BUDUĆIM STUDENTIMA RAČUNARSKIH NAUKA

SA ŽELJOM DA SA LAKOĆOM PRESKOČE PRVI STEPENIK I OTVORE VRATA NOVIM USPESIMA

P R E D G O V O R

Na inicijativu profesora sa Departmana za računarske nauke PMFa U Nišu studenti završne godine OSNOVNIH STUDIJA

DANILO, LAZAR, VUKAŠIN, STEFAN i STRAHINJA

uredili su ovu skriptu ZA VAS, DRAGI NAŠI BUDUĆI STUDENTI, sa željom da vam olakšaju polaganje prijemnog ispita. Jedina naknada koju očekuju za ovaj trud jeste da krenete njihovim stopama, da budete odlični studenti kao i oni, da ih stignete i prestignete U ZNANJU I PROGRAMERSKIM VEŠTINAMA (što vam, neće biti baš lako) I NAJVAŽNIJE, da postanete DOBRI I VREDNI MLADI LJUDI.

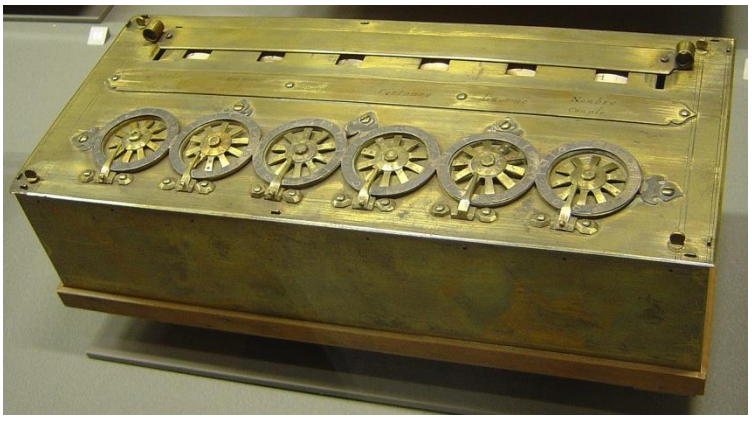
SREĆNO!!!

1. Osnovno o računarima

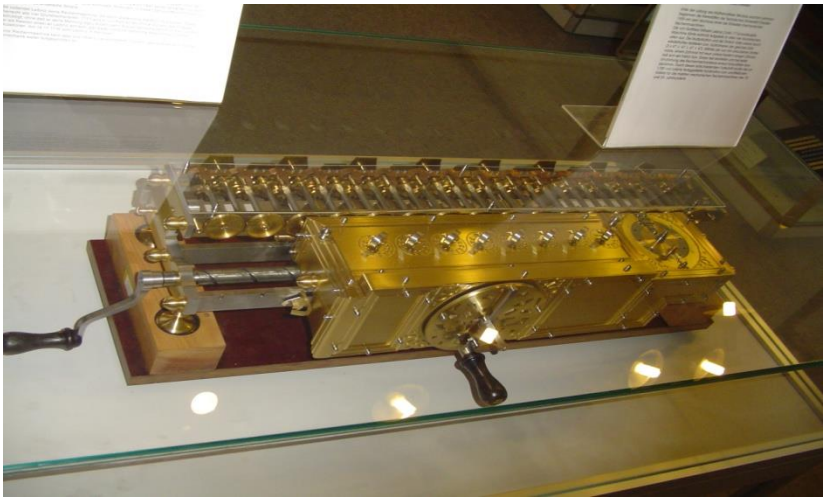
1.1. Kratka istorija računara

Praistorija:

Pre pojave prve generacije računara, još od srednjeg veka, postajalo je nekoliko mašina sposobnih da obavljaju neke od osnovnih matematičkih operacija. Ove mašine predstavljaju, takozvanu, **nultu generaciju** računara. Prva mašina koja je dala značajan doprinos razvoju računara je bio Paskalov računar iz 1942. godine. Blejz Paskal je jako rano ostao bez majke, posle čije smrti se sa ocem i sestrama preselio u Pariz i znanja je uglavnom stekao od oca koji ga je sam podučavao. Sa 19 godina, Paskal je razvio svoju "Paskalinu" da bi pomogao ocu u računanju poreza. Ova mašina je mogla samo da sabira i oduzima, a uzimajući u obzir i cenu same mašine, proizvodnja je stala posle 50 prototipova. Prvi prototip je imao svega nekoliko cifara, dok su kasnije varijante imale i do osam, što je potom omogućilo da se koriste i brojevi do 9.999.999. Cifre su se birale preko metalnih točkića čijim su se okretanjem dobijali odgovarajući brojevi. Odgovori su se pojavljivali u kutijicama na samom vrhu kalkulatora. Pošto su se zupčanici okretali samo u jednom pravcu, negativni brojevi se nisu mogli odmah izračunati. Prilikom sabiranja brojeva korišćen je metod komplemenata od devet. Da bi se pomoglo korisniku, pri unosu broja prikazivao se i njegov komplement iznad kutijice u kojoj je bio napisan uneti broj.



Kasnije, 1671. Gotfrid Vilhelm Lajbnic (Gottfried Wilhelm von Leibniz) je izumeo mašinu za računanje koja je, pored sabiranja i oduzimanja, mogla da množi, deli, kao i da računa kvadratni koren. Množenje je izvršavala ponavljanim pomeranjem i sabiranjem,a deljenje ponavljanim oduzimanjem. Broj sabiranja, odnosno oduzimanja, se kontrolisao sa brojčanikom za množenje. Brojčanik je radio slično onom na telefonu, imao je 10 rupa obeleženih ciframa 0-9. Da bi pomnožili neki broj jednocifrenim brojem, potrebno je samo pritisnuti željenu cifru i povući ručicu za računanje. Brojčanik se onda okreće u smeru kazaljke na satu, po jednom izvršavajući sabiranje za svaku rupu, sve dok se ne vrati na početak. Množenje dvocifrenim brojevima se radilo nešto komplikovanije. Prototip je mogao da radi do 16 cifara, dok su postojale i verzije koje su radile do 12 cifara.



Sledeći veliki korak napravio je Čarls Bebidž (Charles Babbage), koji je izumeo prvi računar koji je mogao da se programira, i zbog uticaja na kasniji razvoj nauke nazvan je ocem računarstva. Bebidžove mašine bili su prvi računari, doduše mehanički, ali ipak istinski računari. U stvari njegove mašine nisu bile završene zbog ličnih i finansijskih problema. Bebidž je uvideo da mašine mogu da rade bolje i pouzdanije od čoveka. Pokrenuo je izgradnju mašine koja je manje-više odrađivala posao i predlagao je mogućnost da se računanje mehanizuje do krajnosti. Iako su Bebidžove mašine bile ogromne njihova struktura je bila slična današnjem računaru. Podaci i programska memorija su bili odvojeni, operacije su bile bazirane na instrukcijama sa čovekove strane. Na Kembridžu je Bebidž uvideo velike probleme grešaka ljudskog računanja i svoj životni rad provodi u pokušajima da izračuna tabele mehanički, uklanjajući sve ljudske greške. 1812. godine uvideo je da se dugačka računanja, pogotovu ona potrebna za izračunavanje matematičkih tablica realizuju nizom već poznatih akcija. Zato je došao na ideju da se ovaj proces automatizuje. Deset godina kasnije razvija mehaničku mašinu koja se zvala diferencijalna mašina. Mogla je automatski da izračunava više matematičkih operacija. Zahvaljujući diferencijalnoj metodi moglo je da se izbegne množenje i deljenje. Prva diferencijalna mašina imala je 25.000 delova, bila je visoka 8 stopa, i teška 15 tona. Mašina je trebalo da radi na paru, da automatski izračunava polinome do šestog stepena, i da štampa rezultate. Iako je imao puno sponzora, nije uspeo da je završi pa je počeo da radi na projektu složenije mašine koja se zvala analitička mašina. Glavna razlika između ove dve mašine je ta da je analitička mašina mogla da bude programirana pomoću bušenih kartica, što je bila ideja ispred njegovog vremena. Shvatio je da nije moglo više programa da stane na jednu karticu, a takođe je morala da bude prisutna i osoba koja bi pravila ostale programe. Analitička mašina je bila programirana da koristi ulančane bušene kartice koje bi kontrolisale mehanički računar, koji je mogao da izračuna naredni rezultat na osnovu nekog prethodno izračunatog rezultata. Mašina je takođe mogla da izvršava naredbe na način nalik onome koji se kasnije koristi kod savremenih računara, uključujući sekvencu, selekciju i iteraciju, što je osnov strukturiranog programiranja.

Atanasov-Beri računar je pionirski izum u kome se po prvi put sreću elementi modernog računarstva, uključujući binarnu aritimetiku i elektronske prekidačke elemente. Od modernih računara razlikuje se po tome što se radilo o mašini specijalne namene, odnosno o mašini kojoj su nedostajali uskladišteni i promenljivi računarski programi. Rad Džona Atanasova na polju računarstva nije bio poznat široj javnosti sve do njegove revizije i revalorizacije tokom 60-ih godina

20. veka kao posledice polemika oko pitanja koji je bio prvi elektronski računar. Do tada se smatralo da je "ENIAC" bio prvi moderni računar, ali je 1973. godine Okružni sud SAD doneo odluku da je to Atanasov-Beri računar.
Džordž Stibic – njegov računar je bio primitivniji od Atanasovljevog, ali je proradio. Rad ove mašine je javno demonstrovan 1940. godine na konferenciji u Darmut Koledžu.
Nemački inženjer Konrad Zuse je radio na izradi mašina za računanje još pre početka Drugog svetskog rata. U periodu od 1936. go 1938. godine izradio je prvi elektromehanički kalkulator poznat kao Z1 koji je imao ograničene mogućnosti programiranja. Ova mašina je potpuno uništena za vreme rata u bombardovanju. Mašina Z2 koju je proizveo 1940. godine predstavljala je poboljšanje mašine Z1, i ona je izrađena od telefonskih releja. Godine 1941. je završio mašinu Z3 za koju mnogi smatraju da predstavlja prvi elektromehanički digitalni računar koji je potpuno pod kontrolom programa. Ovaj računar je uništen 1944. godine u toku rata. Računar je igrađen od 2000 releja, za računanje je koristio binarnu aritmetiku u pokretnom zarezu. Imao je memoriju velikog kapaciteta koja se sastojala od reči dužine 22 bita. Za ulaz podataka i programa su korišćene istrošene filmske trake. Konrad Zuse nije uopšte poznavao radove Čarlasa Babidža, Džordža Bula, ni Alana Tjuringa, jer je bio inženjer a ne matematičar ni naučnik, ali je ipak sam uz pomoć prijatelja konstruisao mašinu za koju je dokazano da ima snagu univerzalne Tjuringove mašine. Konrad Zuse je 1945. godine konstruisao mašinu Z4 koja je jedina preživela rat. On je sa njom pred kraj rata stigao u Cirih u Švajcarskoj. Tamo je osnovao svoju kompaniju koja je 1950. godine proizvela prvi komercijalni računar.
Hauard Ejken sa Harvarda je razvio u saradnji sa IBM-om elektromehanički kalkulator 1944. godine. Nazvan MARK I, ovaj kalkulator je koristio elektromehaničke releje (vrsta električnog prekidača) koji su stimulisani strujom za obavljanje osnovnih aritmetičkih operacija. Bio je dugačak oko 15m, visok 2.4m, dubik 2m, a težio je oko 4.5t. U njega je bilo ugrađeno oko 765.000 komponenata i oko 800km žice. Bio je potpuno automatski, sve operacije je izvršavao bez intervencije čoveka. Mogao je da uskladišti 72 broja, od kojih je svaki mogao da ima 23 decimalne cifre. Mogao je da obavi 2-3 operacije sabiranja u sekundi. Mašina je imala ugrađene programe za izračunavanje nekih matematičkih funkcija. Instrukcija grananja nije postajala.

Istorija računara se deli na generacije računara.

Prva generacija (1945 - 1955)

Motiv za ubrzani rad na razvoju elektronskih računara bio je drugi svetski rat. Prvi računar prve generacije je bio ENIAC (Electronic Numerical Integrator And Computer). ENIAC je imao 20 registara od po 10 dekadnih cifara. Programirao se pomoću 6000(multipozicionih prekidača i šume kablova. Bio je težak 27 tona, zauzimao je 167 m2 , i trošio 150 kW struje. Koristio je četiri akumulatora kontrolisanih posebnom jedinicom množioca i mogao je da obavlja 385 operacija množenja u sekundi. Sa pet akumulatora bilo je upravljano naročitom jedinicom Delioca/Izračunaoca kvadratnog korena i mogao je da obavlja četrdeset operacija deljenja ili tri operacije izračunavanja kvadratnog korena u sekundi. Ostalih 9 jedinica su bile inicijaciona jedinica (pokretala je i zaustavljala mašinu), ciklusna jedinica (sinhronizovala druge jedinice), glavni programer (kontrolisao „krivudavo“ kretanje), čitač (upravljao IBM čitačem bušenih kartica), štampač (upravljao bušačem kartica), stalni prenosnik i tri tablice funkcija. Opis Rohasa i Hašagena (Rojas & Hashagen) pruža više detalja o trajanju operacija, koje se donekle razlikuje od gore

navedenih. Osnovni takt mašine je bio 200 mikrosekundi ili 5.000 ciklusa u sekundi za operacije sa desetocifrenim brojevima. U jednom od ovih ciklusa, ENIAC je mogao da upiše broj u registar, pročita broj iz registra ili da sabere dva broja. Za množenje 10 – cifrenog broja n– cifrenim brojem (za n manje od ili jednako od 10) trebalo je n+4 ciklusa, tako da je za množenje 10 – cifrenog 10 – cifrenim brojem trebalo 14 ciklusa, ili 2800 mikrosekundi – dakle, brzinom od 357 po sekundi. Ukoliko bi jedan broj imao manje od 10 cifara, operacija bi bila tim brža. Deljenje i računanje kvadratnog korena trajalo je 13(n+1) ciklusa, gde je n broj cifara u rezultatu (koeficijent ili kvadratni koren). Tako je deljenje ili kvadratni koren trajalo najviše 143 ciklusa, ili 28600 mikrosekundi – brzina od 35 po sekundi. Ako bi rezultat imao manje od 10 cifara dolaženje do njega zahtevalo bi manje vremena. Osnovne komponente ENIAC– a bile su elektronske cevi korišćene u radio-prijemnicima i ostalim elektronskim uređajima tog vremena. Svaka elektronska cev je imala 8 nožica i ove cevi zvale su se oktalne cevi (po vrsti podnožja). Neki stručnjaci predvideli su da će se kvarovi na cevima javljati tako često da mašina nikada neće biti upotrebljiva. Ova predskazanja ispostavila su se delimično tačnim: nekoliko cevi pregorevalo je skoro svakog dana, ostavljajući mašinu nefunkcionalnom skoro polovinu vremena. Posebne visokopouzdane cevi nisu bile dostupne sve do 1948. Međutim, većina ovih kvarova dešavala se tokom zagrevanja i hlađenja, kada su grejači i katodne cevi bili pod najvećim grejnim pritiskom (termalnim stresom). Zato je uvedeno nekoliko novina: mašina se nikada nije gasila. Ovim veoma jednostavnim (mada izuzetno skupim) rešenjem inženjeri su smanjili broj otkazivanja cevi ENIAC– a na prihvatljivu stopu od samo jedne cevi svaka dva dana. Takođe, napajanje komponenti je smanjeno 10% od nominalnog – smanjenjem napona smanjivala se opterećenost unutar elektronske cevi i time produžavao radni vek. Sve komponente su bile dostupne na prednjoj strani računara što je omogućavalo bržu zamenu elektronskih cevi, te omogućavalo proveru rada računara po paljenju i gašenju pojedinih cevi.

Ekert i Mokli su radili na računaru EDVAC (Electronic Discret Variable Automatic Computer). Ova mašina trebalo je da ima samo desetinu komponenata od kojih je bio sastavljen ENIAC, a da ima sto puta veću memoriju. U konceptu projekta mašine fon Nojman je naveden kao urednik. Ključni koncept kod ove mašine bio je skladištenje programa po kome mašina radi u memoriji. Do tog vremena su se podaci sa kojima je mašina radila i program po kom je radila unosili odvojeno u memoriju. Fon Nojman je predložio da se i program i podaci drže uskladišteni u memoriji računara u isto vreme. Na taj način se program mogao menjati sa istom lakoćom i brzinom kao i podaci. Ovaj princip programiranja računara zadržan je do današnjih dana. Računar EDVAC završen je 1949. godine i bio je prva mašina koja je imala magnetne diskove. IAS mašina i EDSAC su prvi računari sa zapamćenim programom – tj. sa Fon Nojmanovom arhitekturom. Većina današnjih računara ima sličnu arhitekturu. Fon Nojmanova mašina je imala 5 osnovnih delova: memoriju, aritmetičko-logičku jedinicu, upravljačku jedinicu, ulaz, i izlaz. Aritmetika isključivo cjelobrojna, jer je fon Nojman smatrao da će svaki kompetentni matematičar moći da sam odredi poziciju decimalne tačke.

Druga generacija (1955 - 1965)

Drugu generaciju započela je pojava poluprovodničke tehnologije. Prvi tranzistorizovan računar je TX-0 (Transistorized eXperimental computer 0) napravljen u Linkolnovoj laboratoriji na MIT-u. To je bila 16-bitna mašina. Miniračunar PDP-1 se pojavio 1961. godine. Imao je 4096 osamnaestobitnih reči i ciklus instrukcije od 5 milisekundi. Njegove performanse su bile duplo slabije od nabržeg računara tog vremena (IBM 709). PDP-1 je imao prvi CRT video displej sa

ekranom 512x512 tačkica. Cena: 120000 dolara. Godine 1964. je novoosnovana kompanija CDC proizvela model 6600. Ova mašina je skoro za red veličine bila brža od tada moćnog IBM 7094. Tajna njegove brzine ležala je u tome da je njegov procesor bio visoko paralelizovan, odnosno imao je arhitekturu na bazi paralelnog procesiranja, sa ugrađenim manjim računarima. U ovakvoj arhitekturi centralna jedinica je mogla da se bavi samo računanjem, dok su periferne funkcije i komunikacije izvršavali manji računari. Za programiranje tih računara više se ne koristi samo mašinski jezik već i asemblerski jezik, koji je omogućio programerima da instrukcije zapisuju riječima (a ne brojevima, kao što je to bio slučaj u mašinskom jeziku). Takođe u tom periodu nastaju i tzv. viši programski jezici. Prvi takav programski jezik zvao se je Flow-Matic, a iz njega su se kasnije razvili COBOL, FORTRAN, ALGOL i LISP.

Treća generacija (1965 - 1980)

Glavno tehnološko unapređenje računara treće generacije bila je primena integrisanih kola. Tranzistori su bili minijaturizovani i stavljeni u silikonski čip (tranzistori su bili napravljeni na istom parčetu silicijuma; zatim bi to parče silicijuma bilo stavljano u jedno kućište i takav sklop je dobio ime integrisano kolo), što je veoma povećalo brzinu i efikasnost računara. Iz te serije, IBM je napravio čuvenu seriju IBM System 360, sa modelima 30 i 75. Ovi računari imali su upotrebu u naučnim i komercijalnim primenama, pa su zamenili računare IBM 709 i IBM 1401. Uvodi se tehnika multiprogramiranja koji omogućava da se u u memoriji nađe više programa istovremeno, tako da dok se jedan program izvršava, drugi čeka u memoriji da bude izvršen. Više programa ne samo da čekaju na periferije, već dele i vreme procesora.

Sredinom sedamdesetih godina prošlog veka počela je da se primenjuje tehnika u kojoj je operativni sistem delio programe i podatke na blokove jednake veličine (stranice podataka). Prilikom izvršavanja programa operativni sistem je učitavao u centralnu memoriju stranicu po stranicu podataka i izvršavao naredbe. Kada nije više bilo mesta u memoriji, operativni sistem je, na osnovu određenih kriterijuma, jednu stranicu iz memorije upisivao na disk i na njeno mesto učitavao sledeću stranicu. Tako korisnik ima iluziju da veličina centralne memorije nije ograničena. KEŠ memorija je vrlo brza memorija koja se nalazi u samom procesoru. Ova memorija ima višestruko brže vreme pristupa od obične memorije. Zbog toga se u njoj drže podaci koji se često koriste. Operativni sistemi opšte namene mogu se podeliti na: serijske sisteme sa paketnom obradom, multiprogramske sisteme i sisteme sa deljenjem procesorskog vremena Kod sistema programi se prvo čitaju sa čitača kartica, nakon toga izvršavaju i na kraju štampaju. Ove aktivnosti se ponavljaju u toku izvršenja svakog programa. Kod multiprogramskih operativnih sistema dozvoljeno je preklapanje izvršenja aktivnosti tipa unošenje podataka, obrada i generisanje izlaza. Kada CPU treba da čeka neku U/I operaciju (pristup disku ili drugu U/I operaciju) on se prebacuje na izvršenje drugog programa koji nije blokiran zbog U/I. Na ovaj način se dobija veći stepen iskorišćenja CPU-a. "Time-sharing" je oblik multiprogramiranja kod koga postoji interakcija između korisnika i programa (pomoću terminala), dok se program izvršava. Ovo se ostvaruje izvršenjem svakog korisničkog programa za kratak vremenski interval, nakon čega se izvršava drugi korisnički program. Na ovaj način svaki korisnik dobija utisak kao da je jedini korisnik sistema.

Svaki procesor fizički izgleda veoma jednostavno, ali on je u svojoj unutrašnjosti jako kompleksan, jer se radi o stotinama miliona tanzistora koji su smešteni u jednom čipu. Prvi put tako nešto je uspelo 1971. kada je napravljen prvi procesor Intel 4004, koji je doduše mogao samo

sabirati i oduzimati, ali su naučnici prvi put uspeli da u jedan čip smeste silna integrisana kola i tranzistore, što je dalo podsticaj za daljnji razvoj procesora koji su tim napretkom počeli da troše mnogo manje električne energije. Javljaju se i prvi vektorski i protočni računari. Godine 1974. napravljen je prvi superračunar pod nazivom Cray-1. Prvi takav sistem je instaliran u nacionalnoj laboratoriji Los Alamos 1976. godine i postao je jedan od najpoznatijih i najuspešnijih superračunara u istoriji.

Četvrta generacija (1980 - __)

Četvrtu generaciju karakterišu komponente izrađene na bazi poluprovodničkih sklopova korišćenjem LSI (Large Scale Integrated) i VLSI (Very Large Scale Integration) visoko integrisanih sklopova koja omogućava stvaranje mikroprocesora koji predstavlja osnovu današnjih računara. Poboljšanje hardverskih karakteristika dovodi do smanjenja dimenzija računara, povećanja kapaciteta glavne i periferne memorije, znatno brže obrade podataka. Takođe se pojavljuju lični računari (PC-evi) koji predstavljaju pravu revoluciju u demokratizaciji računarske tehnologije, zbog male cene i masovnosti. Tome je doprineo i razvitak grafičkog interfejsa koji je značajno olakšao korišćenje računara do mere da mogu čak i potpuno nestručna lica da ga koriste normalno. Pojavom računarskih mreža i interneta omogućena je neverovatno brza komunikacija i razmena podataka na globalnom nivou.

Peta generacija (u razvoju)

Peta generacija računara je počela da se razvija na inicijativu japanskog Ministarstva za međunarodnu trgovinu i industriju 1982 godine. Ideja je bila da se stvori računar korišćenjem moćnog paralelnog računanja/obrade, kao rezultat velikog državnog/industrijskog istraživačkog projekta u Japanu tokom 1980 godine. Cilj je bio da se napravi "epohalni računar" sa superračunarskim performansama koji bi pružio platformu za budući razvoj vještačke inteligencije. Izraz "peta generacija" je bio namijenjen kao sistemski skok van postojećih mašina. Dok su prethodne generacije računara bile usmjerene na povećanje broja logičkih elemenata u jednom CPU, peta generacije bi se okrenula velikom broju CPU jedinica za dodatne performase. Cilj projekta je bio stvaranje računara u toku desetogodišnjeg perioda nakon čega bi pošelo investiranje u šestu generaciju računara. Mišljenja o rezultatu su podeljena: neki kažu da je projekta bio neuspješan dok drugi tvrde da je bio ispred svog vremena. Očekivanja su bila velika i novi računari je trebalo da svoj rad baziraju na veštačkoj inteligenciji.

PDA (engl. Personal Digital Assistant), kao prvi računari koji uključuju elemente veštačke inteligencije, naročito u funkciji komunikacije sa okolnim svetom predstavlja minijaturni kompjuter, koji staje najčešće na dlan, čije su osnovne namene skladištenje svakodnevnih podataka, razmena e-maila, prenos fajlova, reprodukcija multimedije i dr. Kalendar, adresar i lista tekućih zadataka, pojavom PDA uređaja predstavljaju potpuno rešenje za optimizaciju i lakšu organizaciju vremena. Iako većina korisnika, iskorišćava samo mali broj mogućnosti ovih uređaja, PDA polako ulazi u sferu multifunkcionalnosti, koji uz sve veću ponudu softvera i moćniji hardver mogu omogućiti „krstarenje“ Internetom, prezentacije ili čak bežičnu administraciju udaljenih mreža. Prvi ovakav uređaj napravila je mala engleska firma „Psion“,1984. godine, a uređaj je jednostavno nazvala „Psion 1“. U to doba, to su zaista bile moćne „mašine“, uz adresar, brojeve telefona i listu zadataka, tadašnji PDA su omogućavali i programiranje, na programskom jeziku sličnom *Basic*-u, a mogli su se povezati i na štampače. Najveći nedostatak tadašnjih uređaja bile su skromne mogućnosti ekrana. Naknadnu revoluciju u ovoj oblasti podigao je **Apple** svojim

legendarnim uređajem **Newton Message Pad**, koji je uveo značajnu novost – ekran osetljiv na dodir i tehnologiju prepoznavanja rukopisa. Ono što je mučilo ovaj proizvod jeste relativno visoka cena, komplikovanost, veličina uređaja, te neusavršena tehnologija prepoznavanja rukopisa.
Nevidljivi ugradni računari su počeli da se ugrađuju u razne aplikacije kao što su digitalni časovnici, bankarske kartice i u razne druge proizvode... Dakle, peta generacija se ipak desila, ali na neočekivan način: računari su se počeli smanjivati.
1989 godine firma Grid Systems je stvorila prvi tablet računar koji se zvao GridPad. Sastojao se od malog ekrana na kojme su korisnici mogli pisati posebnom pisaljkom da bi upravljali računarom. Sistem kao što je bio GridPad pokazao je da nije više potrebno sediti za stolom ili u računarskoj sali da bi se koristio računar. Umesto toga, korisnik može koristiti prenosivi računar, displej osetljiv na dodir i softver za prepoznavanje rukopisa. Kasnije mašine ove klase su bili PDA sa poboljšanim interfejsom i postal veoma popularni. Oni su danas evoluirali pametne telefone koji su uključeni u popularne Apple iPhone i Google Android platforme. Ipak nije jasno kada će i dali će uopšte ova generacija potisnuti prethodnu.
Najveća novina u razvoju računara intenzivno proučavanje mogućnosti izrade kvantnih I DNK računara, kod kojih bi tehnologija bila potpuno drugačija u težnji da se povežu znanja iz prirodnih nauka I tehnološki razvoj. Očekuje se da konstrukcijom ovakvih računara, nastane nova tehnološka revolucija.

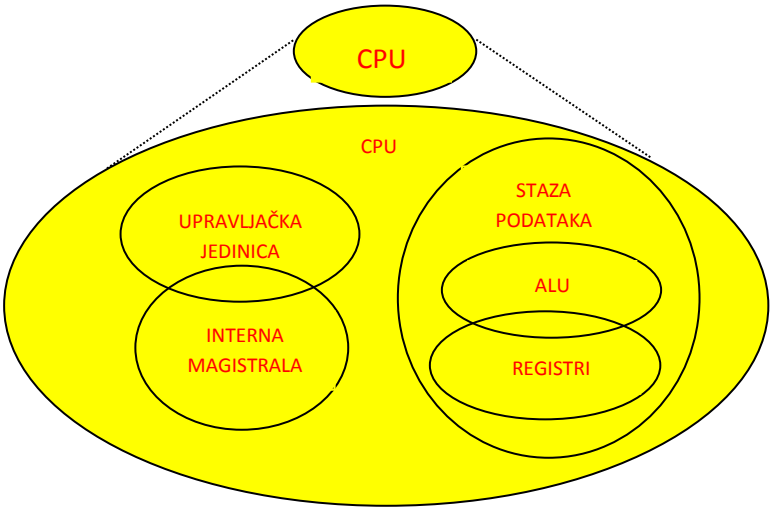
1.2. Osnovne komponente računara

Sve komponente računara dele se u dve osnovne grupe: hardverske i softverske komponente.
Hardver predstavlja samu mašinu, odnosno fizičke delove računara. Hardver čine procesor, memorija, ulazno izlazne jedinice i sistem za povezivanje.
Najbitniji deo računara je svakako procesor, odnosno centralna procesorska jedinica (CPU). Glavne komponente CPU-a su:

- upravljačka jedinica koja upravlja radom CPU-a a time i čitavog računara,
- staza podataka koja obuhvata aritmetičko-logička jedinica (ALU) koja obavlja obradu, tj. različite aritmetičke i logičke operacije nad podacima i registre koji obezbeđuju memorijski medijum unutar procesora,
- interna magistrala CPU-a obezbeđuje komunikaciju između upravljačke jedinice, ALU i registara.

Procesor obrađuje i izvršava mašinski kod (binarni) koji mu govori šta da procesor radi. Jedini razumljivi jezik procesoru je asemblerski jezik. CPU radi tri osnovne stvari:

- Pomoću ALU (eng.Arithmetic/Logic Unit) procesor je u mogućnosti da izvodi osnovne matematičke operacije (sabiranje, oduzimanje, množenje i dijeljenje). Moderni procesori su u mogućnosti da obavljaju i jako komplikovane operacije.
- Procesor prebacuje podatke s jednog memorijskog mesta na drugi
- Shodno naredbama, procesor može skočiti na novi set instrukcija



Sledeća bitna komponenta računara je memorija. Memorija računara može biti unutrašnja i spoljašnja. Unutrašnja memorija je namenjena privremenom pamćenju podataka i programa i procesor joj može direktno pristupiti. Spoljašnja memorija služi za dugotrajno čuvanje programa i podataka. Podela memorije se može izvršiti i na osnovu drugih kriterijuma:

- prema fizičkom načinu zapisivanja podataka (elektronski, magnetni, optički);
- prema metodi pristupa (memorije sa neposrednim pristupom, memorije sa direktnim pristupom, memorije sa sekvencijalnim pristupom, asocijativne memorije);
- prema načinu organizacije (adresne,asocijativne i stek memorije);
- prema trajnosti podataka po nestanku napajanja (postojane i nepostojane memorije);

Najvažnije karakteristike memorije su:

kapacitet -- broj bajtova ili bitova koji se mogu zapamtiti u memoriji;

vreme pristupa -- vremenski interval koji protekne od dovođenja signala za definisanje pristupa do završetka upisa ili čitanja;

brzina prenosa podataka -- broj bitova, bajtova koje uređaj može preneti u jednoj sekundi;

cena bita memorije -- odnos ukupne cene memorije prema kapacitetu memorije;

Sedeći dijagram prikazuje klasifikaciju memorije po nivoima sa primerima za svaki nivo:

Glavna (operativna ili radna) memorija, RAM (Random Access Memory) tj. memorija sa slučajnim pristupom služi da prihvati i čuva ulazne podatke, međurezultate i krajnje rezultate obrade podataka, a takođe služi i za smeštanje korisnikovog programa. Obično se u glavnu memoriju smeštaju podaci koji se obrađuju i instrukcije korisnikovog programa koji se izvršava. Ostali programi i podaci koji se trenutno ne obrađuju nalaze se memorisani na nekoj spoljnoj memoriji. Sadržaj RAM-memorije se gubi po isključivanju računara. Karakteristike ove memorije bitno utiču na performanse računara posebno na brzinu. Ova memorija takođe direktno komunicira sa procesorom koji sve rezultate šalje kroz nju. Većina modernih poluprovodničkih nepostojanih memorija je ili statički RAM (SRAM) ili dinamički ram (DRAM). SRAM zadržava sadržaj sve dok ima struje i ima jednostavan interfejs, ali mu je potrebno šest tranzistora po bitu. Dinamički RAM ima komplikovaniji interfejs i kontrole i potrebni su mu česti ciklusi osvežavanja, kako se uskladišteni podaci ne bi izgubili. Sa druge strane, DRAM koristi samo jedan tranzistor i kondenzator po bitu, što mu omogućava da ima mnogo veću gustinu i da, sa više bitova po čipu, bude jeftiniji. SRAM nije pogodan za sistemsku memoriju desktop računara, gde DRAM dominira, ali se ipak koristi kao keš tih memorija. SRAM je čest u malim, namenskim sistemima, kojima obično treba desetine kilobajta ili manje. Nove tehnologije nepostojanih memorija bi trebalo da se takmiče ili potpuno zamene SRAM i DRAM na tržištu, a u njih spadaju Z-RAM, TTRAM, A-RAM i ETA-RAM.

Memorija samo za čitanje (ROM-Read Only Memory) može samo da se čita, dok je upisivanje u nju nemoguće. Njen sadržaj je upisan prilikom fabrikacije, on se stalno nalazi u ROM-memoriji, tj. ne gubi se ni prilikom gubitka napona napajanja. Zbog osobine da njihov sadržaj ostaje stalno memorisan, tj. trajno je zaštićen, ROM-memorije se koriste za smeštaj važnih informacija, podataka i instrukcija koji su neophodni pri radu računara, npr. bios.

Skrivena ili keš (Cache) memorija je jedna vrsta ultra-brze memorije manjeg kapaciteta i postoji kod računara sa glavnom memorijom velikog kapaciteta. Ona predstavlja lokalnu memoriju procesora. Smisao ove memorije je da se premosti veliki jaz između brzine procesora i glavne memorije. Procesor se u toku rada u većini slučajeva obraća toj memoriji, pa se na taj način postiže veća brzina rada centralne jedinice jer se vreme pristupa glavnoj memoriji efektivno smanjuje oko pet puta. Njen kapacitet obično iznosi oko 5% kapaciteta glavne memorije.

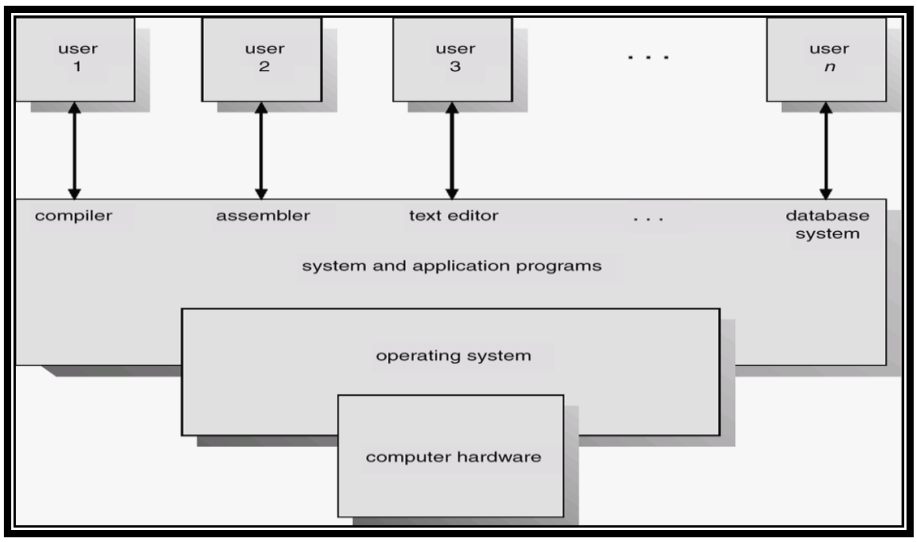
Baferi (buffers) su delovi RAM memorije koje neki programi alociraju za svoje potrebe. Jedna od čestih primena je prilikom ulaza i izlaza podataka. Na primer, ako računar ne može dovoljno brzo da obrađuje podatke koji mu se dostavljaju, oni se privremeno deponuju u bafer dok ne stignu na obradu, da se ne bi prekidao proces unošenja. Slično, pri štampanju, ako štampač ne može dovoljno brzo da odštampa podatke, oni se šalju u bafer (spooler), gde čekaju u redu za štampu.
S obzirom na to da se sadržaj unutrašnje memorije kopira, bajt po bajt na spoljnu memoriju, to se kapacitet spoljne memorije izražava u istim jedinicama kao kapacitet unutrašnje memorije, tj. Brojem bajtova koji može da se uskladišti na spoljnu memoriju. Jedinice spoljašnje memorije mogu biti u obliku hard diska, flopi disketa, optičkog diska, USB memorije, i SSD (Solid State Drive).
Hard disk se sastoji od više ploča premazanih magnetnim materijalom postavljenih na istoj osovini. Brži je i znatno većeg kapaciteta nego disketa. Staze sa istim poluprečnikom sa gornje i donje strane svih ploča čine cilindar po kome se kreću glave za upisivanje i ispisivanje. Značajni parametri za izbor diska su: srednje vreme pristupa podacima, brzina prenosa podataka, i kapacitet diska.
Disketa je okrugla ploča premazana magnetnim materijalom i ugrađena u zaštitno kućište od plastike. Kada se stavi u disketnu jedinicu, disketa se okreće, dok se sa njene gornje i donje strane nalaze upisno-čitajuće glave uređaja kojima se vrši i upis na ploču i čitanje sa nje. Budući da se ploča okreće, ispod položaja glave za upis i ispis kada ona miruje nastaje kružnica koja se naziva staza. Kružnica je podeljena na sektore dužine 512 bajtova. Staze i sektori nisu vidljivi, nego su samo zapisani na magnetnom materijalu. Da bi se upisale staze i sektori, disketa se mora pre prve upotrebe formatirati. Flopi diskete imaju veoma mali kapacitet, i one su odavno prevaziđeni oblik memorije.
Optički diskovi zasnovani su na istom principu kao disk i disketa, ali je razlika u tehnologiji realizacije i kapacitetu. Kod ovih diskova se umesto namagnetisanja magnetnog materijala nanetog na kružnu ploču, na samoj magnetnoj ploči nanose zapisi korišćenjem laserskog zraka. Pošto se ovim postupkom površina diska trajno oštećuje, jednom upisani podaci ne mogu se više menjati, već samo očitavati. Postoje i diskovi koji se mogu očistiti od podataka i napuniti drugim podacima, ali je za to potreban nešto osetljiviji laser. Ovi diskovi se mogu "prerezati" i do 100.000 puta zbog manje dubokog urezivanja podataka i površinu diska.
USB memorije su lako prenosive i mogu se povezati sa bilo kojim računarom nezavisno od operativnog sistema, veličine, ili namene. Njihov kapacitet može dostići i kapacitet hard diska, a po brzini mogu biti i brže nego hard diskovi.
SSD uređaji imaju istu ulogu kao hard diskovi, ali su značajno brži i skuplji, i zbog cene imaju malo manji kapacitet. Ovi uređaji koriste integrisana kola kao memoriju, i koriste elektronski interfejs kompatibilan sa tradicionalnim ulazno-izlaznim blokovima na hard disku. Razlika u brzini se postiže jer nema pokretnih mehaničkih delova (kao glava kod hard diska), što takođe znači da su otporniji na fizičke šokove (udarce, potrese, itd), i znatno su tiši. Takođe postoje i SSHD (Solid State Hybrid Drive) koji imaju veliki hard disk i SSD deo memorije koji služi kao keš, i na taj način se najkorišćeniji podacima brže pristupa.

Pored same mašine računara (hardvera), računarski sistem čine i programi koji se izvršavaju (softver). Softver su programi (naredbe, instrukcije) koji govore računaru kako treba da izvršava određene zadatke. Softver je način zapisa algoritama u obliku koji je razumljiv računaru. Povezan je sa hardverom, koji predstavlja komponente računara. Ukoliko hardver i softver nisu povezani, računar ne funkcioniše. Softver se sastoji od programa i biblioteka, kao i dokumenata koji su povezani sa njima. Pojam softvera se često koristi u užem smislu, kao softver aplikacija. Izvršni kod se sastoji od programskog tj. mašinskog jezika sa instrukcijama specifične za svaki

procesor (engl. central processing unit – CPU). Mašinski jezik čine grupa binarnih vrednosti koje prosleđuju instrukcije do procesora o promeni stanja računara. Kao primer toga imamo kada instrukcija menja vrednost sačuvana u posebnoj lokaciji računara, i u tom slučaju rezultat promene nije vidljiv korisniku. Takođe, instrukcije mogu dovesti do promene na ekranu računara, koji su, dakle, vidljivi korisniku. Procesor će izvršavati radnje sa instrukcijama po redosledum po kom su mu instrukcije dostavljene, ukoliku mu nije zadato da preskoči sa jedne na drugu instrukciju. Softver zapisan u mašinskom jeziku je poznat kao mašinski kod, dok je u praksi najčešće zapisan kao skup programskih jezika visokog nivoa, zbog toga što su mnogo efikasniji, a takođe i jednostavniji za korišćenje čoveku nego mašinskom jeziku. Ovakvi programi se prevode u mašinski jezik pomoću kompajlera ili interpretatora, kao i kombinacijom ova dva programa. Softver takođe može biti zapisan u programskom jeziku niskog nivoa – asembleru, programu koji predstavlja mašinski jezik korišćenjem alfabeta.

Najvažniji je sistemski softver, koji je napravljen za direktnu komunikaciju sa hardverom, i na taj način omogućio korisnicima osnovne funkcije, ali i drugim softverima da osposobi platformu za aktivne aplikacije. Sistemski softver sadrži operativni sistem kao najvažniji deo softvera,i uz njega drajvere i pomoćne programe koji služe za lakše održavanje operativnog sistema i računara.

Operativni sistem je zapravo kolekcija softvera napravljena za upravljanje resursima i koja omogućava zajedničke usluge ostalim softverima. Nadzorni programi, pokretač operativnog sistema i višeznačna jedinica (ljuska – shell) su ključni delovi operativnog sistema. U praksi, operativni sistem dobijamo sa dodatnim softverom (uključujući i softver aplikacija). Drajveri (upravljački programi) omogućavaju funkciju određenom uređaju, ili određenoj vrsti uređaja prikačenog na računarski sistem. Svakom prikačenom uređaju je potreban bar jedan drajver koji povezuje uređaj sa operativnim sistemom. Pored sistemskog softvera postoje i aplikativni softveri koje računar koristi za izvršavanje specijalnih funkcija ili za zabavu (operacije koje ne uključuju osnovne operacije računara). Postoji mnogo vrsta softvera aplikacija, zbog velikog spektra zadataka koje moderan računar izvršava.



Što se tiče poznatih operativnih sistema, nekada se najviše koristio DOS, dok se posle toga koristio Windows 3.x, 95, 98, NT, 2000, XP, Vista... Danas se najčešće koriste Windows 7, 8, 8.1, 10. Pored ovih operativnih sistema postoje i besplatni operativni sistemi kao što su UNIX i varijacije Linux-a

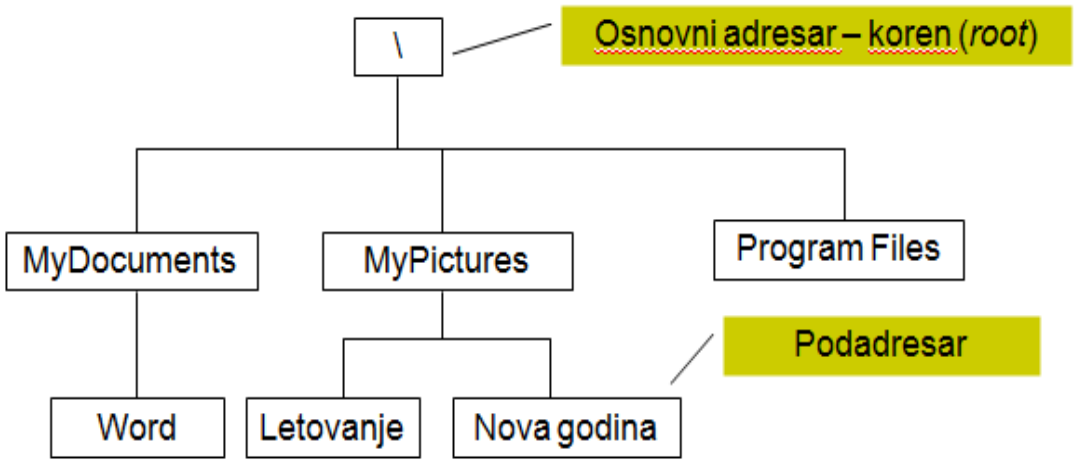
koji se neprestano nadograđuje i ima besplatnu varijantu za većinu plaćenih softvera koji nudi windows.

1.3. Organizacija podataka na računaru

U računaru se sve pamti u vidu datoteka. Datoteka je kolekcija logički srodnih podataka i sadrži program ili podatke. U okviru programa podaci mogu biti deklarisani kao promenljive ili konstante. Promenljive mogu biti prostog ili složenog tipa. Datoteke mogu i da sadrže različite podatke. Sastoje se od slogova, a slogovi se sastoje od polja. Jedno polje obično sadrži podatak nekog prostog tipa. Datoteka ima svoje atribute, atoteka ima atribute, od kojih su najvažniji ime datoteke, veličina, datum kreiranja ili poslednje modifikacije i dr. Ime datoteke sadrži ime te specifične datoteke i ekstenziju (nastavak). Na osnovu nastavka se može zaključiti tip datoteke i na osnovu tipa iskoristiti datoteku adekvatno. Datoteke koje sadrže izvršne programe imaju nastavak exe, com itd. dok datoteke koje sadrže podatke imaju nastavke koji su karakteristični za aplikacije koje su ih kreirale. Za otvaranje ili rad sa datotekom određene ekstenzije neophodan je specijalan softver koji može sa tom ekstenzijom da radi. Neke od poznatijih ekstenzija su:

- **doc** – Word dokument.
- **xls** – Excel tabela.
- **cdr** – CorelDraw crtež.
- **ppt** – PowerPoint prezentacija.
- **jpg**, **png** ... - slike
- **rar**, **zip** ... - arhive
- **pcx** – PaintBrush slika.
- **mp3** – audio zapis u datom formatu, i dr.

Pored datoteka u računaru možemo zapamtiti i folder. Folder predstavlja logičku celinu grupisanih datoteka. Folderi imaju svoje ime, ali nemaju nikakav nastavak. Hijerarhijski su uređeni u vidu stabla. U foldere se mogu organizovati datoteke po želji.



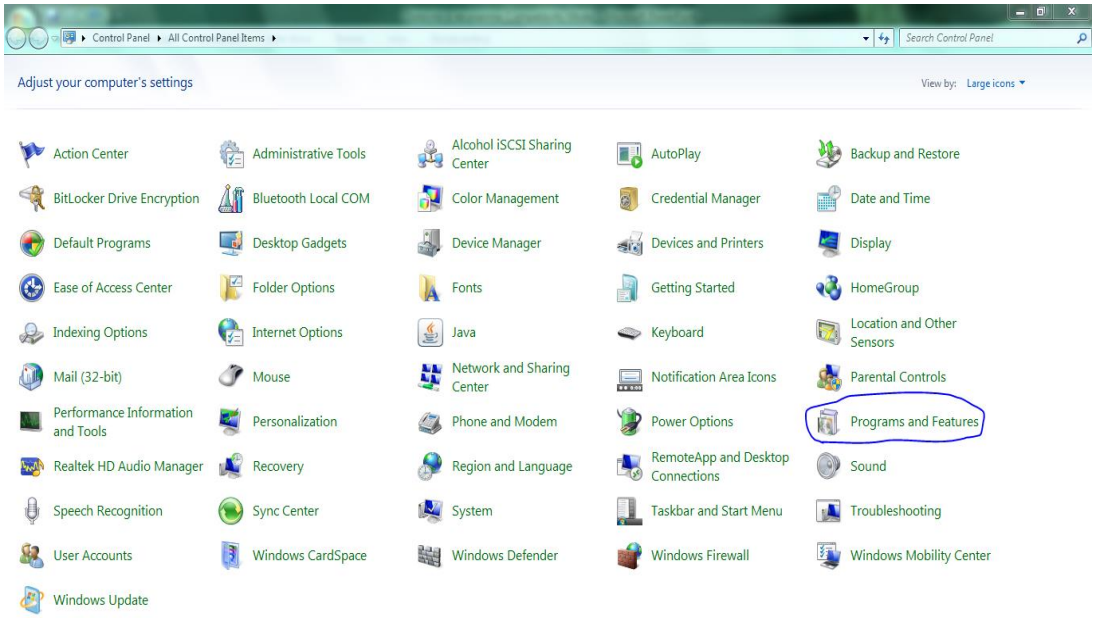
Ime puta neke datoteke ili foldera može biti apsolutno ili relativno.

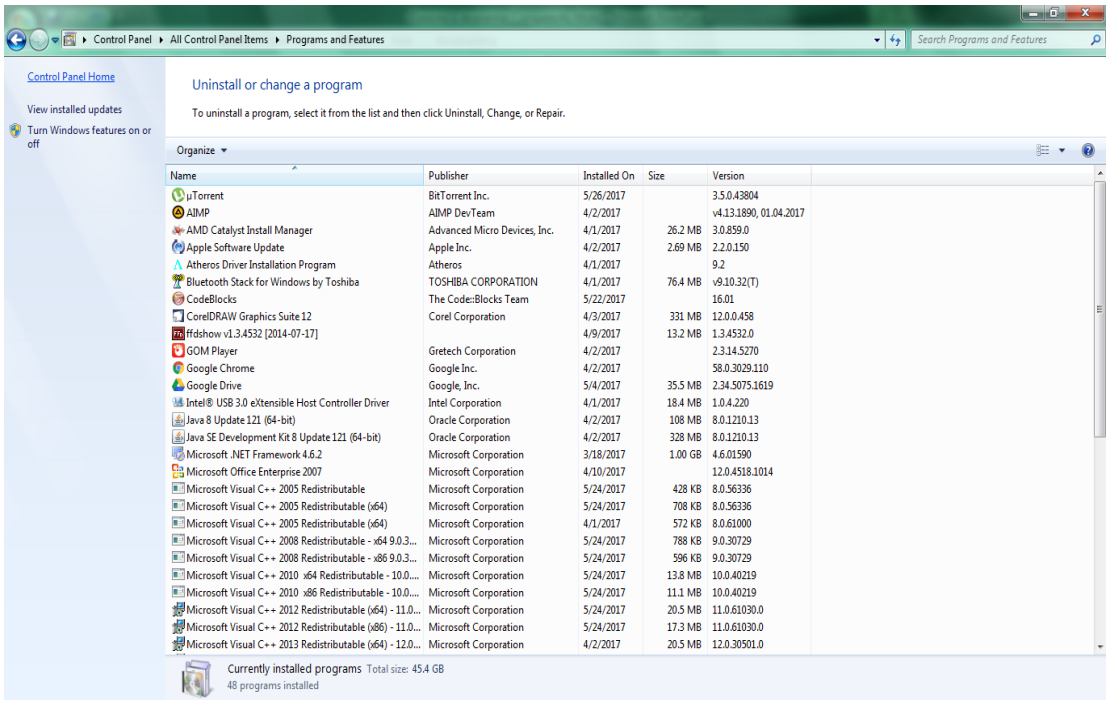
Apsolutno ime puta sadrži popis svih adresara koje treba proći u stablu adresara da bi se stiglo do određenog fajla ili foldera. Ako bi na prethodnom primeru pokazali apsolutno ime puta za folder

Nova godina, taj put bi izgledao ovako: **\MyPictures\Nova godina**. Dok nasurpot tome, relativno ime puta sadrži popis samo onih adresara koji se nalaze između tekućeg i ciljnog. Relativno ime puta za ovaj primer ako je tekući folder MyPictures bi bio samo Nova godina.

1.4. Najčešće korišćeni programi

Pre svega sve instalirane programe na Windows operativnom sistemu možemo videti u Control Panel-u, pa na opciju add or remove programs za XP ili programs and features opciju za novije verzije. U tom prozoru možemo videti, menjati i deinstalirati sve instalirane programe.





Postoji mnogo vrsta različitih softverskih programa koji mogu biti instalirani, ali ovde ćemo pomenuti neke koje većina koristi svakodnevno. Što se tiče kancelarijskih i poslovnih softvera, najčešće se koristi Microsoft Office paket programa, ili besplatne verzije toga Lybre Office i OpenOffice. Linux je besplatan softver koji je preuzeo primat u mnogim Evropskim zemljama, dok apsolutni primat u Americi ima IOS.

Što se tiče obrade teksta, koristi se Microsoft Word, a pored toga se koriste i Apple Pages, OpenOffice Writer, notepad i notepad++. Word pruža mnogo opcija za pravljenje tekstualnih dokumenata i to sa dodacima slika, grafika, tabela, itd. Nudi nam razne opcije za menjanje izgleda slova, formata strane,boje,veličine, pa čak i obradu slika u nekoj meri. Za računanje po matematičkim formulama i grupisanje u tabele najkorišćeniji je Microsft Excel koji nudi opcije za veoma velii broj matematičkih funkcija, kao i za njihove kombinacije. Takođe je moguće bojiti određena polja u tabeli drugačije , kao i ostale manipulacije sa izgledom i strukturom tabela. Alternative bi bile Apple Numbers i OpenOffice Calc. Za prezentacije se koriste Microsoft Power Point, Apple keynote ili OpenOffice Impress.

Sledeća kategorija aplikacija su internet aplikacije. Postoji mnogo internet pretraživača (browsera), a najkorišćeniji su: Mozilla Firefox, Google chrome, Microsoft Edge, Opera, ili za IOS pretraživač pod nazivom Apple Safari. Svi pružaju slične opcije, uz male razlike. Za slanje elektronske pošte koriste se uglavnom Microsoft outlook, Mozilla thunderbird, kao i Apple Mail.

Za rad sa računarskom grafikom i slikama, najčešće se koriste Corel Draw i Adobe photoshop i Adobe Illustrator, ili besplatne varijacije toga GIMP i Inkscape.

2. Pamćenje podataka na računaru

2.1. Jedinice za količinu informacija

Bit je najmanja jedinica za informacije u računarstvu. Jedan bit predstavlja količinu informacije potrebnu za razlikovanje dva međusobna isključiva stanja. Naziv bit potiče od skraćenice engleskih reči Binary Digit (binarna cifra). Kao što samo ime sugeriše, bit može imati samo dve vrednosti, 0 ili 1 (u fizičkom smislu 1 označava da je određeno električno kolo pod višim naponom, a 0 da je pod nižim). Naravno, s obzirom na to da jedan bit može uzeti samo jednu od dve vrednosti, jedan bit ne može sačuvati mnogo informacija (jedan bit može, na primer, sačuvati neki odgovor tipa da/ne). Zbog toga je praktičnije grupisati bitove. Niz od 8 bitova naziva se **bajt**. S obzirom da svaki od bitova u okviru bajta može biti 0 ili 1, to znači da jednim bajtom možemo predstaviti 2^8 različitih nizova bitova (na primer, jedan bit može dati odgovor na pitanje da li je određeni predmet siv. Ali ako posmatramo bajt, onda možemo dati precizan odgovor o konkretnoj nijansi sive boje tog objekta – 256 nijansi sive). Uobičajena oznaka za 1 bajt je 1B (dakle veliko slovo „B“).
Postoje, naravno, i veće jedinice za količinu informacija (veće jedinice memorije). U dekadnom sistemu uobičajeno je dodavanje prefiksa ispred jedinice mere da bi se naznačio red veličine. Na primer, kilo je uobičajeni prefiks koji označava 1.000* jedinica mere (1 kilogram = 1000 grama). Slično „mega“ označava 1.000.000*jedinica mere (na primer, megatona = 1.000.000 tona). Primetimo da je u svim slučajevima u pitanju prefiks koji se tiče reda veličine koji je stepen broja 10.
Slični princip važi i kod jedinica memorije. Međutim, s obzirom na korišćenje binarnog sistema, praktičnije je uvoditi prefikse u odnosu na odgovarajuće stepene broja 2. Tako se prefiksi dodaju za red veličine $2^{10} = 1.024 \sim 1.000$. Tako imamo da je, na primer, 2^{10} bajta = 1 kilobajt, 2^{10} kilobajta = 1 megabajt = 2^{20} bajta.
Dakle, najčešće jedinice memorije veće od bajta su:

- Kilobajt = 1 kB = 2^{10} B
- Megaobajt = 1 MB = 2^{10} KB = 2^{20} B
- Gigabajt = 1 GB = 2^{10} MB = 2^{30} B
- Terabajt = 1 TB = 2^{10} GB = 2^{40} B

2.2. Brojni sistemi i prevođenje brojeva

Brojni sistemi se mogu podeliti na *pozicione i nepozicione*.

Kod **nepozicionih brojnih** sistema pozicija cifre nije u direktnoj vezi sa njenom težinom. Primer nepozicionog brojnog sistema je rimski brojni sistem. Na primer, ako posmatramo brojeve VII i XIX, druga cifra u oba broja je I (1), ali u broju VII ona ima vrednost +1, dok u broju XIX ima vrednost -1. Nepozicioni brojni sistemi se danas ne koriste i nemaju praktični značaj u računarstvu.
Kod **pozicionih brojnih** sistema se brojevi predstavljaju nizom od *n* cifara, pri čemu je svakoj cifri pridružena *težina* saglasna njenoj poziciji u nizu. Svaki pozicioni brojni sistem karakteriše se *osnovom* (*bazom*) brojnog sistema *r* (mogu se naći i oznake N ili B). Cifre brojnog sistema sa osnovom *r* uzimaju vrednosti iz skupa {0, 1, ..., *r* – 1}.

Vrednost n-tocifrenog celog broja A u sistemu sa osnovom r prikazanog nizom cifara $a_{n-1}a_{n-2}\dots a_2a_1a_0$ gde $a_i \in \{0, 1, \dots, r-1\}$ može se odrediti kao $|A| = \sum_{i=0}^{n-1} a_i\, r^i$.
Ukoliko je A razlomljeni broj njegova vrednost računa se formulom $|A| = \sum_{i=-m}^{n-1} a_i\, r^i$, pri čemu smatramo da je broj A dat nizom cifara oblika $a_{n-1}a_{n-2}\dots a_2a_1a_0, a_{-1}a_{-2}\dots a_{2-m}a_{1-m}a_{-m}$.
Dakle, u celom delu broja je n cifara, a u razlomljenom m (za broj 3.14159265, n=1, m = 8).

Primer 1. Ako je dat broj u sistemu sa osnovom 2 (binarni sistem) koji je dat nizom cifara 10010011, njegova vrednost je?

Rešenje: Na osnovu formule $|A| = \sum_{i=0}^{n-1} a_i\, r^i$, imamo:

$$1 * 2^7 + 0 * 2^6 + 0 * 2^5 + 1 * 2^4 + 0 * 2^3 + 0 * 2^2 + 1 * 2^1 + 1 * 2^0 = 128 + 16 + 2 + 1 = 147$$

Primer 2. Neka je dat isti niz cifara binarnog brojnog sistema: 11110011,111001. Treba odrediti njegovu vrednost.

Rešenje: Vrednost celog dela broja već znamo da odredimo na osnovu primera 1. Za ovaj primer, vrednost binarnog broja 11110011 iznosi 243.
Vrednost razlomljenog dela računamo na sledeći način:

$$1 * 2^{-1} + 1 * 2^{-2} + 1 * 2^{-3} + 0 * 2^{-4} + 0 * 2^{-5} + 1 * 2^{-6} = \\ = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{64} = 0.890625$$

Dakle, vrednost broja 11110011,111001 bi bila 243 + 0.890625 = 243. 890625.
Ljudi najčešće koriste *decimalni sistem*, koji je, međutim, nepogodan za korišćenje od strane računara. U računarskim sistemima češće se koriste oktalni (*r* = 8) i heksadekadni sistem (*r* = 16), a najčešće binarni sistem sa osnovom dva (jer digitalna kola koja izvode operacije nad brojevima koriste dva stanja). Zbog razlike između sistema koji najčešće koristi čovek i onih koje koristi računar mora postojati način konverzije između ovih brojnih sistema.
U osnovi, razlikujemo dva postupka prevođenja brojeva iz brojnog sistema sa osnovom r_1 u brojni sistem sa r_2. Jedan je kada se operacije izvršavaju u brojnom sistemu sa osnovom r_2 (tj. ciljanom brojnom sistemu), a drugi je kada se operacije izvršavaju u brojnom sistemu sa osnovom r_1 (polaznom brojnom sistemu).
Prevođenje brojeva kod koga se operacije izvršavaju u ciljanom brojnom sistemu se koristi pri prevođenju broja u dekadni brojni sistem. Postupak se svodi na određivanje vrednosti broja na osnovu datog niza cifara.

Tj. $(a_{n-1}a_{n-2}\dots a_2a_1a_0)_r = (X)_{10}$, gde je $X = \sum_{i=-m}^{n-1} a_i\, r^i$.

Dakle, postupak prevođenja u dekadni sistem iz binarnog se može videti iz primera 1 i 2 (jer je postupak isti kao i određivanje vrednosti tih brojeva). Prevođenje brojeva iz oktalnog ili heksadekadnog u decimalni se takođe može izvriti na osnovu gore navedene formule.
Oktalni sistem koristi 8 cifara za predstavljanje brojeva (0, 1, 2, 3, 4, 5, 6, 7), dok **heksadecimalni** koristi 16 znakova (0, 1, 2, 3, … , 8, 9, A, B, C, D, E, F).

Primer 3. Prevesti broj A14.F u decimalni zapis.

Rešenje: Na osnovu formule $X = \sum_{i=-m}^{n-1} a_i\, r^i$, imamo:

$$X = 10 * 16^2 + 1 * 16^1 + 4 * 16^0 + 15 * 16^{-1} = 2560 + 16 + 4 + 0.9375 = 2580.9375$$

Česta je sledeća notacija: $(\mathrm{A14.F})_{16} = (2580.9375)_{10}$ ili samo $(\mathrm{A14.F})_{16} = 2580.9375$ (ukoliko nije navedena baza u indeksu, podrazumeva se 10, tj. da se radi o dekadnom brojnom sistemu.

Kada se prevode brojevi kod kojih se operacije izvršavaju u polaznom brojnom sistemu može se razlikovati prevođenje celih i prevođenje razlomljenih brojeva.

Prevođenje celih brojeva vrši se celobrojnim deljenjem broja koji želimo da prevedemo osnovom ciljanog brojnog sistema (r_2). Deljenjem se dobija ceo broj i ostatak. Proces deljenja osnovom r_2 se nastavlja dok god ceo deo količnika ne bude jednak nuli. Ostaci pri deljenju predstavljaju cifre broja u sistemu osnove r_1 gde ostatak pri poslednjem deljenju odgovara prvoj cifri broja (cifra najveće težine), pri predposlednjem drugoj itd sve do cifre najmanje težine kojoj odgovara ostatak pri prvom deljenju.

Primer 4. Prevesti broj 555 u binarni zapis:

Rešenje: Ciljani brojni sistem je binarni (ima osnovu 2), pa ćemo vršiti celobrojno deljenje brojem 2.

555: 2 = 277 => ostatak = 1

277: 2 = 138 => ostatak = 1

138: 2 = 69 => ostatak = 0

69: 2 = 34 => ostatak = 1

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

Sada binarnu reprezentaciju broja dobijamo tako što "isčitamo" ostatke odozdo naviše, tj. $(555)_{10} = (1000101011)_2$

Prevođenje razlomljenih brojeva vrši se množenjem osnovom ciljanog brojnog sistema (r_2). Ceo deo dobijenog proizvoda je prva cifra posle tačke osnove. Razlomljeni deo dobijenog proizvoda se dalje množi osnovom r_2 a ceo deo dobijenog proizvoda biće druga cifra itd. Proces se nastavlja dok god razlomljeni deo proizvoda ne bude jednak nuli (u nekim slučajevima nije moguće precizno prevesti broj, pa se process prekida nakon određenog broja cifara u zavisnosti od tačnosti koja nam je potrebna).

Primer 5. Predstaviti broj 0.625 u binarnom zapisu.

Rešenje: Dakle, kao što je ranije opisano, prevođenje razlomljenog broja u binarni vrši se množenjem osnove ciljanog sistema. Pošto se ovde radi o prevođenju u binarni sistem, množićemo brojem 2.

0.625 * 2 = 1.250 (ceo deo = 1, razlomljeni deo = 0.25)

0.25 * 2 = 0.5 (ceo deo = 0, razlomljeni deo = 0.5)

0.5*2 = 1.0 (ceo deo = 1, razlomljeni deo = 0)

Kako smo dobili da je razlomljeni deo jednak nuli, ovde stajemo. Razlomljeni deo sada predstavljamo binarno tako što samo poređamo cele delove koje smo dobili tokom množenja počevši od prvog, pa nadalje (dakle, čitamo bitove odozgo na dole), odnosno $(0.625)_{10} = (101)_2$.

Već je rečeno da se u računarstvu često koriste heksadekadni i oktalni sistem, a da se sve operacije u računaru svode zapravo na operacije u binarnom sistemu. Konverzije između binarnog i oktalnog ili heksadecimalnog sistema mogu se izvršiti gore navedenim metodama, ali jednostavnije je grupisanjem bitova bitova (kod konverzije u heksadekadni ili oktalni), odnosno prevođenjem svake cifre oktalnog ili heksadekadnog u binarnu reprezentaciju date cifre (kod konverzije u binarni sistem). Kod prevođenja cifara iz oktalnog ili heksadecimalnog u binarni bitno je da svaku cifru predstavimo sa tačno 3 (za oktalni), odnosno tačno 4 (za heksadecimalni) cifara. Na primer, ako treba prevesti heksadecimalnu cifru 2 u binarnu reprezentaciju, predstavićemo je kao 0010, a ne kao 10 (dakle, iako nam "ne treba" 4 cifara, cifru ćemo ipak predstaviti korišćenjem sve 4. Jedini izuzetak kada nije neophodno koristiti sve 4 cifre je u slučaju da prevodimo baš prvu cifru traženog broja – onda možemo izostaviti tzv. vodeće nule).

Primer 6. Prevesti sledeće brojeve u binarni zapis:

a) $(123)_8$
b) $(123)_{16}$
c) $(3AF.20C)_{16}$
d) $(430.03)_8$

Rešenje: Konverziju vršimo tako što jednostavno predstavimo svaku cifru binarnog ili oktalnog broja pomoću bitova koji odgovaraju vrednosti te cifre.

a) Dakle, treba predstaviti svaku cifru pomoću bitova koji predstavljaju vrednost te cifre. Tako će $(123)_8$ u binarnom sistemu biti 001010011, odnosno 1010011 bez vodećih nula. Do rezultata smo došli na osnovu toga što vrednost cifara: 1 = 001, 2 = 010, 3 = 011 (za određivanje vrednosti broja videti primer 1).
b) Procedura je slična kao u prethodnom primeru, s tim što sada svaku cifru predstavljamo pomoću 4 bita. Dakle, $(123)_{16} = (000100100011)_2 = (100100011)_2$
c) Kod razlomljenih brojeva procedura je ista. Dakle, imamo:
$$(3AF.20C)_{16} = (001110101111.001000001100)_{16}$$
$$= (1110101111.0010000011)_{16}$$
d) $(430.03)_8 = (100011000.000011)_2$

Kod **konverzije binarnog broja u oktalni (heksadekadni)** grupišemo cifre u grupe po tri (četiri) počev od binarne tačke, i zatim svaku cifru izrazimo jednom oktalnom (heksadekadnom) cifrom. Ukoliko za grupisanje nedostaje jedan ili više bitova, broju se mogu dopisati po potrebi prateće ili vodeće nule.

Primer 7. Sledeće binarne brojeve prevesti u oktalni i heksadecimalni zapis:

a) 101101101
b) 11001010.101010001

Rešenje: Za konverziju u oktalni zapis grupisaćemo bitove u grupe od po 3 bita, a za prevođenje u heksadecimalni zapis, bitove ćemo grupisati u grupe od po 4 bita. Grupisanje kod celog dela broja se vrši sa desna na levo počevši od decimalne tačke (da bi se po porebi dodale vodeće nule), a kod razlomljenog dela grupisanje je s leva na desno počevši od decimalne tačke (da bi se lako dodale prateće nule, ako je to potrebno).

a) 101 | 101 | 101 => $(555)_8$

0001 | 0110 | 1101 => $(16D)_{16}$ – primetimo da je ovde bilo potrebno da dodamo 3 vodeće nule

b) 011 | 001 | 010 **.** 101 | 010 | 001 => $(312.521)_8$ – dodata je jedna vodeća nula u binarnom zapisu

1100 | 1010 **.** 1010 | 1000 | 1000 => $(CA.A88)_{16}$ – dodali smo 3 prateće nule

2.3. Binarna aritmetika

Aritmetika sa binarnim brojevima obavlja se na isti način kao i aritmetika decimalnih brojeva. Razlika je, naravno, u tome što je kod binarne aritmetike osnova sa kojom radimo 2, a jedine cifre su 0 i 1.

Binarno sabiranje i oduzimanje

Kao što je rečeno, algoritmi osnovnih aritmetičkih operacija su isti kao kod dekadnog sistema. Razlika se ogleda u tome što radimo sa osnovom 2, pa je 1 + 1, zapravo jednako 0, pri čemu „prenosimo“ jednu jedinicu (slično situaciji u decimalnom sistemu u kojoj je 5 + 5 = 0, a 1 prenosimo). Kod oduzimanja važi da je, naravno, 1 – 0 = 1, ali 0 – 1 je takođe 1 (jer smo izvršili „pozajmicu“ od cifre s leva).

Primer 8. Izračunati u binarnom sistemu:

a) 100100111 + 100100110
b) 100100111 – 100100110
c) 100111101 – 1011

Rešenje:

a) **1 1 1 1 - bitovi prenosa!**

```
  1 0 0 1 0 0 1 1 1
+ 1 0 0 1 0 0 1 1 0
  1 0 0 1 0 0 1 1 0 1
```

b)

```
  1 0 0 1 0 0 1 1 1
– 1 0 0 1 0 0 1 1 0
  0 0 0 0 0 0 0 0 1
```

c) **-1 - bitovi pozajmice!**

1 0 0 1 1 1 1 0 1
– 0 0 0 0 0 1 0 1 1
1 0 0 1 1 0 0 1 0

Binarno množenje i deljenje

Binarno množenje i deljenje prate iste algoritme kao i decimalno množenje i deljenje (s' tim što su ove operacije u binarnom sistemu mnogo jednostavnije, nego u decimalnom).

Primer 9. Izračunati 10011 * 101 i 1011010 : 101.

Rešenje:

```
  1 0 0 1 1 * 1 0 1
        1 0 0 1 1
      0 0 0 0 0
+     1 0 0 1 1
      1 0 1 1 1 1 1
```

```
1 0 1 1 0 1 0 : 1 0 1 = 1 0 0 1 0
– 1 0 1
  0 0 1
– 0 0 0
    0 1 0
  – 0 0 0
      1 0 1
    – 1 0 1
        0 0 0
      – 0 0 0
          0 0
```

3. Osnovi programiranja

3.1. Predstavljanje podataka u računaru

Brojevi u računaru mogu se predstaviti na 3 načina: *fiksni zarez* (*fixed point*), *pokretni zarez* (*floating point*) i *binarno kodirani dekadni brojevi* (BCD – *Binary Coded Decimals*).

Aritmetika u fiksnom zarezu se koristi kod problema kod kojih se podaci predstavljaju sa fiksnom pozicijom tačke osnove. Operacije u fiksnom zarezu se mogu podeliti na celobrojne (kada je tačka osnove desno od broja) i operacije sa razlomcima (tačka osnove je levo od broja).

Operacije u pokretnom zarezu se mogu podeliti na *normalizovane* i *nenormalizovane*.

Ako za predstavljanje pozitivnih celih brojeva u fiksnom zarezu koristimo *n* bitova, onda na taj način možemo predstaviti brojeve iz opsega [0, 2^n – 1]. Međutim, potrebno je predstavljati i negativne brojeve. Pri tome treba obezbediti sledeće:

- Podjednaku distribuciju negativnih i pozitivnih brojeva
- Jednostavnu detekciju znaka
- Jedinstven prikaz nule
- Jednostavnu implementaciju aritmetičkih operacija

Obično se za kodiranje znaka broja koristi bit najveće težine (za negativne brojeve ima vrednost *r* – 1, za pozitivne 0; u slučaju binarnog sistema 1 za – 0 za +. Ostale cifre predstavljaju ili pravu vrednost broja ili njen komplement. Postoje četiri načina za predstavljanje celih brojeva: *označena vrednost broja* ili *znak-moduo, nepotpuni komplement, potpuni komplement* i *proširena notacija*.

Kod **moduo-znaka** (naziva se još i **označena apsolutna vrednost** broja) se pozitivni i negativni brojevi razlikuju samo u bitu znaka.

Primer: **0001 1011 = +27;**

1001 1011 = -27.

Međutim, ovde se javlja problem zbog postojanja dva ravnopravna načina za predstavljanja nule: **+0 = 0000 0000 i -0 = 1000 0000.**

Javlja se problem i kod sabiranja brojeva različitog znaka, jer se mora najpre izvršiti poređenje apsolutnih vrednosti sabiraka da bi se odredio znak rezultata.

Nepotpuni komplement se još naziva i *komplement najveće cifre* pa se kod binarnog brojnog sistema naziva još i *jedinični komplement*. Nenegativni celi brojevi (u opsegu od 0 do 2^{n-1} – 1, ako je za predstavljanje upotrebljeno *n* bitova) se i dalje predstavljaju kao u binarnom pozicionom sistemu. Negativni brojevi se komplementiraju do najveće cifre (do jedinice kod binarnog sistema): **+5 = 0000 0101 i -5 = 1111 1010 .** Ни овде приказ нуле није јединствен: **+0 = 0000 0000** i **-0 = 1111 1111.**

Sabiranje brojeva u nepotpunom komplementu odvija se tako što se brojevi saberu, a eventualni prenos na mestu za znak odbaci iz međurezultata i sabere sa cifrom najmanje težine i tako dobije rezultat koji je takođe u nepotpunom komplementu.

Primer:

1100 0000 = - 63
0100 0000 = +64
0000 0000 = nekorektno
+1
0000 0001 = +1 korektno!

Potpuni komplement se još naziva i *komplement osnove*, pa je u binarnom brojnom sistemu njegov naziv i *dvojični komplement*. Nenegativni brojevi se predstavljaju na isti način kao i ranije, dok se prezentacija negativnih brojeva dobija tako što se najpre dobije nepotpuni komplement a onda na mestu najmanje težine doda 1. Na ovaj način se od *n* bitova mogu predstaviti brojevi u opsegu od -2^{n-1} do $2^{n-1} - 1$.

Na primer, **+5 = 0000 0101**, jedinični complement od **+5** je **1111 1010**

+1

-5 = 1111 1011

U potpunom komplementu je moguć jedinstven prikaz nule, a i izvođenje aritmetičkih operacija je pojednostavljeno.

Jedinstveni prikaz nule: **0000 0000 = +0**, jedinstveni complement je **1111 1111**

+1

0000 0000

Brojevi u potpunom komplementu se sabiraju tako što se izvrši sabiranje, a eventualni prenos na mestu za znak se odbaci i tako dobije rezultat u potpunom komplementu.

Primer: **1100 0001 = - 63**

0100 0000 = +64

0000 0001 = +1 **korektno!**

U potpunom komplementu je nemoguće dobiti isti broj pozitivnih I negativnih brojeva. Primetimo, takođe, da je potpuni komplement najmanjeg negativnog broj sam taj broj.

1000 0000

0111 1111

0111 1111

+1

1000 0000

Proširena notacija se za brojeve predstavljene sa *m* bitova naziva još i višak 2^{m-1} jer se svaki broj predstavlja kao zbir njega samog i vrednosti 2^{m-1}.

+5 = 0000 0101

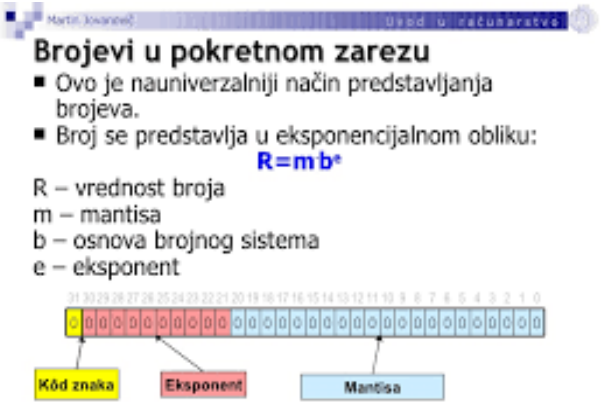
- 5 + 128 = 0111 1011

Практично се врши пресликавање опсега -128 до +127 на опсег од 0 до 255 (за m = 8).

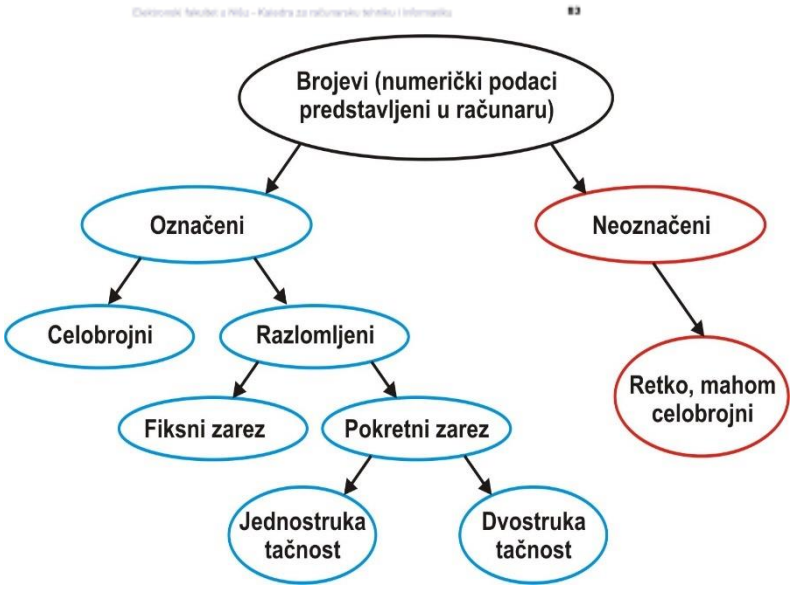
Ovaj sistem je identičan potpunom komplementu sa suprotnim znakom.

Predstavljanje brojeva u pokretnom zarezu i standard IEEE 754

Kod najvećeg broja programskih jezika postoji mogućnost deklarisanja promenljive tipa *real (*ili *float ili double* u nekim jezicima). U računaru se brojne vrednosti ovog tipa predstavljaju u notaciji **pokretnog zareza**. Broj n predstavljen u pokretnom zarezu ima sledeći oblik: $n = (-1)^S * F * R^E$ gde je S - znak, F - mantisa, e - eksponent a R - brojna osnova.

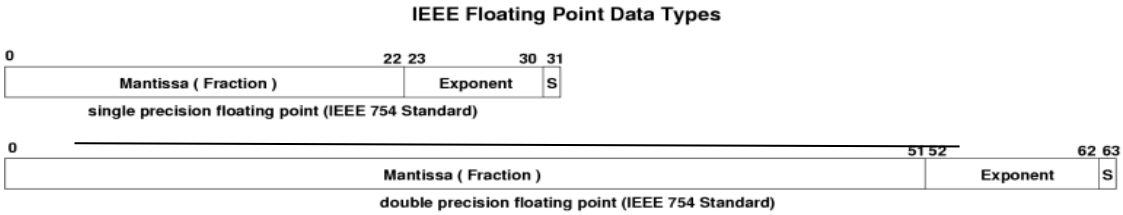


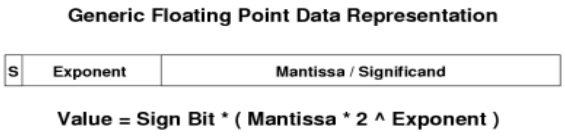
Kod predstavljanja brojeva u pokretnom zarezu opseg je određen brojem cifara u eksponentu, a preciznost brojem cifara u mantisi. Prednost je to što se sa mnogo manje zapamćenih cifara mogu predstaviti brojevi iz znatno većeg opsega. Broj kod koga je najznačajnija cifra mantise različita od nule je *normalizovan*.



Standard za pokretni zarez IEEE 754 koristi normalizovanu mantisu i proširenu notaciju za eksponente i predviđa 3 formata:

1. Format jednostruke preciznosti (32 bita – 1 bit znaka, 8 za eksponent, 23 za signifikant (pošto je mantisa normaizovana podrazumeva se postojanje broja 1 pa se on izostavlja – **impicitna jedinica.** Kombinacija implicitne jedinice, implicitne binarne tačke i 23 zapamćenih bita naziva se signifikant).
2. Format dvostruke preciznosti (64 bita – 1 za znak, 11 za eksponent, 52 za mantisu).
3. Format proširene preciznosti (80 bita)





Kako je mantisa normalizovana, to se jediica koja sledi ne pamti, već se podrazumeva. Kombinacija implicitne 1, implicitne biinarne tačke i 23 zapamćena bita naziva se signifikant (u opsegu 1≤ s< 2).

Item	Single precision	Double precision
Bits in sign	1	1
Bits in exponent	8	11
Bits in fraction	23	52
Bits, total	32	64
Exponent system	Excess 127	Excess 1023
Exponent range	−126 to +127	−1022 to +1023
Smallest normalized number	2^{-126}	2^{-1022}
Largest normalized number	approx. 2^{128}	approx. 2^{1024}
Decimal range	approx. 10^{-38} to 10^{38}	approx. 10^{-308} to 10^{308}
Smallest denormalized number	approx. 10^{-45}	approx. 10^{-324}

Za normalizovane brojeve eksponent ne može imati maksimalnu vrednost ili vrednost jednaku nuli. Oni imaju specijalne namene. Kada se javi rezultat koji je manji od najmanjeg normalizovanog broja koji se može predstaviti, ovaj standard predviđa *denormalizovane* brojeve. Kod njih je eksponent 0, kao i implicitna jednica.

3.2. Vrste programskih jezika

Programski jezici prve generacije

Jezici prve generacije su **mašinski jezici**, tj. jezici pisani na mašinskom nivou. Pojavljuju se sa prvim komercijalnim računarima. Programiranje je izvođeno **binarnim kodom** tj. skupom nula i jedinica. Komande su se pisale direktno u mašinskom jeziku, tj. kao niz naređenja koje se učitavaju u komandni registar procesora i tamo izvršavaju. Komande su zavisile od procesora i zbog toga je portiranje ovakvih programa skoro nemoguće. Danas se ovakav način direktnog programiranja ne koristi, osim u posebnim slučajevima, npr. kada se radi kritičnim hardverskim operacijama.

Primer koda na mašinskom jeziku:

00100111101111011111111111100000
10101111101111110000000000010100
10101111101001000000000000100000
10101111101001010000000000100100
10101111101000000000000000011000
10101111101000000000000000011100
10001111101011100000000000011100
10001111101110000000000000011000
00000001110011100000000000011001
00100101110010000000000000000001
00101001000000010000000001100101
10101111101010000000000000011100
00000000000000000111100000010010
00000011000011111100100000100001
00010100001000001111111111110111
10101111101110010000000000011000
00111100000001000001000000000000
10001111101001010000000000011000
00001100000100000000000011101100
00100100100001000000010000110000
10001111101111110000000000010100
00100111101111010000000000100000
00000011111000000000000000001000
00000000000000000001000000100001

Programski jezici druge generacije

Drugoj generaciji programskih jezika pripadaju **asemblerski jezici**. Jezici ove generacije imaju sledeće karakteristike:

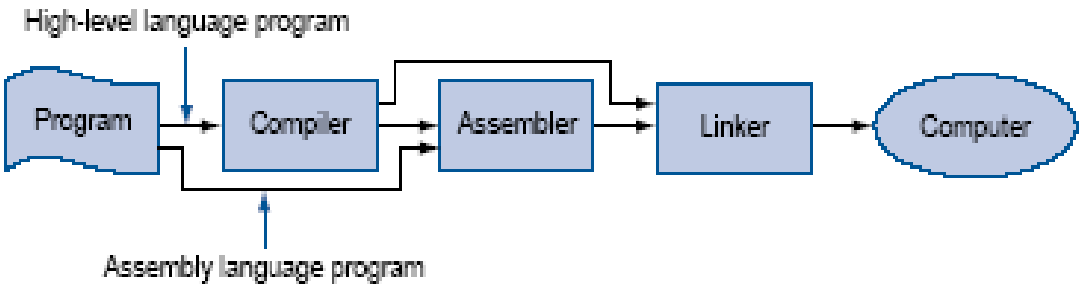
- Kod može da se čita i da se piše od strane programera. Da bi bio pokrenut na računaru on mora da bude pretvoren u mašinski čitljiv oblik, ovaj proces se zove **asembliranje** (**asemblovanje**).
- Jezik je specifičan za određenu porodicu procesora i sredinu.

Jezici druge generacije se ponekad koriste u jezgrima i drajverima uređaja (mada se u modernim jezicima za tu svrhu koristi programski jezik **C**), ali češće nalaze primenu u veoma intenzivnoj obradi kao što su igre, video uredjivanje, grefička manipulacija i td.

Jedna od metoda za stvaranje takvog koda je tako što se prevodiocu dozvoljava generisanje mašinsko-optimizovanih verzija asemblerskog jezika određene funkcije. Ovaj kod se zatim ručno podešava. Primer koda u asemblerskom jeziku:

```
.text
        .align 2
        .globl main
main:
        subu    $sp, $sp, 32
        sw      $ra, 20($sp)
        sd      $a0, 32($sp)
        sw      $0, 24($sp)
        sw      $0, 28($sp)
loop:
        lw      $t6, 28($sp)
        mul     $t7, $t6, $t6
        lw      $t8, 24($sp)
        addu    $t9, $t8, $t7
        sw      $t9, 24($sp)
        addu    $t0, $t6, 1
        sw      $t0, 28($sp)
        ble     $t0, 100, loop
        la      $a0, str
        lw      $a1, 24($sp)
        jal     printf
        move    $v0, $0
        lw      $ra, 20($sp)

        addu    $sp, $sp, 32
        jr      $ra
        .data
        .align 0
str:    .asciiz "The sum from 0 .. 100 is %d\n"
```



Algebarski program piše programer ili je on izlaz kompilatora

3.3. Programski jezici III generacije

Programski jezici treće generacije su **viši programski jezici**. Njih odlikuje visok nivo instrukcija. Jedna naredba ovih programskih jezika prevodi se u više naredbi u mašinskom jeziku. Odlikuje ih nezavisnost od računara što znači da nije potrebno poznavati arhitekturu, instrukcije i registre računara za koji se programira. Programi su bili prenosivi. Jedna od osnovnih karakteristika ovih programskih jezika je akcenat postavljen na strukturi, te se često klasifikuju kao **Strukturni jezici**, što znači da se svi događaji dešavaju po predviđenoj strukturi programskog koda. Zajedničke osobine programskih jezika koji pripadaju trećoj generaciji su: upotreba ključnih reči, komentara, promenljivih, konstanti, petlji i blokova. Veći deo jezika treće generacije podržava i pozivanje stranih funkcija, kao i upotrebu potprograma i procedura. Dele se na:

- Kompilatorske
- Interpretatorske

Kompilatorski programski jezik je programski jezik čije su implementacije tipično kompilatori (prevodioci koji generišu mašinski jezik od izvršnog koda), a ne interpretatori (izvršavaju izvršni kod korak-po-korak, ne vrše prevođenje pre pokretanja).
Interpretatorski programski jezik je programski jezik koji kod većine svojih implementacija izvršava instrukcije direktno, bez prethodnog kompajlovanja programa u uputstvo mašinskog jezika. Interpretator izvršava program direktno, prevođenjem svakoe izjave u sekvencu jednog ili više potprograma koji je već preveden u mašinski kod.
U principu, bilo koji jezik može biti realizovan sa kompilatorom ili sa interpretatorom. Kombinacija oba rešenja je takođe uobičajena: kompilatora može da prevede izvršni kod u neki srednji oblik, koji se zatim prenosi na interpretator koji ga izvršava.
Najvažniji predstavnici programskih jezika treće generacije su:

- FORTRAN
- ALGOL
- BASIC
- COBOL
- LISP
- JOVIAL
- PL/I
- C
- C++
- PASCAL
- JAVA
- Python

Primer koda na programskom jeziku **C**:

```
#include <stdio.h>
int main (int argc, char *argv[])
        {
        int i;
        int sum = 0;
        for (i = 0; i <= 100; i = i + 1)
                sum = sum + i * i;
        printf ("The sum from 0 .. 100 is %d\n", sum);
        }
```

3.4. Programski jezici IV generacije

Programski jezici četvrte generacije su namenjeni za specijalizovane probleme. Zamišljeni su kao dorada stila programskih jezika **treće generacije**. Svaka od generacija programskih jezika ima za cilj da

obezbedi viši nivo apstrakcije u odnosu na hardver računara, praveći jezik više programerski-orijentisanim. Jezici koji pripadaju četvrtoj generaciji mogu uključivati podršku za upravljanje bazom podataka, prikupljanje izveštaja, matematičku optimizaciju, razvijanje GUI ili web razvoj.
Neki od predstavnika jezika četvrte generacije su:

- **Python**
- **Php**
- **Ruby**

Tipovi podataka

Tip podataka ima tri osnovne karakteristike:

- **skup vrednosti** koje podatak tog tipa može da ima;
- **operacije** koje su dozvoljene nad tim tipom;
- **količina bajtova** koja se koristi za predstavljanje podataka tog tipa.

Primetimo da treća karakteristika nije vezana za jezik već za mašinu.
Tip podataka može biti:

- **prost**
- **složen**

Prosti tipovi su:

- **Celobrojni tip** (integer) :
  To je prost, prebrojiv uređen skup brojeva. Oblast vrednosti celobrojnog tipa su 0 i sve pozitivne i negativne celobrojne vrednosti. Aritmetičke operacije koje mogu biti primenjene nad celobrojnim tipom su: sabiranje, oduzimanje, množenje i celobrojno deljenje, kao i operacija poređenja.
- **Realni** (real, float, double)
  Realni tip je podskup skupa realnih brojeva koji se mogu registrovati u nekom programskom jeziku. Nad realnim operandima se mogu izvoditi sledeće aritmetičke operacije koje vraćaju realni tip podataka: množenje, deljenje, sabiranje i oduzimanje. Dozvoljeno je da jedan operand u ovim operacijama bude **celobrojni tip**, izraz će u svakom slučaju vratiti realnu vrednost. Takodje, moguće je realnoj promenljivoj dodeliti celobrojnu vrednost, dok nije moguće uraditi obratno.
- **Znakovni** (char)
  Znakovni tip je još jedan od osnovnih/prostih tipova podataka. Vrednost znakovne promenljive ili konstante je znak iz osnovnog skupa znakova, pri čemu se znakovna konstanta navodi između apostrofa (‘ ‘)
- **Logički** (boolean)
  Logički tip definiše podatke koji mogu imati vrednosti logičkih konstanti: **true** ili **false**. Nad operandima logičkog tipa se mogu primeniti operacije: **not**(negacija, operacija najvišeg nivoa), **not**(konjunkcija) i **or**(disjunkcija)

Složeni tipovi su:

- **Polja** (arrays – vektori i matrice)

- **Zapisi** (record, struct)
- **Datoteke** (files)

3.5. Algoritam – definicija i osobine

Pojam **algoritma** je u matematiku uveo persijski matematičar **Muhamed Al Horezmi (Muhammad ibn Musa al-Khwarizmi)**. Njegova knjiga *Al Horezmi o indijskoj veštini računanja*, biva kasnije prevedena na latinski kao *Algoritmi de numero indorum* i od nepravilnog prevoda njegovog prezimena zapravo nastala reč *Algoritam*.

Algoritam predstavlja konačan skup pravila i postupka kojim se rešava neki problem u konačnom broju koraka, tj. to je precizan opis procesa kojim se date ulazne veličine transformišu u rezultat.

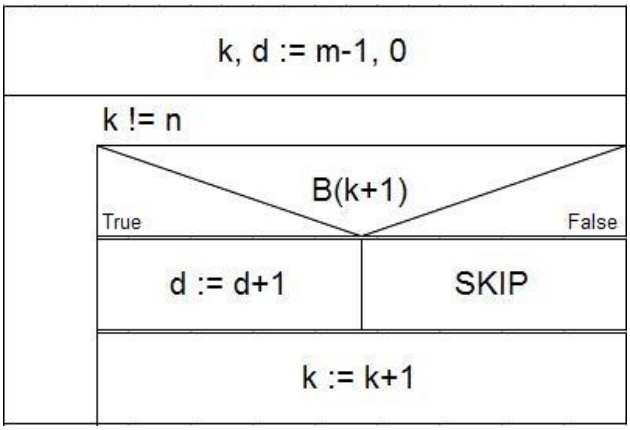
Algoritam se, kao skup pravila i postupaka za izvršenje nekog zadatka, može prikazivati na više načina:

- **Tekstualno**
  Tekstualni opis problema i rešenja
- **Pomoću pseudokoda**
  Tekstualni način dopunjen formalnim izrazima
- **Pomoću strukturograma**
  Kombinacija grafičkog i pseudokoda. Koriste se kao propratna dokumentacija za već završene programe.
- **Grafički** – dijagramom toka algoritma

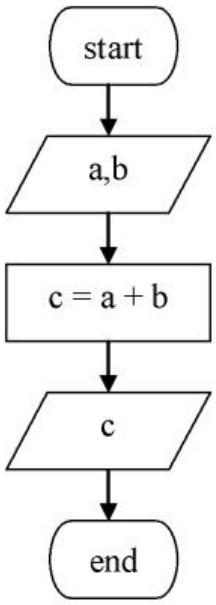
Primer **pseudokoda**:

```
Initialize the Population Do{
    For i = 1 to Population Size{
        Calculate fitness value
        If the fitness value is better than the best one (pBest) in history
        then set current value as the new pBest
    }
    Choose the particle with the best fitness value of all the particles as
    gBest
    For i = 1 to Population Size{
        Calculate new velocity in accordance with Eq. (1)
        Update particle position in accordance with Eq. (2)
    }
} Until termination criterion is met
```

Primer **strukturograma**:



Primer **dijagrama toka programa**:



Svi algoritmi treba da poseduju sledeće osobine:

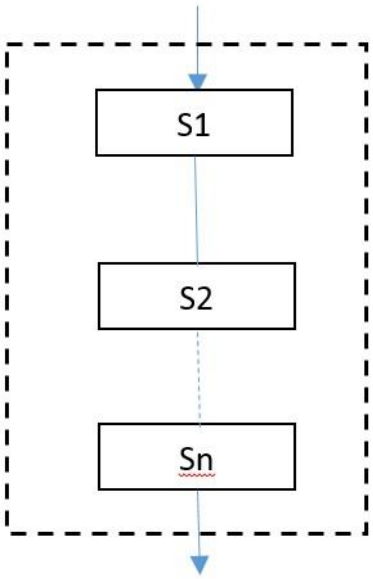
- **Diskretnost –** algoritam predstavlja uređeni niz diskretnih algoritamskih koraka pri čemu se svaki od njih izvršava u toku odgovarajućeg diskretnog vremenskog intervala, a zbir tih intervala predstavlja vreme potrebno za izvršenje algoritma.
- **Determinisanost –** svaki algoritamski korak treba da je tako definisan da se za odgovarajuće ulazne podatke i na osnovu zadate obrade tačno znaju izlazne veličine tog koraka.
- **Rezultativnost –** kod svakog algoritma je neophodno da se unapred zna cilj obrade, tj. šta se smatra rezultatom obrade.
- **Elementarnost –** obrada u pojedinim algoritamskim koracima treba da bude onoliko prosta koliko je to potrebno da bi je razumeo korisnik algoritma
- **Masovnost –** algoritam treba praviti tako da se može koristiti u što je moguće većem broju slučajeva, tj. da ulazne veličine mogu uzimati vrednosti iz što većeg skupa

Pored ovih, postoje i dodatne osobine koje su od posebnog značaja kada se na osnovu algoritma piše računarski program:

- Korišćenje tog algoritma angažuje što je moguće manji deo memorijskog prostora
- Vreme izvršenja algoritma je minimalno moguće
- Struktura algoritma je što moguće prostija

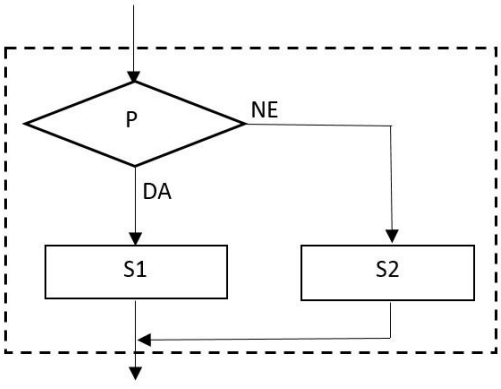
Još sredinom šezdesetih godina pokazano je da se svaki algoritam može predstaviti pomoću svega tri algoritamske strukture:

- **Sekvence**

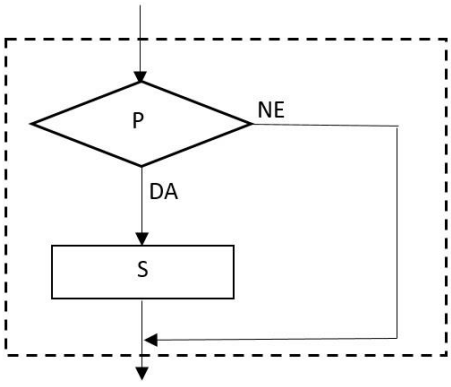


Linijska struktura

- **Alternacije (grananja)**

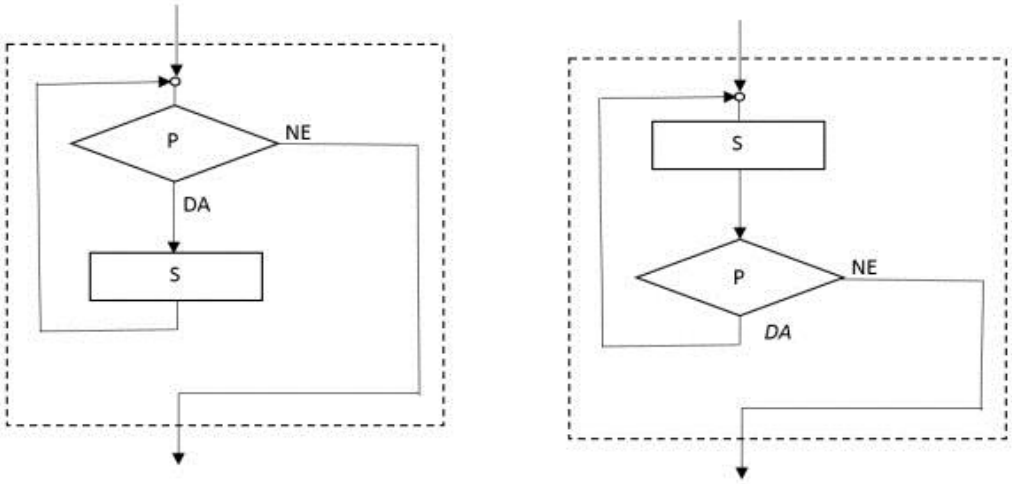


a) If-else (if-then-else)



b) if (if-then)

- **Iteracije (petlje)**



Ciklična struktura

Još iz prve polovine sedamdesetih godina potiče trend ka tzv. ***strukturnom programiranju*** koje predstavlja takvu metodu programiranja gde se programi predstavljaju kao hijerarhijski uređene strukture. Ovakva hijerarhijska struktura algoritma, tj. odgovarajućeg programa, dobija se kao rezultat postepene dekompozicije problema, pri čemu se na svakom od nivoa razlaganja algoritam sastoji samo od osnovnih algoritamskih struktura.

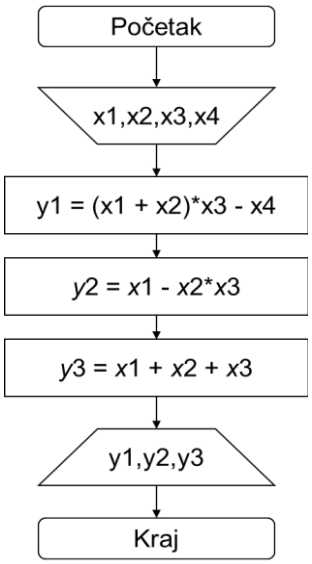
Primer 1. Napisati dijagram toka algoritma za izračunavanje vrednosti sledećih izraza:

$$y_1 = (x_1 + x_2)x_3 - x_4$$

$$y_2 = x_1 - x_2x_3$$

$$y_3 = x_1 + x_2 + x_3$$

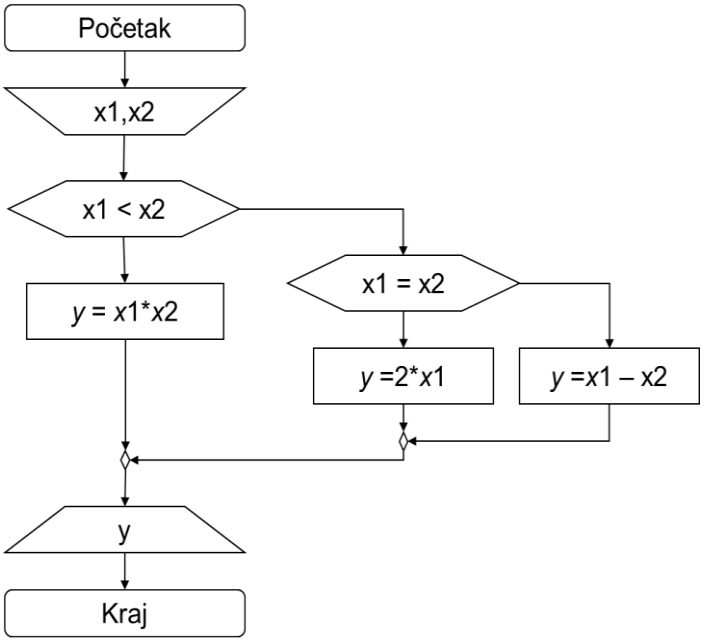
Rešenje:



Primer 2. Napisati dijagram toka algoritma za izračunavanje vrednosti funkcije:

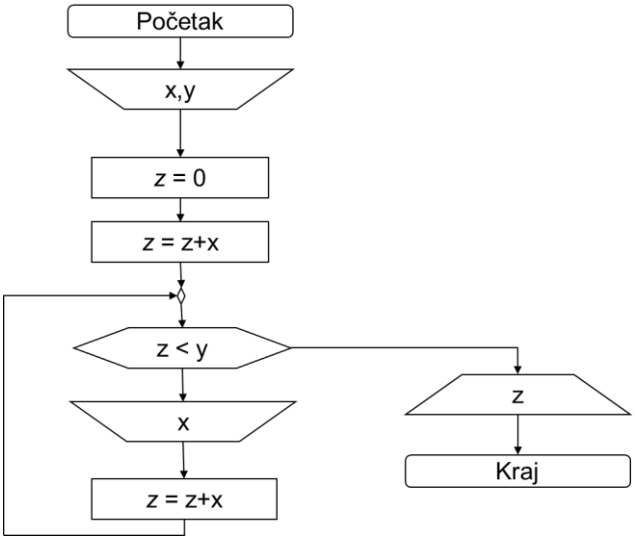
$$y = \begin{cases} x_1 x_2, & x_1 < x_2 \\ 2x_1, & x_1 = x_2 \\ x_1 - x_2, & x_1 > x_2 \end{cases}$$

Rešenje:



Primer 3. Sastaviti dijagram toka algoritma kojim se vrši kumulativno sabiranje prirodnih brojeva sve dok suma ne dostigne zadatu vrednost *y*.

Rešenje:



4. Osnovno o WWW

4.1. Internet

Internet je globalna mreža koja povezuje ogroman broj računara i drugih uređaja koristeći TCP/IP protokol. Nosi nadimak „mreža svih mreža“. Preteča Interneta je ARPANET – vojni projekat ministarstva odbrane SAD. Prva veza je uspostavljena 1969. godine. Internet nudi veliki broj usluga, kao što je WWW, E-mail, VoIP telefonija, razmena fajlova i sl.

Veličina Interneta

Procenjuje se da internet koristi više od 3.5 milijarde korisnika i da je na njemu povezano više od 20 milijardi uređaja. Najveći udeo uzimaju IoT uređaji. Procenjuje se da će do 2020. biti više od 30 milijardi povezanih uređaja.

Google

Svakog meseca Google poseti više od 187 miliona jednistvenih korisnika svakog meseca koji naprave više od 100 milijardi pretraga. Čuva više od 45 milijardi indeksiranih stranica u svojoj bazi. Procenjuje se da je 2014. indeksirao 200 TB podataka, što predstavlja 0.004% svih podataka na Internetu.

YouTube

Poseduje milijardu korisnika. Svakog dana se pregledaju stotine miliona sati video sadržaja. Svakog minuta se otpremi više od 300 sati novog video materijala.

Facebook

Sadrži 1.6 milijardi aktivnih korisnika, dok se više od milijardu njih prijavljuje svakog dana. Svake sekunde se registruje pet novih korisnika, dok se svakog minuta postavi više od 500 komentara i 290.000 statusa. Svakog dana se otpremi 300 miliona novih slika.

Twitter

Korisnici Twittera pošalju preko 500 miliona poruka svakog dana.

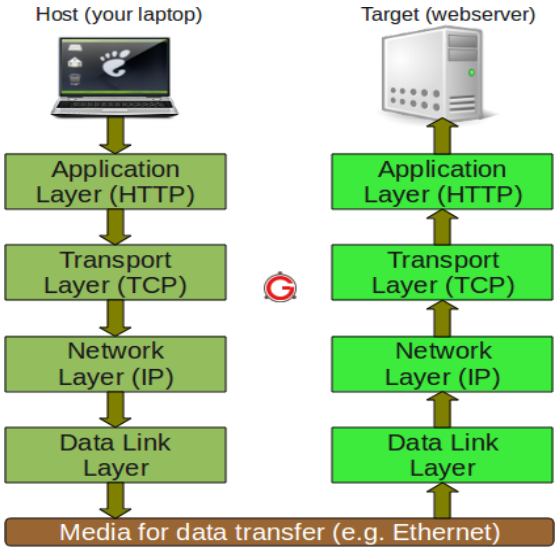
4.2. Internet ekonomija

Internet predstavlja oko 5% BDP-a Sjedinjenih Američkih Država. Internet prodaja je jako popularna, te je preko 40% korisnika vršilo kupovinu preko Interneta. Tržište Internet prodaje u Kini godišnje ostvari profit od 560 milijardi dolara, dok je u SAD ovaj iznos 349 milijardi

dolara. Više od polovine kupaca odustaje od kupovine ukoliko je sajt na kome vrše kupovinu spor. Postoji i velika zastupljenost upotrebe Interneta putem prenosnih uređaja, pa je 75% korisnika Interneta u SAD pristupilo istom preko mobilnog telefona.

TCP/IP

Komunikacija se odvija u više slojeva.



- **Aplikativni sloj**

Koristi se između aplikacija. Neki od primera su HTTP (HyperText Transfer Protokol, portovi 80 i 8080), HTTPS (HTTP Secure - 443), SMTP (Simple Mail Transfer Protocol - 25), FTP (File Transfer Protocol – 20, 21), SSH (Secure Shell - 22), DNS (Domain Name System - 53), POP3 (Posto Office Protocol - 110), RDP (Remote Desktop Protocol – 3389), NTP (Network Time Protocol – 223), kao i mnogi drugi. Računari se adresiraju pomoću broja porta koji se koristi za komunikaciju.

- **Transportni sloj**

Transportni sloj ima zadatak da obezbedi pouzdan prenos podataka, bez obzira na fizičku mrežu koja se nalazi između izvornog i odredišnog računara. Osnovni protokoli su TCP i UDP. TCP je skraćeno za Transmission Control Protocol. Predstavlja pouzdan prenos podataka sa uspostavljanjem direktne veze. Sa druge strane, UDP (User Datagram Protocol) predstavlja protokol bez uspostavljanja direktne veze. Iako može rezultovati gubitkom paketa, glavna prednost ovog protokola je brzina. Koristi se za aplikacije kod kojih je neophodan prenos i obrada podataka u realnom vremenu, kao što je telefonija ili serveri za igre.

- **Mrežni sloj**

Zadatak mrežnog sloja je da sprovede pakete od izvora do odredišta. Prema tome, on mora da poznaje topologiju mreže i bira najoprimalnije putanje. Indentifikacija računara u mrežnom sloju se vrši pomoći IP adresa. IPv4 adrese su sastavljene od 4 bajta (32 bita). Lako je zaključiti da je ukupan broj IPv4 adresa približno 4 milijardi. IPv4 adresa 180.120.127.99 se u računaru prikazuje kao 10110100.01111000.01111111.01100011. Usled nestašice IPv4 adresa, razvijen je IPv6 standard, koji ni dan danas nije u širokoj upotrebi. Nove adrese su sačinjene od 16 bajtova (128 bitova), pa je moguće sastaviti 2^{32} IPv6 adresa. Primer IPv6 adrese je 2001:4860:4860:0:0:0:0:8888.

- **Sloj veze podataka**

Sloj veze podataka preuzima pakete koje dobija od mrežnog sloja i inkapsulira ih u okvire pogodne za transport.

- **Fizički sloj**

Bakarna žica i prateća infrastruktura pomoću koje su računari povezani.

URI

URI je srkaćenica koja predstavlja Uniform Resource Identifier. Pomoću URI adrese se jednoznačno određuje neki resurs. To može biti web stranica, dokument, ili pristup bazi, video tok i slično. Osim toga, postoji i URL (Uniform Resource Locator) koji se koristi za lociranje nekog resursa na Internetu i URN (Uniform Resource Name) koji uvek pokazuje na određeni resurs (za razliku od URL-a, gde se sadržaj dateteke na koji upućuje može nezavisno pomerati i menjati).

Struktura URL-a

- **Protocol** - najčešće ime protokola (npr. http, https, ftp, file, mailto)
- **Host** - IP adresa ili domen hosta

- **Port** - **d**ecimalni broj porta koji se koristi za komunikaciju
- **Directory** - putanja na serveru
- **File** - **n**aziv datoteke kojoj se pristupa
- **Query** - parametri koji se koriste za upit
- **Fragment** - određena pozicija na stranici (skok na određen naslov ili sekciju)

HTTP

Predstavlja osnovu za WWW i služi za razmenu *hypermedia* podataka. Klijent je pretraživač, tj. računar sa koga se pristupa. Server je računar na kome se sajt nalazi. Klijent šalje HTTP zahteve za određenim resursima, a server vraća HTTP odgovor. Sadržaj poruke u HTTP komunikaciji najčešće sadrži URL sa naznačenom metodom i status odgovora (200 OK, 404 Not Found…). U zaglavlju poruke se mogu nalaziti različite vrednosti, kao što su podaci o pretraživaču korisnika, kolačići, tipovi enkodiranja… Najbitnija stavka je, naravno, telo poruke.

HTTP Request metode

Tipovi zahteva se koriste da označe operaciju koja treba da se obavi nad određenim resursom. Najčešći tipovi zahteva su GET (zahtev za podacima), POST (slanje podataka na dalju obradu), PUT (dodavanje ili izmena postojećih resursa), DELETE (briše resurs), HEAD, CONNECT i drugi.

Primer HTTP zahteva i odgovora

```
GET /putanja/fajl.html HTTP/1.0
From: someone@jsomewhere.com
User-Agent: HTTPApp/1.1

HTTP/1.1 200 OK
Date: Fri, 8 Apr 2016 23:59:59 GMT
Content-Type: text/html Content-Length: 1222
<html><body><h1>Pozdrav!!</h1>
HTTP/1.1 404 Not Found

POST /putanja/imenik.asp HTTP/1.0
From: someone@somewhere.com
User-Agent: HTTPApp/1.1
Content-Type: application/x-www-form-urlencoded
Content-Length: 32
ime=Petar&mesto+stanovanja=nis
```

Web stranice

Pri izradi web stranica se koriste jezici kao što su HTML (HyperText Markup Language), CSS (Cascading Style Sheets) koji se koristi za definisanje stilova i JavaScript – programski jazik kojim se dodaje dinamičnost web stranice sa klijentske strane. Veliki broj biblioteka se koristi uz JavaScript – jQuery, AngularJS, Node.js…

Stranice se mogu podeliti na statičke (one napisane u čistom HTML-u, uvek prikazuju isti sadržaj) i dinamičke (generisane putem PHP-a, serverskog programskog jezika).

Primer HTML dokumenta

```
<!DOCTYPE html>
<html>
  <script>
    function f() {
      var broj = parseInt(document.getElementById("broj").text) + 1;
      document.getElementById("broj").text = broj;
    }
  </script>
  <body>
    <a id="broj">0</a>
    <h1 style="color:blue;" onclick="f()">Klikni me!</h1>
  </body>
</html>
```

4.3. Razmena podataka

Podaci se najčešće razmenjuju u XML ili JSON obliku. Oba tipa sadrže hijerarhiju, pa je lako vršiti navigaciju kroz stablo dokumenta.

XML je skraćenica za Extensible Markup Language.

```
<imenik>
  <podatak>
    <ime>Petar Petrovic</ime>
    <brojTelefona>123456789</brojTelefona>
  </podatak>
</imenik>
```

JSON predstavlja JavaScript Object Notation.

```
{
  "imenik": {
    "podatak": {
      "ime": "Petar Petrovic",
      "brojTelefona": "123456789"
    }
  }
}
```

HTML5

Predstavlja najnoviju verziju HTML standarda. Donosi mnoga poboljšanja, kao pto su <video> i <audio> tagovi za multimediju, <canvas> tag za grafiku, Web storage, upravljanje datotekama, web sokete. Moguće je realizovati kompletne aplikacije koje se izvršavaju u pretraživaču, pa je smanjeno korišćenje spoljašnjih komponenti, npr. Adobe Flash Player.

Pretraživači

Najpopularniji pretraživači su Google Chrome (oko 47%), Mozilla Firefox (oko 9.2%), Internet Explorer (9.4%), Safari (12.5%), Opera (5.6%), Microsoft Edge (0.7%).

Zadaci

Linijske strukture

Zadatak 1. Napisati program koji za unete brojeve a, b i c izračunava njihov zbir, proizvod i aritmetičku sredinu.

Rešenje:

```
program ZbirProizvodArSredina;
var a,b,c,z,p:integer; s:real;
begin
      writeln('Unesi brojeve a, b i c ');
      readln(a,b,c);
      z:=a+b+c;
      p:=a*b*c;
      s:=p/3;
      writeln('Zbir:',z);
      writeln('Proizvod:',p);
      writeln('Aritmeticka sredina:',s);
end.
```

Zadatak 2. Napisati program kojim se izračunava hipotenuza i površina pravouglog trougla.

Rešenje:

```
program HipotenuzaIPovrsina;
var a,b,c,p:real;
begin
      writeln('Unesi katete a i b ');
      readln(a,b);
      c:=sqrt(sqr(a)+sqr(b));
      p:=a*b/2;
      writeln('Povrsina trougla je',p);
end.
```

Zadatak 3. Napisati program kojim se uneti ugao izražen u stepenima izražava u radijanima.

Rešenje:

```
program Ugao;
const pi=3.14;
var st:integer; rad:real;
begin
      writeln('Unesi ugao u stepenima:');
      readln(st);
      rad:=st*pi/180;
      writeln('Ugao u radijanima je:',rad);
end.
```

Zadatak 4. Za učitani dvocifreni broj izdvojiti njegove cifre.

Rešenje:

```
program Cifre;
```

```
var n,a,b:integer;
begin
      writeln('Unesi dvocifren broj:');
      readln(n);
      a:=n div 10;
      b:=n mod 10;
      writeln('Cifra desetice:',a);
      writeln('Cifra jedinice:',b);
end.
```

Zadatak 5. Za učitani dvocifren broj izračunati broj zapisan istim ciframa ali u obrnutom poretku.

Rešenje:

```
program Inverz;
var n,inv:integer;
begin
      writeln('Unesi dvocifren broj:');
      readln(n);
      inv:=n mod 10;
      inv:=10*inv + n div 10;
      writeln('Broj sa ciframa u inverznom poretku:',inv);
end.
```

Zadatak 6. Napisati program koji za uneti brojilac i imenilac ispisuje odgovarajući broj u mešovitom obliku.

Rešenje: Mešoviti oblik se označava celim delom i razlomkom gde je brojilac manji od imenioca (npr. 8/3 = 2 cela i 2/3).

```
program Transformacija;
var br,im,ceo,nbr:integer;
begin
      writeln('Unesi brojilac i imenilac:');
      readln(br,im);
      ceo:=br div im;
      nbr:=br-ceo*im;
      writeln('Ceo deo: ',ceo);
      writeln('Razlomljeni deo: ',nbr,'/',im);
end.
```

Zadatak 7. Bankomat raspolaže novčanicama od 1000 din., 500 din., 100 din. i 1 din. Odrediti najmanju količinu novčanica kojom bankomat može isplatiti iznos od n dinara podrazumevajući da bankomat raspolaže dovoljnom količinom svake novčanice.

Rešenje:

```
program Bankomat;
var n, broj1000, broj500, broj100, broj1 : integer;
begin
  writeln('Unesite zeljeni iznos za isplatu');
  readln(n);
```

```
  broj1000:=n div 1000;
  n:=n mod 1000;
  broj500:=n div 500;
  n:=n mod 500;
  broj100:=n div 100;
  n:=n mod 100;
  broj1:=n;
  writeln('Broj novcanica od 1000 din:',broj1000);
  writeln('Broj novcanica od  500 din:',broj500);
  writeln('Broj novcanica od  100 din:',broj100);
  writeln('Broj novcanica od    1 din:',broj1);
end.
```

Zadatak 8. Date su dve tačke koordinatnog sistema, zadate svojim koordinatama. Izračunati njihova rastojanja od koordinatnog početka.

Rešenje:

```
program rastojanje;
var x1,x2,y1,y2,d1,d2:real;
begin
      writeln('Unesi koordinate prve tacke');
      readln(x1,y1);
      writeln('Unesi koordinate druge tacke');
      readln(x2,y2);
      d1:=sqrt(sqr(x1)+sqr(y1));
      d2:=sqrt(sqr(x2)+sqr(y2));
      writeln('Rastojanje prve tacke od koordinatnog pocetka',d1);
      writeln('Rastojanje druge tacke od koordinatnog pocetka',d2);
end.
```

Zadatak 9. Ulazni podaci su početak i kraj nekog vremenskog intervala, izraženi u satima i minutima: (sat1,min1) i (sat2,min2). Napisati program kojim se odredjuje dužina vremenskog intervala izražena u satima i minutima.

Rešenje:

```
program VremenskiIntervali;
var t,sat1,sat2,min1,min2,rezSat,rezMin : integer;
begin
  read(sat1, min1);
  read(sat2, min2);
  t:=sat2*60+min2-sat1*60-min1;
  rezSat:=t div 60;
  rezMin:=t mod 60;
  writeln(rezSat, rezMin);
end.
```

Zadatak 10. Tri tačke u koordinatnom sistemu su zadate svojim koordinatama. Izračunati obim i površinu trougla odredjenog ovim temenima.

Rešenje:

```
program trougao;
var x1, x2, y1, y2, x3, y3, a, b, c, o, p , s:real;
```

```
begin
      writeln('Unesi koordinate prve tacke');
      readln(x1,y1);
      writeln('Unesi koordinate druge tacke');
      readln(x1,y1);
      writeln('Unesi koordinate trece tacke');
      readln(x3,y3);
      a:=sqrt(sqr(x1-x2)+sqr(y1-y2));  {rastojanje izmedju prve i druge
tacke}
      b:=sqrt(sqr(x1-x3)+sqr(y1-y3));  {rastojanje izmedju prve i trece
tacke}
      c:=sqrt(sqr(x3-x2)+sqr(y3-y2));  {rastojanje izmedju druge i trece
tacke}
      o:=a+b+c;                        {obim}
      s:=o/2;                          {poluobim}
      p:=sqrt(s*(s-a)*(s-b)*(s-c));    {povrsina}
      writeln('Obim iznosi',o);
      writeln('Povrsina iznosi',p);
end.
```

Zadatak 11. Učitati dva cela broja x i y, a zatim im promeniti vrednosti.

Rešenje:

```
program Zamena;
var x,y,t:integer;
begin
  writeln('Unesi brojeve x i y:');
  readln(x,y);
  t := x;
  x := y;
  y := t;
  writeln('x=',x);
  writeln('y=',y);
end.
```

Zadatak 12. Uraditi prethodni zadatak bez korišćenja pomoćne promenljive.

Rešenje:

```
program Zamena;
var x,y:integer;
begin
  writeln('Unesi brojeve x i y:');
  readln(x,y);
  x := x+y;
  y := x-y;
  x := x-y;
  writeln('x=',x);
  writeln('y=',y);
end.
```

Grananja

Zadatak 13. Ispitati da li je uneti broj paran ili neparan.

Rešenje:

```
program ParNepar;
var broj: integer;
begin
 write ('Upisi broj: ');
 readln (broj);
 if broj mod 2 <> 0 then writeln('Broj je neparan')
 else writeln('Broj je paran');
end.
```

Zadatak 14. Treba učitati dva broja a zatim ispisati manji pa veći broj.

Rešenje:

```
program ManjiVeci;
var a,b: integer;
begin
 write ('Unesi dva broja: ');
 readln (a,b);
 write('Redosled manji-veci je: ');
 if a<b then write(a,' ',b) else writeln(b,' ',a);
end.
```

Zadatak 15. Odrediti maksimum dva broja.

Rešenje:

```
program MaxDvaBroja;
var x,y,max : integer;
begin
 write('Unesi dva broja:');
 readln(x,y);
 if x > y
   then max:=x
   else max:=y;
 writeln('Maksimum je: ',max);
end.
```

Zadatak 16. Odrediti maksimum četiri broja.

Rešenje:

```
program MaxCetiriBroja;
var x,y,z,t,max : integer;
begin
 write('Unesi cetiri broja:');
 readln(x,y,z,t);
 max := x;
 if max < y then max:=y;
 if max < z then max:=z;
 if max < t then max:=t;
writeln('Maksimum je: ',max);
end.
```

Zadatak 17. Napisati program kojim se ispituje da li su uneta tri broja u neopadajućem redosledu.

Rešenje:

```
program Neopadajuci;
var a,b,c : integer;
begin
 write('Unesi tri broja:');
 readln(a,b,c);
 if (a <= b) and (b <= c)
    then writeln('Brojevi su u neopadajucem poretku')
    else writeln('Brojevi nisu u neopadajucem poretku');
end.
```

Zadatak 18. Za učitani prirodan broj odštampati njegov najbliži broj koji je deljiv sa 17.

Rešenje:

```
program Deljenje;
var broj,ost: integer;
begin
 write ('Upisi broj: ');
 readln (broj);
 ost:=broj mod 17;
 if ost < 17-ost
    then writeln('Najblizi broj je: ',broj-ost)
    else writeln('Najblizi broj je: ',broj+17-ost);
end.
```

Zadatak 19. Napisati program koji za unete dužine stranica a, b i c ispituje da li se od njih može kreirati trougao.

Rešenje:

```
program Trougao;
var a,b,c : integer;
begin
 write('Unesite duzine stranica trougla a,b i c:');
 readln(a,b,c);
 if (a>b+c) and (b>a+c) and (c>a+b)
   then writeln('Moze se kreirati trougao')
   else writeln('Moze se kreirati trougao')
end.
```

Zadatak 20. Za učitane brojeve a1, a2, i a3 napisati program kojim se izračunava zbir brojeva koji se nalaze u intervalu 2 do 5.

Rešenje:

```
program SumaIzIntervala;
var a1,a2,a3,s: integer;
begin
 s:=0;
 write('Unesite brojeve a1, a2 i a3:');
 readln(a1,a2,a3);
 if (2<a1) and (a1<5) then s:=s+a1;
```

```
 if (2<a2) and (a2<5) then s:=s+a2;
 if (2<a3) and (a3<5) then s:=s+a3;
 writeln('Suma brojeva iz intervala 2 do 5 je:',s);
end.
```

Zadatak 21. Napisati program koji izračunava z definisano na sledeće načine:

$$a)\ z=\begin{cases}x^2, & za\ x\le y\\ x*y, & za\ x>y\end{cases}\qquad b)\ z=\begin{cases}-x+1, & za\ -4<x<3\\ x^2, & za\ 3\le x\le 8\\ 1/x & inače\end{cases}$$

$$c)\ z=\begin{cases}\sqrt{x+y}, & za\ x\ge 0, y\ge 0\\ \max(x^2,y^2), & inače\end{cases}$$

Rešenje:

```
program Funkcija_a;
var x,y,z : integer;
begin
 write('Unesite brojeve x i y:');
 readln(x,y);
 if x<y then z:=sqr(x)
 else z:=x*y;
 writeln('z=',z);
end.

program Funkcija_b;
var x : integer; z:real;
begin
 write('Unesite broj x:');
 readln(x);
 if (x>-4) and (x<3)
   then z:=1-x
   else if (x>=3) and (x<=8)
          then z:=sqr(x)
          else z:=1/x;
 writeln('z=',z);
end.

program Funkcija_c;
var x,y : integer; z : real;
begin
 write('Unesite brojeve x i y:');
 readln(x,y);
 if (x>0) and (y>0)
   then z:=sqrt(x+y)
   else if sqr(x) > sqr(y)
          then z:=sqr(x)
          else z:=sqr(y);
 writeln('z=',z);
end.
```

Zadatak 22. Napisati program koji tri data broja a, b, c štampa u neopadajućem redosledu.

Rešenje:

```
program Uredjenje1;
var a,b,c: integer;
begin
 write('Unesite brojeve a, b i c:');
 readln(a,b,c);
 if (a<b) then
   begin
    if b<c then writeln(a,b,c)
    else if a<c then writeln(a,c,b)
         else writeln(c,a,b);
   end
  else
    begin
     if b>c then writeln(c,b,a)
     else if a<c then writeln(b,a,c)
          else writeln(b,c,a)
    end;
end.

program Uredjenje2;
var a,b,c,t: integer;
begin
 write('Unesite brojeve a, b i c:');
 readln(a,b,c);
 if a>b then
   begin
     t:=a; a:=b; b:=t;
   end;
 if a>c then
   begin
     t:=a; a:=c; c:=t;
   end;

 if b>c then
   begin
     t:=b; b:=c; c:=t;
   end;
 writeln(a,b,c);
end.
```

Zadatak 23. Napisati program kojim se učitava znak za operaciju (+, *, -, /) i dva realna operanda, a zatim štampa odgovarajući rezultat.

Rešenje:

```
program Kalkulator;
var x,y,rez : real; op : char; def : boolean;
begin
  writeln('Unesi prvi operand, operaciju i drugi operand');
  readln(x); readln(op); readln(y);
  def:=true;
  case op of
```

```
      '+': rez := x+y;
      '-': rez := x-y;
      '*': rez := x*y;
      '/': if y=0 then
                    begin
                     def := false;
                     write('Nedefinisan rezultat');
                    end
              else rez := x/y;
  end;
  if (def) then write('Rezultat=',rez);
end.
```

Zadatak 24. Napisati program koji za uneti broj meseca (1=Januar, 2=Februar, ...) i trenutnu godinu ispisuje njegov broj dana. Godina je prestupna ako je deljiva sa 4 i nije deljiva sa 100 ili ako je deljiva sa 400.

Rešenje:

```
program Meseci;
var mesec,godina : integer; prestupna : boolean;
begin
 write('Unesi broj meseca od 1 do 12:');
 readln(mesec);
 write('Unesi godinu:');
 readln(godina);
 if (mesec < 1) or (mesec > 12)
    then writeln('Broj meseca je nekorektan')
    else
     case mesec of
       1,3,5,7,8,10,12 : writeln('31 dan');
       4,6,9,11 : writeln('30 dana');
       2 : begin
            prestupna:=((godina mod 4 = 0) and (godina mod 100 <> 0)) or
(godina mod 400 = 0);
            if (prestupna) then writeln('29 dana')
                           else writeln('28 dana');
           end;
     end;
end.
```

Petlje

Zadatak 25. Odrediti sve delioce prirodnog broja n.

Rešenje:

```
program Delioci;
var i,n : integer;
begin
 write('Unesi broj n : ');
 readln(n);
 writeln('Delioci broja ',n, ' su:');
 for i := 1 to n do
   if n mod i = 0 then write(i,' ');
```

```
end.
```

Zadatak 26. Napisati program koji će ispisati tablicu množenja za jednocifrene brojeve.

Rešenje:

```
program TablicaMnozenja;
var i,j : integer;
begin
 for i:= 1 to 9 do begin
  for j:= 1 to 9 do
   write(i*j,' ');
  writeln;
 end;
end.
```

Zadatak 27. Napisati program koji štampa sve brojeve do 1000 koji su deljivi sa 3, nisu deljivi sa 7, a poslednja cifra im je neparan broj.

Rešenje:

```
program Brojevi;
var n,c:integer;
begin
 for n := 1 to 1000 do
  begin
    c:=n mod 10;
    if (n mod 3 = 0) and (n mod 7 <> 0) and (c mod 2 <> 0) then
       writeln(n);
  end;
end.
```

Zadatak 28. Napisati program koji računa zbir prvih n brojeva.

Rešenje:

```
program Suma;
var s,i,n : integer;
begin
 write('Unesi broj n : ');
 readln(n);
 s:=0;
 for i := 1 to n do
   s:=s+i;
 write('Suma prvih ',n,' brojeva je:',s);
end.
```

Zadatak 29. Napisati program koji izračunava n-ti stepen broja x.

Rešenje:

```
program Stepen;
var step,i,n,x : integer;
begin
 write('Unesi step i broj: ');
```

```
 readln(n,x);
 step := 1;
 for i := 1 to n do
   step:=step*x;
 write(x,' na ',n,' je ',step);
end.
```

Zadatak 30. Napisati program koji računa n!.

Rešenje:

```
program Faktorijel;
var fakt : longint; i,n : integer;
begin
 write('Unesi broj n : ');
 readln(n);
 fakt := 1;
 for i := 1 to n do
   fakt := fakt * i;
 write(n,'!=',fakt);
end.
```

Zadatak 31. Napisati program koji štampa sve četvorocifrene brojeve kod kojih su srednje dve cifre jednake i koji su pritom deljivi brojem dobijenim izbacivanjem srednje dve cifre.

Rešenje:

```
program CetvorocifreniBrojevi;
var a,b,c : byte;
    broj : integer;
begin
   for a := 1 to 9 do
    for b := 0 to 9 do
     for c := 0 to 9 do
      begin
        broj := 1000*a+100*b+10*b+c;
        if (broj mod (10*a+c) = 0) then writeln(broj);
      end;
end.
```

Zadatak 32. Napisati program koji će ispisati prirodan broj iz segmenta [a, b] koji ima najveći broj delilaca. Ukoliko postoji više brojeva sa istim brojem delilaca, odštampati najveći od njih.

Rešenje:

```
program Delioci;
var i,n,max,s,broj,a,b : integer;
begin
 write('Unesi granice a i b: ');
 readln(a,b);
 max:=0;
 for n := a to b do begin
   s:=0;
   for i := 1 to n do
     if n mod i = 0 then s:=s+1;
   if s >= max then begin
```

```
        max:=s;
        broj:=n;
    end;
  end;
 writeln('Broj sa najvise delilaca iz segmenta [',a,',',b,'] je:',broj);
end.
```

Zadatak 33. Ispitati da li je uneti broj prost.

Rešenje:

```
program ProstBroj;
var n,i,koren : integer;
    prost : boolean;
begin
  readln(n);
  koren := round(sqrt(n));
  prost := ((n>1) and (n mod 2<>0)) or (n=2);
  i := 3;
  while (prost) and (i<=koren) do
  begin
    prost:=n mod i <> 0;
    i:=i+2;
  end;
  if (prost) then writeln(n,' je prost broj')
  else writeln(n,' nije prost broj');
end.
```

Zadatak 34. Napisati program kojim se nalazi najveći zajednički delilac dva broja.

Rešenje:

```
program NZD;
var a,b,x,y,t : integer;
begin
  write('Unesiti dva broja:');
  readln(a,b);
  x:=a; y:=b;
  while y <> 0 do
     begin
       t := y;
       y := x mod y;
       x := t;
     end;
  writeln('NZD(',a,',',b,')=',x);
end.
```

Zadatak 35. Napisati program kojim se izračunava maksimalna i srednja vrednost niza prirodnih brojeva čija dužina unapred nije poznata. Niz se učitava do unosa prve nule.

Rešenje:

```
program SredinaIMax;
const stop=0;
var i,x,max : integer;
    s : real;
```

```
begin
 max:=0; s:=0; i:=0;
 writeln('Unesite niz koji se zavrsava brojem ',stop);
 readln(x);
 while x <> stop do
  begin
    if x > max then max:=x;
    s:=s+x;
    i:=i+1;
    readln(x);
  end;
  if i > 0 then begin
    s:=s/i;
    writeln('Srednja vrednost je :',s);
    writeln('Maksimalni element je : ',max);
  end
end.
```

Zadatak 36. Napisati program kojim se odredjuje koliko cifara ima uneti broj n.

Rešenje:

```
program BrojCifara;
var n,br: integer;
begin
 write('Uneti zeljeni broj:');
 readln(n);
 br:=0;
 repeat
  br:=br+1;
  n:=n div 10;
 until n = 0;
writeln('Broj cifara je ',br);
end.
```

Zadatak 37. Napisati program kojim se odredjuje broj jedinica u binarnom zapisu broja n.

Rešenje:

```
program BinarnoBrojJedinica;
var br,n : integer;
begin
  write('Uneti zeljeni broj:');
  read(n);
  br:=0;
  repeat
   br:=br + n mod 2;
   n:=n div 2;
  until n = 0;
  writeln('Broj jedinica je :',br);
end.
```

Zadatak 38. Napisati program kojim se od datog prirodnog broja n kreira broj sa istim ciframa ali u inverznom poretku.

Rešenje:

```
program Invertovanje;
var n,inv : longint;
begin
 write('Uneti zeljeni broj:');
 readln(n);
 inv:=0;
 repeat
  inv:=inv*10 + n mod 10;
  n := n div 10;
 until n = 0;
writeln('Broj zapisan u obrnutom redosledu je:',inv);
end.
```

Nizovi (Zadaci)

Zadatak 1. Dat je niz celih brojeva a dužine n. Obrisati element sa k – tog mesta iz niza.

Rešenje:

```
void zad1() {

	int a[30];
	int i, n, k;
	printf("Unesite N:");
	scanf_s("%d", &n);
	printf("Unesite k:");
	scanf_s("%d", &k);
	printf("Unesite elemente niza.\n");
	for (i = 0; i < n; i++) {
		scanf_s("%d", &a[i]);
	}
	for (i = 0; i < n; i++) {
		printf("Element[%d] = %d\n", i, a[i]);
	}
	for (i = 0; i < n; i++) {
		if (i >= k - 1)
			a[i] = a[i + 1];
	}
	printf("\n");
	for (i = 0; i < n - 1; i++) {
		printf("Element[%d] = %d\n", i, a[i]);
	}
}
```

Zadatak 2. Dat je niz celih brojeva a dužine n. Elementi niza su uređeni u neopadajući redosled. Napisati program kojim se dodaje novi element x u niz a, tako da niz ostane uređen.

Rešenje:

```
void zad2() {
	int a[30];
	int noviNiz[30];
	int i, n, x, j = 0;

	printf("Unesite N:");
	scanf_s("%d", &n);

	printf("Unesite X:");
	scanf_s("%d", &x);

	printf("Unesite elemente niza.\n");
	for (i = 0; i < n; i++) {
		scanf_s("%d", &a[i]);
	}
	if (a[0] > x) {
		noviNiz[j] = x;
		j++;
	}
	for (i = 0; i < n; i++) {
		if (x >= a[i] && x < a[i + 1]) {
			noviNiz[j++] = a[i];
			noviNiz[j++] = x;
		}
		else {
			if (i == n - 1 && x > a[i]) {
				noviNiz[j++] = a[i];
				noviNiz[j++] = x;
			}
			else {
				noviNiz[j++] = a[i];
			}
		}
	}
	for (i = 0; i < n + 1; i++) {
		printf("Element[%d] = %d\n", i, noviNiz[i]);
	}
}
```

Zadatak 3. Dat je niz celih brojeva a dužine n. Rotirati elemente niza za jedno mesto

1. u levo;
2. u desno.

Rešenje:

```
void zad3_levo() {
	int a[30];
	int i, n, x;
	printf("Unesite N:");
	scanf_s("%d", &n);
	printf("Unesite elemente niza.\n");
```

```
	for (i = 0; i < n; i++) {
		scanf_s("%d", &a[i]);
	}
	// levo
	x = a[0];
	for (i = 0; i < n - 1; i++) {
		a[i] = a[i + 1];
	}
	a[n - 1] = x;
	for (i = 0; i < n; i++) {
		printf("Element[%d] = %d\n", i, a[i]);
	}

}

void zad3_desno() {
	int a[30];
	int i, n, x;
	printf("Unesite N:");
	scanf_s("%d", &n);
	printf("Unesite elemente niza.\n");
	for (i = 0; i < n; i++) {
		scanf_s("%d", &a[i]);
	}
	x = a[n - 1];
	for (i = n; i > 0; i--) {
		a[i] = a[i - 1];
	}
	a[0] = x;
	for (i = 0; i < n; i++) {
		printf("Element[%d] = %d\n", i, a[i]);
	}
}
```

Zadatak 4. Dat je niz celih brojeva a dužine n. Rotirati elemente niza za k mesta

1. u levo;
2. u desno.

Rešenje:

```
void zad4_desno() {
	int a[30];
	int i, n, x, j, k;
	printf("Unesite N:");
	scanf_s("%d", &n);
	printf("Unesite K:");
	scanf_s("%d", &k);
	printf("Unesite elemente niza.");
	for (i = 0; i < n; i++) {
		scanf_s("%d", &a[i]);
	}
	for (j = 0; j < k; j++) {
		x = a[n - 1];
		for (i = n; i > 0; i--) {
			a[i] = a[i - 1];
		}
		a[0] = x;
```

```
	}


	for (i = 0; i < n; i++) {
		printf("Element[%d] = %d\n", i, a[i]);
	}
}

void zad4_levo() {
	int a[30];
	int i, n, x, k, j;
	printf("Unesite N:");
	scanf_s("%d", &n);
	printf("Unesite K:");
	scanf_s("%d", &k);
	printf("Unesite elemente niza.\n");
	for (i = 0; i < n; i++) {
		scanf_s("%d", &a[i]);
	}
	for (j = 0; j < k; j++) {
		x = a[0];
		for (i = 0; i < n - 1; i++) {
			a[i] = a[i + 1];
		}
		a[n - 1] = x;
	}
	for (i = 0; i < n; i++) {
		printf("Element[%d] = %d\n", i, a[i]);
	}
}
```

Zadatak 5. Dat je niz celih brojeva a dužine n. Zameniti svaki uzastopni niz nula jednom nulom.

Rešenje:

```
void zad5() {
	int a[30];
	int i, n, j = 0;
	int noviNiz[30];
	int nula = 0; // ako je 1, jedna 0 je upisana u novi niz u suprotnom false
	printf("Unesite N:");
	scanf_s("%d", &n);
	printf("Unesite elemente niza.\n");
	for (i = 0; i < n; i++) {
		scanf_s("%d", &a[i]);
	}
	for (i = 0; i < n; i++) {
		if (a[i] != 0) {
			noviNiz[j++] = a[i];
			nula = 0;
		}
		else {
			if (!nula) {
				noviNiz[j++] = 0;
				nula = 1;
			}
		}
	}
```

```
	for (i = 0; i < j; i++) {
		printf("Element[%d] = %d\n", i, noviNiz[i]);
	}
}
```

Zadatak 6. Dat je niz celih brojeva a dužine n. Odrediti dužinu najdužeg neprekidnog niza nula.

Rešenje:

```
void zad6() {
	int a[30];
	int i, n, maxD = 0, trD = 0;
	printf("Unesite N:");
	scanf_s("%d", &n);
	printf("Unesite elemente niza.\n");
	for (i = 0; i < n; i++) {
		scanf_s("%d", &a[i]);
	}
	for (i = 0; i < n; i++) {
		if (a[i] == 0) {
			trD++;
			if (trD > maxD) {
				maxD = trD;
			}
		}
		else {
			trD = 0;
		}
	}
	printf("Duzina najveceg uzatopnog niza nula je %d\n", maxD);
```

Zadatak 7. N dece je raspoređeno u krug i igraju sledeću igru: prvo dete počinje brojanje od deteta koje je desno od njega i broji k dece u desno. Dete koje je na k – tom mestu ispada iz igre, a prvo sledeće dete počinje brojanje na isti način. Odrediti koje dete će poslednje ostati u igri.

Rešenje:

```
void zad7() {
	int a[20];
	int n, i, k, tmp = 0, brOstalih;
	int ok = 1;
	printf("Unesite N:");
	scanf_s("%d", &n);
	// 1 - dete koje je u igri, 0 - dete izbaceno iz igre
	// na pocetku sva deca su u igri
	for (i = 0; i < n; i++) {
		a[i] = 1;
	}
	printf("Unesite K:");
	scanf_s("%d", &k);
	// brojanje se pocinje od prvog deteta
	i = 0;
	brOstalih = n;
	while (brOstalih > 1) {
		// indeks sledeceg deteta
		if (i == n - 1) {
			i = 0;
```

```
			}
			else {
				i++;
			}
			// printf("I = %d \n", i);
			// printf("A[%d] = %d\n", i, a[i]);
			// brojanje koraka
			if (a[i] == 1) {
				// printf("Tmp = %d\n", tmp);
				tmp++;
			}
			// izbacivanje deteta iz igre, a[i] = 0
			if (tmp == k) {
				a[i] = 0;
				tmp = 0;
				brOstalih--;
				// printf("Izbacuje se %d, brpreostalih = %d\n", i, brOstalih);
			}
			// printf("\n");
		}
		printf("\n\n\n");
		if (brOstalih == 1) {
			for (i = 0; i < n; i++) {
				if (a[i] == 1) {
					printf("Indeks deteta je %d \n", i + 1);
				}
			}
		}
}
```

Zadatak 8. Urediti dati niz celih brojeva

a. U neopadajući redosled.
b. U neopadajući redosled po kriterijumu rastojanja od aritmetičke sredine niza.
c. Tako da na početku budu prvo parni brojevi uređeni u neopadajući redosled, a za njima neparni brojevi uređeni u nerastući redosled.

Rešenje:

```
void zad8_a() {
		int a[30];
		int i, n, j, tmp;
		printf("Unesite N:");
		scanf_s("%d", &n);
		printf("Unesite elemente niza.\n");
		for (i = 0; i < n; i++) {
			scanf_s("%d", &a[i]);
		}
		printf("Original.\n");
		for (i = 0; i < n; i++) {
			printf("Element[ %d ] = %d\n", i, a[i]);
		}
		for (i = 0; i < n; i++) {
			for (j = i + 1; j < n; j++) {
				if (a[i] > a[j]) {
					tmp = a[i];
					a[i] = a[j];
```

```
                        a[j] = tmp;
                    }
                }
        }
        printf("\n\nSort.\n");
        for (i = 0; i < n; i++) {
                printf("Element[ %d ] = %d\n", i, a[i]);
        }
}


void zad8_b() {
        int a[50], nizRazlika[50];
        int i, n, j, as, suma = 0, tmp1, tmp2;

        printf("Unesite N:");
        scanf_s("%d", &n);
        printf("Unesite elemente niza.\n");
        for (i = 0; i < n; i++) {
                scanf_s("%d", &a[i]);
        }
        for (i = 0; i < n; i++) {
                suma += a[i];
        }
        as = suma / n;
        for (i = 0; i < n; i++) {
                if (a[i] > as) {
                        nizRazlika[i] = a[i] - as;
                }
                if (a[i] < as) {
                        nizRazlika[i] = as - a[i];
                }
                if (a[i] == as) {
                        nizRazlika[i] = 0;
                }
        }
        printf("\nNiz razlika pre sortiranja.\n");
        for (i = 0; i < n; i++) {
                printf("Element[ %d ] = %d\n", i, nizRazlika[i]);
        }
        for (i = 0; i < n; i++) {
                for (j = i + 1; j < n; j++) {
                        if (nizRazlika[i] > nizRazlika[j]) {
                                tmp1 = a[i];
                                a[i] = a[j];
                                a[j] = tmp1;
                                tmp2 = nizRazlika[i];
                                nizRazlika[i] = nizRazlika[j];
                                nizRazlika[j] = tmp2;
                        }
                }
        }
        printf("\n\nAritmeticka sredina: %d\n", as);
        printf("\n\nNiz razlika posle sortiranja.\n");
        for (i = 0; i < n; i++) {
                printf("Element[ %d ] = %d\n", i, nizRazlika[i]);
        }
        printf("\n\nNiz pocetni posle sortiranja.\n");
```

```
	for (i = 0; i < n; i++) {
		printf("Element[ %d ] = %d\n", i, a[i]);
	}
}


void zad8_c() {
	int a[30], parni[30], neparni[30], noviNiz[30];
	int i, n, j, k, x, tmp;
	printf("Unesite N:");
	scanf_s("%d", &n);
	printf("Unesite elemente niza.\n");
	for (i = 0; i < n; i++) {
		scanf_s("%d", &a[i]);
	}
	printf("Original.\n");
	for (i = 0; i < n; i++) {
		printf("Element[ %d ] = %d\n", i, a[i]);
	}
	k = 0;
	j = 0;
	for (i = 0; i < n; i++) {
		if (a[i] % 2 == 0) {
			parni[k++] = a[i];
		}
		else {
			neparni[j++] = a[j];
		}
	}
	// k - broj parnih
	for (i = 0; i < k; i++) {
		for (x = i + 1; x < k; x++) {
			if (parni[i] > parni[x]) {
				tmp = parni[i];
				parni[i] = parni[x];
				parni[x] = tmp;
			}
		}
	}
	// j - broj neparnih
	for (i = 0; i < j; i++) {
		for (x = i + 1; x < j; x++) {
			if (neparni[i] < neparni[x]) {
				tmp = neparni[i];
				neparni[i] = neparni[x];
				neparni[x] = tmp;
			}
		}
	}
	/*
	printf("\n\nSort. Parni\n");
	for (i = 0; i < k ; i++ ) {
	printf("Element[ %d ] = %d\n", i, parni[i] );
	}
	printf("\n\nSort. Neparni\n");
	for (i = 0; i < j ; i++ ) {
	printf("Element[ %d ] = %d\n", i, neparni[i] );
	}
```

```
	*/
	printf("\n");
	for (i = 0; i < k + j; i++) {
		if (i < k) {
			noviNiz[i] = parni[i];
		}
		else {
			noviNiz[i] = neparni[i - k];
		}
		printf("Element[ %d ] = %d\n", i, noviNiz[i]);
	}
}
```

Zadatak 9. Odrediti podniz uzastopnih elemenata datog niza koji ima najveću sumu.

Rešenje:

```
void zad9() {
	int a[30];
	int i, n, j, poc = 0, kraj = 0, trSuma = 0, maxSuma = 0;
	printf("Unesite N:");
	scanf_s("%d", &n);
	printf("Unesite elemente niza.\n");
	for (i = 0; i < n; i++) {
		scanf_s("%d", &a[i]);
	}
	printf("Original.\n");
	for (i = 0; i < n; i++) {
		printf("Element[ %d ] = %d\n", i, a[i]);
	}
	for (i = 0; i < n; i++) {
		trSuma = 0;
		for (j = i; j < n; j++) {
			trSuma += a[j];
			if (trSuma > maxSuma) {
				maxSuma = trSuma;
				poc = i;
				kraj = j;
			}
		}
	}
	printf("\n\n Suma: %d\n", maxSuma);
	for (i = poc; i < kraj + 1; i++) {
		printf("Element[ %d ] = %d\n", i, a[i]);
	}
}
```

Zadatak 10. Na takmičenju je učestvovalo n (n ≤ 1000) učenika. Oni su mogli da osvoje između 0 i 100 poena. U nizu a su dati rezultati za svakog učenika.

a. Ukoliko je data granica p koja određuje minimalan broj potrebnih poena za prolaz u sledeći krug takmičenja, odrediti koliko učenika prolazi dalje.
b. Odrediti granicu p tako da broj učenika koji prolazi dalje nije veći od unapred zadatog broja k.

Rešenje:

```
void zad10_a() {
```

```
	int a[1000];
	int n, i, p, br = 0;
	printf("Unesite N:");
	scanf_s("%d", &n);
	printf("Unesite P:");
	scanf_s("%d", &p);
	printf("Unesite poene.\n");
	for (i = 0; i < n; i++) {
		scanf_s("%d", &a[i]);
	}
	for (i = 0; i < n; i++) {
		if (a[i] >= p) {
			br++;
		}
	}
	printf("Broj ucenika koji prolazi dalje je %d\n", br);
}

void zad10_b() {
	int a[1000];
	int n, i, tmp, j, k;
	printf("Unesite N:");
	scanf_s("%d", &n);
	printf("Unesite K:");
	scanf_s("%d", &k);
	printf("Unesite poene.\n");
	for (i = 0; i < n; i++) {
		scanf_s("%d", &a[i]);
	}
	for (i = 0; i < n; i++) {
		for (j = i + 1; j < n; j++) {
			if (a[i] > a[j]) {
				tmp = a[i];
				a[i] = a[j];
				a[j] = tmp;
			}
		}
	}
	printf("Granica je %d\n", a[n - k]);
}
```

Zadatak 11. Dat je niz karaktera koji predstavlja izraz sa zagradama. Proveriti da li su zagrade ispravno zadate (interesuju nas samo zagrade, a ne i ostatak izraza).

Rešenje:

```
void zad11() {
	char a[50];
	int n, i, brO = 0, brZ = 0;
	printf("Duzina izraza: ");
	scanf_s("%d", &n);
	printf("Unesite izraz.");
	for (i = 0; i < n; i++) {
		scanf_s(" %c", &a[i]);
	}
	for (i = 0; i < n; i++) {
```

```
			if (a[i] == '(') {
				brO++;
			}
			if (a[i] == ')') {
				brZ++;
			}
			if (brZ > brO) {
				printf("Greska!!!!!");
				break;
			}
		}
		if (brO == brZ) {
			printf("\nOk.\n");
		}
		else {
			printf("Greska!!!!!");
		}
		/*
		printf("\n");
		for ( i = 0; i < n; i++){
		printf("%c", a[i]);
		}
		*/
		printf("\n");
}
```

Zadatak 12. Data su dva niza karaktera a i b sa po n elemenata. Odrediti da li je niz b anagram niza a.

Rešenje:

```
void zad12_anagram() {
		char a[20], b[20];
		int n, i, j = 0, tmp;
		int ok = 1;
		printf("Unesite N:");
		scanf_s("%d", &n);
		printf("Unesite elemente niza A.\n");
		for (i = 0; i < n; i++) {
			scanf_s(" %c", &a[i]);
		}
		printf("Unesite elemente niza B.\n");
		for (i = 0; i < n; i++) {
			scanf_s(" %c", &b[i]);
		}
		/*
		printf("\nA:\n");
		for (i = 0; i < n ; i++ ) {
		printf("Element[%d] = %c\n", i, a[i] );
		}
		printf("\nB:\n");
		for (i = 0; i < n ; i++ ) {
		printf("Element[%d] = %c\n", i, b[i] );
		}
		*/
		for (i = 0; i < n; i++) {
			for (j = i + 1; j < n; j++) {
				if (a[i] > a[j]) {
```

```
                        tmp = a[i];
                        a[i] = a[j];
                        a[j] = tmp;
                }
        }
    }
    for (i = 0; i < n; i++) {
        for (j = i + 1; j < n; j++) {
            if (b[i] > b[j]) {
                tmp = b[i];
                b[i] = b[j];
                b[j] = tmp;
            }
        }
    }
    /*
    printf("\n\nSort.");
    printf("\nA:\n");
    for (i = 0; i < n ; i++ ) {
    printf("Element[%d] = %c\n", i, a[i] );
    }
    printf("\nB:\n");
    for (i = 0; i < n ; i++ ) {
    printf("Element[%d] = %c\n", i, b[i] );
    }
    */
    for (i = 0; i < n; i++) {
        if (a[i] != b[i])
            ok = 0;
    }
    if (ok == 1) {
        printf("\nJesu\n");
    }
    else {
        printf("\nNisu\n");
    }
}
```

Zadatak 13. Dati su nizovi a i b koji, redom, imaju n i m elemenata. Svi elementi ovih nizova su cifre. Ovim nizovima su predstavljena dva broja. Formirati niz c koji predstavlja zbir brojeva datih sa prva dva niza.

Rešenje:

```
void zad13() {
    int a[20], b[20], c[20], noviB[20], noviA[20];
    int m, n, i, j, prenos = 0, ost = 0, max = 0;
    // n = duz(a), m = duz(b)
    printf("Unesite N:");
    scanf_s("%d", &n);
    printf("Unesite M:");
    scanf_s("%d", &m);


    printf("\nUnesite A:");
    for (i = 0; i < n; i++) {
        scanf_s("%d", &a[i]);
```

```
	}
	printf("\nUnesite B:");
	for (i = 0; i < m; i++) {
		scanf_s("%d", &b[i]);
	}
	j = 0;
	if (n > m) {
		for (i = 0; i < n; i++) {
			if (i < n - m) {
				noviB[i] = 0;
			}
			else {
				noviB[i] = b[j++];
			}
		}
		max = n;
		for (i = max - 1; i > -1; i--) {
			printf("\nA[ %d ] = %d\t", i, a[i]);
			printf("B[ %d ] = %d\n", i, noviB[i]);
		}
		prenos = 0;
		for (i = max - 1; i > -1; i--) {
			ost = (a[i] + noviB[i]) % 10;
			c[i] = ost + prenos;
			prenos = (a[i] + noviB[i]) / 10;
		}
	}
	else {
		for (i = 0; i < m; i++) {
			if (i < m - n) {
				noviA[i] = 0;
			}
			else {
				noviA[i] = a[j++];
			}
		}
		max = m;
		for (i = max - 1; i > -1; i--) {
			printf("\nA[ %d ] = %d\t", i, noviA[i]);
			printf("B[ %d ] = %d\n", i, b[i]);
		}
		prenos = 0;
		for (i = max - 1; i > -1; i--) {
			ost = (noviA[i] + b[i]) % 10;
			c[i] = ost + prenos;
			prenos = (noviA[i] + b[i]) / 10;
		}
	}
	printf("\n\n\n");
	for (i = 0; i < max; i++)
	{
		printf(" %d ", c[i]);
	}
}
```

Zadatak 14. Polinom $a_0 + a_1x + \cdots + a_nx^n$ se može predstaviti nizom koeficijenata $a_0,\ldots,a_n$. Napisati program kojim se određuju koeficijenti:

a. zbira dva polinoma;
b. proizvoda dva polinoma.

Rešenje:

```
void zad14_a() {
	int a[20], b[20], zbir[20];
	int n, i;
	// pretpostavimo da su polinomi istog stepena N
	printf("Unesite N:");
	scanf_s("%d", &n);
	printf("Polinom a:");
	for (i = 0; i < n; i++) {
		scanf_s("%d", &a[i]);
	}
	printf("Polinom b:");
	for (i = 0; i < n; i++) {
		scanf_s("%d", &b[i]);
	}
	for (i = 0; i < n; i++) {
		zbir[i] = a[i] + b[i];
	}
	printf("Koeficijenti polinoma koji predstavlja zbir.\n");
	for (i = n - 1; i > -1; i--) {
		printf("Koeficijent[ %d ] = %d\n", i, zbir[i]);
	}
}

void zad14_b() {
	int a[20], b[20], proizvod[20];
	int n, i, j;
	printf("Unesite stepen polinoma N:");
	scanf_s("%d", &n);

	n++;
	printf("Polinom a:");
	for (i = 0; i < n; i++) {
		scanf_s("%d", &a[i]);
	}
	printf("Polinom b:");
	for (i = 0; i < n; i++) {
		scanf_s("%d", &b[i]);
	}
	for (i = 0; i < n + n; i++) {
		proizvod[i] = 0;
	}
	for (i = 0; i < n; i++) {
		for (j = 0; j < n; j++)
		{
			proizvod[i + j] += a[i] * b[j];
		}
	}
	printf("Koeficijenti polinoma koji predstavlja proizvod.\n");
	for (i = n + n - 2; i > -1; i--) {
```

```
			printf("Koeficijent[ %d ] = %d\n", i, proizvod[i]);
		}
}
```

Zadatak 15. U nekom gradu ima n zgrada. Visine tih zgrada (broj spratova svake od njih) su date u nizu a. Odrediti za svaki broj $0 < m \leq n$ koliko je najmanje potrebno dograditi spratova da bi u gradu bilo bar m zgrada sa istim brojem spratova.

Rešenje:

```
void zad15() {
	int a[20];
	int n, i, m, j, tmp, min = INT_MAX, indL, indD;
	printf("Broj zgrada je: ");
	scanf_s("%d", &n);
	printf("M: ");
	scanf_s("%d", &m);
	for (i = 0; i < n; i++) {
		scanf_s("%d", &a[i]);
	}
	for (i = 0; i < n; i++) {
		for (j = i + 1; j < n; j++) {
			if (a[i] > a[j]) {
				tmp = a[i];
				a[i] = a[j];
				a[j] = tmp;
			}
		}
	}
	for (i = 0; i < n - m; i++) {
		tmp = 0;
		for (j = i; j < i + m - 1; j++) {
			tmp += a[i + m - 1] - a[j];
		}
		if (tmp < min) {
			min = tmp;
			indL = i;
			indD = i + m - 1;
		}
	}
	printf("Najmanje spratova: %d\n", min);
	for (i = indL; i < indD + 1; i++) {
		printf("%d ", a[i]);
	}
}
```

Nizovi (nastavak)

Zadatak 16. Napisati program u kojem se unosi niz celih brojeva, maksimalno 10. Formirati novi niz čiji su elementi parni elementi prvog niza, a umesto neparnih brojeva staviti nule. Prikazati novodobijeni niz.

Primer: a=(2, 8, 5, 7, 4) => b=(2, 8, 0, 0, 4)

Rešenje:

```
program p;
var
a,b:array [1..10] of integer;
i,n:integer;
begin
  writeln('koliko elemenata ima niz?');
  readln(n);
  for i:=1 to n do
  begin
    write('unesi ',i,'. element: ');
    readln(a[i]);
    if a[i] mod 2=0 then b[i]:=a[i];
  end;
  writeln('novi niz');
  for i:=1 to n do
    write(b[i]:5);
  readln;
end.
```

Zadatak 17. Napisati program u kojem se unose dva niza od n celih brojeva. Formirati novi niz čiji su elementi veći brojevi istog indeksa iz unetih nizova. Prikazati novodobijeni niz.

Primer: a=(2, 8, 5, 7, 4) b=(4, 5, 3, 10, 1) => c=(4, 8, 5, 10, 4)

Rešenje:

```
program p;
var
a,b,c:array [1..10] of integer;
i,n:integer;
begin
  writeln('koliko elemenata ima niz?');
  readln(n);
  writeln('unos prvog niza');
  for i:=1 to n do
    begin
      write('unesi ',i,'. element: ');
      readln(a[i]);
    end;
  writeln('unos drugog niza');
  for i:=1 to n do
  begin
    write('unesi ',i,'. element: ');
    readln(b[i]);
  end;
  for i:=1 to n do
```

```
    if a[i] > b[i] then c[i]:=a[i] else c[i]:=b[i];
  writeln('novi niz');
  for i:=1 to n do
    write(c[i]:5);
  readln;
end.
```

Zadatak 18. Napisati program koji za uneti niz od maksimalno 10 članova prebrojava koliko je članova niza jednako jedan.

Rešenje:

```
program p;
var a:array [1..10] of integer;
i,n,br:integer;
begin
  writeln('koliko elemenata ima niz?');
  readln(n);
  for i:=1 to n do
    begin
      write('unesi ',i,'. element: ');
      readln(a[i]);
    end;
   br:=0;
   for i:=1 to n do
     if a[i]=1 then br:=br+1;
     writeln('elemenat jednakih jedan ima ',br);
     readln;
end.
```

Zadatak 19. Napisati program koji učitava niz celih brojeva i prikazuje sve elemente manje od 10.

Rešenje:

```
program p;
var a:array [1..10] of integer;
i,n,br:integer;
begin
  writeln('koliko elemenata ima niz?');
  readln(n);
  for i:=1 to n do
    begin
      write('unesi ',i,'. element: ');
      readln(a[i]);
    end;
  br:=0;
  writeln('elementi manji od 10 su');
  for i:=1 to n do
    if a[i]<10 then begin br:=br+1; write(a[i]:5);end;
  writeln;
  writeln('ukupno ih ima ',br);
  readln;
end.
```

Zadatak 20. Napisati program u kojem se unosi niz celih brojeva. Prikazati prosečnu vrednost elemenata niza i koliko je elemenata niza veće od prosečne vrednosti.

Rešenje:

```
program p;
var
a:array [1..10] of integer;
i,n,s,br:integer; pr:real;
begin
  writeln('koliko elemenata ima niz?');
  readln(n);
  s:=0;br:=0;
  writeln('unos niza');
  for i:=1 to n do
    begin
      write('unesi ',i,'. element: ');
      readln(a[i]);
      s:=s+a[i];
    end;
  pr:=s/n;
  for i:=1 to n do
    if a[i]>pr then br:=br+1;
  writeln('prosecna vrednost elemenata niza je ',pr:0:2,' vecih
  od proseka ima ',br,' elemenata');
  readln;
end.
```

Zadatak 21. Unosi se niz realnih brojeva. Formirati novi niz čiji su elementi zaokruženi elementi unetog niza.

Rešenje:

```
program p;
var a:array [1..10] of real;
b:array [1..10] of integer;
i,n:integer;
begin
  writeln('koliko elemenata ima niz?');
  readln(n);
  for i:=1 to n do
    begin
      write('unesi ',i,'. element: ');
      readln(a[i]);
      b[i]:=round(a[i]);
    end;
  writeln('novi niz');
  for i:=1 to n do
    write(b[i]:5);
  readln;
end.
```

Zadatak 22. Napisati program u kojem se unosi niz celih brojeva, a zatim se sortira po rastućem redosledu. Prikazati sortirani niz.

Rešenje:

```
program p21;
var a,b: array [1..100] of integer;
n,i,j,p,k:integer;
begin
  write ('koliko elemenata ima niz?');
  readln(n);
  writeln('unesi elemente niza');
  for i:=1 to n do
    begin
      write('unesi ',i,'. element: ');
      readln(a[i]);
    end;
    for i:=1 to n-1 do
      for j:=i+1 to n do
        if a[j]<a[i] then
          begin
            p:=a[i];
            a[i]:=a[j];
            a[j]:=p;
          end;
    for i:=1 to n do
      write(a[i],' ');
    readln;
end.
```

Zadatak 23. Napisati program u kojem se unosi niz celih brojeva, a zatim se računa zbir pozitivnih i zbir negativnih elemenata niza.

Rešenje:

```
program p;
var
a:array [1..10] of integer;
i,n,sp,sn:integer;
begin
  writeln('koliko elemenata ima niz?');
  readln(n);
  sp:=0;sn:=0;
  writeln('unos niza');
  for i:=1 to n do
    begin
      write('unesi ',i,'. element: ');
      readln(a[i]);
      if a[i]>0 then sp:=sp+1 else sn:=sn+1;
    end;
  writeln('pozitivnih elemenata ima ',sp,' a negativnih (i nule)    ima
',sn);
  readln;
end.
```

Zadatak 24. Napisati program u kojem se unosi niz celih brojeva, a zatim se traži najveći elemenat niza. Prikazati najveći elemenat i njegov indeks.

Rešenje:

```
program p;
var a:array [1..100] of integer;
i,n,maxi,max:integer;
begin
  writeln('koliko elemenata ima niz?');
  readln(n);
  for i:=1 to n do
    begin
      write('unesi ',i,'. element: ');
      readln(a[i]);
    end;
  max:=-maxint;
  for i:=1 to n do
    if a[i]>max then begin
      max:=a[i];
      maxi:=i;
    end;
  writeln('Najveci element je ',max,' a njegov indeks je ',maxi);
  readln;
end.
```

Zadatak 25. Zelimo da se kandidujemo za ministra prosvete. U vladi postoji **n** stranaka, a u **i**-toj $\mathbf{a_i}$ poslanika. Da bi stranka glasala za nas potrebno je da vise od pola poslanika iz te stranke bude za nas. Da bismo dobili na izborima potrebno je da vise od pola od ukupnog broja stranaka glasa za nas. Koliko poslanika najmanje moramo da podmitimo da bismo dobili izbore?

Izlaz:

Na standardni izlaz ispisati najmanji broj poslanika koje moramo podmititi da bismo dobili izbore.

Primer:

Ulaz:	**Izlaz:**
3 6 5 6	7

Objašnjenje.

Ukoliko podmitimo 4 poslanika iz prve i 3 poslanika iz druge stranke, ove dve stranke (od tri) će glasati za nas i mi pobedjujemo. Nije moguće pobediti tako da se podmiti manje od 7 poslanika.

Rešenje:

```
program podmicivanje;
  const k=20;
  type
    niz=array [1..k] of integer;
  var
    n,i,j,min,pom:integer;
    a:niz;

  procedure citaj(n:integer; var a:niz);
    var i:integer;
    begin
```

```
    for i:=1 to n do
      readln(a[i])
  end;

procedure pisi(n:integer; var a:niz);
  var i:integer;
  begin
    for i:=1 to n do
      writeln(a[i])
  end;
begin
write('Unesi broj stranaka: ');
readln(n);
writeln('Unesi broj poslanika za svaku stranku: ');
for i:=1 to n do
  read(a[i]);
{sortiramo niz u rastuci poredak}
for i:=1 to n-1 do
  for j:=i+1 to n do
    if(a[i] > a[j]) then
     begin
       pom:=a[i];
       a[i]:=a[j];
       a[j]:=pom;
     end;
writeln('Sortiran niz izgleda: ');
pisi(n,a);
min:=0;
for i:= 1 to (n div 2) + 1 do
  min := min + (a[i] div 2) +1;
writeln('Minimalni broj poslanika koji treba potkupiti je: ');
writeln(min);
end.
```

Zadatak 26. Napisati program koji računa n-ti član Fibonačijevog niza, koji je definisan sa F[0] = 0, F [1] = 1 i F [n+1] = F [n] + F [n– 1]. Na primer, za n = 6 odgovor je 8.

Rešenje:

```
program fibonaci;
const k = 20;
type
  niz = array [1..k] of integer;
var
  n,res:integer;

  function fibonaci(n:integer):integer;
    var
      f1,f2,next,i:integer;
    begin
      f1:=0;
      f2:=1;
      for i:= 3 to n do
       begin
         next := f1 + f2;
         f1:=f2;
```

```
        f2:=next;
      end;
      fibonaci := next;
    end;
begin
  writeln('Koji clan fibonacijevog niza zelite: ');
  readln(n);
  res:=fibonaci(n);
  writeln(n,'-ti clan fibonacijevog niza je: ',res);
end.
```

Zadatak 27. Dat je polinom P(x) = a[n] x^n + a[n – 1] x^{n-1} + … + a[1] x + a[0] sa realnim koeficijentima a[0], a[1], ..., a[n]. Odrediti vrednost polinoma u tački x. Na primer, za polinom P (x) = x^3 + 2x – 1 stepena 3 sa koeficijentima (-1, 2, 0, 1) i x = 2 – odgovor je 11.

Rešenje:

```
program polinomi;
const k = 20;
type
  niz = array [0..k] of real;
var
  n,i:integer;
  x,xn,res:real;
  p:niz;

procedure citaj(n:integer; var a:niz);
  var i:integer;
  begin
    for i:=0 to n do
      readln(a[i])
  end;

procedure pisi(n:integer; var a:niz);
  var i:integer;
  begin
    for i:=0 to n do
    begin
      write(a[i]:10:2,'*x^',i);
      if i <> n then
        write(' + ');
    end;
    writeln();
  end;
begin
  writeln('Unesi red polinoma: ');
  readln(n);
  writeln('Unesi koeficijente polinoma pocev od slobodnog clana');
  citaj(n,p);
  writeln('Uneti polinom je:');
  pisi(n,p);
  writeln('U kojoj tacki zelis vrednost polinoma?');
  readln(x);
  xn:=1;
  res:=0;
  for i:=0 to n do
```

```
    begin
      res:=res + p[i]*xn;
      xn:=xn*x;
    end;
  writeln('Vrednost polinoma u zadatoj tacki je: ',res:10:2);
end.
```

Zadatak 28. Dat je niz brojeva A = (a_1, ..., a_N). Posmatrajmo skup svih suma uzasotpnih članova S = {a_i + ... + a_j | 1 ≤ i ≤ j ≤N}.

1. Ispisati vrednost iz skupa S koja se najčešće pojavljuje, kao i koliko puta se pojavljuje. U slučaju da ima više takvih, ispisati onu čija je vrednost najveća.

Ulaz.

(Ulazni podaci se učitavaju sa standardnog ulaza.) U prvom redu standardnog ulaza nalazi se prirodan broj N (1 ≤ N ≤ 3000). U sledećem redu nalazi se N prirodnih brojeva, redom a_1 , ..., a_N, svaki iz intervala [0, 3000].

Izlaz.

(Izlazne podatke ispisati na standardan izlaz.) U prvi i jedini red standardog izlaza ispisati dva prirodna broja, koji redom predstavljaju, broj koji se najšeće pojavljuje u skupu S, i koliko puta se pojavljuje. U slučaju da postoji više takvih brojeva, ispisati onaj koji ima najveću vrednost.

Ograničenja.

U 30% test primera će biti 1 ≤ N ≤ 100.

Primer 1.

standardni ulaz	standardni izlaz
3	32
1 2 3	

Objašnjenje.

Skup S = {1, 1 + 2, 1 + 2 + 3, 2, 2 + 3, 3} = {1, 3, 6, 2, 5, 3}.

Primer 2.

standardni ulaz	standardni izlaz
8	303
17 13 17 13 5 6 5 6	

Objašnjenje.

Primetimo da se i suma 11 pojavljuje 3 puta, ali je 30 veća suma.

Rešenje:

```
#include <iostream>
#include <cstdio>
#define ffor(_a,_f,_t) for(int _a=(_f),__t=(_t);_a<__t;_a++)
#define SET(__set, val) memset(__set, val, sizeof(__set))
#define FOR(__i, __n) ffor (__i, 0, __n)
using namespace std;
int cnt[9000001], sum[3001];

int main() {
	int n, val;
	scanf("%d", &n);
```

```
	sum[0] = 0;
	FOR(i, n) {
		scanf("%d", &val);
		sum[i + 1] = sum[i] + val;
	}

	SET(cnt, 0);
	ffor(i, 1, n + 1) {
		val = sum[i];
		FOR(j, i)
			cnt[val - sum[j]]++;
	}
	int mm = 0, ret = -1;
	FOR(i, 9000001)
		if (cnt[i] >= mm) {
			mm = cnt[i];
			ret = i;
		}
	printf("%d %d\n", ret, mm);
	return 0;
}
```

Zadatak 29. Will Rogers fenomen se dobija kada se element iz jednog skupa prebaci u drugi, pri čemu se srednje vrednosti oba skupa povećaju. Baziran je na citatu Willa Rogera:
Kada su Okie (domorovci) napustile Oklahomu i preselili se u Kaliforniju, podigli su prosečan nivo inteligencije u obe države.
Data su dva skupa prirodnih brojeva a i b veličine n odnosno m. Naći broj elemenata skupa a koji prebacivanjem u skup b povećavaju prosečne vrednosti oba skupa.
Ulaz.
(Ulazni podaci se učitavaju sa standardnog ulaza.) U prvom redu standarnog ulaza nalaze se dva prirodna broja n i m($2 \leq n, m \leq 5000$). Naredna dva reda sadrže po n i m prirodnih brojeva koji predstavljaju elemente nizova a i b, redom. Elementi nizova su iz segmenta [1, 1000].
Izlaz.
(Izlazne podatke ispisati na standardan izlaz.) U prvom i jedinom redu standardnog izlaza ispisati traženi broj.
Primer 1.

standardni ulaz	**standardni izlaz**
3 3 6 5 4 1 2 3	1

Objašnjenje.
Jedini element koji zadovoljava uslove je broj 4.

Rešenje:

```
#include<stdio.h>
#define MAX_N 5005
int n, m, a[MAX_N], b[MAX_N], sol;
double avgA, avgB;
void input()
```

```
{
	scanf("%d %d", &n, &m);
	for (int i = 0; i < n; i++)
		scanf("%d", &a[i]);
	for (int i = 0; i < m; i++)
		scanf("%d", &b[i]);
}

void solve()
{
	avgA = 0;
	for (int i = 0; i < n; i++)
		avgA += a[i];
	avgA = avgA / n;

	avgB = 0;
	for (int i = 0; i < m; i++)
		avgB += b[i];
	avgB = avgB / m;
	sol = 0;
	for (int i = 0; i < n; i++)
		if ((a[i] < avgA) && (avgB < a[i]))
			sol++;
}
int main()
{
	input();
	solve();
	printf("%d\n", sol);
	return 0;
}
```

Zadatak 30. Dato je n segmenata i m tačaka na x-osi. Za svaku od datih m tačaka odrediti broj segmenata kojima ona pripada. Tačka x pripada segmentu $[a, b]$ ako je $a \leq x \leq b$.

Ulaz.

(Ulazni podaci se učitavaju sa standardnog ulaza.) U prvom redu standradnog ulaza nalaze se dva prirodna broja $n \leq 10^5$ i $m \leq 10^5$ - broj segmenata i broj tačaka, redom. U sledećem redu se nalaze m brojeva razdvojenih razmakom - koordinate tačaka. U sledećih n redova se nalaze po dva broja razdvojena razmakom - leva i desna koordinata odgovarajućeg segmenta (leva koordinata je strogo manja od desne). Sve koordinate su prirodni brojevi ne veći od 10^9.

Izlaz.

(Izlazne podatke ispisati na standardan izlaz.) Na standardni izlaz za svaku tačku ispisati broj segmenata kojima ona pripada, svaki broj u posebnom redu i u redosledu kojim su tačke date na ulazu.

Primer 1.

standardni ulaz	**standardni izlaz**
3 4 5 1 8 9 6 7 4 9 2 5	2 0 1 1

Rešenje:

```
const
  MaxN = 100010;
  MaxM = 100010;
type
  Point = Record
    x: longint;
    t: longint;
    num: longint;
  end;

var
  a : array[0..2*MaxN + MaxM] of Point;
  sol : array[0..MaxM] of longint;
  n, m, i, sum : longint;
  x, y : Point;
procedure QS(l, r : longint);
var
  i, j: longint;
begin
  i := l; j := r; x := a[(l + r) DIV 2];
  repeat
    while ((a[i].x < x.x) or ((a[i].x = x.x) and (a[i].t > x.t))) do i :=
i + 1;
    while ((x.x < a[j].x) or ((x.x = a[j].x) and (x.t > a[j].t))) do j :=
j - 1;
    if (i <= j) then begin
      y := a[i]; a[i] := a[j]; a[j] := y;
      i := i + 1; j := j - 1;
    end;
  until (i > j);
  if (l < j) then QS(l, j);
  if (i < r) then QS(i, r);
end;
begin
   readln(n, m);
   for i := 0 to m - 1 do begin
     read(a[i].x);
     a[i].t := 0;
     a[i].num := i;
   end;
   for i := 0 to n - 1 do begin
     readln(a[m + 2*i].x, a[m + 2*i + 1].x);
     a[m + 2*i].t := 1;
     a[m + 2*i + 1].t := -1;
   end;
   QS(0, m + 2 * n - 1);
   sum := 0;
   for i := 0 to m + 2 * n - 1 do begin
     if (a[i].t = 0) then
       sol[ a[i].num ] := sum
     else
       sum := sum + a[i].t;
   end;
```

```
   for i := 0 to m - 1 do
     writeln(sol[i]);
end.
```

Zadaci za samostalni rad

1. Napisati program koji računa zbir niza koji je definisan kao:
   S = 0 – 1 + 2 – 3 + 4 ... $(-1)^n n$.

2. Napisati program za proveru tačnosti matematiče formule za izračunavanje sume:
   1 + 3 + 5 + ... + (2n – 1) = n^2.

3. Izračunati sumu parnih dvocifrenih brojeva (djeljivih sa 2).

4. Izračunati sumu neparnih dvocifrenih brojeva (nisu djeljivih sa 2).

5. Izračunati sumu dvocifrenih brojeva djeljivih sa 3.

6. Izračunati sumu dvocifrenih brojeva koji nisu djeljivi sa 3.

7. Izračunati sumu dvocifrenih brojeva kod kojih je pri djeljenju sa 3 ostatak 1.

8. Izračunati sumu dvocifrenih brojeva kod kojih je pri djeljenju sa 3 nije ostatak 1.

9. Izračunati sumu dvocifrenih brojeva kod kojih je pri djeljenju sa 3 ostatak 1.

10. Izračunati sumu dvocifrenih brojeva kod kojih pri djeljenju sa 3 je ostatak 2.

11. Izračunati sumu dvocifrenih brojeva kod kojih pri djeljenju sa 3 nije ostatak 2.

12. Izračunati sumu dvocifrenih brojeva cifre djeljiva sa 4.

13. Izračunati sumu dvocifrenih brojeva kod kojih je razlika cifara djeljiva sa 2.

14. Izračunati sumu dvocifrenih brojeva razlika cifara manja od 3.

15. Izračunati sumu dvocifrenih brojeva druga cifra deljiva sa 4.

16. Izračunati sumu dvocifrenih brojeva prva manja od polovine druge.

17. Izračunati sumu dvocifrenih brojeva kod kojih je suma cifara manja od 7.

18. Izračunati sumu dvocifrenih brojeva kod kojih je suma cifara veća od 7.

19. Izračunati sumu trocifrenih brojeva.

20. Izračunati sumu trocifrenih brojeva sa zadnjom cifrom 3.

21. Izračunati sumu trocifrenih brojeva sa istim ciframa.

22. Izračunati sumu trocifrenih brojeva sa različitim prvom i trećom cifrom.

23.Izračunati sumu trocifrenih brojeva sa većom prvom cifrom od druge.

24. Izračunati sumu trocifrenih brojeva brojeve sa većom drugom cifrom od treće.

25. Izračunati sumu trocifrenih brojeva sa drugom cifrom većom od 6.

26. Izračunati sumu trocifrenih brojeva sa drugom cifrom manjom od 4.

27. Izračunati sumu trocifrenih brojeva čija je suma cifara manja od 7.

28. Izračunati sumu trocifrenih brojeva djeljive sa 2.

29. Izračunati sumu trocifrenih brojeva koji nisu deljivi sa 2.

30. Izračunati sumu trocifrenih brojeva deljive sa 3.

31. Izračunati sumu trocifrenih brojeva koji nisu deljivi sa 3.

32.Izračunati sumu trocifrenih brojeva kod kojih je pri deljenju sa 3 ostatak 1.

33. Izračunati sumu trocifrenih brojeva kod kojih pri deljenju sa 3 je ostatak 2.

34. Izračunati sumu trocifrenih brojeva čija je druga cifra deljiva sa 3.

35. Izračunati sumu trocifrenih brojeva čija druga cifra nije deljiva sa 3.

36. Izračunati sumu trocifrenih brojeva čija je prva cifra deljiva sa 3 a druga cifra deljiva sa 4.

37. Izračunati sumu trocifrenih brojeva kod kojih je razlika prve dve cifre 2 a treća neparna.

38. Izračunati sumu trocifrenih brojeva kod kojih je prva veća od druge a ona od treće.

39. Izračunati sumu trocifrenih brojeva kod kojih je prva veća od suma druge dve cifre.

40. Izračunati sumu trocifrenih brojeva kod kojih je prva manja od polovine suma druge dve cifre.

41. Izračunati sumu trocifrenih brojeva kod kojih je prva manja od polovine proizvoda druge dve cifre.

42. Izračunati sumu trocifrenih brojeva koji se sastoje od susednih cifara u nizu prirodnih brojeva (npr. 123, 132, 213, 231, 312, 321).

43. Izračunati sumu trocifrenih brojeva čija je suma cifara jednaka učitanom broju **X**. Učitani broj **X** mora biti iz intervala od 1 do 27.

44. Napisati program koji ispisuje sve trocifrene brojeve koji su jedanki sumi kubova svojih cifara.

45. Ispisati sve trocifrene brojeve veće ili jednake 600.

46. Ispisati permutacije brojeva od 1 do 4.

47. Napisati program kojim se štampaju svi trocifreni brojevi abc koji imaju svojstvo $(abc)=(ab)^2-c^2$. $(147=14^2-7^2)$

48. Napisati program kojim se štampaju svi trocifreni brojevi koji imaju osobinu da su deljivi brojem koji se dobija izbacivanjem srednje cifre. (100:10, 121:11...)

49. Napisati program koji izračunava sumu recipročnih vrednosti trocifrenih brojeva.

50. Učitati članove niza. Ispisati: članove niza koji su veći od sledećeg člana.

51. Učitati članove niza. Ispisati parne članove niza.

52. Učitati članove niza. Ispisati članove niza sa neparnim indeksom.

53. Učitati članove niza. Ispisati: pozitivne, nule pa negativne članove niza.

54. Učitati članove niza. Ispisati susedne članove niza čija je suma parna.

55. Učitati članove niza. Ispisati članove niza u obrnutom redosledu.

56. Učitati članove niza. Izračunati i ispisati sumu članova niza.

57. Učitati članove niza. Izračunati i ispisati sumu članova niza sa parnim indeksom.

58. Učitati članove niza. Izračunati i ispisati sumu članova niza čiji je indeksom deljiv sa 3.

59. Učitati članove niza. Izračunati i ispisati sumu parnih i neparnih članova niza odvojeno.

60. Sabirati članove niza sve dok je suma manja od 241.

61. Izračunati sumu članova niza čija je vrednost u intervalu [3,9].

62. Učitati članove niza. Izračunati i ispisati proizvod članova niza.

63. Množiti članove niza sve dok je proizvod manja od 541.

64. Učitati članove niza. Izračunati i ispisati aritmetički sredinu svih članova niza.

65. Učitati članove niza. Izračunati aritmetičku sredinu i ispisati članove niza manje od aritmetičke sredine.

DODATAK

PITANJA SA RANIJIH PRIJEMNIH ISPITA NA STUDIJSKOM PROGRAMU INFORMATIKA NA PRIRODNO-MATEMATIČKOM FAKULTETU U NIŠU

Prijemni ispit na studijskom programu Informatika na Prirodno-matematičkom fakultetu u Nišu sastoji se od pitanja i zadataka koji mogu da budu napisani u Pascalu ili C-u (C++).

PRIMERI PITANJA

1. Prirodni brojevi su

a) 3.14, 5.15, 7.45, ...
b) 1, 2, 3, 4, 5, ...
c) ..., -2, -1, 0, 1, 2, 3, ...
d) 0, 1, 2, 3, 4, ...
e) $1e^{-1}$, $2e^{-2}$, $3e^{-3}$, $4e^{-4}$, ...

2. Obeležiti proste brojeve

a) 0,1,2,3,4,5,6,7,8,9, ...
b) 2, 3, 5, 7, 11,13, 17...
c) 1,3,5,7,9,11,13,15,17, ...
d) 2,4,6,8,10,12,14,16, ...

3. Promenljiva logičkog tipa može poprimiti sledeće vrednosti:

a) Manje od, jednako, veće od
b) Tačno i netačno
c) 2.11, 3.11, 4.11, ...
d) A, B, C, D
e) 1, 2, 3, 4, ...

4. Broj koji nastavlja niz 1, 1, 2, 3, 5, 8, 13, 21 je:

a) 42
b) 34
c) 31
d) 29
e) 22

5. Šta nije tačno?

a) 2/10=0.2
b) 0/5=0
c) 2×10-3=0.02
d) 2×102=200

6. Koji izraz je ekvivalentan izrazu a^2-b^2 ?

a) $a^2-2ab+b^2$
b) (a-b)+(a+b)
c) (a+b)(a-b)
d) (a+b)+(a+b)

7. Koji izraz je ekvivalentan izrazu $(a+b)^2$?

a) $a^2-2ab-b^2$
b) (a-b)+(a+b)
c) $a^2+2ab+b^2$
d) (a+b)+(a+b)

8. Koji izraz je ekvivalentan izrazu $(a-b)^2$?

a) $a^2-2ab-b$
b) (a-b)+(a+b)
c) $a^2-2ab+b^2$
d) (a+b)+(a+b)

9. Koja operacija nije dozvoljena?

a) 0×0
b) 0×1
c) 1/0
d) 0/1

10. Šta je tačno?

a) $2^8=128$
b) $2^8=265$
c) $2^8=256$
d) $2^8=116$

11. Binarni zapis broja 3 je:

a) 0111
b) 0011
c) 0300
d) 1001
e) 0001
f) 0003

12. Binarni zapis broja 2 je:

a) 0011
b) 0010
c) 0020
d) 1001
e) 0002

13. Binarni zapis broja 4 je:

a) 0101
b) 0100
c) 0020
d) 4001
e) 0004

14. Najmanja jedinica informacije koju računar može razumeti i obraditi je:

a) Cifra
b) Bajt
c) Bit
d) Kilobajt
e) Reč

15. Jedan bajt sadrži:

a) 4 bita
b) 8 bita
c) 10 bita
d) 12 bita
e) 16 bita

16. Obeležiti pogrešnu cifru u heksadecimalnom brojnom sistemu:

a) M
b) A
c) F
d) 7
e) 0

17. Obeležiti pogrešnu cifru u heksadecimalnom brojnom sistemu:

a) X
b) 1
c) D
d) E
e) A
f) 9

18. Broj različitih cifara u heksadecimalnom brojnom sistemu je:

a) 16
b) 6
c) 2
d) 10
e) 8
f) 4

19. Cifre u heksadecimalnom brojnom sistemu su:

a) 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
b) 0,1,00,11,000,111,0000,11111,X,Y,Z,A,B,C,D,E
c) 0,1,2,3,4,5,6,7,8,9,X,Y,Z,K,L,M
d) 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16
e) Q,W,E,R,T,Y,Z,1,2,3,4,5,6,7,8,9

20. Koji od navedenih iskaza je tačan?

a) 1Mb je 1024Gb
b) 1Mb je 1024Kb
c) 1Kb je 1024Mb
d) 1Gb je 1024Kb

21. Šta znači prefiks "mega"?

a) Stotinu
b) Hiljadu
c) Milion
d) Hiljaditi
e) Milioniti

22. Šta je tačno?

a) 32 bita < 3 bajta
b) 16 bita > 2 bajta
c) 8 bita < 2 bajta
d) 2 bajta < 15 bita

23. Obeležiti veličine u bajtima koje su u rastućem redosledu:

a) B, KB, MB, GB, TB, PB
b) PB, KB, B, GB, TB, MB
c) MB, PB, TB, KB, B, GB
d) B, MB, KB, PB, GB, TB
e) B, PB, MB, GB, TB, KB

24. Šta je to operativni sistem računara?

a) Program koji upravlja radom hardvera računara
b) Program za tabelarne proračune
c) Program za zaštitu od virusa
d) Program za rad sa bazama podataka
e) Program za rad sa operativnim podacima

25. Koji od navedenih programa NE označava operativni sistem računara?

a) Windows
b) MacOS
c) Excel
d) Linux

26. Koji od navedenih programa označava operativni sistem?

a) Linux
b) StarOffice 6.0
c) Lotus 123
d) PowerPoint 2010

27. Ulazni uređaj za računar je:

a) Monitor
b) Miš
c) Procesor
d) Memorija

28. Izlazni uređaj za računar je:

a) Monitor
b) Miš
c) Procesor
d) Tastatura

29. Koji je od navedenih uređaja izlazni uređaj?

a) Miš
b) Štampač
c) Mikrofon
d) Tastatura

30. Koji je od navedenih uređaja ulazni uređaj?

a) Tastatura
b) Ploter
c) Monitor
d) Štampač

31. Koji od navedenih uređaja je periferijski?

a) Ploter
b) RAM
c) EPROM
d) Matherboard

32. Obeležiti periferijski uređaj računara

a) Matična ploča
b) Procesor
c) Hladnjak
d) Monitor

33. Program Excel se primarno koristi za

a) Rad sa slikama
b) Rad sa tekstom
c) Rad sa bazom podataka
d) Rad sa tabelama

34. Računar koji je u lokalnoj mreži povezan sa serverom naziva se

a) Domen
b) Palm top
c) Notebook
d) Klijent

35. Računar koji omogućava deljenje svojih resursa drugim računarima u mreži naziva se:

a) Server
b) Klijent
c) Radna stanica
d) Domain

36. Koji program se koristi za kreiranje prezentacija?

a) Adobe PhotoShop
b) HTML
c) Word 2007
d) PowerPoint

37. Koji od pojmova označava periferijski uređaj računara?

a) CPU
b) CMOS baterija
c) Miš
d) Video memorija
e) Rek orman

38. RAM označava

a) Razne memorije
b) Neizbrisivu memoriju
c) Stalnu memoriju
d) Privremenu memoriju

39. ROM označava

a) Magistralu računara
b) Standardni konektor na računaru
c) Video memoriju
d) Memoriju iz koje se samo čita

40. RAM memorija je namenjena za:

a) Isključivo čitanje podataka
b) Isključivo pisanje podataka
c) Čuvanje operativnog sistema u odsustvu napajanja
d) I čitanje i pisanje podataka

41. CPU

a) Čuva podatke na računaru
b) Predstavlja ulazno/izlazni uređaj
c) Obezbeđuje napajanje za računar
d) Izvršava program računara

42. Koji od navedenih programa se koristi za tabelarna izračunavanja?

a) Word
b) Outlook
c) Norton
d) Adobe acrobat
e) Excel

43. Učitavanje operativnog sistema se naziva:

a) starting
b) cashing
c) upping
d) operating
e) booting
f) maping

44. Najnovija verzija Microsoft Windows-a je:

a) Windows 10
b) Windows XP
c) Windows 7
d) Windows NT

45. Operativni sistem je:

a) Microsoft Office
b) Windows 10
c) Word
d) CorelDRAW

46. Operativni sistem je:

a) SMS
b) GPS
c) MMS
d) UNIX

47. UPS je

a) Operativni sistem računara
b) Uređaj za neprekidno napajanje računara
c) Vrsta štampača
d) Standardni interfejs na računaru

48. Kapacitet modernih hard diskova za PC meri se

a) Kilo-bajtima
b) Mega-bajtima
c) Kilo Herzima
d) Giga-bajtima

49. Formatiranjem hard diska

a) Brišu se samo željeni podaci i programi
b) Briše se radna memorija računara
c) Briše se ROM u računaru
d) Brišu se svi podaci i programi

50. Koja od navedenih oznaka nije programski jezik?

a) Java
b) C
c) C++
d) PDF

51. MS Word je namenjen za:

a) Tabelarne kalkulacije
b) Rad sa bazama podataka
c) Kreiranje prezentacija
d) Obradu teksta

52. MS Access je namenjen za:

a) Pristup programima na računaru
b) Rad sa bazama podataka
c) Rad sa grafikom
d) Antivirusnu zaštitu

53. Program za slanje e-mail poruka je:

a) Outlook
b) Windows Explorer
c) PowerPoint
d) Norton

54. Koju ekstenziju imaju fajlovi koji predstavljaju program za izvršavanje aplikacije?

a) EXE
b) DOC
c) IZV
d) PRG

55. Ekstenzija datoteke programa PowerPoint je:

a) *.ppt
b) *.ptt
c) *.doc
d) *.dot
e) *.dat

56. Šta sadrže fajlovi sa ekstenzijom TXT?

a) Sliku
b) Video
c) Grafiku
d) Tekst

57. Koja od navedenih ekstenzija se koristi za tekstualne datoteke?

a) *.com
b) *.exe
c) *.bmp
d) *.txt
e) *.jpg

58. Internet kôd za oznaku geografskog domena Srbija je:

a) rs
b) sr
c) se
d) yu

59. Koji program se koristi za rad sa PDF fajlovima?

a) WordPerfect
b) Netscape
c) Word
d) Adobe Acrobat

60. ZIP je ekstenzija za:

a) Dokumenta
b) Arhivu
c) Izvršne fajlove
d) Video materijale

61. RAR je ekstenzija za:

a) Dokumenta
b) Arhivu
c) Izvršne fajlove
d) Zastarele fajlove

62. Komandom UNDO se:

a) Vraća u prethodni folder
b) Deinstalira se softver
c) Briše se željeni fajl
d) Sortiraju se dokumenta u folderu
e) Poništava prethodna radnja

63. Termin *Source code* označava:

a) Izvršni kod programa
b) Izvorni kod programa
c) Mašinski kod programa
d) Drajver uređaja
e) Proizvođača softvera

64. Jedan MB (megabajt) iznosi:

a) 1000 KB
b) 1024 KB
c) 1000 B

65. Operativni sistem je:

a) kolekcija sistemskih programa koji omogućavaju efikasno korišćenje računarskog sistema
b) skup programa koji omogućava obradu slike i teksta
c) operativna grupa programa koja isključivo kontroliše rad računarskih komponenti

66. Program SETUP se koristi za:

a) Instaliranje programa
b) Osnovna podešavanja operativnog sistema
c) Brisanje nepotrebnih datoteka posle instalacije

67. U operativnom sistemu Windows, Recycle Bin se koristi za:

a) Čuvanje podataka posle brisanja
b) Čuvanje oštećenih podataka
c) Uništavanje nepotrebnih podataka

68. Sa gledišta autorskih prava, na tržištu se mogu naći programi:

a) Privatni (PRIVATE) i poslovni (BUSINESS)
b) **Javni (Public domain) i deljeni (Shereware) i vlasnički (Property)**
c) Svi programi pripadaju pod ista autorska prava

69. Editori su:

a) Najednostavnija vrsta programa za rad sa tekstom
b) Jednostavna vrsta programa za editovanje slike
c) Moćniji programi za rad sa tekstom

70. Deo teksta se može premestiti iz jednog dokumenta u drugi instrukcijama:

a) COPY, PASTE
b) CUT, MOVE
c) CUT, PASTE

71. Kompajler je program za:

a) Prevođenje programa u mašinskom jeziku na izvršni program
b) Prevođenje izvornog programa na mašinski jezik
c) Program za prevođenje programa u željeni viši programski jezik

72. Ako je vrednost promenljive b1=FALSE i b2=TRUE, vrednost izraza b1 AND b2 i b1 OR b2 respektivno su:

a) FALSE i FALSE
b) TRUE, FALSE
c) FALSE, TRUE

73. Programski jezik:

a) Služi za upravljanje svim resursima računara
b) Služi za upravljanje velikim bazama podataka
c) Omogućava pisanje aplikativnog programa

74. Glavne komponente računarskog sistema su:

a) Hardver i softver
b) Hardver, komunikaciona mreža i sistemski softver
c) Hardver i aplikacioni softver

75. U operativnu memoriju se smeštaju:

a) Svi sistemski i aplikativni programi koji se nalaze u računarskom sistemu
b) Svi programi i podaci koji se u tom trenutku obrađuju u računaru
c) Svi podaci koji pripadaju informacionom podsistemu na kome se radi

76. Aplikacioni softver služi da:

a) Upravlja radom resursa računara
b) Rasporedi programe i podatke na tačno utvrđena mesta na spoljnoj disk memoriji računara
c) Automatizuje poslovne procese korisnika

77. Najmanja, elementara memorijska jedinica je:

a) bit
b) bajt
c) kilobajt

78. Skener je uređaj koji omogućava:

a) unošenje slike ili crteža u računar u digitalnom obliku
b) analizu čovekovog tela
c) prikaz slike

79. Jedinice govornog ulaza su:

a) mikrofoni
b) pojačivači govora
c) čitači glasa sa magnetnim zapisom

80. Operativni sistem:

a) služi za operativno funkcionisanje bankarskog sistema
b) nadzire operativni rad bankarskih službenika
c) upravlja radom računarskog sistema koji ga čini funkcionalnom celinom

81. BIOS je:

a) softver koji upravljanje isključivo radom memorije
b) softver koji upravljanje poslovnim procesima
c) softverski modul koji upravlja i kontroliše rad ulazno – izlaznih jedinica, smešten u posebnom čipu ROM memorije

82. Koji od sledećih programskih paketa se koriste za rad sa bazama podataka:

a) MS Access
b) Open Office
c) oba gore navedena

83. U mrežnom okruženju skraćenica LAN predstavlja:

a) Logical Application
b) Local Area Network
c) Local Access Number

84. Modem je:

a) Uređaj za kriptografiju podataka
b) Ni jedno od ove dve definicije
c) Uređaj za povezivanje na mrežu

85. Neki od servisi Interneta su:

a) Paralelni rad
b) Elektronska pošta
c) Sigurnosna pošta

86. Serveri su:

a) Više povezanih računara u mreži
b) Računari koji opslužuju umrežene korisnike
c) Računari velikog kapaciteta

87. Koja je funkcija tekst procesora:

a) Pisanje, modifikovanje, čuvanje i štampa dokumenta
b) Priprema grafičkih prezentacija
c) Obrada tabela

88. Šta je font:

a) Memorijska jedinica
b) Skup znakova koji imaju iste vizuelne karakteristike
c) Deo računarske jedinice

89. U okviru "kretanja" na www koriste se programi za navigaciju – browser, kao što su

a) Internet Explorer
b) MS Office
c) CHTTP

90. Bajt je

a) Deo baze podataka
b) Osnovna jedinica za izražavanje kapaciteta memorije računara
c) Komunikacioni deo softvera

91. Kursor definiše:

a) Tekuću poziciju na ekranu

b) Tekuću poziciju podataka na magnetnom medijumu

c) Obe navedene stvari

92. SQL je

a) Programski jezik za komunikaciju sa relacionim bazama podataka

b) Baza podataka

c) Sistemski softver

93. Akronim Linux označava

a) Programski jezik

b) Aplikativno rešenje

c) Operativni sistem

94. Delphi je:

a) Aplikativno rešenje

b) Programski jezik četvrte generacije

c) Program za obradu teksta

95. Šta predstavlja Blu-Ray:

a) magnetno-optički medijum velikog kapaciteta

b) magnetni medijum velikog kapaciteta

c) optičko-magnetni medijum velikog kapaciteta

d) optički medijum velikog kapaciteta

96. Koji je pravilan redosled uređaja u odnosu na brzinu prenosa podataka počev od najmanje?

a) RAM, disketa, hard disk, CD-ROM

b) Hard disk, RAM, Flash, disketa

c) Disketa, Flash, RAM, Hard disk

d) CD-ROM, Hard disk, RAM

97. Prema osnovnoj klasifikaciji softver može biti:

a) sistemski i aplikativni

b) sistemski, aplikativni i drajveri

c) sistemski, aplikativni, operativni sistemi i drajveri

d) softver za obradu teksta, tabela, grafike, muzike itd…

98. Šta od navedenog je računar koji obezbeđuje resurse drugim računarima u mreži?

a) klijent

b) server

c) korsinik

d) provajder

99. Koja od navedenih karakteristika opisuju RAM?

a) memorija samo za pisanje
b) nije moguće upisivanje podataka
c) memorija samo za čitanje
d) privremeno čuva podatke

100. Koja se od navedenih karakteristika odnosi na ROM

a) izlazno/ ulazna komponenta
b) privremeno čuva podatke
c) nije moguće upisivanje podataka

101. Koja od navedenih karakteristika opisuju RAM?

a) nije moguće upisivanje podataka
b) nije moguće čitanje podataka
c) podaci mogu da se menjaju
d) podaci NE mogu da se menjaju

102. Skraćenica CPU označava?

a) Calculating Process Unit
b) Control Program Unit
c) Control Process Unit
d) Central Processing Unit

103. Šta od navedenog izvršava BIOS?

a) testiranje hardvera računara
b) kompajliranje izvornog koda
c) rad sa fajlovima
d) BIOS sadrži operativni sisetem

104. Jedna od navedenih reči ne može biti ime datoteke pod operativnim sistemom Windows:

a) File[1]
b) Pismo<1>
c) T@bela
d) {lika

105. Razlika brojeva AB – AA u heksadecimalnom brojnom sistemu iznosi

a) A
b) B
c) 1
d) 0

106. Između brojeva 1101 i 110 koji su zapisani u binarnom brojnom sistemu, razlika u binarnom brojnom sistemu iznosi:

a) 110
b) 100
c) 111
d) 101

107. Najveći neoznačen ceo pozitivan broj koji može da se registruje u računaru korišćenjem jednog bajta je:

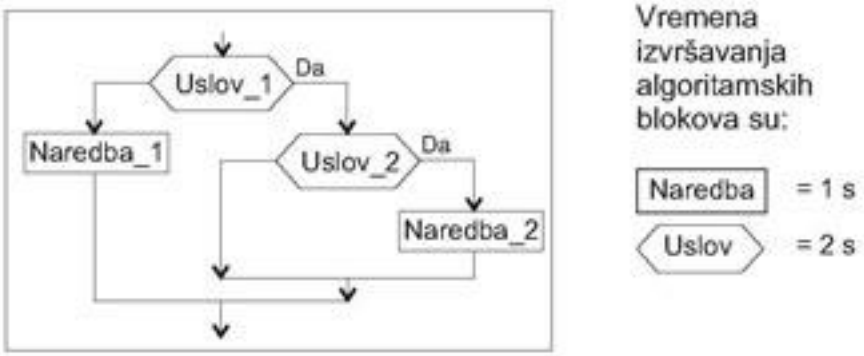
a) 127
b) 255
c) 256
d) 128

108. Jedan od navedenih programa se ne odnosi na program za pretraživanje na Internetu (web browser):

a) Opera
b) Google Chrome
c) Drama
d) Mozilla

109. Za deo algoritma koji je dat na slici, odrediti:

a) Minimalno vreme izvršavanja T_{min} __2_____
b) Maksimalno vreme izvršavanje T_{max} _5_____



---

110. Koja je najmanja jedinica za količinu informacija?

a) Bit
b) Byte
c) Word

111. Jedan bajt čini:

a) 1 bit
b) 8 bita
c) 64 bita

112. Koliko bajtova sadrži 1 kilobajt?

a) 100
b) 1000
c) 1024

113. Softver računara je:

a) Njegov elektronski sklop
b) Skup svih podataka u njemu
c) Skup svih programa u njemu

114. Hardveru računara pripadaju:

a) Operativni sistem i aplikativni programi
b) Centralna (operativna) memorija i operativni sistem
c) Centralna procesorska jedinica i centralna (operativna) memorija

115. Softveru računara pripadaju:

a) Operativni sistem i aplikativni programi
b) Centralna (operativna) memorija i operativni sistem
c) Centralna procesorska jedinica i centralna (operativna) memorija

116. U operativnoj (centralnoj) memoriji računara podaci se čuvaju:

a) U binarnom obliku (pomoću cifara 0 i 1)
b) U decimalnom obliku (pomoću cifara od 0 do 9)
c) U obliku koji zavisi od operativnogsistema

117. Operacije nad podacima u računaru realizuju se:

a) U binarnom obliku
b) U decimalnom obliku
c) U obliku koji zavisi od operativnog sistema

118. Koja od navedenih komponenata je neophodna da bi računar radio?

a) CD-ROM
b) Procesor
c) Mrežna kartica

119. Šta od navedenog predstavlja hardversku komponentu računara?

a) Procesor
b) Internet browser
c) Operativni sistem

120. Koja se od navedenih komponenti nalazi na osnovnoj ploči računara?

a) Procesor
b) Tastatura
c) Harddisk

121. Koji iskazi su tačni za procesor?

a) Izvršava aritmetičko-logičke operacije
b) Brzina rada procesora ne utiče na brzinu računara
c) Sadrži programe

122. Šta može značajno da ubrza rad računara?

a) Novi monitor visoke rezolucije
b) Brži procesor
c) Brži pristupInternetu

123. Jedinica koja ukazuje na brzinu rada procesora je:

a) Gigabajt (GB)
b) Gigaherc (GHz)
c) Vat (W)

124. RAM je skraćenica od:

a) Random Access Memory
b) Radio Access Memory
c) Real Access Memory
d) Ne znam

125. ROM je skraćenica od:

a) Random Only Memory
b) Read Only Memory
c) Read On-line Memory

126. Koja se od navedenih oznaka odnosi na hard disk?

a) HDD
b) FDD
c) CD

127. Ukupna količina podatakakoja može da se čuva na DVD-u izražava se u:

a) KB
b) MB
c) GB

128. Računar sa jednim procesorom može da izvrši:

a) Samo jednu instrukciju u jednom trenutku vremena
b) Više instrukcija u jednom trenutku vremena
c) Broj instrukcija koji zavisi od operativnog sistema

129. Hard disk u računaru predstavlja njegovu:

a) Operativnu (centralnu) memoriju
b) Spoljnu memoriju
c) Operativnu (centralnu) ili spoljnu memoriju, zavisi od računara

130. RAM u računaru predstavlja njegovu:

a) Operativnu (centralnu) memoriju
b) Spoljnu memoriju
c) Masovnu memoriju

131. DVD je skraćenica od:

a) Disk Video Drive
b) Digital Veb Disk
c) Digital Video Disk

132. Štampač u računarskom sistemu predstavlja njegovu:

a) Operativnu (centralnu) memoriju
b) Spoljnu memoriju
c) Izlazni uređaj

133. Šta predstavlja DVD?

a) Magnetni medijum velikog kapaciteta
b) Optičko-magnetni medijum velikog kapaciteta
c) Optički medijum velikog kapaciteta

134. Koji od nabrojanih uređaja ima najveći kapacitet?

a) DVD
b) CD
c) CD-RW

135. Po uključenju personalnog računara, instalirani operativni sistem:

a) Ne učitava se u operativnu (centralnu) memoriju
b) Dobija svoje mesto u operativnoj (centralnoj) memoriji
c) Učitavase u operativnu (centralnu) memoriju ili ne, u zavisnosti od operativnog sistema

136. BIOS je skraćenica za:

a) Black Inline System
b) Basic Input Output System
c) Binary Input Output System

137. Koji je osnovni zadatak BIOS-a?

a) Testiranje hardvera računara i pokretanje operativnog sistema
b) Rad sa datotekama
c) BIOS sadrži operativni sistem

138. Koji je od navedenih iskaza tačan?

a) Kada je računar isključen, operativni sistem se nalazi u RAM-u
b) Operativni sistem je neophodan za rad sa korisničkim programima i datotekama
c) Operativni sistem nije neophodan za rad sa računarom

139. Linux je:

a) Operativni sistem
b) Komponenta Windows operativnihsistema
c) Mašinski programski jezik

140. Operativnim sistemima pripadaju:

a) UNIX, Linux, Mac OS, Acrobat Reader
b) Linux, AMD, Windows
c) Mac OS, DOS, OS/2

141. C++ je:

a) Komponenta Windows operativnih sistema
b) Objektno-orijentisani programski jezik
c) Mašinski programski jezik

142. Program u računaru predstavlja:

a) Skup datoteka
b) Skup direktorijuma
c) Skup instrukcija

143. Faze razvoja programa realizuju se ovim redosledom:

a) Prevođenje koda, projektovanje i izrada koda, testiranje rada programa
b) Projektovanje i izrada koda, prevođenje koda, testiranje rada programa
c) Prevođenje koda, testiranje rada programa, projektovanje i izrada koda

144. Algoritam je:

a) Programski jezik
b) Skup programa
c) Opis toka programa

145. Algoritam programa može biti predstavljen:

a) Samo tekstualno
b) Samo grafički
c) Tekstualno ili grafički

146. Programski jezik predstavlja:

a) Samo skup simbola za izradu programa na tom jeziku
b) Samo skup pravila koja se koriste pri izradi programa na tom jeziku
c) Skup simbola i pravila za korišćenje simbola pri izradi programa na tom programskom jeziku

147. Mašinski jezik predstavlja jezik od:

a) cifara 0 i 1
b) cifara od 0 do 7
c) cifara od 0 do 9

148. Računar razume

a) Samo mašinski jezik
b) Samo objektno-orijentisane programske jezike
c) Bilo koji programski jezik

149. Programskim jezicima pripadaju:

a) Windows Explorer, Net Meeting, Photo Shop
b) Pascal, Visual Basic, C++
c) Control Panel, Excel, Word

150. Word je:

a) Pretraživač
b) Aplikativni program
c) Operativni sistem

151. Excel je:

a) Program za formatiranje diska
b) Aplikativni program
c) Operativni sistem

152. Paint je:

a) Programski jezik
b) Aplikativni program
c) Operativnisistem

153. Pascal je:

a) Programski jezik
b) Aplikativni program
c) Operativni sistem

154. Visual Basic je:

a) Programski jezik
b) Aplikativni program
c) Operativni sistem

155. Java je:

a) Programski jezik
b) Aplikativni program
c) Operativnisistem

156. PHP je:

a) Programski jezik
b) Aplikativni program
c) Operativni sistem

157. Aplikacije Windows operativnog sistema mogu biti pozvane:

a) Samo iz menija Start izborom opcije All Programs
b) Samo preko odgovarajuće ikone aplikacije
c) Iz menija Start izborom opcije All Programs ili preko odgovarajuće ikone aplikacije

158. Za konfigurisanje računara mogu se koristiti opcije Windows Start menija:

a) Paint
b) Search
c) Settings

159. Koji program služi za obradu teksta?

a) Corel Draw
b) Auto Cad
c) Word

160. Koji program služi za obradu slike?

a) Corel Draw
b) Word
c) Excel

161. Toolbar u Windows aplikacijama predstavlja:

a) Samo paletu boja
b) Samo paletu fontova
c) Paletu alata

162. Izbor standardnih Windows komandi File i Close:

a) Zatvara aktivan dokument aplikacije, ali ne i aplikaciju
b) Zatvara sve otvorene dokumente aplikacije, ali ne i aplikaciju
c) Zatvara otvorenu aplikaciju

163. Izbor standardnih Windows komandi File i Exit:

a) Zatvara aktivan dokument aplikacije, ali ne i aplikaciju
b) Zatvara sve otvorene dokumente aplikacije, ali ne i aplikaciju
c) Zatvara otvorenu aplikaciju

164. Izabrati tačan iskaz:

a) Word dokumenta imaju ekstenziju .pdf
b) Word dokumenta imaju ekstenziju .docx
c) Word dokumenta imaju ekstenziju .cdr

165. Izabrati tačan iskaz:

a) Adobe Acrobat dokumenta imaju ekstenziju .pdf
b) Adobe Acrobat dokumenta imaju ekstenziju .doc
c) Adobe Acrobat dokumenta imaju ekstenziju .cdr

166. Kombinacija tastera Ctrl+X koristi se u Windows aplikacijama za komandu:

a) Cut
b) Copy
c) Paste

167. Kombinacija tastera Ctrl+C koristi se u Windows aplikacijama za komandu:

a) Cut
b) Copy
c) Paste

168. Kombinacija tastera Ctrl+V koristi se u Windows aplikacijama za komandu:

a) Cut
b) Copy
c) Paste

169. Izborom tastera BackSpace:

a) Briše se jedan karakter sa leve strane kursora
b) Briše se jedan karakter sa desne strane kursora
c) Brišu se svi karakteri sa desne strane kursora, do kraja reda

170. Izborom tastera Delete:

a) Briše se jedan karakter sa leve strane kursora
b) Briše se jedan karakter sa desne strane kursora
c) Brišu se svi karakteri sa desne strane kursora, do kraja reda

171. Izborom tastera PrintScreen, sadržaj ekrana:

a) Briše se
b) Postaje nevidljiv
c) Kopira se na Clipboard

172. U jednu računarsku mrežu mogu biti povezaniračunari:

a) Samo ako imaju zajednički operativni sistem
b) Samo ako imaju međusobno isti operativnisistem
c) Bez obzira na operativnesisteme, pomoću zajedničkog protokola

173. Modem je:

a) Elektronski uređaj koji omogućava pristup Internetu
b) Operativni sistem koji omogućava pristup Internetu
c) Aplikacija koja omogućava pristup Internetu

174. LAN je skraćenica za:

a) Local Auto Network
b) Local Area Network
c) Land Area Neighbour

175. WAN je skraćenica za:

a) Wide Area Network
b) Web Area Network
c) Web Area Neighbour

176. Programi koji se koriste za pristup internetu su:

a) MS Word, Internet Explorer
b) Pow er Point, Corel Draw
c) Mozilla Firefox, Internet Explorer

177. Google je:

a) Internet usluga koja omogućava pretraživanje Web dokumenata
b) Web enciklopedija (skup Web dokumenata)
c) Internet društvena mreža

178. Koliko najmanje (umreženih) računara čini računarsku mrežu?

a) Dva
b) Pet
c) Više od petnaest

179. IP je skraćenica od:

a) Internet Protokol
b) Intelligent Protocol
c) Internet Path

180. HTML je skraćenica od:

a) Hyper Text Markup Language
b) Hyper Text Multi Language
c) Hyper Text Multimedia Language

181. Šta se koristi za identifikaciju računara na Internetu?

a) IP adresa
b) Korisničko ime
c) Ime firme

182. WWW je skraćenica od:

a) World Wide Web
b) Wireless Wide Web
c) Word Wire Web

183. FTP je skraćenica od:

a) File Transfer Protocol
b) File Transfer Priority
c) File Trapping Protocol

184. U slučaju URL-a *ime_prezime@viser.edu.rs* sadržaj ispred znaka @, predstavlja:

a) Korisničko ime u e-mail adresi
b) Ime računara
c) Ime servera

185. Koji od navedenih naziva nije naziv Internet pretraživača?

a) Google
b) Front Page
c) Yahoo

Primeri zadataka

1. Dato je 5 različitih prirodnih brojeva. Odrediti srednji po veličini. Na primer, za unete brojeve 2, 7, 3, 4, 5 – odgovor je 4.

2. Napisati program koji računa n-ti član Fibonačijevog niza, koji je definisan sa F [0] = 0, F [1] = 1 i F [n+1] = F [n] + F [n – 1]. Na primer, za n = 6 – odgovor je 8.

3. Napisati program koji broji pojavljivanja samoglasnika (A, E, I, O, U) u tekstu koji se unosi sa standardnog ulaza. Na primer, za tekst 'Prirodno matematicki FAKULTET' – odgovor je 11.

4. Napisati program koji štampa prvih n Nivenovih brojeva (oni brojevi koji su deljivi sumom svojih cifara). Na primer, za n = 15, štampati 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 18, 20, 21, 24.

5. Data je kvadratna matrica a dimenzije n x n. Odrediti proizvod elemenata ispod glavne dijagonale. Glavnu dijagonalu čine elementi sa koordinatama (1, 1), (2, 2), ..., (n, n). Na primer za matricu a = ((1, 2, 3), (4, 5, 6), (7, 8, 9)) – odgovor je 224.

6. Odrediti vrednost izraza e^x preko obrasca $e^x = 1 + x + x^2/2! + x^3/3! + \ldots + x^n/n! + \ldots$ dodavanjem sabiraka koji su veći ili jednaki od 0.00001. Na primer za x = 1, štampati e = 2.71828 na pet decimala.

7. Dat je polinom $P(x) = a [n] * x^n + a [n – 1] * x^{n-1} + \ldots + a [1] * x + a [0]$ sa realnim koeficijentima a [0], a [1], ..., a [n]. Odrediti vrednost polinoma u tački x. Na primer, za polinom $P (x) = x^3 + 2x – 1$ stepena 3 sa koeficijentima (-1, 2, 0, 1) i x = 2 – odgovor je 11.

8. Za dati broj x formiramo niz x, p(x), p(p(x)), … gde je p(x) proizvod cifara broja x. Ispisati sve brojeve u nizu do pojave prvog jednocifrenog broja. Na primer, za n = 199 ispisati 199, 81, 8.

9. Data su dva stringa a i b. Ispitati da li se string a može dobiti izbacivanjem nekih karaktera stringa b. Na primer, za a = 'imati' i b = 'informatika' – odgovor je 'DA'.

10. Napisati program koji za dati niz ocena, izbacuje najveću i najmanju ocenu i računa prosek preostalih ocena. Na primer, za niz ocena 2, 3, 5, 3, 2, 2, odstranjujemo ocene 5 i 2 – pa je odgovor (2 + 2 + 3 + 3) / 4 = 2.5.

11. Za datu Cezarovu šifru, kodirati string sastavljen od velikih slova engleske abecede. Kod ovog šifriranja, svako slovo se menja odgovarajućim slovom iz drugog reda. Na primer, za reč 'MATEMATIKA' štampati 'PDWHPDWLND'.
    Original: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
    Cezar: D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

12. Data su dva niza a i b dužina n i m, koja su sortirana u rastućem redosledu. Što efikasnije novi niz c, koji predstavlja uniju ova dva niza, sortiranu takođe u rastoćem redosledu. Na primer, za a = (1, 3, 5, 6) i b = (2, 4, 7, 8, 9) – odgovor je c = (1, 2, 3, 4, 5, 6, 7, 8, 9).

13. Ispred blagajne je poređano n ljudi sa svojim visinama h [1], h [2], ..., h [n] u tom redosledu. Odrediti broj ljudi koje vidi blagajnik sa početka reda. Na primer, za ljude sa visinama 155, 170, 165, 180, 175, 195, blagajnik vidi prvog, drugog, četvrtog i šestog čoveka – odgovor je 4.

14. Dat je niz n prirodnih brojeva. Odrediti dužinu najdužeg podniza uzastopnih brojeva koji je rastući. Na primer za niz 5, 1, 3, 2, 5, 8, 9, 4 najduži rastući uzastopni podniz je 2, 5, 8, 9 – pa je odgovor 4.

15. Dato je n dasaka sa svojim dužinama. Potrebno je odrediti najveću moguću površinu trougla koji se može napraviti pomoću neke tri daske. Heronov obrazac za površinu trougla sa stranicama a, b i c glasi P = sqrt (s * (s – a) * (s – b) * (s – c)), gde je s = (a + b + c) / 2. Na primer, za daske sa dužinama 1, 3, 4, 5, 10, najveći trougao koji možemo napraviti je trougao sa stranicama 3, 4 i 5 – pa je odgovor 6.

16. Data su dva stringa a i b. Odrediti da li je string a anagram stringa b (da li se string b dobija premeštanjem slova stringa a). Na primer, za a = 'matematika' i b = 'temamakati' odgovor je 'DA', dok je za a = 'matematika' i b = 'matematike' odgovor 'NE'.

17. Dato je n ponuda za punta u dinarima. Vaš zadatak je da odredite minimalnu jedinstvenu ponudu (ukoliko postoji). Na primer, za ponude 5, 4, 1, 2, 1, 2, 6, 2 – odgovor je 4 (pošto se ponude 1 i 2 javljaju više puta).

18. Dat je prirodan broj n. Odrediti da li je broj n jednak proizvodu tačno dva prosta broja. Na primer, za n = 85 – odgovor je 'DA'.

19. Odrediti najveći zajednički delilac i najveći zajednički sadržalac za dva broja a i b. Na primer, za brojeve a=20 i b = 15 – odgovor je nzd (a, b) = 5 i nzs (a, b) = 60.

20. Za date brojeve a i b, odrediti vrednost a/b na 1000 decimala. Na primer, za a = 80 i b = 21 – odgovor je 3.4782608695652173913…

21. Za dva cela broja, odrediti njihovo Hemingovo rastojanje (broj bitova na kojima se brojevi razlikuju). Na primer, za brojeve 93 = 01011101 i 75 = 01001011 Hemingovo rastojanje je 3.

22. Data su dva pravougaonika u ravni sa svojim donjim levim i gornjim desnim temenom. Odrediti da li se pravougaonici seku. Na primer, za pravougaonike sa koordinatama (0, 0) i (10, 10), i pravougaonik sa koordinatama (5, 5) i (15, 8) – odgovor je "DA".

23. Odrediti vrednost broja m posle izvršavanja sledećeg koda

```
int n = 123456789;
int m = 0;
while (n != 0) {
  m = (10 * m) + (n % 10);
  n = n / 10;
}
```

```
n := 123456789;
m := 0;
while (n <> 0) do begin
  m := (10 * m) + (n mod 10);
  n := n div 10;
end;
```

24. Šta se dobija izvršavanjem sledećeg koda

```
int f = 0;
int g = 1;
for (int i = 0; i <= 15; i++) {
  cout << f;
  f = f + g;
  g = f - g;
}
```

```
f := 0;
g := 1;
for i := 0 to 15 do begin
  WriteLn (f);
  f := f + g;
  g := f - g;
end;
```

25. Šta se dobija izvršavanjem sledećeg programa

```
for (int i = 0; i < 10; i++)
  a[i] = 9 - i;
for (int i = 0; i < 10; i++)
  a[i] = a[a[i]];
for (int i = 0; i < 10; i++)
  cout << a[i];
```

```
for i := 1 to 10 do
  a [i] := 10 - i;
for i := 1 to 10 do
  a [i] := a [a [i]];
for i := 1 to 10 do
  WriteLn (a [i]);
```

26. Odrediti vrednost promenljive x posle izvršavanja sledećeg koda u zavisnosti od N:

```
int x = 0;
for (int i = 0; i < N; i++)
  for (int j = i + 1; j < N; j++)
    for (int k = j + 1; k < N; k++) {
      x++;
          }
```

```
x := 0;
for i := 1 to N do
  for j := i + 1 to N do
    for k := j + 1 to N do begin
      x := x + 1;
    end
```

27. Odrediti vrednost promenljive s, kada se sledeći kod startuje za N=3 i N=4.

```
s = "";
for (int i = 1; i <= N; i++) {
  itoa (i, t, 10);
  if (i % 2 == 0)
    s = s + i + s;
  else
    s = i + s + i;
}
```

```
s := '';
for i := 1 to n do begin
  Str (i, t);
  if (i mod 2 = 0) then
    s := s + t + s
  else
    s := t + s + t;
end;
```

28. Odrediti šta se dešava sa promenljivama a i b posle poziva procedura

29. Izračunati minimum tri učitana realna broja a, b, c.

30. Urediti tri zadata realna broja u poredak x < y < z.

31. Napisati program kojim se dati brojevi x, y, z udvostručuju ako je x>= y>= z, a u protivnom menjaju znak.

32. Napisati program kojim se simulira digitron: učitavaju se dva operanda i aritmetički operator. Izračunati vrednost unetog izraza.

33. Izračunati

$$y = \begin{cases} -5, & x<0, \\ x+2, & 0<=x<1, \\ 3x-1, & 1<=x<5, \\ 2x, & x>=5. \end{cases}$$

34. Napisati program koji izračunava stepen realnog broja X na stepen n, gde je n nenegativni ceo broj.

35. Izračunati parcijalnu sumu H(N) = 1+1/2 +1/3 + ... + 1/N harmonijskog reda.

36. Data su tri realna broja u poretku x < y < z. Umetnuti realni broj t tako da među njima bude odnos x < y < z < t.

37. Diskutovati rešenje sistema linearnih jednačina
$a_1 x + b_1 y = c_1$
$a_2 x + b_2 y = c_2$.

38. Rešiti jednačinu $ax^2+bx+c=0$.

39. Izračunati
1+1/2 + 1/3 +…
gde se sumiranje prekida kada vrednost 1/*i* bude manja od zadate tačnosti *G*.

40. Ispisati ceo broj s desna na levo.

41. Izračunati približno vrednost broja π = 3.14159 koristeći formulu $\pi/4$ = 1-1/3+1/5-1/7+… Sumiranje prekinuti kada apsolutna vrednost člana koji se dodaje bude manja od zadate vrednosti *eps*.

42. Izračunati faktorijel prirodnog broja n.

43. Šta se ispisuje izvršenjem sledećeg programa?

```
program Parametri;
var A, B :Integer ;
procedure Dodaj1(X : integer; var Y : integer);
  begin
   X := X+1; Y := Y+1; Writeln(X,’ ‘, Y)
end { Dodaj1 } ;
begin
A := 0; B := 0; Dodaj1(A,B); Writeln(A,’ ‘, B)
end { Parametri }.
```

Rezultat:
1 1
0 1

44. Šta se ispisuje izvršenjem sledećeg programa?

```
program LocalGlobal;
var  global: integer;
procedure PPG(var local: integer);
  begin
local := local+1;
writeln('global = ',global);
local := local+global;
  end;
begin
  global := 2;  PPG(global); writeln('global = ', global);
end.
```

Rezultat:

```
  global = 2
  global = 2
```