata dozvoljavaju autorima da se startni ili završni tag izostavi, mada se u tom slučaju gubi kompatibilnost sa XHTML-om. Tako na primer tagovi **<P>** i **** ne moraju da imaju završne tagove, a **<BODY>** i **<HEAD>** ne moraju da imaju startne tagove. Neki tagovi nemaju sadržaj, pa nema potrebe da imaju i završni tag. Primer ovakvog taga je **
**, kojim se označava završetak linije teksta.

PRIMER HTML ELEMENATA

HTML, HEAD, TITLE, BODY i P

Neki osnovni tagovi su prikazani na slici 1, a to su HTML, HEAD, TITLE, BODY i P. Element HTML definiše čitav HTML dokument i njegov početni tag je **<HTML>**, a završni tag je **</HTML>**. Njegov sadržaj predstavljaju drugi elementi, HEAD i BODY. Oni predstavljaju ugnjezdene elemente.

Element BODY definiše telo HTML dokumenta i njegov početni tag je **<BODY>**, a završni tag je **</BODY>**. Sadržaj ovog elementa je drugi element, P. Element P ima početni tag **<P>** i definiše novi paragraf. Ovaj element ne zahteva završni tag, mada se može kao završni tag koristiti **</P>**. Sadržaj ovog elementa je tekst "...Sadržaj ovog dokumenta..."

Treba naglasiti da HTML tagovi ne moraju biti isključivo napisani velikim slovima. Dozvoljeno je da se koriste i mala i velika slova u imenu tagova, odnosno **<HTML>** i **<html>** se može koristiti.

Slika 4.1.3 Primer HTML dokumenta sa ilustrovanim primerima za elemente i attribute

HTML ATRIBUTI

Atributi i njihove vrednosti se zadaju unutar početnog taga

Elementi imaju pridružene attribute koji mogu uzimati podrazumevajuće vrednosti, vrednosti zadate od strane autora ili vrednosti definisane nekom skriptom. Atributi i njihove vrednosti se zadaju unutar početnog taga. Vrednost atributa se upisuje između znakova navoda. U sledećem primeru atribut **align**, koji ima vrednost **"left"**, upisan je unutar početnog taga.

<h1 align="left"> Tekst u Heading 1 formatu koji će biti postavljen uz levu marginu </h1>

Kada se poziva hiperlink u hipertekstu, atributi se takođe koriste. Na primer, atribut *href* se može koristiti na sledeći način:

Ovo je link

HTML KARAKTER REFERENCE

Karakter reference su numerička ili simbolička imena za karaktere koji mogu biti uključeni u HTML dokument

Karakter reference su numerička ili simbolička imena za karaktere koji mogu biti uključeni u HTML dokument. Ova konstrukcija se koristi kada je unutar dokumenta potrebno uključiti karaktere koji se retko koriste ili nisu podržani od alata za dizajniranje veb strana (na primer znak „<“). Karakter referenca počinje sa znakom „&“, a završava se znakom „;“. Na primer karakter referenca: < “ predstavlja znak „<“.

HTML KOMENTARI

Komentari se pišu između karatera "<! -" i "->"

HTML kod treba da bude iskomentarisan, kao i bilo koji drugi kod. Cilj komentarisanja koda je lakše snalaženje u kodu, kako bi se u njemu lakše obavile izmene u budućnosti. Komentari se pišu između karatera "<! -" i "->". Primer jednog komentara je:

<!-Ovo je komentar

koji može da bude napisan

u više redova ->

Informacije u komentaru ne interpretira čitač veb strana, što znači da korisnik neće videti sadržaj komentara među ostalim sadržajem veb stranice.

▼ 4.3 Zaglavlje HTML dokumenta

HEAD (ZAGLAVLJE HTML DOKUMENTA)

Element zaglavila dokumenta HEAD sadrži informacije o dokumentu, kao što su naslov, ključne reči koje će biti korišćene od strane mašina za pretraživanje i druge podatke koji se ne smatraju

Element zaglavila dokumenta HEAD sadrži informacije o dokumentu, kao što su naslov, ključne reči koje će biti korišćene od strane mašina za pretraživanje i druge podatke koji se ne smatraju sadržajem dokumenta. Korisnički agent ne iscrtava elemente koji se pojavljuju u zaglavljtu kao sadržaj, već koristi neke druge mehanizme da bi ih učinio dostupnim korisniku. Unutar HEAD elementa se mogu naći i META elementi. META element služi za specificiranje metapodataka, odnosno informacija o dokumentu, kao što su autor, lista ključnih reči, datum publikovanja, jezik sadržaja dokumenta. Svi ovi podaci mogu biti jako korisni mašinama za pretraživanje.

Sledeći primer pokazuje kako je pomoću META elementa specificiran autor veb stranice:

```
<head>
<meta charset="UTF-8">
<meta name="description" content="Free Web tutorials">
```

```
<meta name="keywords" content="HTML,CSS,XML,JavaScript">
<meta name="author" content="Milisav Mitrović">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
```

Kada je potrebno specificirati jezik kojim je zadata vrednost atributa *content* u META elementu, koristi se atribut *lang*. Navođenje jezika može da bude korisno kako mašinama za pretraživanje tako i audio čitačima veb strana koji pomoću ovog atributa određuju koja će se pravila za izgovaranje primeniti za sadržaj.

U sledećem primeru lang atribut je iskorišćen da bi se opisalo na kom jeziku su date ključne reči iz sadržaja dokumenta. Primer se odnosi na US engleski, britanski engleski i francuski.

```
<-- For speakers of U.S. English -->
<META name="keywords" lang="en-us"
      content="vacation, Greece, sunshine">
<-- For speakers of British English -->
<META name = "keywords" lang = "en"
      content = "holiday, Greece, sunshine">
<-- For speakers of French -->
<META name = "keywords" lang = "en"
      content = "vacances, Grèce, soleil ">
```

▼ 4.4 Telo HTML dokumenta

BODY (TELO HTML DOKUMENTA)

Telo dokumenta se sastoji od opisa sadržaja dokumenta

Telo dokumenta se sastoji od opisa sadržaja dokumenta. Autori HTML dokumenta treba da vode računa o tome da je od verzije 4 preporučeni način za specifikaciju prezentacije dokumenata pomoću stilova (engl. **style sheets**) i da su svi prezentacioni atributi elementa BODY označeni za izbegavanje (engl. **deprecated**). Naredni primer ima dva dela.

<body> element definiše telo dokumenta. Telo dokumenta ima početni tag **<body>** i završni tag **</body>**.

U primeru se navode još dva HTML elementa **<h1>** i **<p>**.

Primer

```
<html>  
<body>  
    <h1>Moje prvo zaglavlje </h1><p> Moj prvi paragraf. </p>  
</body>  
</html>
```

Element **<h1>** definiše zaglavlje. Ima počtni tag **<h1>** i završni tag **</h1>**. Sadržaj elementa je Moje prvo zaglavlje.

Element **<p>** definiše novi paragraf. Ima počtni tag **<p>** i završni tag **</p>**. Sadržaj elementa je Moj prvi paragraf.

HTML zaglavla su definisana tagovima od **<h1>** do **<h6>**.

<h1> definiše najvažnije, odnosno najveće zaglavlje, dok **<h6>** definiše najmanje važno, odnosno najmanje zaglavlje.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

KORIŠĆENJE STILOVA

Korišćenje stilova kroz attribute

Bojenje pozadine

background-color definiše boju pozadine za HTML element. Sledeci primer definiše boju pozadine da bude puder plavo (engl. *powderblue*):

```
<body style="background-color:powderblue;">  
<h1> Ovo je zaglavlje </h1>  
<p> Ovo je paragraf. </p>  
</body>
```

Bojenje teksta

color definiše boju za HTML element:

```
<h1 style=" color:blue;"> Ovo je zaglavlje </h1>  
<p style=" color:red;"> Ovo je paragraf. </p>
```

Stil slova

font-family definiše stil slova HTML elementa:

```
<h1 style=" font-family:verdana;"> Ovo je zaglavlje </h1>  
<p style=" font-family:courier;"> Ovo je paragraf. </p>
```

Veličina teksta

font-size definiše veličinu teksta HTML elementa:

```
<h1 style="font-size:300%;"> Ovo je zaglavlj </h1>
<p style="font-size:160%;"> Ovo je paragraf. </p>
```

Poravnanje teksta

text-align definiše poravnanje teksta HTML elementa

```
<h1 style="text-align:center;"> Ovo je centrirano zaglavlj </h1>
<p style="text-align:center;"> Ovo je centrirani paragraf. </p>
```

KORIŠĆENJE STILOVA- PRIMER

Specifične boje pozadine

Primer 1 predstavlja HTML dokument koji koristi BODY atribute koji su označeni za izbegavanje. Primer 2 predstavlja document koji koristi stilove kako bi se postigao isti efekat kao u primeru 1. U prvom slučaju se atributi elementa BODY koriste da se specificira boja pozadine (bela), teksta (crna) i linkova (“**fuchsia**” kada je aktivan i “**maroon**” kada je link već posećen).

Primer 1. Upotreba atributa za izbegavanje

```
<!DOCTYPE>
<html>
  <head>
    <title> Example of HTML document </title>
  </head>
  <body bgcolor="pink" text="black" link="blue" alink="fuchsia" vlink="maroon">
    ... Sadržaj dokumenta...
  </body>
</html>
```

Primer 2.

Upotreba stilova (Cascading Style Sheet)

```
<!DOCTYPE>
<html>
  <head>
    <title> Example of HTML document </title>
    <style type="text/css">
      body {background: white; color: black}
      a:link {color: blue }
      a:visited {color: maroon }
      a:active {color: fuchsia }
    </style>
  </head>
  <body>
```

```
... Sadržaj dokumenta...
</body>
</html>
```

STIL STRANICE U EKSTERNOJ DATOTECI

Stil stranice se može referencirati u eksternoj datoteci

Željeni stil ne mora da bude opisan u samom dokumentu, kao što je to urađeno u prethodnom primeru, već se može definisati u nekoj eksternoj datoteci, kao što je prikazano primerom 3. U primeru 3 se više strana mogu referencirati na isti stil. Stil stranice je definisan u eksternoj datoteci *style.css*.

Primer 3

```
<!DOCTYPE>
<html>
    <head>
        <title> Primer HTML dokumenta </title>
        <link rel="stylesheet" type="text/css" href="style.css">
    </head>
    <body>
        ... Sadrzaj dokumenta...
    </body>
</html>
```

HTML TABELE

*HTML tabela se definiše tagom **<table>**.*

HTML tabela se definiše tagom **<table>** .

Svaki red u tabeli se definiše tagom **<tr>**.

Zaglavlje tabele se definiše **<th>** tagom. Podrazumevani stil tabele je da su zaglavlja boldovana i centrirana.

Ćelija tabele je definisana **<td>** tagom.

Primer

```
<table style="width:100%">
    <tr>
        <th> Prezime </th>
        <th> Ime </th>
        <th> Godine </th>
    </tr>
    <tr>
```

```
<td> Ivana </td>
<td> Ilić </td>
<td>45</td>
</tr>
<tr>
    <td> Eva </td>
    <td> Kolić </td>
    <td> 64 </td>
</tr>
</table>
```

▼ Poglavlje 5

Pokazna vežba: Izrada HTML dokumenta

WWW

Najvažnija mogućnost HTML jezika je kreiranje hiperveza (engl. hyperlink), odnosno sposobnost međusobnog povezivanja stranica

Predviđeno vreme pokazne vežbe je 45 minuta.

World Wide Web, skraćeno WWW, čini ogroman broj dokumenata – Web strana, koje su međusobno povezane tako da se može prelaziti sa jedne na drugu. HTML (engl. Hypertext Markup Language) je jezik koji služi za pripremu dokumenata koje prikazuju Web čitači i koristi se za formatiranje teksta, dodavanje slika, zvuka, video zapisa itd. HTML predstavlja ključni mehanizam za međusobno povezivanje Web strana.

Organizacija koja se često naziva „Ujedinjenim nacijama Web-a“ je World Wide Web Consortium (www.w3.org), skraćeno W3C. Cilj organizacije je da propagira univerzalnost u kreiranju dokumenata za web. Ideja je da se sve kompanije koje se na posredan ili neposredan način bave web-om ujedine i slože oko standarda i onda da probaju da razdvoje svoje proizvode prema brzini, lakoći korišćenja, ceni i sl.

HTML TAGOVI

Tagovi se nalaze između znakova manje od < i veće od >, a koji označavaju tip sadržaja.

Najvažnija mogućnost HTML jezika je kreiranje hiperveza (engl. hyperlink), odnosno sposobnost međusobnog povezivanja stranica. Hiperveze omogućavaju prelazak sa jedne strane na drugu jednostavnim klikom miša. Suština HTML jezika je u tagovima, ključnim rečima koje se nalaze između znakova manje od (<) i veće od (>), a koji označavaju tip sadržaja.

Generalno, postoje početni i završni tag. Krajnji tag se od početnog razlikuje dodatnim znakom / ispred ključne reči. Tagovi su način da se Web čitaču izdaju instrukcije.

Sledi spisak najčešće korišćenih HTML tagova:

<HTML></HTML> Identificuje dokument kao HTML dokument

<HEAD></HEAD> Identificuje glavu HTML dokumenta

<BODY></BODY> Identificuje telo HTML dokumenta

<TITLE></TITLE> Precizira naslov HTML dokumenta

<H1></H1>...<H6></H6> Postavlja veličinu fonta kao jednu od šest unapred definisanih veličina

<P></P> Označava pasus na stranici

**** Bold format teksta

<I></I> Italic format teksta

<U></U> Podvlačenje teksta (**Underline**)

**** Postavljanje slike

<A> Postavljanje hiperveze

**
** Pomeranje tačke umetanja u sledeći red

**** Kreiranje spiska uz pomoć oznaka

**** Kreiranje numerisanog spiska

**** Identificuje jednu stavku u okviru spiska

KREIRANJE HTML DOKUMENTA

Za kreiranje HTML dokumenta može se koristiti bilo koji program za obradu teksta

Predviđeno vreme izrade sledećeg primera je 10 minuta.

Primer:

Koristeći *Notepad* editor kreirati HTML dokument sa osnovnim delovima: zaglavljem i telom, koji će u Web čitaču prikazati tekst: UM – Univerzitet Metropolitan. Dobar dan. Dokument potom snimiti pod nazivom **primer1.html** u folderu **C:\IT101\Vezba7**, koji treba formirati iz editora.

Rešenje:

Za kreiranje HTML dokumenta može se koristiti bilo koji program za obradu teksta (tekst editor) kao što su *Notepad*, *Notepad ++*, *MS Word*, *WordPad*, *EmEditor*, *Emacs* i sl. Za ovaj primer koristiće se *Notepad* – editor koji postoji u svim novijim verzijama *Windows* operativnog sistema. *Notepad* se pokreće na više načina. Najlakše je pronaći program preko START-a i pretrage naziva programa, u ovom slučaju *Notepad* nakon čijeg odabira se pojavljuje osnovni prozor *Notepad*.

ZAGLAVLJE I TELO HTML DOKUMENTA

Osnovni delovi jednog HTML dokumenta su zaglavlj i telo

Osnovni delovi jednog HTML dokumenta su zaglavlj i telo. U njih se unose različiti sadržaji o čemu će više reči biti kasnije. Za razdvajanje ovih delova koriste se odgovarajući tagovi.

Za ovaj primer:

- potrebno je uneti sledeće početne tagove: <HTML>, <HEAD> i <BODY>,
- kao i odgovarajuće "zatvarajuće", krajne ili završne tagove </HTML>, </HEAD> i </BODY>.

Tagovi <HTML> i </HTML> označavaju da se radi o HTML dokumentu. Između tagova <HEAD> i </HEAD> nalazi se **zaglavlj**, a <BODY> i </BODY> **tel** HTML dokumenta.

Da bi se neki tekst prikazao u Web čitaču, potrebno je uneti ga u telo dokumenta.

ČUVANJE HTML DOKUMENTA

Za snimanje dokumenta potrebno je u polju File name: unesite naziv dokumenta sa ekstenzijom .htm ili .html.

Za snimanje HTML dokumenta koristi se opcija **File→Save As...** u *Notepad* editoru.

Za snimanje dokumenta potrebno je u polju **File name:** unesite naziv dokumenta sa ekstenzijom **.htm** ili **.html**. Za ovaj primer naziv dokumenta je **primer1.html**

Nakon unosa imena i pritiska na taster **Enter** ili aktiviranjem opcije **Save** u istom prozoru, pripremljeni dokument snimljen je kao HTML dokument.

Aktiviranje snimljenog dokumenta, npr. dvostrukim klikom levog tastera miša u *Windows Explorer*-u na lokaciji **C:\IT101\Vezba7**, pokreće Web čitač i prikazuje se sadržaj snimljenog HTML dokumenta.

PRIKAZIVANJE TEKSTA

Za prikazivanje teksta u naslovnoj liniji koriste se tagovi <TITLE> i </TITLE>

Predviđeno vreme izrade sledećeg primera je 5 minuta.

Primer:

Kreirati HTML dokument koji u naslovnoj liniji naslovnoj liniji Web čitača prikazuje tekst: UM, i tekst u telu formatirati tako da bude ispisani u dva reda na sledeći način:

UM - Univerzitet Metropolitan

Dobar Dan.

Rešenje:

Da bi se koristio već postojeći HTML dokument, što je čest slučaj, potrebno je u npr. *Windows Explorer*-u postaviti kurzor na željeni dokument, pritisnuti desni taster miša i iz menija **Open With** izabrati opciju **NotePad**.

Za prikazivanje teksta u naslovnoj liniji koriste se tagovi **<TITLE>** i **</TITLE>**. Željeni tekst unosi se između ova dva taga i sve to je potrebno uneti u zaglavlje dokumenta. Da bi neki tekst bio prikazan **Bold** formatom potrebno je da se nalazi između **** i ****. Za početak novog reda koristi se tag **
** bez završnog taga. Za pisanje teksta slovima *Italic* formata koriste se **<I>** i **</I>**.

PRIMER

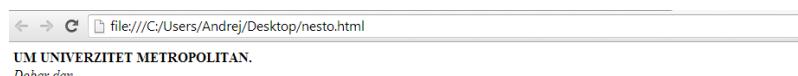
Za početak novog reda koristi se tag </BR> bez završnog taga

Za zadati primer kod izgleda ovako:

```
<html>
  <head>
    <title> UM </title>
  </head>

  <body>
    <b> UM UNIVERZITET METROPOLITAN. </b><br/>
    <i> Dobar dan. </i>
  </body>
</html>
```

Nakon unetih promena dokument se snima uz pomoć opcije **File→Save As...** opisane u 1. Primeru, u već postojećem folderu, pod nazivom **primer2.html**. Nakon otvaranja u Web čitaču dobija se rezultat kao na slici 1 .



Slika 5.1 HTML dokument u browseru

UGNJEŽDAVANJE TAGOVA

Važno izbeći preklapanje tagova

U naslovnoj liniji ispisan je željeni tekst (UM), a postojeći tekst formatiran je na traženi način.

Tagovi imaju mogućnost „ugnježdavanja“. Na primer, ako je potrebno da neki deo teksta bude istovremeno ispisan kao **Bold** i **Underline**, potrebno je sadržaj navesti između početnog

i završnog **** i ****, odnosno **<U>** i **</U>** taga. Na primer, ako želimo da se u Web čitaču tekst UM ispiše Bold formatom, a da istovremeno bude i podvučen, koristićemo sledeću kombinaciju tagova:

<U>UM</U>.

U ovom slučaju veoma je važno izbeći preklapanje tagova. Na primer, u slučaju neka dva proizvoljna taga **<PRVI>** i **<DRUGI>**, njihovo preklapanje bi izgledalo ovako:

<PRVI> <DRUGI> </PRVI> </DRUGI>

To može da dovede do problema prilikom prikazivanja dokumenta, pa je potrebno „ugneždavanje“ datih tagova:

<PRVI> <DRUGI> </DRUGI> </PRVI>

HTML ATRIBUTI

Osim korišćenja tagova, prilikom kreiranja HTML dokumenta mogu se koristiti i odgovarajući atributi

Osim korišćenja tagova, prilikom kreiranja HTML dokumenta mogu se koristiti i odgovarajući atributi.

Tag se koristi da Web čitaču „objasni“ da je potrebno da nešto uradi, a atribut kako da to uradi. Atributi se navode u okviru početnog taga. Na primer, ako želite da deo teksta: „Dobar dan.“, iz prethodnih primera bude isписан većim slovima veličine 6, potrebno je u postojeći kod dodati sledeće:

** Dobar dan. **

Kada se menja postojeći HTML dokument, promene se mogu snimiti jednostavno uz pomoć opcije

File→Save, a prikaz u čitaču ažurirati opcijom **Reload**.

Ovo praktično znači da ćete naredne primere moći da provežbate tako što ćete imati otvoren i Notepad sa kodom iz prethodnog primera i odgovarajući Web čitač, pa kada kod promenite i promenu snimite, dovoljno je da stranu u čitaču ažurirate.

PODEŠAVANJE FONTA

Za veličinu slova, tip fonta ili boju postoje podrazumevane vrednosti

Za veličinu slova, tip fonta ili boju postoje podrazumevane vrednosti (**default**). Podrzumevane vrednosti su one koje će Web čitač prepostaviti, pod uslovom da nije drugačije definisano. Za tekst to na primer znači veličinu 3 (što odgovara 12pt), crnu boju i font **Times New Roman**. Preporuka je da se vodi računa o izboru pre svega fonta, jer moguće je da definisani font ne

postoji na nekom računaru. Ukoliko čitač ne pronađe font koji je definisao onaj koji je kreirao dokument, tada će čitač koristiti podrazumevani font.

U okviru jednog taga može se pojaviti više atributa. Na primer ako želimo da tekst iz primera: „Univerzitet Metropolitan”, bude ispisana fontom **Verdana**, veličine 4 i crvene boje, atributte ćemo koristiti na sledeći način:

Univerzitet Metropolitan

Za opisivanje boje se često koriste posebne oznake, npr. za crvenu boju to znači vrednost atributa COLOR="#FF0000".

PARAGRAF

Tag za označavanje paragrafa

U drugom primeru videli smo kako se koristi tag **
** za početak novog reda. Sličan je i tag **<P>** koji označava početak novog pasusa, odnosno prelazak u novi red uz preskakanje jednog reda. Na primer, ako bi smo kod iz primera napisali na sledeći način:

```
<html>
  <head>
    <title> UM </title>
  </head>
  <body>
    <p><b> UM </b></p>
    <p><b> Univerzitet Metropolitan. </b></p>
    <p><i> Dobar dan. </i></p>
  </body>
</html>
```

PRIMER DOKUMENTA SA NAŠIM SLOVIMA

U delu Encoding prozora izaberite opciju Unicode

Predviđeno vreme izrade sledećeg primera je 5 minuta.

Primer:

Kreirati HTML dokument sa ispisanim našim velikim slovima: Ž, Ć, Č, Š i Đ. Dokument snimiti kao **primer3.html**.

Rešenje:

Kod u ovom slučaju, može da izgleda ovako

```
<html>
  <head>
    <title> UM </title>
```

```
<head>  
<body>  
    Ž Ć Č Š Đ  
</body>  
</html>
```

Ako ovaj kod unesete u Notepad i pokušate da dokument snimite kao npr. **primer3.html** verovatno će se pojaviti upozorenje.

Ukoliko nastavite sa snimanjem klikom na **OK**, moguće je da neka naša slova neće biti prikazana pravilno. Zato je preporuka da u prozoru kliknete na **Cancel**. Zatim, ponovo aktivirajte

File→Save i u delu **Encoding** prozora izaberite opciju **Unicode**. Na taj način, sačuvaćemo naša slova prilikom snimanja.

DEKLARACIJA TIPOA DOKUMENTA

U skladu sa standardima, za svaki HTML dokument zahteva se korišćenje deklaracije HTML tipa na početku dokumenta

U skladu sa standardima, za svaki HTML dokument zahteva se korišćenje deklaracije HTML tipa na početku dokumenta. Prilikom otvaranja Web strane, Web čitač prvo proverava vrh strane u potrazi za DOCTYPE deklaracijom. Ukoliko je pronađe, Web čitač je u mogućnosti da datu stranu prikaže ispravno.

DOCTYPE deklaracijom dokumenta, definiše se koji tip HTML će se koristiti i Web čitačima se "stavlja do znanja" šta da "očekuju" od dokumenta koji treba da prikažu. DOCTYPE deklaracija nije HTML element i nema svoj završni tag. Deklaracija mora uvek da se nalazi u prvoj liniji HTML dokumenta. Validni HTML dokument deklariše koja verzija HTML jezika je korišećena za izradu dokumenta.

Deklaracija tipa dokumenta (engl. Document Type Declaration([DTD](#))) predstavlja definiciju tipa koji je korišćen za dati dokument. **Verzija 4.01** HTML određuje tri DTD definicije, koje bi trebalo uključiti u dokumente, a razlikuju se po podršci koju pružaju različitim elementima. Ova verzija je bila zasnovana na SGML-u ([Standard Generalized Markup Language](#)).

HTML 5 verzija nije zasnovana na SGML-u i zato ne zahteva DTD definiciju. U novoj verziji, deklaracija se vrši veoma jednostavno, tako što se na početku dokumenta definiše **<!DOCTYPE>**. Takođe se u istom elementu može i započeti html tag na sledeći način: **<!DOCTYPE html>**.

KODNI RASPORED

Od velikog značaja da se na početku HTML dokumenta pravilno definiše kodni raspored

Od velikog značaja da se na početku HTML dokumenta pravilno definiše kodni raspored. Za deklaraciju kodnog rasporeda u zaglavlje dokumenta ubacuje se sledeći red:

<meta http-equiv="Content-Type" content="text/html; charset=windows-1250">
ili <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-2">

ako želite da čitač pravilno pročita naša slova. Ako se radi o cirilici koristi se:

<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">

Za ovaj slučaj dokument će izgledati ovako:

```
<!DOCTYPE>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=windows-1250">
    <title> UM </title>
  </head>
  <body>
    <b> UM - Univerzitet Metropolitan. </b><br/>
    <i> Dobar dan. </i>
  </body>
</html>
```

VALIDACIJA HTML DOKUMENTA

Validacija je proces provere ispravnosti HTML dokumenta

Validacija je proces provere ispravnosti HTML dokumenta. Validacija ovog dokumenta može se izvršiti na Web strani <http://validator.w3.org>.

U delu **Validation by File Upload** potrebno je kliknuti na **Browse** i pronaći odgovarajući dokument na lokalnom disku. Nakon što se u polju pojavi putanja do ovog dokumenta potrebno je kliknuti na **Check**. Nakon toga dobija se odgovarajući izveštaj.

▼ Poglavlje 6

Pokazna vežba: Povezivanje HTML dokumenata

POVEZIVANJE DOKUMENATA NA VEBU

Hiperveze su važan deo svih HTML dokumenata objavljenih na WWW

Predviđeno vreme pokazne vežbe je 30 minuta.

Mogućnost povezivanja različitih dokumenata je jedno od najvažnijih svojstava WWW. Ova sposobnost, bez obzira na razdvojenost različitih sadržaja, zaslužna je za neverovatno moćan sistem razmene informacija. Uticaj ovog pristupa je toliko jak da primorava čitave industrije da mu se prilagođavaju. Web je promenio i nastavlja da menja način ljudskog života – učenja, komuniciranja, kupovine...

Hiperveze su važan deo svih HTML dokumenata objavljenih na WWW. Ove veze se koriste da bi povezale stranice sa nekim drugim resursima na WWW. Taj resurs može biti druga stranica, slika ili epošta. Tekst hiperveze se obično prikazuje u drugačijoj boji da bi se razlikovao od ostalog teksta i po pravilu je podvučen. Kada se mišem prelazi preko teksta hiperveze cursor se menja, obično u simbol šake, kako bi ukazao na to da biranje tog teksta vodi do nekog povezanog materijala.

ZADATAK: POVEZIVANJE DOKUMENATA I PRIKAZIVANJE SLIKE

Za kreiranje hiperveza koristimo početni <A> i završni tag

Predviđeno vreme izrade sledećeg primera je 10 minuta.

Primer:

Napraviti dokument **strana1.html** na kojem će se nalaziti tekst: „**KLIKNI OVDE**“. Zatim, kreirati dokument na kojem će se nalaziti slika **smiley.jpg** koju možete preuzeti sa e-learning sistema.

Dokument snimiti kao **strana2.html**. Potom hipavezom povezati ova dva dokumenta tako da kada se u dokumentu **strana1.html** klikne na tekst: „**KLIKNI OVDE**“, u Web čitaču se prikaže dokument **strana2.html**.



Slika 6.1 Smiley

Rešenje:

Za kreiranje hiperveza koristimo početni `<A>` i završni `` tag (A – skraćeno od **Anchor**, na engleskom sidro). U okviru početnog `<A>` taga definišemo hipervezu uz pomoć atributa HREF čija vrednost predstavlja lokaciju HTML dokumenta koji želite da prikažete u čitaču kada se na vezu klikne.

Ako dokumente snimamo u isti direktorijum dovoljno je da se navede samo naziv dokumenta.

REŠENJE PRIMERA

Da bi hiperveza kreirana na ovaj način mogla da funkcioniše, potrebno je da dokumenti budu u istom direktorijumu

Dokument **strana1.html** može da izgleda ovako:

```
<!DOCTYPE>
<html>
  <head>
  </head>
  <body>
    <a href="strana2.html">KLIKNI OVDE</a>
  </body>
</html>
```

Dokument se zatim snima na već poznati način. Na isti način pripremamo i sledeći dokument, i snimamo ga kao **strana2.html**. Da bi hiperveza kreirana na ovaj način mogla da funkcioniše, potrebno je da dokumenti budu u istom direktorijumu. Isto važi i za sliku koju prikazujemo kod drugog dokumenta.

```
<!DOCTYPE>
<html>
  <head>
  </head>
  <body>
```

```
  
</body>  
</html>
```

PRIKAZ REZULTATA PRVE I DRUGE STRANE

*Da biste u dokumentu napravili vezu prema nekoj postojećoj lokaciji na Internetu, atributu **Href** dodeljujete odgovarajući URL*

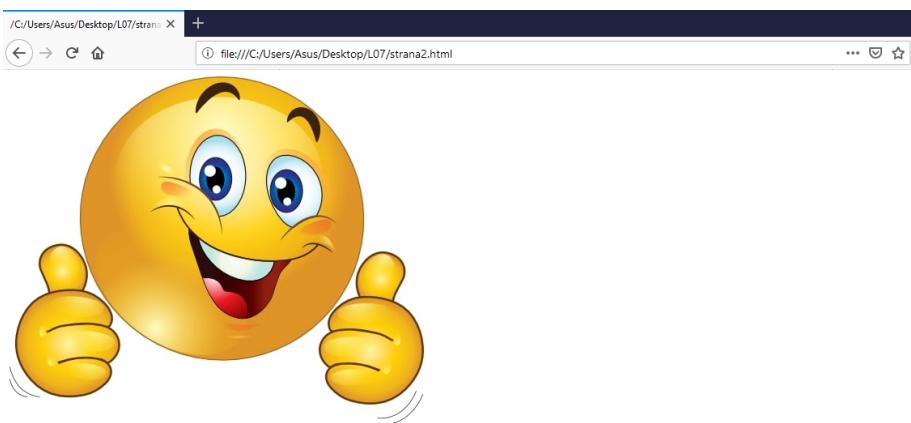
Nakon otvaranja prvog dokumenta i klika na tekst KLIKNI OVDE, u Web čitaču pojavljuje se drugi dokument.

Primetićete da se prilikom postavljanja kursora na hipervezu kod nekih Web čitača u statusnoj liniji, na primer u donjem levom uglu, pojavljuje adresa dokumenta koji se otvara klikom na vezu.



Slika 6.2 Prva strana

Da biste u dokumentu napravili vezu prema nekoj postojećoj lokaciji na Internetu, atributu **Href** dodeljujete odgovarajući URL.



Slika 6.3 Druga strana

PRIMER HIPERVEZE ZA SLANJE EMAILA

*Kod prve veze vrednost atributa **Href** je Web adresa FIT-a*

Predviđeno vreme izrade sledećeg primera je 5 minuta.

Primer:

Kreirati HTML dokument koji će u Web čitaču prikazati sledeće:

Studiram na UM-u. Kliknite ovde da mi pošaljete poruku.

FIT predstavlja hipervezu prema sajtu Univerziteta Metropolitan – www.metropolitan.ac.rs, a tekst: „Kliknite ovde da mi pošaljete mail“, treba da omogući slanje elektronske pošte studentu Petrović Petru na adresu:

mailto:petrovic.petar@metropolitan.ac.rs

Rešenje:

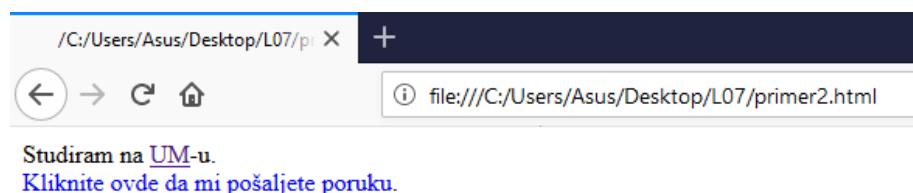
Za kreiranje ovog dokumenta koriste se isti tagovi kao i kod prethodnog primera. Kod prve veze vrednost atributa HREF je Web adresa FIT-a. Za pokretanje odgovarajućeg programa za razmenu elektronske pošte, atributu dodelujemo e-mail adresu ispred koje se nalazi deo: **mailto:**. Rešenje za ovaj primer može izgledati ovako:

```
<!DOCTYPE>
<html>
  <head>
  </head>
  <body>
    Studiram na <a href="https://www.metropolitan.ac.rs">UM</a>-u.<br/>
    <a href="mailto:petrovic.petar@metropolitan.ac.rs">Kliknite ovde da mi
    pošaljete poruku</a>.
  </body>
</html>
```

SLANJE EMAILA POMOĆU HIPERVEZE

Klikom na drugi red otvara se odgovarajući program, odnosno prazna e-mail poruka

Snimljeni dokument prikazan u čitaču izgleda kao na slici 4.



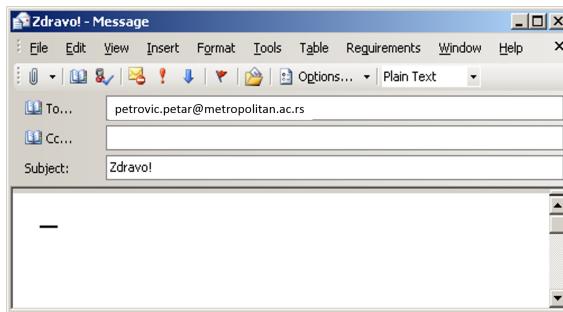
Slika 6.4 Dokument sa vezom

Klikom teksta UM u prvom redu dokumenta u Web čitaču otvara se sajt FIT-a. Klikom na drugi red otvara se odgovarajući program, odnosno prazna e-mail poruka. Obratite pažnju da efekti

ove druge opcije zavise od toga koji program koristite za razmenu pošte, ali i od podešavanja pojedinih Web čitača. U slučaju da želite da odmah bude popunjeno i Subject: polje poruke (Slika 5), potrebno je da deo dokumenta sa vezom za pisanje poruke izgleda ovako:

Kliknite ovde da mi pošaljete poruku.

gde je vrednost parametra subject, tekst koji će se nalaziti u datom polju programa za pisanje poruke. Pojedini programi za pisanje i razmenu pošte ne podržavaju ovu opciju.



Slika 6.5 Unapred popunjena subject i to polja

SLIKA KAO HIPERVEZA

U red gde je u dokument umetnuta slika dodajemo početni <A> i završni tag

Predviđeno vreme izrade sledećeg primera je 5 minuta.

Primer:

Kreirati HTML dokument gde će slika **smiley.jpg** korišćena kod prvog primera ove vežbe predstavljati vezu prema sajtu UM-a.

Rešenje:

Za ovaj primer možemo da koristimo postojeći dokument **strana2**. Potom, red gde je u dokument umetnuta slika dodajemo početni **<A>** i završni **** tag, na sledeći način:

U početnom **<A>** tagu nalazi se atribut HREF čija je vrednost URL sajta FIT-a. Kada promenu snimimo i dokument otvorimo u Web čitaču, klikom na sliku otvaramo sajt FIT-a.

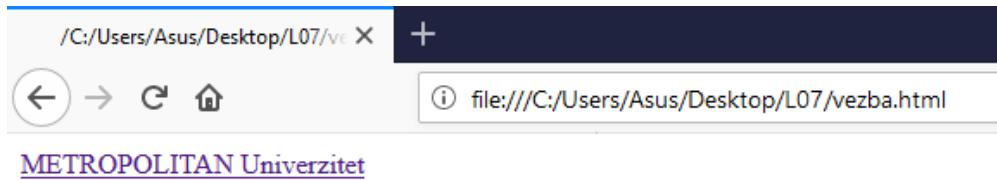
ZADATAK ZA VEŽBU KREIRANJA HIPERLINKA

Prikaz zadatka za vežbu hiperlinkova

Napraviti hyperlink ka omiljenom sajtu.

Rešenje:

```
<html>
  <body>
    <p><a href="https://www.metropolitan.ac.rs">METROPOLITAN Univerzitet</a></p>
  </body>
</html>
```



Slika 6.6 Prikaz rešenja

ZADATAK: POVEZIVANJE ČETIRI DOKUMENTA

Za kreiranje veza između dokumenata koristi se standardni <A> tag

Predviđeno vreme izrade sledećeg primera je 10 minuta.

Primer:

Kreirati četiri HTML dokumenta. Na svakom od njih treba da se nalazi redni broj, npr. na prvom dokumentu treba da piše: „1. dokument“. U svakom dokumentu potrebno je da se nalazi navigaciona linija koja će omogućavati prelazak na naredni i prethodni dokument, kao i na prvi, odnosno poslednji (četvrti) dokument. Hiperveza za prelazak na naredni, odnosno prethodni dokument, data je slikom Strelica, a prelazak na prvi i poslednji dokument opisan je rečima (**Prvi** i **Poslednji**).

Rešenje:

Kao što je rečeno, za kreiranje veza između dokumenata koristi se standardni [<A> tag](#). U početnom tagu kao vrednosti atributa HREF, navode se adrese HTML dokumenata koji su povezani. Budući da ćemo sve dokumente snimiti u jednom direktorijumu, onda navodimo samo nazive.

U prethodnom primeru videli smo da ukoliko se kao veza koristi slika, između početnog [<A>](#) i završnog [](#) taga nalazi se tag sa atributom SRC koji upućuje na odgovarajuću sliku.

Korišćenje reči: „Prvi“ i „Poslednji“ kao hiperveza, podrazumeva smeštanje tih reči, takođe, između početnog [<A>](#) i završnog [](#) taga.

PRVA STRANA

Dokument snimite u željeni direktorijum kao prvi.html.

Prvi dokument može da izgleda ovako:

```
<!DOCTYPE>
<html>
  <head>
  </head>
  <body>
    1. dokument<br/>
    <a> Prvi </a>
    <a> </a>
    <a href="drugi.html"></a>
    <a href="cetvrti.html"> Poslednji </a>
  </body>
</html>
```

Dokument snimite u željeni direktorijum kao **prvi.html**.

DRUGA STRANA

Dokument snimite kao drugi.html.

Drugi dokument izgleda ovako:

```
<!DOCTYPE>
<html>
  <head>
  </head>
  <body>
    2. dokument<br/>
    <a href="prvi.html"> Prvi </a>
    <a href="prvi.html"></a>
    <a href="treci.html"></a>
    <a href="cetvrti.html"> Poslednji </a>
  </body>
</html>
```

Dokument snimite kao **drugi.html**.

TREĆA I ČETVRTA STRANA

Poslednji dokument snimite kao cetvrti.html.

U nastavku je dat treći HTML dokument:

```
<!DOCTYPE>
<html>
  <head>
  </head>
  <body>
```

```
3. dokument<br/>
<a href="prvi.html"> Prvi </a>
<a href="drugi.html"></a>
<a href="cetvrti.html"></a>
<a href="cetvrti.html"> Poslednji </a>
</body>
</html>
```

Dokument snimite kao **treci.html**.

Četvrti dokument može da izgleda ovako:

```
<!DOCTYPE>
<html>
<head>
</head>
<body>
4. dokument<br/>
<a href="prvi.html"> Prvi </a>
<a href="treci.html"></a>
<a></a>
<a> Poslednji </a>
</body>
</html>
```

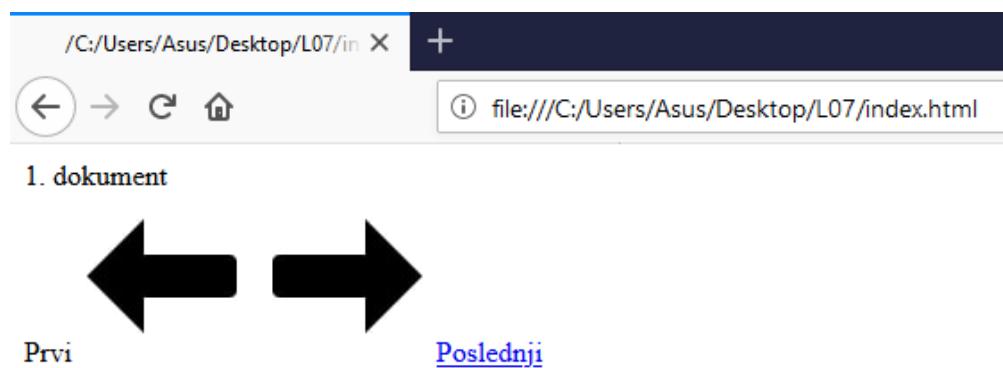
Poslednji dokument snimite kao **cetvrti.html**.

PRIKAZ REZULTATA

U prvom i poslednjem (četvrtom) dokumentu, veze prema prethodnoj, odnosno poslednjoj strani nisu aktivne, tj. ne postoji adresa opisana atributom HREF prema nekom određenom dokumentu

U prvom i poslednjem (četvrtom) dokumentu, veze prema prethodnoj, odnosno poslednjoj strani nisu aktivne, tj. ne postoji adresa opisana atributom HREF prema nekom određenom dokumentu.

Razlog je činjenica da se sa prvog ne može preći na prethodni, odnosno sa poslednjeg na naredni dokument. Otvaranjem prvog dokumenta omogućeno je „kretanje“ od prve do četvrte strane, uz mogućnost direktnog „skoka“ sa bilo kojeg na početni ili poslednji dokument.



Slika 6.7 Prikaz početne strane

▼ Poglavlje 7

Pokazna vežba: Stilovi u HTML dokumentu

STILOVI

Šta predstavlja stilove u HTML dokumentu?

Predviđeno vreme pokazne vežbe je 30 minuta.

U bilo kom HTML tagu možemo dodati parametar style a kao njegovu vrednost koristimo odreðene deklaracije, koje će se kasnije prilikom učenja CSS-a koristiti.

HTML style atribut se definiše sledećom sintaksom:

```
<tagname style="svojstvo: vrednost;">
```

Svojstvo je CSS svojstvo, Vrednost je CSS vrednost.

Više detalja može se videti izučavanjem CSS-a. Kao preporuka može se navesti i sajt:
https://www.w3schools.com/html/html_css.asp

PROMENA POZADINSKE BOJE

Prikaz menjanja pozadine u HTML-u.

Promena pozadine može se uraditi na dva načina. Prvi način je korišćenjem svojstva „background-color“. Primer koji će promeniti pozadinsku boju u puder plavu.

```
<body style="background-color:powderblue;">
<h1>Ovo je heading 1</h1>
<p>Ovo je paragraf <p>
</body>
```



Slika 7.1 Prikaz promene pozadine

Drugi način je korišćenjem BODY atributa „bgcolor“.

Primer za promenu pozadinske boje:

```
<html>
<body bgcolor="#E6E6FA">
<h1>Hello world!</h1>
</body>
</html>
```

PROMENA BOJE TEKSTA, VELIČINE FONTA KORIŠĆENJEM STILA U HTML-U

Prikaz promene boje teksta koristeći stilove prilikom HTML kodiranja

Jedan od načina promene boje teksta je već prikazani FONT atribut „color“,

Drugi način je korišćenjem stila:

```
<!DOCTYPE html>
<html>
<body>
<h1 style="color:blue;">Ovo je heading 1</h1>
<p style="color:red;">Ovo je paragraf.</p>

</body>
</html>
```

Ovo je heading 1

Ovo je paragraf.

Slika 7.2 Prikaz promene boje teksta

Definisanje familije fonta se definiše korišćenjem "font-family" svojstva.

Primer:

```
<!DOCTYPE html>
<html>
<body>

<h1 style="font-family:verdana;">Ovo je heading</h1>
<p style="font-family:courier;">Ovo je paragraph.</p>

</body>
</html>
```

Ovo je heading

Ovo je paragraph.

Slika 7.3 Prikaz promene familije fonta

Definisanje veličine teksta se izražava u procentima:

```
<!DOCTYPE html>
<html>
<body>

<h1 style="font-size:300%;">Ovo je heading</h1>
<p style="font-size:160%;">Ovo je paragraf</p>

</body>
</html>
```

Ovo je heading

Ovo je paragraf

Slika 7.4 Prikaz promene veličine teksta

KORIŠĆENJE KLASA ZA STILIZOVANJE

Klase se koriste za stilizovanje više elemenata u HTML dokumentu

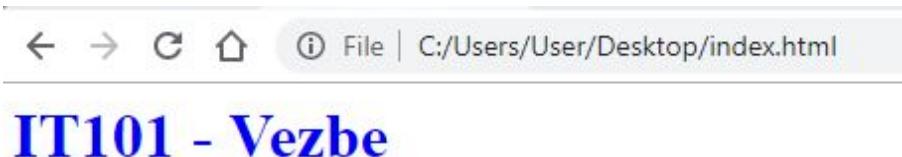
Klase se mogu primeniti na više različitih HTML elemenata. Dovoljno je dodati klasu na neki element i stil koji je definisan za tu klasu će se dodeliti elementu. Klasa se u CSS dokumentu definiše sa tačkom i nazivom klase. Sintaksa je sledeća:

```
<style>
  .blue-text {
    color: blue;
  }
</style>
```

U HTML dokumentu bismo mogli imati element h1 na koji želimo da primenimo definisani stil.

```
<h1 class="blue-text"> IT101 - Vezbe</h1>
```

Rezultat ovog koda je prikazan na slici.



Slika 7.5 Prikaz rezultata

KORIŠĆENJE ID-JA ZA STILIZOVANJE

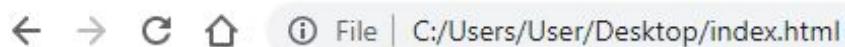
Id se može iskoristiti samo za jedan element u dokumentu

Kao i klase, Id se može koristiti za selektovanje određenog elementa. Id mora biti jedinstven za jedan element, odnosno ne može se koristiti na više različitih elemenata kao klasa. Id se u CSS-u definiše sa znakom "#" i nazivom Id-ja. Sintaksa je sledeća:

```
<style>
#green-text {
    color: green;
}
</style>
```

Ako u HTML dokumentu imamo element h1 koji želimo da stilizujemo, on će izgledati ovako:

```
<h1 id="green-text"> IT101 - Vezbe</h1>
```



The screenshot shows a browser window with the address bar containing 'C:/Users/User/Desktop/index.html'. The main content area displays the text 'IT101 - Vezbe' in a large, bold, green font.

IT101 - Vezbe

Slika 7.6 Prikaz rezultata

✓ Poglavlje 8

Zadatak za samostalni rad: HTML

ZADATAK ZA SAMOSTALNI RAD 1

Kreirati HTML stranicu

Predviđeno vreme za izradu sledećeg zadatka je 10 minuta.

Zadatak

Napraviti HTML stranicu sa svim elementima koji su korišćeni tokom vežbi. Ova stranica treba da prezentuje arhitekturu računara i njegove komponente.

ZADATAK ZA SAMOSTALNI RAD 2

Izrada povezanih HTML stranica

Predviđeno vreme za izradu sledećeg zadatka je 10 minuta.

Zadatak

- Kreirati 3 dokumenta: index.html, strana1.html i strana2.html.
- Na strani index.html postaviti sliku omiljenog glumca ili glumice, sa dva linkovana teksta ispod slike "Biografija" i "Filmovi". Klikom na link "Biografija", otvara se strana1.html, a klikom na link "Filmovi" otvara se strana2.html
- Na strana1.html se nalazi biografija glumca
- Na strana2.html se nalaze filmovi u kome je glumac glumio u formi slika. Svaki film je potrebno predstaviti sa jednom slikom. Klikom na svaku od slika potrebno je da se ode na opis tog filma na IMDB stranici.

ZADATAK ZA SAMOSTALNI RAD 3

Upotreba stilova

Predviđeno vreme za izradu sledećeg zadatka je 10 minuta.

Zadatak

- Nadogradite prethodni zadatak sa povezanim stranicama omiljenog glumca korišćenjem stilova

- Na početku dokumenta potrebno je da stoje ime i prezime glumca – koje će biti Heading1 pri čemu je tekst u nekoj boji.
- Zatim je potrebno da se napiše ko je autor te stranice u formi vašeg imena i prezimena, opet u nekoj boji.
- Stilizujte sajt modifikujući stilove za tekst i pozadinu.

✓ Poglavlje 9

Domaći zadatak

OPIS DOMAĆEG ZADATKA - DZ07

Pripremiti HTML dokument na kome se nalazi vaše ime, prezime i broj indeksa i spisak predmeta koje slušate

Očekivano vreme izrade zadatka: 60min.

Pripremiti HTML dokument koji sadrži:

- Vaše ime, prezime i broj indeksa (centar stranice)
- Smer koji studirate
- Početno slovo imena i prezimena treba da budu boldovani
- Spisak omiljenih filmova, serija, glumaca. Za svaku od ovih kategorija, potrebno je navesti neke od karakterističnih detalja (naziv, godina izdanja, ocena publike, glumci, i sl.) Svaka od ovih kategorija treba da bude prezentovana proizvoljnom bojom.
- Svaku kategoriju navesti kao odvojenu stranicu koja ima link na početnu stranicu
- Spisak za svaku kategoriju treba napisati u obliku numerisane liste
- Svaki unos treba napisati u novom redu, u različitoj boji i veličini fonta
- Naziv svake kategorije mora imati link koji vodi ka stranici te kategorije (filmovi, serije, glumci)
- Ime kategorije podesiti da bude bold, a unos za svaku kategoriju *italic*.
- Vodite računa da sva slova mogu biti ispisana, uključujući čđe.

Napomena:

Domaći zadatak poslati kao: IT101-DZ07-Ime_Prezime_briIndexa.html, gde su Ime, Prezime i briIndexa vaši podaci.

Domaći zadatak pošaljite predmetnom asistentu na e-mail.

▼ Zaključak

ZAKLJUČAK

O lekciji

U ovom predavanju je dat uvod u razvoj veb sajtova, od koraka koje je potrebno primeniti tokom razvoja svakog sajta, pa do same realizacije. Date su osnove korišćenja HTML jezika. U narednim predavanjima će biti dalje razrade veb protokola, standarda i skripting jezika, koji se koriste u razvoju veb stranica.

Literatura

1. Gary B. Shelly, Misty E. Vermaat, Discovering Computers 2011-Introductory: Living in a Digital World, Cengage Learning 2010.



IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

OSNOVE PROGRAMSKIH JEZIKA

Lekcija 08

PRIRUČNIK ZA STUDENTE

IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

Lekcija 08

OSNOVE PROGRAMSKIH JEZIKA

- ▼ OSNOVE PROGRAMSKIH JEZIKA
- ▼ Poglavlje 1: Programske jezice
- ▼ Poglavlje 2: Osnovna struktura programa
- ▼ Poglavlje 3: Petlje
- ▼ Poglavlje 4: Uslovni iskaz
- ▼ Poglavlje 5: Algoritmi
- ▼ Poglavlje 6: Programske paradigme
- ▼ Poglavlje 7: Pokazna vežba: Osnove JavaScript-a
- ▼ Poglavlje 8: Zadaci za samostalni rad: JavaScript
- ▼ Poglavlje 9: Domaći zadatak
- ▼ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

▼ Uvod

UVOD

Cilj ovog univerziteta je da se obrade osnovne karakteristike programskih jezika i strukture podataka koji se koriste u programiranju

Iako cilj ovog kursa nije da se nauči programiranje, neophodno je da se pozabavimo osnovnim karakteristikama programskih jezika i strukturama podataka koji se koriste u programiranju. U ovom predavanju će biti reči o programske jezicima višeg i nižeg nivoa, s tim što će se najviše pažnje obratiti na osnovne programske strukture kao i najčešće korištene programske paradigme.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 1

Programski jezici

JEZICI VIŠEG I NIŽEG NIVOA

Kada se pravi softverska aplikacija programeri će izabrati jezik koji im nudi najbolje funkcionalnosti i pruža najviše fleksibilnosti za taj konkretan slučaj

Kompjuterski program predstavlja niz instrukcija koji se koriste da izdaju računaru neki zadatak. Niz instrukcija napisanih u nekom jeziku predstavljaju program koji se takođe naziva kod. Programski jezik predstavlja skup unapred definisanih reči, simbola, uputstava i skraćenica koji prate određena pravila što je poznato kao jezička sintaksa. Program koji se koristi za razvoj i izgradnju programa se zove razvojno programsко okruženje.

Postoji mnogo različitih jezika koji se koriste za pisanje programskih instrukcija. Jezici su dizajnirani za određenu svrhu i svaki jezik ima svoje prednosti za neke primene, a i nedostatke. Drugim rečima, kada se pravi softverska aplikacija programeri će izabrati jezik koji im nudi najbolje funkcionalnosti i pruža najviše fleksibilnosti za taj konkretan slučaj.

Jezici se mogu podeliti u dve grupe: **jezici nižeg i jezici višeg nivoa**. **Jezici nižeg nivoa su mašinski zavisni** i obično rade samo na određenoj vrsti računara. Jedna instrukcija jezika nižeg nivoa obično predstavlja jednu mašinsku instrukciju, dok jedna instrukcija jezika višeg nivoa obično predstavlja više mašinskih instrukcija. Osim toga, **jezici višeg nivoa su obično nezavisni od mašina** i mogu se pokrenuti na više tipova računara i operativnih sistema.

▼ 1.1 Osnovne strukture podataka

VARIJABLE I KONSTANTE

Sa aspekta računara promenljive i konstante su vrednosti koje su zapisane na nekoj memorijskoj lokaciji

Računarski programi barataju sa veličinama koje mogu biti promenljive ili konstantne tokom izvršenja programa. Tako će vrednost $\pi = 3.14$ biti konstanta tokom celog izvršenja programa, a neka druga veličina, na primer, stanje na računu će se menjati tokom izvršenja programa. **Sa aspekta računara promenljive i konstante su vrednosti koje su zapisane na nekoj memorijskoj lokaciji**. Prilikom prevođenja programa svakoj promenljivoj ili konstanti se dodeljuje relativna memorijska lokacija, a svako njihovo pojavljivanje u algoritmu se

zamenjuje adresom memorijske lokacije. Prilikom startovanja programa ovim promenljivim i konstantama se dodeljuju inicijalne vrednosti tako što se upišu u definisanu memorijsku lokaciju. Uvek kada se tokom izvršenja programa radi sa nekom promenljivom, njena vrednost se čita sa te programske lokacije ili se u nju zapisuje.

Na primer, neka se promenljiva *starost* u nekom programu koristi kao celobrojna promenljiva kojoj se dodeljuju godine starosti osoba. **Neka je trenutna vrednost te promenljive 69.** Svako pojavljivanje promenljive starost će tokom izvršenja programa biti zamenjeno referencom na memorijskoj lokaciji (u ovom slučaju **00110101110100010010001111100101** (Slika 1)).

Adresa memorijske lokacije	Sadržaj
00110101110100010010001111100101	01000101

Slika 1.1.1 Vrednost promenljive starost 01000101 je upisana na memorijskoj adresi 00110101110100010010001111100101

1.2 Tipovi podataka

ŠTA PREDSTAVLJA TIP PODATAKA?

Tip podataka ograničava vrednosti koje promenljiva može da ima, što je praktično definisano veličinom memorijskog prostora koji se dodeljuje tom tipu promenljivih

Svaka promenljiva i konstanta u većini programskih jezika ima definisan **tip** koji je poznat prilikom prevodenja. Na ovaj način se ograničavaju vrednosti koje promenljiva može da ima, što je praktično definisano veličinom memorijskog prostora koji se dodeljuje tom tipu promenljivih. **Tip podataka** takođe definiše operacije, koje su u jeziku podržane, nad tim vrednostima, a kod nekih jezika određuje i značenje operacije.

Svaki jezik ima definisane tipove podataka. Radi ilustracije, ovde će biti prikazani tipovi podataka koji su specificirani u programskom jeziku Java. Tipovi se u Javi mogu podeliti u dve kategorije:

- **Primitivni tipovi**
- **Referentni tipovi.**

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

PRIMITIVNI TIPOVI PODATAKA

Boolean, byte, short, int, long, float, double, char

Programski jezik Java deklariše osam primitivnih tipova podataka. To su:

- **boolean** – ovaj tip podataka se koristi za rad sa logičkim promenljivim. Promenljiva može da uzme samo jednu od dve vrednosti: **true** (tačno) ili **false** (netačno). Za vrednosti promenljive ovog tipa koristi se samo jedan bit u kome se memoriše nula ili jedinica.
- **byte** – ovaj tip podataka se koristi za promenljive čije su vrednosti vrlo mali označeni celi brojevi. Za smeštanje promenljivih ovog tipa se koristi osam bitova, pa vrednost promenljive može da se kreće u intervalu od -128 do 127. Ovaj tip podataka je koristan za čuvanje velikih nizova, kada je potrebno da se sačuva memorija.
- **short** - ovaj tip podataka se koristi za promenljive čije su vrednosti mali označeni celi brojevi. Za smeštanje promenljivih ovog tipa se koristi 16 bitova, tako da vrednost promenljive može da se kreće u intervalu od -32768 do 32767. Slično kao i za tip byte, short se može koristiti kod upotrebe velikih nizova za očuvanje memorije.
- **char** – za promenljive čija je vrednost jedan Unicode karakter koristi se tip char. Za smeštaj promenljive se koristi 16 bitova. Njegova minimalna vrednost je '\u0000' (ili 0), a maksimalna '\uffff' (ili 65,535).
- **int** – za promenljive čije su vrednosti označeni celi brojevi najčešće se koristi tip int. Za smeštanje promenljivih ovog tipa se koriste 32 bita, tako da vrednost promenljive može da se kreće u intervalu od -2.147.483.648 do 2.147.483.647, što je za većinu potreba sasvim dovoljno. Kada je potrebno da vrednost promenljive bude veća od toga, može se koristiti tip long.
- **long** – kada neka promenljiva ima ekstremno velike vrednosti celih brojeva koristi se tip long, koji za smeštaj promenljive koristi 64 bita, odnosno 8 bajta. Vrednost promenljive može da se kreće u intervalu od -9.223.372.036.854.775.808 do 9.223.372.036.854.775.807.
- **float** – ovaj tip podataka se koristi za promenljive čije su vrednosti označeni racionalni (realni) brojevi. U literaturi se promenljive ovog tipa nazivaju brojevi sa plivajućim zarezom (engl. **floating point**). Za smeštanje promenljivih ovog tipa se koriste 32 bita, a koristan je za čuvanje memorije kada se koriste nizovi realnih brojeva.
- **double** – ovaj tip podataka se koristi za promenljive čije su vrednosti ekstremno veliki označeni racionalni (realni) brojevi. Ove promenljive imaju dvostruko veću preciznost od promenljivih tipa float, pa se zbog toga nazivaju double. Za smeštanje promenljivih ovog tipa se koriste 64 bita.

KARAKTERISTIKE TIPOVA PODATAKA

Pregledni prikaz karakteristika tipova podataka

U Tabeli 1 su prikazane karakteristike definisanih tipova podataka.

Tip	Broj bita	Minimalna vrednost	Maksimalna vrednost
boolean	1		
byte	8	-128	127
short	16	-32768	32767
int	32	-2,147,483,648	2,147,483,647
long	64	-9,223,372,036,854,775,808	9,223,372,036,854,775,807
float	32	-3.4x10 ³⁸	3.4x10 ³⁸
double	64	-1.7x10 ³⁰⁸	-1.7x10 ³⁰⁸
char	16	0	65,535

Slika 1.2.1 Tabela-1 Karakteristike tipova podatka

▼ 1.3 Niz

JEDNODIMENZIONALNI NIZOVI

Jednodimenzionalni nizovi odgovaraju vektorima

Nizovi ili polja (engl. **array**) imenovani su set promenljivih istog tipa. Svaka promenljiva u nizu naziva se element niza. Jednodimenzionalni nizovi odgovaraju vektorima. Dvodimenzionalni nizovi odgovaraju matricama. Nizovi mogu biti i višedimenzionalni.

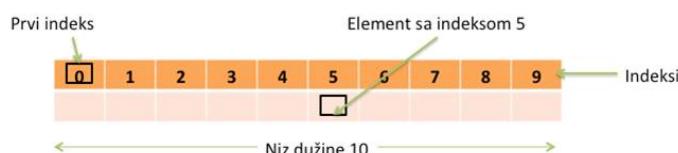
Ako se, na primer, u Javi deklariše jednodimenzionalni niz koji se zove **xKoordinata** čiji su elementi promenljive tipa double, to će se uraditi iskazom

double[] xKoordinata;

Dužina niza se definiše prilikom kreiranja niza. Jednom kada je niz kreiran, njegova dužina je fiksna. Definisanje niza koji ima deset elemenata se vrši iskazom

xKoordinata = new double [10];

Ovaj iskaz rezerviše 10 mesta u memoriji za 10 elemenata ovog niza, gde se svakom elementu može pristupiti preko numeričkog indeksa. Prikaz kako se niz od deset elemenata smešta u memoriji prikazano je na slici 1.



Slika 1.3.1 Niz od deset elemenata

Inicijalna vrednost svih elemenata niza će biti 0.0. Međutim, ako se želi da se promenljivim dodeli neka tačno određena vrednost pri inicijalizaciji, onda se to može postići kao u sledećem primeru:

```
xKoordinata[0] = 100;  
// inicijalizacija drugog elementa  
xKoordinata[1] = 200;  
// itd.  
xKoordinata[2] = 300;  
xKoordinata[3] = 400;  
xKoordinata[4] = 500;  
xKoordinata[5] = 600;  
xKoordinata[6] = 700;  
xKoordinata[7] = 800;  
xKoordinata[8] = 900;  
xKoordinata[9] = 1000;
```

ALTERNATIVNI NAČIN DODELJIVANJA VREDNOSTI ELEMENTU U NIZU

```
double[] xKoordinata = {100, 200, 300, 400, 500, 600, 700, 800, 900,  
1000};
```

Alternativa i kraći način da se dodeli tačno određena vrednost svakom elementu u nizu je:

```
double[] xKoordinata = {  
100, 200, 300,  
400, 500, 600,  
700, 800, 900, 1000  
};
```

Dvodimenzionalni niz mora da sadrži coordinate x i y, a primer takvog niza je dat u sledećem iskazu:

```
double [ ] [ ] koordinate = new double [3] [10];
```

▼ 1.4 String

STRING - NIZ KARAKTERA

String predstavlja niz karaktera

`String` predstavlja niz karaktera. Java koristi standardnu **String** klasu da kreira string. Jednostavan način za kreiranje stringa pod nazivom *message* se može odraditi na sledeći način:

String message = "Happy New Year";

Kao i sa drugim tipovima, tako se i sa stringovima mogu napraviti nizovi. Ako se u Javi pravi jednodimenzionalni niz koji se zove *imena* i koji ima 100 elemenata, onda bi to zahtevalo pisanje sledećeg iskaza:

String [] imena = new String [100];

▼ Poglavlje 2

Osnovna struktura programa

PROGRAMSKE KONSTRUKCIJE

Različiti programski jezici imaju različite programske konstrukcije koje se sastoje od osnovnih elemenata, komentara, separatora, ključnih reči, identifikatora, operatora, izraza, iskaza i komand

Različiti programski jezici imaju različite programske konstrukcije koje se sastoje od osnovnih elemenata, komentara (engl. **comments**), separatora (engl. **separators**), ključnih reči (engl. **keywords**), identifikatora (engl. **identifiers**), operatora (engl. **operator**), izraza (engl. **expressions**), iskaza (engl. **statements**) i komandi (engl. **commands**).

Ovde će se navesti samo najčešće programske konstrukcije koje se sreću u jezicima koji se danas najčešće koriste (Java, C, C++, Visual Basic).

▼ 2.1 Dodeljivanje

DODELJIVANJE (MENJANJE ILI POSTAVLJANJE VREDNOSTI)

Dodeljivanje postavlja ili menja vrednost dodeljenu nekoj promenljivoj

Dodeljivanje (engl. **assignment**) jedna je od osnovnih operacija u bilo kom jeziku. Ona postavlja ili menja vrednost dodeljenu nekoj promenljivoj. To znači da promenljiva sa jednim imenom može da ima različite vrednosti tokom izvršenja programa.

Iskaz kojim se vrši dodeljivanje ima različite oblike, a ovde se daju neki koji se često sreću:

variable := expression (Pascal)

variable = expression (Java, C++)

variable ← expression

Radi bližeg objašnjenja pojma dodeljivanja, posmatraćemo primer promenljive *i* koja predstavlja vrednost brojača u nekoj programskoj petlji. Neka je, na primer, trenutna vrednost promenljive *i* jednaka 5. Pogledajmo sledeći iskaz dodeljivanja:

i = i + 1

Matematički gledano ovaj iskaz nema smisla. Međutim, računarski gledano, on je potpuno ispravan. U ovom slučaju se procesoru daje instrukcija da:

- pročita vrednost promenljive
- da tu vrednost uveća za 1
- da dobijeni rezultat dodeli promenljivoj i , što zapravo znači da dobijeni rezultat upiše u istu memorijsku lokaciju sa koje je pročitana vrednost za i .

Isti iskaz se može napisati kao:

i++;

a koji ima istu funkcionalnost kao i iskaz **i=i+1.**

▼ Poglavlje 3

Petlje

VIDEO - FOR, WHILE I DO WHILE PETLJE

Prodiskutujmo osnovne karakteristike i primere za FOR, WHILE i DO WHILE petlje

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

FOR PETLJA

U FOR petlji broj iteracija je unapred poznat, a svaka interacija ima varijablu koja predstavlja brojač.

Kod imperativnih jezika **for petlja** je kontrolna struktura koja omogućuje da se neki programski blok izvršava iterativno. U ovakvim slučajevima broj iteracija je unapred poznat, a svaka interacija ima varijablu koja predstavlja brojač. U primeru 1 varijabla i predstavlja brojač. Ova varijabla je tipa *integer*. Inicijalna vrednost variable i je 0 (*i=0*). Sa svakom iteracijom, vrednost variable i se uvećava za 1 (*i++*). Ova petlja će biti izvršena 100 puta, tako da vrednost variable i se kreće od 0 do 99 (*i<100*).

Primer 1

Primer dvostrukе petlje u jeziku C:

```
for(i = 0; i < 100; i++) {  
    for(j = i; j < 100; j++) {  
        function(i, j);  
    }  
}
```

Primer 2

Primer for petlje u kojoj se izračunava faktorijel broja 5 u jeziku C ili C++

```
unsigned long factorial = 1;  
  
for (unsigned int counter = 1; counter <= 5; counter++)  
  
    factorial *= counter;  
  
printf("%lu\n", factorial);
```

Primer 3

Primer for petlje u kojoj se izračunava faktorijel broja 5 u jeziku Java

```
long factorial = 1;  
for (int counter = 1; counter <= 5; counter++)  
    factorial *= counter;  
System.out.println(factorial);
```

FOR PETLJA - PRIMER

Koja je vrednost parametra x nakon izvršenja FOR petlje

Pitanje:

Koju vrednost će imati parametar x, kada se ova petlja izvrši do kraja?

int x;

for(x=0; x<10; x++) {}

Odgovor:

x = 10

Pitanje:

Koliko će se puta izvršiti sledeće petlje?

1. **for(intcount = 1; count <= 10; count++)**
2. **for(intcount = 1; count <10; count++)**
3. **for(intcount = 0; count < 10; count++)**
4. **for(intcount = 10; count > 10; count--)**

Odgovor:

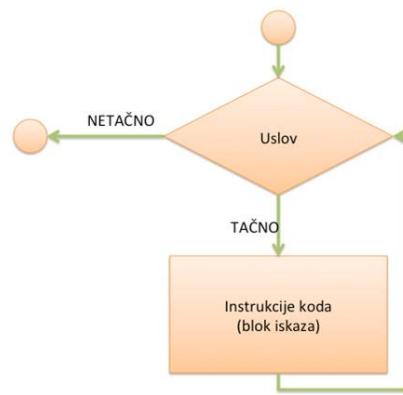
1. 10

- 2. 9
- 3. 10
- 4. 0

WHILE PETLJA

WHILE petlja omogućava ponavljanje izvršenja nekog programskog bloka ako je neki logički uslov zadovoljen

Većina programskih jezika ima while petlju. Ova programska konstrukcija **omogućava ponavljanje izvršenja nekog programskog bloka ako je neki logički uslov zadovoljen**. Slika 1 predstavlja logički tok koji opisuje kako ova petlja funkcioniše.



Slika 3.1 Dijagram while petlje

WHILE konstrukcija se sastoji od uslova i bloka iskaza. Prvo se određuje uslov, pa ukoliko je on zadovoljen (tačno), izvršava se sledeći blok iskaza. Ovo se ponavlja sve dok je uslov zadovoljen. Zato što se u WHILE konstrukciji najpre proverava uslov, pa tek onda izvršava blok, ova konstrukcija se naziva petlja sa prethodnim testiranjem.

Primer 1

Primer while petlje u kojoj se izračunava faktorijel broja u jeziku C ili C++:

```
unsigned int counter = 5;
unsigned long factorial = 1;
while (counter > 0)
    factorial *= counter--;
printf("%i\n", factorial);
```

Primer 2

Primer while petlje u kojoj se izračunava faktorijel broja u jeziku Java:

```
int counter = 5;
long factorial = 1;
```

```
while (counter > 0)
factorial *= counter--; // Multiply, then decrement.
System.out.println(factorial);
```

WHILE PETLJA - PRIMER

Koliko će se puta izvršiti WHILE petlja?

Pitanje: Koliko će se puta izvršiti sledeća petlja? Koja je vrednost parametra x nakon izvršenja petlje?

```
x = 1;
while (x < 2) {
    x = x * 2;
}
```

Odgovor:

Petlja će se izvršiti 1 put.

Nakon izvršenja petlje x=2.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Pitanje: Koliko će se puta izvršiti sledeća petlja?

```
int count = 1;
while (count <= 10) {
    out.println(count);
    count = count -1;
}
```

Odgovor: Petlja će se izvršiti beskonačan broj puta .

WHILE PETLJA - PRIMER 2

Koliko će se puta izvršiti WHILE petlja

Pitanje: Koliko će se puta izvršiti sledeća petlja? Koja je vrednost parametra x nakon izvršenja petlje?

```
x = 1;
while(x < 2) {
    x = x * 2;
```

}

Odgovor:

Petlja će se izvršiti 1 put, a vrednost x je 2.

Pitanje: Koliko će se puta izvršiti sledeća petlja? Koja je vrednost parametra count nakon izvršenja petlje?

```
int count = 1;  
  
while (count <= 10) {  
  
    out.println(count);  
  
    count = count + 1;  
  
}
```

Odgovor:

Petlja će se izvršiti 10 put, a vrednost count je 11.

Pitanje: Koliko će se puta izvršiti sledeća petlja?

```
int count = 1;  
  
while (count != 10) {  
  
    out.println(count);  
  
    count = count + 2;  
  
}
```

Odgovor:

Petlja će se izvršiti beskončan broj puta. Parametar count počinje od broja 1 i onda se povećava na neparne brojeve 3, 5, 7, 9, 11 i tako dalje, ali nikada nije jednak broju 10.

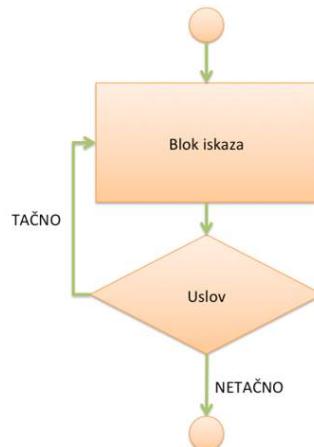
PETLJA DO WHILE

Za razliku od while petlje, prvo se izvršavaju iskazi u bloku, a onda se proverava uslov

U većini programskih jezika **DO WHILE petlja** je programska upravljačka struktura koja dozvoljava da se ponavlja izvršavanje dela programa koji se naziva blok sve dok je zadovoljen neki logički uslov. Slika 2 prikazuje dijagram petlje do while. Za razliku od WHILE petlje, prvo se izvršavaju iskazi u bloku, a onda se proverava uslov. Ako je uslov ispunjen, onda se blok izvršava ponovo. Ovo se ponavlja sve dok je logički uslov ispunjen.

Treba napomenuti da je moguće, a ponekada i neophodno, da se uslov uvek ispunjava, što znači da će se petlja beskonačno izvršavati. U tom slučaju je neophodno da se odredi struktura koja predviđa izlaženje iz petlje. Takođe, korišćenje do while konstrukcije garantuje da će se blok izvršiti bar jednom.

Zbog toga što se testiranje uslova vrši nakon izvršavanja bloka, ova konstrukcija se naziva petlja sa naknadnim testom, za razliku od WHILE petlje koja ima prethodni test.



Slika 3.2 Dijagram petlje DO WHILE

Primer 1

Primer do while petlje u jeziku C koja će se izvršiti 3 puta:

```
x = 0;  
do  
{  
    x = x + 1;  
}while (x < 3);
```

DO WHILE PETLJA - PRIMER

Koliko će se puta izvršiti DO WHILE petlja?

Pitanje: Koliko će se puta izvršiti sledeća petlja? Koja je vrednost parametra x nakon izvršenja petlje?

```
x = 0;  
do {  
    x = x * 2;  
} while(x < 2)
```

Odgovor:

Petlja će se izvršiti beskonačni broj put.

Ova petlja se neće izvršiti do kraja, a vrednost x je uvek 0.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 4

Uslovni izraz

NAČIN IZVRŠAVANJA USLOVNOG ISKAZA

Zavisno od ispunjenosti uslova procesor će izvršiti jedan od blokova koji su deo ove konstrukcije

Većina računarskih jezika ima jednu ili više različitih konstrukcija koje omogućuju uslovno izvršenje izraza ili blokova izraza. Zavisno od ispunjenosti uslova procesor će izvršiti jedan od blokova koji su deo ove konstrukcije.

Tipičan oblik ove konstrukcije je:

If (condition) Then

(Blok izraza #1)

Else

(Blok izraza #2)

End If

U ovom slučaju, procesor će prvo ispitati uslov. Ukoliko je uslov ispunjen, prvo će se izvršiti prvi blok izraza (Blok izraza #1 u primeru). U suprotnom, prvi blok neće biti izvršen, već drugi blok (Blok izraza #2 u primeru).

Slične ovoj konstrukciji su i konstrukcije **switch** i **case**.

▼ Poglavlje 5

Algoritmi

ŠTA JE ALGORITAM?

Algoritam je dobro definisana računarska procedura koja uzima neku vrednost ili set vrednosti kao ulaz i proizvodi neku vrednost ili set vrednosti kao izlaz

Neformalno, algoritam je dobro definisana računarska procedura koja uzima neku vrednost ili set vrednosti kao ulaz i proizvodi neku vrednost ili set vrednosti kao izlaz. Dakle, algoritam je sekvenca računarskih koraka koji transformišu ulaz u izlaz.

Algoritam se definiše i kao alat za rešavanje dobro definisanih računarskih problema. Pri definisanju problema potrebno je definisati željeni odnos između ulaza i izlaza. Algoritam opisuje računarske procedure kojim će se postići takav odnos između ulaza i izlaza.

Kao primer definisanja problema razmotrimo vrlo čest problem sortiranja brojeva po neopadajućem redosledu. Ovo je formalna definicija problema:

Ulaz: niz n brojeva $a_1, a_2, a_3, \dots, a_n$

Izlaz: preuređeni niz istih brojeva $b_1, b_2, b_3, \dots, b_n$ such that $b_1 \leq b_2 \leq b_3 \leq \dots \leq b_n$.

Ako bi, na primer, ulaz u algoritam bio niz brojeva 44, 12, 82, 14, 55, onda bi izlaz iz algoritma trebalo da bude niz 12, 14, 44, 55, 82. Ovakav ulaz se naziva instanca problema sortiranja. Generalno, **instanca problema** se sastoji od ulaza potrebnog da se izračuna rešenje problema, a koji zadovoljava ograničenja koja nameće algoritam.

Algoritam je ispravan ukoliko svaki ulaz daje tačan izlaz. Algoritam može da se opiše nekim govornim jezikom, nekim simboličkim jezikom, pseudokodom, računarskim jezikom, ali se može realizovati i nekim hardverskim sklopom.

Isti problem često može da bude rešen različitim algoritmima. Poželjno je da algoritam bude efikasan. Uobičajena mera efikasnosti algoritma je vreme potrebno za izvršavanje programa radi dobijanja izlaza. Pored toga, treba težiti ka algoritmima koji ne zahtevaju prevelike memorijske resurse.

OSNOVNE KARAKTERISTIKE ALGORITMA

Algoritam treba da bude konačan, dobro definisan

Algoritam treba da ispunjava tri osnovne karakteristike:

1. Treba da bude konačan - što znači da algoritam treba da ima neki izlaz. Ako se algoritam nikada ne završi onda on ne može da da rešenje za problem za koji je postavljen.
2. Treba da bude dobro definisan - svaki korak algoritma mora biti precizno definisan; uputstva treba da budu nedvosmisleno definisana za svaki mogući slučaj.
3. Treba da bude efikasan - algoritam trebalo da reši problem za koji je postavljen na što efikasniji način.

Informally, an algorithm is any well-defined computational procedure that takes some value, or set of values, as input and produces some value, or set of values, as output. An algorithm is thus a sequence of computational steps that transform the input into the output.

Source: Thomas H. Cormen, Chales E. Leiserson (2009), Introduction to Algorithms 3rd edition.

Važno je naglasiti da se algoritmi ne koriste samo u računarskim naukama, oni su važan entitet i u matematici.

Međutim, neki od algoritama koji se koriste u računarskim naukama su:

- Merge Sort, Quick Sort i Heap Sort
- Furijeva transformacija
- Dijkstra algoritam
- RSA algoritam
- Bezbedni hash algoritam
- Faktorizacija celih brojeva
- Algoritmi za kompresiju podataka
- Generator nasumičnih brojeva

VIDEO

Computer Science Basics: Algorithms

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 6

Programske paradigme

ŠTA OBEZBEĐUJE PROGRAMSKA PARADIGMA?

Programska paradigma obezbeđuje i definiše način viđenja ili daje ideju programeru kako se izvršava program

Programska paradigma obezbeđuje i definiše način viđenja ili daje ideju programeru kako se izvršava program. Na primer, u objektno-orientisanom programiranju programer posmatra program kao skup objekata koji međusobno deluju jedan na drugi, dok u proceduralnom programiranju postoji neka unapred definisana procedura ili tok izvršenja programa.

Neki programski jezici obezbeđuju samo jednu programsku paradigmu, a neki više. Na primer, jezici Java i Smalltalk podržavaju samo objektno-orientisani paradigmu, a C++ podržava elemente proceduralnog programiranja, objektno-baziranog programiranja, objektno-orientisanog programiranja i generičkog programiranja tako da programer može da odluči koje će paradigmе koristiti. Neke programske paradigmе su poznatije po tome što zabranjuju nego po tome što omogućuju. Na primer, paradigm strukturnog programiranja sprečava pojavu takozvanog špageti koda tako što zabranjuje korišćenje ikaza tipa go to.

Postoje mnoge programske paradigmе, ali će se ovde navesti originalna imena samo nekih:

- Structured programming, suprotno od Unstructured programming
- Imperative programming, suprotno od Declarative programming
- Message passing programming, suprotno od Imperative programming
- Procedural programming, suprotno od Functional programming
- Value-level programming, suprotno od Function-level programming
- Flow-driven programming, suprotno od Event-driven programming
- Scalar programming, suprotno od Array programming
- Constraint programming, suprotno od Logic programming
- Component-oriented programming (kao u OLE)
- Symbolic programming (kao u programu Mathematica)

JEZICI NIŽEG NIVOA

Jezici nižeg nivoa su mašinski zavisne, odnosno zavise od rada samog računara na mašinskom nivou

Jezici nižeg nivoa su mašinski zavisne, odnosno zavise od rada samog računara na mašinskom nivou. Postoje dva tipa jezika nižeg nivoa: **mašinski jezici nižeg nivoa** i **asembli** (engl. **assembly**) jezici. Jedini jezik koji računar zaista razume je mašinski jezik,

koji je korišćen u prvim generacijama programskih jezika. Mašinski jezik koristi skup binarnih brojeva koje predstavljaju neku instrukciju. Ovakvo programiranje nije jednostavno i intuitivno za čoveka, a pri tom oduzima mnogo vremena. Asembli jezici koriste skup instrukcija u formi simbola koji se koriste da se napiše kod. Jedna instrukcija asembla predstavlja jednu jezičku reč. Ove instrukcije su obično skraćenice engleskih reči **addition, load, multiply, shift**, itd. Iako je reč o rečima koje su razumljive za čoveka, logika asembla je teška za učenje.

PROCEDURALNO PROGRAMIRANJE

Proceduralno progamiranje se naziva i imperativnim programiranjem, a ima za cilj da izvrši niz koraka kako bi izvršio neki zadatak

S obzirom da su mašinski jezici veoma teški za korišćenje i da njihovo korišćenje oduzima puno vremena pri pisanju koda, nastala je potreba da se kreira jezik koji je približniji čoveku, a koji i dalje može da komunicira logikom računara. Jedan od tipova jezika višeg nivoa jeste i proceduralno progamiranje. **Proceduralno progamiranje se naziva i imperativnim programiranjem, a ima za cilj da izvrši niz koraka kako bi izvršio neki zadatak.** Proceduralno programiranje se zasniva na nizu poziva odredjenih procedura, gde procedura može biti rutina, podrutina, funkcija i sl.

U proceduralnom programiranju programer piše instrukcije koristeći sintaksu određenog programskog jezika. Ovi jezici koriste engleske reči ili neku formu tih reči zajedno sa aritmetičkim simbolima. Ove instrukcije se konvertuju u mašinski jezik pomoću prevodioca ili kompjajlera (engl. **compiler**). Kompajler je poseban program koji prevodi ceo program pre izvršavanja. Postoji mnogo proceduralnih jezika, međutim, samo mali broj je često korišćen, a to su jezici kao što su C i COBOL.

Programski jezik C je razvijen 1970 godine od strane Bell laboratorije za razvijanje operativnog sistema UNIX. Iako je bio kreiran da razvija sistemski softver, koristi se i za kreiranje aplikativnog softvera. Mnogi programeri koriste C za poslovna i naučna rešenja. C je imao veliki uticaj i na druge programske jezike kao što je na primer C++.

COBOL (**Common Business-Oriented Language**) je programski jezik koji je razvijen za razvoj poslovnih aplikacija. Nove verzija COBOL je COBOL 2002 koji podržava objektno-orientisani razvoj. COBOL je poznato po svojoj jednostavnosti i lakoći održavanja pošto su njegovi iskazi veoma slični engleskom jeziku. Izuzetno je koristan za automatizaciju platnih spiskova, obračuna, finansija i administrativnih sistema.

OBJEKTNO-ORIJENTISANO PROGRAMIRANJE

OO programiranje se zasniva na ideji da je program sastavljen od posebnih jedinica koje se nazivaju objekti

Paradigma objektno-orientisanog programiranja, ili skraćeno OOP, zasniva se na ideji da je program sastavljen od posebnih jedinica koje se nazivaju objekti. Svaki objekat je sposoban

da prima poruke, obrađuje podatke i šalje poruke ka drugim objektima. OOP se zasniva na konceptima kao što su objekat, apstrakcija, učaurenje, polimorfizam i nasleđivanje.

Objekat je softverski skup koji se sastoji od podataka i programskog koda kojim su opisane metode kojima objekt raspolaže. Objekti su osnova za modularnost i strukture u OOP. Objekat je reprezentacija nekog entiteta. Ne mora da predstavlja sve osobine realnog objekta, nego samo neke koje su od interesa za problem. Objekat može biti realan ili zamišljen (konceptualan). Objekti se definišu kao instance neke klase. Svaki objekat ima tri karakteristike: stanje, ponašanje i identitet.

Objekti u OOP se mogu posmatrati kao objekti u stvarnom svetu. Oni imaju stanje (engl. **state**) i ponašanje (engl. **behavior**). Ako posmatramo psa kao objekat, stanje može biti ime, boja, rasa, a ponašanje je lajanje, hvatanje loptice ili mahanje repom. Objekti u softveru takođe imaju stanje i ponašanje, gde je stanje polje - **field** (variabla u programskom jeziku), a ponašanje se izražava kroz metodu **-method** (funkcija u programskom jeziku). Stanje objekta se izražava kroz attribute. Pošto se stanje jednog istog objekta može promeniti, tako se i vrednost atributa menja. Ponašanje definiše na koji način objekat reaguje na zahteve drugih objekata. Ponašanje simbolizuje šta objekat može da uradi i definiše metode (procedure) koji su deo klase kojoj objekat pripada. Svaki objekat ima identitet, što ga čini jedinstvenim, čak i ako je njegovo stanje identično stanju drugog objekta iste klase.

PRIMER

Razmotrimo klasu cilindara

Svaki cilindar koji je instanca te klase je jedan objekt. Tako da cilindar može da ima karakteristike kao što su poluprečnik (r), visina (h) i specifičnu težinu (w). Ove karakteristike su definisane u klasi Cilindar.

Instanca ove klase mogu biti cilindar A i cilindar B (Slika 1), gde cilindar A ima karakteristike

$r = 5, h = 1, w=19$

dok su karakteristike cilindra B

$r = 2, h = 6, w=14$

Oba cilindra imaju iste osobine i isto ponašanje pa pripadaju istoj klasi. Posmatrajmo cilindar B. Njegovo stanje je opisano njegovim atributima $r = 2$, $h = 6$ i $g = 14$. Njegova zapremina se izračunava po poznatom izrazu, čime je definisano njegovo ponašanje. Na kraju ovaj objekat ima svoj identitet **cilindar B**.



Slika 6.1 Instance klase cilindar

OSOBINE OO PROGRAMIRANJA

Osnovne osobine OO programiranja su učaurenje, nasleđivanje, polimorfizam, apstrakcija

Učaurenje (engl. **encapsulation**) je osobina objekta da ne mogu drugi korisnici na neočekivan način promeniti njegovo interno stanje. Pristup njegovim atributima jedino je dozvoljen njegovim sopstvenim metodama. Na zahtev drugih objekata jedan objekat može svoje podatke i metode učiniti dostupnim po pravilima koje on definiše.

Nasleđivanje je osobina OO jezika koja omogućuje da se stvore potklase koje nasleđuju osobine od svojih „roditelja“, ali mogu da imaju i svoje specijalizovane osobine, odnosno podatke i metode. Na primer, planinski bicikl, drumski bicikl i tandem bicikl su svi bicikli, međutim, oni takođe imaju karakteristike koje se razlikuju jedna od druge (na primer, imaju dva mesta ili jedno mesto). Zahvaljujući ovoj osobini, omogućen je polimorfizam i učaurenje.

Polimorfizam označava sposobnost prihvatanja više formi ili oblika. U programiranje, polimorfizam omoguće podklasama da definišu svoja unikatne karakteristike, kao i da naslede neke karakteristike svoje „parent“ klase. Na primer, na poruku „izračunaj zapreminu“ objekat klase cilindar će to uraditi na jedan, a objekat klase kocka na drugi način.

Apstrakcija je sposobnost objekta da manipuliše samo podacima koji su potrebni njegovim metodama. Prilikom kreiranja nove klase definišu se atributi i metode koje klasa treba da ima. Izbor atributa zavisi od namene klase, tj. od metoda kojima će se opisati ponašanje klase.

Kao što je rečeno, objekti međusobno komuniciraju razmenjujući poruke. Poruke mogu da sadrže i informacije koje se nazivaju **parametri**. Poruka se sastoji od tri komponente:

- imena objekta kome se upućuje poruka
- imena metode koja treba da se izvede
- parametara koji su potrebni (ako jesu) metodi

MODULARNOST

Modularnost omogućuje lako korišćenje jednom napisanih modula u drugim programima

Objekat koji je primio poruku, na osnovu primljenih parametara može da izvrši neku svoju metodu i po potrebi pošalje novu poruku pošiljaocu ili nekom drugom objektu.

Sve osobine OO programiranja omogućuju da se OO programi pišu modularno, pri čemu su moduli, budući da sadrže metode i podatke, nezavisni od drugih modula, što ima mnoge prednosti. Modularnost omogućuje lako korišćenje jednom napisanih modula u drugim programima (engl. **reusability**), čime se proces programiranja ubrzava. Takođe je održavanje programa i nalaženje grešaka olakšano.

Prvi jezik koji je uveo pojam objekta bio je Simula 67. Jezik Smalltalk je prvi objektno-orientisani jezik. Danas postoje mnogi jezici koji imaju objektno-orientisanu paradigmu. Među najpoznatijim su: Ada 95, C#, C++, Delphi, Eiffel, Fortran 2003, Java, Perl, PHP, Python i Visual Basic.

EVENT-DRIVEN PROGRAMIRANJE

Pojam event-driven programa bi mogao da se prevede kao program čije se izvršavanje zasniva na događajima ili zavisi od eksternih događaja

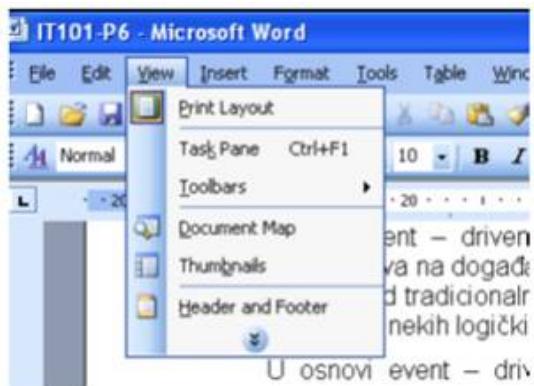
Pojam **event-driven** programa bi mogao da se prevede kao program čije se izvršavanje zasniva na događajima ili zavisi od eksternih događaja. Ovakvi programski jezici se po tome bitno razlikuju od tradicionalnih, kod kojih se kontrolom toka programa upravlja u tačkama grananja zavisno od ispunjenja nekih logičkih uslova. Event-driven programiranje se obično sadrži od dva dela:

- Detekcija događaja
- Upravljanje događajima

U osnovi **event-driven** programske paradigmе je događaj (engl. **event**). Pod događajem se podrazumeva neka akcija korisnika na ulaznim uređajima, neki događaj koji je proizveo operativni sistem, vremenski događaj ili događaj koji je proizvela neka druga aplikacija. Glavnu upravljačku strukturu ovakvih programa čini sistemski preprogramirana petlja koja ispituje događaje (engl. **event loop**) i reaguje na njih aktiviranjem adekvatnih programskih modula. Ove procedure se nazivaju manipulatori događaja (engl. **event handlers**). Pored toga, postoji i poseban modul koji se naziva dispečer (engl. **dispatcher**) čiji je zadatak da poziva manipulatore događaja i čuva neprocesirane događaje koji su se u međuvremenu dogodili.

Neka nam kao ilustracija mogućih događaja posluži program Word. Upravljanje radom ovog programa vrši se preko događaja. Korisnik u svakom trenutku može da prekine unos teksta i da pozicionira pokazivač iznad nekog elementa menija. Ovaj događaj se naziva **mouse over**. U tom slučaju će se promeniti boja pozadine tog menija ili ako je u pitanju neka ikona iz linije

sa zadacima (engl. **task bar**), pojaviće se naziv komande koji odgovara ikoni. Ali ako se pri tom pritisne levi taster miša, registrovaće se drugi događaj koji se zove **klik** i kojim se aktivira izvršenje neke komande (Slika 2).



Slika 6.2 Primer grafičkog korisničkog interfejsa event-driven programa

▼ Poglavlje 7

Pokazna vežba: Osnove JavaScript-a

OSNOVE JAVASCRIPT-A

JavaScript je interpreterski jezik, a svi moderni web browseri imaju ugradjen JS interpreter

Predviđeno vreme pokazne vežbe je 105 minuta.

JavaScript je jedan od najpopularnijih skriptnih jezika današnjice. Stekao je popularnost zbog široke upotrebe na webu, u početku za animacije, validaciju forme i slične funkcije na klijentskoj strani, a danas se koristi i za kreiranje kompletnih aplikacija (**rich client**), pa čak i za serversko programiranje.

JavaScript je interpreterski jezik, a svi moderni web browseri imaju ugradjen JS interpreter. Ovaj jezik je objektno-orientisan, a ima i neke elemente preuzete iz funkcionalnog programiranja. JS sistem tipova se može klasifikovati kao dinamički ili **weakly typed**.

Sintaksa jezika JavaScript liči na sintaksu jezika C. Važno je naglasiti da JS nema nikakve veze sa jezikom Java.

DEKLARACIJA I INICIJALIZACIJA VARIJABLI

Interpreter će sam zaključiti o kojem se tipu radi prilikom inicijalizacije.

Pošto je JS dinamički jezik, varijable nije potrebno posebno deklarisati i navoditi njihov tip. Ovo ne znači da tipovi podataka ne postoje, nego da će interpreter sam zaključiti o kojem se tipu radi prilikom inicijalizacije.

Sintaksa za deklarisanje i inicijalizaciju lokalne promenjive je sledeća:

var ime_promenjive = vrednost;

Na primer, deklarisaćemo varijable a i b tipa broj i string.

var a = 10;

var b = "IT101";

String literali se navode uz pomoć znakova navoda (" " i ' ')

ARITMETIČKI OPERATORI

*Javascript podržava osnovne aritmetičke operatore +, -, * i /.*

Javascript podržava osnovne aritmetičke operatore +, -, * i /. Na primer,

```
var a = 10;  
  
a = 10 + 1; //a postaje 11
```

Operator % označava ostatak od deljenja.

U radu sa stringovima, operator + označava konkatenaciju (spajanje stringova). Na primer,

```
var tekst = "IT 101 " + " JavaScript"; // Varijabla tekst će biti IT101 JavaScript
```

OPERATORI POREDJENJA I LOGIČKI OPERATORI

Operatori == i != su operatori poredjenja jednakosti, tj. vraćaju true ako su operandi jednaki ili nisu jednaki, respektivno.

Javascript ima sledeće operatore poredjenja: <, >, <=, >=, ==, !=.

Operatori == i != su operatori poredjenja jednakosti, tj. vraćaju true ako su operandi jednaki ili nisu jednaki, respektivno.

Logički operatori u JS su sledeći:

- **&& (and)**
- **|| (or)**
- **! (not)**

DEFINISANJE STRINGA

String je tip podatka koji se koristi za prikazivanje teksta, odnosno karaktera

Kao i sa ostalim tipovima podataka, kod kreiranja stringa, prvo se deklarše promenljiva i zatim joj se dodeljuje vrednost. Kada je u pitanju string, vrednost se navodi pod znacima navoda.

```
var tekst = 'IT101 vezbe'
```

ili

```
var tekst = "IT101 vezbe"
```

Pri definisanju stringa mogu se koristiti obe vrste navodnika.

Stringovi se takođe mogu i spajati zajedno, kao i sa drugim tipovima podataka. Kod u nastavku će spojiti vrednosti dve promenljive i od njih napraviti jedan string.

```
var tekst1 = 'Vezbe';
var tekst2 = 'IT101'
document.write(tekst1 + tekst2);
```

Ovo će dati sledeći rezultat:

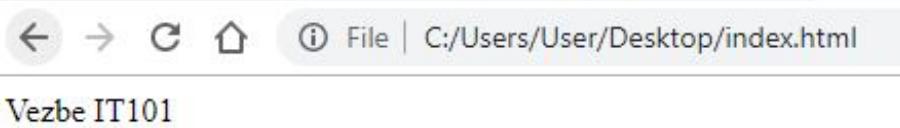


Slika 7.1 Prikaz rezultata

U ovom slučaju stringovi su spojeni, ali između njih nema razmaka. Ovo se može rešiti sledećim kodom:

```
var tekst1 = 'Vezbe';
var tekst2 = 'IT101'
document.write(tekst1 + " " + tekst2);
```

Rezultat će sada izgledati ovako:



Slika 7.2 Prikaz rezultata

NIZOVI

Pristupanje elementu niza po poziciji elementa (indeksu) naziva se indeksiranje u niz.

Nizovi u JS prate C sintaksu. Niz se inicijalizuje na sledeći način:

var niz = [10, 20, 30];

Pristupanje elementu niza po poziciji elementa (indeksu) naziva se indeksiranje u niz.

Pristupanje trećem i n-tom elementu niza:

niz [3];

niz [n];

Na sličan način moguće je dodeliti vrednost elementu niza, npr:

niz[3] = 47;

DODAVANJE ELEMENATA U NIZ

Niz se automatski proširuje dodavanjem elemenata

Nizovi u JS nemaju fiksnu dužinu. Niz se automatski proširuje dodavanjem elemenata. Za dodavanje elementa na kraj niza koristi se funkcija push:

niz.push("nesto");

Funkcija pop briše i vraća element sa kraja niza.

niz.pop;

Dužini niza se priistupa na sledeći način:

niz.length

GRANANJE

Grananje u programskoj logici se postiže korišćenjem if/else konstrukcije

Grananje u programskoj logici se postiže korišćenjem if/else konstrukcije. Opšti oblik ove konstrukcije je:

```
if (uslov)
{
    Blok koda 1...
}
else
{
    Blok koda 2...
}
```

Uslov je izraz koji se evaluira u boolean vrednost. Ako je uslov tačan, izvršiće se prvi blok koda, a u suprotnom se izvršava drugi blok koda. If blok se može koristiti bez else bloka.

PETLJE

Za ponavljanje operacija određeni broj puta koriste se petlje

Za ponavljanje operacija određeni broj puta koriste se **petlje** (engl. **loops**). Ovde ćemo obraditi sledeće vrste petlji:

while (uslov)

```
{  
  Blok koda...  
}
```

U ovom slučaju, blok koda se izvršava sve dok je uslov tačan. U svakoj iteraciji petlje, uslov se ponovo proverava.

```
do  
{  
  Blok koda...  
}  
while(uslov);
```

Slično kao i prethodni slučaj, s tim da se u ovoj vrsti petlje blok koda izvršava bar jednom, tj. prvo će se izvršiti blok koda, pa onda proveriti uslov.

```
for( inicijalizacija; uslov; inkrement/dekrement)  
{  
  Blok koda...  
}
```

Deo inicijalizacije se izvršava jednom pre početka izvršavanja petlje. Blok koda u telu petlje se izvršava sve dok je uslov tačan. U trećem delu petlje se vrši ažuriranje stanja, a to je obično povećavanje ili smanjenje vrednosti brojača definisanog u prvom delu.

PRIMER PETLJE

Prva petlja će se izvršiti 10 puta i ispisati brojeve od 1 do 9.

Analizirati sledeće blokove koda:

Ova petlja će se izvršiti 10 puta i ispisati brojeve od 1 do 9.

```
var i = 0; // Varijabla i je inicijalizovana na vrednost 0  
  
while( i < 10 )  
{  
  i = i + 1;  
  console.log (i);  
}
```

Prvo se definiše početna vrednost promenljive i. Dodeljuje joj se vrednost 0. Uslov while petlje je da se izvršava sve dok je i manje od 10. Zatim se u telu petlje definiše kako će se ona

izvršavati. U ovom slučaju, linija 5 određuje da će se svakom iteracijom petlje na promenljivu i dodati 1 i zatim će se to sačuvati kao vrednost promenljive. Sada će promenljiva i imati vrednost 1. U sledećoj iteraciji promenljiva će imati vrednost 2, i tako sve dok vrednost ne bude 10. Tada će petlja prestati da se izvršava.

Ova petlja će se izvršiti 10 puta i ispisati brojeve od 0 do 10.

```
var i = 0; // Varijabla i je inicijalizovana na vrednost 0

while( i < 10 )
{
    console.log (i);
    i = i + 1;
}
```

U čemu je razlika izmedju ove dve petlje? U prvom slučaju se varijabla i inkrementuje pre ispisivanja, tako da je prva vrednost koja se ispiše 1, a u drugom slučaju će se varijabla prvo ispisati a njena vrednost na početku prve iteracije je 0.

PRIMER EKVIVALENTNE FOR PETLJE

Ovaj primer je ekvivalentan prvom slučaju.

Ovaj primer je ekvivalentan prvom slučaju.

```
for( var i = 0;i < 10; i++)
{
    console.log (i);
}
```

U ovom primeru koristi se for petlja. Ponovo se definiše promenljiva i, njena početna vrednost je 0. Zatim sledi uslov, i na kraju brojač koji se svakom iteracijom povećava za 1.

ITERACIJA KROZ NIZ

Često je potrebno proći kroz niz i izvršiti neku operaciju nad svakim članom niza

Često je potrebno proći kroz niz i izvršiti neku operaciju nad svakim članom niza (iterirati kroz niz). Ovo postižemo tako što inicijalizujemo brojač na vrednost 0 i u svakoj iteraciji koristimo brojač kao indeks u niz. U svakoj iteraciji brojač se inkrementuje za 1. Uslov u petlji je da je brojač manji od dužine niza. Za određivanje dužine niza koristi se niz.length property.

Sledeći primer postavlja sve članove niza na vrednost 0.

Prvo je kreiran niz koji se zove "niz". Zatim je definisana for petlja u kojoj je definisana promenljiva i sa početnom vrednošću 0. Sledi uslov, da se petlja izvršava sve dok je brojač manji od dužine niza, i na kraju brojač koji se inkrementuje. U samoj petlji piše se kod kojim

će se svaki član menjati sa 0. Jednom članu niza se pristupa na sledeći način: niz [i] = nova vrednost. Prilikom prve iteracije, brojač će biti 0 i predstavlja prvi element niza (10), i njega će zameniti sa nulom. U sledećoj iteraciji, brojač će imati vrednost 1, što predstavlja drugi element niza, odnosno 11. Sada će se i ova vrednost zameniti nulom. Ovo će se ponavljati sve dok se svi elementi niza ne zamene nulom.

```
var niz = [ 10, 11, 12, 15, 9 ];

for( var i = 0; i < niz.length; i++)
{
    niz[i] = 0;
}
// [ 0, 0, 0, 0, 0]
```

PRIMER ITERACIJE KROZ NIZ

Drugi primer na svaki element niza dodaje broj 100.

Drugi primer na svaki element niza dodaje broj 100.

Ono što je drugačije u ovom primeru je što se umesto zamene elemenata, sada na svaki element dodaje 100.

```
var niz = [ 10, 11, 12, 15, 9 ];

for( var i = 0; i < niz.length; i++)
{
    niz[i] = niz[i] + 100;
}
// [ 110, 111, 112, 115, 109]
```

TRAŽENJE NAJVEĆEG ELEMENTA U NIZU

U svakoj iteraciji proveravamo da li je trenutni element niza veći od trenutnog maksimuma, i ako jeste postavljamo trenutni maksimum na vrednost tog elementa

Sledeći primer je program koji nalazi vrednost najvećeg elementa niza. Uvodimo dodatnu varijablu max, koja se na početku inicijalizuje na vrednost prvog elementa niza. U svakoj iteraciji proveravamo da li je trenutni element niza veći od trenutnog maksimuma, i ako jeste postavljamo trenutni maksimum na vrednost tog elementa.

```
var niz = [10, 11, 2, 100, 6];
var max = niz[0];

for ( var i = 0; i < niz.length; i++)
{
```

```
if (niz[i] > max)
{
    max = niz[i];
}
}

console.log(max); // ispisuje broj 100
```

ISPISIVANJE BROJEVA DELJIVIH SA 7

Sledeći primer je program koji treba da ispiše sve brojeve deljive sa 7 do broja 100.

Sledeći primer je program koji treba da ispiše sve brojeve deljive sa 7 do broja 100. Jedno moguće rešenje je da iteriramo petljom kroz sve brojeve do broja 100 i u svakoj iteraciji proveravamo da li je broj deljiv sa 7, tj. da li je ostatak od deljenja datog broja i broja 7 jednak 0.

```
for ( var i = 1; i <= 100; i++)
{
    if(i % 7 == 0)
    {
        console.log(i);
    }
}
```

UGNJEŽDENE PETLJE

Za svaku iteraciju vanjske petlje izvršavaju se sve iteracije unutrašnje petlje.

Petlje mogu biti ugnježdene jedna u drugu. U tom slučaju, za svaku iteraciju vanjske petlje izvršavaju se sve iteracije unutrašnje petlje.

```
for(var i = 0; i < 3;i++)
{
    for (var j = 0; j < 3; j++)
    {
        console.log("i , j = " + i + " " + j);
    }
}
```

Ovaj blok koda će ispisati sledeće:

i, j = 0 0

i, j = 0 1

i, j = 0 2

i, j = 1 0

i, j = 1 1

i, j = 1 2

i, j = 2 0

i, j = 2 1

i, j = 2 2

REŠENI ZADACI ZA VEŽBU

Prikaz zadataka sa rešenjem

Predviđeno vreme izrade sledećeg zadatka je 10 minuta.

1. Zadatak

Napisati program kojim se štampaju brojevi od 1 do 10, i pored svakog označava da li je paran ili neparan:

1 je neparan

2 je paran

3 je neparan

...

(Uputstvo: koristite % operator.)

Rešenje:

```
for(i=1; i<=10; i++){
    document.write(i);
    if(i % 2 == 0) document.write(" je paran<br/>");// Kada je ostatak pri
deljenju 0, broj je paran
    else document.write(" je neparan<br/>")
}
```

Predviđeno vreme izrade sledećeg zadatka je 10 minuta.

2. Zadatak

Napisati program kojim se štampaju prvih 20 pozitivnih celih brojeva i njihovi faktorijeli.

Rešenje:

```
for(i=1; i<=20; i++){
    fakt = 1;
    for(j=1; j <=i; j++)
        fakt = fakt*j;
    document.write(fakt+"<br/>");
}
```

Predviđeno vreme izrade sledećeg zadatka je 10 minuta.

3. Zadatak

Napisati program kojim se štampa sledeći trougao:

k
kk
kkk
kkkk
kkkkk
kkkkkk
kkkkkkk

```
for(var i = 0; i < 7; i++){
    for(var j = 0; j < i + 1; j++){
        document.write("k");
    }
    document.write("<br/>");
}
```

✓ Poglavlje 8

Zadaci za samostalni rad: JavaScript

ZADACI ZA SAMOSTALNO VEŽBANJE

Napisati program koji će naći najmanji element u nizu.

Predviđeno vreme za izradu sledećih zadataka je 30 minuta.

1. Napisati program koji će naći najmanji element u nizu. (Vreme izrade: 3 minuta)
2. Napisati program koji će naći drugi najmanje element u nizu. (Vreme izrade: 3 minuta)
3. Napisati funkciju koja prima dva broja, i vraća veći od njih. (Vreme izrade: 3 minuta)
4. Napisati program koji ispisuje indeks najvećeg elementa u nizu. (Vreme izrade: 3 minuta)
5. Napisati program koji će ispisati sve brojeve deljive sa brojem 5 i brojem 7 do broja 100. (Vreme izrade: 4 minuta)
6. Napisati program koji će ispisati koliko brojeva deljivih sa 7 postoji u rasponu od broja 100 do broja 500. (Vreme izrade: 4 minuta)
7. Napisati funkciju koja će primiti jedan broj, a vratiti true ili false ako je broj prost ili ne, respektivno. (Vreme izrade: 5 minuta)
8. Napisati program koji će ispisati sve brojeve deljive sa 7 u nekom nizu. (Vreme izrade: 5 minuta)

✓ Poglavlje 9

Domaći zadatak

ZADATAK 1

Odgovoriti na pitanja:

Očekivano vreme izrade zadatka br. 1 i br. 2: 60min.

Odgovoriti na pitanja, gde broj indeksa predstavlja broj vašeg indeksa. Obrazložite korake svog rada i pošaljite ga u dokumentu. Zadatak dostaviti kao IT101-DZ08-Ime_Prezime_BrojIndeksa.doc.

Vreme izrade po zadatku u zadatku 1 je 10 minuta, ukupno 30 minuta.

1. Koliko iteracija će izvršiti sledeće petlje? Koja je vrednost brojača nakon obavljenе poslednje iteracije?

```
for (var i = 1; i < brojIndeksa; i++) {  
} while(true) {  
}  
  
for(var i = brojIndeksa; i >= 1;i--) {  
}
```

2. Koja je vrednost brojača i posle treće iteracije petlje? Koja je vrednost brojača nakon obavljenе poslednje iteracije?

```
for(i = 1; i < brojIndeksa; i += 20) {  
}
```

3. Koliko puta će se izvršiti sledeća petlja:

```
int i=10; do{console.log(i); i--;}while(i>1);
```

ZADATAK 2

Drugi zadatak 8. domaćeg zadatka

Vreme izrade po zadatku u zadatku 2 je 6 minuta, ukupno 30 minuta

1. Napisati program koji će ispisati sve elemente niza manje od broja 70

var niz = [10, 64, 11, 12, 99, 9, 3, 7, 1, 61, 89, 76, 6, 94];

2. Napisati program koji će naći sve brojeve deljive sa 3 i nisu deljivi sa brojem 10 do broja 1000.

3. Napisati program koji će naći sve proste brojeve do broja 350.

Broj je prost ako je deljiv **samo** sa 1 i sa samim sobom. Na primer, broj 7 je prost broj.

4. Napisati program kojim se štampa prvih 10 Fibonačijevih brojeva. Fiboničijev broj je jednak sumi prethodna dva Fibonačijeva broja. Fibonačijev niz počinje ovako: 0, 1, 1, 2, 3, 5...

5. Napraviti niz od 15 članova. Napisati program koji štampa 3 najveća elementa.

Zadatak dostaviti kao IT101-DZ08-Ime_Prezime_BrojIndeksa.txt

▼ Zaključak

ZAKLJUČAK

Na ovom predavanju smo obradili jezike nižeg i višeg nivoa. Pokazali smo karakteristike i razlike između programskih paradigm proceduralnog, objektno orijentisanog i event programiranja. Fokus ovog predavanja je stavljen na programske strukture i tipove podataka.

Literatura

1. Gary B. Shelly, Misty E. Vermaat, Discovering Computers 2011-Introductory: Living in a Digital World, Cengage Learning 2010.



IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

RAZVOJ VEB SAJTOVA

Lekcija 09

PRIRUČNIK ZA STUDENTE

IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

Lekcija 09

RAZVOJ VEB SAJTOVA

- ▼ RAZVOJ VEB SAJTOVA
- ▼ Poglavlje 1: Internet i WWW
- ▼ Poglavlje 2: Veb standardi
- ▼ Poglavlje 3: HTTP
- ▼ Poglavlje 4: Karakteristike veb čitača
- ▼ Poglavlje 5: Markap jezici
- ▼ Poglavlje 6: Pokazna vežba: HTML tabele i forme
- ▼ Poglavlje 7: Pokazna vežba: CSS - Cascading Style Sheet
- ▼ Poglavlje 8: Zadatak za samostalni rad: HTML i CSS
- ▼ Poglavlje 9: DOMAĆI ZADATAK
- ▼ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

✓ Uvod

UVOD

Cilj ovog predavanje je da objasni način funkcionisanja HTTP protokola, da se prikažu osnovni elementi HTML jezika i da se upoznate sa osnovama XML-a

Na ovom predavanju biće obrađene sledeće teme:

- Struktura i način funkcionisanja HTTP protokola
- Osnove markap jezika
- Struktura HTML-a
- Osnove XML-a

Cilj ovog predavanje je da objasni način funkcionisanja HTTP protokola, da se prikažu osnovni elementi HTML jezika i da se upoznate sa osnovama XML-a.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 1

Internet i WWW

INTERNET SERVISI

Razvoj Interneta omogućio je razvoj novih tehnologija i upotrebu mnogih novih servisa koji koriste Internet kao infrastrukturu

Internet, ili popularnije nazvan "net", između ostalog, pruža veliki izvor informacija, pruža olakšano poslovanje i unapređuje komunikaciju. Internet ustvari predstavlja milione mreža koje povezuju preduzeća, vladine organizacije, obrazovne institucije, pojedince i sl.

Razvoj Interneta omogućio je razvoj novih tehnologija i upotrebu mnogih novih servisa koji koriste Internet kao infrastrukturu. Najpopularniji Internet servisi su:

- E-mail
- Telnet
- Forum
- FTP
- Internet Chat
- World Wide Web
- Veb portali
- Veb Servisi

World Wide Web i email su dva servisa koja se najčešće koriste na Internetu. Iako se World Wide Web (WWW), ili jednostavno "veb", često meša sa Ineternetom, veb ustvari predstavlja samo jedan od servisa koji radi nad infrastrukturom Interneta.

WWW je Internet servis koji predstavlja skup elektronskih dokumenata. Ta dokumenta nazivaom veb stranicama. Veb strane obično sadrže sadržaj u formi teksta, zvuka, videa, grafike i animacije.

Za korišćenje veb servisa korisnicima služe **vеб читачи** (engl. **web browser**). Veb čitač je aplikacioni softver koji koristi korisnik da pristupi vebu. Veb čitač koriste i neki Internet servisi kao što su email ili FTP. Veb čitači obično imaju grafički korisnički interfejs koji je specijalno dizajniran kako bi se što efikasnije prikazale veb stranice. Veb čitači omogućavaju korisnicima da pregledaju hipertekstualne stranice. Najčešće korišćeni veb čitači su Internet Explorer, Firefox, Opera, Chrome, Safari.

Hipertekst je pojam kojim se opisuje dokument u kome se pojedini pojmovi iz tog dokumenta referenciraju na druge dokumente ili datoteke sa drugim sadržajima, na primer sa zvučnim ili video zapisima. Za referenciranje se koriste **hiperlinkovi** (engl. **hyperlink**), koji sadrže putokaz do resursa na koji izabrani pojmom iz hiperteksta referencira. Traženi resursi mogu biti na istom računaru ili na nekim udaljenim računarima koji su vezani na Internet. Za smeštaj i pristup

hipertekst stranica koriste se **veb serveri**. Oni imaju zadatak da veb čitačima pripreme i pošalju zahtevani sadržaj. Kolekcija hipertekst stranica na jednom veb serveru se naziva veb sajt (engl. **Web site**).

▼ Poglavlje 2

Veb standardi

ŠTA SU VEB STANDARDI?

Veb standardi su pravila i smernice koje je uspostavio World Wide Web Consortium

Veb standardi su pravila i smernice koje je uspostavio **World Wide Web Consortium** (W3C) razvijen da promoviše konzistentnost u kodu koji čini veb stranicu.

Jednostavno rečeno, veb standardi predstavljaju smernice za mark-up jezike koji čine jednu veb stranicu.

Prednosti u pridržavanju standarda su mnoge:

- Veb stranice će se prikazivati u širokom spektru čitača i računara, uključujući nove tehnologije
- W3C standardi promovišu upotrebu "**Cascading Style Sheets**" (CSS) umesto da je ugrađuju u samu stranicu. Korišćenje CSS značajno smanjuje veličinu datoteke, što znači ne samo brže vreme učitavanja stranice, već i niže troškove hostinga za često posećene lokacije zbog korišćenja manjeg propusnog opsega.
- Mogućnosti dizajna, kao što su boje i fontovi, lako se menjaju samo izmenom jednog stila u CSS datoteci umesto da se ažurira svaka stranica pojedinačno, smanjujući troškove za modifikovanje sajta.
- Pretraživači mogu pristupiti i indeksirati stranice dizajnirane za veb standarde sa većom efikasnošću.

OSNOVNI WWW STANDARDI

*Osnovu World Wide Web-a čini nekoliko standarda, među kojima su najvažniji **URI**, **HTTP**, **HTML***

Osnovu World Wide Web-a čini nekoliko standarda, među kojima su najvažniji:

- **URI** - Uniform Resource Identifier- jedinstveni identifikator resursa koji je univerzalni sistem za referenciranje resursa na vebu, kao što su na primer veb strane
- **HTTP** - HyperText Transfer Protocol- protokol za transfer hiperteksta koji specificira kako komuniciraju veb čitač i veb server međusobno
- **HTML** - HyperText Markup Language- hipertekst markup jezik koji se koristi da se definije struktura i sadržaj hipertekstualnog dokumenta

✓ 2.1 URI, URL i URN

RAZLIKA IZMEĐU URI, URL I URN

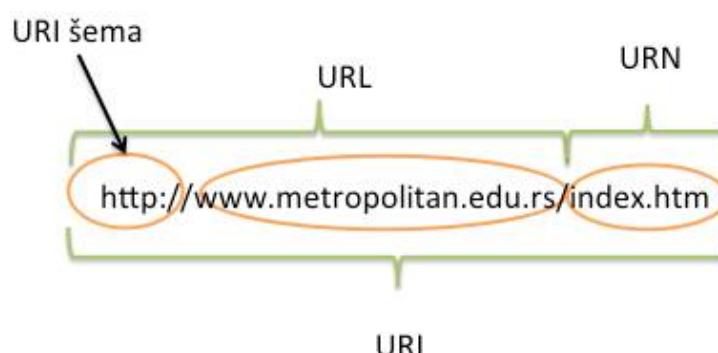
URI se sastoji iz URL i URN-a

URI (Uniform Resource Identifier) predstavlja jedinstveni identifikator resursa. URI je element Internet protokola koji se sastoji od stringa, odnosno od niza karaktera koji odgovaraju dogovorenoj sintaksi. String sadrži ime ili adresu koja se može upotrebiti da se referencira resurs. URI ima funkciju lokatora i imena. Tipičan URI se sastoji od tri elementa:

- naziva šeme mehanizma koji se koristi da se pristupi resursu
- ime domena gde se resurs nalazi
- ime samog resursa dato kao putokaz na hostu.

URI se može klasifikovati kao lokator **URL** (Uniform Resource Locator), kao **URN** (Uniform Resource Name) ili kao oba. URL sadrži informacije o tome kako da se pristupi resursu. U praksi, termin URL zamenjuje termin "veb adresa", iako to nije u potpunosti tačno s obzirom da URL adresa sadrži ime domena i URI šemu. URL predstavlja podskup URI-a. Kada se gleda čitav URI sintaksa, URI počinje sa URI šemom kao što su http, ftp, mailto i dr., zatim sledi dvotačka ":", a zatim sledi URL, pa URN.

URN je podskup URI i on identificuje resurs po imenu u definisanom domenskom prostoru. URN omogućava pozivanje resursa bez razmišljanja o njegovoj lokaciji. Na primer, neke knjige imaju svoj URN (na primer urn: ISBN : 0 - 395 - 36341 - 1) koji omogućava da se neka knjiga definiše po tom broju, istovremeno ne uzimajući u obzir gde se ona tačno nalazi. Primeri jedinstvenih identifikatora dati su na Slici 1.



Slika 2.1.1 Primer odnosa URI prema URL i URN

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

VIDEO

URIs, URLs, and URNs | Difference between URI and URL | URL Explained

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

✓ Poglavlje 3

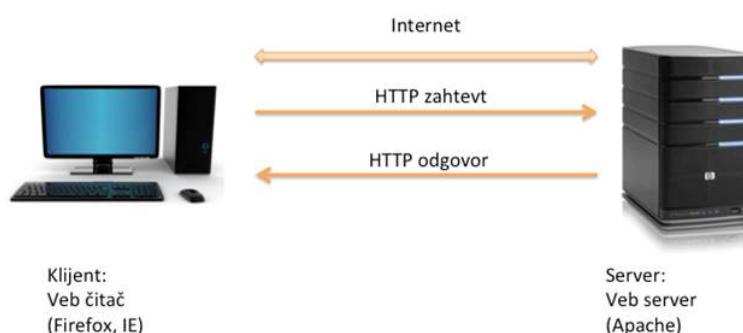
HTTP

KARAKTERISTIKE HTTP

Komunikacija između veb klijenta i veb servera je definisan HTTP-om

Da bi se veb stranice i sadržaj koji oni sadrže i šalju (slike, rezultati upita i sl.) mogli isporučiti korisniku kada korisnik to zahteva, neophodno je da se prate određena pravila. **Hypertext Transfer Protocol** (HTTP) je protokol koji predstavlja skup pravila koja se koriste za prenos stranica na Internetu. Drugim rečima, komunikacija između veb klijenta i veb servera je definisan HTTP-om. U ovom slučaju, program koji šalje zahtev serveru (veb čitač) predstavlja klijent. Na osnovu zahteva server formuliše odgovor i šalje ga klijentu.

HTTP koristi klijent/server model koji omogućava klijentu da uspostavi vezu i da pošalje zahtev. Server šalje odgovor, obično zajedno sa traženim sadržajem. Kada se odgovor isporuči, server raskida uspostavljenu vezu.



Slika 3.1 Klijent/server HTTP komunikacioni model

HTTP ima nekoliko važnih karakteristika:

- HTTP/1.0 podrazumeva da kada se zahtev ispunji, da klijent raskida vezu sa serverom. Vezu je potrebno uspostaviti za svaki novi zahtev/odgovor. Server nema potrebu da održava otvorenu sesiju kako bi slao zahtevani sadržaj.
- HTTP/1.1 koristi perzistentne veze što podrazumeva da se koristi ista konekcija za slanje više zahteva i odgovora, a da pri tom nema potrebu da otvara novu konekciju za svaki zahtev i odgovor.
- HTTP je nezavistan od medija i može da prenosi bilo koji tip podataka dokle god i klijent i server mogu da ih obrade.
- HTTP ne zahteva ni od klijenta ni od servera da održava podatke o konekciji, tako da ni klijent ni server ne čuvaju informacije jedan o drugom kada se zahtev ispunji.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

HTTP ZAHTEV

Najčešće korišćena HTTP metoda koja se koristi u zahtevu je metoda GET. HTTP metode su GET, POST, PUT, HEAD, DELETE, TRACE, CONNECT i OPTIONS

Prvi red u HTTP zahtevu (engl. **request**) se razlikuje od prvog reda HTTP odgovora. Prvi red sadrži ime HTTP metode, HTTP verziju i putanju do traženog resursa. Najčešće korišćena HTTP metoda koja se koristi u zahtevu je metoda **GET**. Druge HTTP metode su **POST, PUT, HEAD, DELETE, TRACE, CONNECT i OPTIONS**. Verzija HTTP se uvek navodi u obliku “/HTTP/x.x” gde “x.x” predstavlja broj verzije.

Niže navedeni primer predstavlja primer GET metode i to zahtev za pregled resursa na stranici www.metropolitan.ac.rs . Može se primetiti da se vidljiva verzija HTTP, koja je 1.1. Zatim u zahtevu je navedeno ime hosta i “user-agent” koji obično predstavlja ime veb čitača koji šalje zahtev kao i njegovu verziju. “Accept” predstavlja metodu koja navodi šta može veb čitač da prihvati od jezika, metode šifrovanja i tip resursa.

GET / HTTP/1.1

Host: www.metropolitan.ac.rs

User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:8.0) Gecko/20100101 Firefox/8.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

Accept-Language: en-us,en;q=0.5

Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7

Connection: close

HTTP ODGOVOR

Početni red odgovora sadrži HTTP verziju, statusni deo koda koji opisuje rezultat zahteva i reč na engleskom jeziku koja opisuje taj rezultat

Početni red odgovora (engl. **response**) sadrži HTTP verziju, statusni deo koda koji opisuje rezultat zahteva i reč na engleskom jeziku koja opisuje taj rezultat. U primeru ta engleska reč je “OK”. Primer prvog reda HTTP odgovora je:

HTTP/1.1 200 OK ili

HTTP/1.1 404 Not Found

U prethodnom primeru, zahtev je prikazan za hosta www.metropolitan.ac.rs. Odgovor na taj zahtev izgleda ovako:

HTTP /1.1 200 OK

Date: Tue,26 June 2020 11:37:00 GMT

Server: Apache-Coyote/1.1

Content-Type: text/html; charset=UTF-8

Connection: close

Transfer-Encoding: chunked

(Content)

Ukoliko je klijent tražio stranicu ili resurs koji ne postoji, na primer sa sledećih zahtevom:

GET /it101.html HTTP/1.1

Host: www.metropolitan.ac.rs

Tada će server odgovoriti sa sledećim odgovorom

HTTP /1.1 404 Not Found

Date: Tue,26 June 2020 11:52:00 GMT

Server: Apache-Coyote/1.1

Content-Type: text/html

Connection: close

Transfer-Encoding: chunked

HTTP METODE

Najčešće korišćene HTTP metode su GET i POST

HTTP metode su:

- GET
- POST
- PUT
- HEAD
- DELETE
- PATCH
- TRACE
- TUNNEL
- OPTIONS

HTTP **GET** metoda je jedna od najčešće korišćenih metoda i ona se koristi da pošalje zahtev kako bi pristupila određenom resursu. Upit se šalje u okviru URL linka

/test/demo_form.php?name1=value1&name2=value2

HTTP **POST** metoda šalje podatke serveru kako bi kreirao ili ažurirao neki resurs. Podaci koji se šalju ovom metodom se nalaze u telu HTTP zahteva

POST /test/demo_form.php **HTTP/1.1 Host: w3schools.com name1=value1&name2=value2**

HTTP **PUT** metoda se koristi da kreira ili ažurira neki resurs. Razlika između PUT i POST metoda je ta što PUT pravi novu verziju podatka, dok se stara briše, dok POST samo pridružuje već postojećem resursu podatke.

HTTP **HEAD** metoda je skoro ista kao i GET, samo što se u odgovoru na HEAD zahtev ne dobija telo poruke, nego samo zaglavlj. HEAD zahtevi su korisni da provere šta je to što će se dobiti GET zahtevom, što može biti korisno kada se proverava veličina fajla koja treba da se preuzme.

HTTP **DELETE** metoda briše određeni resurs.

HTTP **PATCH** metoda se koristi da napravi delimična ažuriranja na nekom resursu.

HTTP **TRACE** metoda se koristi za vraćanje eha poslatog zahteva odredišnom serveru. Drugim rečima, klijent može da vidi kakvu je poruku primio odredišni server. Na taj način klijent može da sazna koji su međuserveri menjali originalni zahtev, što je vrlo korisno za dijagnostiku.

HTTP **OPTIONS** metoda opisuje moguće komunikacione metode sa traženim resursom.

VIDEO

Http Methods

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

✓ Poglavlje 4

Karakteristike veb čitača

PREGLED OSOBINA VEB ČITAČA

Veb čitač je u stvari korisnički interfejs prema vebu

Da bi korisnik mogao da pristupi veb resursima kao što su veb stranice ili veb servisi potrebno je da ima veb čitač (engl. **web browser**). Veb čitač je u stvari korisnički interfejs prema vebu, mada gotovo svi današnji veb čitači nude integrisane interfejse i za druge Internet servise, na primer e-mail ili ftp.

Veb čitači su uglavnom grafički korisnički interfejsi specijalno projektovani za prikaz veb strana i interpretaciju objekata koje veb strana sadrži. Unutar veb strane se mogu umetnuti različiti objekti kao što su: digitalne slike, vektorske slike, digitalni video zapisi, digitalni audio zapisi. Veb čitač treba da je u stanju da interpretira sve ove objekte.

Važne osobine veb čitača su:

- dostupnost (engl. **accessibility**).
- ergonomičnost
- koje veb tehnologije podržava
- koje protokole podržava pored HTTP-a
- koje formate slika može da prikaže
- koje jezike podržava
- kakav sistem zaštite od napada koristi.

U tabeli 1 dat je pregled osobina veb čitača koje se odnose na dostupnost.

Web browser	Tabbed browsing	Pop-up blocking	Incremental Search	Ad filtering	Page Zooming	Full Text Search of history	Content-modal dialogs
Google Chrome	Da	Delimično	Da	Ne	Da	Ne	Da
Internet Explorer	Da	Da	Da	Da	Da	Da	Ne
Internet Explorer (MAC)	Ne	Ne	Ne	Ne	Ne	Ne	Ne
Opera	Da	Da	Da	Da	Da	Da	Da
Safari	Da	Da	Da	Ne	Da	Da	Da
Galeon	Da	Da	Da	Da	Da	Ne	Ne
Mozilla Firefox	Da	Da	Da	Da	Da	Ne	Da

Slika 4.1.1 Tabela-1 Pregled osobina veb čitača (2020.god.)

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

WEB TEHNOLOGIJE KOJE PODRŽAVAJU VEB ČITAČI

Prikaz veb čitača i tehnologija koje podržavaju.

U tabeli 2 prikazane su veb tehnologije koje podržava veliki broj čitača.

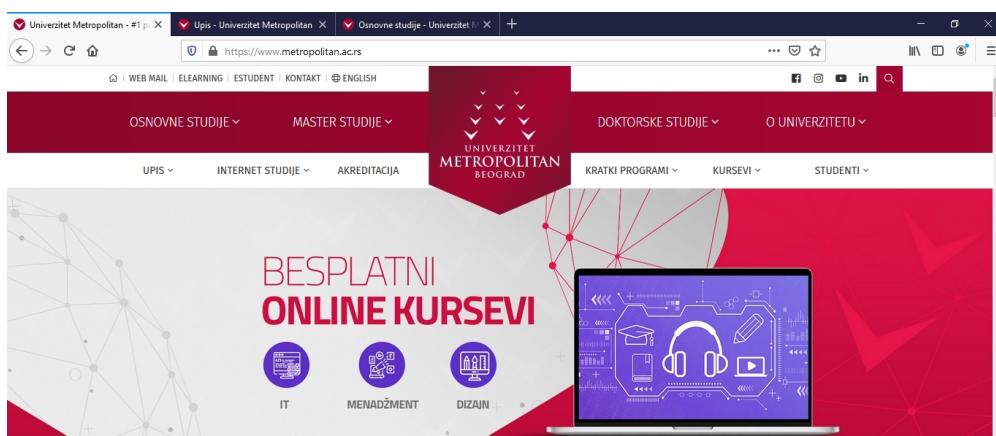
Web browser	CSS2.1	Frames	XSLT	XHTML 1.0	XHTML 1.1	MathML	XForms	WebForms	VoiceXML
Google Chrome	Da	Da	Da	Da	Da	Ne	Ne	Ne	Ne
Internet Explorer	Da	Da	Da	Da	Da	Ne	Ne	Ne	Ne
Internet Explorer (MAC)	Delimično	Da	Da	Da	Da	Ne	Ne	Ne	Ne
Opera	Da	Da	Da	Da	Da	Da	Ne	Da	Da
Safari	Da	Da	Da	Da	Da	Ne	Ne	Ne	Ne
Galeon	Da	Da	Da	Da	Da	Da	Ne	Ne	Ne
Mozilla Firefox	Da	Da	Da	Da	Da	Da	Da	Da	Ne

Slika 4.1.2 Tabela-2 Veb tehnologije koje podržavaju veb čitači (2020.god.)

TABBED BROWSING

“Tabbed browsing” omogućava da se više dokumenata otvori u jednom prozoru

“Tabbed browsing” omogućava da se više dokumenata otvori u jednom prozoru. Primer tabova je dat na slici 3 gde se mogu videti tri otvorena taba u čitaču Mozilla Firefox.



Slika 4.1.3 Korišćenje “tabbed browsing” u Mozilla Firefox-u

POP-UP

“Pop-up blocking” omogućava blokiranje raznih prozora koji se najčešće pojavljuju kao reklama

“Pop-up blocking” omogućava blokiranje raznih prozora koji se najčešće pojavljuju kao reklama za neke proizvode ili druge sajtove, a odvraća nas od stranice na kojoj smo ili koju želimo da posetimo. “Pop-up” prozor se najčešće generiše kroz JavaScript. Varijacija na “pop-

up” je “pop-under” koji se pojavljuje iza aktivnog prozora umesto ispred njega. Oni se obično ne uočavaju odmah, tako da je teško ustanoviti koji ih je sajt otvorio, što kod pop-up prozora možemo odmah utvrditi. Na slici 4 je dat primer “pop-up” prozora.

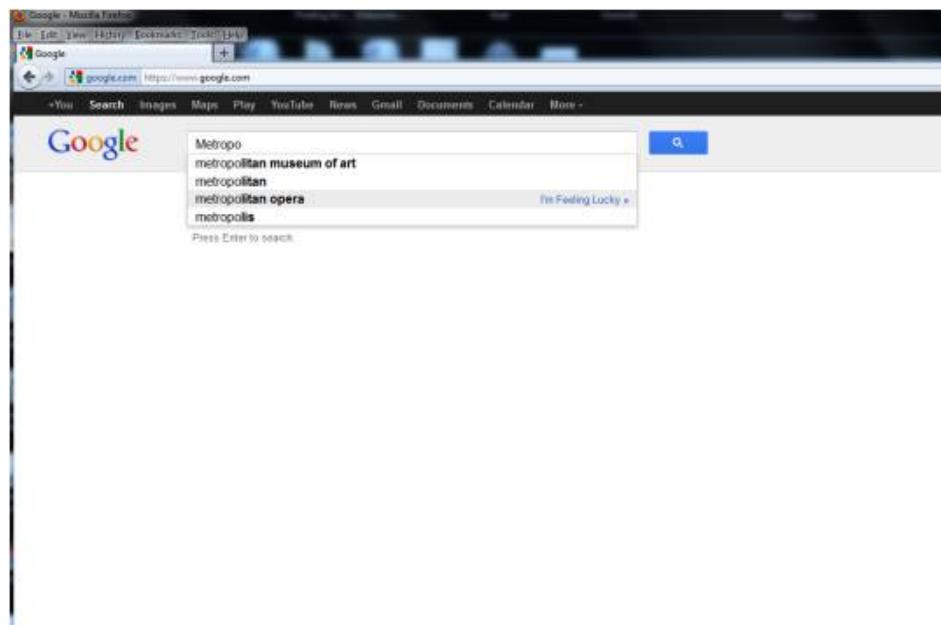


Slika 4.1.4 Pop-up prozori

INKREMENTALNE PRETRAGE

Inkrementalna pretraga predstavlja interaktivni metod koji u realnom vremenu daje predlog šta pokušavate da ukucate za pretraživanje

U računarstvu, inkrementalna pretraga (engl. *incremental finding*) predstavlja interaktivni metod koji u realnom vremenu daje predlog šta pokušavate da ukucate za pretraživanje. Pronalazi se nekoliko mogućnosti i odmah se prezentuju korisniku, kako korisnik ne mora do kraja da otkuca tu reč ili frazu i u ponuđenoj listi može da izabere šta traži (Slika 5).



Slika 4.1.5 Inkrementalna pretraga

AD FILTERING, ZUMIRANJE, POTPUNA PRETRAGA ISTORIJE

Blokiranje reklama je uklanjanje reklama ili menjanje njihovog sadržaja

Ad filtering ili blokiranje reklama je uklanjanje reklama ili menjanje njihovog sadržaja. Reklame se mogu pojavljivati kao slike, tekstovi, animacije ili pop-up prozori. Napredni filteri omogućavaju kontrolu tih reklama. Mozilla Firefox koristi Adblock Plus, dok Internet Explorer koristi Simple AdBlock, IE7 Pro, Adblock Pro i Quero. Mogu se koristiti i eksterna rešenja poput podešavanja proksija.

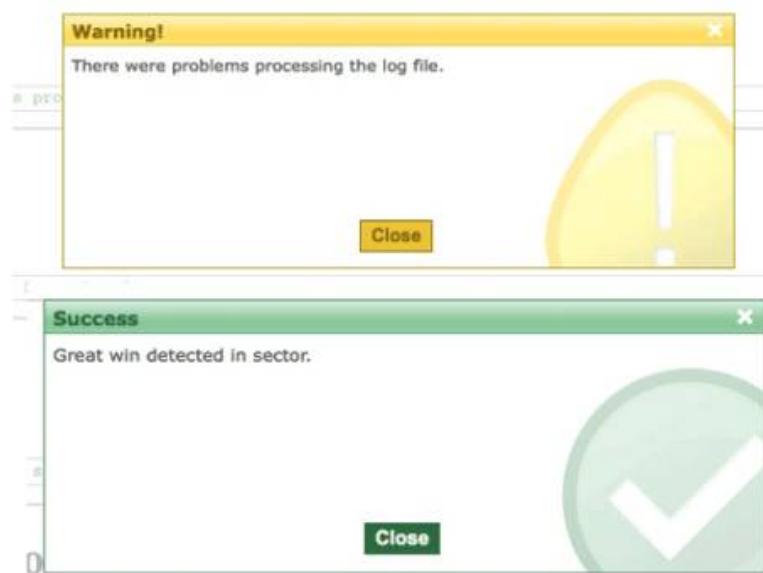
Zumiranje stranice (engl. *page zooming*) se koristi da se poveća ili smanji stranica.

“Full-text search of history”, odnosno pretraživanje komplettnog teksta se odnosi na pretraživanje kompletnih reči u posećenim sajtovima na jednom računaru. Veoma je sličan opciji inkrementalne pretrage, ali se ovde odnosi samo na URL koji ukucavamo za sajt koji želimo da pretražimo. Ukoliko smo taj sajt već posetili, njegov URL će se pojaviti kao jedan od opcija u padajućem meniju.

CONTENT MODAL DIALOGS

Content modal dialogs se odnosi na prozore koji se pojavljuju kada žele da nam skrenu pažnju na neku operaciju

“Content modal dialogs” se odnosi na prozore koji se pojavljuju kada žele da nam skrenu pažnju na neku operaciju koju smo izvršili ili treba da izvršimo. Primer modalnog dijaloga se nalazi na slici 6.



Slika 4.1.6 Modalni dijalog

❖ 4.1 Protokoli koje podržavaju veb čitači

PREGLED PROTOKOLA KOJE PODRŽAVAJU VEB ČITAČI

Veb čitači pored prikaza strana kao osnovne funkcije, podržavaju i druge mogućnosti

Neki veb čitači osim prikaza strana imaju i druge funkcije, pa pored HTTP protokola podržavaju i druge protokole. U tabeli 1 dat je prikaz protokola koje podržavaju najpopularniji veb čitači.

Web browser	HTTP	E-mail	FTP	SSL	Gopher	Data:URI
Google Chrome	Da	Ne	Da	Da	Ne	Da
Internet Explorer	Da	Ne	Da	Da	Ne	Delimično
Internet Explorer (MAC)	Ne	Ne	Da	Da	Da	Ne
Opera	Da	Da	Da	Da	Ne	Da
Safari	Da	Ne	Delimično	Da	Ne	Da
Galeon	Ne	Ne	Da	Da	Da	Da
Mozilla Firefox	Da	Ne	Da	Da	Ne	Da

Slika 4.2.1 Tabela-1 Prikaz protokola koje podržavaju veb čitači (2020.god.)

Kada otvorimo veb čitač i želimo da pogledamo neku veb stranu, tada naš veb čitač ima ulogu klijenta koji šalje zahtev serveru na kome se ta veb stranica nalazi. Klijenti su ustvari ti koji šalju zahteve. Oni mogu veb serverima da pošalju različite zahteve. Ti zahtevi mogu biti

zahet za pregledanje veb strane ili slanja obrasca sa podacima. Kada veb server primi zahtev, on isti obrađuje i šalje svoj povratni odgovor. Ovaj odgovor klijent dobija u posebnom formatu, kako bi iste mogao da prikaže. Standardni jezici kojim klijent i server komuniciraju na vebu su protokol za prenos hiperteksta (engl. **HyperText Transfer Protocol** (HTTP)). HTTP-ov rad se oslanja na TCP i IP.

▼ 4.2 Zastupljenost veb čitača na tržištu

STATISTIČKI PRIKAZ PROCENTUALNE ZASTUPLJENOSTI OD 2002. DO 2010. GODINE

Tokom 2002. godine najčešće korišćeni veb čitači su bili Internet Explorer, AOL i Netscape

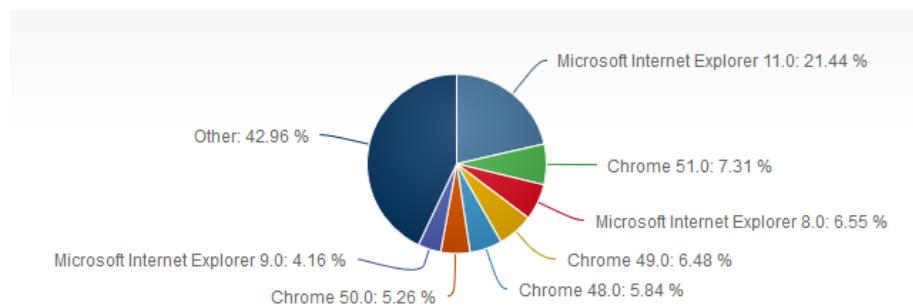
Tokom 2002. godine najčešće korišćeni veb čitači su bili Internet Explorer, AOL i Netscape. Tokom godina struktura aktuelnih veb čitača se menjala. Tokom 2010. godine Internet Explorer je izgubio primat i dominaciju je preuzeo Firefox sa 44,1% korisnika. U tabeli 1 je prikazan statistički prikaz njihove procentualne zastupljenosti od 2002. do 2010. godine.

2002	Internet Explorer	AOL		Netscape	
November	83.4 %	5.2%		8.0%	
2003	Internet Explorer		Mozilla	Netscape	Opera
November	84.9 %		7.2%	2.6%	1.9%
2004	Internet Explorer		Mozilla	Netscape	Opera
November	76.2 %		16.5%	1.7%	1.6%
2005	Internet Explorer	Firefox	Mozilla	Netscape	Opera
November	68.9 %	23.6%	2.8%	0.4%	1.5%
2006	Internet Explorer	Firefox	Mozilla	Netscape	Opera
November	60.6 %	29.9%	2.5%	0.2%	1.5%
2007	Internet Explorer	Firefox	Mozilla	Safari	Opera
November	56.0 %	36.3%	1.2%	1.8%	1.6%
2008	Internet Explorer	Firefox	Chrome	Safari	Opera
November	47.0 %	44.2%	3.1%	2.7%	2.3%
2009	Internet Explorer	Firefox	Chrome	Safari	Opera
November	37.7 %	47.0%	8.5%	3.8%	2.3%
2010	Internet Explorer	Firefox	Chrome	Safari	Opera
October	29.7 %	44.1%	19.2%	3.9%	2.2%

Slika 4.3.1 Tabela-1 Statistički prikaz procentualne zastupljenosti od 2002. do 2010. godine

ZASTUPLJENOST VEB ČITAČA 2016

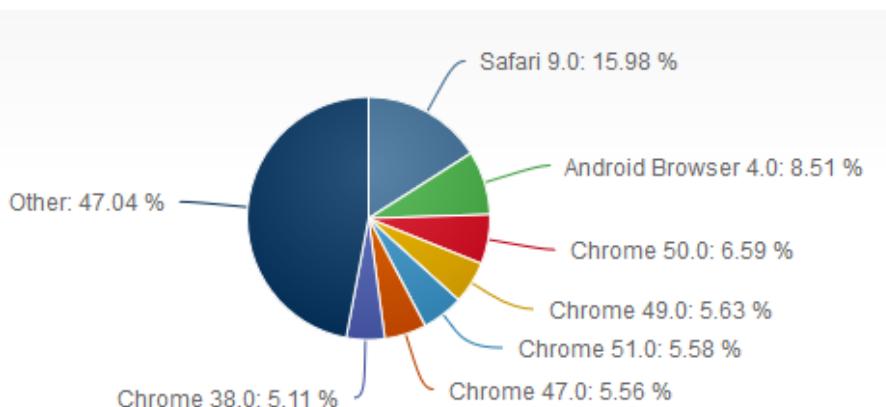
Zastupljenost veb čitača u 2016. godini



Slika 4.3.2 Zastupljenost veb čitača na desktop računarima u 2016. godini

Na slici 2 možemo videti zastupljenost veb čitača u 2016. godini. Kao što je promena korišćenja pokazala i između 2002. godine i 2010., tako se primećuje promena tržišta i u 2016. godini. Internet Explorer je daleko od dvoje nekadašnje popularnosti, iako i dalje dominira, ali samo sa 21,44%.

S druge strane, interesantno je pogledati stepen korišćenja veb čitača na mobilnim uređajima u 2016. godini na slici 3. Interesantno je sigurno uporediti ovu statistiku sa zastupljenošću različitih proizvođača telefona i tableta.

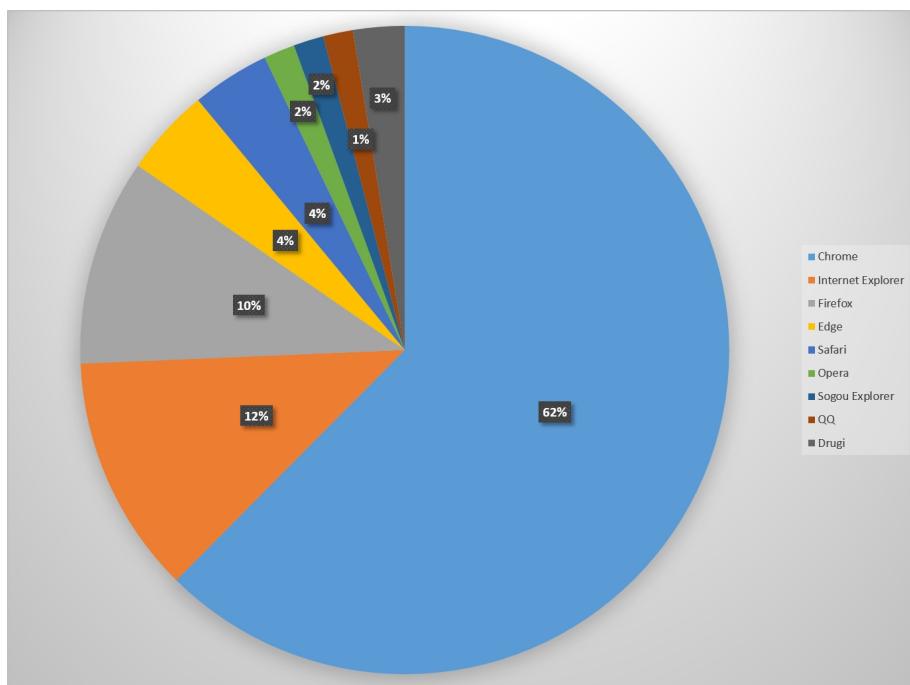


Slika 4.3.3 Zastupljenost veb čitača na mobilnim uređajima i tabletima u 2016. godini

ZASTUPLJENOST VEB ČITAČA 2018

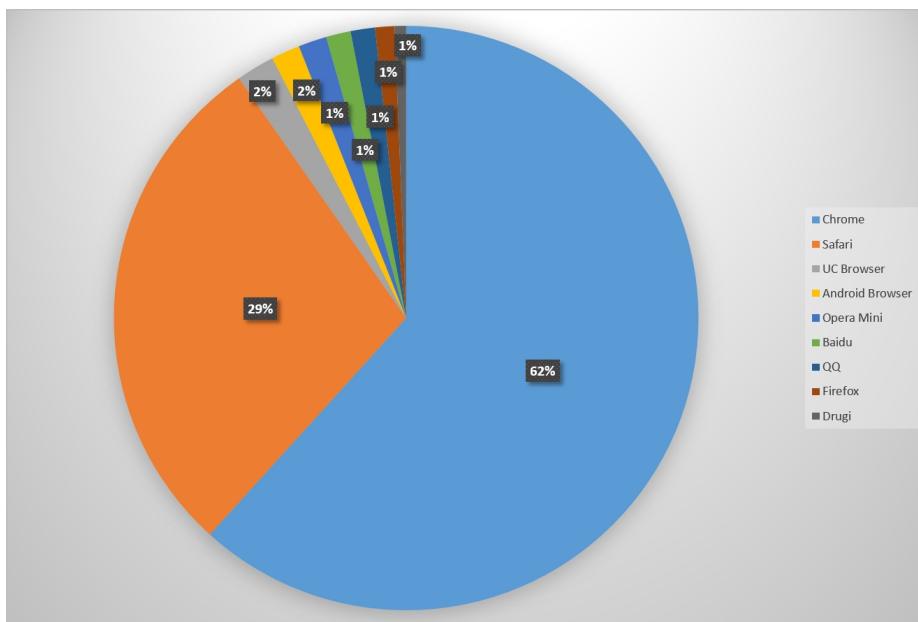
Zastupljenost veb čitača u 2018. godini

Zastupljenost veb čitača na desktop i laptop računarima u 2018. godini je prikazan na slici 4.



Slika 4.3.4 Zastupljenost veb čitača u 2018. godini koji su korišćeni na desktop i laptop računarima

Na slici 5 je prikazana zastupljenost veb čitača na mobilnim uređajima i tabletima u 2018. godini.



Slika 4.3.5 Zastupljenost veb čitača na mobilnim uređajima i tabletima u 2018. godini

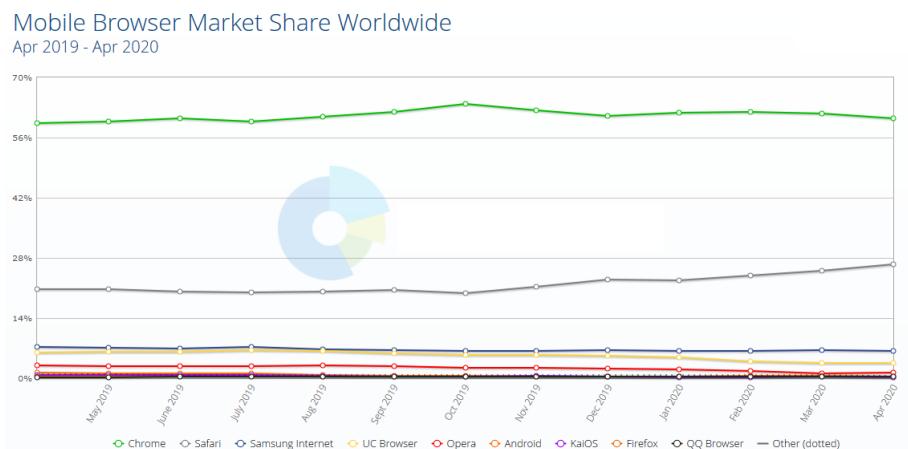
ZASTUPLJENOST VEB ČITAČA 2020

Zastupljenost veb čitača u 2020. godini

Zastupljenost veb čitača na desktop računarima i mobilnim uređajima u 2020. godini je prikazan na slici 6, dok je na slici 7 prikazana njihova zastupljenost na mobilnim uređajima.



Slika 4.3.6 Zastupljenost veb čitača u 2020. godini koji su korišćeni na desktop računarima



Slika 4.3.7 Zastupljenost veb čitača na mobilnim uređajima u 2020. godini

▼ Poglavlje 5

Markap jezici

WEB MARKUP JEZICI

Gotovo svi markup jezici su zasnovani na internacionalnom standardu ISO 8879

Web markup jezici (engl. **markup language**) su posebni jezici kojima se opisuje veb dokument. Unutar samog teksta dokumenta se upisuju posebne sekvene karaktera poznate kao tagovi ili oznake kojim se obeležavaju pojedini elementi ili grupe elemenata radi prikazivanja ili identifikacije. Zato se ovi jezici mogu nazvati i jezici obeležavanja.

Gotovo svi markup jezici su zasnovani na internacionalnom standardu ISO 8879 – **Standard Generalized Markup Language (SGML)**. Ovaj standard je nastao 1986, dakle pre pojave veba, sa idejom da se kreira jezik kojim će se definisati format u tekstualnim dokumentima. Međutim, SGML se koristi i za definisanje drugih specijalizovanih jezika za predstavljanje dokumenata. SGML dokument koristi posebnu datoteku koja se zove **Document Type Definition (DTD)**, koja definiše tagove koji se koriste za opis formatiranja teksta, atribute i entitete nekog markup jezika, kao i način na koji se oni zajedno koriste.

Pošto SGML opisuje svoje sopstveno formatiranje naziva se meta-jezik. SGML je vrlo složen i obiman jezik koji između ostalog uključuje i hipertekst linkove. Značajni markup jezici koji su nastali na osnovu SGML-a su HTML, XHTML i XML.

SGML DTD je u početku definisao HTML na taj način da se HTML mogao koristiti samo za jednostavnu isporuku veb sadržaja. S jedne strane ovo je bila prednost, s obzirom da je na taj način HTML bio jednostavniji za korišćenje od SGML, ali s druge strane HTML je iz istog razloga ograničen sa svojom efikasnošću i fleksibilnosti. Kada se koristi DTD to podrazumeva da su svi elementi potpuno prilagodljivi i omogućava da pravila korišćenja budu definisana u samom dokumentu.

▼ 5.1 XML

EXTENSIBLE MARKUP LANGUAGE (XML)

Za razliku od HTML-a koji toleriše fleksibilan način kodiranja, XML stranice moraju da odgovore strogim pravilima

Veliki nedostatak HTML je što nema mogućnosti da radi sa složenim strukturnim podacima ili dokumentima zato što ima fiksirani set unapred definisanih tagova i nema mehanizme da definiše nove elemente. Ova osobina je neophodna za aplikacije kao što je automatska razmena podataka. Rešenje je nađeno u novom jeziku koji se zove XML.

XML je skraćenica od **EXtensible Markup Language** i spada u klasu meta jezika. XML je napravljen kako bi čuvao i prenosi podatke. Napisan je tako da bi bio čitljiv i za čoveka, a i računaru. Koristi se za definisanje podataka koji se publikuju na veb stranama i u **business-to-business** dokumentima. XML koristi sličnu strukturu kao HTML, ali dok HTML definiše kako će se elementi prikazati, XML definiše šta elementi sadrže. Dok HTML koristi unapred definisane standardne tagove, XML dozvoljava da autor strane definiše tagove. Tako je moguće identifikovati bilo koju vrstu podataka, na primer naziv proizvoda, cenu, iznos za plaćanje i slično. Na taj način veb strana postaje slična zapisu u bazi podataka. Ovo je od posebnog značaja za **business-to-business** veb aplikacije i za elektronsku razmenu podataka.

Za razliku od HTML-a koji toleriše fleksibilan način kodiranja, XML stranice moraju da odgovore strogim pravilima. Zato se stranice napisane u XML kontrolišu i verifikuju parserima.

Primer

```
<note>
<to> Ivan </to>
<from> Popovic </from>
<heading> Alarm </heading>
<body> Odlazak kod zubara ! </body>
</note>
```

XML ima važnu ulogu u mnogim IT sistemima. Zbog toga je važno, da svako ko se bavi razvojem softvera, dobro poznaje XML.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

RAZLIKA IZMEĐU HTML I XML

XML definiše značenje podataka, a HTML definiše način na koji će se tekst prikazati

U sledećem primeru ćemo pokazati razliku između HTML-a i XML-a.

XML

```
<ime>Vladimir</ime>
< prezime>Stepić</ prezime>
<predmet>Osnove IT</predmet>
<ocena>10</ocena>
```

HTML

```
<font size="3"> Vladimir Stepić </font>
<b> Osnove IT 10 </b>
```

Treba primetiti da XML iskaz definiše značenje podataka, a HTML definiše način na koji će se tekst prikazati.

XML i HTML su projektovani sa različitim ciljevima:

- XML je projektovan za prenos podataka - sa fokusom na to što su podaci
- HTML je projektovan tako da prikazuje podatke - sa fokusom na to kako će podaci biti prezentovani
- XML oznake nisu unapred definisane kao što su HTML oznake
- XML je proširiv

Koja je razlika između markap jezika? Pogledajte video "SGML HTML XML What's the Difference?"

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

DEFINISANJE XML PREZENTACIJE

Pošto XML dokument nema unapred definisanu prezentaciju u veb čitaču, potrebno je njeno definisanje

Pošto XML dokument nema unapred definisanu prezentaciju u veb čitaču, potrebno je njeno definisanje. Za ovo se koristi nekoliko tehnika kao što su:

- Smeštanje veb sadržaja u XML dokument, a zatim njegovo prevodenje u HTML ili XHTML format korišćenjem XSL (eXtensible Stylesheet Language) transformacije
- CSS (Cascading Style Sheets) upotrebom eXtensible Stylesheet Language Transformations (XSLT)
- neki oblik server-side programiranja
- direktnim renderovanjem XML-a u veb čitaču vezivanjem CSS-a direktno za korisnički definisane elemente.

Korišćenjem XML-a su razvijeni mnogi specijalizovani jezici, uključujući i: XHTML, RSS (Really Simple Syndication), WML (Wireless Markup Language), SVG (Scalable Vector Graphics), SOAP (Simple Object Access Protocol).

5.2 HTML i STILOVI

HTML STILOVI ZA POZADINU I TEKST

Podešavanje stila HTML elementa može se uraditi s atributom style.

Podešavanje stila HTML elementa može se uraditi s atributom **style**. Atribut style ima sledeću sintaksu: **<tagname style="property:value;">**

I property i value se odnose na osobinu i vrednost CSS.

Boja pozadine se određuje sa background-color. Sledeći primer podešava plavu pozadinu stranice:

```
<!DOCTYPE html>
<html>
<body style="background-color:powderblue;">

<h1> Ovo je naslov </h1>
<p> Ovo je paragraf. </p>

</body>
</html>
```

Boja teksta se može podesiti na sledeći način:

```
<!DOCTYPE html>
<html>
<body>

<h1 style="color:blue;"> Ovo je plavi naslov </h1>
<p style="color:red;"> Ovo je crveni paragraf. </p>

</body>
</html>
```

Stil slova, odnosno podešavanja fontova se vrši na sledeći način:

```
<!DOCTYPE html>
<html>
<body>

<h1 style="font-family:verdana;"> Naslov u Verdana fontu </h1>
<p style="font-family:courier;"> Paragraf u Courier fontu. </p>

</body>
</html>
```

Veličina i pozicioniranje teksta se može videti na sledećim primerima:

```
<!DOCTYPE html>
<html>
<body>

<h1 style="font-size:300%;"> Ovo je naslov </h1>
<p style="font-size:160%;"> Ovo je paragraf. </p>

</body>
</html>
```

```
<!DOCTYPE html>
<html>
<body>

<h1 style="text-align:center;"> Naslov koji je centriran </h1>
<p style="text-align:center;"> Tekst paragrafa koji je centriran. </p>

</body>
</html>
```

FORMATIRANJE TEKSTA

Pored atributa, HTML ima specijalne elemente koji se koriste za formatiranje teksta

Pored atributa, HTML ima specijalne elemente koji se koriste za formatiranje teksta. Na primer, HTML koristi elemente **** i *<i>* za formatiranje teksta, kao što su **bold** ili *italik* tekst. Elementi za formatiranje su:

- **** - bold tekst
- **** - važan tekst
- *<i>* - italic tekst
- **** - naglašen (engl. **emphasized**) tekst
- **<mark>** - obeležen (engl. **marked**) teks
- **<small>** - mali tekst
- **** - obrisani tekst
- **<ins>** - ubačeni (engl. **inserted**) tekst
- **<sub>** - subskript tekst
- **<sup>** - superskript tekst

```
<!DOCTYPE html>
<html>
<body>

<p> <b> Ovaj tekst je boldovan </b> . </p>
<p> <strong> Ovo je važan tekst </strong> . </p>
<p> <i> Ovaj tekst je italic </i> . </p>
<em> Ovaj tekst je naglašen </em>
<h2> HTML <small> malo </small> formatiranje </h2>
```

```
<h2> HTML <mark> obeleženo </mark> formatiranje </h2>
<p> Moja omiljena boja je <del> crna </del> . </p>
<p> Moja omiljena boja je <ins> plava </ins> . </p>
<p> Ovo je <sub> subskript </sub> . </p>
<p> Ovo je <sup>superskript </sup> . </p>

</body>
</html>
```

Ovaj tekst je boldovan.
Ovo je važan tekst.
Ovaj tekst je italik.
Ovaj tekst je naglašen

HTML malo formatiranje

HTML obeleženo formatiranje

Moja omiljena boja je ~~crna~~.
Moja omiljena boja je plava.
Ovo je _{subskript}.
Ovo je ^{superskript}.

Slika 5.1.1 Prikaz HTML formatiranja teksta

HTML BOJE

Nazivi boja: tomato, orange, dodgerblue, mediumseagreen, gray, violet, lightgray

U HTML-u, boja se može odrediti pomoću naziva boje:

```
<!DOCTYPE html>
<html>
<body>

<h1 style="background-color:Tomato;">Tomato</h1>
<h1 style="background-color:Orange;">Orange</h1>
<h1 style="background-color:DodgerBlue;">DodgerBlue</h1>
<h1 style="background-color:MediumSeaGreen;">MediumSeaGreen</h1>
<h1 style="background-color:Gray;">Gray</h1>
<h1 style="background-color:SlateBlue;">SlateBlue</h1>
<h1 style="background-color:Violet;">Violet</h1>
<h1 style="background-color:LightGray;">LightGray</h1>

</body>
</html>
```



Slika 5.1.2 Primeri HTML boja

HTML elementi mogu imati svoje boje u pozadini:

```
<!DOCTYPE html>
<html>
<body>

<h1 style="background-color:DodgerBlue;">Naslov</h1>

<p style="background-color:Tomato;">
Tekst sa crvenom pozadinom
</p>

</body>
</html>
```

Naslov

Tekst sa crvenom pozadinom

Slika 5.1.3 Primeri elemenata sa obojenom pozadinom

BOJE TEKSTA I OKVIRA

Pomoću HTML atributa se može obojiti tekst ili okviri oko teksta

Primer bojenja teksta:

```
<!DOCTYPE html>
<html>
<body>

<h3 style="color:Tomato;">Naslov</h3>

<p style="color:DodgerBlue;"> Plavi tekst. </p>

<p style="color:MediumSeaGreen;"> Zeleni tekst. </p>
```

```
</body>  
</html>
```

Primer teksta sa okvirima u boji:

```
<!DOCTYPE html>  
<html>  
<body>  
  
<h1 style="border: 2px solid Tomato;"> Naslov 1 sa crvenim okvirom </h1>  
  
<h1 style="border: 2px solid DodgerBlue;"> Naslov 2 sa plavim okvirom </h1>  
  
<h1 style="border: 2px solid Violet;"> Naslov 3 sa ljubičastim okvirom </h1>  
  
</body>  
</html>
```

Naslov 1 sa crvenim okvirom

Naslov 2 sa plavim okvirom

Naslov 3 sa ljubičastim okvirom

Slika 5.1.4 Primer teksta sa okvirima u boji

STILIZOVANJE HTML POMOĆU CSS

Najčešći način za dodavanje CSS je pomoću odvojenog CSS fajla

CSS je skraćenica za Cascading Style Sheets. CSS opisuje kako će HTML elementi biti prikazani. CSS štedi mnogo vremena. On može da upravlja prikazom više veb stranica istovremeno. CSS se može dodati na tri načina:

1. unutar HTML elemenata koristeći style atribut
2. unutar HTML datoteke koristeći element `<style>` u sekciji `<head>`
3. eksterno koristeći CSS datoteku

Najčešći način za dodavanje CSS je pomoću odvojenog CSS fajla, ali ćemo ovde opisati sva tri načina.

CSS UNUTAR HTML ELEMENATA KORISTEĆI STYLE ATRIBUTE

CSS unutar HTML elemenata se koristi za primenu jedinstvenog stila na jedan HTML element

CSS unutar HTML elemenata se koristi za primenu jedinstvenog stila na jedan HTML element. Ovaj primer postavlja boju teksta elementa `<h1>` da bude plava:

```
<!DOCTYPE html>
<html>
<body>

<h1 style="color:blue;"> Ovo je plavi naslov </h1>

</body>
</html>
```

UNUTAR HTML DATOTEKE KORISTEĆI ELEMENT

Interni CSS se koristi za definisanje stila za jednu HTML stranicu

Interni CSS se koristi za definisanje stila za jednu HTML stranicu. Interni CSS je definisan u sekciji `<head>` HTML stranice unutar elementa `<style>`:

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      body {background-color: powderblue;}
      h1 {color: blue;}
      p {color: red;}
    </style>
  </head>
  <body>

    <h1> Ovo je naslov </h1>
    <p> Ovo je paragraf. </p>

  </body>
</html>
```

EKSTERNI CSS

Kako bi se koristio eksterni CSS potrebno je da se doda link ka CSS datoteci u sekciji `<head>`

Eksterni CSS se koristi za definisanje stila za mnoge HTML stranice. Kako bi se koristio eksterni CSS potrebno je da se doda link ka CSS datoteci u sekciji `<head>` :

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" href="styles.css">
  </head>
```

```
<body>

    <h1> Ovo je naslov </h1>
    <p> Ovo je paragraf. </p>

</body>
</html>
```

Eksterni CSS može biti napisan u bilo kom editoru. CSS datoteka ne treba da sadrži HTML kod i treba da se sačuva pod ekstenzijom *.css. Ovako izgleda primer styles.css datoteke:

```
body {
background-color: powderblue;
}

h1 {
color: blue;
}

p {
color: red;
}
```

✓ Poglavlje 6

Pokazna vežba: HTML tabele i forme

HTML TABELE

HTML tabele se sastoje od redova i ćelija. Svaki red tabele može da sadrži jednu ili više ćelija.

Predviđeno vreme pokazne vežbe je 60 minuta.

Tabela je HTML element za prikazivanje tabelarnih podataka. U prošlosti su se HTML tabele često koristile kao mehanizam za raspoređivanje elemenata na stranici ([layout](#)), ali takvo korišćenje tabela se smatra lošom praksom od uvodjenja CSS-a, i zato ga treba izbegavati.

HTML tabele se sastoje od redova i ćelija. Svaki red tabele može da sadrži jednu ili više ćelija.

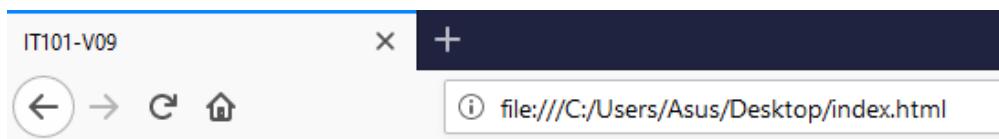
HTML tag za tabele je `<table>`, redove `<tr>` i ćelije `<td>`.

PRIMER HTML TABELE

Tabla ima dva reda, sa po dve ćelije.

Sledi kratak primer HTML tabele. Tabela ima dva reda, sa po dve ćelije.

```
<table>
  <tr>
    <td> Dejan </td>
    <td> Maksimovic </td>
  </tr>
  <tr>
    <td> Milos </td>
    <td> Ljubinkovic </td>
  </tr>
</table>
```



Primer HTML tabele

Dejan Maksimovic
Milos Ljubinkovic

Slika 6.1 Prikaz HTML tabele

ATRIBUTI HTML TABELA

Redove tabele je moguće grupisati u header, telo i footer `thead`, `tbody` i `tfoot`

Redove tabele je moguće grupisati u header, telo i footer tagovima `<thead>`, `<tbody>` i `<tfoot>`.

Ćelije tabele ne moraju biti jednakih dimenzija, tj. jedna ćelija može da zauzima više redova i kolona. Atributi `rowspan` i `colspan` elementa `td` kontrolisu veličinu ćelije.

Atribut `border` kontroliše debljinu okvira tabele.

PRIMER TABELE SA ATRIBUTIMA

Ćelije tabele ne moraju biti jednakih dimenzija, tj. jedna ćelija može da zauzima više redova i kolona. Atributi `rowspan` i `colspan` elementa `td` kontrolišu veličinu ćelije.

Sledi primer tabele sa atributima `border` i `colspan`.

```
<table border="1">
  <thead>
    <tr>
      <th> Ime </th>
      <th> Prezime </th>
      <th> Plata </th>
    </tr>
  </thead>

  <tbody>
    <tr>
```

```
<td> Marko </td>
<td> Markovic </td>
<td> 1000$ </td>
</tr>

<tr>
<td> Mirko </td>
<td> Mirkovic </td>
<td> 1200$ </td>
</tr>

<tr>
<td colspan="2"> Ukupna plata </td>
<td colspan="1"> 2200$ </td>
</tr>
</tbody>
</table>
```

Tabela se u browseru prikazuje na sledeći način:

Ime	Prezime	Plata
Marko	Markovic	1000\$
Mirko	Mirkovic	1200\$
Ukupna plata		2200\$

Slika 6.2 Prikaz tabele

HTML FORME

Početak forme za slanje informacija počinje sa <FORM>

Ako je potrebno da korisnik internet strane pošalje neke podatke to može da se uradi na dva načina.

Prvi je da se pošalju podaci na sajt gde bi te podatke dočekala CGI skripta sa serverske strane ili neki drugi serverski program i obradili poslate podatke. Drugi način bi bio da se prikupljeni podaci pošalju na neku email adresu. Možemo da pošaljemo i na više email adresa ali svaka adresa mora od sledeće da bude odvojena zarezom. Način slanja na email će biti korišćen u našem primeru. Poslate informacije će biti kodirane kao običan tekst.

Korisnik podatke unosi u polja za unos podataka. Ta polja imaju standardne grafičke komponente kao što su combobox, tekstualno polje... Kada korisnik odluči da informacije pošalje pritiska dugme Submit i time se sve informacije koje se nalaze u poljima skupljaju i šalju.

Da bi mogli da definišemo koje informacije treba da se prikupe pre slanja imamo tag *FORM* koji je zadužen za definisanja formi.

Početak forme za slanje informacija počinje sa **<FORM>** i završava se sa**</FORM>**

```
<FORM METHOD=POST ACTION="mailto: primeremaila@metropolitan.ac.rs "
ENCTYPE="text/plain">
```

POLJE ZA UNOS TEKSTA

*Polje u koje korisnik može da unese tekstualni podatak je *TextField**

Najčešći podatak koji očekujemo od korisnika je običan tekstualni podatak. Polje u koje korisnik može da unese tekstualni podatak je *TextField*. Sva polja u koje korisnik unosi podatke se u FORM-ama nazivaju inputi, a tipom određujemo koja je vrsta inputa. Odnosno tag kojim se označava polje za unos podatka je *INPUT*. Za tekstualna polja tip inputa je *TEXT*.

```
<INPUT TYPE=TEXT NAME="Ime" VALUE="Bezimeni" SIZE=30>
```

Svaki input mora da ima naziv (*name*). Tako dolazimo do toga da svaki input mora da ima dva parametra, to su *TYPE* i *NAME*. Osim ova dva obavezna možemo da imamo i opcione parametre.

Opcioni parametri ponekad zavise od tipa inputa. Za tekstualne inpute ne obavezni parametri mogu biti *SIZE*, *VALUE* i *MAXLENGTH*.

Parametar *VALUE* nam daje default vrednost tog polja. To znači ako korisnik ne unese ništa u to polje podrazumevana vrednost definisana ovim parametrom će se poslati umesto korisnikovih informacija.

SIZE određuje veličinu polja u koje korisnik unosi podatke dok parametar *MAXLENGTH* određuje koliko karaktera korisnik može maksimalno da unese. Polje mu neće dozvoliti da unese više karaktera nego što je vrednost ovog parametra:

```
<INPUT TYPE=TEXT NAME="Ime" SIZE=30 MAXLENGTH=10>
```

PASSWORD POLJE

*Password polje se razlikuje od običnog tekstualnog samo po tome što ispisuje * dok se kuca u njemu.*

Pod varijanta tekstualnog polja je *password* polje. Ovo polje se razlikuje od običnog tekstualnog samo po tome što ispisuje * dok se kuca u njemu.

<INPUT TYPE=PASSWORD NAME="Korisnički password">

RADIO DUGME

Ako korisnik treba da bira jednu od ponuđenih mogućnosti to možemo da realizujemo pomoću komponente Radio Button

Ako korisnik treba da bira jednu od ponuđenih mogućnosti to možemo da realizujemo pomoću komponente *Radio Button*. Vrednost TYPE parametra INPUT taga je *RADIO* za ovu komponentu. Ona obezbeđuje kružić sa tekstom pored. Samo jedan kužić može da bude odabran u jednom trenutku. Naravno ovo se odnosi samo na one Radio Dugmiće koji pripadaju istoj grupi. Nekoliko Radio dugmića smeštamo u istu grupu tako što im damo istu vrednost NAME parametra. Da bi funkcionisanje ove komponente imalo smisla moramo da imamo bar još jedan parametar VALUE koji treba da ima različite vrednosti za različite radio dugmiće. U mogućnosti smo da odredimo koji od radio dugmića će biti odabran u trenutku pojavljivanja HTML strane. Dodatni parametar *CHECKED* bez vrednosti određuje da komponenta koja ima ovaj parametar je od početka odabrana. Moramo sami da vodimo računa o tome da ne bi trebalo da imamo više Radio dugmića sa ovim parametrom.

<INPUT TYPE=RADIO NAME="BEST FRIEND" VALUE="Ed" CHECKED> Edvard Butić

<INPUT TYPE=RADIO NAME="BEST FRIEND" VALUE="Rick"> Ricky Martin

<INPUT TYPE=RADIO NAME="BEST FRIEND" VALUE="Tom"> Tomislav Budić<P>

ŠTIKLIRANO POLJE

Za razliku od radio dugmića gde ime ostaje isto radi grupisanja kod štikliranog polja ime je različito za svako polje ali je vrednost (VALUE) obično ista za sva polja

Sledeća često korišćena komponenta je štiklirano polje. To je text koji pored sebe ima kvadratić koji se štiklira kad korisnik klikne na njega i odštiklira kad se ponovo klikne na isti. Ova komponenta je *INPUT TYPE=CHECKEDBOX*. Za razliku od radio dugmića gde ime ostaje isto radi grupisanja kod štikliranog polja ime je različito za svako polje ali je vrednost (VALUE) obično ista za sva polja. Kao i kod radio dugmeta možemo da koristimo parametar *CHECKED* da naznačimo da je neka stavka štiklirana još prilikom prvog učitavanja HTML strane. Za razliku od radio dugmića kod štikliranog polja parametar *CHECKED* može da se pojavi i na više stavki istovremeno.

<INPUT TYPE=CHECKBOX NAME="ED" VALUE="YES" CHECKED> Edvard Butić

<INPUT TYPE=CHECKBOX NAME="Rick" VALUE="YES"> Ricky Martin

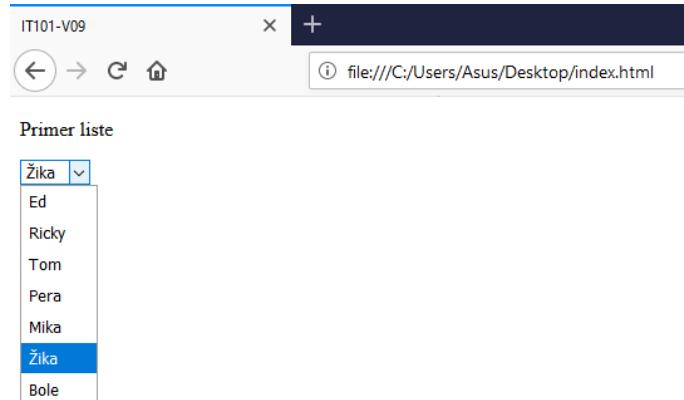
<INPUT TYPE=CHECKBOX NAME="Tom" VALUE="YES" CHECKED> Tomislav Budić

LISTA

Ako želimo da jedna od stavki sa liste bude uvek ponuđena onda koristimo parametar SELECTED.

Kada želimo da nekoliko stvari stavimo u listu pa da korisnik odabira jednu iz liste možemo da korisimo tag *SELECT*. Tag *SELECT* ima sličnu namenu kao i tag *INPUT*. Kao i kod *INPUT* taga neophodno je da *SELECT* tag ima naziv. Nakon toga idu tagovi (ne parametri kao kod *INPUT* taga) *OPTION* pri čemu svaki predstavlja novu stavku u ponuđenoj listi. Svaka opcija treba da ima *VALUE*. Ako želimo da jedna od stavki sa liste bude uvek ponuđena onda posle *VALUE* parametra odgovarajuće *OPTION* stavke stavimo i parametar *SELECTED*.

```
<SELECT NAME="BEST FRIEND">
  <OPTION VALUE="ED">Ed
  <OPTION VALUE="RICK">Ricky
  <OPTION VALUE="TOM">Tom
  <OPTION VALUE="PERA">Pera
  <OPTION VALUE="MIKA">Mika
  <OPTION VALUE="ZIKA" SELECTED>Žika
  <OPTION VALUE="BOLE">Bole
</SELECT>
```



Slika 6.3 Prikaz liste

TEXT AREA

Pomoću atributa ROWS možemo da definišemo koliko redova će biti visoko polje za unos

Ponekad želimo da ostavimo korisniku veći prostor za unos teksta. Ovo se radi pomoću komponente koja se zove Text Area i ima odgovarajući tag istog naziva *TEXTAREA*. Kao i ostale komponente za unos podataka neophodno je da ima ime (NAME). Dok su ostali parametri samo opcioni. Pomoću atributa *ROWS* možemo da definišemo koliko redova će biti visoko polje za unos a pomoću atributa *COLS* definišemo štinu polja za unos.

```
<TEXTAREA NAME="COMMENTS" ROWS=3 COLS=30 WRAP=VIRTUAL>  
</TEXTAREA>
```

Kad je reč o ovom tipu polja za unos postavlja se pitanje kako će duži text da se ponaša u takvom prostoru. Da li će da se prelama na sledeći red ili ne kad je tekst duži nego što je širina polja. Ovo se pitanje reguliše pomoću parametra *WRAP*. Parametar WRAP može da ima jednu od sledeće tri vrednosti: *VIRTUAL*, *PHYSICAL* ili *OFF*. OFF vrednost parametra određuje da nema prelamanja teksta dok druge dve određuju da li će da se tekst prelama samo vizuelno korisniku a slaće se ne prelomljen (vrednost parametra *VIRTUAL*) ili će da se prelama i korisniku i prilikom slanja (vrednost parametra *PHYSICAL*).

WRAP=VIRTUAL | PHYSICAL | OFF

SKRIVENO POLJE

Skriveno polje se ne prikazuje u browseru.

U našem primeru prijave za kurs imamo problem da sve prijave imaju istu formu. Uzimamo iste podatke. Kako će se znati za koji kurs se korisnik prijavio.

Ovo može da se reši ubacivanjem skrivenih polja. Skriveno polje se ubacuje kao tag INPUT a type mu je *HIDDEN*. Ovaj tag mora da ima svoj naziv (parametar NAME) i vrednost (parametar VALUE). Naše prijave za kurseve mogu da izgledaju isto ali će na svakoj prijavi da se razlikuje ovo skriveno polje. Kad dobijemo email sa prijavom pored podataka koje je korisnik uneo stojache i ovaj sakriveni podatak.

```
<INPUT TYPE=HIDDEN NAME="Prijava" VALUE="DB + ORM">
```

POLJE ZA UPLOAD FAJLA

U slučaju da šaljemo fajlove vrednost ENCTYPE parametra ne može da bude text/plain već mora da bude multipart/form-data

Kada pravimo forme ponekad želimo da nam korisnik pošalje neki fajl. Ovo se postiže upotrebom INPUT taga tipa *FILE*. Kada u okviru forme postoji i ovo polje onda mora da se vodi računa o vrednosti parametra *ENCTYPE*, FORM taga. U slučaju da šaljemo fajlove vrednost ovog parametra ne može da bude text/plain već mora da bude multipart/form-data.

```
<INPUT TYPE=FILE NAME="myfile">  
<FORM METHOD=POST ACTION="mailto:primeremaila@metropolitan.ac.rs">  
ENCTYPE="multipart/form-data">
```

SUBMIT DUGME

*Za potrebe slanja forme postoji **SUBMIT** dugme*

Da bi forma bila poslata neophodno nam je da napravimo neku akciju koja će da izvrši samo slanje forme. Za potrebe slanja forme postoji *SUBMIT dugme*. Ovo dugme pravimo tako što INPUT tag-u kažemo da je tipa SUBMIT. Kada korisnik klikne na ovo dugme forma u kojoj je bilo ovo dugme će biti poslata. Ako se ne definiše parametar VALUE onda će na dugmetu pisati "Submit", VALUE parametar definiše text na dugmetu. Pored Submit dugmeta možemo da imamo i dugme za restartovanje forme.

Pomoću ovog dugmeta smo u mogućnosti da kompletну formu restartujemo na prvobitne vrednosti.

Ovo dugme se pravi isto kao i SUBMIT samo što je TYPE parametar *RESET*.

INPUT TYPE=SUBMIT VALUE="Posalji!">

<INPUT TYPE=RESET VALUE="Ponovo!">

Obično dugme se često ne uklapa u neki definisani izgled strane pa nam treba mogućnost da umesto običnog dugmeta postavimo sliku. Ovo je moguće uraditi ako kao parametar TYPE INPUT taga stavimo IMAGE. Moramo da obezbedimo i SRC parametar ako ovo radimo. Na ovaj način možemo da napravimo samo SUBMIT dugme a ne i reset dugme. Kao i obično kad je reč o slikama možemo da obezbedimo još dodatnih parametara kao što su *WIDTH*, *HEIGHT*, *BORDER* i *ALT*.

```
<INPUT TYPE=IMAGE SRC="submit.gif" WIDTH=94 HEIGHT=26 BORDER=0  
ALT="Submit">
```

Name	Value
Name	
Sex	<input checked="" type="radio"/> Male <input checked="" type="radio"/> Female
Eye color	<input type="text" value="green"/>
Check all that apply	<input type="checkbox"/> Over 6 feet tall <input type="checkbox"/> Over 200 pounds
Describe your athletic ability:	
<input type="text"/> <input type="button" value="Enter my information"/>	

Slika 6.4 HTML forma

POKAZNA VEŽBA SA REŠENJEM

Prikaz pokazne vežbe sa rešenjem

Predviđeno vreme za izradu sledećeg zadatka je 10 minuta.

Potrebno je kreirati HTML stranu sa sledeće slike:

Ime:

Prezime:

Nešto o vama:

Omiljeni predmet: Matematika Fizika

Uplođujte vašu sliku: No file chosen

Slika 6.5 Zadatak

Rešenje:

```
<!DOCTYPE html>
<html>
    <head>
        <title>Primer</title>
    </head>
    <body>
        <form>
            Ime: <input type = "text" name = "ime" /> <br>
            Prezime: <input type = "text" name = "prezime"/> <br>
            Nešto o vama: <br>
            <textarea rows = "5" cols = "50" name = "description">
                Enter description here...
            </textarea><br>
            Omiljeni predmet:
            <input type = "radio" name = "subject" value = "math"> Matematika
            <input type = "radio" name = "subject" value = "physics"> Fizika<br/>
            Uplođujte vašu sliku:
            <input type = "file" name = "fileupload" accept = "image" >
        </form>
    </body>
</html>
```

▼ Poglavlje 7

Pokazna vežba: CSS - Cascading Style Sheet

CSS SELEKTORI

Selektor se koristi za određivanje HTML elementa na kome će se primeniti stil.

Predviđeno vreme pokazne vežbe je 25 minuta.

Selektor se koristi za određivanje HTML elementa na kome će se primeniti stil. Zatim se definiše stil tako što se navede CSS svojstvo i njegova vrednost. Svako svojstvo se odvaja sa " ; ".

Primer:

```
p {  
    color: red;  
    text-align: center;  
}
```

U ovom primeru su selektovani svi elementi sa tagom <p>. Oni će biti centrirani i crvene boje.

Elementi se mogu selektovati i pomoću id-ja. Id elementa mora da bude jedinstven, odnosno može se iskoristiti samo jednom na stranici. Element sa id-jem se označava znakom " # ".

Primer:

```
#paragraf {  
    text-align: center;  
    color: red;  
}
```

U ovom primeru, selektovan je samo jedan element koji ima id "paragraf". Samo taj element će biti centriran i crvene boje.

Elementi se još mogu selektovati i pomoću klase. Klasa se razlikuje od id-ja po tome što se klase mogu iskoristiti na više elemenata. Element sa klasom se označava znakom " . ".

Primer:

```
p.center {  
    text-align: center;
```

```
color: red;  
}
```

Sada će samo elementi taga <p> koji imaju klasu "center" biti centrirani i crvene boje.

MARGIN

Margine stvaraju prostor između elemenata

Margine prave prostor oko elementa. Mogu se odrediti gornja, donja, leva i desna margina.

```
p {  
    margin-top: 100px;  
    margin-bottom: 100px;  
    margin-right: 150px;  
    margin-left: 80px;  
}
```

Ovo znači da će oko selektovanog elementa sa gornje i donje strane postojati 100px prostora, sa desne 150px i sa leve 80px.

Margine se mogu određivati po dužini (pikseli), po procentima, automatski a element može i da nasledi margine od roditeljskog elementa.

PADDING

Padding određuje rastojanje sadržaja elementa od njegovih ivica

Padding određuje rastojanje sadržaja elementa od njegovih ivica. Kao i za margine, može se odrediti donji, gornji, levi i desni padding.

```
div {  
    padding-top: 50px;  
    padding-right: 30px;  
    padding-bottom: 50px;  
    padding-left: 80px;  
}
```

PRIMER MENIJA SA OPCIJAMA

Primer interaktivnog menija

Predviđeno vreme izrade sledećeg primera je 10 minuta.

Prvo treba napraviti index.html stranicu gde će biti definisan sadržaj menija u nenumerisanoj listi.

```
<!DOCTYPE>
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="style.css">
  </head>
  <body>
    <body>
      <header>
        <div class="nav">
          <ul>
            <li class="home"><a href="#">Home</a></li>
            <li class="tutorials"><a class="active" href="#">Tutorials</a></li>
            <li class="about"><a href="#">About</a></li>
            <li class="news"><a href="#">Newsletter</a></li>
            <li class="contact"><a href="#">Contact</a></li>
          </ul>
        </div>
      </header>
    </body>
  <html>
```

MENI SA OPCIJAMA - NASTAVAK

Nastavak primera interaktivnog menija

Zatim treba napraviti css dokument koji određuje stil menija.

```
body {
  margin: 0;
  padding: 0;
  background: #ccc;
}

.nav ul {
  list-style: none;
  background-color: #444;
  text-align: center;
  padding: 0;
  margin: 0;
}

.nav li {
  font-family: 'Oswald', sans-serif;
  font-size: 1.2em;
  line-height: 40px;
  height: 40px;
  border-bottom: 1px solid #888;
}

.nav a {
  text-decoration: none;
```

```
color: #fff;
display: block;
transition: .3s background-color;
}

.nav a:hover {
background-color: #005f5f;
}

.nav a.active {
background-color: #fff;
color: #444;
cursor: default;
}

@media screen and (min-width: 600px) {
.nav li {
width: 120px;
border-bottom: none;
height: 50px;
line-height: 50px;
font-size: 1.4em;
}

.nav li {
display: inline-block;
margin-right: -4px;
}
}
```

Prvo se određuju margine za body element kao i boja pozadine. Zatim se stilizuje nenumerisana lista, određuje joj se boja pozadine, margine, pozicioniranje.

.nav li se odnosi na pojedinačne elemente liste. Određuje im se font, veličina fonta, visina, kao i okvir.

.nav a se odnosi na link koji predstavlja svaki od elemenata liste. Svojstvo koje se odnosi na tranziciju određuje da se boja postepeno menja kada se mišem pređe preko.

@media pravila se koriste kada je potrebno primeniti stilove na različite forme medija, na primer na različite veličine ekrana. U ovom primeru, min-width određuje da će se ovaj stil primenjivati ukoliko je veličina ekrana veća od 600px.

✓ Poglavlje 8

Zadatak za samostalni rad: HTML i CSS

ZADATAK ZA SAMOSTALNI RAD - HTML TABELE I FORME

Kreirati HTML stranicu

Predviđeno vreme za izradu sledećih zadataka je 30 minuta.

Zadatak 1

Kreirati HTML stranicu na temu po vašem izboru. Strana treba da sadrži tekst različite boje i veličine fonta i numerisane liste.

Zadatak 2

- Kreirati dva HTML dokumenta: *tabela.html* i *polja.html*.
- *tabela.html* je potrebno sadrži elemente tabele koja je data na sledećoj slici:

IME	PREZIME	ADRESA	BROJ TELEFONA	GRAD
Mika	Mikić	Juhorska 5/3	064 888 99 99	BEOGRAD
Pera	Perić	Episkopska 6	063 55 66 444	Niš
Nikola	Nikolić	Duvanska 6	067 99 88 88	Vršac

Slika 8.1 Tabela-1 Prikaz tabele sa njenim elementima

- Strana *polja.html* je potrebno da izgleda kao prikaz na sledećoj slici. *Naruči* dugme može biti druge boje. Adresa je potrebno da bude *text area*.

Ime

Prezime

Adresa

Grad

Država Srbija

Poštanski broj:

Broj telefona:

Email

Narucujem: Barbie Computer Engineer

Naruči

Slika 8.2 Prikaz strane polja.html

ZADATAK ZA SAMOSTALNI RAD - CSS

Zadatak za vežbanje stilova

Predviđeno vreme za izradu sledećih zadataka je 20 minuta.

1. Napraviti funkcionalni meni. Klikom na svaki stavku otvara se nova stranica.
2. Svaka stranica treba da ima isto zaglavje i pozadinu.
3. Na novim stranicama primeniti naučeno o formama i tabelama.
4. Koristiti CSS za stilizovanje formi i tabela.

✓ Poglavlje 9

DOMAĆI ZADATAK

OPIS DOMAĆEG ZADATKA - DZ09

Kreirati HTML front end sistem koji će omogućiti unos i pregled osoba koji su doprineli razvoju računarstva

Očekivano vreme izrade zadatka: 45min.

1. Kreirati tri HTML stranice.
2. Početna stranica treba da sadrži naslov i meni sa opcijama (svaka opcija iz menija vodi ka ostalim stranicama - postaviti linkove za drugu i treću stranicu).
3. Druga stranica treba da sadrži formu za unos podataka o predispitnim obavezama studenata na predmetu IT101 (iskoristiti sve pomenute elemente).
4. Na trećoj stranici treba da se nalazi popunjena tabela sa poljima koja odgovaraju formi sa druge stranice. Tabelu popunite sa Vašim predispitnim obavezama.
5. Druga i treća stranica takođe poseduju meni sa opcijama i klikom na određenu stavku iz menija prikazati izabranu stranu.
6. Koristiti CSS za stilizovanje forme i tabele.

Zadatak dostaviti kao arhivu **IT101-DZ09-Ime_Prezime_BrojIndeksa.zip**. Arhiva treba da sadrži tri HTML fajla: index.html, prikaz.html i unos.html. Domaći zadatak pošaljite predmetnom asistentu na e-mail.

▼ Zaključak

ZAKLJUČAK

U ovom predavanju bilo je reči o markap jezicima, sa fokusom na HTML jezik. U ovom predavanju je dat pregled osnovnih HTML konstrukcija, kao i način njihove primene. Ne treba zaboraviti da je osnova prenosa podataka preko veba, kada je reč o veb stranicama, HTTP protokol. U predavanju su date njegove osnovne karakteristike i način rada klijent/server komunikacionog modela.

Literatura

1. **W3, The Global Structure of HTML Document, <http://www.w3.org/TR/1999/PR-html40-19990824/struct/global.html> (Last accessed 17.07.2015).**



IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

SERVER/KLIJENT PROGRAMIRANJE

Lekcija 10

PRIRUČNIK ZA STUDENTE

IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

Lekcija 10

SERVER/KLIJENT PROGRAMIRANJE

- ▼ SERVER/KLIJENT PROGRAMIRANJE
- ▼ Poglavlje 1: Programiranje sa klijentske strane
- ▼ Poglavlje 2: Skripting jezici
- ▼ Poglavlje 3: Ugnježdeni objekti
- ▼ Poglavlje 4: Programiranje sa serverske strane
- ▼ Poglavlje 5: Veb serveri
- ▼ Poglavlje 6: Pokazna vežba: Upotreba JavaScript-a u HTML-u
- ▼ Poglavlje 7: Zadatak za samostalni rad: JS u HTML
- ▼ Poglavlje 8: Domaći zadatak
- ▼ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

❖ Uvod

UVOD

Cilj ove lekcije je da opiše razlike između klijentskog i serverskog programiranja

Na ovom predavanju biće reči o opštim karakteristikama programiranja sa serverske i klijentske strane, kao i navedeni osnovni primeri za neke od često korišćenih skripting jezika.

Sadržaj ove lekcije je sledeći:

- programiranje klijentske strane
- programiranje serverske strane
- kada se preporučuje programiranje sa klijentske, a kada sa serverske strane

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 1

Programiranje sa klijentske strane

TEHNIKA PROGRAMIRANJA SA KLIJENTSKE STRANE

Programiranje sa klijentske strane se može postići pisanjem skripta i ugneždenim objektima

Na klijentskoj strani veba se nalazi veb klijent, koji je uobičajeno neki čitač veb strana (engl. **web browser**). Zadatak veb čitača je da prikaže veb stranicu dobijenu od veb servera. Pri tom se korisniku ne pruža nikakva mogućnost za interakciju. Interakcija zahteva da se neke rutine izvršavaju i na klijentskoj strani. Mogućnost izvršenja koda na klijentskoj strani može da bude od velikog značaja.

Posmatrajmo primer izvršenja neke veb aplikacije u kojoj udaljeni klijent treba da popuni polja u nekoj formi. Validnost unetih podataka se može obaviti na serverskoj strani, ali u slučaju nevalidnih ili nepotpunih podataka korisnik mora više puta da pristupa serverskoj strani. Umesto toga, korišćenjem neke rutine koja može da se izvrši na klijentskoj strani verifikacija se može izvršiti lokalno, što je mnogo brže i smanjuje mrežni saobraćaj.

Postavlja se pitanje da li je i na klijentskoj strani moguće realizovati određenu interaktivnost, odnosno izvršiti neki program. Odgovor je potvrđan jer i klijentska strana koristi računarski sistem, pa može i da izvršava neki programski kod. Problem je, međutim, u tome što klijenti veb stranama pristupaju korišćenjem različitih hardverskih i softverskih platformi i različitih veb čitača, što otežava pisanje rutina koje će se izvršavati na klijentskoj strani.

Izvršavanje programa unutar veb strane na klijentskoj strani može da se postigne na dva načina:

- pisanjem skripta
- ugneždavanjem programiranih objekata.

Skript jezici nisu tako kompleksni kao uobičajeni programski jezici, pa nisu podesni za pisanje kompleksnih programa. Uglavnom se koriste kada je potrebno napraviti jednostavne rutine kojim se proveravaju podaci na klijentskoj strani ili kada je potrebno manipulisati elementima veb stranice. Za veb programere je najlakši način dodavanja dinamičkog aspekta veb strani pisanje skripta koji se izvršava na klijentskoj strani (**client-side scripting**), uglavnom, korišćenjem JavaScript jezika.

Ako je reč o složenijim zahtevima onda se koristi tehnika ugneždenih objekata. Ovi objekti se programiraju u računarskim jezicima kao što su Java ili C++.

KADA SE KORISTI PROGRAMIRANJE SA KLIJENTSKE STRANE?

Primeri skripting jezika koji se koriste na klijentskoj strani su JavaScript, VBScript, HTML, CSS, AJAX, jQuery i drugi

Programiranje sa klijentske strane se koristi kada:

- želimo da postignemo interaktivnost veb stranice
- uopšteno kada želimo dinamičnost
- imamo interakciju sa privremenim skladištem
- pravimo interfejs između korisnika i servera
- šaljemo zahtev serveru
- imamo interakciju sa lokalnim skladištem
- želimo da pružimo udaljeni pristup serverskom delu programa
- želimo da obavimo verifikaciju lokalno, kako bi smanjili mrežni saobraćaj

Postoje mnogi skripting jezici koji se koriste na klijentskoj strani, od kojih su neki:

- JavaScript
- VBScript
- HTML (struktura)
- CSS (dizajn)
- AJAX
- jQuery itd.

Pored ovih mogu se koristiti i drugi jezici za modelovanje, grafiku, animaciju i dodatne funkcionalnosti.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 2

Skripting jezici

JAVASCRIPT

U slučaju JavaScript-a kod skripta se ili umeće direktno u HTML dokument ili se iz tog dokumenta referencira neki JavaScript

Skript jezik za programiranje klijentske strane **JavaScript** je razvijen od strane Netscape-a. Microsoft ima svoju verziju ovog jezika koja se zove JScript i koristi se u veb čitaču Internet Explorer. JavaScript je standardizovan od strane **European Computer Manufacturers Association** (ECMA) 1997. godine i od tada se mnogi proizvođači veb čitača deklarišu da su kompatibilni sa ECMAScript-om.

U slučaju JavaScript-a kod skripta se ili umeće direktno u HTML dokument ili se iz tog dokumenta referencira neki JavaScript. Sintaksa JavaScript-a je slična jeziku C ili Java. S ciljem demonstriranja lakoće korišćenja daje se sledeći primer koda koji iscrtava boks sa pozdravom nakon što se pritisne taster „Click me“. U ovom slučaju JavaScript funkcija koja je nazvana **hello()** je pokrenuta korisnikovim klikom na dugme “Click me”.

Primer 1

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en">
<head>
<title>My first JavaScript</title>
<meta http-equiv="content-type" content="text/html; charset=ISO-8859-1" />
<script type="text/javascript">
<!--
function hello()
{
    alert("Hello! Isn't this easy?");
}
//-->
</script>
</head>
<body>
<h1 align="center">My first JavaScript</h1>
<div align="center">
<form action="#">
<input type="button" value="Click me" onclick="hello()" />
</form>
</div>
```

```
</body>  
</html>
```



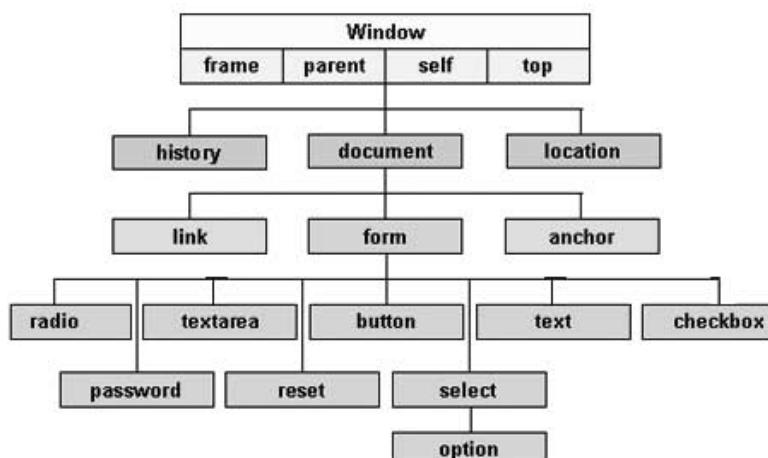
Slika 2.1 Rezultat pokretanja skripte iz Primera 1

JAVASCRIPT - DOCUMENT OBJECT MODEL (DOM)

Način na koji se pristupa i modificuje sadržaj dokumenta se naziva Document Object Model

Svaka veb stranica se prikazuje u prozoru veb čitača, koji možemo da posmatramo kao objekat . Objekat *Dokument* predstavlja HTML dokument koji je prikazan u ovom prozoru . Objekat *Dokument* ima različite osobine koje se odnose na druge objekte koji omogućavaju pristup i modifikaciju sadržaja dokumenta .

Način na koji se pristupa i modificuje sadržaj dokumenta se naziva **Document Object Model** ili skraćeno **DOM**. Objekti su organizovani hijerarhijski. Ova hijerarhijska struktura se odnosi na organizaciju objekata u veb dokumentu. Na slici 2 je prikazana jednostavna hijerarhija sa DOM objektima.



Slika 2.2 Primer jednostavne hijerarhije DOM objekata

- Window objekat – na vrhu hijerarhije. To je spoljni element objekta hijerarhije.
- Document objekat – Svaki HTML dokument koji se prikazuje u prozoru čitača postaje objekat Dokument. Dokument sadrži sadržaj stranice.
- Form objekat – Sve što se nalazi između tagova <form>...</form> predstavlja ovaj objekat.
- Kontrolni elementi forme (eng. **Form control elements**) – objekti forme sadrže sve elemente koji su definisani u objektu kao što su tekst, polje za tekst (**text field**), dugme (**button**), radio dugme (**radio button**) i boks za štikliranje (**checkbox**).

Postoji nekoliko modela DOM-a:

- Legacy DOM – Ovo je model koji je se pojavio tokom ranih verzija JavaScript jezika, tako da je dobro podržan od strane svih pretraživača, ali dozvoljava pristup samo određenim ključnim delovima dokumenata, kao što su forme, elementi forme i slike.
- W3C DOM – Ovaj model dokument objekta omogućava pristup i modifikaciju svih sadržaja dokumenta i standardizovan je od strane **World Wide Web Consortium** (W3C). Ovaj model je podržan od strane skoro svih modernih veb čitača.
- IE4 DOM - Uveden je u verziji 4 Microsoft-ovog Internet Explorer-a. IE 5 i novije verzije uključuju podršku za većinu osnovnih W3C DOM funkcija.

VBSCRIPT

VBScript se može izvršavati samo u MS Explorer-u

VBScript, odnosno **Visual Basic Scripting Edition** je deo Microsoft-ovog jezika Visual Basic koji je kreiran kao Microsoftov odgovor na JavaScript. Nažalost, ove skripte se mogu izvršavati samo u MS Exploreru, pa ga programeri izbegavaju.

Kao primer VBScript-a dajemo neke osnovne primere. Primer 2 kao izlaz daje niz karaktera koja će biti prikazana u veb čitaču. Kada se izvrši ova skripta, u čitaču će biti prikazano "Hello everyone!".

Primer 2

```
<html>
<body>

<script type="text/vbscript">
    document.write("Hello to everyone!")
</script>

</body>
</html>
```

Hello to everyone!

Slika 2.3 Rezultat pokretanja skripte iz Primera 2

Primer 3 koristi *for petlju*, koja se izvršava 6 puta kako bi 6 puta prikazala drugačiji stil zaglavlja (engl. **header**).

Primer 3

```
<html>
<body>
<script type="text/vbscript">
  for i=1 to 6
    document.write("<h" & i & ">This is Header" & i & "</h" & i & ">")
  Next
</script>
</body>
</html>
```

This is Header1

This is Header2

This is Header3

This is Header4

This is Header5

This is Header6

Slika 2.4 Rezultat nakon izvršenja VBScript-e iz Primera 3

FUNKCIJA U VBSCRIPT

*Najčešći način da se definiše funkciju u VBScript je pomoću **Function** ključne reči, a zatim navodimo jedinstveno ime te funkcije*

Pre nego što koristimo funkciju, moramo da definišemo tu konkretnu funkciju. Najčešći način da se definiše funkciju u VBScript je pomoću **Function** ključne reči, a zatim navodimo jedinstveno ime te funkcije, nakon čega može, ali i ne mora da se nađe lista parametara. Iskaz sa ključnom reči **End Function** označava kraj funkcije. Osnovna sintaksa je prikazana niže:

```
<!DOCTYPE html>
<html>
```

```
<body>
<script language="vbscript" type="text/vbscript">

Function ImeFunkcije(parameter-list)
    iskaz 1
    iskaz 2
    iskaz 3
    .....
    iskaz n
End Function

</script>
</body>
</html>
```

Primer 4

```
<!DOCTYPE html>
<html>
<body>
<script language="vbscript" type="text/vbscript">

Function kaziZdravo()
    msgbox("Zdravo svima!")
End Function

</script>
</body>
</html>
```

POZIVANJE FUNKCIJE U VBSCRIPT

Da biste aktivirali funkciju negde kasnije u skripti potrebno je da se napiše ime te funkcije sa ključnom reči Call.

Da biste aktivirali funkciju negde kasnije u skripti potrebno je da se napiše ime te funkcije sa ključnom reči **Call**.

```
<!DOCTYPE html>
<html>
<body>
<script language="vbscript" type="text/vbscript">

Function kaziZdravo()
    msgbox("Zdravo svima!")
End Function

Call kaziZdravo()

</script>
```

```
</body>  
</html>
```

Do sada smo videli funkciju bez parametara, ali postoji mogućnost da se proslede različiti parametri kada se poziva funkcija.

```
<!DOCTYPE html>  
<html>  
<body>  
<script language="vbscript" type="text/vbscript">  
  
Function kaziZdravo(ime, godine)  
    msgbox(ime & " ima " & godine & " godina.")  
End Function  
  
Call kaziZdravo("Mila", 7)  
  
</script>  
</body>  
</html>
```

VRAĆANJE VREDNOSTI IZ FUNKCIJE

Funkcija u VBScript može imati i iskaz koji vraća neku vrednost.

Funkcija u VBScript može imati i iskaz koji vraća neku vrednost. Na primer, možete funkciji proslediti dva broja, a od nje očekivati da vrati njihov zbir. Funkcija može da vrati više vrednosti razdvojenih zarezom.

Ova funkcija uzima dva parametra, spaja ih i vraća rezultat. U VBScript-i vrednosti se vraćaju koristeći ime funkcije. U slučaju da želimo da vratimo dva ili više vrednosti, tada se ime funkcije vraća sa nizom vrednosti.

```
<!DOCTYPE html>  
<html>  
<body>  
<script language="vbscript" type="text/vbscript">  
  
Function concatenate(first, last)  
    Dim full  
    full = first & last  
    concatenate = full  'Returning the result to the function name itself  
End Function  
  
</script>  
</body>  
</html>
```

Sada ovu funkciju možemo pozvati na sledeći način:

```
<!DOCTYPE html>
<html>
<body>
<script language="vbscript" type="text/vbscript">
Function concatenate(first, last)
    Dim full
    full = first & last
    concatenate = full  'Returning the result to the function name itself
End Function
' Here is the usage of returning value from function.
dim result
result = concatenate("Zara", "Ali")
msgbox(result)
</script>
</body>
</html>
```

DINAMIČKI HTML

DHTML = HTML + CSS + JavaScript

Sa pojavom veb čitača četvrte generacije pojavila se i nova tehnologija nazvana **Dynamic HTML** ili skraćeno **DHTML**. Korišćenjem ove tehnologije omogućeno je dinamičko manipulisanje elementima veb stranice. Najočigledniji primeri su pomeranje elemenata po stranici, prikazivanje ili skrivanje dela stranice. Ovo je omogućeno ukrštanjem **HTML, CSS i JavaScript-a**. Zato se može napisati da je

DHTML = HTML + CSS + JavaScript

Zbog mešavine svih ovih tehnologija postoji realna opasnost da će različiti veb čitači različito prikazivati jednu DHTML stranu.

OSNOVE DOM I JAVASCRIPT

Essentials of the DOM and JavaScript in 10 Minutes

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

JQUERY

Svrha jQuery je da olakša korišćenje JavaScripta na veb stranici.

Svrha **jQuery** je da olakša korišćenje JavaScripta na veb stranici. Dobra strana jQuery-a je mogućnost da obavi neki zadatak za koji je potrebno mnogo linija JavaScript koda i da ga reši

ponekad i samo sa jednom linijom koda. Drugim rečima, JavaScript uprošćava kompleksnost i komplikovanost JavaScript, u šta spada pozivanje AJAX i DOM manipulacija.

jQuery biblioteka sadrži sledeće karakteristike:

- HTML/DOM manipulacija
- CSS manipulacija
- HTML metode za događaje
- Razne efekte i animacije
- AJAX i dr.

Neke od prednosti jQuery-a su:

- Smanjenje količina potrebnog koda u odnosu na čist JavaScript što ga čini lakim za čitanje, pa samim tim je i smanjeno vreme potrebno za razvoj aplikacije
- Poboljšanje performansi sajta
- Lakši je za učenje od JavaScript
- Organizovana dokumentacija i prilično velika zajednica za podršku
- Lakša optimizacija veb sajta za različite veb čitače.

▼ Poglavlje 3

Ugnježdeni objekti

PLUG-INS

Plug-ins su male programske komponente koje daju veb čitaču novu funkcionalnost.

Plug-ins su male programske komponente koje daju veb čitaču novu funkcionalnost. Tipičan primer plug-ins-a su programi za emitovanje video i audio zapisa kao što su Flash ili QuickTime. Plug-ins su Netscape-ova tehnologija, ali su podržani i od drugih čitača. Nedostatak korišćenja plug-inova je činjenica je potrebno da budu instalirani na računaru, a ponekad je neophodno i da se računar restartuje. Takođe, često su pisani samo za određeni tip operativnih sistema, što znači da ukoliko korisnik ima drugačiji OS od onoga za koji je plug-in pisan, onda mu je on beskoristan.

Plug-in objekti se integrišu u veb stranu korišćenjem tagova **<embed>** ili **<object>**. Tag **<embed>** se koristi češće, a tag **<object>** je deo XHTML specifikacije. Embed tag koristi **src** atribut da specificira URL na kome se nalazi binarni objekt, kao što je audio ili video zapis. U sledećem primeru unutar veb strane se poziva film u **Audio Video Interleaved** (AVI) formatu koji se zove video.avi. Ovaj video se može pozivati iz HTML ili XHTML datoteke. Atributi **height** i **width** se koriste da se odredi veličina videa u pikselima.

```
<embed src="video.avi" height="150" width="150"></embed>
```

ACTIVE-X

ActiveX je Microsoftova tehnologija za kreiranje malih programskih komponenti ili kontrola unutar veb strane.

ActiveX je Microsoftova tehnologija za kreiranje malih programskih komponenti ili kontrola unutar veb strane. Kao i plug-insovi, tako su i ActiveX kontrole zavisne od operativnog sistema i potrebno je da se instaliraju, a mogu i automatski da se instaliraju. Veliki problem sa ActiveX kontrolama je sigurnost sistema, budući da one mogu da imaju pristup resursima klijentskog računarskog sistema pa mogu ozbiljno i da ga ugroze. Ova činjenica se zloupotrebljava za izradu virusa. Microsoft je uveo autentikacione informacije da bi se videlo ko je napisao kontrolu, ali to i dalje ne garantuje da je sistem zaštićen od prevare. Zbog toga ActiveX kontrole treba preuzimati samo sa sajtova koji imaju dobru reputaciju.

Za umetanje ActiveX kontrole se koristi **<object>** tag. koji ima sledeću opštu formu:

```
<object classid="clsid:class-identifier"  
height="pixels or percentage" width="pixels or percentage"  
id="unique identifier">  
Parameters and alternative text rendering  
</object>
```

Atributi **height** i **width** se koristeza definisanje veličine prozora u kome će se izvršavati ActiveX kontrola. Između tagova **<object>** i **</object>** se nalaze različiti parametri kojima se zadaju informacije koje se prenose kontroli, na primer, URL datoteke koju čita ActiveX kontrola.

ACTIVEX-PRIMER

Primer ActiveX kontrole

Primer 4 prikazuje ActiveX kontrole koje se mogu integrisati u veb stranu. Ovaj primer konkretno prikazuje integraciju flash datoteke.

Neki alati za razvoj veba, kao što su Macromedia Dreamweaver i Microsoft Visual Studio, automatski insertuju kontrole u stranu i pri tom specificiraju vrednosti parametara. Treba primetiti da i plug-insovi i ActiveX kontrole kada se jednom prevuku na klijentsku stranu ostaju stalno instalirani. Veb čitač prilikom interpretacije **object** taga najpre proverava da li je taj objekt već instaliran na sistemu, pa ga učitava jedino ako ga nema ili je ranije instalirana neka starija verzija.

Primer 4

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"  
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  
<html xmlns="http://www.w3.org/1999/xhtml" lang="en">  
<head>  
<title>ActiveX Demo</title>  
<meta http-equiv="content-type" content="text/html; charset=ISO-8859-1" />  
</head>  
<body>  
  
<h1 align="center">You should see a clock with moving second's hand</h1>  
<hr />  
  
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"  
       codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/  
swflash.cab#version=6,0,0,0"  
       id="flash1" name="flash1"  
       width="500" height="500">  
    <param name="movie" value="http://flash-clocks.com/extdocs/tzClock.swf" />  
    <param name="quality" value="high" />
```

```
</object>
</body>
</html>
```

JAVA APPLETS

Za programiranje klijentske strane Java koristi male programe koji se nazivaju apleti

Veliki nedostatak plug-insova i ActiveX kontrola je što su platformski zavisne. Nasuprot tome Java tehnologije su platformski nezavisne, jer dozvoljavaju da se napišu jednom i izvršavaju na bilo kojem procesoru i operativnom sistemu koji podržava Java virtuelnu mašinu (JVM). Većina operativnih sistema danas dolazi sa JVM. Za programiranje klijentske strane Java koristi male programe koji se nazivaju apleti. Oni su pisani u Javi i prevedeni u mašinski nezavisni bajtkod (byte-code) koji se interpretira od strane JVM. Po zatvaranju stranice koja ih je pozvala, apleti se brišu sa klijentskog sistema i tako ne opterećuju nepotrebno sistem. Ali, to znači da će prilikom njihovog ponovnog korišćenja biti potrebno da se ponovo prevuku na klijentsku stranu. Tehnologija apleta je danas podržana od strane svih najvažnijih veb čitača.

Najbolja osobina apleta je sigurnost njihovog korišćenja. Apleti se izvršavaju u veb čitaču i gotovo da nemaju pristupa resursima klijentske strane. Ako je takav postupak ipak neophodan, aplet će obavestiti korisnika kroz modalni dijalog o potrebnim koracima koje je neophodno izvršiti. Ukoliko korisnik to ne želi da uradi, aplet može nastaviti da radi, ali ta specifična funkcionalnost ostaje blokirana.

Primer 5 ilustruje Java aplet koji ispisuje poruku "Hello everyone" u veb čitaču. Kako bi se ovaj aplet ubacio u veb stranicu potrebno je da se koristi tag <applet> kao što je prikazano u primeru 6.

Primer 5

```
import java.applet.Applet;
import java.awt.Graphics;

public class hello extends Applet
{
    public void paint(Graphics graph)
    {
        graph.drawString("Hello everyone", 50, 50);
    }
}
```

PRIMER

Ubacivanje apleta u veb stranicu

Primer 6

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en">
<head>
<title>Java Applet Demo</title>
<meta http-equiv="content-type" content="text/html; charset=ISO-8859-1" />
</head>
<body>
<h1 align="center">Java applet Hello</h1>
<hr />
<applet code="hello.class"
        height="100" width="125">
</applet>
</body>
</html>
```

▼ Poglavlje 4

Programiranje sa serverske strane

RAZLOZI ZA PROGRAMIRANJE SA SERVERSKE STRANE

Kada je potrebno dodati interaktivnost veb strani ili kada je potrebna veća obrada podataka, onda je razumnije tu funkcionalnost ugraditi na serverskoj strani

Kada je potrebno dodati interaktivnost veb strani ili kada je potrebna veća obrada podataka, onda je razumnije tu funkcionalnost ugraditi na serverskoj strani, osim ako interaktivnost nije čisto klijentska po prirodi, kao što je, na primer, u slučaju animacije ili provere podataka u formi. Postoje dva razloga zašto se radi programiranje serverske strane:

- kada je potrebno da se ima potpuna kontrola, s obzirom da se jedino serverska strana može kompletno upravljati, što je posebno važno sa aspekta bezbednosti
- klijentska strana generalno nedefinisana jer se ne zna ni hardverska ni softverska platforma, a i instalirani veb čitači mogu biti različiti. To ograničava mogućnosti za ozbiljniju obradu podataka na klijentskoj strani.

Neki od primera kada je poželjno programirati sa serverske strane su:

- kada se obrađuje korisnikov unos
- kada je potrebno obaviti interakciju sa serverom ili njegovim skladištem
- kada se šalju upiti bazi podataka ili kada se ona ažurira

Za programiranje serverske strane se koriste različite tehnologije, kao što su:

1. **CGI programi**
2. **Serverski moduli - Web server API programi**
 - Apache moduli
 - Java servleti
3. **Server-side scripting**
 - Server-side Includes (.shtml)
 - ASP ili ASP.NET (.asp/.aspx)
 - ColdFusion (.cfm/.cfmx)
 - PHP (.php)
 - Java Server Pages (.jsp)

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

CGI

CGI je standardni protokol kojim se specificira kako se informacije prenose od veb čitača do veb servera, a zatim prenose na obradu nekom programu koji se izvršava na serverskoj strani

CGI je skraćenica za **Common Gateway Interface**. CGI je standardni protokol kojim se specificira kako se informacije prenose od veb čitača do veb servera, a zatim prenose na obradu nekom programu koji se izvršava na serverskoj strani i, obratno, kako se informacije koje generiše program na serverskoj strani prenose preko veb servera do veb čitača. Programi na serverskoj strani koji pomoću CGI protokola primaju i šalju informacije veb čitaču nazivaju se **CGI programi** iako su pisani u jezicima kao što su C++, Java, Python ili Perl. Tipičan scenario upotrebe CGI programa biće prikazan na primeru popunjavanja neke forme na veb stranici podacima koje treba proveriti i onda zapisati u bazu podataka na serverskoj strani.

Pri tom je sledeća procedura:

- Nakon popunjavanja podataka korisnik šalje formu veb serveru. Uz podatke iz forme klijentska strana šalje i podatke o svom radnom okruženju, kao što je na primer podatak o tipu veb čitača ili vremenu slanja.
- Transfer podataka do veb servera se obavlja HTTP protokolom.
- Veb server prima podatke i zahtev za obradu podataka i detektuje kom programu treba proslediti podatke na obradu.
- Veb server locira traženi program obično u nekom unapred definisanom direktorijumu, pokreće ga, a zatim mu prosleđuje podatke pristigne od korisnika.
- Program obrađuje podatke, eventualno ih zapisuje i po potrebi izveštava veb server o rezultatima obrade. Rezultati obrade mogu da budu generisani, na primer, kao HTML dokument, kako bi klijentska strana mogla da ih prikaže. Na ovaj način se praktično može kreirati bilo kakav dinamički sadržaj na vebu.
- Veb server prima informacije o obradi podataka od CGI programa.
- Veb server šalje podatke veb čitaču korišćenjem HTTP protokola.
- Veb čitač prikazuje primljenu poruku. U ovom slučaju to može biti potvrda o prihvatanju podataka ili zahtev za korekciju neispravno ili nepotpuno unetih podataka.

Na sličan način se mogu izvršavati različiti programi kojima se omogućuje da se sa klijentske strane izvrši autentikacija korisnika, pristupi podacima u nekoj bazi ili aktivira neka složena aplikacija za obradu podataka na serverskoj strani.

Nedostatak CGI programa je u tome što je cela procedura sa njima relativno dugotrajna. Naime, u prethodno opisanom scenariju klijentska strana treba da sačeka da veb server pronađe i startuje CGI program. Startovanje programa takođe zahteva neko vreme. Sve to dovodi do nedopustivo dugog odziva na klijentskoj strani.

VIDEO

Web technology tutorial: Server-side scripting | lynda.com

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ 4.1 Serverski moduli

WEB SERVER API

Jedno od rešenja za postizanje bržeg odziva serverske strane je da programi koji treba da obrađuju zahteve dobijene od klijentske strane budu moduli samog veb servera

Jedno od rešenja za postizanje bržeg odziva serverske strane je da programi koji treba da obrađuju zahteve dobijene od klijentske strane budu moduli samog veb servera. Ovakvi moduli se nazivaju **Web Server API** (Application Programming Interface) moduli. Različiti veb serveri koriste različite tipove modula. Tako Microsoft-ov veb server IIS koristi **Internet Server Application Programming Interface** (ISAPI) module, Apache serveri koriste Apache module itd. Ovi moduli se obično pišu u jezicima kao što su C ili C++ i odgovaraju API-u koga dati veb server koristi.

Ova tehnologija obezbeđuje veću brzinu odziva, ali ima i određene nedostatke. Pisanje API modula je mnogo složenije nego pisanje CGI programa. Osim toga CGI programi su nezavisni od vrste servera, dok se serverski API moduli pišu za svaki tip veb servera posebno.

VIDEO

What is REST API? | Web Service

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ 4.2 Skripte za serversku stranu

SKRIPTE ZA SERVERSKU STRANU (SERVER-SIDE SCRIPTING)

Server-side scripting je veb tehnologija koja omogućuje da se korisnički zahtev ispuni tako što se pokrene neki skript direktno na veb serveru i tako generiše dinamička HTML strana

Treći način programiranja serverske strane je korišćenje **skript jezika** (engl. **server-side scripting**). **Server-side scripting** je veb tehnologija koja omogućuje da se korisnički zahtev

ispuni tako što se pokrene neki skript direktno na veb serveru i tako generiše dinamička HTML strana. Uobičajeno je da veb strana svoj sadržaj ili delove sadržaja preuzme iz neke baze podataka, na primer, na osnovu upita koji je formiran na klijentskoj strani. Ova metoda omogućuje izradu složenih veb aplikacija na vrlo jednostavan način.

U osnovi metod se sastoji u tome da se u kreirane HTML ili XHTML strane umeću pozivi nekih rutina koje su napisane u nekom od skript jezika specijalno razvijenih za serversku stranu. Ove rutine mogu da budu različite akcije koje treba izvršiti na serverskoj strani, kao što je na primer čitanje nekog zapisa iz baze podataka ili umetanje tražene slike unutar HTML strane. Kada klijentska strana pošalje zahtev za čitanje jedne ovakve strane, veb server je najpre čita i izvršava uključene rutine shodno zahtevima klijentske strane. Na taj način veb server dinamički generiše sadržaj HTML strane. Tek tada se ovako generisana HTML strana šalje veb klijentu.

Da bi se pokazalo da neka strana sadrži skriptove menja se sufiks u njenom imenu shodno vrsti skript jezika koji je korišćen, na primer, ime.php za stranice koje uključuju php skript.

Jedini nedostatak ove tehnologije je što serverska strana uvek mora da interpretira skriptove kako bi generisala stranicu koju šalje veb klijentu. Ovo može da izazove zagruženje veb servera.

Najčešće korišćeni skript jezici za programiranje na serverskoj strani su **Server-Side Includes (SSI)**, PHP, **ColdFusion**, **Active Server Pages (ASP)**, VBScript.

SERVER-SIDE INCLUDES

Osnovni cilj ovog skript jezika je da sadržaj jednog veb dokumenta uključi u drugi

Server-Side Includes (SSI) najjednostavniji je skript jezik za programiranje na serverskoj strani. Kao što mu samo ime govori, osnovni cilj ovog skript jezika je da sadržaj jednog veb dokumenta uključi u drugi. SSI su kratke direktive koje mogu da se umetnu u HTML dokument da bi ukazale koje datoteke treba da se pročitaju i uključe u finalni izlaz koji se šalje ka veb klijentu. Na taj način se, na primer, mogu specificirati standardna zaglavila i futeri veb stranica, a zatim po potrebi uključivati u druge HTML stranice.

Na primer, potrebno je da se na dnu svake stranice uključi poruka o autorskim pravima. Ovo se može opisati u nekom HTML dokumentu koji će biti zapisan kao **futer.html** (Primer 7).

Primer 7

```
<hr noshade="noshade" />
<div align="center">
<font size="-1">
Copyright 2020, Fakultet Informacionih Tehnologija.<br/>
</font>
</div>
```

Kako bi se ubacio ovaj futer u HTML stranicu može se napisati:

```
<!--#include file="futer.html" -->
```

Stranice koje uključuju SSI skript imaju ekstenziju imena *.shtml ili ređe *.shtm. SSI direktive se uključuju u veb stranicu kao HTML komentar i u opštem slučaju imaju sledeći oblik:

```
<!--#directive parameter=value parameter=value-->.
```

Uglavnom se koriste samo dve direktive **include** i **exec**. Include umeće sadržaj neke HTML datoteke u dokument iz koga se poziva. Exec poziva neki program, skriptu ili shell komandu na serveru.

PHP

PHP je scripting jezik koji je napravljen za dinamičko generisanje veb strana na serverskoj strani

PHP je scripting jezik koji je napravljen za dinamičko generisanje veb strana na serverskoj strani. Prvobitno, skraćenica PHP bila skraćenica za **Personal Home Page**. Današnje zvanično ime za ovaj skriptni jezik je PHP: **Hypertext Preprocessor**. PHP je originalno razvijen za UNIX - Linux platforme ali sada postoje i verzije za Windows i Mac OS X operativne sisteme. Snaga ovog jezika leži pre svega u njegovoj jednostavnosti i mogućnosti da se poveže sa velikim brojem relacionih baza podataka, kao što su MySQL, Oracle, IBM DB2, Microsoft SQL Server itd. Pored toga, programerima koji koriste PHP na raspolažanju je ogroman broj javnih biblioteka pomoću kojih se bitno ubrzava proces pisanja aplikacija. Sve ovo je učinilo PHP jednim od najpopularnijih skript jezika, što je rezultiralo i njegovom podrškom od strane svih značajnijih veb servera. Specifična osobina PHP-a je da se tip promenljivih određuje dinamički, što znači da se one ne moraju deklarisati i da se mogu odnositi na bilo koji tip objekta.

Radi ilustracije korišćenja PHP skripta daje se primer ispisivanja jednostavne poruke (Primer 8). Kod PHP skripta odvojen je od ostatka dokumenta delimitirima **<?php** i **?>**. Ako se ovakav skript zapiše kao datoteka sa ekstenzijom php, onda će veb server koji interpretira php skript generisati adekvatnu HTML stranicu i poslati je veb klijentu.

Primer 8

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en">
<head>
<title>PHP Script</title>
<meta http-equiv="content-type" content="text/html; charset=ISO-8859-1" />
</head>
<body>
<?php
    print("This was displayed by PHP script!");
?>
```

```
</body>  
</html>
```

ACTIVE SERVER PAGES

Sa ASP-om je moguće kombinovati HTML, skript kod i ActiveX komponente namenjene serverskoj strani radi kreiranja dinamičkih veb stranica

Active Server Pages ([ASP](#)) i novija verzija ASP.NET je Microsoft-ov skript jezik pisan da se izvršava pre svega na Microsoft-ovom [Internet Information Server-u](#) (IIS). Sa ASP-om je moguće kombinovati HTML, skript kod i ActiveX komponente namenjene serverskoj strani radi kreiranja dinamičkih veb stranica. ASP prihvata skript kod pisan u različitim skript jezicima, kao što su VBScript, JavaScript, Perl i drugi. Skript se umeće u ASP stranice koje se zapisuju kao *.asp ili *.aspx u slučaju ASP.NET-a. Veb server koji ima mogućnost rada sa aktivnim serverskim stranicama čita ASP dokumente, interpretira skript u njima i šalje tako dobijenu veb stranicu klijentskoj strani.

▼ Poglavlje 5

Veb serveri

ŠTA JE VEB SERVER?

Veb server je računar kome može da se pristupi preko Interneta i može da odgovori na HTTP zahteve dobijene od klijenata – veb čitača

Da bi se neki veb sajt stavio u funkciju potrebno je formirati veb server. Veb server je računar kome može da se pristupi preko Interneta i može da odgovori na HTTP zahteve dobijene od klijenata – veb čitača. Pored toga, veb server ima i funkciju fajl servera, jer su na njemu smešteni i dokumenti koji se publikuju na vebu, kao što su HTML i XML strane, skriptovi i druga dokumenta i podaci. Softver koji radi na veb serveru koji odgovara na zahteve veb klijenata takođe se naziva veb server. Njegov zadatak je da pronađe traženi dokument na serveru, da ga po potrebi interpretira kako bi od skripta napravio HTML stranu i da to prosledi klijentskoj strani. Pri tom ne treba zaboraviti da veb server treba da ima mogućnost da istovremeno opslužuje veliki broj klijenata.

U 2012-oj godini najčešće korišćeni veb serveri su Apache i Microsoft. Prema Netcraft-ovojoj statistici iz jula 2012 Apache je bio instaliran na 61.45% od svih server na internetu, a Microsoft na 14.62%. U Tabeli 1 se mogu videti uporedne zastupljenosti servera uporedno iz 2007 i 2012 godine. Ukoliko vas interesuje trenutna zastupljenost, kao i promene tokom godina, možete pogledati statistiku koju Netcraft redovno objavljuje svakog meseca.

Apache veb server svoju popularnost duguje tome što je vrlo brz i spada u kategoriju slobodnog softvera. Apache podržava SSI, skript jezike kao što su PHP, Perl i Python i popularne module za autentikaciju korisnika. Mada je inicijalno napisan za UNIX operativne sisteme, danas postoje verzije za sve značajne operativne sisteme, uključujući i Windows.

Microsoft-ov **Internet Information Server** - IIS se koristi samo uz operativne sisteme Windows, što mu znatno smanjuje šanse da dominira tržištem. Pored toga, IIS je vrlo tesno integriran sa operativnim sistemom, tako da neki nedostaci operativnog sistema utiču i na njegove performanse. Uglavnom se koristi za komercijalne veb sajtove.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

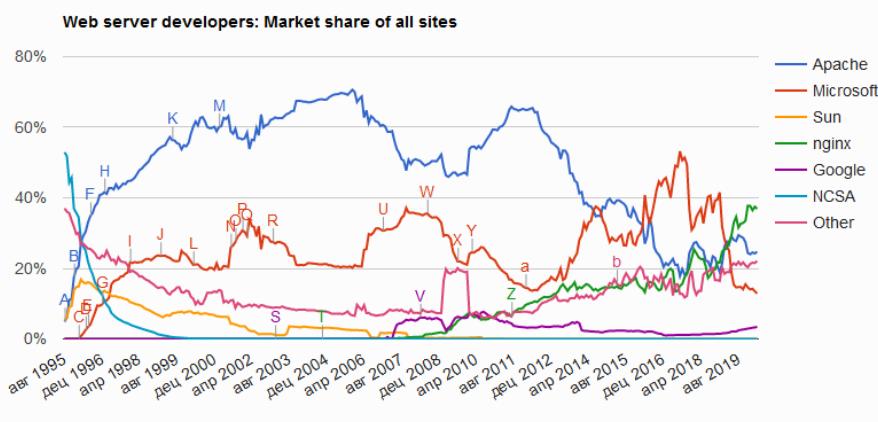
ZASTUPLJENOST SERVERA NA TRŽIŠTU

Netcraft-ova analiza

Ime	Mart 2020	%	April 2020	%	Promena
nginx	473,308,955	37.47%	459,886,788	36.91%	-0.57
Apache	306,114,673	24.24%	308,143,708	24.73%	0.49
Microsoft	170,567,386	13.50%	160,121,865	12.85%	-0.66
Google	41,227,959	3.26%	42,648,748	3.42%	0.16

Slika 5.1 Tabela-1 Zastupljenost servera na tržištu

. Netcraft, <http://news.netcraft.com/archives/2020> (Last accessed 29.05.2020).



Slika 5.2 Netcraft-ova analiza zastupljenosti servera na tržištu

▼ Poglavlje 6

Pokazna vežba: Upotreba JavaScript-a u HTML-u

FUNKCIJE

Funkcija se poziva navođenjem imena funkcije i prosleđivanje stvarnih parametara u zagradama

Predviđeno vreme pokazne vežbe je 100 minuta.

Često je potrebno da neku funkcionalnost koja se ponavlja u programu izdvojimo u celinu koja se može koristiti iz ostatka programa. Za ovo se koriste funkcije. Sintaksa za definisanje funkcije je sledeća:

```
function ime_funkcije (spisak_parametara) {  
    Telo funkcije...  
}
```

Parametri funkcije su varijable koje se u funkciju prosleđuju prilikom pozivanja funkcije i mogu se koristiti u telu funkcije kao obične varijable. Funkcija može vratiti vrednost kalkulacije koristeći ključnu reč return.

Funkcija se poziva navođenjem imena funkcije i prosleđivanje stvarnih parametara u zagradama. Povratna vrednost funkcije se može dodeliti varijablama na uobičajni način. Na primer:

```
zbir = saberi(2, 5);
```

Posle pozivanja funkcije, izvršavanje koda se zaustavlja na liniji na kojoj je funkcija pozvana, i onda tok programa „skače“ u telo funkcije. Posle izvršavanja tela funkcije, nastavlja se izvršavanje prethodnog bloka koda. Moguće je pozivati funkcije iz drugih funkcija, gde je jedino praktično ograničenje broja poziva veličina stack memorije.

JAVASCRIPT KOD U HTML DOKUMENTU

*U HTML dokumentu JavaScript mora da se nađe između tagova
<script> </script>*

U HTML dokumentu JavaScript mora da se nađe između tagova **<script> </script>**

Takođe JavaScript može da se nađe u okviru **<head>** ili **<body>** sekcija.

Kada želimo da ispišemo neki tekst u HTML dokument onda se koristi **document.write**. U prvom primeru ispisujemo tekst sa *document.write* koja ispisuje string *Hello World!*

```
<!DOCTYPE>
<html>
  <body>
    <script type="text/JavaScript">
      document.write("Hello World!")
    </script>
  </body>
</html>
```

PROGRAMSKO ISPISIVANJE TEKSTA U DOKUMENT

U niže navedenim primerima prvi korak je da obavestimo veb čitač da će biti korišćena skripta navodjenjem taga <script>.

U sledećem primeru ispisujemo tekst u delu **<body>** prilikom prikazivanja stranice. U niže navedenim primerima prvi korak je da obavestimo veb čitač da će biti korišćena skripta navodjenjem taga **<script>**.

```
<!DOCTYPE>
<html>
  <body>
    <script>
      document.write("<h1>SERVER/KLIJENT VEŽBE</h1>");
      document.write("<h2>Rezultat primera skripte u BODY tagu</h2>");
      document.write("<p>Ispisivanje teksta paragrafa...</p>");
    </script>
  </body>
</html>
```

SERVERT/KLIJENT VEŽBE

Rezultat primera skripte u BODY tagu

Ispisivanje teksta paragrafa...

Slika 6.1 Rezultat izvršavanja skripte

PRIMER 1: UPISIVANJE TEKSTA U DOKUMENT

InnerHTML property se koristi za postavljanje sadržaja HTML elemenata

Vreme predviđeno za izradu sledećeg primera je 10 minuta.

U sledećem primeru Javascript funkcija se stavlja u delu HEAD taga, a koja se izvršava kada se klikne na dugme. **innerHTML** je funkcionalnost koja se koristi za interaktivnost sa korisnikom. Kako bi se on koristio potrebno je da prvo elementu koji želite da dodelite ime odnosno **id**, a kasnije sa tim *id*-em možete koristiti funkciju **getElementById** koja radi na svim veb čitačima.

```
<!DOCTYPE>
<html>
  <head>
    <script type="text/javascript">
      function izmenaTeksta(){
        document.getElementById('boldovano').innerHTML = 'na IT101 Osnove
informacionih tehnologija';
      }
    </script>

  </head>
  <body>
    <p>Dobrodošli <b id='boldovano'>budući IT-evci</b> </p>
    <input type='button' onclick='izmenaTeksta()' value='Klikni i izmeni tekst'/>

  </body>
</html>
```

REZULTAT PRIMERA 1

Klikom na dugme u dokument se upisuje dati tekst.

Rezultat prilikom učitavanja stranice:

Dobrodošli **buduci IT-evci**

Klikni i izmeni tekst

Slika 6.2 Učitana stranica

Rezultat nakon klika na dugme “Klikni i izmeni tekst”:

Dobrodošli na IT101 Osnove informacionih tehnologija

Klikni i izmeni tekst

Slika 6.3 Stranica posle klika na dugme

PRIMER 2: IZMENA TEKSTA U DOKUMENTU

Korigujte prethodni primer tako što ćete izmeniti tekst koji se ispisuje na početku, tekst dugmeta, kao i tekst koji zamenjuje početni tekst nakon klika na dugme.

Vreme predviđeno za izradu sledećeg primera je 10 minuta.

ZADATAK: Korigujte prethodni primer tako što ćete izmeniti tekst koji se ispisuje na početku, tekst dugmeta, kao i tekst koji zamenjuje početni tekst nakon klika na dugme.

```
<!DOCTYPE>
<html>
<head>
<script>
    function primerFunkcije(){
        document.getElementById("IT101").innerHTML="Tekst koji će se ispisati nakon
        klika na dugme";
    }
</script>
</head>
<body>
    <h1>IT101 - Primer</h1>
    <p id="IT101">Tekst koji se ispisuje prilikom pokretanje stranice, a koji će
    biti zamenjen nakon klika na dugme </p>
    <button type="button" onclick="primerFunkcije ()">Probaj</button>
</body>
</html>
```

REZULTAT PRIMERA 2

Klikom na dugme u dokumentu će se izmeniti već upisani tekst.

Prikaz prilikom učitavanja stranice:

IT101 - Primer

Tekst koji se ispisuje prilikom pokretanje stranice, a koji će biti zamenjen nakon klika na dugme

Probaj

Slika 6.4 Prikaz prilikom učitavanja stranice

Rezultat nakon klika na dugme "Probaj":

Slika 6.5 Rezultat nakon klika na dugme

PRIMER 3: IZMENA TEKSTA

Korigujte prethodni primer tako što ćete zameniti tekst "IT101-Primer" sa tekstrom "IT101 - Dorada primera", a tekst nakon klika na dugme sa vašim imenom i prezimenom

Vreme predviđeno za izradu sledećeg primera je 10 minuta.

ZADATAK: Korigujte prethodni primer tako što ćete zameniti tekst "IT101-Primer" sa tekstrom "IT101 - Dorada primera", a tekst "Tekst koji će se ispisati nakon klika na dugme" sa vašim imenom i prezimenom.

```
<!DOCTYPE>
<html>
  <head>
    <script type="text/javascript">
      function izmeniTekst(){
        var userInput = document.getElementById('userInput').value;
        document.getElementById('boldovano').innerHTML = userInput;
      }
    </script>
  </head>
  <body>
    <p>Dobrodošao <b id='boldovano'>na IT101</b> </p>
    <input type='text' id='userInput' value='Unesite tekst ovde' />
    <input type='button' onclick='izmeniTekst()' value='Klikom na dugme se menja
tekst' />
  </body>
</html>
```

PRIMER 4: UKLANJANJE ELEMENTA IZ DOKUMENTA

Izmenite prethodni primer tako da se nakon klika na dugme izmeni tekst, a da se pri tom ne prikazuje više polje za unos teksta, kao ni dugme.

Vreme predviđeno za izradu sledećeg primera je 10 minuta.

ZADATAK: Izmenite prethodni primer tako da se nakon klika na dugme izmeni tekst, a da se pri tom ne prikazuje više polje za unos teksta, kao ni dugme.

```
<!DOCTYPE>
<html>
<head>
<script type="text/javascript">
    function izmeniTekst(){
        var oldHTML = document.getElementById('tekstParagrafa').innerHTML;
        var newHTML = "<span style='color:#ff0fff'>" + oldHTML + "</span>";
        document.getElementById('tekstParagrafa').innerHTML = newHTML;
    }
</script>
</head>
<body>
    <p id='tekstParagrafa'>IT101 <b id='boldovano'>Osnove IT</b> </p>
    <input type='button' onclick='izmeniTekst()' value='Izmeni tekst' />
</body>
</html>
```

PRIMER 5: KORIŠĆENJE RANDOM FUNKCIJE

Prikaz korišćenja random funkcije

Vreme predviđeno za izradu sledećeg primera je 10 minuta.

Random funkcija vraća brojeve između 0 i 1. Sintaksa funkcije je Math.random().

```
<!DOCTYPE>
<html>
<head>
    <title>Primer</title>
<script>
    document.getElementById("msg").innerHTML = Math.random( 1, 100 );
</script>
</head>
<body>
    <p id="msg"></p>
    <p>funkcija <code>Math.random()</code> vraca brojeve izmedju 0 (inclusive) i 1
(exclusive).</p>
```

```
</body>  
</html>
```

0.0392076291865906
funkcija `Math.random()` vraca brojeve izmedju 0 (inclusive) i 1 (exclusive).

Slika 6.6 Izvršenje HTML-JS fajla

KORIŠĆENJE SKRIPTE IZ POSEBNOG FAJLA

U praksi je bolje kada se skripta piše van samog HTML dokumenta tako da se skripta može koristiti više puta i na više stranica

U praksi je bolje kada se skripta piše van samog HTML dokumenta tako da se skripta može koristiti više puta i na više stranica. HTML dokument ima sledeći sadržaj u kome je naznačeno ime datoteke mojJavaScript.js koja ustvari sadrži skriptu

```
<!DOCTYPE>  
<html>  
  <head>  
    <script src="mojJavaScript.js">  
    </script>  
  </head>  
  <body>  
    <input type="button" onclick="popup()" value="Click Me!">  
  </body>  
</html>
```

Sadržaj mojJavaScript.js dokumenta:

```
function popup() {  
  alert("Hello World")  
}
```

Važno je napomenuti da se eksterna .js datoteka može koristiti i kada se JavaScript koristi u head ili u body tagovima. Međutim, ne treba zaboraviti da **<script>** tagove ne treba stavljati u .js datoteku, već ona ide u HTML datoteku.

GRANANJE U PROGRAMU

Za grananje logike u programu koristi se if else konstrukcija

Logika if i else if iskaza u JavaScript-u je slična kao i u drugim programskim jezicima. Pogledajmo primer ako imamo slučaj da želite da se prikaže različita poruka u odnosu na to ko posećuje vaš sajt. Na primer, ako želite da se ispiše na vašem sajtu dobrodošlica posetiocu u zavisnosti od vremena kada posećuje sajt može se koristiti sledeća skripta

```
<!DOCTYPE>
<html>
  <body>
    <p>Kliknite na dugme da dobijete pozdrav u zavisnosti od trenutnog vremena</p>
    <button onclick="vremePozdrav()">Probaj</button>
    <p id="demo"></p>
    <script>
      function vremePozdrav(){
        var x="";
        var time=new Date().getHours();
        if (time<12){
          x="Dobro jutro";
        }
        else if(time>=12 && time<18){
          x="Dobar dan";
        }
        else{
          x="Dobro veče";
        }
        document.getElementById("demo").innerHTML=x;
      }
    </script>
  </body>
</html>
```

ARTIMETIČKI OPERATORI

*JavaScript čita sve korisničke unose kao string, zbog toga treba koristiti **parseInt()** kako bi se string pretvorio u broj. Može se koristiti i **parseFloat()**.*

Operatori koji se koriste u JavaScript su veoma slični kao i operatori koji se koriste u drugim programskim jezicima. Najčešće korišćeni aritmetički operatori su operatori za sabiranje (+), oduzimanje (-), množenje (*), deljenje (/) i moduo (%).

Napomena: JavaScript čita sve korisničke unose kao string, zbog toga treba koristiti **parseInt()** kako bi se string pretvorio u broj. Može se koristiti i **parseFloat()**.

```
<!DOCTYPE>
<html>
<head>
<script type="text/javascript">
function unetiBrojeve(){
var m, n, rezultat;

m = document.getElementById('broj1').value;
n = document.getElementById('broj2').value;

rezultat = parseInt(m)+parseInt(n);

document.getElementById("rezultat").innerHTML = 'Rezultat: ' + rezultat;

}
</script>
</head>
<body>
<p>Unesite brojeve u polja: <b> </b> </p>
<input id='broj1' />
<input id='broj2' />

<p id="rezultat"> Rezultat?</p>
<input type='button' onclick='unetiBrojeve()' value='Saberि' />

</body>
</html>
```

OPERATORI POREDJENJA

Operatori za upoređivanje vraćaju boolean vrednosti

Takođe treba spomenuti i druge operatore koji se često koriste, a koji se koriste za upoređivanje, kao što su jednako(==), nije jednako(!=), manje od(<), veće od(>), manje ili jednako od (<=), veće ili jednako od (>=). Operatori za upoređivanje vraćaju boolean vrednosti.

```
<!DOCTYPE>
<html>
<head>
<script type="text/javascript">
function unetiBrojeve(){
var m, n, rezultat;

m = document.getElementById('broj1').value;
n = document.getElementById('broj2').value;

rezultat = parseInt(m)==parseInt(n);

if(rezultat==true)
```

```
        rezultat='brojevi su isti';
    else
        rezultat='brojevi nisu isti';

    document.getElementById("rezultat").innerHTML = 'Rezultat: ' + rezultat;

}
</script>

</head>

<body>
<p>Unesite brojeve u polja: </b> </p>
<input id='broj1' />
<input id='broj2' />

<p id="rezultat"> Rezultat?</p>
<input type='button' onclick='unetiBrojeve()' value='Uporedite da li su brojevi
jednaki' />

</body>
</html>
```

POKAZNI PRIMERI

Pokazni primeri sa rešenjem

Vreme predviđeno za izradu sledećeg zadatka je 5 minuta.

1. Zadatak

Korišćenje JavaScript-a učitavanjem eksternog JavaScript fajla - Napisati primer JavaScript fajla koji se učitava u okviru HTML strane.

Rešenje:

Primer.js sadrži sledeći kod:

```
document.write("Ova skripta je eksterna")
```

Primer.html je potrebno da sadrži sledeći kod

```
<html>
<head>
</head>
<body>
<script src="Primer.js">
</script>
<p>Ova skripta je eksterni JS file Primer.js</p>
</body>
</html>
```

Vreme predviđeno za izradu sledećeg zadatka je 10 minuta.

2. Zadatak

Napisati funkciju u JavaScript-u koja izračunava faktorijel unetog broja.

Rešenje:

```
<html>
<head>
<script>
function show(){

var i, no, fact;
fact=1;
no=Number(document.getElementById("num").value);
for(i=1; i<=no; i++)
{
fact= fact*i;
}
document.getElementById("answer").value= fact;
}
</script>
</head>
<body>
Unesite vrednost: <input id="num">
<button onclick="show()">Factorial</button>
<input id="answer">
</body>
</html>
```

PRIMERI FUNKCIJA

Primer JavaScript funkcije

Vreme predviđeno za izradu sledećeg primera je 10 minuta.

Zadatak: Napisati program koji će da pretvori početno slovo svake zdate reči u veliko slovo.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width">
<title>JavaScript</title>
</head>
<body>

</body>
</html>
```

Java Script funkcija:

```
function capital_letter(str)
{
    str = str.split(" ");

    for (var i = 0, x = str.length; i < x; i++) {
        str[i] = str[i][0].toUpperCase() + str[i].substr(1);
    }

    return str.join(" ");
}

console.log(capital_letter("Write a JavaScript program to capitalize the first
letter of each word of a given string."));
```

▼ Poglavlje 7

Zadatak za samostalni rad: JS u HTML

ZADACI ZA SAMOSTALNI RAD

JS program

Predviđeno vreme za izradu sledećih zadataka je 35 minuta.

- a) Napisati JS program koji će odrediti prosečnu vrednost brojeva u nizu, a onda ispisati koliko članova niza je veće i manje od prosečne vrednosti. (Vreme izrade: 5 minuta)
- b) Napisati program koji će ispisati brojeve koji se ponavljaju u nizu. (Vreme izrade: 10 minuta)
- c) Kreirati program koji će između brojeva od 1 do 10 da vrati random vrednost. (Vreme izrade: 5 minuta)
- d) Koristeći HTML i JS, kreirati input polje i jedno dugme. Nakon unošenja vrednosti i klika na dugme, potrebno je uzeti vrednost i upisati u OL listu. (Vreme izrade: 15 minuta)

✓ Poglavlje 8

Domaći zadatak

OPIS DOMAĆEG ZADATKA - DZ10

Napraviti HTML stranicu sa Javascript-om u eksternoj datoteci

Očekivano vreme izrade zadatka: 60min.

Napraviti HTML stranicu sa Javascript-om u eksternoj datoteci koja će imati sledeći sadržaj i funkcionalnosti:

- Naslov dokumenta će sadržati vaše ime, prezime i broj indeksa
- Stranicu stilizovati bojom
- Podnaslov će sadržati šifru i naziv predmeta
- Veličina fonta podnaslova će sadržati poslednje dve vrednosti broja Vašeg indeksa
- Omogućite unos za sve vaše predispitne obaveze iz predmeta (15 domaćih zadataka, 5 testova, projektni zadatak i ukupan broj negativnih poena)
- Na dnu će se nalazi dugme "Proveri uslov"
- Kada se klikne na dugme, potrebno je da skripta proveri da li imate uslov za izlazak na ispit (minimalno 35 poena na svim predispitnim obavezama)
- Kao rezultat, bez brisanja svog unosa, na dnu treba ispisati:

Ime i prezime - ima uslov (boldovano i plavom bojom) za izlazak na ispit (italic) - ukoliko student ima uslov za izlazak na ispit

ili Ime i prezime - nema uslov (boldovano i crvenom bojom) za izlazak na ispit (italic) - ukoliko nema uslov za izlazak na ispit

Ispod dugmeta za proveru uslova napraviti još jedno polje gde možete uneti broj poena sa ispita i ispod njega dugme koje izračunava konačnu ocenu sa nazivom "Izračunaj ocenu".

Rezultat klika na dugme "Izračunaj ocenu" daje ocenu (bez brisanja svega prethodno unetog) po sledećoj stavkama:

- Do 50 poena - ocena 5
- 51-60 poena - ocena 6
- 61-70 poena - ocena 7
- 71-80 poena - ocena 8
- 81-90 poena - ocena 9
- 91-100 poena - ocena 10

Krajnji prikaz ocene obojiti bojom po želji.

Zadatak dostaviti kao arhivu **IT101-DZ10-Ime_Prezime_BrojIndeksa.zip**. Domaći zadatak pošaljite predmetnom asistentu na e-mail.

▼ Zaključak

ZAKLJUČAK

Na ovom predavanju bilo je reči o programiranju serverske i klijentske strane. Ono što je najosnovnije da treba shvatiti na ovom predavanju jesu prednosti i mane korišćenja skripting jezika sa serverske i klijentske strane. Potrebno je da svaki IT profesionalac zna gde je najefikasnije da se obave određene funkcionalnosti kako bi sistemi bili što efikasniji i kako se ne bi ugrozia njihova bezbednost. U predavanju je dat i pregled najčešće korišćenih skripting jezika, kao i njihovi primeri koje je potrebno znati razumeti.



IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

ISTORIJA RAČUNARSTVA

Lekcija 11

PRIRUČNIK ZA STUDENTE

IT101 - OSNOVE INFORMACIONIH TEHNOLOGIJA

Lekcija 11

ISTORIJA RAČUNARSTVA

- ▼ ISTORIJA RAČUNARSTVA
- ▼ Poglavlje 1: Period pre pojave prvih elektronskih računara
- ▼ Poglavlje 2: Period pojave prvih računara
- ▼ Poglavlje 3: Generacije Računara
- ▼ Poglavlje 4: Istorija razvoja programskih jezika
- ▼ Poglavlje 5: Prva generacija programskih jezika - mašinski jezici
- ▼ Poglavlje 6: Pokazna vežba: JavaScript objekti
- ▼ Poglavlje 7: Zadaci za samostalni rad: Forme, nizovi
- ▼ Poglavlje 8: Domaći zadatak
- ▼ Zaključak

Copyright © 2017 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2017 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

✓ Uvod

UVOD

U ovoj lekciji ćemo dati istorijski pregled razvoja računara i programskih jezika

U ovom predavanju biće reči o razvoju informacionih tehnologija kroz istoriju. Biće reči o sledećim temama:

- Period pojave prvih računara
- Razvoj hardverskih platform
- Razvoj računarskih jezika

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

▼ Poglavlje 1

Period pre pojave prvih elektronskih računara

ISTORIJA HARDVERA

Reč „computer“ u originalu označava osobu koja rešava jednačine

Retke su tehnologije u kojima su promene toliko brze kao u računarstvu. Želja da se ovlada tržištem nameće potrebu da se sa novim rešenjima, koja u nekim slučajevima i nisu tako uspešna, izade što pre. U uslovima ovakve presje nema vremena za istoriju. Ipak, istorija računarstva je važna jer pokazuje kojim putevima su činjeni pokušaji da se dođe do rešenja koja se danas koriste, kao i zašto su neka rešenja bila neuspešna i odbačena. Istovremeno, istorija računarstva pomaže da se u budućem radu izabere pravi put a ne stranputica.

Ljudi su od uvek želeli da računaju brže. Paralelno sa razvojem novih matematičkih disciplina radilo se i na razvoju uređaja koji će automatizovati proces računanja i time ga učiniti bržim i tačnijim. Za razliku od drugih pronalazaka za koje se tačno zna ko ih je i kada izumeo, za računare se ne može imenovati samo jedna osoba kao pronalazač, jer se do prvih računara došlo evolutivnim putem u kome su učestvovali mnogi istraživači koji su dali svoj doprinos. Ali, kao što je računarska tehnologija evoluirala, tako se i menjala namena računarskih sistema. Prvi računari su projektovani sa idejom da pomognu istraživačima u rešavanju kompleksnih matematičkih problema, tj. rešavanju sistema diferencijalnih jednačina.

Reč „computer“ u originalu označava osobu koja rešava jednačine, pa su računari tako dobili ime koje se i danas koristi. Iako sadašnji računari mogu da se koriste i za rešavanje jednačina, oni se danas najmanje koriste za to. Računarski sistemi se danas koriste za čuvanje, obradu i pretraživanje podataka, upravljanje računarskim mrežama, komuniciranje, obradu teksta, grafike, audio i video materijala, modeliranje proizvoda i procesa, stvaranje umetničkih dela, zabavu itd. Većina današnjih korisnika nikada nije rešavala sistem jednačina na računaru, ali nabavlja i koristi „kompjutere“.

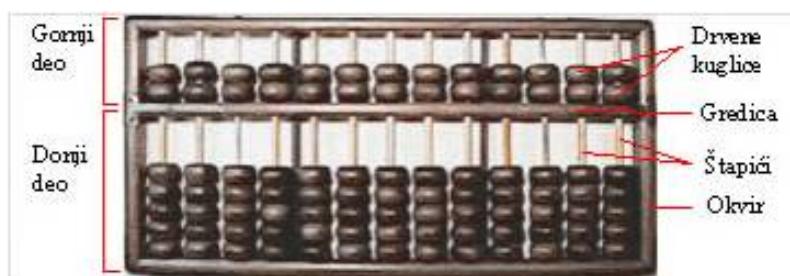
▼ Poglavlje 2

Period pojave prvih računara

ABAKUS I ŠIBER

Šiber je uređaj koji se sastojao od dva pokretna lenjira i omogućavao množenje dva realna broja i druge matematičke operacije

Jedan od prvih uređaja koji se koristio za računanje je bio **abakus**. Najstariji primerci otkriveni u Aziji potiču iz perioda 3000. – 2500. godine pre nove ere, ali su se slični uređaji koristili i u Grčkom i Rimskom carstvu. U nekim delovima Azije, abakus se i danas koristi.

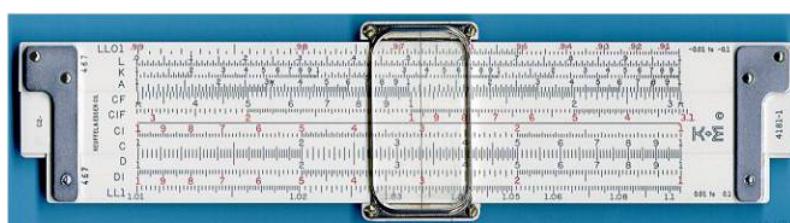


Slika 2.1 Abakus

Rad abakusa se zasnivao na poznavanju pozicionog brojnog sistema. Sastoje se od više štapića po kojima se mogu pomerati drvene kuglice. Svaki štapić ima svoju težinu (1, 10, 100, ..) koja odgovara njegovoj poziciji. Kuglice u donjem delu imaju vrednost 1, a u gornjem delu 5. Neki broj se predstavlja tako što se odgovarajući broj kuglica iz gornjeg i donjeg dela približi centralnoj gredici. Vrednost broja odgovara sumi proizvoda vrednosti kuglica i težina koje odgovaraju pojedinim štapićima.

Muhammad ibn Musa Al'Khwarzimi, taškentski sveštenik je u osmom veku razvio koncept pisanih procesa koji treba slediti da bi se postigao neki cilj i objavio knjigu koja je u naslovu sadržala i reč **al-jabr** od koje je kasnije nastao moderan izraz **algebra**.

Mnogo kasnije, negde oko 1620. godine englez Edmund Gunter izumeo je šiber, uređaj koji se sastojao od dva pokretna lenjira i omogućavao množenje dva realna broja i druge matematičke operacije. Šiberi su se koristili sve do sredine 70-tih godina prošlog veka kada su se pojavili elektronski kalkulatori.



Slika 2.2 Šiber

CHARLES BABBAGE - DIFFERENCE ENGINE, ANALYTICAL ENGINE

Analitička mašina se smatra pretečom današnjeg računara, jer je projektovana da nađe rešenje matematičkog izraza za koji se zna redosled operacija

Engleski mašinski inženjer Čarls Bebidž(Charles Babbage, 1792-1871) je projektovao 1821. godine prvi mehanički računar - **Difference Engine** - kojim su upravljale spoljašnje naredbe. Diferencijalna mašina je zamišljena za računanje četiri aritmetičke radnje: sabiranje, oduzimanje, množenje i deljenje. Zbog nedovoljne preciznosti mašinske obrade računar se loše pokazao na testu 1833.godine, ali je Bebidž nastavio sa usavršavanjem sve do 1842. godine. Tek je 1991. *National Museum of Science and Technology* izgradio radni model **Difference Engine**.



Slika 2.3 Difference Engine

. An Illustradet History of Computers, <http://www.computersciencelab.com/ComputerHistory/HistoryPt2.htm>

Međutim Čarls Bebidž se ne predaje i 1934. godine ulazi u novi projekat pod nazivom **Analiticcal Engine**. Njegova ideja je bila da napravi mašinu koja će moći da ima mogućnost „memorisanja“ programa, pa čak i uslovna grananja u programu. Zbog toga se Bebidž smatra ocem računarstva. Analitička mašina se smatra pretečom današnjeg računara, jer je projektovana da nađe rešenje matematičkog izraza za koji se zna redosled operacija pomoću kojih taj izraz može biti rešen, što je kasnije dobilo naziv algoritam. Probni model Analitičke mašine je završen 1971. godine, ali Bebidž onda umire.

Njegova koleginica, Ada Bajron Lavlejs (Augusta Ada Byron, the Countess of Lovelace, 1815-1852) čerka lorda Bajrona, se potpuno upoznala sa radom analitičke mašine i javno publikovala rad o tome. Ovaj rad se smatra prvi koji opisuje programiranje jednog računara. Računarska istorija priznaje Adu za prvog programera u istoriji, a jedan od savremenih računarskih jezika „**Ada**“ nosi ime po njoj.



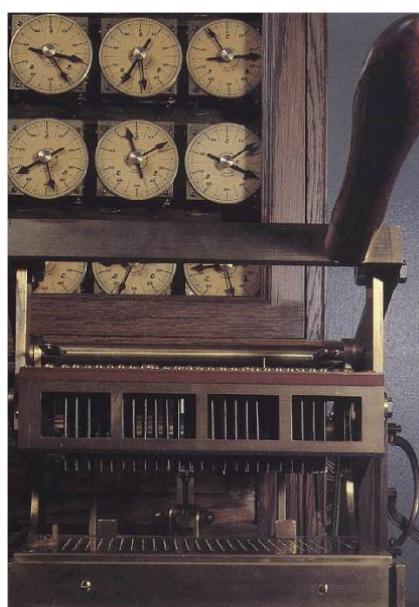
Slika 2.4 Analytical engine

. Encyclopedia, <http://www.pcmag.com/encyclopedia/term/37763/analytical-engine>

DŽORDŽ BUL, HERMAN HOLERIT, KARL BRAUN

Tabulating Machine Company koja je prodavala mašine za različite knjigovodstvene namene, kasnije će postati jedan od tri osnivača IBM-a.

Veliki doprinos razvoju računarstva dao je engleski matematičar Džorž Bul (George Boole) koji je 1854. godine objavio radove o binarnoj logici, koji će tek 90 godina kasnije poslužiti računarskim pionirima da projektuju elektronske računare.



Slika 2.5 Tabulating machine

Godine 1890. Herman Holerit (Herman Hollerith), koji je radio u Američkom popisnom birou, je napravio mašinu za čitanje i sortiranje bušenih kartica, čime je omogućio da se skrati obrada podataka sa deset godina na dve i po. Za zapisivanje podataka koristila se pravougaona kartonska kartica na kojoj su podaci zapisivani ručnim bušenjem rupica.



Slika 2.6 Tabulating machine

Mašina za čitanje kartica (**tabulating machine**) je imala čitač sa iglicama koje su na mestima gde su postojali otvoreni električni kontakti što je izazivalo inkrementiranje brojača i automatsko sortiranje kartica. Ovo je praktično bio prvi sistem za automatsku obradu podataka i njime su tada obrađeni podaci o 62 miliona stanovnika Amerike. 1896. Herman Hollerit je osnovao firmu Tabulating Machine Company koja je prodavala ove mašine za različite knjigovodstvene namene, a koja će kasnije postati jedan od tri osnivača IBM-a.

Godine 1897. Karl Braun je napravio prvu katodnu cev koja će se kasnije koristiti kod televizora i monitora.

KONRAD ZUSE

Prvi elektro-mehanički računar Z1 koji je koristio binarni kod

Neposredno pre početka drugog svetskog rata u nekoliko razvojnih centara u Evropi i Americi se paralelno radi na razvoju računara. Nemački inženjer Konrad Cuze (Konrad Zuse) je 1936. godine patentirao mehaničku memoriju da bi već 1938. napravio prvi elektro-mehanički računar **Z1** koji je koristio binarni kod i pouzdano radio. Kasnije je napravio i njegove naslednike **Z2**, **Z3** i **Z4**. Ovi računari su učitavali sekvenце instrukcija sa mehanizma koji je čitao papirnu bušenu traku. Manje je poznato da je Cuze predložio i jedan od prvih jezika koji se zvao **Plankalkul**.

DŽON ATANASOV, HAUARD AJKEN, ALEN TJURING

Alen Tjuring je dao jedan apstraktni model takve mašine koji je danas poznat kao "Tjuringova mašina"

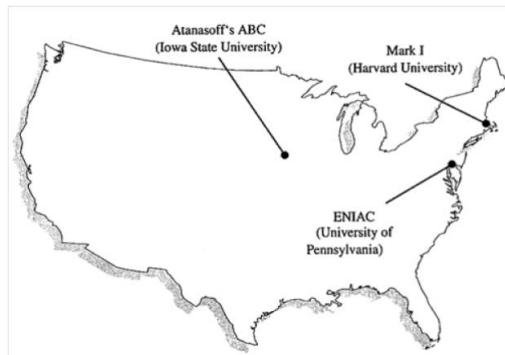
Godine 1937. fizičar Džon Atanasov (John Atanasoff) koji je stalno imao problema sa rešavanjem sistema linearnih jednačina i njegov student Kliford Beri (Clifford Berry) sa Iowa

State University započeli su rad na prvom elektronskom digitalnom računaru **ABC**. Ovaj računar je koristio elektronsku binarnu logiku, a za memoriju je koristio dinamički osvežavane kondenzatore, što je metod koji se i danas koristi u RAM memorijama. Interesantno je da je računar bio paralelan i da je mogao da obavlja 30 operacija simultano iako je imao mnogo manje delova nego serijski računari iz 40-tih godina prošlog veka. Računar je bio funkcionalan 1942. godine, ali je zbog rata rad na njemu prekinut. Atanasov je planirao da umesto ručnog unosa instrukcija sa kontrolnog panela dogradi uređaj koji bi omogućio da se sekvenca instrukcija automatski učitava. Skoro deset godina kasnije na osnovu njegovog pionirskog rada biće napravljen **ENIAC**.

Inženjeri s Harvara, predvođeni Ajkenom ([Howard Aiken](#)) koji je bio doktorant iz oblasti fizike i imao problema sa rešavanjem parcijalnih diferencijalnih jednačina, su zajedno sa IBM-om 1944. napravili elektromehanički kalkulator **Mark I**. Ovaj računar je imao 765.000 delova i mogao je da vrši tri sabiranja u sekundi, dok su mu za množenje trebale 6 sekunde a za deljenje 15,3 s.

Jedan od pionira računarstva svakako je [Alan Tjuring \(Alan Turing, 1912-1954\)](#) koji se smatra jednim od prvih teoretičara računarstva. Tjuring je tokom drugog svetskog rata bio angažovan na problem dešifrovanja nemačkih tajnih poruka. Njegov pristup tom problemu zasivao se na iznalaženju mašine koja će biti u stanju da reši svaki problem predstavljen nizom elementarnih operacija, a njena memorija je trebalo da bude dovoljno velika da može da skladišti instrukcije potrebne za račun.

On je dao jedan apstraktni model takve mašine koji je danas poznat kao "Tjuringova mašina".



Slika 2.7 Mapa

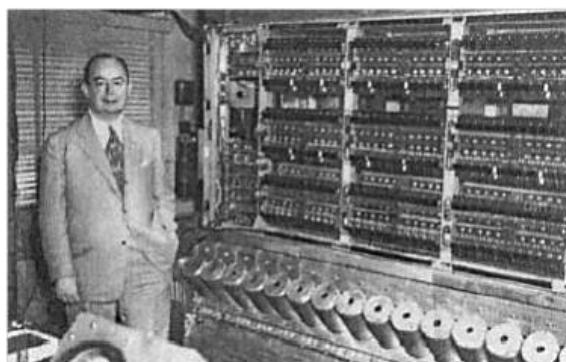
DŽON FON NOJMAN

Njegov predlog koncepta memorisanog programa (stored program concept), koji se i danas primenjuje, podrazumevao je da se program i podaci smeštaju u istu memoriju.

Drugi važan teoretičar računarstva je [Džon fon Nojman](#) (John von Neumann ,1903-1957, rođen kao Janoš u Budimpešti) koji je dao osnovne principe arhitekture današnjih računara. On je prvi predložio 1945. godine arhitekturu računarskog sistema u kojoj se računar jasno deli na hardvera i softvera.

Njegov predlog koncepta memorisanog programa (**stored program concept**), koji se i danas primenjuje, podrazumeva da se program i podaci smeštaju u istu memoriju. Računar razlikuje podatke od programa, a podacima može da pristupi na najbrži mogući način. Program se dekodira pomoću neke vrste fizičkih logičkih kola, a onda se instrukcije izvršavaju pomoću hardvera. Ova arhitektura računara je nazvana fon Nojmanova arhitektura.

Džon fon Nojman 1943. počinje sa radom u laboratoriji Los Alamos gde 1944. zajedno sa Džon Moklijem (John Mauchley) i Džon P. Ekertom (John P. Eckert) radi na projektu **EDVAK (EDVAC)**. Oni su projektovali prvi potpuno elektronski računar koji je radio na osnovu unapred zadatog programa.



Slika 2.8 John von Neumann i IAS kompjuter

. The New York Times, http://www.nytimes.com/2012/05/06/books/review/turings-cathedral-by-george-dyson.html?pagewanted=all_r=0 (1.7.2015)

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

VIDEO - HISTORY OF THE COMPUTER DOCUMENTARY

History Of The Computer Documentary

<p> <youtube width="817" height="460" src="https://www.youtube.com/embed/sOb2gTdAcCc" /> </p>

VIDEO - ISTORIJA IBM-A

IBM Centennial Film: 100 X 100 - A century of achievements have changed the world

<p> <youtube width="817" height="460" src="https://www.youtube.com/embed/sSuk9i_UZyl" /> </p>

VIDEO - COMPUTER PIONEERS: PIONEER COMPUTERS PART 1

Part 1 of 2 The Dawn of Electronic Computing

<p> <youtube width="817" height="460" src="https://www.youtube.com/embed/qundvme1Tik" /> </p>

▼ Poglavlje 3

Generacije Računara

PODELA RAČUNARA NA GENERACIJE

U starijoj literaturi se susreće podela računara na četiri generacije

Prvi elektronski računari su se pojavili 1945. godine. Od tada su se računari stalno usavršavali primenjujući mnoge nove tehnologije. U starijoj literaturi se susreće podela računara na četiri generacije, mi dodajemo i petu. Ova podela je načinjena prema njihovom tehnološkom nivou. Ove generacije su:

- I generacija – sa elektronskim cevima, akustičnim memorijama
- II generacija – sa tranzistorima i magnetnim memorijama
- III generacija – sa integriranim sklopovima niskog stepena integracije
- IV generacija – sa integriranim sklopovima visokog stepena integracije i mikroprocesorima
- V generacija – portabilni računari sa fokusom na mobilnošću korisnika.

Iako ova podela nije idealna predstavlja relativno dobru osnovu za pogled na istorijski razvoj računara.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

I GENERACIJA RAČUNARA

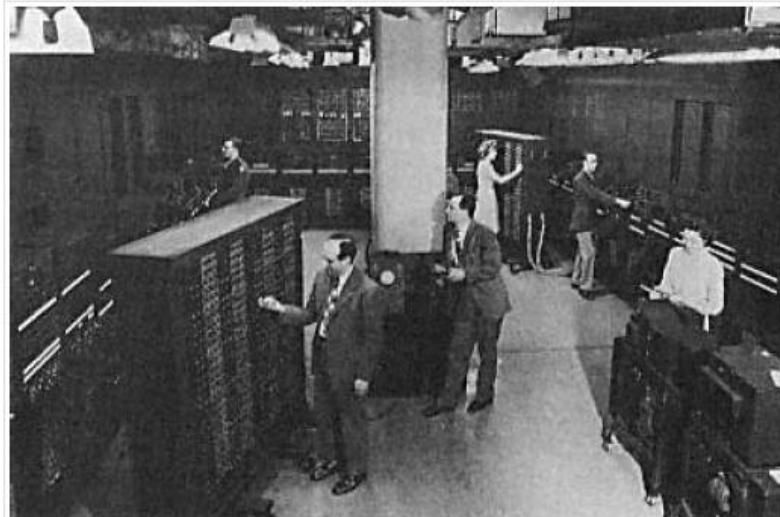
Prvi elektronski računar opšte namene su projektovali i izgradili Presper Eckert i dr John W. Mauchly s Pensilvanijskog univerziteta

Prvi elektronski računar opšte namene su projektovali i izgradili Presper Eckert i dr John W. Mauchly s Pensilvanijskog univerziteta. Razvoj ovog računara trajao je od jula 1943. do novembra 1945. godine. Računar je nosio ime **ENIAC** (**E**lectronic **N**umerical **I**ntegrator and **C**omputer) i bio je projektovan za potrebe Ballistic Research Laboratory kako bi se ubrzali njihovi proračuni. ENIAC je za to doba radio „brzinom svetlosti“ jer je mogao da vrši 5000 operacija sabiranja ili oduzimanja desetocifrenih brojeva u sekundi. To je bilo 1000 puta brže od bilo koje postojeće mašine u to doba. ENIAC je imao 18000 elektronskih cevi, 1500 releja, 70000 otpornika i 10000 kondenzatora i težio 30 tona.

Za razliku od drugih mašina, ENIAC nije vršio samo uobičajene kalkulacije kao što su sabiranje, oduzimanje, množenje, deljenje i korenovanje, već je mogao da memorise među

rezultate i komunicira sa različitim jedinicama. Mogao je da izvršava umetnute petlje i instrukcije grananja, kao i čitanje i štampanje brojeva.

Ipak ENIAC nije imao mogućnost memorisanja programa. Programiranje se vršilo tako što su se na njegovim jedinicama ručno podešavali programski prekidači, a pojedine jedinice povezivale, što je odnosilo dosta vremena.



Slika 3.1 ENIAC

. University of Pennsylvania, ENIAC (Electronic Numerical Integrator and Computer), 1946 ca., Pennsylvania

UNIVAC

Jedan od najuspešnijih računara I generacije je bio UNIVAC I

Engleski profesori Freddie Williams i Tom Kilburn sa Univerziteta u Mančesteru su 1948. godine razvili prvi elektronski računar koji je mogao da memoriše program i da ga izvršava. Ovaj računar je nazvan **Baby**. Računar je razvijen oko Kilburnovog koncepta memorije na bazi katodne cevi (CRT). Ova memorija, nazvana Viljamsova cev, je mogla da čuva 1024 bita nekoliko časova bez reprogramiranja. Baby je imao 6000 vakumskih cevi, bio je 5 m dug i 2 metara visok, i težio je jednu tonu. Na osnovu ovog računara razvijena je 1949. godine poboljšana verzija nazvana Manchester Mark I. Kanadska kompanija Ferranti-Packard je 1951. godine počela proizvodnju komercijalne verzije ovog računara pod nazivom Ferranti Mark I.

Jedan od najuspešnijih računara I generacije je bio **UNIVAC I** (**UNI**versal **A**utomatic **C**omputer) koga su 1951. godine razvili Mokli i Džon Ekert. Ovo je bio i prvi komercijalni elektronski računar koga je proizvodila kompanija Remington Rand. Prodato je ukupno 40 primeraka, a jedan je bio instaliran u Američkom birou za popis. Radna memorija je mogla da čuva 1000 brojeva sa 12 cifara. UNIVAC je imao i spoljnu memoriju. Korišćena je magnetna traka koja je mogla da sačuva 1 MB podataka sa gustinom od 128 karaktera po inču dužine.

Godine 1952. UNIVAC I uspešno predviđa ubedljivu pobedu kandidata Dvajta Ajzenhauera na predsedničkim izborima u Americi protiv Adlaja Stivensona, uprkos suprotnim prognozama stručnjaka. UNIVAC I se uspešno koristio sve do početka 60-tih godina prošlog veka.



Slika 3.2 UNIVAC

. Computer History, <http://www.computerhistory.org/timeline/?year=1951> (1.7.2015)

II GENERACIJA RAČUNARA

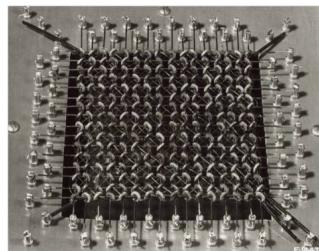
Poseban uticaj na razvoj računarstva imao je pronađak tranzistora i memorija sa magnetnim jezgrima

Poseban uticaj na razvoj računarstva imao je pronađak **tranzistora**, čime je u stvari počela era poluprovodnika. Decembra 1947. godine William Shockley, Walter Brattain, i John Bardeen uspešno su testirali prvi tranzistor, a njegova poboljšana verzija razvijena u AT&T Bell Laboratorijama zamenila je vakumske cevi korišćene u tadašnjim računarima.



Slika 3.3 Tranzistor

Druga komponenta koja je predstavljala okosnicu II generacije računara bila je **memorija sa magnetnim jezgrima**. Prva ovakva memorija ugrađena je na MIT-u na računaru Whirlwind i imala je 256 bita.



Slika 3.4 Memorija od 256 bita

Tipičan predstavnik II generacije računara je IBM-ov računar 650 koji je proizведен 1953. Ovo je bio prvi računar namenjen širokom tržištu; kompanija je prodala 1500 primeraka pre nego što ga je 1969. povukla sa tržišta. Podaci su se u računar učitavali sa bušenih kartica ili magnetne trake. Za radnu memoriju se koristio magnetni bubanj sa fiksnom glavom koji se okretao sa 12.500 o/min. Na njemu je moglo da se memoriše 1000 do 2000 desetocifarnih reči. Od 1956. u ovaj model se ugrađivao i magnetni disk.

Ono što je još interesantno za ovo doba je da je kompanija American Airlines instalirala prvu veliku mrežu koja je povezivala 1200 teleprinteru koji su radili sa baze podataka koju je napravio IBM.

III GENERACIJA RAČUNARA

Pojava integrisanih kola u mnogome je smanjila veličinu komponenata računara

Kompanija Texas Instruments je 1954. godine počela prodaju silikonskih tranzistora po ceni od samo 2,5 dolara za komad. Međutim mnogo veći napredak za računarstvo kompanija je postigla 1958 godine proizvodnjom prvog **integrisanog kola**. Njihov inženjer Jack Kilby je dokazao da je moguće da otpornik i kondenzator egzistiraju u istom delu poluprovodničkog materijala.

Pojava integrisanih kola u mnogome je smanjila veličinu komponenata računara, a kasnije sa povećanjem stepena integracije komponenata i do minijaturizacije računara.

Tipični predstavnici računara III generacije bili su IBM-ov System/360 i mini računar PDP-1 kompanije DEC (Digital Equipment Corporation). System/360 je, nakon četvorogodišnjeg razvoja koji je koštao 4 milijarde dolara, uveden u proizvodnju 1964. godine. Prvi put u istoriji računarstva najavljena je kompletna linija računara u isto vreme. Ovaj proizvod se pored FORD-ovog modela T i Boeing-a 707 smatra najvećim američkim poslovnim uspehom. Zahvaljujući ovom računaru IBM je postao vodeći proizvođač računara na svetu.

Mnogo manji komercijalni uspeh doživeo je i DEC-ov računar Programmed Data Processor PDP-1 iz 1959. godine koji je prodat u 50 primeraka. Međutim, ovo je bio prvi mini računar koji se prodavao po tada neverovatno niskoj ceni od 120.000 \$. Računar je bio revolucionaran i po tome što je koristio katodnu cev kao grafički displej i tastaturu za unos podataka i komandi. Računar je imao 18-bitni procesor, memoriju od 4000 18-bitnih reči i mogao je da vrši 100.000 sabiranja u sekundi. PDP-1 je uveo neka revolucionarna rešenja. Jedno od njih je da su podaci sa ulaznih uređaja bili direktno prosleđivani memoriji uz minimalno angažovanje procesora.

Ova tehnologija je kasnije nazvana DMA- **Direct Memory Access** i realizovana je uvođenjem prekida (**interrupts**). Kasnije je za ovaj računar napravljena prva video igra SpaceWar koja je u demonstracione svrhe instalirana na svih 50 prodatih računara.

Naslednik ovog računara, 12-bitni minikompjuter PDP-8 iz 1965. godine je prodat u 50.000 primeraka uz cenu od 20.000 \$. Zbog obilja literature i otvorenosti cele kompanije prema korisnicima PDP računari su bili omiljeni u akademskim i razvojnim institucijama.

ODLIKE RAČUNARA III GENERACIJE

Pored razvoja samih računara, počinje se i sa standardizacijom

Osnovne odlike računara III generacije su:

- Poluprovodnička tehnologija integrisanih kola
- Poluprovodničke memorije
- OS sa multiprogramskim radom
- Interaktivni rad sa korisnikom

Pored razvoja samih računara, počinje se i sa standardizacijom. Tako je 1964. godine Američko udruženje za standarde usvojilo je ASCII kod kao standardni kod za prenos podataka.



Slika 3.5 IBM System/360

. IBM 360, <http://www.ibm360.info/> (1.7.2015)

IV GENERACIJA RAČUNARA

Četvrtoj generaciji računara karakteriše primena komponenata sa kolima visokog stepena integracije (VLSI) i mikroprocesora

Četvrtoj generaciju računara karakteriše primena **komponenata sa kolima visokog stepena integracije (VLSI) i mikroprocesora**. 1971. godine se pojavio prvi četvorobitni

mikroprocesor Intel 4004, da bi ista kompanija već sledeće godine izbacila na tržište 8 bitni mikroprocesor Intel 8080. 1974. Godine kompanija General instrument proizvodi prvi 16 bitni mikroprocesor.

Sve ove komponente su bitno uticale na to da se smanje dimenzije računara uz istovremeno povećanje njihove pouzdanosti. Razvijen je čitav niz novih ulazno – izlaznih uređaja koji je olakšao korišćenje računara i proširio oblast njihove primene.

Godine 1981. IBM je predstavio svoj prvi personalni računar - PC s operativnim sistemom MS DOS. Od tada personalni računari potiskuju druge vrste računara, tako da sada predstavljaju dominantni oblik računara.

Prvi PC je sagrađen oko Intelovog procesora Intel 8080 koji je radio na taktu 4,77 MHz. Standardne verzije su isporučivane sa 16 ili 64 KB RAM-a. Računar nije imao hard disk, ali je dolazio sa jednom ili dve disketne jedinice. Isporučivan je sa monohromatskim displejom.



Slika 3.6 Prvi IBM PC računar

. Code Forty Two, <https://web.archive.org/web/20120825135915/http://crashplan.com/blog/misc-rambling/happy-birthday-pc> (15.05.2020.)

V GENERACIJA RAČUNARA

Mobilni računari, ugrađeni računari, tableti, telekomunikacioni uređaji i sl.

Prognoziranje budućeg razvoja računarstva je krajnje neizvesno. Istorija je pokazala da su mnoga paralelna otkrića (na primer tranzistora) uticala na neočekivani razvoj računara. Ono što se može uočiti su trendovi u računarstvu. Navešćemo samo neke:

- Minijaturizacija računara i perifernih uređaja uz dalji pad cena,
- Sveprisutno računarstvo. Računari postaju deo gotovo svih uređaja koje čovek koristi.
- Automobili, klima uređaji, mašine za pranje i drugi uređaji imaju u sebi ugrađene računare,
- Konvergencija računara i telekomunikacionih uređaja,
- Mobilni računari.

▼ Poglavlje 4

Istorija razvoja programskih jezika

ISTORIJSKA ZAVISNOST SOFTVERA OD HARDVERSKE PLATFORME

Softver je uvek razvijan za određeni nivo funkcionalnosti koju je nudio hardver u datom trenutku svog razvoja

Razvoj softvera nije moguće posmatrati nezavisno od razvoja hardvera. Softver je uvek razvijan za određeni nivo funkcionalnosti koju je nudio hardver u datom trenutku svog razvoja. Iz ovoga sledi i konstatacija da napredak u razvoju hardvera nije mogao da daje očekivane rezultate sve dok se ne razvije i adekvatni softver. Pogledajmo, kao primer, pojavu šezdesetčetvorobitnih procesora. Njihovu snagu nije bilo moguće iskoristiti dok se nisu pojavili šezdesetčetvorobitni operativni sistemi i aplikacije.

PAPIRNA BUŠENA TRAKA

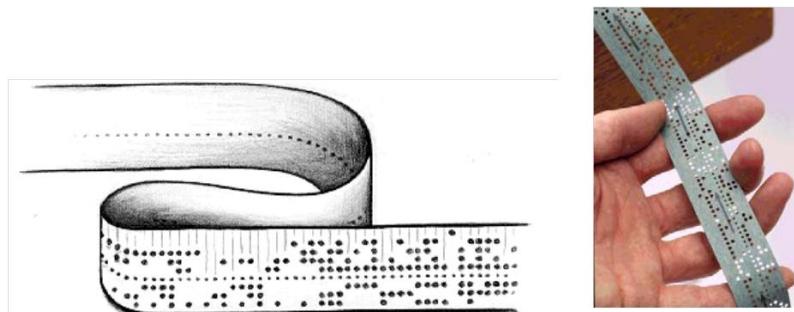
Računari koje je projektovao Konrad Cuze mogli da čitaju sekvence instrukcija sa mehanizma koji je čitao papirnu bušenu traku

Prvi elektronski računari nisu imali operativni sistem niti su imali pisane programe. Umesto toga, njihovi projektanti su, za svaki poseban problem, vršili rekonfiguraciju komponenata kako bi računar mogao da izvrši određene računske operacije. I pre pojave prvih elektronskih računara bilo je pokušaja da se operacije koje računar treba da izvrši unapred definišu i zapišu na način koji bi bio pogodan da ih računar pročita i automatski izvršava. Tako su, na primer, računari koje je projektovao Konrad Cuze mogli da čitaju sekvence instrukcija sa mehanizma koji je čitao papirnu bušenu traku. Tokom projektovanja računara Z4 on je predložio izradu „mašine za pripremu plana“ (**Planfertigungsgeräte**) koja će pripremati traku sa programom i vršiti proveru sintakse. Ova mašina nikad nije napravljena u tom obliku, ali kada je u Federalnom tehničkom institutu u Cirihi 1952. godine ponovo napravljen računar Z4, Cuze je predložio izradu sličnog uređaja koga je nazvao **programator**.

Koristeći računar Z4, Hajnc Rutišauzer (Heinz Rutishauser) je uočio da računar uopšte može da bude programiran da obavlja i funkciju programatora. On je predložio da se problemi rešavaju u dva koraka. U prvom bi se računar programirao da proveri i prevede korisnikove komande i zapiše ih u binarnom obliku na papirnu traku.

U drugom koraku bi računar izvršavao instrukcije sa papirne trake. Rutišauzer je to definisao kao „korišćenje računara kao njegove sopstvene mašine za pripremu plana“. Njegova

zapažanja su bila osnova za koncipiranje rada budućih elektronskih računara koji su mogli da memorišu program, odnosno za stvaranje softvera.



Slika 4.1 Papirna bušena traka

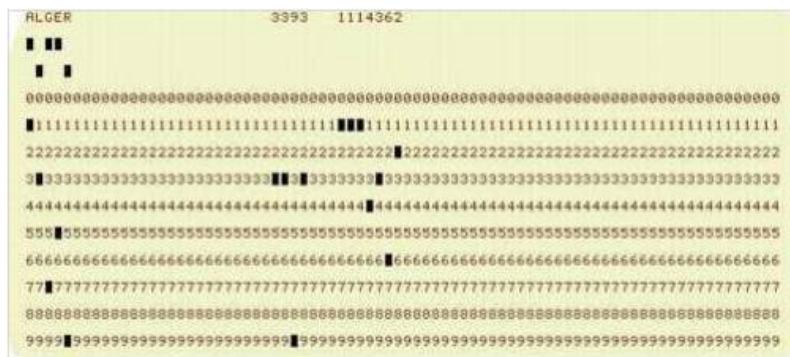
. PC Mag - Encyclopedia, <http://www.pc当地>/encyclopedia/term/48807/paper-tape
(2.7.2015)

BUŠENA KARTICA

Programeri su mogli da iz biblioteka uzmu potrebne špilove i da naprave paket kartica koji čini jedan kompletan set instrukcija (programski paket) za rešavanje nekog problema

Grejs Marej Hoper (Grace Murray Hopper 1906 – 1992), profesorka matematike, je takođe koristila papirnu bušenu traku, na kojoj je moglo da se izbuši 24 otvora po širini, da bi 1944. godine učitavala programske sekvene u elektromehanički kalkulator Mark I razvijen na Harvardu. Ona je vrlo brzo uočila da se pojedine programske sekvene ponavljaju i da bi bilo dobro da postoji biblioteka sekvenci koje bi po potrebi mogle da budu ponovo korišćene pri rešavanju nekih drugih problema.

Ova iskustva su projektantima računara dala novu ideju. Sekvene binarnih instrukcija koje su se često ponavljale, na primer za izračunavanje sinusa, logoritma i sličnih funkcija, su zapisivane na grupi bušenih kartica. Tako su formirani špilovi (engleski decks) bušenih kartica koji rade neku, tačno određenu operaciju. Špilovi koji su rešavali neku klasu instrukcija predstavljali su biblioteku. Ovo je u mnogome ubrzalo proces pripreme koda za neki novi problem. Programeri su mogli da iz biblioteka uzmu potrebne špilove i da naprave paket kartica koji čini jedan kompletan set instrukcija za rešavanje nekog problema. Odatle je potekao naziv **programski paket**. Drugi izraz koji je potekao iz ovog doba je kompajliranje. Naime izbor i sastavljanje (engleski **compile**) potrebnih špilova je postala aktivnost kojom se stvarao novi program.



Slika 4.2 Izgled bušene kartice



Slika 4.3 Uredaj za bušenje i čitanje kartica (key punch machine)

▼ Poglavlje 5

Prva generacija programskih jezika - mašinski jezici

PRVA GENERACIJA PROGRAMSKIH JEZIKA

Mašinski jezici

Ranih pedesetih godina dvadesetog veka razvijeni su programi koji su automatizovali ove aktivnosti i takvi programi su nazvani **kompajleri** (engleski **compilers**). Grejs Hoper je celu aktivnost koja je koristila kompjajlere nazvala „**Automatic Programming**“. Početkom 1952. godine kompjajler pod nazivom **A-0** je radio na računaru UNIVAC. Izraz kompjajler se i danas koristi da označi program koji prevodi instrukcije pisane u jeziku višeg nivoa, kao što je na primer C++, u binarni kod razumljiv nekom specifičnom procesoru.

Pisanje programa u mašinskom kodu, čak i uz upotrebu biblioteka, predstavljalo je težak zadatak koji je mogla da obavlja mala grupa ljudi koja je odlično poznavala funkcionisanje specifičnog računara. Stoga se rodila ideja da se napravi programski jezik koji će biti sličan prirodnom jeziku i imati neka sintaksna pravila. Iskazi ovog jezika su bili mnemonici izvedeni iz engleskog jezika. Umesto da se piše binarni kod neke procesorske instrukcije pisao se mnemonik. Tipičan primer iskaza iz ovog jezika je

LR adresa

U ovom slučaju LR je mnemonik dobijen od engleskih reči **Load Register** i predstavlja instrukciju kojom se sadržaja memorijske lokacije naveden kao adresa učitava u registar procesora. Slične su bile i drugi iskazi ovog jezika. Svakom iskazu jezika odgovarala je jedna binarna instrukcija procesora ili, u redim slučajevima kada su se koristile makro instrukcije, nekoliko. Očigledna je prednost u tome da programer nije morao da piše niz binarnih cifara koje odgovaraju instrukciji **Load Register**. Pored toga memorijska adresa je mogla da se napiše u obliku simboličke labele čija je numerička vrednost mogla da se promeni iz razloga koji nisu poznati u trenutku pisanja programa. Program nije prevođen nego montiran (engleski **assembled**), a programi koji su to radili su nazvani **asemblieri**.

✓ 5.1 Druga generacija programskih jezika - asembleri

DRUGA GENERACIJA PROGRAMSKIH JEZIKA

Zadatak asembler programa je bio da prevede mnemoničke instrukcije i da odredi stvarne fizičke adrese promenljivih u memoriji

Zadatak asembler programa je bio da prevede mnemoničke instrukcije i da odredi stvarne fizičke adrese promenljivih u memoriji u trenutku izvršenja programa. Kasnije su uz pomoć velikih biblioteka makroa programeri koji su koristili asembler postali mnogo produktivniji.

Asembleri jezici su svrstani u **II generaciju** programskih jezika, u odnosu na mašinske jezike koji predstavljaju prvu generaciju.

✓ 5.2 Treća generacija - jezici visokog nivoa

FORTRAN

FORTRAN (skraćenica od Formula Translation)

Treću generaciju jezika čine takozvani jezici visokog nivoa (engleski **high-level languages**). Prvi takav jezik počeo je da razvija Jon Bakus (John Backus) 1954. godine za IBM-ov računar 704. Jezik je bio namenjen za rešavanje naučnih i inženjerskih problema koji su obilovali proračunima pa je nazvan

FORTRAN (skraćenica od **Formula Translation**). Razvoj jezika i prvih prevodioca – kompjajlera za ovaj jezik završen je 1957. godine. FORTRAN je bio vrlo brzo prihvaćen u naučnim krugovima i doživeo je ogroman uspeh, jer je masovno korišćen preko 40 godina. Sintaksa jezika se oslanjala na engleske reči i klasične aritmetičke izraze pa je jezik bilo lako naučiti i koristiti. Uspeh i opstanak ovog jezika leži u činjenici da je, iako jezik visokog nivoa, omogućavao, onda kada je to potrebno, pristup i operacijama niskog nivoa. Tok programa pisan u FORTRANU definisao je potpuno proceduru koju će računar izvršavati, tako da ovaj jezik spada u tipično proceduralne jezike.

U daljem tekstu se daje primer jednog FORTRAN-skog programa koji koristi dve DO petlje da izračuna koordinate tačaka koje su pravilno raspoređene duž x i y ose.

```
C PRIMER IZRACUNAVanja KOORDINATA TACAKA KOJE SU RASPOREDJENE
C KAO MATRICA OD 10 X 10 TACAKA. RASTOJANJE IZMEDJU TACAKA
C PO X OSI JE DX, A PO Y OSI DY.
C POCETNA TACKA IMA KOORDINATE X0,Y0
DIMENSION X(10), Y(10)
READ (2,210) X0,Y0,DX,DY
```

```
DO 100 J=1,10
Y(J)= Y0 + (J-1)*DX
DO 150 I=1,10
X(I)=X0 + (I-1)*DX
WRITE (3,220) X(I),Y(J)
ENDDO
ENDDO
150 CONTINUE
100 CONTINUE
210 FORMAT (2F8.2, 2F8.2)
220 FORMAT (2E12.4)
STOP
END
```

COBOL

COBOL je postao prvi jezik koji je garantovao da će isti program moći da se izvršava na različitim računarima pružajući pri tom iste programske rezultate

Iako je FORTRAN bio idealan jezik u oblasti rešavanja naučnih i tehničkih problema, on nije mogao da zadovolji zahteve u oblasti izrade poslovnih aplikacija. Zbog toga je U.S. Department of Defense obrazovalo maja 1959. godine komitet sa zadatkom da se razvije zajednički poslovni jezik. Do kraja godine komitet je sačinio preliminarnu specifikaciju jezika koji je nazvan **COBOL** (skraćenica od punog engleskog naziva **Common Business Oriented Language**). Odmah nakon objavljivanja specifikacije, nekoliko proizvođača softvera je objavilo da piše COBOL prevodioce za odgovarajuće računare. Sledeće godine američka vlada je objavila da neće kupovati računare koji ne rade sa COBOL-om. Kao rezultat ove objave, COBOL je postao prvi jezik koji je garantovao da će isti program moći da se izvršava na različitim računarima pružajući pri tom iste programske rezultate. Ovo je dokazano decembra 1960. godine kada je isti COBOL program testiran na računarima UNIVAC II i RCA 501. Treba primetiti da je ovo bio veliki korak u računarstvu, jer su se do tada programi pisali samo za tačno određeni računar.

Snažan uticaj fisionomiju COBOL-a dala je Grejs Marej Hoper koja je 1956. godine razvila jezik B-0, kasnije nazvan MATH-MATIC i FLOW-MATIC koji je bio orijentisan ka poslovnim aplikacijama. Pored toga na COBOL su uticali IBM-ov projekat Commercial Translator i Honeywell-ov jezik FACT. Programski jezik **FACT** je po mnogim osobinama bio bolji od COBOL-a, ali nije imao podršku vlade. Njega je na osnovu ugovora sa Honeywell-om razvila jedna od prvih komercijalnih softverskih kompanija Computer Sciences Corporation. Pod ovim uticajima COBOL je dobio „čitljivost“ kakvu FORTRAN nije imao. Na primer, iskaz u FORTRAN-u koji glasi

IF A > R

bi u COBOL-u bio napisan kao

IF BROJ-RADNIH-SATI IS GREATER THAN RADNO-VREME

što je očigledno mnogo čitljivije i razumljivije za menadžere koji nisu dovoljno znali programiranje. Na neki način sam kod COBOL-a je bio samo-dokumentovan.

ALGOL I LISP

ALGOL je uveo mnoge nove koncepte koji se do tada nisu koristili u programskim jezicima, kao što je, na primer, rekurzivno pozivanje funkcija

Nedostatak FORTRAN-a je bio pre svega u tome što je inicijalno pisan za IBM-ov procesor 704. Zbog toga je 1958. godine naučna zajednica uglavnom u Evropi započela sa razvojem jezika **ALGOL** (*Algorithmic Language*) koji je trebao da bude alternativa FORTRAN-a nezavisna od bilo koje hardverske konfiguracije. Jezik je pažljivo formalno definisan tako da nije bilo dvoznačnosti u tome šta neki izraz treba da radi. Same definicije su prvi put zadate u BNF jeziku (*Backus-Normal-Form* ili *Backus-Naur-Form*). ALGOL je uveo mnoge nove koncepte koji se do tada nisu koristili u programskim jezicima, kao što je, na primer, rekurzivno pozivanje funkcija. ALGOL je uglavnom razvijan u Evropi i postojala su velika nadanja da će se stvoriti opšte prihvaćeni jezik za rešavanje naučno-tehničkih problema koji neće biti zavisan od IBM-a, ali se to nije dogodilo. Njegova kasnija verzija ALGOL 68 bila je preobimna, što je nateralo programere da pređu na mnogo kompaktnije jezike, kao što je Pascal. Ipak ALGOL je značajan jezik jer je uticao na razvoj mnogih drugih jezika kao što su Pascal, C, C++ i Java.

Prvi jezik koji je bio namenjen za rešavanje problema iz oblasti veštačke inteligencije, **LISP** (*LIST Processing*) kreirao je 1958. godine John McCarty sa MIT-a. Jezik je bio namenjen obradi simbola, a ne obradi algebarskih izraza, pa je zbog svoje specifične namene imao sintaksu koja do tada nije bila viđena [3]. U praksi LISP se koristio za gradnju ekspertnih sistema, u simboličkoj algebri, projektovanju integrisanih elektronskih komponenata, robotici, prepoznavanju oblika itd. LISP ima samo jedan tip podataka – listu, koja se sastojala od niza elemenata unutar zagrada. Sam LISP program je pisan kao set listi, tako da je LISP program imao jedinstvenu osobinu da može da menja samog sebe i tako raste. Činjenica da se LISP koristio najčešće u laboratorijama za veštačku inteligenciju doprinela je da se u LISP okruženju stvore mnogi novi programerski alati kao što su: prvi struktuirani editor (*structured editor*), *pretty-printing*, prvi sistem za automatsku korekciju grešaka u programu, ali i tehnologija automatskog upravljanja memorijom tokom izvršenja programa – *garbage collection*.

LISP

Prvi jezik koji je bio namenjen za rešavanje problema iz oblasti veštačke inteligencije, LISP (LIST Processing)

Sledeći primer pokazuje kako je lako implementirati rekurziju u LISP-u. Funkcija filter uzima kao argument listu brojeva koji mogu biti pozitivni ili negativni realni brojevi, a kao rezultat vraća novu listu koja se sastoji samo od pozitivnih brojeva. Algoritam funkcije je sledeći:

- Uzmi listu brojeva

- Ako je lista prazna, nema šta da se radi pa vrati praznu listu
- Ako lista nije prazna, i ako je prvi broj u listi manji od nule, ignoriši ga i počni od broja jedan sa ostatkom liste
- Ako prvi broj u listi nije manji od nule, ostavi ga u listi i nastavi istu proceduru do kraja liste.

```
(defun filter (list-of-numbers)
```

```
  (if (null list-of-numbers) nil
```

```
    (if (< (car list-of-numbers) 0)
```

```
      (filter (cdr list-of-numbers))
```

```
      (cons (car list-of-numbers)
```

```
        (filter (cdr list-of-numbers))))))
```

Primer LISP funkcije filter

Ako bi se sada primenila funkcija filter na listu brojeva kao što je sledeća

```
(filter '(1 2 4.5 -6 20 -3/5 .003))
```

dobio bi se rezultat u obliku sledeće liste:

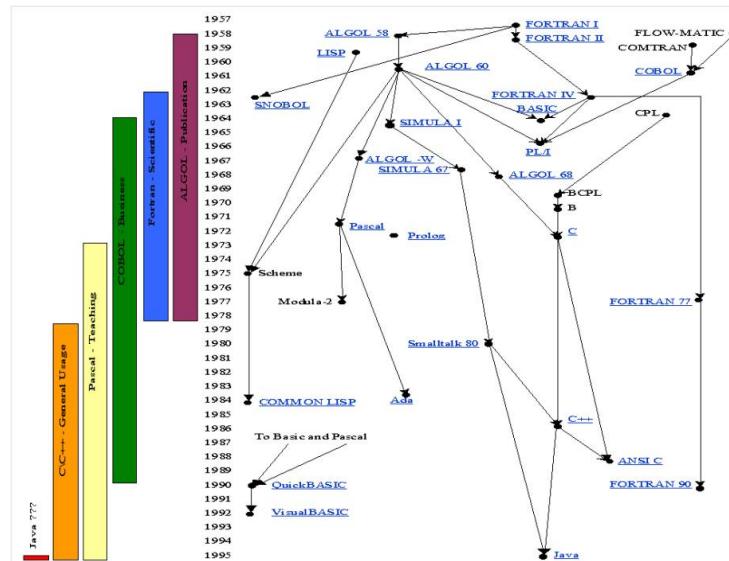
```
(1 2 4.5 20 .003)
```

ISTORIJSKI RAZVOJ

Prikaz istorijskog razvoja programskega jezika

FORTRAN, COBOL i LISP su kao prva tri jezika treće generacije bili inspiracija za kasniji razvoj mnogih drugih jezika koji su danas u primeni. Tako je Pascal nastao na osnovu ALGOL-a, koji je nastao od FORTRAN-a. Slične korene su imali i jezici C i BASIC. Jezik C++ je nastao na osnovu iskustva u razvoju jezika C i Smalltalk. Java je nastala na osnovu jezika C++ ali sa idejom da bude jezik za razvoj Internet, odnosno web aplikacija.

Na narednoj slici je data uproščena šema istorijskog razvoja najvažnijih jezika.



Slika 5.1.1 Istorijski razvoj programskih jezika

. Gary T. Leavens's WWW Pages, <http://www.eecs.ucf.edu/~leavens/> (15.05.2020.)

▼ Poglavlje 6

Pokazna vežba: JavaScript objekti

JAVASCRIPT OBJEKTI

Objekat u JavaScriptu je skup polja

Predviđeno vreme pokazne vežbe je 30 minuta.

JavaScript je objektno orijentisan jezik sa relativno jednostavno objektnom paradigmom. Za razliku od Java i sličnih jezika, JavaScript je dinamički jezik. U JavaScriptu ne postoji pojam klase. Objekti se definišu direktno, ili pomoću prototipa.

Objekat u JavaScriptu je skup polja (**field, property**), a polje je kombinacija ključa i vrednosti. Svako polje definiše njegov ključ, tj. ime i vrednost. Ime polja je string, dok vrednost može biti bilo kojeg tipa: string, broj, niz, pa čak i funkcija.

Postoji veliki broj već definisanih objekata (**document, window** itd.) koje konstruiše sam interpreter, a programer skripte može definisati dodatne objekte.

Objekti u JavaScriptu se definišu na sledeći način:

```
var korisnik = { ime_polja1: vrednost_polja2, ime_polja2: vrednost_polja2 }
```

Na primer, sledeći objekat bi mogli koristiti da predstavimo korisnika veb sajta:

```
var korisnik = { email: "luka.sekulic.3278@metropolitan.ac.rs", ime: "Luka", prezime: "Sekulić", indeks: 3278 }
```

PRISTUP POLJIMA OBJEKTA

Za pristupanje poljima objekta koristi se operator „..“

Za pristupanje poljima objekta koristi se operator „..“ (tačka). Tokom izvršavanja programa moguće je dodavati nova polja objektima, čitati i menjati njihovu vrednost.

Poljima je moguće pristupiti i koristeći sintaksu indeksiranja u niz.

Sledeći primer demonstrira pristup poljima objekta:

```
//Objekat "student" sa atributima "ime, prezime i indeks"  
  
var student = {  
    ime: "Luka",  
    prezime: "Sekulić",  
    indeks: 3278  
}
```

```
indeks: 3278,  
informacijeStudenta: function() {  
    return this.indeks + " " + this.ime + " " + this.prezime;  
}  
};  
  
document.getElementById("test").innerHTML =student.informacijeStudenta();  
  
//Funkciji "informacijeStudenta" unutar objekta "student" pristupamo sa ". "
```

METODE

JavaScript tretira funkcije kao gradjane prvog reda, što znači da je funkcije moguće tretirati kao bilo koju drugu vrednost

JavaScript tretira funkcije kao gradjane prvog reda, što znači da je funkcije moguće tretirati kao bilo koju drugu vrednost: dodeliti ih kao vrednost promenjivoj, prosledjivati u funkcije kao parametar, vraćati iz funkcija kao rezultat itd.

Na primer,

```
var f = function() {  
    //Promenjiva f sada ima vrednost funkcije  
    //Ovu promenjivu mogu proslediti u druge funkcije, vratiti kao rezultat, dodati  
    u niz itd.  
    console.log("Hello world");  
};  
  
f(); //Poziv funkcije, ispisuje hello world
```

Funkcije je moguće dodijeliti kao vrednost polju objekta. Ovakve funkcije se nazivaju **metode**.

```
var student = {  
    ime: "Luka",  
    prezime: "Sekulić",  
    zdravo: function() {  
        console.log("Pozdrav!");  
    }  
};  
  
student.zdravo(); //Rezultat pozviva je "Pozdrav!"
```

PRISTUP POLJIMA OBJEKTA IZ METODA

Metode imaju svojstvo da pripadaju nekom objektu

Metode imaju svojstvo da pripadaju nekom objektu. Kao takve, mogu pristupiti poljima tog objekta.

Za pristup poljima objekta kojem metoda pripada, koristi se specijalna referenca objekta **this**.

U sledećem primeru, koristi se referenca this za pristup poljima ime i prezime.

```
var student = {  
    ime: "Luka",  
    prezime: "Sekulić",  
    zdravo: function() {  
        console.log(this.ime);  
        console.log(this.prezime);  
    }  
};  
  
student.zdravo(); //Rezultat poziva je ispis "Luka Sekulić"
```

DOM

Kada browser parsira HTML, on kreira objektni model HTML dokumenta (DOM).

Kada browser parsira HTML, on kreira objektni model HTML dokumenta (**DOM**). Ovo znači da browser pravi objekte koji odgovaraju svakom HTML elementu, atributu itd.

Korišćenjem ovih objekata moguće je pristupiti vrednostima HTML elemenata.

document.getElementById() je metoda na objektu document, koja vraća referencu na objekat sa datim identifikatorom.

Polja nekih DOM objekata je moguće postaviti na vrednost neke funkcije, što se generalno koristi za reagovanje na akciju korisnika, tj. event handling.

Npr. već smo koristili event onclicked na sledeći način:

```
<button onclick="funkcija()">
```

Istu funkcionalnost je moguće postići javascriptom:

```
var dugme = document.getElementById("dugme");  
  
dugme.onclick = function(){  
};
```

INPUT ELEMENTI

HTML input elementi imaju polje value, kojim se može pristupiti trenutnoj vrijednosti elementa

HTML input elementi imaju polje value, kojim se može pristupiti trenutnoj vrednosti elementa. U slučaju elementa za unos teksta, vrednost će biti uneseni tekst.

Checkbox element ima polje checked, koje može biti true ili false.

Input element ima sledeće evenete na koje je moguće reagovati:

- **onclick** – klik miša
- **onchange** – promena vrednosti
- **onfocus** – kontrola je dobila fokus
- **onmouseover** i onmouseout – pokazivač miša je ušao ili izašao u region kontrole
- **onkeydown** – korisnik je pritisnuo taster
- **onkeyup** – korisnik je pustio taster

SELECT ELEMENT

Select HTML element ima polje options, koje sadrži niz options elemenata

HTML Select element ima polje options, koje sadrži niz options elemenata, i polje selectedIndex koje sadrži indeks izabranog elementa u prethodnom nizu.

Sledeća skripta demonstrira kako naći trenutno selektovanu stavku iz select elementa:

```
var e = document.getElementById("ddlViewBy");
var sValue = e.options[e.selectedIndex].value;
var sText = e.options[e.selectedIndex].text;
```

DODAVANJE NOVOG INPUT ELEMENTA U STRANU

InnerHTML polje bilo kojeg HTML elementa predstavlja vrednost izmedju otvorenog i zatvorenog taga elementa

InnerHTML polje bilo kojeg HTML elementa predstavlja vrednost izmedju otvorenog i zatvorenog taga elementa. Korišćenjem ovog polja moguće je dodati novi element na željeno mesto u stranu.

Na primer, ako postoji div element koji ima identifikator d, sledeći kod će u taj element upisati novi tekstbox.

```
var d = getElementById("div");
d.innerHTML = "<input type='text' id='textbox' />";
```

▼ Poglavlje 7

Zadaci za samostalni rad: Forme, nizovi

ZADACI ZA SAMOSTALNI RAD 1- FORME

Zadaci za vežbu

Predviđeno vreme za izradu sledećih zadataka je 30 minuta.

- Kreirati stranicu sa dva elementa za unos (ime i prezime) i dugmetom. Klikom na dugme u tabelu na stranici se dodaje novi red. Za svaki red u tabelu dodati i dugme za brisanje reda. (Vreme izrade: 6 minuta)
- Kreirati *select* element koji sadrži nazine boja. Kada korisnik promeni izabranu stavku iz liste, postaviti izabranu boju za pozadinu u stranici. (Vreme izrade: 6 minuta)
- Kreirati stranicu sa *select* elementom i dugmetom. Klikom na dugme pobrisati trenutno selektovanu stavku iz liste. (Vreme izrade: 6 minuta)
- Kreirati stranicu sa tri *checkbox* elementa i dugmetom. Ako su sva tri elementa čekirana, klikom na dugme ispisuje se poruka. (Vreme izrade: 6 minuta)
- Napisati program koji nudi funkcionalnost *to do* liste. Korisnik unosi stavku koju dodaje u listu. Pored svake stavke u listi, ili tabeli prikazuje se *checkbox*. U vrhu strane prikazati ukupan broj stavki koje su označene kao uradjene, tj. čiji je *checkbox* čekiran. (Vreme izrade: 6 minuta)

ZADACI ZA SAMOSTALNI RAD 2 – STRINGOVI I NIZOVI

Zadaci za vežbu 2

Predviđeno vreme za izradu sledećih zadataka je 30 minuta.

- Napisati JS program koji će proveriti da li je uneseni string palindrom. Ignorisati whitespace karaktere. (Vreme izrade: 6 minuta)
- Napisati JS program koji će reči na početku rečenice u unesenom tekstu pretvoriti u velika slova. (Vreme izrade: 6 minuta)
- Napisati JS program koji će proveriti i ispisati prvo slovo koje se ne ponavlja u unesenom stringu. Na primer, u stringu „aaabbbbcaa“ rezultat je karakter „c“. (Vreme izrade: 6 minuta)
- Napisati JS program koji će prebrojati koliko se puta ponavlja svaki karakter u unesenom stringu. (Vreme izrade: 6 minuta)

- Napisati *spellcheck* program koji će crvenom bojom označiti reči koje nisu u rečniku. Kreirati listu od 10 reči koja će se koristiti kao rečnik. (Vreme izrade: 6 minuta)

ZADACI ZA SAMOSTALNI RAD 3

Zadaci za vežbu 3

Predviđeno vreme za izradu sledećih zadataka je 45 minuta.

- Kreirati stranu sa dva elementa za unos teksta i dugmetom. Klikom na dugme tekst iz jednog elementa upisati u drugi element. (Vreme izrade: 5 minuta)
- Kreirati stranu sa elementom za unos teksta, checkboxom i dugmetom. U slučaju da je checkbox čekiran, klikom na dugme ispisati uneseni tekst. (Vreme izrade: 7 minuta)
- Kreirati stranu sa <select> elementom i elementom za unos teksta. Kada korisnik izabere stavku iz select elementa, ispisati tekst izabrane stavke u element za unos teksta. (Vreme izrade: 7 minuta)
- Kreirati stranu dva dugmeta. Klikom na prvo dugme u stranu se dodaje novi element za unos teksta. Klikom na drugo dugme ispisuje se zbir svih brojeva unesenih u elemente za unos. (Vreme izrade: 10 minuta)
- Kreirati stranu sa slikom i checkboxom. U slučaju da je checkbox čekiran, kada korisnik predje pokazivačem miša preko slike, ispisati poruku sa opisom slike. (Vreme izrade: 8 minuta)
- Kreirati stranu sa nekoliko slika i <p> elementom. Kada korisnik predje pokazivačem miša preko slike, u paragraf se ispisuje kratak opis slike. (Vreme izrade: 8 minuta)

✓ Poglavlje 8

Domaći zadatak

OPIS DOMAĆEG ZADATKA - DZ11

Na stranici implementirati skriptu koja računa sumu vrednosti u zadatom nizu.

Očekivano vreme izrade zadatka: 30 minuta

Opis zadatka:

- Koristeći HTML i JavaScript napraviti interaktivni HTML sajt. Sajt treba da sadrži bar tri strane sa elementima koji reaguju na akcije korisnika kao što su klikovi, pomeranje miša, unos teksta itd.
- Početna strana mora da ima alert koji ispisuje informacije studenta (ime, prezime, broj indeksa, omiljena pozdravna fraza) kada se stranica otvori.
- Druga strana mora da ima dugme na koje kada korisnik klikne, naslov ili tekst mora da se promeni.

Zadatak dostaviti kao IT101-DZ11_Ime_Prezime_BrojIndeksa

▼ Zaključak

ZAKLJUČAK

U ovom predavanju bilo je reči o razvoju informacionih tehnologija kroz istoriju sa fokusom na razvoj tehnologija u periodu pojave prvih računara, razvoj hardverskih platform i razvoj računarskih jezika.