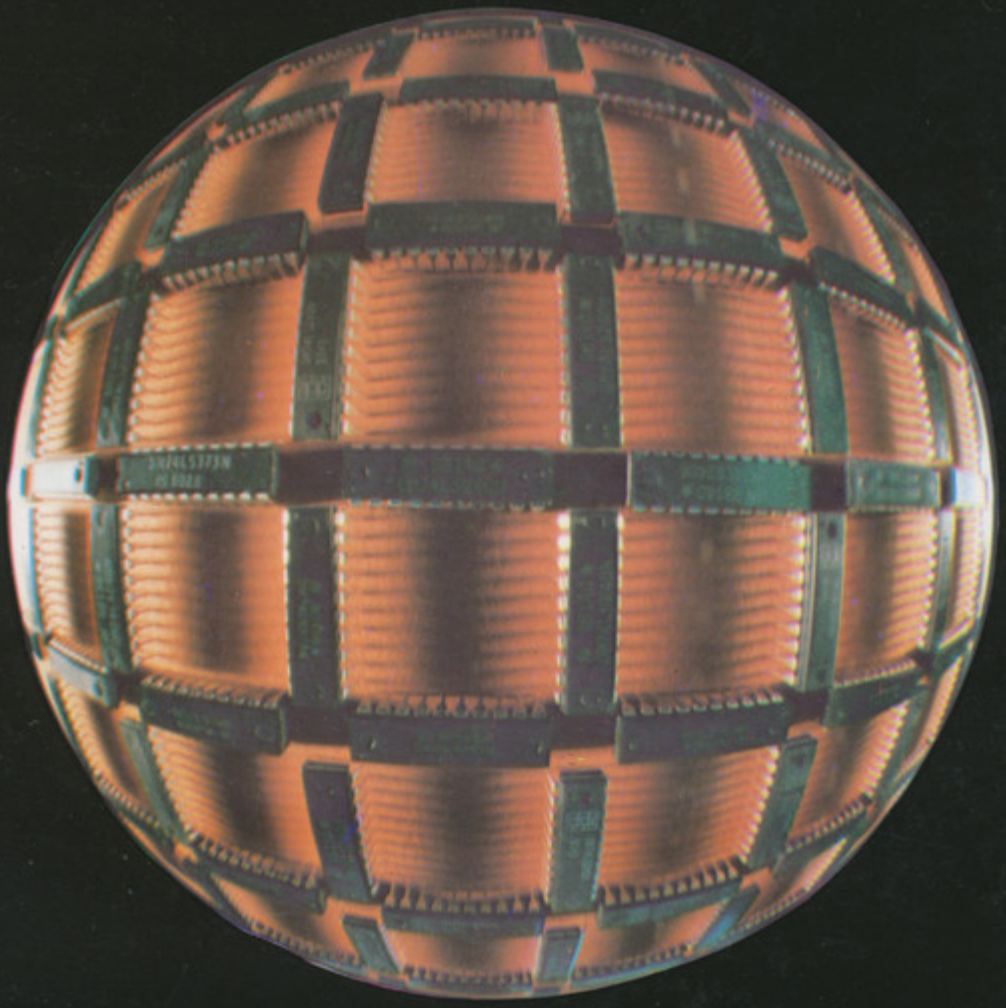
ABC RAČUNALNIŠTVA PARTNER HR-84

Ivan Gerlič



računalniški sistemi delta

ABC RAČUNALNIŠTVA PARTNER HR-84

Ivan Gerlič

Iskra

ISKRA DELTA IN ZOTKS

Ljubljana 1984

KNJIŽNA ZBIRKA DELTA

Peta knjiga

PRI NASTANKU KNJIŽICE SO SODELOVALI:

avtor: Ivan Gerlič, mag.
avtor 8. poglavja: Vasja Herbst, ing.
lektor: Damjana Simončič, prof.
ilustracije: Duša Škulj

UVOD

Drugi, dopolnjeni izdaji na pot!

Pred vami je druga, dopolnjena izdaja knjižice ABC računalništva. Z razumevanjem in sodelovanjem sodelavcev iz Iskre je „napisana na kožo" računalnikoma Partner in HR 84. Avtor in lektor sta si prizadevala odstraniti pomanjkljivosti, ki so nastale v naglici, s katero se je porajala prva izdaja.

S prizadevanjem in razumevanjem sodelavcev iz Iskre-Delte pa je nastalo še nekaj drugega: knjižica je v tej obliki dosegljiva večjemu številu bralcev, namenjena je tudi poslovnežem, in v Knjižni zbirki Delta nadaljuje vrsto izvirnih, domačih, avtorskih del na področju računalniške literature.

Knjižna izdaja te vrste je tudi dokaz, da je še mnogo možnosti za sodelovanje med družbenimi organizacijami in organizacijami združenega dela.

Gorazd Marinček
Zveza organizacij za
tehnično kulturo

1. UVODNA RAZMIŠLJANJA IN OSNOVNI POJMI

Od pojava prvega elektronskega računalnika pa do danes je preteklo komaj četrt stoletja, a ne glede na to je svet računalnika prinesel niz pomembnih novosti in sprememb v sodobno družbo. Vsak mesec nam prinese poštar televizijske in telefonske račune, bančne izpiske, itd., ki jih vodijo in izdelujejo računalniki v centrih za obdelavo podatkov določenih ustanov. Danes na tisoče hranilcev vpisuje obračunane obresti takoj po novoletnih praznikih, nekoč to ni bilo mogoče pred sredino februarja. Nobena novost ni več, če v neki delovni organizaciji računalnik v nekaj urah pripravi plačilni spisek za nekaj tisoč zaposlenih; milijoni televizijskih, telefonskih računov ali računov elektrodistribucije so pripravljeni v nekaj deset urah. Torej smo že v vsakodnevnem življenju obkroženi z nizom pojavov, ki so rezultat dela računalnika:

a) telefonski, televizijski računi, pokojninski odreziki, davčni obračuni, itd.;

b) računalniki kontrolirajo plačila in avtomatsko pošiljajo opomine;

c) v industriji računalniki izdajajo delovne naloge, nadzorujejo proizvodni proces in kvaliteto;

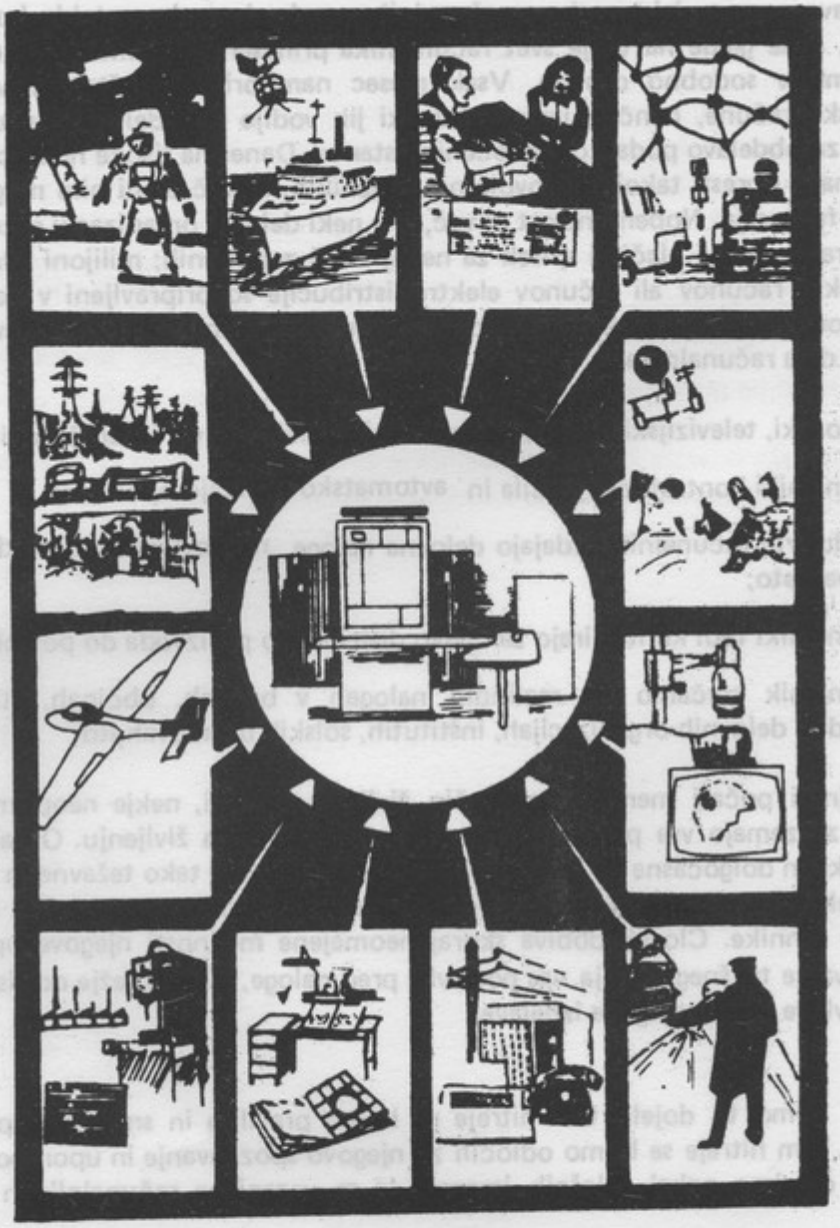
d) računalniki tudi kontrolirajo zaloge in distribucijo proizvoda do potrošnika;

e) računalnik srečamo na različnih nalogah v bankah, občinah, statističnih zavodih, delovnih organizacijah, inštitutih, šolskih ustanovah, itd.

Računalniki počasi menjajo naš način življenja. Počasi, nekje neopazno drugje opazno, zavzemajo vse pomembnejše mesto v sodobnem življenju. Opravljajo vse tiste težke in dolgočasne rutinske naloge, ki so včasih bile tako težavne in zamudne za človeka. Za računalnik lahko rečemo, da je eden najpomembnejših dosežkov sodobne tehnike. Človek dobiva skoraj neomejene možnosti njegove uporabe in obvladovanje takšnega stroja nas postavlja pred naloge, mnogo težje od tistih, ki so se postavljale v teku njegove izdelave.

Čimprej bomo to dojeli, tem hitreje ga bomo pravilno in smiselno uporabljali, oziroma, tem hitreje se bomo odločili za njegovo spoznavanje in uporabo. Pa si za začetek oglejmo nekaj splošnih izrazov, ki so vezani na računalnik in računalništvo.

V sodobni literaturi in v vsakodnevnih sredstvih javnega obveščanja (kot so časopisi, radio in televizija), se srečujemo s pojmi, kot so informacija, računalnik, avtomatska obdelava podatkov (informacij), kibernetika, itd. in obenem spoznavamo probleme, ki se v zvezi z njimi javljajo. Prvi od teh problemov je **informacija**. Beseda informacija izvira iz besede informatio (latinsko), ki pomeni pojem, predstavo, skupek spoznanj, sporočilo. Običajno pravimo, da sporočilo vsebuje informacijo, če nam o čem kaj novega pove. V sodobni družbi vsakodnevno raste število informacij, ki jih je potrebno sprejeti, obdelati in nato uporabiti. Zaradi tega tudi raste pomen računalnika kot generatorja obdelave informacij.



Slika 1: Nekatera področja uporabe računalnika

Vedo, ki se ukvarja z vsem, kar je v zvezi z obravnavo informacij oziroma podatkov, imenujemo **informatika**.

Za industrijsko revolucijo je bilo značilno, da je s stroji poskušala nadomestiti moč človekovih mišic. Bremena, ki smo jih včasih dvigali ročno, danes dvigajo stroji. Podobne cilje si zastavljamo, ko govorimo o strojni obdelavi informacij oziroma podatkov in o njihovi predstavitvi. Ideja o obdelavi podatkov s strojem ni nova kot bomo pozneje v kratkem zgodovinskem pregledu razvoja računalnika podrobneje videli. Diferenčni stroj Charlesa Babbagea iz leta 1833 štejemo za enega izmed najresnejših poskusov, ki pa ni uspel zaradi neustreznosti tehnologije. Naprava je bila mehanska. Sestavljena je bila iz ogromnega števila vzvodov in kolesc. Šele elektronska tehnologije, ki se je pojavila po drugi svetovni vojni, je omogočila realizacijo učinkovitih tovrstnih naprav.

Kaj je torej **računalnik?** Zakaj beremo v časopisih toliko vesti o novih računalnikih, zakaj se je ta novi pojem tako nagloma vrinil v našo zavest in v naše življenje? V svojem razvoju je človek iznašel že množico strojev in pripomočkov, pa vendarle so šle in še gredo iznajdbe kar nekako mimo nas. O računalniku pa toliko besed. Vreden je toliko besed, saj govorimo o stroju, ki bo zelo kmalu postal tako nepogrešljiv del našega vsakdanjika, kot so danes avtomobil, radio, telefon, televizija ali pa električna energija.

Pravijo, da sta v naše življenje prinesla največ sprememb podreditev ognja in iznajdba kolesa. Sodijo, da je iznajdba računalnika za človeštvo skoraj toliko pomembna, kot ti dve pridobitvi. Gotovo pa sodi v enako skupino, kot izum tiskarskega stroja.

Glavna lastnost računalnika je,da izredno hitro računa; poleg tega pa zmore opraviti še primerjave, prikazati odločitve brez človeških slabosti z matematično natančnostjo in najti najboljše rešitve. V svojem elektronskem drobovju je sposoben premleti take množice podatkov, ki jih ne bi obvladala niti cela armada uradnikov, statistikov in matematikov. Skratka, brez računalnika bi ves naš razvoj usodno zastal. Če bi naenkrat po vsem svetu odpovedali vsi računalniki, bi dobesedno v hipu nastala nepopisna zmeda. Letala, ki jih vodijo računalniki, bi zgrešila smer; v računalniško vodeni proizvodnji bi začeli iz strojev dobivati izdelke, ki bi jih lahko odpeljali kar na odpad, v skladiščih bi imeli nenadoma preveč izdelkov iste vrste, po drugih pa bi zaradi pomanjkanja naraslo povpraševanje, prav tako ne bi vedeli, koliko ljudi mora plačevati razne dajatve, koliko je vojaških obveznikov in kje stanujejo itd. Brez računalnika bi se človeštvo vrnilo za kakih 40 let nazaj. Na srečo pa se kolesje zgodovine ne vrti nazaj in tako smo sodobniki tehnološke revolucije, ki jo povzročajo računalniki in kakršne človeštvo še ni doživelo.

Z imenom računalnik torej označujemo najpomembnejšo napravo za avtomatsko obdelavo podatkov. Čeprav beseda računalnik izvira iz besede računati, računalnik ni le stroj za avtomatsko računanje. Računski ali kot jih imenujemo numerični podatki, so le ena vrsta podatkov. Še več! Večina podatkov v praksi je nenumerične narave. Primer nenumeričnih podatkov je npr. množica imen in priimkov, ki jih obdelamo z računalnikom tako, da jih razvrstimo po abecednem redu in podobno.

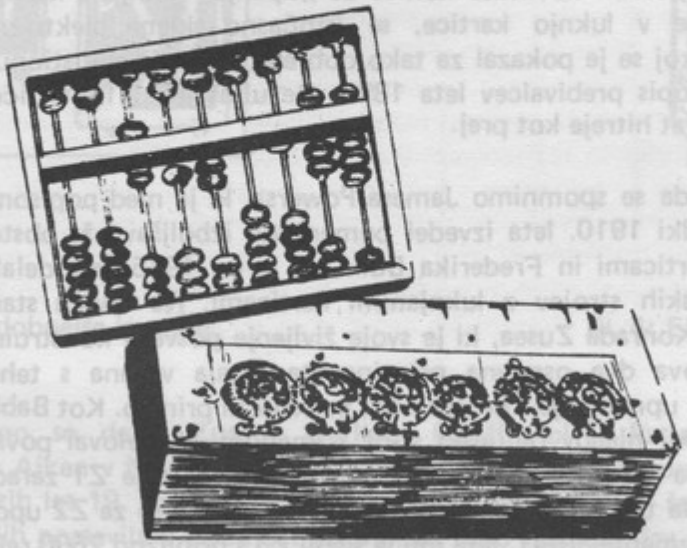
Računalništvo ni le veda o računalnikih, ampak o vsem, kar je v zvezi z avtomatsko obdelavo podatkov. Besedi računalništvo in informatika označujeta dve področji, ki se v veliki meri prekrivata.

Zakaj potrebujemo informacije oziroma razne podatke? V splošnem lahko rečemo, da je informacija bistveni element v procesih odločanja in upravljanja. Vse naše razumne odločitve temeljijo na podatkih, ki so nam na voljo. Pomislimo samo na vožnjo s kolesom. Kako pomembna je informacija o okolju in samem kolesu, če ga želimo smiselno in varno upravljati. Vedo, ki se v širšem smislu ukvarja s procesi upravljanja bodisi v živih organizmih ali strojih, imenujemo **kibernetika**.

Tako smo v kratkem spoznali osnovne pojme in vede, ki spremljajo računalnik in njegov razvoj. Vidimo, da brez ustreznih informacij ni smiselnega upravljanja, obenem pa, da je poznavanje osnovnih principov informatike, računalništva in kibernetike že nuja in obveza tega časa.

2. RAZVOJ RAČUNALNIŠTVA

Pričnimo naš sprehod skozi zgodovino razvoja računalnika z letom 1884, ko je znani francoski matematik in fizik Blaise Pascal pri svojih 19 letih izdelal enostaven digitalni računalnik, ki je zmogel seštevanje in odštevanje. S tem smo preskočili zgodnji razvoj računalnikov, npr. prvi digitalni mehanizem abacus (računalo s kroglicami), ki je nastal več kot 3000 let prv.n.št. in ga še vedno učinkovito uporabljajo v mnogih delih sveta. Bil je delo španskega misionarja Raimundusa Lullusa, ki je pod vplivom verske motiviranosti poskušal 1275. leta zgraditi stroj, sposoben logičnega zaključevanja in ne nazadnje računski stroj z zelo enostavnim mehanizmom za seštevanje in odštevanje, ki ga je leta 1623 skonstruiral Wilhelm Schichard.



Slika 2: Prva „računalnika": ABACUS in PASCAL-ov računski stroj

Vsekakor moramo omeniti leto 1672, ko je v Nemčiji Wilhelm von Leibnitz uporabil zobata kolesca v stroju, ki ni znal samo seštevati in odštevati, ampak tudi množiti, deliti in celo izračunati kvadratni koren. Več kot 150 let pozneje, leta 1835 je cambriški matematik Charles Babbage zasnoval stroj, ki mu je kljub temu, da ga ni nikoli zgradil, prislužil skoraj splošno priznanje očeta modernega računalnika. Imel je vhodne in izhodne naprave, ki so uporabljale luknjane kartice podobne tistim, ki so tkalskemu stroju Jacquarda služile za izdelavo vnaprej določenega vzorca. Poleg tega je Rabbase predvidel „shrambo" (pomnilnik) in mlin (procesna enota). Obe enoti naj bi prav tako nadzoroval s karticami, na katerih so bili ukazi s številkami shranjeni, dokler niso bili potrebni za upravljanje stroja. Kot smo že omenili, ideje ni ustvaril, saj takratna tehnologija ni bila zmožna izdelati tako zahtevnega stroja.

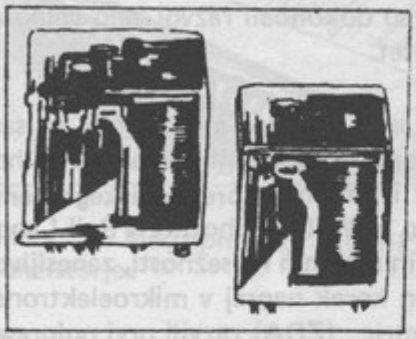
Babbage in njegove „računalniške" ideje so odšle v pozabo. Šele po drugi svetovni vojni so odkrili, da je zasnova njegovega analitičnega stroja v bistvu predhodnik današnjih računalnikov. Tako se lahko potegne črta razvoja, ki vodi od stroja s štirimi računskimi operacijami, preko analognega računala do centralne enote današnjega računalnika.

Druga črta razvoja vodi preko perifernih enot: iznajdbe pisalnega stroja in njegove kombinacije s papirnim trakom, kar je bilo plod razvoja telegrafije, dalje uvajanje luknjanih kartic na področjih, kjer se pojavljajo velike količine podatkov in ob tem specialne enote za luknjanje kartic. Pri Jacquardu je luknjana kartica služila za upravljanje procesa (tkanje . . .), Hollerithu pa pripada zasluga, da je luknjana kartica uvedena kot nosilec podatkov in to predvsem numeričnih. Herman Hollerith je uporabil kovinske konice za „tipanje" luknjane kartice; ko kovinska konica zadene v luknjo kartice, se istočasno sklene električni kontakt. Ta Hollerithov stroj se je pokazal za tako dobrega, da ga je statistični urad v Ameriki uporabil za popis prebivalcev leta 1890. Rezultat je bil fantastičen, saj so popis izvedli desetkrat hitreje kot prej.

Prav je tudi, da se spomnimo Jamesa Powersa, ki je med popisom prebivalstva v Severni Ameriki 1910. leta izvedel pomembno izboljšavo že obstoječih naprav z luknjanimi karticami in Frederika Bulla, ki je od 1915. leta delal na izboljšavah elektromehanskih strojev z luknjanimi karticami. Na koncu starejše generacije omenimo še Konrada Zusea, ki je svoje življenje posvetil konstruiranju računskih strojev. Njegova dva osnovna principa sta ostala vezana s tehniko modernih računalnikov: upravljanje s programom in dvojiški princip. Kot Babbageov analitični stroj je tudi njegov računski stroj z imenom Z1 deloval povsem mehanično (1936. leta), a že s programiranim upravljanjem. Ker je Z1 zaradi mehaničnega prenosa signala deloval zelo počasi in s težavo, je Zuse za Z2 uporabil releje kot prekinjalne elemente. 1941. leta mu je sledil Z3 s približno 2600 releji in z razvitim algoritmičnim jezikom PK, ki je predhodnik PL/1 in ALGOL-a 68.

Epohalne ideje pa se navadno ne rojevajo samo na enem mestu sveta ali samo v enih možganih. Zato ni nič čudnega, da je Louis Couffignol leta 1936 v Franciji opisal računski stroj s programiranim upravljanjem in dvojiškim številskim sistemom. Istega leta je začel G. Stibitz v laboratorijih Bell Telephone konstrukcijo relejnih računalnikov. Leta 1942 je bil v Bell Telephone končan prvi računalnik s programiranjem na osnovi 50 relejev. „Ballistic computer" je vseboval leto kasneje že preko 1300 relejev. Največji računalnik v razvoju te vrste je bil Bellov model z 9000 releji.

Leta 1937, ko je Stibitz začel svoje delo v Bellu, je Howard Aiken, profesor na harvardski univerzi, predlagal IBM-u, da s sredstvi, namenjenimi za naprave z luknjanimi karticami izdela programiran računalnik. To izredno plodno sodelovanje je botrovalo, da se je 1944. leta pojavil znani MARK I (Automatic Sequence Controlled Calculator). Model MARK I je bil večji in zmogljivejši od Z3 in modela Bell. Leta 1948 se je pojavil na tržišču model MARK II s preko 13000 releji.



Sl. 3: Releja sodobnejše izvedbe

Sl. 4: Elektronska cev

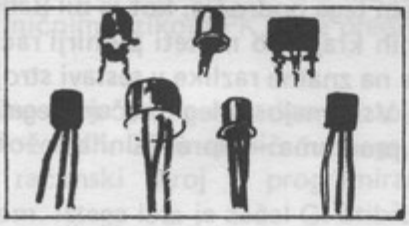
Skoraj istočasno so delali Zuse v Berlinu, Couffignol v Parizu, Stiblitz v Murray-Hillu in Aiken v New Yorku na nadaljnjih izboljšavah računalnikov. Tako so ideje tridesetih let 19. stoletja dale krila snu o računalnikih – šele 30. leta 20. stoletja pa so jih postavila v stvarnost in prakso. Delo iznajditeljev 20. stoletja je našlo razumevanje in tehnologijo, ki je pionirjem tega področja, kot je bil Babbage, manjkala. Neodvisno eni od drugih, na različnih krajih so našteti pionirji računalništva delali za uresničitev istega cilja. Ne glede na znatne razlike v sestavi strojev je bila njihova osnovna konstrukcija podobna. Vsi imajo poleg računskega dela: vhod podatkov – spomin podatkov – vhod programa – upravljalni del – izhod podatkov.

Odločilni korak v tehniki je bil prehod od elektromehaničnih k elektronskim elementom: namesto relejev kot prekinjal so uporabili elektronko (elektronsko cev). Uporaba elektronke, ki so jo izumili Fessenden, De Forest in drugi v začetku 20. stoletja za kontrolo tokov, je začela prodirati v računalnike v začetku štiridesetih let. Leta 1945 je znani matematik John von Neumann zasnoval pojem shranjenega programa. Misel von Neumanna je bila, če se lahko ukazi prenašajo kot podatki, zakaj se ne bi tudi program v teku dela sam modificiral. Ukazi za delo računalnika so tako shranjeni znotraj računalnika v številski obliki. Kot rezultati so se lahko izvajale logične odločitve v računalniku, ki je med delom tudi spreminjal ukaze. To je bil pomemben korak naprej, ki se je v mnogočem opiral na Babbageovo delo, staro več kot 100 let. Tako so bili zgrajeni do leta 1949 EDVAC (Electronic Discrete Variable Automatic Computer) na Princetonu, zelo znani ENIAC (Electronic Numerical Integrator and Calculator) v Pennsylvaniji in EDSAC (Electronic Delay Storage Automatic Calculator) v Cambridgeu v Angliji. ENIAC je bil prvi čisto elektronski računalnik. Zasnovala sta ga John Mauckly in Prosper Eckert. Končan je bil leta 1946, stal pa naj bi okoli 10 milijonov dolarjev; vseboval je 18000 elektronk in tehtal okoli 30 ton.

S pomočjo novih medijev za pomnilnik, kot so magnetni bobni in magnetna jedra, je rasla njihova zmogljivost. Sledili so novi in čedalje bolj komplicirani računalniki.

IBM-ov „Poppa" je leta 1948 že vključil pogojne skoke, EDSAC (1949) je bil prvi, ki je imel hitri pomnilnik za dvojiška števila. SSEC, EDVAG, ILLIAC, MANIAC, „Whirewind" ali vrtinec, MADM in UNIVAX so dokončali razvoj zelo velikih in zelo dragih strojev v začetku in sredi petdesetih let.

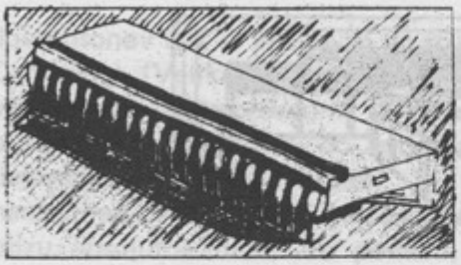
Do te točke je bil napredek sorazmerno počasen. Računalniki so bili ne samo dragi, ampak so tudi zahtevali veliko prostora in sestavnih delov za njihovo izdelavo. Vse to se je dramatično spremenilo z izumom **polprevodniškega elementa.** Leta 1948 so John Bardeen, Walter Brattain in William Shottky v Bell Laboratories v ZDA razvili **tranzistor,** ki je zaradi svojih majhnih razsežnosti, zanesljivosti in majhne porabe moči pomenil revolucionaren korak naprej v mikroelektronsko dobo. V letu 1951 so raziskovalci Western Electric (ZDA) razvili prvi polprevodniški ojačevalnik — tranzistor s točkastim kontaktom. Leta 1958 je Fairchild kot (ZDA) izdelal slojni tranzistor z uporabo silicijevega oksida kot izolatorja. V letu 1959 sta Texas Instruments in Fairchild razvila polprevodniški paket z dvema ali več tranzistorji v isti silicijevi podlagi. Cena se je tako bistveno znižala in pot je bila odprta za čedalje več komponent na isti ploščici silicija.



Sl.5: Nekaj vrst in oblik tranzistorjev

Sl.6: TO in DIL ohišje integriranih vezij

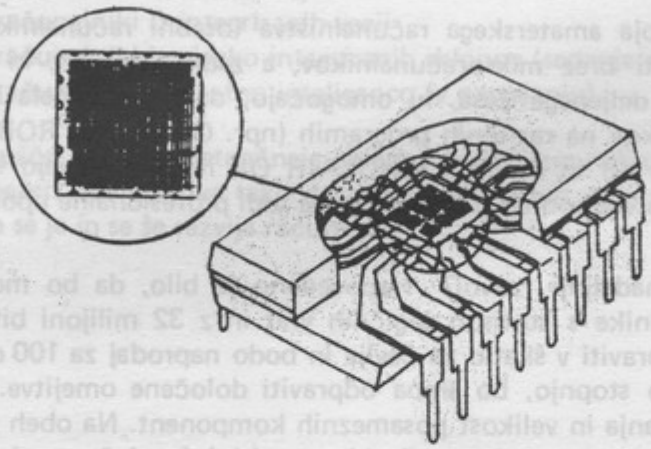
Leta 1965 je zabeležen še eden od pomembnih korakov v razvoju računalnikov: uvedena so bila **integrirana vezja,** kot novi prekinjalni elementi. Za primerjavo lahko integrirano vezje primerjamo s kontrolno ploščo, na kateri so vsa stikala in ostale komponente, kot so upori in kondenzatorji, pritrjeni na urejen način in čimbolj skupaj. To ploščo sedaj zmanjšamo iz velikosti mize na površino, manjšo kot poštna znamka. Vsako stikalo zavzema le nekaj milijonink metra, povezave med njimi potekajo v dveh ali več nivojih v izolacijski plasti, ki prekriva stikala. Plošča še vedno zahteva ozemljitev in priključek za električno napajanje, prav tako vhodne in izhodne priključke, zato tudi toliko priključnih nožic na sliki 7.



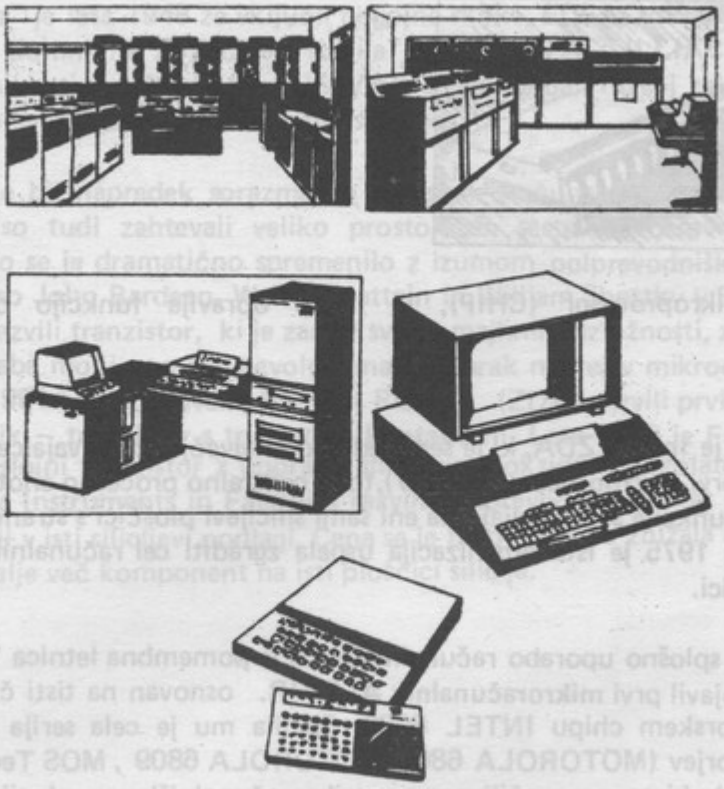
Slika 7: Mikroprocesor (CHIP), ki lahko opravlja funkcijo okoli 35.000 tranzistorjev

V letu 1971 je Intel v ZDA, ki je sedaj eden od največjih proizvajalcev integriranih vezij, razvil prvi **mikroprocesor** (slika 7),to je centralno procesno enoto. Logične in aritmetične funkcije se opravljajo na eni sami silicijevi ploščici s stranico,manjšo od 0,5 cm. Leta 1975 je ista organizacija uspela zgraditi cel računalnik na eni sami tiskani ploščici.

Za široko in splošno uporabo računalnikov pa je pomembna letnica 1974, ko se je na tržišču pojavil prvi **mikroračunalnik ALTAIR**, osnovan na tisti čas najboljšem mikroprocesorskem chipu INTEL 8080. Sledila mu je cela serija kvalitetnejših mikroprocesorjev (MOTOROLA 6800, MOTOROLA 6809 , MOS Techology 6500 in 6502, itd.), ki so povzročili pravo „mikroračunalniško revolucijo" v ZDA in pozneje tudi drugod po svetu, saj se je njihova uporaba eksplozivno širila in to predvsem zaradi znatno nižje cene, majhne velikosti in minimalnih stroškov vzdrževanja.



Slika 8. Notranjost mikroprocesorja



Slika 9: Zmanjševanje sestavnih delov računalnika omogoča stalno zmanjševanje celotne naprave

Svetovnega razvoja amaterskega računalništva (osebni računalniki), si danes ne moremo zamisliti brez mikroračunalnikov, a zadnji dosežki, ki vsebujejo tudi majhne sisteme deljenega časa, ki omogočajo, da na njih dela istočasno večje število uporabnikov na različnih programih (npr. GENERAL ROBOTIC CORPORATION: MUT/X3 in FD/X3, SINCLAIR QL itd.) prispevajo k še hitrejšemu prodoru le-teh na vsa področja amaterske pa tudi profesionalne uporabe.

In kakšen bo nadaljnji razvoj? Napovedano je bilo, da bo mogoče današnje največje računalnike s stotisoč logičnih vrat in z 32 milijoni bitov pomnilnika konec stoletja spraviti v škatlo za čevlje in bodo naprodaj za 100 dolarjev. Da bi lahko dosegli to stopnjo, bo treba odpraviti določene omejitve. Med njimi sta hitrost preklapljanja in velikost posameznih komponent. Na obeh področjih je bil dosežen napredek, prav tako tudi pri uporabi infrardeče svetlobe, kot nosilca digitalne informacije.

Med drugim potekajo poskusi s sintetičnimi kovinami (električno prevodne organske trdne snovi) z galijevim arzenidom. Silicij na safirju (SOS) je poskus zmanjšati električne izgube in s tem zmanjšati razsežnosti komponent. Najbolj znan eksperiment predstavlja **Josephsonov stik.**

Josephsonov stik je imenovan po študentu podiplomskega študija, ki je na Univerzi Cambridge (Velika Britanija) odkril, da teče tok skozi stik med dvema superprevodnikoma (superprevodnik je skoraj brez električne upornosti: dobimo ga z ohlajanjem kovin skoraj do temperature absolutne ničle). Čas, ki je potreben za delovanje vrat, je zelo kratek, 1/7 triljoninke sekunde. Superprovodnike lahko namestimo zelo tesno skupaj. Slabost je v zahtevi, da moramo doseči temperature blizu absolutne ničle. To pomeni, da je treba komponente obdati s tekočim helijem, kar ni poceni. Vendar, če bomo kot rezultat dobili računalnik z več kot 100 milijoni operacij v sekundi, bodo stroški opravičeni.

Z Josephsonovim računalnikom tekmuje „optični računalnik". Za preklapljanje potrebuje 2 stabilni stanji (za predstavitev „0" in „1"). To bistabilnost dosežemo z optičnim resonatorjem, ki je znan kot FABRIPERTOV interferometer, kjer se svetloba odbija od dveh vzporednih plošč, da se doseže zaželeni učinek. Njegovi zagovorniki trdijo, da je prav tako hiter kot Josephsonov stik, ima pa dodatno prednost: združljiv je z optičnimi vlakni.

Za infrardečo svetlobo so ugotovili, da je zelo zmogljiv prenosni medij za digitalne podatke. Ker reagira prav tako kot vidna svetloba, je uporabna za majhne razdalje. Signal proizvaja svetlobna dioda (LED), sprejema pa ga foto dioda.

Druge razvojne in za sedaj raziskovalne možnosti so za naš krajši sprehod v osnove računalništva skoraj preštevilne. Naštejmo le najvažnejše: uporaba zelo majhnih laserjev za tiskanje in za shranjevanje podatkov v tekočih kristalih, računalniške kontrolne enote in kontrolne enote mehurčkastega pomnilnika, itd.

Omenimo še razdelitev računalnikov v generacije:

1. generacija: računalniki iz elektronk
2. generacija: računalniki iz tranzistorjev
3. generacija: računalniki iz integriranih vezij
4. generacija: računalniki iz visoko integriranih sklopov (sedanje obdobje)
5. generacija: računalniki z umetno inteligenco (v nastajanju).

Prostor ne dopušča, da bi si natančneje ogledali zgodovino razvoja računalništva, toda tudi naš bežen oris odkriva tako del narave modernih računalnikov, kot tudi hitrost, s katero se je in se še razvija računalnik.

3. O INFORMACIJAH IN NJIH PREDSTAVITVI

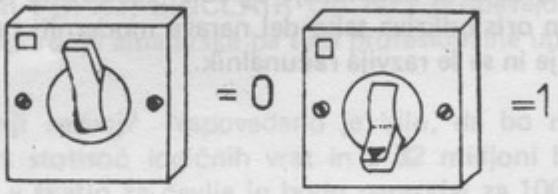
V prvem poglavju smo že omenili, da lahko računalnik imenujemo napravo za obdelavo informacij. Povedali smo tudi, kaj razumemo pod pojmom informacija. Poglejmo si še nekaj osnovnih pojmov o informacijah, ki so za nadaljnjo predstavitev in razumevanje računalnika ter računalniške tehnologije nasploh pomembni.

Kadar hočemo s kom izmenjati informacijo, jo moramo predstaviti z izraznimi sredstvi, ki so nam na voljo, npr. s svetlobnimi, zvočnimi, pisanimi in drugimi znaki. Ne glede na to, ali govorimo o predstavitvi podatkov v zvezi s človekom ali s strojem, ločimo dva bistveno različna načina predstavitve:

- analogni ali zvezni (neprekinjen) način
- digitalni ali diskretni (prekinjen, skokovit) način.

Termometer je odličen primer, pri katerem se dvig oziroma padec temperature meri s pripadajočo dolžino živosrebrnega stolpca, ki se giblje „zvezno" in lahko glede na temperaturo zavzame na skali poljubno lego. Torej je ta informacija analogna. Digitalne naprave pa so povezane s štetjem (npr. digitalna ura, števec prevoženih kilometrov v avtomobilu, itd.).

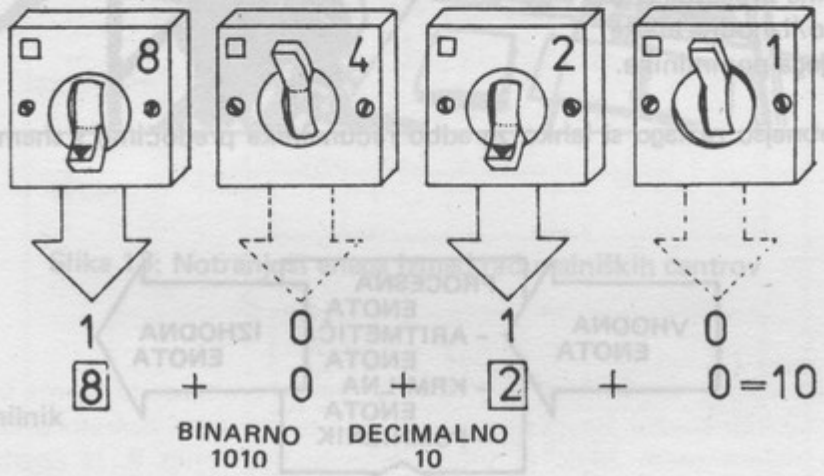
Glede na ta dva načina delimo računalnike v analogne in digitalne. V prvih je informacija predstavljena analogno, v drugih pa digitalno. Ker sodijo običajni sodobni računalniki med digitalne, se bomo v nadaljevanju omejili na njim ustrezno predstavitev podatkov. Sodobni računalniki torej temeljijo na t.i. digitalni elektroniki, kjer sta za opis informacij na voljo le dva znaka: 0 in 1. V matematiki imenujemo takšen številčni sistem **dvojiški ali binarni sistem**. Ko je George Boole razvil algebro, s katero se razmerje med velikim številom elementov popiše s ponavljanjem operacij med vrsto parov elementov, je postavil osnovo za današnji napredek mikroelektronike, saj simbola 0 in 1 nista omejena samo na števila. Če se uporabljata za „vključen" in „izključen", predstavljata dva položaja stikala (Slika 10).

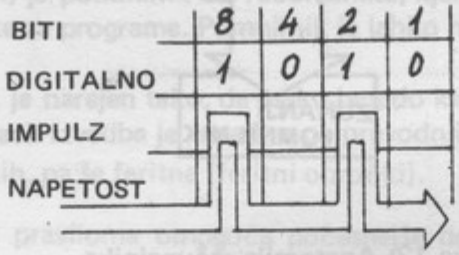


Sl. 10: Predstavitev osnovne informacije s stikalom

Lahko predstavljata tudi druge alternative,kot „DA" ali „NE" ali „in" in „ali' , itd. Čemu ta skromnost v zapisu? To narekuje narava elektronskih komponent računalnika in s tem je omogočena natančnost in zanesljivost delovanja.

Zanesljivost je pojasnjena s tem, da lažje razločimo med dvema simboloma,kot npr. med več kot 50 simboli, ki jih uporabljamo v zapisovanju. Lažje ugotovimo, ali po nekem vodniku teče električni tok (znak 1) ali pa ne (znak 0), kot pa kolikšen je ta tok (npr. 15,6, mA). Z ustrezno kombinacijo teh dveh znakov lahko zapišemo katerikoli znak abecede, števila itd. (npr. število 9 je 1001, črka M je 01001101 itd.). Vidimo torej, da s kombinacijo ničel in enic lahko zapišemo katerikoli alfanumerični znak ali informacijo.





Slika 11: Pretvorba decimalnega zapisa v digitalni ter primerjava analognega in digitalnega zapisa
(1010 = (1x8) + (0x4) + (1x2) + (0x1) = 10)

Osnovna enota informacije se imenuje BIT, kar je analogno 0 in 1 dvojiškega sistema, oziroma „pravilno" ali „napačno" Boolove algebre. Torej se z biti lahko opišejo črke, znaki, besede in zvoki. Za ponazoritev enega znaka vzamemo ponavadi osem bitov–BYTE. Osem bitov opiše lahko 2 na osmo potenco,to je 256 različnih možnosti, kar je dovolj za latinsko abecedo, številke od 0 do 9 in ločila. Prav tako se vse besede v splošnem lahko razbijejo v štirideset osnovnih glasov, ki jih imenujemo „fonemi". Vsak fonem sestavlja vrsta frekvenc v določenem zaporedju. V računalniku ga označimo s šestimi biti.

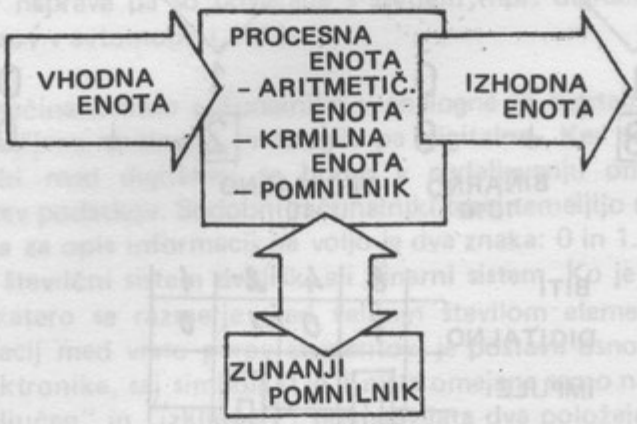
Velikost računalnikovega pomnilnika navadno izrazimo v BYTIH. Hitrost prenosa informacij izražamo v BAUDIH, ki so pogosto enaki hitrosti v „BITIH NA SEKUNDO". Byte in baud se pretežno uporabljata v računalniških krogih. Izraz za enoto informacije „bit" pa je že splošno sprejet.

4. ANATOMIJA IN DELOVANJE RAČUNALNIKA

Vsak računalnik je v glavnem sestavljen iz treh osnovnih komponent:

- centralne procesne enote
- vhodno/izhodne enote
- zunanjega pomnilnika.

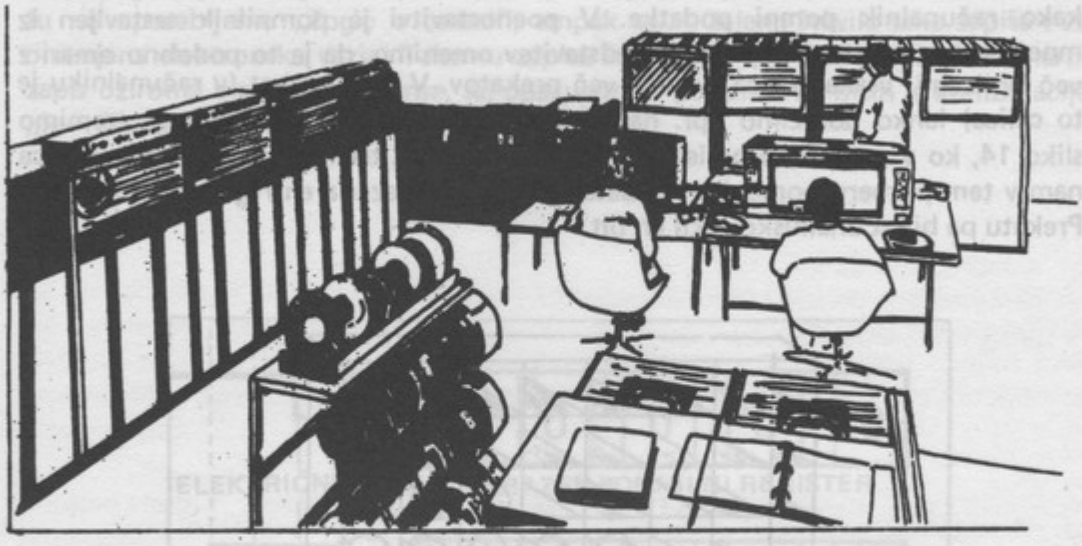
Za podrobnejšo razlago si lahko zgradbo računalnika predočimo s shemo na sliki 12.



Slika 12: Anatomija računalnika

Osrednji del računalnika je centralna procesna enota (CPE), saj v njej poteka obdelava podatkov. Sestavljajo jo trije osnovni deli in sicer: aritmetične enote, krmilna enota in pomnilnik. V pomnilniku je lahko shranjena kakršnakoli informacija, ki jo je računalnik sposoben sprejeti; to so programi, podatki, vmesni in končni rezultati itd. V aritmetični enoti se izvajajo aritmetične operacije, kot so seštevanje, odštevanje, množenje, deljenje itd. Krmilna enota pa bdi nad delovanjem celotnega računalnika in odvijanjem programa v računalniku. Nadzira in vsklajuje delovanje posameznih enot, da le-te predstavljajo delujočo celoto, ki jim pravimo računalnik. Vhodno/izhodne enote omogočajo izmenjave podatkov med računalnikom in uporabnikom. Zunanji pomnilnik lahko obravnavamo na dva načina: kot razširitev notranjega pomnilnika in kot vhodno/izhodno enoto.

V nadaljevanju tega poglavja si nekoliko pobliže oglejmo pomembne „organe" računalnika!



Slika 13: Notranjost enega izmed računalniških centrov

4.1. Pomnilnik

Kot smo že omenili, je pomnilnik del računalnika, kjer shranjujemo podatke, to je programe in podatke za programe. Pomnilnik je lahko notranji ali zunanji.

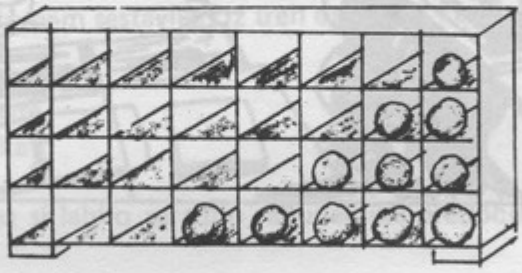
Notranji pomnilnik je narejen tako, da vsako besedo kar najhitreje najdemo in tudi uporabimo v obdelavi. Izvedba je navadno polprevodniška, pri nekaterih, predvsem starejših računalnikih, pa še feritna (feritni obročki).

Zunanji pomnilnik praviloma omogoča počasnejše doseganje podatkov, njegova kapaciteta pa je navadno mnogo večja. Tu ge za naprave, ki delujejo podobno kot magnetofon. Podatke shranjujejo na magnetne površine v obliki trakov, diskov, disket, kaset, bobnov,itd.

Magnetni trak lahko primerjamo z magnetofonskim trakom. Navadno je več sto metrov dolg in 1 cm širok. Na vsak cm dolžine lahko zapišemo nekaj sto računalniških besed. Magnetni disk ima magnetno zapisovalno površino v obliki okrogle plošče. Informacija se zapisuje v koncentričnih krogih. Na vsaki strani je nekaj sto takih informacijskih krogov ali sledi. Premer plošče je nekaj dm. Navadno so plošče naložene druga na drugo, kar veča površino magnetnega zapisa in s tem pomnilno kapaciteto.

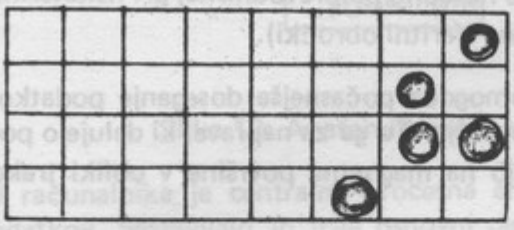
Seveda imamo še druge vrste magnetnih pomnilnikov kot npr. gibke diskete, kasete različnih velikosti z magnetnimi trakovi, magnetne bobne itd. V uporabi so tudi starejši načini zapisa na karticah, papirnih trakovih,itd.

Naj ne bo odveč, da v tem poglavju skušamo razvozlati in razložiti še postopek, kako računalnik pomni podatke. V poenostavitvi je pomnilnik sestavljen iz množice t.i. registrov. Za lažjo predstavitev omenimo, da je to podobno omari z več policami, vsaka polica pa ima več prekatov. V vsak prekat (v računalniku je to celica) lahko položimo npr. največ eno žogico (sl. 14). Kot primer vzemimo sliko 14, ko v zaporedne police vložimo po eno, dve, tri in pet žogic. Vsaka polica nam v tem primeru pomeni en podatek in torej ponazarja en register pomnilnika. Prekatu pa bi računalniško rekli en bit.



Slika 14

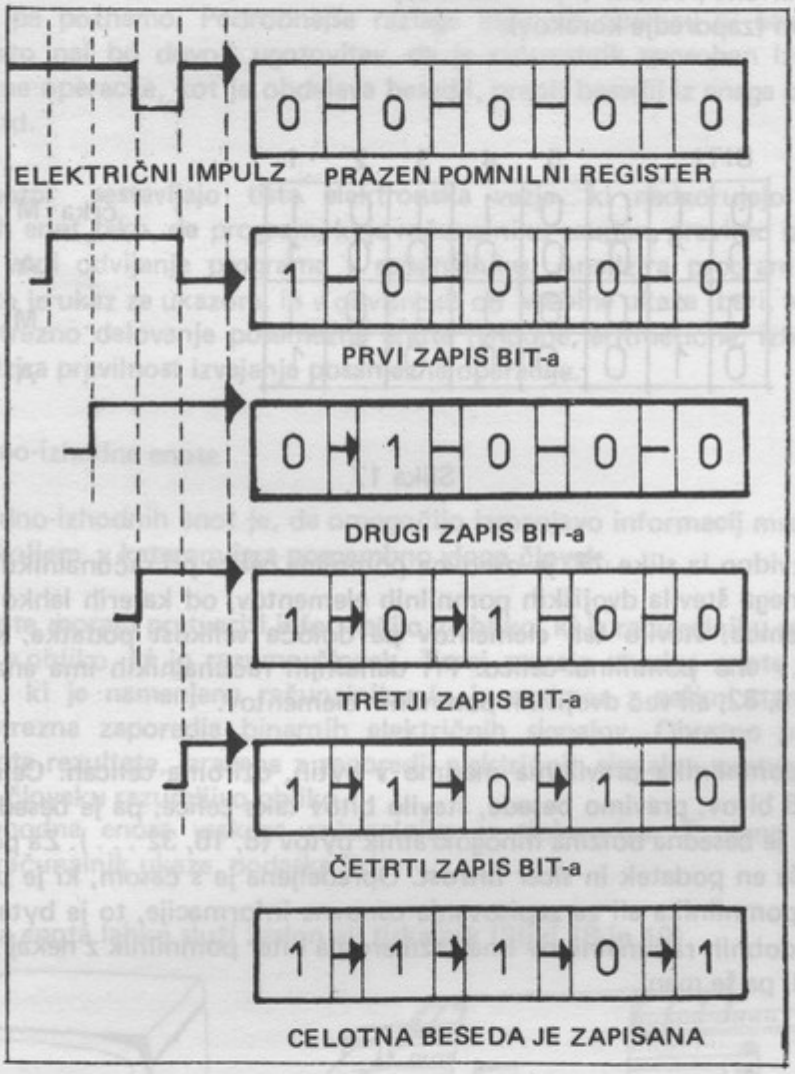
Slaba stran takega pomnilnika je v tem, da število prekatov (oziroma bitov) zelo omejuje največji možni številčni podatek. Na police s po osmimi prekati lahko položimo največ osem žogic in torej pomnimo le število 8. Iz zagate nas reši majhen trik, ki ga nazorno pokaže slika 15. Bite smo označili s številkami 1, 2, 3, 4 in 5 in to od desne proti levi.



Slika 15

Število 1 pomnimo sedaj z žogico v bitu 1, število 2 v bitu 2, število 3 z žogico v bitih 1 in 2, itd. Iz slike je razvidno, da lahko števila od 0 do 3 pomnimo že samo z dvema bitoma oziroma s kombinacijo žogic v dveh prekatih. Na enak način lahko v treh bitih pomnimo števila od 0 do 7, v osmih bitih pa imamo na voljo kar 256 kombinacij, kar pomeni, da lahko v njih pomnimo števila od 0 do 255.

Enak princip pomnenja ali zapisa števil uporabljamo v praksi pri računalnikih, le da tu ne uporabljamo „žogic v omari", ampak se želeno število tako zapiše v že omenjena elektronska vezja. Takemu zapisu števil smo že rekli dvojiški ali binarni zapis oziroma binarno kodiranje, saj opazujemo v posameznih bitih le kombinacije „je–ni", oziroma „da–ne", oziroma „1–0".



Sl 16: Prikaz zapisa besede „11101" in njegova zapomnitev v pomnilni celici

Binarno kodiranje lahko uporabljamo tudi za pomnenje oziroma zapis nenumeričnih podatkov. Za primer vzemimo, da kombinaciji 01000001 ustreza črka A, kombinaciji 01001101 pa črka M. Če v pomnilnik shranimo kombinacije s slike 17, smo tako pomnili besedo MAMA. Z osmimi biti imamo seveda na voljo dovolj kombinacij za vse črke, ločila in druge znake, kot smo že omenili. V pomnilnik lahko torej „zašifriramo" poljubno besedilo. Računalnik torej lahko v pomnilniku pomeni binarno kodirane:

- numerične podatke,
- nenumerične podatke (npr. besedilo),
- program (zaporedje korakov).

BITI			5	4	3	2	1	
0	1	0	0	1	1	0	1	črka M
0	1	0	0	0	0	0	1	A
0	1	0	0	1	1	0	1	M
0	1	0	0	0	0	0	1	A

Slika 17

Kot je razvidno iz slike 17, je osnovna pomnilna celica pri računalniku sestavljena iz določenega števila dvojiških pomnilnih elementov, od katerih lahko vsak hrani ničlo ali enico, število teh elementov pa določa velikost podatka, ki ga lahko spravimo v eno pomnilno celico. Pri današnjih računalnikih ima ena pomnilna celica 8, 16, 32, ali več dvojiških pomnilnih elementov.

Velikost pomnilnika praviloma merimo v bytih, oziroma celicah. Celicam, ki so večje od 8 bitov, pravimo **besede**, število bitov take celice, pa je besedna dolžina. Praviloma je besedna dolžina mnogokratnik bytov (8, 16, 32 . . .). Za pomnilnik je značilen še en podatek in sicer **hitrost**. Opredeljena je s časom, ki je potreben za branje iz pomnilnika ali za zapisovanje osnovne informacije, to je byta ali besede. Večina sodobnih računalnikov ima razmeroma hiter pomnilnik z nekaj milijardink sekunde ali pa še manj.

4.2. Aritmetično—logična in krmilna enota

Aritmetično-logična, ali kot jo tudi imenujemo računsko-logična enota računalnika, je splet elektronskih vezij, ki z našimi binarnimi podatki izvaja razne operacije, kot so aritmetične, logične operacije, itd. Kot že vemo, sodijo sem operacije seštevanja, odštevanja, ugotavljanje relacije enakosti, večjega in manjšega, itd.

Aritmetična enota lahko:

- zbriše vsebino nekega registra ,
- vpiše v nek register poljubno število,
- prepiše vsebino nekega registra v drug register ,
- izvaja aritmetične operacije (+, −, /, *, itd) med vsebinami registrov in zapiše rezultat v določen register,
- izvaja t.i. logične operacije nad vsebinami registrov.

Za uporabnika, ki se je doslej srečal le s kalkulatorjem, so logične operacije novost, vse ostalo pa poznamo. Podrobnejše razlage logičnih operacij bi nas zavedle v globine, zato naj bo dovolj ugotovitev, da je računalnik sposoben izvajati tudi nenumerične operacije, kot je obdelava besedil, prepis besedil iz enega pomnilnika v drugega,itd.

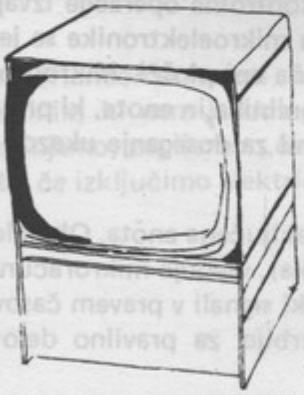
Krmilno enoto sestavljajo tista elektronska vezja, ki nadzorujejo delovanje posameznih enot tako, da program, ki je računalniku zaupan, pravilno izvrši. To je enota, ki vodi odvijanje programa v računalniku. Analizira program korak za korakom, to je ukaz za ukazom, in v odvisnosti od vsebine ukaza (beri, seštej, ipd.) sporoči ustrezno delovanje posamezne enote (vhodne, aritmetične, izhodne . . .) in tudi nadzira pravilnost izvajanja posamezne operacije.

4.3. Vhodno-izhodne enote

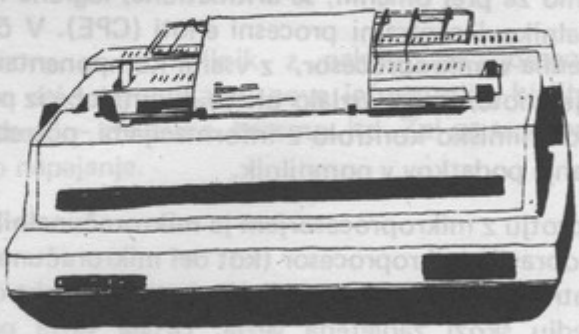
Naloga vhodno-izhodnih enot je, da omogočijo izmenjavo informacij med računalnikom in okoljem, v katerem igra pomembno vlogo človek.

Vhodne enote morajo pretvoriti informacijo v obliko, ki je računalniku razumljiva, izhodne pa v obliko, ki jo razume človek. Torej morajo vhodne enote pretvoriti informacijo, ki je namenjena računalniku in je zapisana z našimi standardnimi znaki v ustrezna zaporedja binarnih električnih signalov. Obratno pa morajo izhodne enote rezultate, izražene z zaporedji električnih signalov, pretvoriti v čim običajnejšo,človeku razumljivo obliko.
Običajna vhodna enota vsakega računalnika je tipkovnica. Z njeno pomočjo vnašamo v računalnik ukaze, podatke itd.

Kot izhodna enota lahko služi zaslon ali tiskalnik (Sliki 18 in 19).



Slika 18: Katodni zaslon



Slika 19: Tiskalnik

Poznamo še vrsto drugih vzhodno-izhodnih enot, kot so: čitalniki in luknjalniki kartic, teleprinterji, čitalniki in luknjalniki kartic, teleprinterji, čitalniki in luknjalniki traku, razni optični čitalniki pisanih in tiskanih znakov, grafične plošče, risalniki, svetlobna pisala itd.

Za udobnost izmenjave informacij bi bila velikokrat najprimernejša akustična komunikacija, vendar so uspehi na tem področju še dokaj skromni.

Ponavadi na procesni del računalnika ni priključena le ena vhodna in ena izhodna enota. Računalnik ima lahko več vhodnih in izhodnih enot (npr. terminal). S tem je dana možnost, da se zmanjša razkorak med hitrostjo teh enot in procesnega dela računalnika. Računalnik lahko deluje tudi tako, da obdeluje več programov hkrati in s tem vzporedno včita več vrst podatkov ali izpisuje več vrst rezultatov. Takšen sistem dela v računalniških krogih imenuje sistem deljenega časa (TIME–SHARING). To pomeni, da računalnik ciklično opravlja delo za vsakega izmed uporabnikov, ki so nanj istočasno priključeni. Ko je izvršil ukaz za enega uporabnika, preide računalnik na drugega itd. Na prvega se zopet vrne šele, ko je ugodil vsem delnim zahtevam uporabnikov. Seveda nobeden izmed uporabnikov tega ne zazna, saj se vse izvaja v izredno kratkem času (okoli 1/100 sekunde in še manj).

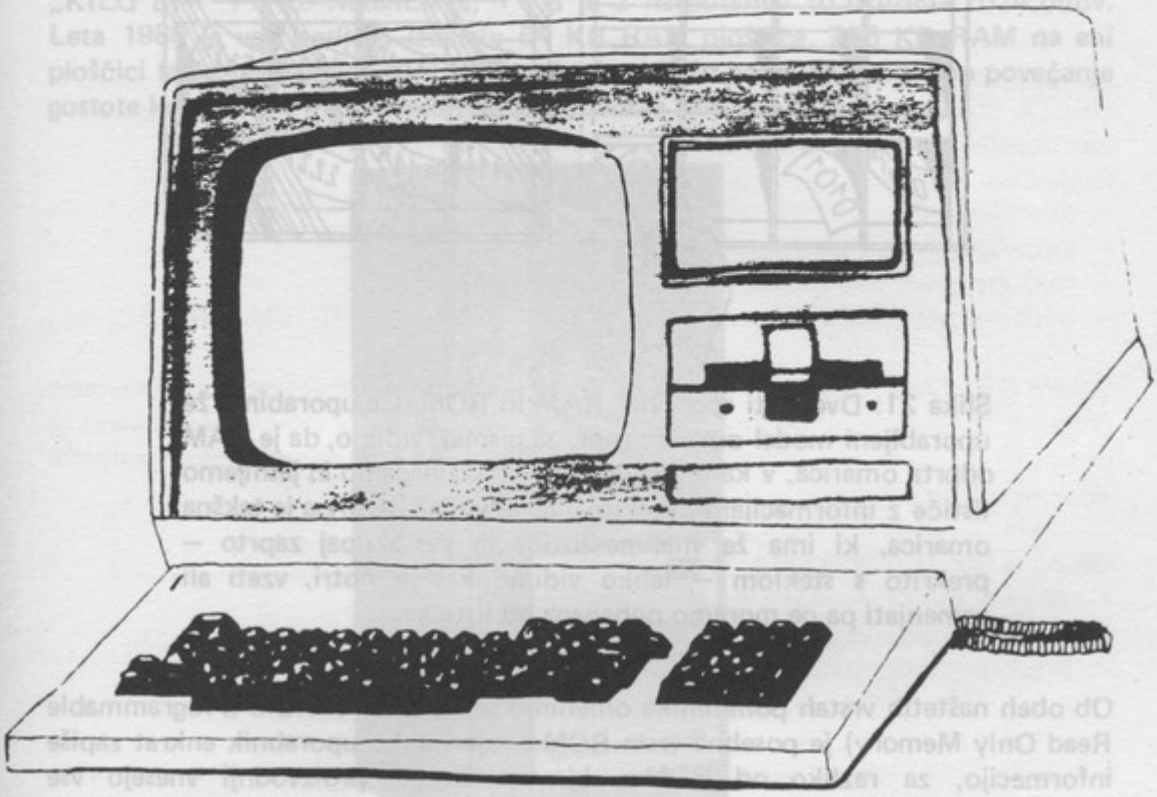
Če uporablja računalnik več uporabnikov hkrati, mora imeti vsak uporabnik svoj komplet vhodnih in izhodnih enot. Takemu kompletu za vhod in izhod pravimo TERMINAL. Terminal predstavlja navadno katodni zaslon, tipkovnice in tisklanika, lahko pa še dodatni elementi, kot npr. risalnik, razni čitalniki itd. Če ima terminal le zaslon in tipkovnico ga imenujemo zaslonski terminal (npr. PAKA 2000, ki so ga razvili v ISKRA DELTI in ga izdeluje GORENJE). Terminali so lahko v drugem kraju kot je računalnik. Za povezavo med terminali in računalniki moremo uporabljati že obstoječe telekomunikacijske naprave, kot npr. navadne telefonske linije, itd. Tako ima uporabnik na voljo računalnik na svojem delovnem mestu, ki je lahko tudi precej oddaljeno od centralnega računalnika.

5. MIKROPROCESORJI IN MIKRORAČUNALNIK

Kot smo že prej omenili, se aritmetične, logične in kontrolne operacije izvajajo v računalnikovi centralni procesni enoti (CPE). V času mikroelektronike se je CPE spremenila v **mikroprocesor**, z vsemi komponentami na eni plošči. Mikroprocesor vsebuje enote, ki razvozlajo ukaze, ki jih dobe iz pomnilnika, in enote, ki preskrbujejo pomnilniško kontrolo z informacijami, potrebnimi za doseganje ukazov in za pošiljanje podatkov v pomnilnik.

V nasprotju z mikroprocesorjem je **mikroračunalnik** zaključena enota. Ob nalogah, ki jih opravlja mikroprocesor (kot del mikroračunalnika), vsebuje mikroračunalnik še kontrolna vezja, ki zagotavljajo, da tečejo elektronski signali v pravem časovnem zaporedju skozi zapletena vezja. Druga vezja poskrbijo za pravilno delovanje notranjega pomnilnika in za vhodne-izhodne naloge.

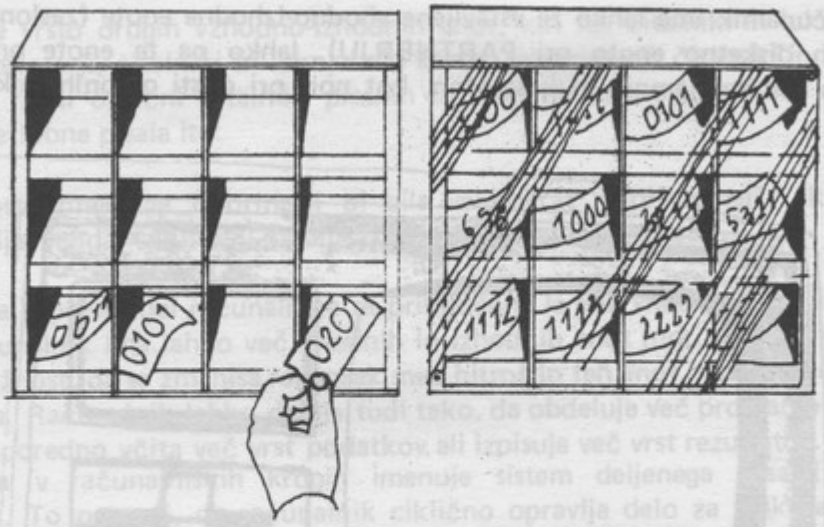
Mikroračunalnik ima lahko že vstavljene vhodno-izhodne enote (zaslon, diskovno enoto in disketno enoto pri PARTNERJU), lahko pa te enote predstavljajo običajen TV sprejemnik in kasetofon, kot npr. pri dosti osebnih mikroračunalnikih.



Slika 20: Mikroračunalnik PARTNER (Iskra Delta) Z 80, s pomnilnikom kapacitete 128 KB in z zunanjim pomnilnikom 1 MB (disketna enota) in 10 MB (diskovna enota).

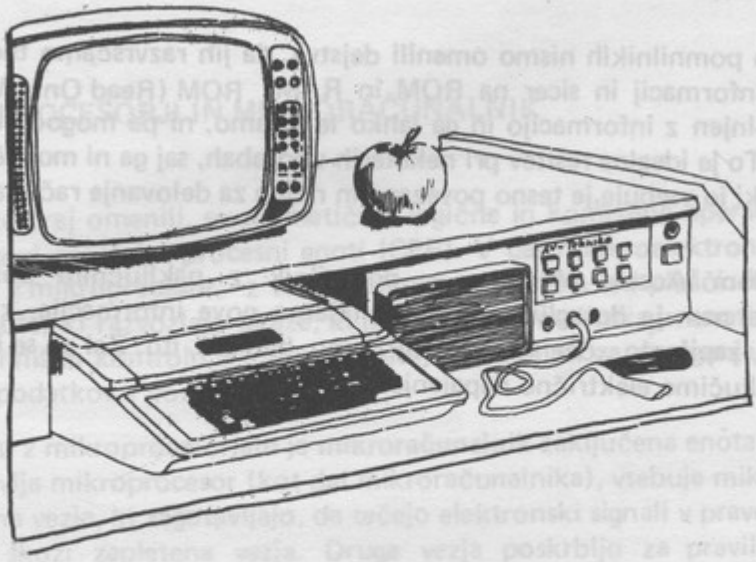
V poglavju o pomnilnikih nismo omenili dejstva, da jih razvrščamo tudi glede na dosegljivost informacij in sicer na ROM in RAM. ROM (Read Only Memory) je vnaprej napolnjen z informacijo in ga lahko le čitamo, ni pa mogoče informacije spreminjati. To je idealna rešitev pri nekaterih uporabah, saj ga ni mogoče izbrisati. Informacija, ki jo vsebuje,je tesno povezana in nujna za delovanje računalnika.

RAM (Random Access Memory – pomnilnik z naključnim dosegom), je pomnilnik, ki nam je dosegljiv, saj vanj vnašamo nove informacije, ki jih lahko spreminjamo, zapišemo „čez stare" – zbrišemo, beremo, itd. Žal pa se informacija zbriše, če izključimo električno napajanje.



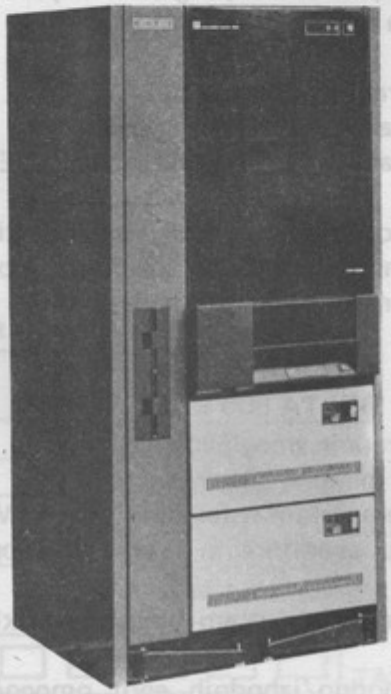
Slika 21: Dve vrsti spomina: RAM in ROM. Če uporabimo že uporabljeni model omarice (npr. za pisma), vidimo, da je RAM odprta omarica, v katere predalčke lahko vlagamo in jemljemo lističe z informacijami, kolikor nas je volja. ROM pa je takšna omarica, ki ima že vnešene lističe in vse skupaj zaprto – prekrito s steklom – lahko vidimo kaj je notri, vzeti ali zamenjati pa ne moremo nobenega od lističev.

Ob obeh naštetih vrstah pomnilnika omenimo še izpeljanke. ROM (Programmable Read Only Memory) je posebna vrsta ROM-a, kjer lahko uporabnik enkrat zapiše informacijo, za razliko od ROM-a, kjer se že pri proizvodnji vnesejo vse informacije. Nadaljnji razvoj je dal EPROM (Erasable Programmable Read Only Memory) – pomnilnik, ki ga normalno lahko le čitamo, vendar ga uporabnik večkrat lahko programira, t.j. spremeni informacijo.



Slika 22: Mikroračunalniški sistem APPLE

Kapaciteto pomnilnikov merimo v bitih. Število bitov, ki jih lahko spravijo, je približno enako polovici števila komponent. RAM ploščica, ki ima kapaciteto 65.536 bitov, zahteva okrog 132 000 komponent. Zaradi velikih števil in zaradi dvojiškega sistema, ki ga uporabljamo v računalnikih, je osnovna enota postal „KILO BIT" /1KB. Natančneje, 1 KB je 2 na potenco 10 oziroma 1024 bitov. Leta 1980 je več podjetij izdelalo 64 KB RAM ploščice. 256 KB RAM na eni ploščici lahko pričakujemo v bližnji prihodnosti. To bo seveda podobno povečanje gostote komponent in oznanilo začetek obdobja zelo visoke integracije!



Slika 23: Miniračunalniški sistem DELTA 800

Mikroračunalniki in mikroprocesorji se torej razvrščajo glede na število bitov v podatku, ki ga obdelujemo. Osem bitni mikroprocesor je omejen s količino informacij, ki jih lahko obdela z enim ukazom. To je osem bitov oziroma števila od 0 do 256 (2 na 8). Šestnajsti bitni mikroprocesor ima mnogo večje zmogljivosti in se lahko v isti časovni enoti ukvarja s števili (2 na 16=65 536) do približno 65 000, toda zahteva okrog 10 krat več komponent na ploščici.

Miniaturizacija je vplivala tudi na velike drage računalnike, ker jih čedalje pogosteje sestavljajo iz več skupin mikroprocesorjev Zato se je njihova zmogljivost povečala in uporaba energije zmanjšala. Često slišimo izraze miniračunalnik, makroračunalnik ali mikroračunalnik.

Kdaj lahko torej katerega izmed njih poimenujemo z določenim imenom? Zahvaljujoč izrednemu razvoju tehnike in tehnologije na področju miniaturizacije in integracije, torej mikroelektronike, se razlike med posameznimi vrstami, še posebej pa med mikro in miniračunalnikom, vse bolj zmanjšujejo. To je opazno prav sedaj, ko je za mikroračunalnike na voljo čedalje več pomožnih naprav in modularnih

dodatkov. Razlika pa vsekakor obstaja in to glede na zmožnosti, s katerimi računalniške enote razpolagajo. Če makroračunalnike označimo kot zelo „močne" naprave ogromnih kapacitet, s centralno procesorsko enoto 32, 64 ali z več biti, potem bi v manjše sisteme prištevali tiste, ki imajo vsekakor manjše kapacitete. Mikroračunalniki danes najpogosteje uporabljajo pomnilniško besedo s 16, ali z 18 biti, dočim je ta pri mikroračunalnikih 4 in 8, lahko pa tudi 16 bitov.

DELTA 800 predstavlja najnovejši primer izredno zmogljivega miniračunalnika in nov dosežek lastnega razvoja aparature in programske opreme v Iskri Delti.
Poglavitne značilnosti sistema so:
- modularnost,
- kompatibilnost z družino računalnikov DELTA,
- možnost povezave z računalniki DELTA,
- možnost povezave z računalniki drugih proizvajalcev (DEC, IBM, UNIVAC, itd.).

Sistem DELTA 800 je nov korak h tehnološki neodvisnosti: integriran s sistemsko programsko opremo, programskimi orodji ter številnimi uporabniškimi rešitvami za poslovno in procesno informatiko, je osnovni gradnik računalniško podprtih informacijskih sistemov.

Osnovne lastnosti SISTEMA DELTA 800 so:
- 16-bitni mini računalnik srednje zmogljivosti,
- razširitev pomnilnika do 4 milijone bytov,
- večuporabniško delo z operacijskim sistemom DELTA/M,
- kompatibilnost v pogledu aparature in programske opreme znotraj družine računalnikov DELTA,
- inteligentni mikroprocesorski podsistem omogoča priključitev disketnih enot ter različne komunikacije,
- velika izbira različnih vhodno/izhodnih enot omogoča velike možnosti za konfiguriranje sistema za različne namene uporabnika,
- operacijski sistem DELTA/M podpira programske jezike ASSEMBLER, COBOL, FORTRAN, BASIC in PASCAL,
- Možnost uporabe različnih programskih orodij IDA:
  - sistem za upravljanje podatkovnih struktur,
  - podatkovni slovar,
  - generator zaslonskih slik,
  - uporabniški programski generator,
  - generator programov, v cobolski izvirni kodi.

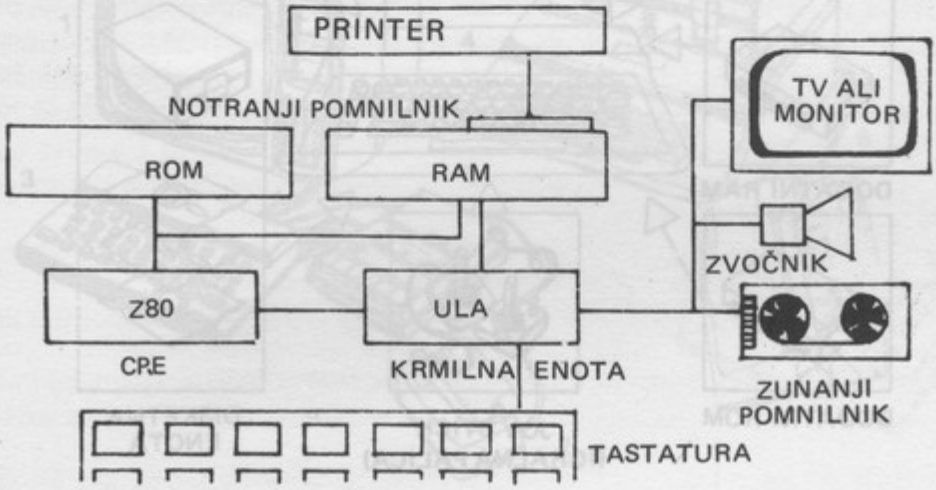
6. ANATOMIJA MIKRORAČUNALNIKA

Tako kot makroračunalniki so tudi mikroračunalniki sestavljeni iz dveh osnovnih komponent:

1. Osnovna oprema
   - procesne enote
   - vhodno–izhodne enote
   - zunanji pomnilnik

2. Dodatna oprema:
   - dodatni pomnilnik
   - tiskalnik
   - risalnik
   - svetlobno pisalo, igralna palica . . .

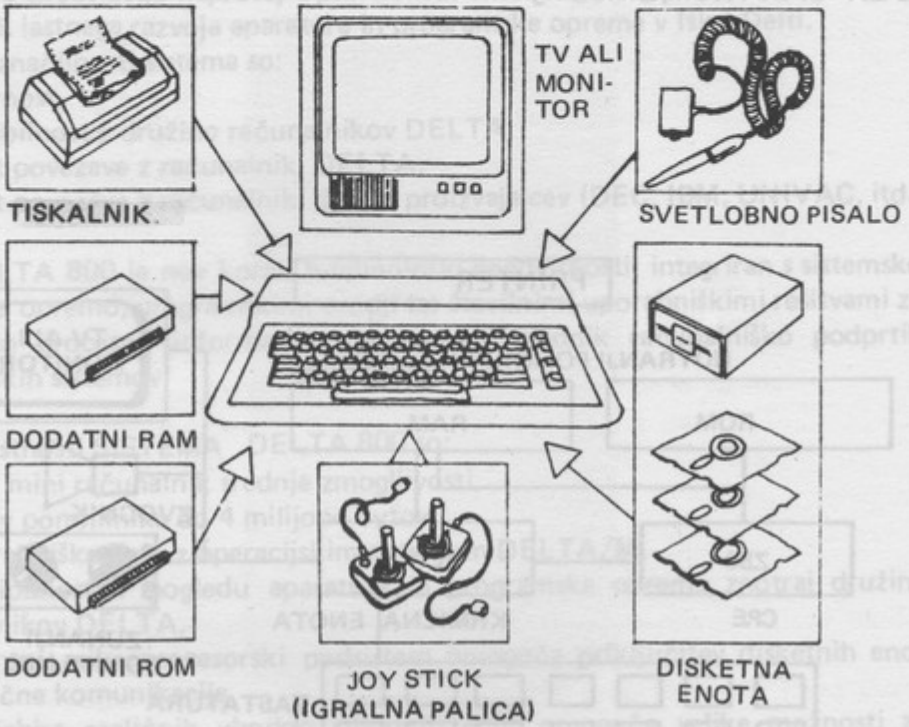
Oglejmo si te komponente na primeru verjetno najbolj razširjenega mikroračunalnika ZX–SPECTRUM, znane angleške firme SINCLAIR (slika 24).



Slika 24: Osnovne komponente mikroračunalnika ZX–SPECTRUM

O centralni procesni enoti, krmilni enoti in notranjem pomnilniku (ROM in RAM) ne bomo govorili še posebej na tem mestu, saj smo o tem že v predhodnih poglavjih precej povedali. Svetovali pa bi vam, da ne kupujete računalnika, ki ima ROM manjši od 10 KB in RAM manjši od 16 KB. Večina današnjih mikroračunalnikov ima ROM med 12 in 16 KB, kar zadostuje za izvajanje dokaj zahtevnih programov. Srečali pa boste tudi računalnike, ki imajo samo 2 KB RAM-a. Ti računalniki imajo v ROM-u spravljene le najosnovnejše programe, a pred začetkom programa morajo vse ostalo dobiti iz zunanjega pomnilnika (npr. kasetofona, minidiskovne

enote itd.) S stališča izkušenega uporabnika mikroračunalnikov je računalnik z malim ROM-om tudi dober, saj lahko po potrebi spreminja programski jezik, s katerim želi delati. Za začetnika ali občasnega uporabnika pa je dokaj neugodno, da mora vsakič, ko želi opravljati še tako osnovne operacije, le-te prečitati iz zunanjega pomnilnika.

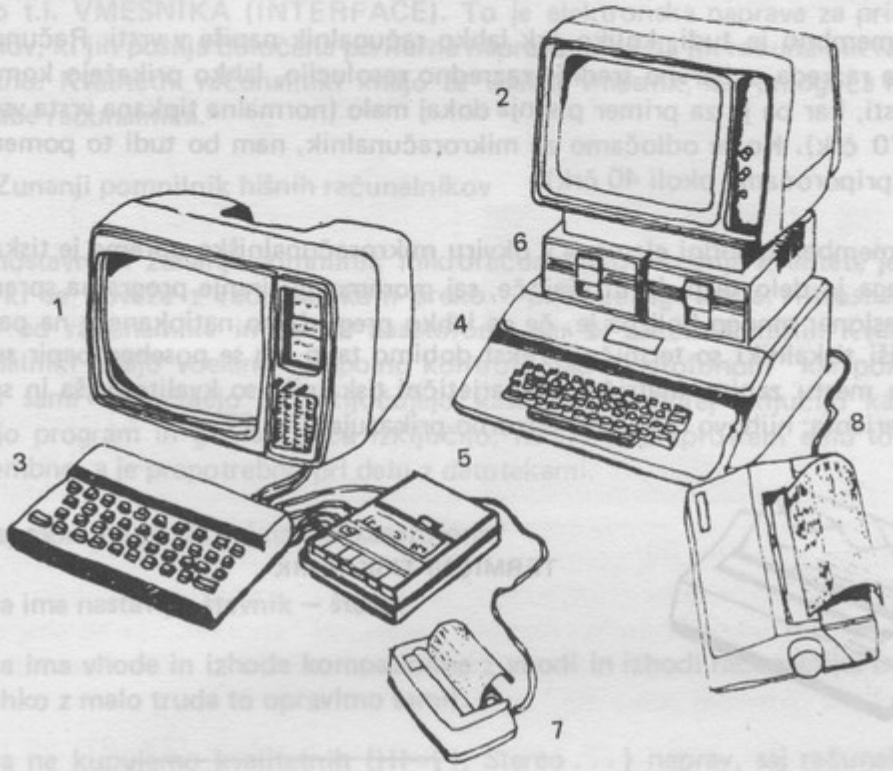


Slika 25: Mikroračunalnik s pomembnejšimi dodatnimi elementi in osnovno opremo.

6.1. Vhodno-izhodna enota

Tipkovnica je pri mikroračunalnikih in računalnikih nasploh pomemben element. Pri večini računalnikov so tipke razporejene po t.i. QWERTY shemi, kot je to na standardnem pisalnem stroju, seveda pa so nekaterim tipkam dodeljene še posebne funkcije. Boljši računalniki imajo več tipk (profesionalna tipkovnica), slabši pa imajo manjšo, dokaj nabito tipkovnico, na kateri ima vsaka tipka tudi več funkcij. Boljši računalniki imajo tipkovnico z mehaničnimi tipkami, manj kakovostni pa senzorske tipke, kot je to primer za ZX 81.

Med senzorsko in profesionalno tipkovnico je t.i. specializirana tipkovnica, ki jo ima ZX Spectrum. Je seveda mnogo boljša od senzorske, toda še vedno je majhnih dimenzij, tako, da imajo tipke po več funkcij. Tako za senzorsko kot tudi za specializirano tipkovnico je ugodno, če priključimo enostaven multivibrator, ki pri vsakem dotiku odda zvok določene frekvence. ZX Spectrum ima to možnost že vprogramirano.



Slika 26. Osnovna oprema mikroračunalnika:

1. TV sprejemnik – izhodna enota
2. monitor – izhodna enota
3. mikroračunalnik enostavnejše izvedbe (SPECTRUM SINCLAIR)
4. mikroračunalnik zahtevnejše izvedbe (APPLE)–centralna enota z vhodno enoto
5. kasetofon – zunanji pomnilnik
6. disketna enota – zunanji pomnilnik
7., 8. tiskalnik – izhodna enota.

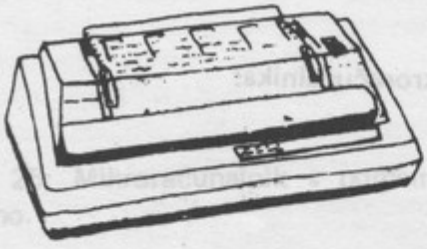
Večina mikroračunalnikov se povezuje s standardnim črnobelim ali z barvnim televizorjem, nekateri imajo monitorski izhod, drugi pa tudi že vstavljen monitor. Mikroračunalnik z vstavljenim monitorjem (slika monitorja je boljša in manj utrujajoča od TV) je seveda dokaj velika investicija, tako da se bolj izplača uporaba manjšega televizorja (npr. ISKRA MINIRAMA) in nakup tiskalnika.

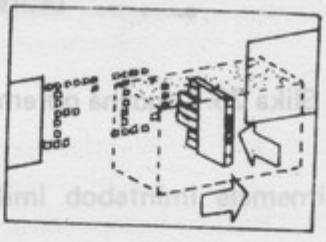
Pomembna karakteristika vsakega računalnika je t.i. grafična resolucija, kar pomeni število točk na zaslonu, ki jih lahko računalnik kontrolira. Vsako takšno točko lahko uporabnik s pomočjo določenega ukaza osvetli ali zatemni, s čimer riše določeno sliko na zaslonu – računalniška grafika. Če imamo računalnik z barvno resolucijo, dobimo na ta način barvno sliko. Slabši računalniki imajo grafiko nižje resolucije z okoli 3000 točk, srednji z okoli 50 000 točk, kvalitetni pa tudi čez 200 000. Seveda grafika in barve zasedajo kar precejšen del RAM-a.

Zelo pomembno je tudi, koliko črk lahko računalnik napiše v vrsti. Računalniki srednjega razreda, z barvno srednje razredno resolucijo, lahko prikažejo komaj 32 črk v vrsti, kar pa je za primer pisanja dokaj malo (normalna tipkana vrsta vsebuje 60 do 70 črk). Ko se odločamo za mikroračunalnik, nam bo tudi to pomemben kriterij (priporočamo okoli 40 črk).

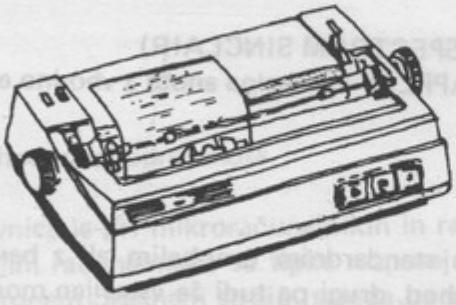
Zelo pomemben izhodni element v okviru mikroračunalniške opreme je tiskalnik. Brez njega je delo težje in utrujajoče, saj moramo testiranje programa spremljati preko zaslona; mnogo bolj pa je, če ga lahko pregledamo natipkanega na papirju. Najcenejši tiskalniki so termični. Tekst dobimo tako, da se poseben papir segreje samo na mestu zapisa. Matrični in marjetični tiskalniki so kvalitetnejša in seveda dražja varianta; njihovo delovanje nazorno prikazuje slika 27.

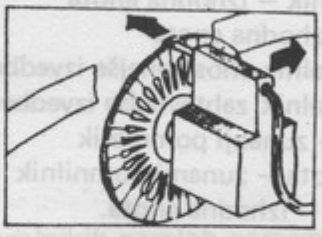
TERMIČNI TISKALNIK





MATRIČNI TISKALNIK





MAGNETNI TISKALNIK

Slika 27: Najpogostejše vrste tiskalnikov

Nekateri mikroračunalniki imajo tudi vdelan miniaturni zvočnik s tonskim generatorjem, ki lahko proizvaja tone nekaj oktav. Seveda ta karakteristika ni pomembna, toda zelo je prikladna za razna zvočna opozorila (npr. zvočni signal pri tipkanju, zvočni signal pri igrah, sinteza glasu in govora, itd.).

Za vsak računalnik lahko kupimo še dodatno periferno opremo, kot npr. risalnik, diskovno enoto, itd. Ta periferna oprema se lahko priključi na mikroračunalnik le preko t.i. VMESNIKA (INTERFACE). To je elektronska naprava za prilagajanje signalov, ki jih pošilja določena periferna naprava, tako da jih računalnik razume in obratno. Kvalitetni računalniki imajo že vdelan vmesnik, kar omogoča razširitev uporabe računalnika.

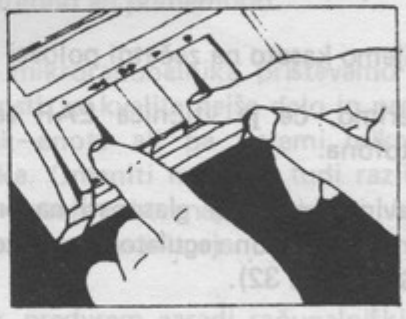
6.2. Zunanji pomnilnik hišnih računalnikov

Najenostavnejši zunanji pomnilnik mikroračunalnikov srednje kvalitete je kasetofon, ki se poveže z računalnikom preko priključnega kabla. Hitrost snemanja zavisi od računalnika in ne od kasetofona, kar bi začetnik mislil. Kvalitetnejši računalniki imajo vdelano „popolno kontrolo nad kasetofonom" kar pomeni, da lahko sami vključujejo in izključujejo kasetofon (najprej vključijo kasetofon, včitajo program in ga na koncu izključijo, itd.). Pri preprostem delu to ni tako pomembno, a je prepotrebno pri delu z datotekami.

Na kaj moramo paziti pri nabavi kasetofona?

1. Da ima nastavljiv števnik – števec
2. Da ima vhode in izhode kompatibilne z vhodi in izhodi računalnika (seveda lahko z malo truda to opravimo sami).
3. Da ne kupujemo kvalitetnih (HI–FI, Stereo . . .) naprav, saj računalnik zelo dobro dela na običajnem mono kasetofonu. Večji del kupljenih programov je snemanih za to vrsto kasetofonov.
4. Da ima vtičnici MIC in EAR (vhodno in izhodno, priključek za mikrofon in slušalke).

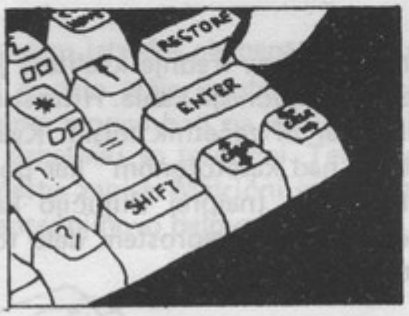
Slika 28



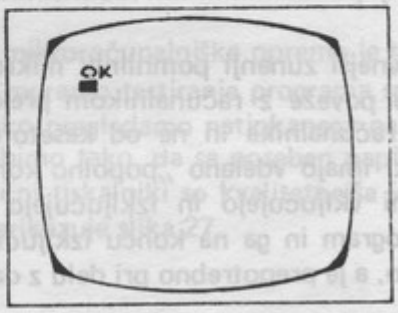
Slika 29

Običajen postopek pri snemanju programa na kaseto je naslednji:

1. Vzamemo čisto, nesnemano kaseto, lahko pa tudi tako, s katere lahko zbrišemo posneto vsebino (slika 28).
2. S pomočjo mikrofona snemamo na kaseto ime programa. To seveda ni obvezno, a nam kasneje pomaga pri iskanju programov. Računalnik spojimo s kasetofonom (priključek MIC kasetofona na priključek IZHOD računalnika)
3. Vtipkamo ukaz SAVE in ime programa (npr. SAVE „GOGI").
4. Vključimo (snemamo na srednji jakosti in srednji višini tonov) kasetofon na snemanje (slika 29).



Slika 30



Slika 31

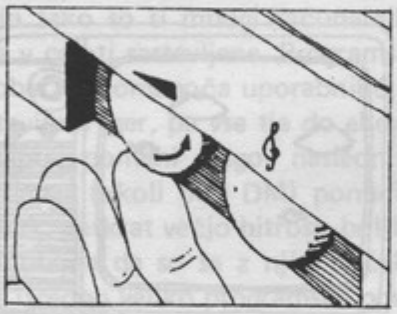
5. Pritisnemo tipko NEWLINE (ZX–81) ali ENTER (ZX–Spectrum) ali RETURN (HR–84) (slika 30).
6. Opazujemo TV zaslon, na katerem opazimo črnobele (ZX–81) ali barvne (ZX–Spectrum) proge. Ko na zaslonu dobimo izpis OK 0/0, izključimo kasetofon (slika 31).
7. Če imamo ukaz VERIFY, s katerim program verificiramo, to opravimo, drugače pa ga poskusno poženemo na računalniku!

Pomembnejši postopki pri snemanju programa iz kasete (iskanje) so:

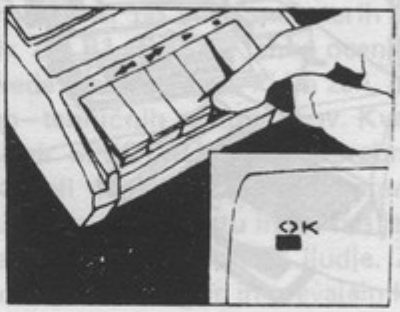
1. Previjemo kaseto na začetni položaj programa (slika 28).
2. Preverimo, če je vtičnica EAR računalnika spojena z vtičnico za slušalke kasetofona.
3. Postavimo regulator glasnosti na približno 1/1 do 2/3 maksimalne vrednosti. Če ima kasetofon regulator barve tonov, ga postavimo v položaj za visoke tone (HIGH) (slika 32).
4. Odtipkamo ukaz „LOAD" in ime (v kolikor poznamo ime programa) ali pa samo ukaz LOAD „ " – torej ukaz load in dva narekovaja (če ne poznamo imena programa).
5. Pritisnemo tipko NEWLINE ali ENTER (slika 30).

6. Vključimo kasetofon (slika 33).

7. Opazujemo TV zaslon.



Slika 32



Slika 33

Kvalitetnejši, a zato tudi mnogo dražji zunanji pomnilnik je disketna enota; v bistvu ne omogoča nič, kar ne bi bilo mogoče tudi s kasetofonom, a svojo nalogo shranjevanja in iskanja opravlja mnogo hitreje in v večjem obsegu (100 KB, pa tudi več). To je izrednega pomena za tiste, ki uporabljajo mikroračunalnik tudi v poslovne namene. Medij za shranjevanje programov je tudi gibka disketa. To je gibka plošča, prevlečena s posebno občutljivim magnetnim materialom in razdeljena na t.i. „sektorje" in „steze". Na njih računalnik zelo hitro vpisuje in z njih čita podatke, pri čemer omogoča takorekoč trenuten pristop k želenim informacijam.

Za uporabnike ZX Spectrum je zanimiv še t.i. mikrodrive, ki predstavlja magnetni trak, ki se zaradi vstavljenega mehanizma vrti po načelu brezkončne zanke.

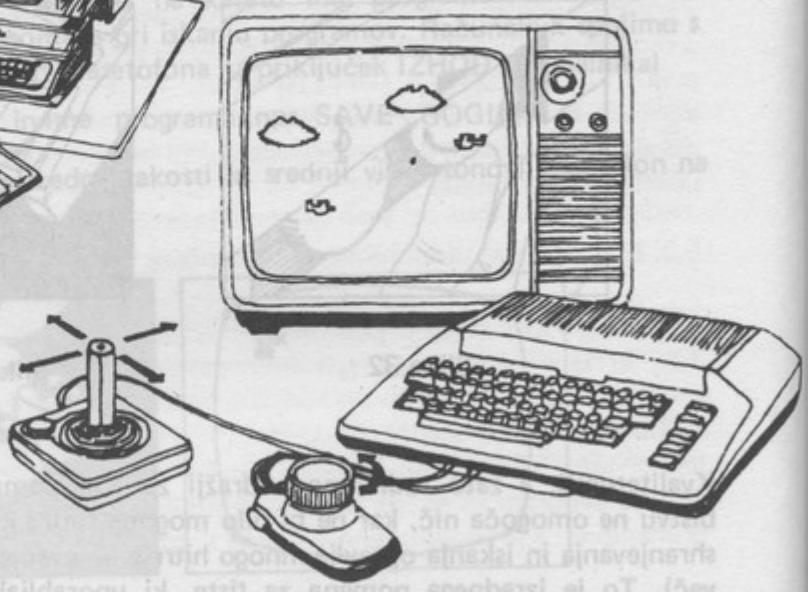
6.3. Dodatna oprema

Kot smo že omenili, prištevamo k mikroračunalniški dodatni opremi vse tisto, kar za nujno osnovno delovanje sistema ni potrebno ali pomembno.

Med najpomembnejšo dodatno opremo mikroračunalnika prištevamo tiskalnik. Brez njega uporabnik nima velike možnosti za kvalitetnejše delo in predvsem za širšo uporabo računalnika. Sledijo disk–enote ali pa sistemi mikrodrive za razširitev kapacitete zunanjega pomnilnika. Omeniti moramo tudi različne vrste grafičnih tabel (slika 34), ki omogoča to risanje zahtevnejših risb v eni barvi ali pa večbarvno, direktno na zaslon TV sprejemnika ali monitorja.

Za tiste, ki uporabljajo mikroračunalnik predvsem zaradi računalniških iger, je zanimiva igralna palica (JOY STICK – slika 35), to je gibljivo vodilo – stiskalo, ki omogoča kontrolo – vodenje objektov na zaslonu.

Slika 34



Slika 35

V šolske namene je zelo uporaben dodatek k mikroračunalniku t.i. svetlobno pisalo. S pomočjo njega lahko rišemo različne slike, ki bodo registrirane v pomnilniku računalnika, ne da bi se morali truditi, da z ukazi računalniške grafike opravimo isto.

K zelo koristni komunikacijski opremi spada tudi **modem**. To je naprava, ki več računalnikov (tudi mikroračunalnikov) poveže po telefonskih vodih v mrežo.

7. MIKRORAČUNALNIKI V SVETU IN PRI NAS

Hišni računalniki (home computers) so v bistvu mikroračunalniki za splošno rabo in z nekoliko manjšo kapaciteto, programsko in periferno opremo. Nekatere modele je mogoče vključiti in uporabiti brez velikih programskih izkušenj. Njihovo število močno raste, posebno še v tujini. Prvotno so bili namenjeni ljubiteljem elektronike, njihova uporabnost pa je tako velika, da so postali privlačni za mnogo širši trg. Do leta 1979 jih je bilo prodanih okoli 250 000 in podobno število v letu 1980 trend prodaje pa še vedno raste.

Prvi računalnik za domačo rabo, ALTAIR 8800, je bil v prodaji 1975. leta; prodajali so ga v obliki sestavnice, za manj kot 400 dolarjev. Hvalili so ga, da je zmogljivejši kot ENIAC. To je bilo verjetno res, saj je imel večji pomnilnik in je bil dvajsetkrat hitrejši. Od leta 1975 do danes sta bila hitrost širjenja in prirast zmogljivosti neverjetna.

Val osebnih računalnikov je prišel tudi do nas, seveda mnogo pozneje (v letu 1981-82). Prodor je opravil slavni SINCLAIR ZX 81 istoimenske angleške firme. Sicer dobra „centrala", 8 bitni mikroprocesor Z80A je bila odeta v škatlo s precej pomanjkljivostmi (senzorska tipkovnica, vezani ukazi, prepočasnost, itd.) a z izredno pomembno prednostjo: nizko ceno. Uvozni pogoji so v tistih časih bili še zmerni in tako so si mnogi računalnike naročali kar po pošti, nekaterih v delih, druga pa v celoti sestavljene. Programski jezik ZX 81 (BASIC) lahko ocenimo kot dokaj dober, saj omogoča uporabniku, da izvede vse programe, ki jih želi, in to od mnogoštevilnih iger, pa vse tja do ekonomsko–tehničnih predračunov. Kvaliteten korak naprej pomeni njegov naslednjih Sinclair SPECTRUM, saj uporabniku za zmeren denar (okoli 530 DM) ponudi 16 KB ali 48 KB notranjega pomnilnika, barve, zvok, petkrat večjo hitrost, boljšo tipkovnico itd. V svetu in pri nas je postal tako priljubljen da so se z njim začeli ukvarjati tudi sposobnejši ljudje. Zanj je napisane izredno veliko programske opreme, od čudovitih igric in prevajalnikov, do izobraževalnih, strokovno-tehničnih in ekonomskih programov. Velikemu številu ljudi, predvsem mladim, se je tako odprla možnost, da se spoznajo s poljubno množico zanimivih in težkih nalog.

Prav izziv velikim računalniškim sistemom šele prihaja iz proizvodnje Sinclair. To je SINCLAIR QL (Quantum Leap) izrednih zmogljivosti in hitrosti ter pomembno nizke cene (približno v vrednosti treh Spectrumov).

Seveda je na svetovnem trgu cela množica bolj ali manj kvalitetnih osebnih računalnikov. Tabela 1 v nadaljevanju prikazuje le najbolj poznane.

Tabela 1

Ime	ROM (KB)	RAM (KB)	osnovni program. jezik	grafika	približna cena (DM)
Apple I	12	48–128	BASIC in ostali	zelo dobra	2600
Atari 400	10	16	"	"	800
Commodore 64	20	64	"	"	1100
Dragon 32	16	32–64	"	slaba	990
Genie I in II	13	16–48	"	srednja	1100
Hewlett–Packard 85	32	16–120	"	zelo dobra	6500
Tewas TI 99/4 A	12	16–32	"	srednja	525
TRS 80 I	12	4–48	"	"	1100
TRS 80 color	8	4–32	"	zelo dobra	870
Sinclair ZX 81	8	1	"	slaba	179
Sinclair ZX 81	8	16	"	"	249
Sinclair Spectrum	16	16	"	zelo dobra	428
Sinclair Spectrum	16	48	"	"	575
Sinclair QL	32	138	"	"	1700

Tudi domači proizvajalci hišnih računalnikov so se zbudili iz „zimskega sna", tako da je v tem trenutku, ko pišemo to knjižico, obstajajo na našem tržišču (bolje rečeno pri naših proizvajalcih – saj jih je na trgu izredno malo) že nekateri osebni računalniki. Zanje je značilna visoka cena, v večini kvalitetna profesionalna tipkovnica z dokaj ugodnimi karakteristikami. Vsi uporabljajo programski jezik BASIC in so med seboj nekopatibilni, kar pomeni, da programov ni moč uporabiti medsebojno brez večjih korekcij.

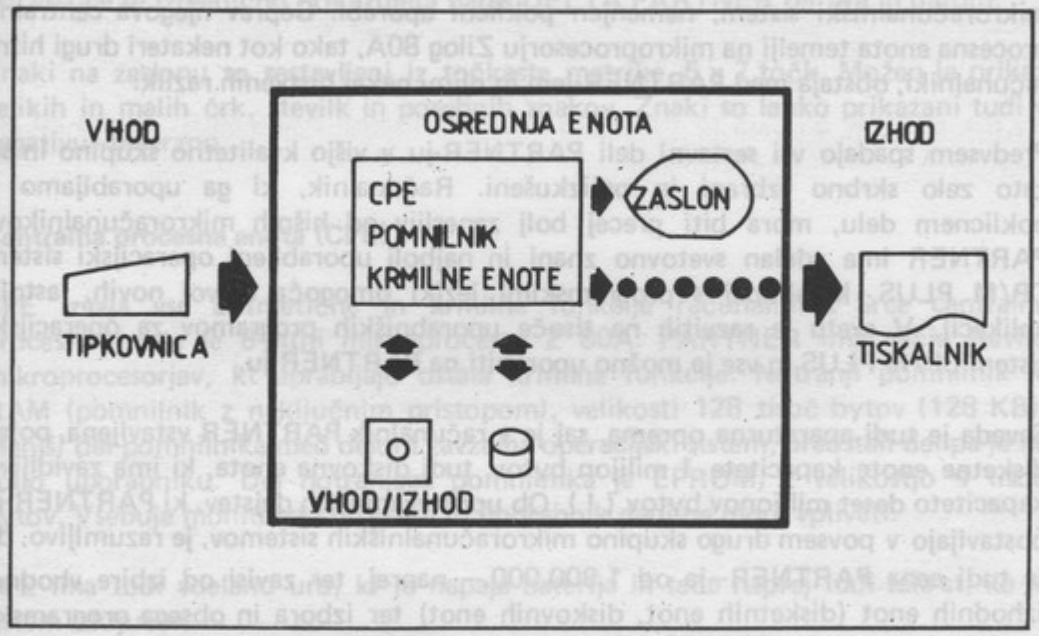
8. MIKRORAČUNALNIŠKI SISTEM PARTNER

8.1. Kratka predstavitev delovne organizacije ISKRA DELTA

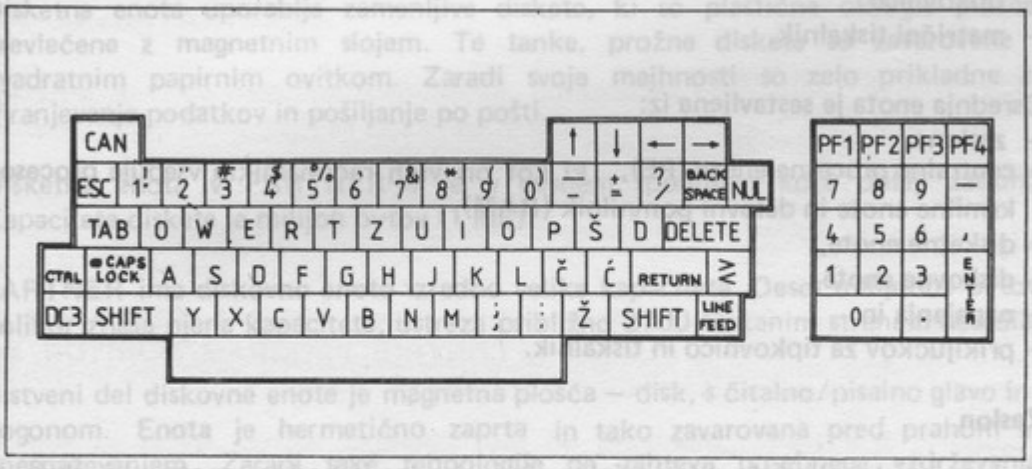
15. aprila 1982 so se delavci Elektrotehnine delovne organizacije DELTA (ustanovljene 1977) in TOZD Računalniki iz Iskrine delovne organizacije Telematika (ustanovljena 1977) združili v enovito delovno organizacijo ISKRA DELTA v okviru SOZD ISKRA, ki se uvršča na prvo mesto v elektroindustriji v Jugoslaviji. Z združitvijo računalniških potencialov in z ustanovitvijo nove delovne organizacije ISKRA DELTA, stopa razvoj, proizvodnja, trženje in vzdrževanje računalniških sistemov DELTA na nova pota.

Moč in rezultati te delovne organizacije so v strokovnih in kreativnih kadrih. V DO ISKRA DELTA je danes zaposlenih 1000 delavcev, ki jih odlikuje visoka kvalifikacijska struktura. ISKRA DELTA intenzivno goji in pospešuje razvojno-raziskovalno dejavnost; na tem področju dela okoli 300 strokovnjakov.

Osnovni koncept DO ISKRA DELTA sloni na proizvodnji računalniških sistemov in perifernih enot z maksimalno uporabo domače tehnologije in znanja, ter na osnovi najnovejših svetovnih dosežkov na tem področju. ISKRA DELTA načrtuje in uvaja poleg aparaturne opreme tudi pripadajočo sistemsko in aplikacijsko programsko opremo.



Shema mikroračunalnika PARTNER



Tipkovnice z jugoslovanskimi znaki

8.2. ISKRA DELTA je aprila 1983 predstavila mikroračunalniški sistem PARTNER, kot rezultat lastnega, domačega razvoja. PARTNER je sodoben osebni mikroračunalniški sistem, namenjen poklicni uporabi. Čeprav njegova centralna procesna enota temelji na mikroprocesorju Zilog 80A, tako kot nekateri drugi hišni računalniki, obstaja med PARTNER-jem in njimi nekaj bistvenih razlik.

Predvsem spadajo vsi sestavni deli PARTNER-ju v višjo kvalitetno skupino in so zato zelo skrbno izbrani in preizkušeni. Računalnik, ki ga uporabljamo v poklicnem delu, mora biti precej bolj zanesljiv od hišnih mikroračunalnikov. PARTNER ima vdelan svetovno znani in najbolj uporabljeni operacijski sistem CP/M PLUS, ki skupaj s programskimi jeziki omogoča razvoj novih, lastnih aplikacij. V svetu je razvitih na tisoče uporabniških programov za operacijski sistem CP/M PLUS in vse je možno uporabiti na PARTNER-ju.

Seveda je tudi aparaturna oprema, saj je v računalnik PARTNER vstavljena, poleg disketne enote kapacitete 1 milijon bytov, tudi diskovna enota, ki ima zavidljivo kapaciteto deset milijonov bytov (!). Ob upoštevanju teh dejstev, ki PARTNER-ja postavljajo v povsem drugo skupino mikroračunalniških sistemov, je razumljivo, da je tudi cena PARTNER–ja od 1.900.000.– naprej, ter zavisi od izbire vhodno-izhodnih enot (disketnih enot, diskovnih enot) ter izbora in obsega programske opreme (cene so iz julija 1984).

8.3. Zgradba PARTNER-ja

Sistem PARTNER je sestavljen iz aparaturne opreme, (hardware) in programske opreme (software).

Aparaturna oprema obsega:
- osrednjo enoto z zaslonom,
- tipkovnico,
- matrični tiskalnik.

Osrednja enota je sestavljena iz:
- zaslona,
- centralne procesne enote (PE), ki kot pri vseh računalnikih vsebuje procesor, krmilne enote in delovni pomnilnik (RAM),
- disketne enote,
- diskovne enote,
- napajanja in
- priključkov za tipkovnico in tiskalnik.

Zaslon

Zaslon predstavlja skupaj s tipkovnico vhodno/izhodno enoto, ki služi za komuniciranje med človekom in računalnikom. Zaslon omogoča interaktivno delo, kar pomeni, da je uporabnik v neposrednem dialogu s PARTNER-jem.

Diagonalna zaslona znaša 31 cm. Zaslon je prevlečen s slojem zelenega fosforja in je nebleščeč. Intenziteta osvetlitve je nastavljiva z gumbom na zadnji strani ohišja.

Uporabnik lahko uporablja 24 vrstic s po osemdeset znakov. 25. vrstica je prazna, medtem pa 26. vrstica prikazuje sistemska obvestila računalnika. Na desni strani 26. vrstice se izmenično prikazujeta napis DELTA PARTNER ter ura in datum.

Znaki na zaslonu so sestavljeni iz točkaste matrike, 5 x 7 točk. Možen je prikaz velikih in malih črk, številk in posebnih znakov. Znaki so lahko prikazani tudi v negativu—inverzno.

Centralna procesna enota (CPE)

CPE izvaja vse aritmetične in krmilne funkcije računalnika. Srce centralne procesne enote je 8-bitni mikroprocesor Z 80A. PARTNER ima večje število mikroprocesorjev, ki uprabljajo ostale krmilne funkcije. Notranji pomnilnik je RAM (pomnilnik z naključnim pristopom), velikosti 128 tisoč bytov (128 KB). Manjši del pomnilnika med delom zavzema operacijski sistem, preostali del pa je na voljo uporabniku. Del notranjega pomnilnika je EPROM, z velikostjo 4 tisoč bytov. Vsebuje monitorski program in uporabnik nanj ne more vplivati.

CPE ima tudi vdelano uro, ki jo napaja baterija in teče naprej tudi takrat, ko je sistem izključen.

Zunanji pomnilniki

Uporabniki mikroračunalnika PARTNER se pri svojem delu srečajo z dvema vrstama zunanjih pomnilnikov.

Disketna enota

Disketna enota uporablja zamenljive diskete, ki so plastične okrogle plošče, prevlečene z magnetnim slojem. Te tanke, prožne diskete so zavarovane z kvadratnim papirnim ovitkom. Zaradi svoje majhnosti so zelo prikladne za shranjevanje podatkov in pošiljanje po pošti.

Disketna enota v PARTNER-ju je v desnem spodnjem kotu poleg zaslona. Kapaciteta diskete je milijon bytov (1 MB).

PARTNER ima **diskovno enoto** izredno velike kapacitete. Deset milijonov bytov, kolikor znaša njena kapaciteta, ustreza približno 2700 tipkanim stranem besedila.

Bistveni del diskovne enote je magnetna plošča – disk, s čitalno/pisalno glavo in s pogonom. Enota je hermetično zaprta in tako zavarovana pred prahom in onesnaževanjem. Zaradi take tehnologije ne zahteva posebnega vzdrževanja, uporabnikovi posegi v njeno notranjost pa so sploh prepovedani. Diskovna enota nima zamenljivih diskov, zato podatke in programe, ki jih želimo shraniti ali prenesti drugam, najprej prepišemo z diska na disketo. Ker se disk stalno vrti z visoko hitrostjo, je čas pristopa do podatkov zelo kratek – 85 ms.

Vmesniki za vhodno/izhodne naprave

Tiskalnik priključimo preko vmesnika, ki ustrezna znanemu standardu RS 232 C. Nanj lahko priključimo matrični tiskalnik TRS 835, proizvod TVORNICE RAČUNSKIH STROJEVA iz Zagreba.

ISKRA DELTA prodaja te tiskalnike skupaj s sistemi PARTNER. Lahko pa na računalnik priključimo tudi katerikoli drug tiskalnik, ki ustreza standardu RS 232 C.

Tipkovnica

Tipkovnica je ločena od sistemske enote in z njo povezana s prožnim kablom, in omogoča, da si tipkovnico namestimo v najugodnejši delovni položaj. Tipke so zasnovane ergonomsko in so nebleščeče. Njihova razporeditev je QWERTZ, z jugoslovanskimi črkami Č, Ć, Š, Đ in Ž.

Manjšo skupino tipk vključuje numerične tipke in štiri funkcijske tipke (slika).

Tiskalnik TRS 835

Matrični tiskalnik TRS 835 je izhodna enota za izpis na papir. Hitrost pisanja je 180 znakov/s. Izpis je matrične oblike, vsak znak je sestavljen iz matrike 9 x 9 točk. Mehanski del tiskalnika sestavljajo ohišje, pokrov, vodilo papirja, valj, glava za pisanje in regulator števila kopij.

V razvijajoči informatiki predstavljajo pomemben del tudi povezave – komunikacije med računalniki.

Posebna izvedba takega sistema je PARTNER C, ki v računalniški mreži opravlja funkcijo inteligentnega terminala. PARTNER C se uporablja kot terminal ali kot lokalno delovno mesto.

Programska oprema obsega:
- operacijski sistem,
- sistemsko programsko opremo in
- aplikacijsko programsko opremo.

Operacijski sistem

Operacijski sistem CP/M PLUS je izdelek firme DIGITAL RESEARCH iz Californije, ZDA. CP/M je okrajšava za CONTROL PROGRAM/MONITOR, kar pomeni v bistvu kontrolno–monitorski program.

Operacijski sistem CP/M PLUS je nastal kot nadaljevanje razvoja operacijskega sistema CP/M iz leta 1973. Zaradi množične uporabe je postal standard na področju mikroračunalnikov.

CP/M PLUS je operacijski sistem visoke zmogljivosti, namenjen enouporabniškem delu. To pomeni, da na sistem dela istočasno le en program enega uporabnika. Namenjen je mikroračunalnikom, ki imajo vdelane mikroprocesorje Intel 8080 ali ZILOG Z80.

Sistem PARTNER je sestavljen iz številnih, medsebojno odvisnih elektronskih sklopov in programov, ki morajo delovati vsklajeno. Operacijski sistem opravlja nalogo vsklajevanja med delovanjem in vodenjem mikroračunalniškega sistema kot celote.

Operacijski sistem skrbi tudi za ustrezno zaporedje izvajanja programov v računalniku in vodi dialog z uporabnikom preko tipkovnice in zaslona. Vse do takrat, ko vključimo v delo uporabniške programe.

Sistemska programska oprema

Sistemsko programsko opremo sestavljajo prevajalniki za programske jezike: BASIC, FORTRAN, PASCAL in COBOL, programi za upravljanje podatkov in podatkovnih zbirk ter pomožni sistemski programi. Posebno skupino tvorijo programska orodja; to so programi, ki služijo kot pomoč programerju pri delu z računalnikom, pri razvoju programov in izdelavi dokumentacije v zvezi s tem.

Aplikacijska (uporabniška) programska oprema

Izvajanje in upraba aplikacijskih programov je glavni cilj, zaradi katerega imamo računalnik. Vse prej opisana in našteta aparaturna ter sistemska programska oprema služi temu, da lahko mikroračunalniški sistem PARTNER opravlja vlogo človekovega pomočnika v pisarni, skladišču, tovarni, konstrukcijskem biroju, izobraževalni ustanovi in drugod.

8.4. Uporaba PARTNER-ja

Navajeni smo, da nam mikroračunalnik, posebno če je to hišni mikroračunalnik razreda SPECTRUM, služi za igro in razvedrilo.

Slišali smo tudi o uporabi v izobraževalne namene, veliko pa se govori o poslovnih obdelavah, posebno v primeru osebnega mikroračunalnika velikih zmogljivosti, kot je PARTNER.

Uporaba poslovnih uporabniških programov nudi uporabniku hiter in enostaven dostop do podatkov. Klasične kartoteke nadomeščajo pregledi na zaslonu.

Seveda pa lahko kartico izpišemo tudi na tiskalniku. Uporaba je prilagojena specifičnim uporabnikom.

K enostavnosti pomagajo navodila, ki se sproti pojavljajo na zaslonu. Prednost je tudi ta, da obdelava podatkov poteka na mestih, kjer podatki nastajajo, to je v računovodstvu, skladišču, ipd.

Napake, ki so včasih nastajale pri prenosu podatkov in obdelavah v oddaljenih centralnih računalniških centrih, klasičnih ERC-ih, tako odpadejo.

Vendar je daleč najpomembnejša uporaba mikroračunalnikov v vodenju idustrijskih procesov, kjer njihova uporaba pomeni prihranek energije in surovin.

Tu pa je lahko računalnik močno orožje v bitki za stabilizacijo gospodarstva. V ISKRI DELTI so, zavedajoč se odgovornosti v znanstveno-tehnološki revoluciji, že pred leti organizirali močno skupino strokovnjakov za tehnično procesno informatiko. Rezultat teh prizadevanj je razvita programska oprema za vodenje procesov z lastnimi mikro in miniračunalniki DELTA na področjih:

- energetike
- kemije in farmacije
- prehrambene industrije
- lesne, papirne in tekstilne industrije
- radarske meteorologije
- računalniške grafike.

Izmed številnih uporabnih rešitev naj omenimo le nekatere, vezane na sistem PARTNER.

Mikroračunalniški dozirno–nadzorni sistem – MIDOS je namenjen za vodenje in nadzor procesov doziranja in mešanja v prehrambeni in kemijski industriji, farmaciji ter gradbeništvu.

MIDOS omogoča vodenje doziranja v mešalnicah, ki imajo do štiri dozirne tehtnice, dva mešalca in sto dozirnih celic.

Pomembna lastnost MIDOS-a je stalen nadzor vseh elementov delovnega procesa. V primeru okvare programa takoj izpiše opozorilo na zaslonu in tiskalniku; v primeru težjih okvar pa zaustavi proces.

SIKIP je programski proizvod za spremljanje in kontrolo industrijske proizvodnje. Poseben vmesnik omogoča priključitev do 1024 digitalnih vhodov, ki omogočajo spremljanje proizvodnje v pogledu količine in kvalitete izdelkov.

Sistem SVIDO omogoča vodenje prizvodnje tekoče hrane v prašičereji. SVIDO omogoča vodenje priprave tekoče hrane v mešalnici, vodenje distribucije tekoče hrane v objektih in obdelavo podatkov v zvezi s tem.

Mejno področje med čisto procesnimi in poslovnimi obdelavami je **obračun in vodenje proizvodnje.**

Programski proizvod PROTEKS–P na PARTNER-ju uporabljamo pri opremljanju in vodenju proizvodnje na področju, ki ni izključno računovodsko-finančne narave.

S spremljanjem zalog in prometa materiala planiramo potrebe po materialih, polizdelkih in izdelkih. Tak način dela omogoča prilagajanje proizvodnega sistema zahtevam tržišča.

Inženirski računalnik predstavlja uporabo PARTNER-ja v različnih vejah inženirskega dela in omogoča, poled administrativnih funkcij, predvsem inženirsko – znanstvena izračunavanja na področju:

- energetike,
- strojništva,
- lesne industrije,
- gradbeništva.

Obdelava besedila je ena osnovnih uporab mikroračunalnika v sodobnem poslovanju. Nadomešča nam pisalni stroj v pisarnah in drugje, vendar ima ob tem pred običajnim pisalnim strojem številne prednosti.

S programom za obdelavo besedil dosežemo večjo produktivnost pri pisarniškem poslovanju in večjo kvaliteto izdelkov, ki so lahko poslovni dopisi, tehnična dokumentacija, samoupravni akti in prevodi.

Besedila so shranjena na disketi ali disku in so hitro dosegljiva za ponovni vpis ali spremembe.

Obdelava besedil na PARTNER-ju je preprosta in enostavna. Besedilo vnašamo preko tipkovnice in ga sproti preverjamo na zaslonu.

Ker program sam začne novo vrstico, nam na to ni treba paziti, kar pomeni velik prihranek časa, zlasti pri daljših besedilih.

Bistveno pri takem načinu dela pa je popravljanje besedil, ki je zelo preprosto in hitro.

Besedilo lahko popravimo že ob vpisu ali pa kasneje pokličemo na zaslon katerikoli del besedila. Potem lahko brišemo, zamenjamo ali vnesemo na katerekoli mestu določeni znak ali večji del besedila.

Spremembe sproti opazujemo na zaslonu. Vnešeno besedilo lahko pred izpisom oblikujemo, glede na višino in širino strani, poravnavo levega ali desnega robu; centriranje besedil, naslovov ali opomb.

Veliko pomoč pomeni avtomatsko oštevilčenje strani.

Izpis na tiskalniku je pri takem načinu vedno brezhiben izdelek, saj smo napake odpravili že na zaslonu.

Tehnična kvaliteta izpisa je seveda odvisna od vrste tiskalnika.

Program za obdelavo besedil omogoča tudi obdelavo pisem.

V že pripravljeno besedilo pisma lahko vnašamo za vsako pismo posebej: naslov, ime in primek naslovljene osebe, tekoči datum in dopolnilni del besedila. Vsako pismo je tako prilagojeno individualnemu naslovniku in ima pečat osebnega pisma.

ISKRA DELTA nudi uporabnikom sistem PARTNER poleg opisanega programa za obdelavo besedil še številne že izdelane programe.

Eno izmed področij uporabe, kjer se je PARTNER že uveljavil zaradi svojih prednosti pred klasično obdelavo podatkov, so **poslovne obdelave**. Zajema področje glavne knjige, osnovnih sredstev, saldakontov kupcev in dobaviteljev, področja prodaje s fakturiranjem in skladiščnim poslovanjem.

Že izdelani programi, ki jih uporabnik lahko kupi pri ISKRI DELTI, skrajšajo čas, sicer potreben za izdelavo programov pri uporabniku, in omogočajo takojšnjo uporabo računalnika PARTNER.

ISKRA DELTA izdaja vsako leto katalog programskih proizvodov. Tako najdemo v aprilski izdaji leta 1984 še naslednje, že izdelane uporabniške programe za:

- obračun osebnih dohodkov,
- izračunavanje amortizacije,
- obdelava podatkov s področja potrošniških kreditov,
- avtomatizacija prodaje hotelskih kapacitet in recepcijsko poslovanje,
- enostavno generiranje risb (grafika),
- lekarniško poslovanje, itd.

V poslovne obdelave spada posebna izvedba PARTNER-ja v vlogi bančno-poštnega delovnega mesta.

Osnovna funkcija bančno-poštnega delovnega mesta je zajem podatkov in obdelava v zvezi z denarnim poslovanjem v bankah in poštah. Deluje kot samostojno delovno mesto, kjer obdelane podatke shranimo na disketi, mogoča pa je tudi povezava z glavnim sistemom.

8.5. Instalacija in vključitev

Mikroračunalniški sistem PARTNER položimo na primerno delovno mizo. Posamezne sestavne dele med seboj povežemo tako, da tipkovnico priključimo na zadnji del v okroglo priključnico z oznako TAST.

Če imamo v kupljenem kompletu tudi tiskalnik, ga priključimo na 25 – polno priključnico J 7 na zadnji strani. Poseben kabel mora biti priložen tiskalniku.

Nazadnje priključimo priključni kabel za napajanje z napetostjo 220 V/50 Hz. Priključni kabel je dobavljen skupaj z računalnikom in se vključi v vtičnico na zadnji strani, ob kateri sta tudi dve omrežni varovalki.

Poleg omenjenih priključkov so na zadnji strani ohišja sistemske enote tudi stikala za vklop/izklop, stikalo RESET in potenciometer za nastavitev osvetljenosti zaslona.

Tu so tudi dodatni priključki J 6, J 8 in J 9, ki niso del standardne verzije.

Pred vključitvijo kabla v omrežje preverimo, če je stikalo za vklop izključeno (v položaju Ø); preverimo, če sta priključena stikalnik in tipkovnica. Sistem vključimo s stikalom za vklop/izklop. Nato vklopimo tiskalnik. Izključujemo v obratnem vrstnem redu.

Ob vklopu sistema PARTNER na omrežno napetost se avtomatično izvede program (monitor), ki je zapisan v EPROMU.

Ta program najprej izvede testiranje pomnilnika RAM, nato pa prepis ("nalaganje") operacijskega sistema z diska v pomnilnik.

Ko je ta del operacije uspešno izveden, se po obvestilih o testiranju in proizvajalčevih sporočilih pojavi na zaslonu glavni MENU.

Menu nas usmerja v nadaljnje delo:

A – operacijski sistem
M – izvajanje aplikacij
F – formatiranje disket
D – ažuriranje sistemskega datuma

Po vtipkanju črke A pridemo v okrilje operacijskega sistema CP/M PLUS. To pomeni, da moramo od sedaj dalje uporabljati ukaze v skladu z navodili za uporabo operacijskega sistema.

Ta omogoča na PARTNER-ju tudi uporabo naslednjih programskih jezikov: BASIC, FORTRAN, PASCAL in COBOL.

Če preidemo v izvajanje uporabniških programov, ki smo jih seveda pred tem kupili ali izdelali sami, nas bo novi menu popeljal naprej.

8.6. Vzdrževanje in šolanje

Ob PARTNER-ju dobe kupci tudi uporabniške priročnike. V času garancije in potem imajo na voljo razvejano vzdrževalno službo na področju celotne Jugoslavije.

Vzdrževanje vključuje servis programske in aparaturne opreme.

ISKRA DELTA ima tudi lasten izobraževalni center, kjer se lahko naučite vse, kar potrebujete v svojem računalniškem poklicu.

Izobraževanje vodijo strokovnjaki, ki so svoje izkušnje pridobili v razvoju, proizvodnji in izobraževanju.

Ko smo pred več kot letom dni v tržnem komuniciranju pripravljali tiskovno konferenco ob predstavitvi novega PARTNER-ja, nismo vedeli, da bo ta isti PARTNER postal naš resnični sodelavec. Danes si ne moremo predstavljati dela brez njega, saj nam pomaga pri pripravi besedil za prospekte, priročnike, članke in prevode. Na PARTNER-ju opravljamo tudi razne analize dejavnosti v marketingu, pripravljamo programe in predračune za našo dejavnost.

Takrat nismo vedeli da se bo uresničil stavek, ki smo ga sami zapisali v prvi prospekt za mikroračunalniški sistem PARTNER:

„PARTNER BO VAŠ RESNIČEN SODELAVEC PRI DELU".

9. POGOVOR Z RAČUNALNIKOM

Kako pa se pogovarjamo z računalnikom, da ga razumemo mi, oziroma, da računalnik razume nas? Kot smo že povedali, reagira računalnik le na dva simbola (0 ali 1, oziroma DA ali NE, itd.), ki pa sta za komunikacijo, kot jo je navajen človek, dokaj nenavadna in nesprejemljiva. Zato bi bilo najbolje, če bi lahko naučili računalnike človeškega jezika. Tako daleč znanstveniki še niso, vendar pa so le dosegli precejšen napredek, saj so do danes sestavili na ducate različnih t.i. **programskih jezikov**, ki se jih je razmeroma lahko naučiti in ki jih razumejo tudi računalniki. Programski jeziki so torej človeku bližji, kot računalnikov strojni jezik, to pomeni, da so mu lažje razumljivi, obenem pa so tudi bliže strojnemu jeziku kot naravni jezik.

9.1. Programski jeziki

Računalnik lahko vodimo s posebno vrsto jezika, ki se imenuje **strojni jezik**. Računalnik bo izvršil določene naloge samo takrat, ko mu damo napotke (ukaze), prevedene na njegov strojni jezik. Za komuniciranje z računalnikom uporabljamo **simbolični jezik**, ki je sestavljen iz pojmovnih kratic. Te si zelo lahko zapomnimo, a obenem pa nam omogočajo, da se ti simboli v računalniku prevedejo v binarne kode, ki ustrezajo operacijskim kodam strojnega jezika.

Računalniki prvih generacij so uporabljali programe pisane v strojnem jeziku, kar je bilo za programerje, še posebej pa za druge uporabnike, zelo težak in zapleten posel. Danes se programi pišejo s **simboličnim – programskim jezikom**. Simbolični programski jezik se s pomočjo specialnih programov prevaja v strojni jezik, s katerim razpolaga računalnik. Simbolične jezike lahko delimo v dve skupini:

- nižji programski jeziki (orientirani na stroj)
- višji programski jeziki (orientirani na problem)

Za nižji programski jezik obstaja prevajalni program, ki se imenuje ASSEMBLER.

Vsak simbolični ukaz se v assemblerju prevaja v en strojni ukaz. Čeprav je tako dobljeni jezik (**zbirni jezik**) precej bolj razumljiv kot strojni jezik, pa je podajanje posameznih korakov programa še vedno dokaj zapleteno in tudi nepregledno.

Z združevanjem niza ukazov zbirnega jezika v človeku enostavneje, razumljive zapise so nastali višji programski jeziki, v katerih se en simbolični ukaz prevede v niz ukazov strojnega jezika. Njihovi prevajalni programi se imenujejo **prevajalniki (compilers)**. Prevajalni programi so v splošnem tem bolj zahtevni, čim manj je programski jezik podoben strojnemu jeziku. Ko torej damo program v nekem višjem programskem jeziku v računalnik, ga računalnik najprej prevede v svoj strojni jezik, nato ga lahko šele izvaja; za to svoje delo potrebuje pomnilni prostor in računalniški čas, kar pa odpade, če bi napisali program v strojnem jeziku. Prevajanje se izvede tako, da prevajalni program prečita naš program, ki smo ga že zapisali v pomnilnik, in na ustrezno mesto v pomnilniku zapiše prevod. Ta prevod je torej že zaporedje dejanskih operacij računalnika, po katerem bo rešil zadani problem.

V višje programske jezike spadajo med drugim naslednji jeziki (v oklepaju navajamo angleške besede, iz katerih je sestavljena kratica):

FORTRAN (Formula Translation) — programski jezik, namenjen za pisanje programov iz znanstveno-tehničnega področja uporabe računalnikov.

ALGOL (Algoritem Language) — je namenjen pravtako za pisanje programov iz znanstveno-tehničnega področja.

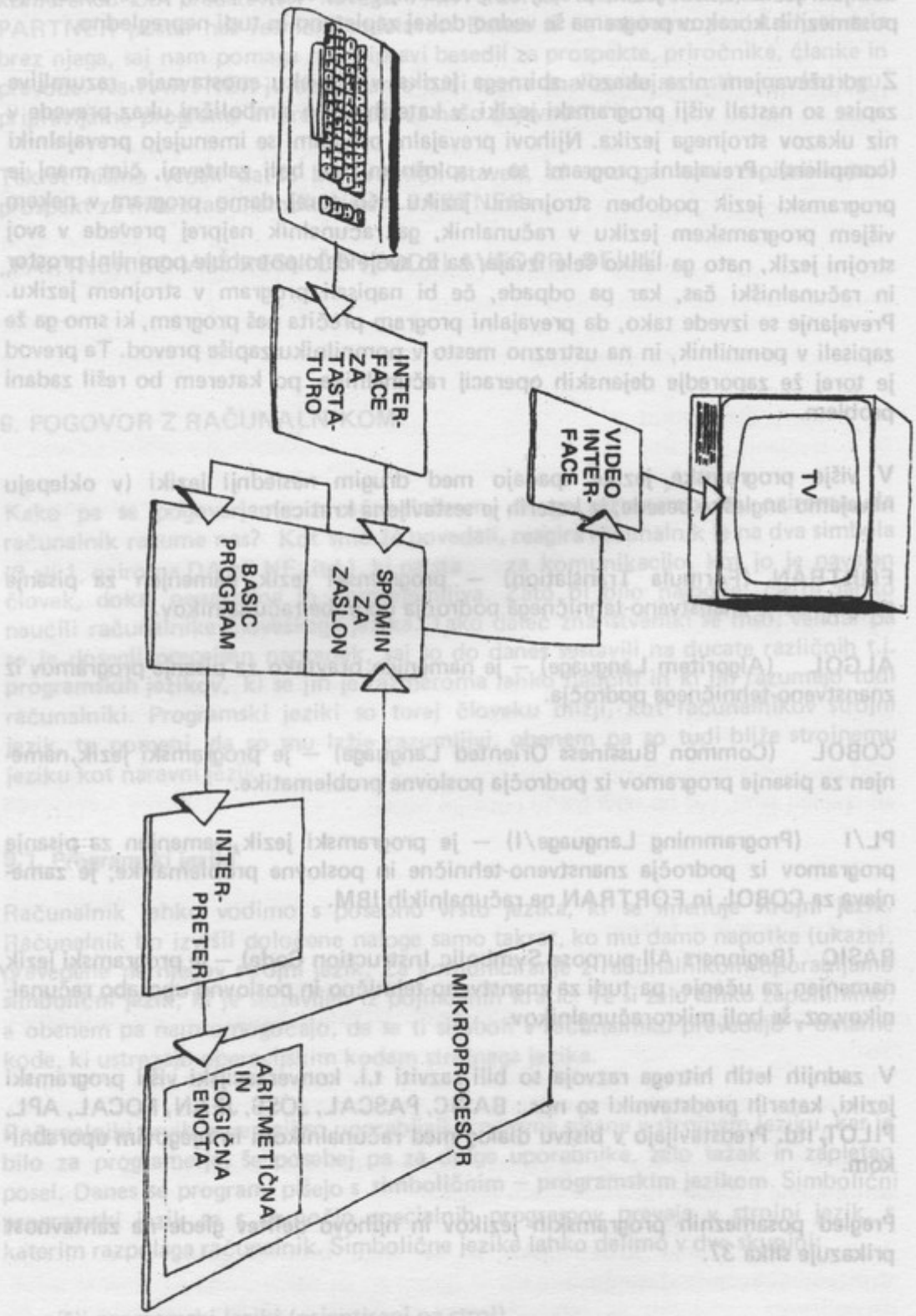
COBOL (Common Bussiness Oriented Language) — je programski jezik, namenjen za pisanje programov iz področja poslovne problematike.

PL/I (Programming Language/I) — je programski jezik, namenjen za pisanje programov iz področja znanstveno-tehnične in poslovne problematike; je zamenjava za COBOL in FORTRAN na računalnikih IBM.

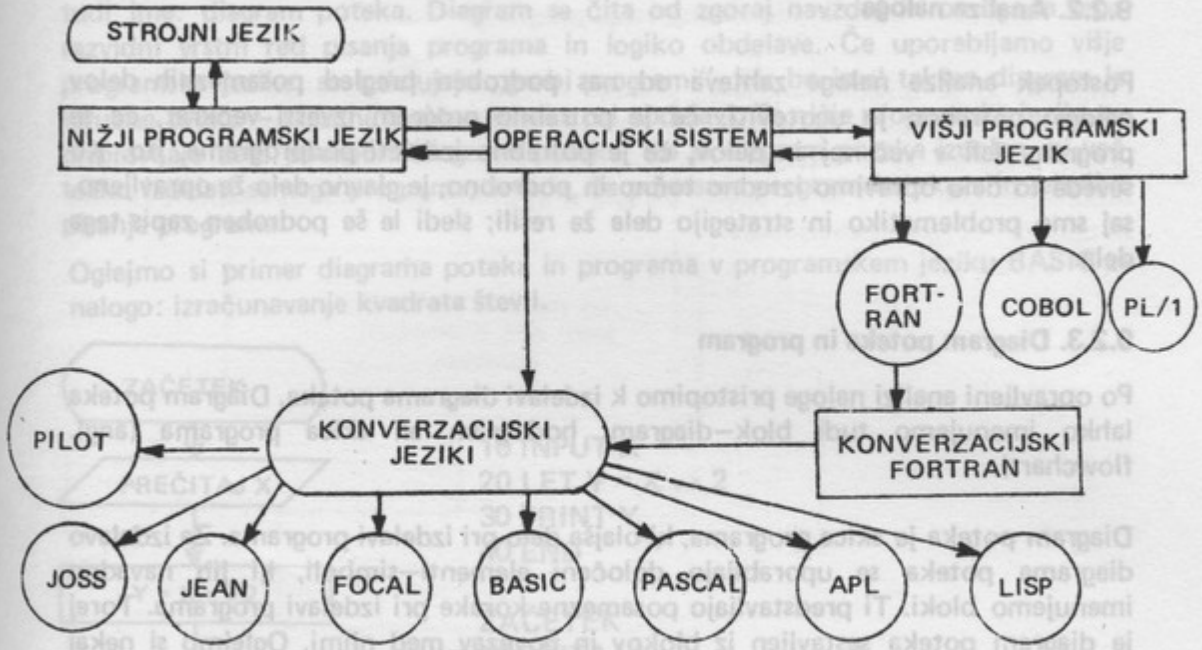
BASIC (Beginners All-purpose Symbolic Instruction Code) — je programski jezik, namenjen za učenje, pa tudi za znanstveno-tehnično in poslovno uporabo računalnikov, oz. še bolj mikroračunalnikov.

V zadnjih letih hitrega razvoja so bili razviti t.i. konverzacijski višji programski jeziki, katerih predstavniki so npr.: BASIC, PASCAL, JOSS, JEAN, FOCAL, APL, PILOT, itd. Predstavljajo v bistvu dialog med računalnikom in njegovim uporabnikom.

Pregled posameznih programskih jezikov in njihovo delitev glede na zahtevnost prikazuje slika 37.



Slika 36: Diagram poteka dela v mikroračunalniku



Slika 37: Pregled programskih jezikov

9.2. Še nekaj o tehniki programiranja

Najprej odgovorimo na vprašanje, kaj je **program**. Možen je kratek in hiter odgovor: program je niz logično povezanih ukazov. Lahko bi tudi rekli, da je program niz podatkov računalniku, kako obdelati določene podatke, da bo lahko rešil zadani problem. To je tudi enostavnejša definicija **programiranja**. Seveda pa v toku programiranja naletimo na več faz dela. Najprej je potrebno definirati nalogo, zatem izvršiti analizo naloge, potem izdelamo diagram poteka in program. Delo s programiranjem še ni gotovo, ker je potrebno izvršiti prevajanje, kontrolo, testiranje programa in šele nato lahko program varno shranimo na element zunanjega spomina.

Glede na vse to je za programiranje potrebno poznati simbolične ukaze, pravila pisanja programa, potrebno pa je znati nalogo tudi analizirati. Poskušajmo pobliže na kratko spoznati nekaj osnovnih elementov za vsako fazo dela pri programiranju.

9.2.1. Definiranje naloge

V toku definiranja naloge je potrebno točno določiti, kje bodo podatki nameščeni, ugotoviti za kakšne podatke gre, ali si je potrebno podatke zapomniti, ali je dovolj če opravljamo samo določene računske ali kakšne druge operacije, ter v kakšni obliki moramo formulirati rezultate posameznih podatkov.

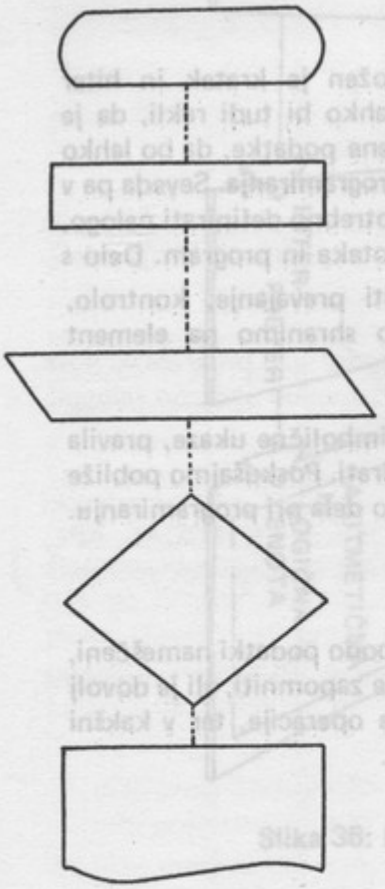
9.2.2. Analiza naloge

Postopek analize naloge zahteva od nas podroben pregled posameznih delov naloge; potrebno je ugotoviti, če je potrebno program izvesti večkrat, če se program deli v več vej in delov, če je potrebno izdelati podprograme, itd. Ko seveda to delo opravimo izredno točno in podrobno, je glavno delo že opravljeno, saj smo problematiko in strategijo dela že rešili; sledi le še podroben zapis tega dela.

9.2.3. Diagram poteka in program

Po opravljeni analizi naloge pristopimo k izdelavi diagrama poteka. Diagram poteka lahko imenujemo tudi blok–diagram, hodogram ali skica programa (angl. flowchart).

Diagram poteka je skica programa, ki olajša delo pri izdelavi programa. Za izdelavo diagrama poteka se uporabljajo določeni elementi–simboli, ki jih navadno imenujemo bloki. Ti predstavljajo posamezne korake pri izdelavi programa. Torej je diagram poteka sestavljen iz blokov in povezav med njimi. Oglejmo si nekaj blokov, ki so najčešče v uporabi:



Začetni ali končni blok, kot začetek ali konec programa.

Prireditveni blok, ki se uporablja za zapisovanje posameznih korakov v programu, oziroma je to oznaka operacije.

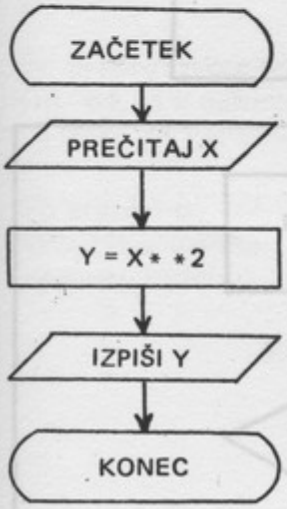
Blok za vhodno izhodne ukaze (pomeni npr.: preberi podatke in prebrane vrednosti priredi spremenljivkam, ki so vpisane v blok).

Odločitveni blok, v katerega vpisujemo pogoje, od tega, ali je pogoj izpolnjen ali ne, je odvisno, po kateri veji je treba izstopiti iz bloka.

Izpis na tiskalnik, ki pomeni, da naj tiskalnik izpiše vrednosti spremenljivk, ki so vpisane v tem bloku.

Vsi bloki so povezani s puščicami in zato lahko vidimo, kako program teče, odtod tudi ime: diagram poteka. Diagram se čita od zgoraj navzdol in omogoča lepo razvidni vrstni red pisanja programa in logiko obdelave. Če uporabljamo višje programske jezike, se izdelujejo „grobi programi", ker bo imel takšen diagram le ključne točke – ključne logične korake pri obdelavi. Za nižje programske jezike se pripravljajo zelo podrobni diagrami poteka. Ko je diagram poteka izdelan, ni več težko izdelati samega programa, seveda, če poznamo programske ukaze in tehniko pisanja programa.

Oglejmo si primer diagrama poteka in programa v programskem jeziku BASIC za nalogo: izračunavanje kvadrata števil.




```
10 INPUT X
20 LET Y = X ** 2
30 PRINT Y
40 END

ZAČETEK
PREČITAJ X
Y = X ** 2
IZPIŠI Y
KONEC
```

9.2.4. Vnos programa in kontrola

Program, ki smo ga po opisanem vrstnem redu pripravili, prenesemo na spominske elemente, ki so lahko kartice, papirni trak, itd. ali pa kar direktno na računalnik/ mikroračunalnik ali pa terminal/. Direkten prenos programa v računalnik je dokaj ekonomičen in hiter postopek, ki nam omogoča tudi popravljanje večjih programskih napak, saj nam jih običajno računalnik sam javi v postopku prevajanja ukazov v strojni jezik.

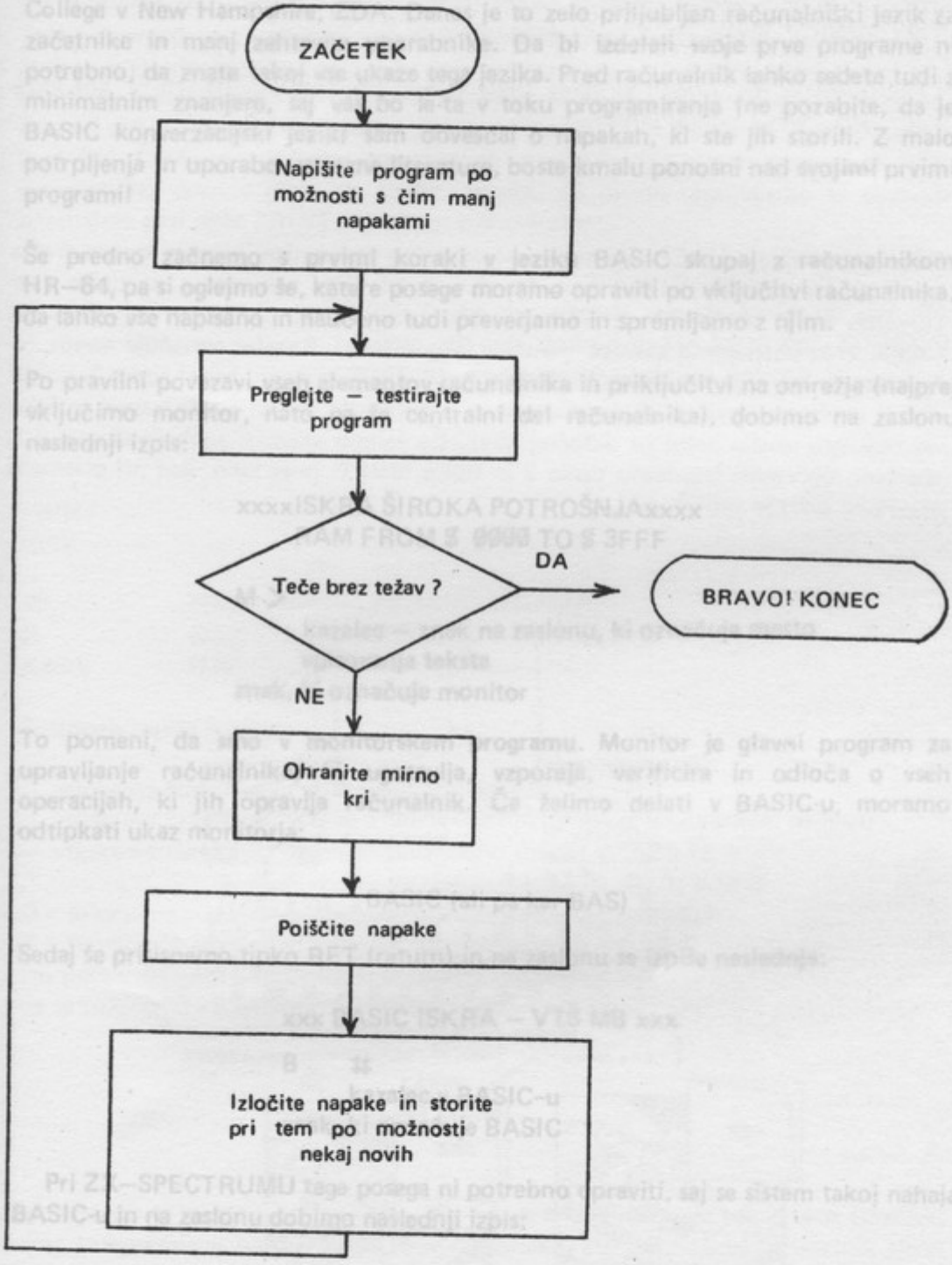
9.2.5. Testiranje programa

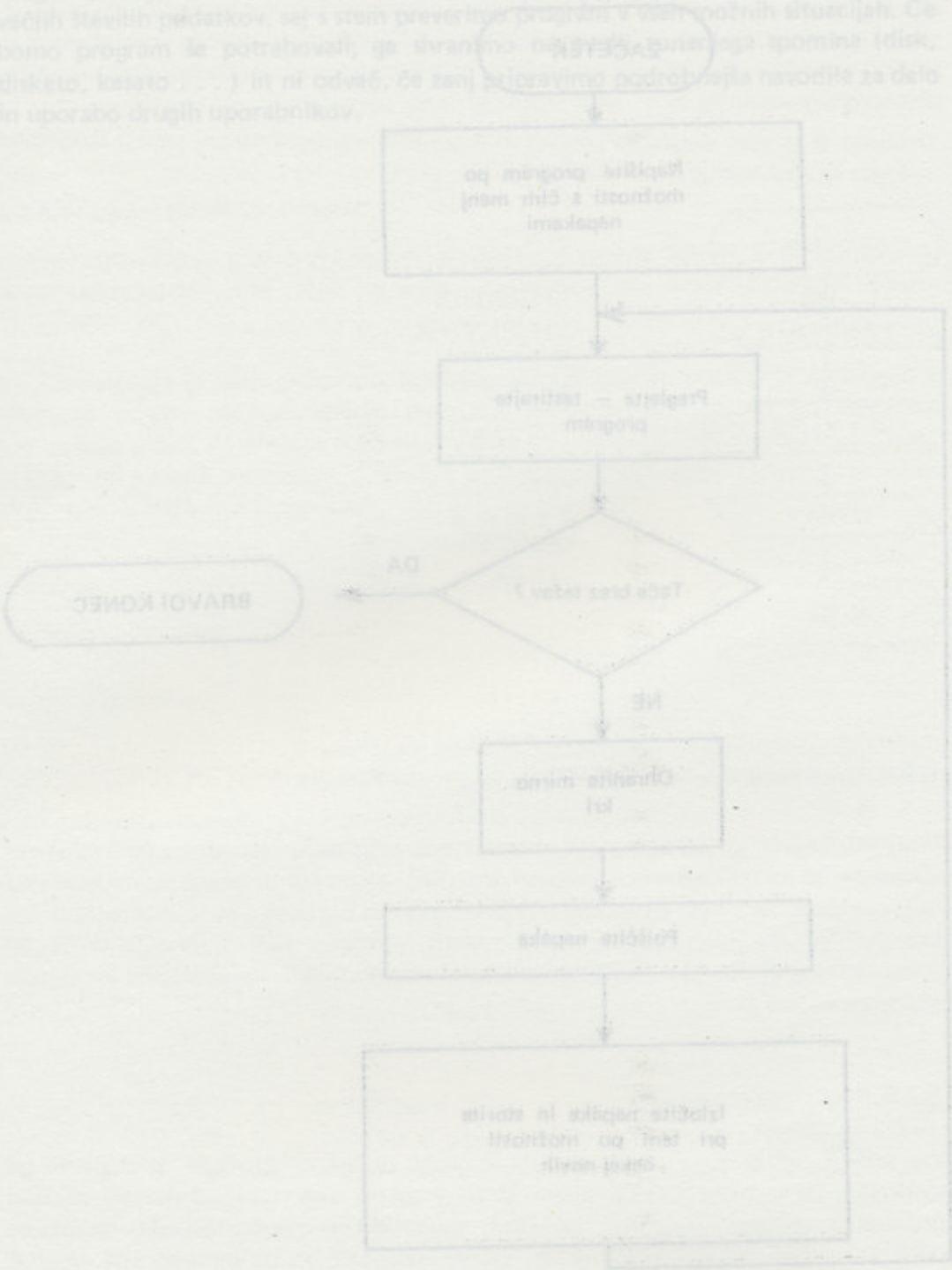
Ko smo program vnesli in ga s tem v grobi obliki kontrolirali, prehajamo na testiranje, saj še nimamo zagotovila, da bo program tudi resnično deloval in dajal pravilne rezultate. Testiramo ga tako, da vnašamo poenostavnejše izmišljene podatke. Na ta način lahko vidimo, če so vrednosti, ki jih dobimo kot rezultat procesa, pravilne in logične. Torej razumemo pod testiranjem logično preverjanje pravilnosti programa, saj so ostale napake že popravljene v prejšnji fazi prevajanja v strojni jezik.

9.2.6. Preizkusna obdelava in shranjevanje programa

Program, ki je testiran, moramo nadalje še nekaj časa preverjati na manjših in večjih številih podatkov, saj s stem preverimo program v vseh možnih situacijah. Če bomo program še potrebovali, ga shranimo na medij zunanjega spomina (disk, disketo, kaseto . . .) in ni odveč, če zanj pripravimo podrobnejša navodila za delo in uporabo drugih uporabnikov.

9.2.7. Diagram poteka – malo za šalo, malo zares





10. PRVI KORAKI V BASIC Z RAČUNALNIKOM ISKRA HR–84

BASIC je zelo enostaven programski jezik, ki je bil razvit 1964. leta na Darmounth College v New Hampshire, ZDA. Danes je to zelo priljubljen računalniški jezik za začetnike in manj zahtevne uporabnike. Da bi izdelali svoje prve programe ni potrebno, da znate takoj vse ukaze tega jezika. Pred računalnik lahko sedete tudi z minimalnim znanjem, saj vas bo le-ta v toku programiranja (ne pozabite, da je BASIC konverzacijski jezik) sam obveščal o napakah, ki ste jih storili. Z malo potrpljenja in uporabo ustrezne literature, boste kmalu ponosni nad svojimi prvimi programi!

Še predno začnemo s prvimi koraki v jeziku BASIC skupaj z računalnikom HR–84, pa si oglejmo še, katere posege moramo opraviti po vključitvi računalnika, da lahko vse napisano in naučeno tudi preverjamo in spremljamo z njim.

Po pravilni povezavi vseh elementov računalnika in priključitvi na omrežje (najprej vključimo monitor, nato pa še centralni del računalnika), dobimo na zaslonu naslednji izpis:

```
xxxxISKRA ŠIROKA POTROŠNJAxxxx
    RAM FROM $ 0000 TO $ 3FFF

M >
```

kazalec – znak na zaslonu, ki označuje mesto vpisovanja teksta

znak, ki označuje monitor

To pomeni, da smo v **monitorskem programu**. Monitor je glavni program za upravljanje računalnika, ki ugotavlja, vzporeja, verificira in odloča o vseh operacijah, ki jih opravlja računalnik. Če želimo delati v BASIC-u, moramo odtipkati ukaz monitorja:

```
BASIC (ali pa kar BAS)
```

Sedaj še pritisnemo tipko RET (return) in na zaslonu se izpiše naslednje:

```
xxx BASIC ISKRA – VTŠ MB xxx

B    #
```

kazalec v BASIC-u

znak, ki označuje BASIC

Pri ZX–SPECTRUMU tega posega ni potrebno opraviti, saj se sistem takoj nahaja v BASIC-u in na zaslonu dobimo naslednji izpis:

```
c 1982 Sinclair Research Ltd
```

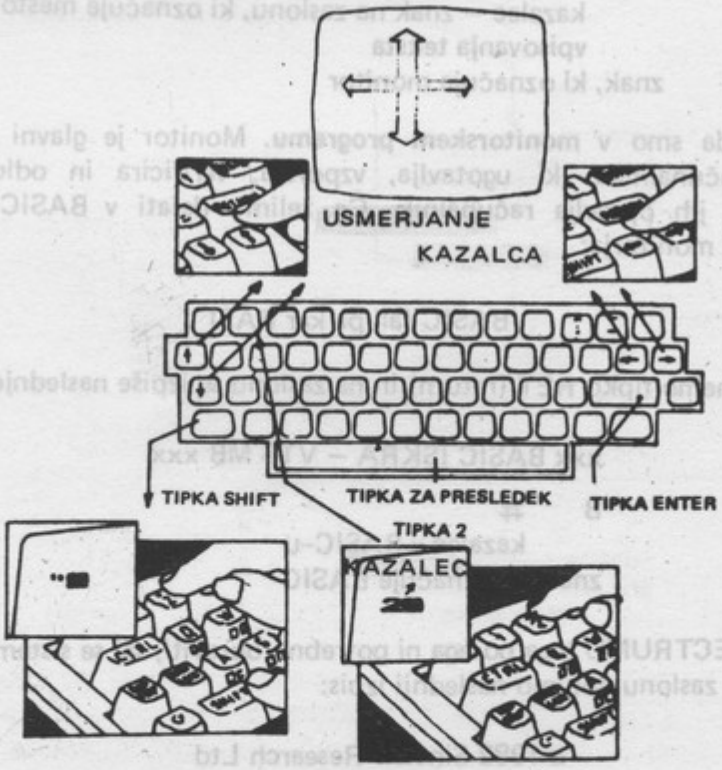
Zakaj pa pri HR–84 ni tega? Predvsem zato, ker je pripravljen na uporabo tudi drugih jezikov, kot npr. PASCAL . . . lahko začnemo!

10.1. Pa začnimo!

Učenje BASIC-a je kot učenje kateregakoli drugega jezika. Naučiti se moramo izraze in pravila. Izraze oziroma ukaze računalniku navadno posredujemo preko TIPKOVNICE. Preko nje odtipkavamo ukaze na dva načina:

- črka za črko (npr. za ukaz RUN vsako črko posebej: R–U–N, kot kaže slika 38 za primer HR–84),
- več–funkcijski način (tipka je nosilec večih ukazov)

Boljši je seveda prvi način. Zahteva profesionalno **tipkovnico**, ki pa je seveda dražja varianta. Če želimo zapisati npr. številko 2 (sl. 38), pritisnemo na tipko s to označbo in na katodnem zaslonu se izpiše številka 2, za njo pa utripajoča svetla točka, ki jo imenujemo **kazalec – kursor** (ang. cursor). Kazalec označuje mesto, na katerem se bo pojavil naslednji vtipkani znak. Če pritisnemo zopet tipko 2, bomo dobili zapisano številko 22, kazalec pa se je pomaknil za eno mesto naprej in čaka na nov vpis znaka oziroma določen ukaz. Če želimo vtipkati znak narekovaj ("), moramo uporabiti istočasno tipko 2 in tipko SHIFT, prav tako, kot pri pisalnem stroju za pisanje velikih črk.



Slika 38: Osnovne informacije o uporabi tipkovnice

Tako kot na koncu vsakega stavka zapišemo piko, tako tudi v BASIC-u na koncu vsakega zapisa pritisnemo tipko RETURN (RET) ali (ENTER ali EXECUTE ali NEWLINE itd., glede na vrsto računalnika), kar pove računalniku, da smo odtipkali vse, kar smo želeli in da je sedaj na vrsti računalnik, da izvrši ukaz.

Tako, spoznali smo nekaj osnovnih informacij o načinu vpisovanja posameznih znakov preko tipkovnice, seveda pa nam to ne zadošča, saj želimo izdelati svoj prvi program. Kot že vemo, je program sestavljen iz **programskih navodil – stavkov**. Eno ali več programskih navodil skupaj tvori stavek. Vsak stavek ima svojo številko (oznako). Če napišemo določen programski stavek, bo računalnik njegovo pozicijo avtomatično odredil glede na število stavka. Za primer uporabimo in obenem spoznajmo prvi ukaz PRINT, ki računalniku ukaže:

„PIŠI". Kaj bo napisal, seveda zavisi od nas. Če želimo, da nam izpiše določen tekst, npr. ZDRAVO MILENA, to opravimo na sledeč način:

PRINT
(piši)

10	PRINT	„ZDRAVO MILENA"
št.	ukaz:	Tekst, ki ga želimo izpisati
stavka	PIŠI	in to v narekovajih!

Postopek zapisa stavka:

– vtipkamo številki 1 in Ø (Ø = nič, 0 = črka o!)
– vtipkamo črke P, R, I, N in T (ali samo tipko P v večfunkcijskem načinu)
– vtipkamo narekovaj (obenem pritisnemo tipki 2 in SHIFT)
– vtipkamo črke Z, D, R, A, V in O
– pritisnemo tipko za presledek
– vtipkamo črke M, I, L, E, N in A
– vtipkamo narekovaj
– pritisnemo tipko RETURN (RET) (v nadaljevanju bomo zapisovali samo znak R)

O = črka o
Ø = število nič
(to velja v računalništvu, da se izognemo zamenjavi)

Ko smo vse to naredili, vidimo, da računalnik še vedno zapiše vse in sicer številko, ukaz in besedilo v narekovajih. Tega ne želimo, saj želimo, da računalnik samo pozdravi Mileno, ne pa da ji pokaže, kako mora biti zapisan programski stavek v BASIC-u. Računalnik bo Mileno pozdravil šele, ko mu to ukažemo z ukazom RUN, kar

RUN
(teči)

pomeni TEČI oziroma izvršitev programa po zadanih ukazih.

Ko je računalnik izpisal svoj pozdrav Mileni , si ta želi, da bi spoznala, kako izgleda program v BASIC-u, ki je vodil računalnik, da jo je pozdravil. To ne bo težko, le spoznati in uporabiti bo morala ukaz LIST, ki ukaže računalniku, da izpiše na zaslonu vsebino programa, ki ga ima v spominu. Spodnja slika prikazuje obe obliki izpisov, ki jih bomo za lažje razumevanje v nadaljevanju še uporabljali. Toda dodan je še en stavek in sicer: 2Ø END.

LIST
(Izpiše na zaslonu vsebino programa v spominu !)

LIST	RUN
1Ø PRINT „ZDRAVO MILENA"	ZDRAVO MILENA
2Ø END	

SI. 40

END
STOP
(konec programa)

Drugi stavek (stavek št. 20) je na koncu vsakega programa in obvešča računalnik, da je prišel na konec programa ter da naj izključi z izvrševanjem. Uporabljamo lahko ukaz END ali STOP. Če te zanima razlika, si oglej abecedni pregled značilnih ukazov na mikroračunalnikih na koncu knjige (pri END in STOP).

10.2. Prvim programom naproti

PRINT
(piši)

Da bomo lahko napisali svoje prve programe, spoznajmo najprej programski ukaz PRINT, kar pomeni računalniku: PIŠI. Navadno pišemo računalniške ukaze z velikimi črkami, (SINCLAIR, ZX 81 . . .), za HR 84 pa ni nič hudega, če jih napišemo z majhnimi.

Sedaj pa izvedimo še naslednji primer:

PRINT 25 + 30 {RET}

B#
ali
OK znak, da je računalnik opravil svoje delo

Primer nam je pokazal, da lahko računalnik uporabljamo tudi kot **kalkulator**, saj nam je izračunal vsoto dveh števil. Pa preizkusimo še druge **računske operacije!**

```
PRINT 3+4+5      RET      12
PRINT -3-4-5     RET     -12
PRINT 3-4-5      RET      -6
PRINT 3.5+4.6    RET     8.1    pika pomeni decimalno vejico
PRINT 5*6        RET      30    * je znak za množenje
PRINT 10/2       RET       5
PRINT -10/2      RET      -5
PRINT 2*2        RET       4
```

Seveda računalnik zmore tudi zahtevnejše računske operacije, kot npr.:

```
PRINT SQR (36) ...  RET   6         izračuna kvadratni koren
PRINT EXP ( 1 ) ... RET   2.71828   izvede se matematična operacija
                                    e^x ( e = 2.71828 )
PRINT LOG (2)  ...  RET   0.693147  izračuna naravni logaritem števila
```

PRINT SIN (X) izračuna sinus x, pri čemer je x podan v radianih. Če imamo kot podan v stopinjah, ga moramo pretvoriti v radiane (npr. A = sin (x PI/180). Prav tako izračuna tangena, cosinus in arkustangens).

Oglejmo si še nekaj primerov pravilnega zapisovanja določenih matematičnih operacij:

```
1. A^2 ..... A ** 2
2. B^3 ..... B ** 3
3. a^1/2 ..... a ** (1/2)
4. a^3/2 ..... a ** (3/2)
5. a^2/5 ..... a ** (2/5)
6. a^-2 ..... a ** (-2)
7. a^u-1 ..... a ** (u-1)
```

Izvedimo sedaj naslednje primere:

```
PRINT 23456 * *   2      5.50184E+08
PRINT 34567 * *  1000     3.4567E+07
PRINT .4567 * *  0001     4.567E-05
```

Kaj pa je sedaj to? Nič posebnega! Računalnik je izpisal rezultat v tako imenovani eksponentni obliki, prav tako kot nekateri kalkulatorji. Torej:

5,50184E+08 pomeni: 5,50184 . 10^8 ali 5,50184 . 100000000 ali 550184000

Tak zapis se imenuje tudi zapis s **PLAVAJOČO VEJICO** in nam omogoča večjo natačnost, saj omogoča pri HR 84 natančnost računanja več kot osnovnih šest mest.

Spregovorimo še nekaj besed o prioriteti, to je o vrstnem redu izvajanja računskih operacij. Računalnik opravlja računske operacije po naslednjem vrstnem redu:

1. potenciranje (** ali !)
2. minus kot predznak (npr. n^{-2})
3. množenje in deljenje (* in /)
4. seštevanje in odštevanje (+ in –)

Enakovredne računske operacije se vršijo v vrstnem redu od leve proti desni. Vrstni red računskih operacij lahko uredimo drugače s pomočjo okroglih oklepajev na običajni način, kakor smo v matematiki navajeni. Poglejmo primer:

```
PRINT 4 + 2 * 4 RET
PRINT (4+2) * 4 RET
```

Naj ne bo odveč naslednji primer pravilnega zapisovanja algebrajskih izrazov:

1.) $x - y$ ..... x – y

2.) $a - 2b + 3c$ ..... a – 2*b + 3 * c

3.) $ab - 4cd + 3xy$ ..... a * b – 4 * c * d + 3 * x * y

4.) $\frac{a}{x} - 2\frac{b}{y} + 3\frac{cd}{u}$ ..... a/x – 2 * b/y + 3 * c * d/u

5.) $\frac{a}{b} - c$ ..... a/b – c

6.) $\frac{a}{b-c}$ ..... a/(b–c)

7.) $\frac{a}{b \cdot c}$ ..... a/b/c ali a/(b * c)

8.) $\frac{a-b}{c+d}$ ..... (a–b) / (c+d)

9.) $ax^4 - 2bx^3 + cx + d$ ..... a * x ! 4–2 * b * x!3+c * c + d

10.) $(ax)^3$ ..... (a * x) ! 3 ali (a * x) ** 3)

11.) $\sqrt{a^2 + b^2}$ ..... (a * a + b * b) ! 1/2

12.) $\sqrt[3]{\frac{a+b}{c(d-e)^2}}$ ..... ((a + b) / (c (d–e) ! 2)) ! (1/3)

Oglejmo si še nekaj drugih oblik in variant stavka PRINT:

```
PRINT B RET          2
```

računalnik izpiše iz spomina vrednost spremenljivke B (npr. 2)

```
PRINT "B" RET        B
PRINT  RET
```

računalnik ne izpiše ničesar – uporabljamo za pisanje razmakov med vrsticami

```
PRINT "GOGI GERLIČ" RET      GOGI  GERLIČ
PRINT "      GERLIČ" RET           GERLIČ
```

izpusti toliko prostih mest, kolikor mu jih zapišemo v narekovaje

Torej vse, kar je zapisano med dvema narekovajema, bo računalnik zapisal. Namesto števil lahko pišemo tudi črke kot spremenljivke, a le, če imamo v spominu shranjena ustrezna števila (v našem primeru B = 2).

S tem znanjem že lahko pišemo zelo enostaven program, ki ga prikazuje slika 41.

```
LIST                                    RUN

10 PRINT "ZDRAVO" RET                   ZDRAVO
20 PRINT "PRIJATELJ" RET                PRIJATELJ
30 PRINT "RAČUNALNIKOV!" RET            RAČUNALNIKOV!
40 END
                                        B #
```

Slika 41

Pa je resnično enostaven; a nič zato! Tudi težje bomo kaj kmalu znali napisati.

Kot vidimo, ima tipkovnica HR–84 vse značilne črke jugoslovanske abecede (Č, Š, Ž, Ć, Đ). To je za naše delo zelo ugodno. Toda precej računalnikov, predvsem uvoženih (npr. COMMODORE, SINCLAIR ZX 81 ali SPECTRUM itd.), tega nima, saj te črke niso značilne za angleško abecedo, ki jo ti računalniki uporabljajo. Če bomo delali na takšnih računalnikih, bomo pač namesto šumnikov pisali C,S,Z, namesto Đ pa DJ, namesto Ć pa kar C.

Že prej smo povedali, da lahko med posameznimi vrstami izpisa določenega teksta v programu pišemo tudi razmake. Opravimo to pri našem prvem programu s slike 41!

```
LIST                                    RUN
10 PRINT "ZDRAVO" RET                   ZDRAVO
15 PRINT RET
20 PRINT "PRIJATELJ" RET                PRIJATELJ
25 PRINT RET
30 PRINT "RAČUNALNIKOV!" RET            RAČUNALNIKOV!

                                        OK
```

Skušajmo z našim dosedanjim znanjem narisati poševno črto iz treh zvezdic (* * *). Ali bi šlo? Seveda, nič lažjega!

```
LIST                                    RUN
10   PRINT "***"                        ***
20   PRINT " ***"                        ***
30   PRINT "  ***"                        ***
40   PRINT "   ***"          ———           ***
50   PRINT "    ***"                        ***
60   PRINT "     ***"                        ***
70   PRINT "      ***"                        ***
80   END
                                        OK
```

Slika 42

Pozneje bomo videli, da lahko tak program napišemo mnogo krajše in to le s 4 programskimi stavki. Do takrat pa moramo spoznati še nekaj ukazov! Ukaz PRINT in ostale osnovne rutine za pisanje in vnašanje programov v računalnik že poznamo. Zato si lahko zadamo naslednji program, kot problem za preveritev osvojenega znanja.

```
LIST                                    RUN
 50 PRINT "000   U  000"
 70 PRINT "    0    0"
 10 PRINT "     000"
 90 PRINT " 000000000"
 40 PRINT " 00      00 "
 8
 80 PRINT "     000"
 30 PRINT "   00   00"
 20 PRINT "    00000"
 60 PRINT "   0 === 0"
100 PRINT " 0 0Ø0Ø0 0"
120 END
110 PRINT " 00000 0"
```

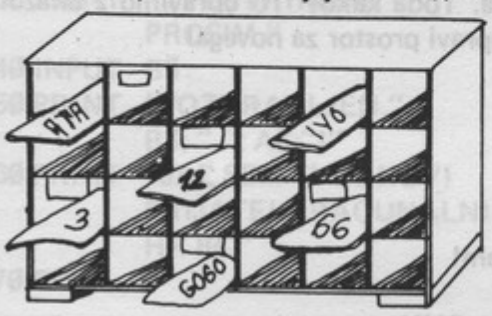
Slika 43

Vprašanja so naslednja:

– Ali bo računalnik ta program sprejel kot ustreznega?
– Ali bo program stekel?
– Program bo izrisal risbico. Kaj bo izrisal kot risbico?

10.3. Spremenljivke

Že v uvodu, pa tudi pozneje smo povedali, kako si računalnik določene podatke zapomni – vstavlja jih v posamezne celice spomina. Če zadržimo primerjavo z omaro in policami v njej (sl. 14), lahko naš problem prav tako zelo enostavno razložimo, kot kaže to slika 44. Vsakemu številu ali besedi, moramo torej dati imena. „Imena" za **števila** so A, B, C, . . . Y, Z . . ABC, D7, MINA, milo itd., za besede, to je **nize znakov** (po tem tudi naziv **NIZI ali STRINGI**) pa A$ (S dolar – izgovorjava: A dolar), B$, C$ . . . Y$, Z$. V spominu računalnika so lahko istočasno tako števila kot nizi; če v spominu že obstaja število z „imenom" A, ni nobenega problema, da vanj ne bi mogli spraviti še niza z „imenom" A$, saj računalnik obe vrsti spremenljivk zelo dobro loči. V našem primeru (sl. 44) imamo naslednje vrste spremenljivk:



Slika 44

– števila: A = 3, B = 12, C = 66
– nizi: A$ = ATA, B$ = GOGO, C$ = IVO

Kako pa opravimo postopek „poimenovanja" celic oziroma spremenljivk? Za to potrebujemo novi ukaz, to je ukaz LET (naj je . . .). Oglejmo si, kako bi uporabili ta ukaz v primerih slike 44:

LET
(Naj je . . .)

```
LIST
10 LET A  =  3
20 LET B  =  12
30 LET C  =  66
40 LET A$ = "ATA"
50 LET B$ = "GOGO"
60 LET C$ = "IVO"
80 PRINT A;B;C
90 PRINT A$;B$;C$;
100 END
```

```
LIST
10 PRINT A, B, C
20 PRINT A$, B$, C$
30 END
```

```
RUN
 30      12      66
ATA GOGO IVO

B#
```

Računalnik nam jih je resnično poiskal, a zelo nerodno izpisal, saj so združeni kot ena številka in ena sama beseda. Namesto podpičja (;) vpišimo vmes vejico (,)!

To je že bolje. Oba primera sta nas naučila, da ni vseeno ali med spremenljivkami pišemo vejico(,), ali podpičje (;): če napišemo vejico, bodo vrednosti spremenljivk med seboj v določenem večjem ali manjšem razmaku (**glede na vrsto računalnika**), če pa napišemo podpičje (;), bo za vrednostjo prve spremenljivke brez razmaka zapisana druga, itd.

Kot primer si oglejmo še enostavni matematični program. Toda, če želimo v BASIC-u napisati naslednji program, je dobro ali predhodnega shraniti v zunanji spomin (npr. disketo, kaseto . . .), če pa ni tako pomemben (kot naš, ki je le za vajo in primer), pa ga izbrisati iz spomina. Toda kako? To opravimo z ukazom NEW, ki pomeni: zbriši stari program in pripravi prostor za novega!

N E W
(Zbriši stari program in
pripravi prostor za novega)

Oglejmo si ta enostavni matematični program!

LIST

```
1Ø LET x = 5
2Ø PRINT "x = "; x
3Ø PRINT "x**2 = "; x ! 2
4Ø PRINT "x**3 = "; x ! 3
5Ø END
```

RUN

```
x = 5
x**2 = 25
x**3 = 125

B #
```

Slika 45

Z programskim ukazom LET smo točno določili vrednost vsake spremenljivke (npr. 1Ø LET x = 5). Na ta način nam vsak ponovni potek programa da kot rezultat vedno isto vrednost konstante x. Če želimo vrednost x spremeniti, moramo menjati program, oziroma samo stavek 1Ø. Toda takšna stalna korektura programa ni prav nič prikladna. Zato potrebujemo novo instrukcijo za računalnik, ki nas reši teh režav. To je ukaz INPUT, ki pomeni: vstavi vrednost za spremenljivke! Ko računalnik v programu preide do ukaza INPUT, ostane na tem tako dolgo (pri računalniku HR 84 se izpiše na zaslonu vprašaj (?)), dokler uporabnik ne vnese vrednost spremenljivk. Oglejmo si to kar na primeru:

I N P U T
(Vstavi vrednost za spremenljivko)

LIST

```
1Ø PRINT "Izračunam ti lahko
         kvadrat katerega koli
         števila!"
2Ø PRINT
3Ø PRINT "Koliko je x? "
```

RUN

```
Izračunam ti lahko kvadrat kateregakoli
števila!
```

```
40 INPUT x                    Koliko je x?
50 PRINT "x !2 = ";x!2
60 END                        ? 10
                              x!2=100
                              B
```

Napišimo program, ki nas bo vprašal za ime, nas pozdravil in se tudi sam predstavil. Bi ga znali sami? No, poizkuswte in šele nato preverite z našim!

```
LIST                                  RUN

10 PRINT „TVOJE IME, PROSIM."         TVOJE IME PROSIM
20 INPUT A$                           ? GORAZD
30 PRINT "TVOJ PRIIMEK,               TVOJ PRIIMEK PROSIM
   PROSIM."
40 INPUT B$                           ? GERLIČ
50 PRINT "POZDRAVLJEN,";
   B$;" ";A$;"!"                      POZDRAVLJEN, GERLIČ GORAZD!
60 PRINT "JAZ SEM TVOJ NOVI           JAZ SEM TVOJ NOVI PRIJATELJ
   PRIJATELJ RAČUNALNIK               RAČUNALNIK HR 84
   HR 84."
70 END
```

V našem programu vam bi menda delal težave stavek 50, saj ima kar precej znakov, zato ga analizirajmo skupaj!

```
50 PRINT"POZDRAVLJEN,";B$;" ";A$;"!"
```

Napiše klicaj

Poskrbi, da se klicaj napiše takoj za besedo imena

Napiše ime

Poskrbi, da se za krajšim presledkom napiše takoj ime

Poskrbi (zapiše) za presledek med priimkom in imenom (npr. v našem primeru smo pritisnili tipko za presledek dvakrat

Poskrbi, da se presledek zapiše takoj za priimkom

Napiše priimek

Poskrbi, da se priimek zapiše takoj za vejico

Izpiše besedo POZDRAVLJEN in vejico

Ukaz PRINT (piši)

Številka stavka

Za vajo in obenem kot primer za pisanje enostavnih, a kar uporabnih programov s področja matematike, vam podajamo še naslednja dva programa. Vaša naloga je, da jih dobro pregledate in napišete, kaj bo računalnik izpisal na svojem zaslonu, ko mu bomo odtipkali ukaz RUN!

1. Program za računanje ploščine trikotnika, ki ima podani: stranico "a" in višino "v":

```
1Ø PRINT "IZRAČUNAL TI BOM PLOŠČINO TRIKOTNIKA S STRANICO 'A'
       IN VIŠINO 'V'."
2Ø PRINT
3Ø PRINT "KOLIKA JE STRANICA 'A'? "
4Ø INPUT A
5Ø PRINT "KOLIKA JE VIŠINA 'V'? "
6Ø INPUT V
7Ø LET P = A V/2
8Ø PRINT "PLOŠČINA TRIKOTNIKA =";P
9Ø END
```

Isti program lahko zapišemo tudi v tej obliki:

```
1Ø PRINT "IZRAČUNAL TI BOM PLOŠČINO TRIKOTNIKA S STRANICO 'A'
       IN VIŠINO 'V'."
2Ø PRINT:PRINT
3Ø PRINT "STRANICA 'A' JE: ":INPUT A
4Ø PRINT "VIŠINA 'V' JE:" :INPUT V
45 PRINT:PRINT
5Ø PRINT "PLOŠČINA TRIKOTNIKA S STRANICO A =";A;" IN VIŠINO
       V = ";V;"JE:":A*V/2
6Ø END
```

V tem programu smo spoznali dve pomembni informaciji za programiranje in sicer:

- v narekovajih (" ") stavka PRINT lahko pišemo in povdarimo določene informacije le z enojnimi narekovaji (npr. PRINT "STRANICA 'A' JE:")
- z dvopičjem lahko ločimo več programskih ukazov v enem samem stavku (npr. stavki 20, 30, 40 in 45).

2. Program za izračunanje enostavnejših matematičnih operacij:

```
1Ø REM    PROGRAM ZA IZRAČUNANJE ENOSTAVNIH
2Ø REM    MATEMATIČNIH OPERACIJ
3Ø REM    SESTAVIL: IVAN GERLIČ, MARIBOR 1984
35 PRINT  "IZRAČUNAL TI BOM VSOTO, RAZLIKO, PRODUKT,
          KOLIČNIK IN"
4Ø PRINT  "KVADRAT DVEH ŠTEVIL, KI MI JIH BOŠ PODAL!"
5Ø REM    VNOS SPREMENLJIVK
55 PRINT  "PROSIM, PRVO ŠTEVILO"
6Ø INPUT  A
```

```
 70 PRINT  "TVOJE PRVO ŠTEVILO JE:";A
 75 PRINT  "PROSIM, DRUGO ŠTEVILO"
 80 INPUT  B
 90 PRINT  "TVOJE DRUGO ŠTEVILO JE:";B
100 REM    RAČUNANJE VSOTE
110 PRINT : PRINT
120 PRINT  "VSOTA TVOJIH ŠTEVIL JE:";A+B
130 PRINT
140 REM    RAČUNANJE RAZLIKE
150 PRINT  "RAZLIKA TVOJIH ŠTEVIL JE: ";A – B
160 PRINT
170 REM    RAČUNANJE PRODUKTA
180 PRINT  "PRODUKT TVOJIH ŠTEVIL JE: "; A B
190 PRINT
200 REM    RAČUNANJE KOLIČNIKA
210 PRINT  "KOLIČNIK TVOJIH ŠTEVIL JE: "; A/B
220 PRINT
230 REM    RAČUNANJE KVADRATA PRVEGA ŠTEVILA
240 PRINT  "KVADRAT TVOJEGA PRVEGA ŠTEVILA JE: "; A!2
250 PRINT
260 REM    RAČUNANJE KVADRATA DRUGEGA ŠTEVILA
270 PRINT  "KVADRAT TVOJEGA DRUGEGA ŠTEVILA JE: "; B!"
280 PRINT : PRINT : PRINT
290 REM    NAPOTKI ZA NADALJEVANJE DELA
300 PRINT  "UPAM, DA SI ZADOVOLJEN Z MOJIM IZRAČUNOM"
310 PRINT
320 PRINT  "ČE ŽELIŠ, DA IZRAČUNAM ŠE NOVI DVE ŠTEVILI,
           OPRAVIŠ TO PO NASLEDNJEM ZAPOREDJU:"
330 PRINT  " – ODTIPKAŠ ' RUN '"
340 PRINT  " – PRITISNEŠ TIPKO RET"
350 END
```

REM
REMARK
(komentar)

V programu smo se zopet srečali z novim ukazom in sicer REM. Često bomo napisali večje programe, ki bodo zavzemali tudi več strani formata A4. Ko takšen program pregledujemo, oziroma ko ga pregleduje drug uporabnik, ima kar precej dela za razpoznavanje posameznih programskih postopkov. Zato imamo v BASIC-u ukaz REM (kratica) oziroma REMARK (dovolj je že, če odtipkamo kratico REM!), kar računalniku pomeni, da je ta stavek oziroma vrstica v programu, ki jo naj pri izvrševanju preskoči. Torej, za ukazom lahko napišemo kakršnokoli obvestilo, komentar, napotek, itd., ki bo nam oziroma drugim uporabnikom programa olajšal njegovo prepoznavanje in sestavo. Torej bomo besedilo pod ukazom REM lahko prebrali le, ko bomo odtipkali ukaz LIST, s katerim pregledamo zgradbo programa.

V našem programu smo uporabili ukaz REM za tri namene, ki so običajni tudi v splošni programerski praksi, in sicer:

- v začetku programa za vpisovanje naslova oziroma problema naloge, ki jo program rešuje;
- za vpis sestavljalca programa;
- za komentarje v toku programa.

Sedaj pa še naloga za tebe! Program za izračunavanje enostavnih matematičnih operacij napiši v krajši obliki, tako kot smo to storili v drugem primeru programa za izračunavanje trikotnika. Menda ne to težko!

V tem poglavju smo spoznali dva načina za vnašanje podatkov in sicer LET in INPUT. Oglejmo si še tretji zanimiv način, ki ga uporabljamo navadno takrat, ko želimo shraniti podatke znotraj posameznega programa. Za to potrebujemo ukaze:

DATA
READ
RESTORE

DATA, READ in RESTORE. Oglejmo si posamezne. Ukaz READ računalniku pomeni: „Prečitaj podatek, ki je vpisan v ukazu DATA". Ob ukazu READ so še samo imena spremenljivk (njihove vrednosti so lahko numeričnega ali alfanumeričnega tipa), katerih vrednost naj računalnik prebere iz zkaza DATA. Torej je ukaz DATA neke vrste skladišče – banka podatkov, s katerimi želimo v programu delati. Dovolj je besedičenja, raje si oglejmo enostaven primer!

LIST

```
10 READ X
20 PRINT X
30 PRINT X!2
40 PRINT X!3
50 PRINT X!(1/2)
60 PRINT X!(1/3)
70 DATA 10
80 END
```

RUN

```
 10
 100
 1000

 3.16228
 2.15443

B #
```

Razložimo program:

10 Preberi vrednost spremenljivke x, ki je podana v stavku DATA. Računalnik prične iskanje in to vedno od začetka programa, ter v 70 stavku najde, da je vrednost x = 10

20 napiši vrednost za x (10)

30 napiši vrednost za x^2 (10)

40 napiši vrednost za x^3 (100)

50 napiši vrednost za $x^{1/2}$ (3.16228)

60 napiši vrednost za $x^{1/3}$ (2.15443)

70 pride zopet do ukaza DATA in, ker ga sem ni poslal ukaz READ, stavek 70 preskoči

80 konec programa

Poizkusimo sedaj naš program na sliki 45 napisati in izvesti z ukazom READ in DATA!

```
LIST                          RUN
10 READ X
20 PRINT "X=";X               X= 5
30 PRINT "X!2=";X!2           X!2= 25
40 PRINT "X!3=";X!3           X!3= 125
50 DATA 5
60 END                        B #
```

Slika 46

Nič lažjega, boste rekli. Da, tako je. Zato pa kar naprej. Do sedaj smo se ukvarjali le z eno spremenljivko, toda v READ in DATA se lahko nahaja več spremenljivk. Povejmo še to, da ima vsak ukaz DATA „kazalnik – puščico", ki računalniku pove, koliko podatkov je že prebral iz DATA stavka. Vsako branje torej premakne kazalec za eno mesto naprej (od leve k desni strani). Oglejmo in razložimo si to na naslednjem primeru:

```
LIST                                    RUN
10 PRINT  "A  ";"B  ";"A+B  ";"A B "    A     B     A+B   A B
20 READ A,B                             5     10    15    50
30 PRINT A;B;A+B;A B                    15    20    35    300
40 READ A,B
50 PRINT A;B;A+B;A B                    25    30    55    750
60 READ A;B
70 PRINT A;B,A+B;A B                    B #
80 DATA 5,10,15,20,25,30
90 END
```

Slika 47

10 Napiši naslove : A B A+B A B
20 Preberi vrednosti za spremenljiviki A in B v DATA stavku, ki je v 80 vrstici.
Takšen stavek s kazalcem izgleda tako:

80 DATA 15, 20, 25, 30
po včitanju (položaj kazalca)
5, 10 pred včitavanjem

30 Napiši zahtevane vrednosti (5, 10, 15, 50)
40 Preberi vrednosti za spremenljivki A in B:

80 DATA 5, 10
25, 30
po včitavanju
15,20 pred včitavanjem

50 Napiši zahtevane vrednosti (15, 20, 35, 300)
60 Preberi vrednosti za spremenljivki A in B:

80 DATA 25, 30
po včitavanju
5, 10, 15, 20 pred včitavanjem

70 Napiši zahtevane vrednosti (25, 30, 55, 750)
80 Preskoči, ker ga sem ni poslal ukaz READ
90 Konec programa.

V zgornjem primeru je po odčitanju vseh šestih vrednosti za spremenljivki A in B kazalec na koncu spiska podatkov v ukazu DATA. Če želimo ponovno prebrati iste podatke, je potrebno kazalec postaviti na začetni položaj. To nam omogoči ukaz RESTORE. Torej, ukaz RESTORE vrača kazalec v DATA stavku na začetno pozicijo in s tem omogoča ponovno čitanje istih podatkov. Oglejmo si to na preprostem primeru!

LIST

```
10 READ A,B,C
20 PRINT "(A+B) C=";(A+B) C
30 RESTORE
40 READ X,Y,Z
50 PRINT "(X+Y)!Z=";(X+Y)!Z
60 DATA 2, 8, 3
70 END
```

RUN

```
(A+B) C = 30
(X+Y)!Z = 1000
B
```

Za primer uporabe alfanumeričnih spremenljivk v DATA stavkih, si oglejmo enostaven primer programa za shranjevanje pomembnejših naslovov, telefonskih številk . . . Seveda ga lahko razširite!

```
 5 REM   SESTAVIL: IVAN GERLIČ,PA, MARIBOR
10 READ  A$, B$, C$
20 PRINT "IME IN PRIIMEK ==NASLOV==TELEFON"
30 PRINT "----------------------------------"
40 PRINT A$
42 PRINT B$
43 PRINT C$
50 DATA  "GOGI GERLIČ ==  RAPOČEVA 18, MARIBOR== 32782"
60 DATA  "MITJA KRAJNC == PRI POŠTI 12, MARIBOR==34768"
70 DATA  "MICI KOKOŠ== VILHARJEVA 34, LJUBLJANA==987654"
80 END
```

10.4. Premikanje zapisov

Najprej se povrnimo nekoliko nazaj in sicer na sliko 43, ob kateri smo ti zadali tri vprašanja. Menim, da niso bila težka! Oglejmo si odgovore!

- DA
- DA (številke stavkov računalnik uredi sam, ne glede na neurejen vnos!)
- Narisal je portret Micke, zato ga lahko imenujemo tudi: program MICKA

Toda našemu programu MICKA lahko očitamo, da je izrisal našo Micko v skrajnem levem zgornjem kotu zaslona; želimo pa, da bi nam narisal Micko lepo v sredini zaslona. Je to mogoče? Je! Opraviti in rešiti moramo dva postopka oz. problema, in sicer:

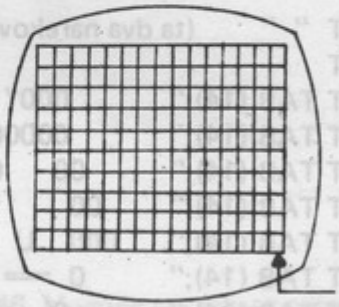
- premakniti Mickino glavo na zaslonu za nekaj centimetrov niže
- premakniti Mickino glavo na zaslonu za nekaj centimetrov proti desni strani zaslona.

Prvi problemček je enostaven. Uporabimo samo ukaz: PRINT " ", kar pomeni, naj računalnik izpiše prazen prostor.

Program MICKA bi sedaj izgledal tako:

```
 2 PRINT " "
 4 PRINT " "
1Ø PRINT "      000"
2Ø PRINT "     00000"
       itd.
```

Slika 48



Slika 49: Vrstice in stolpci

Da bomo lahko rešili tretji problem, moramo spoznati nekaj novih informacij in tudi nov ukaz. Seveda boste rekli, saj si lahko tudi tukaj pomagamo z večimi prostimi mesti v PRINT stavkih od 1Ø do 11Ø; npr. dodamo v vseh po 10 prostih mest in je problem rešen. Pravilno! Tudi tako bi lahko rešili našo nalogo in narisali Micko približno v sredino zaslona. Toda, spoznati želimo več, zato spoznajmo še drug način rešitve tega problema, ki pa je uporaben še za marsikaj drugega, kot bomo videli pozneje.

Najprej povejmo, da je področje katodnega zaslona (ekran) razdeljeno na večje število polj, ki tvorijo stolpiče in vrstice, kot kaže slika 49.

V našem primeru (računalnik HR 84) ima zaslon 24 vrstic in 40 stolpcev, kar da 24 x 40 = 960 možnih polj zapisa. Seveda imajo računalniki več stolpcev in vrstic in s tem več možnih polj zapisa. Čimveč je teh, tem kvalitetnejši je računalnik, saj ima možnost večjega števila zapisov znakov na zaslonu.

Če se sedaj vrnemo na našo Micko in sliko 48, lažje razumemo, zakaj je bila Mickina glava v tem primeru premaknjega navzdol. Narisana je bila torej nekaj vrstic navzdol zato, ker stavka 2 in 4 izpišeta v teh vrsticah le prazne prostor, šele v vrstici 3 prične stavek 10 risati glavo – pričesko naše lepe Micke.

PRINT TAB

Za premik zapisa na določenem mestu vrstice se uporablja ukaz TAB, ki mora biti vedno vezan na ukaz PRINT, torej ima obliko:

```
10 PRINT TAB (5);"     "
```
št. stavka ukaz: PIŠI NA 5. POLJU VRSTICE

Če želimo svoj priimek in ime napisati v sredini prve vrstice, bomo to opravili z naslednjim stavkom:

```
1Ø PRINT TAB (14); "GERLIČ GORAZD"
```

No, sedaj pa še h končni obliki našega programa Micka. Imel bo naslednje stavke:

```
 2 PRINT " "      (ta dva narekovaja lahko tudi izpustimo)
 4 PRINT
1Ø PRINT TAB (14);"      000"
2Ø PRINT TAB (14);"     00000"
30 PRINT TAB (14);"    00   00"
4Ø PRINT TAB (14);"   00     00"
5Ø PRINT TAB (14);"  000  U  000"
6Ø PRINT TAB (14);"    0 === 0"
7Ø PRINT TAB (14);"     0   0"
8Ø PRINT TAB (14);"      000"
9Ø PRINT TAB (14);"  000000000"
100 PRINT TAB (14);"   0 0Ø0Ø 0"
110 PRINT TAB (14);"   0 00000 0"
120 END
```

Menim, da program razumete in da vam ne bi bilo več težko narisati katero drugo enostavno risbico, kot npr. avto, hišo, itd. Za vajo izdelajte tak program!

Seveda je bila naša **"računalniška grafika"** izredno enostavna, tako da jo težko imenujemo s tem imenom. Toda kakorkoli že, uspeli smo narisati svojo prvo risbo.

Povrnimo se zopet k ukazu TAB in si oglejmo še eno njegovo koristno uporabo in sicer pri oblikovanju oziroma tabeliranju (zato tudi ime TAB) izpisov. Primer programa iz področja matematike z uporabno TAB ukaza je pred vami:

```
1Ø REM          MATEMATIKA
2Ø REM   SESTAVIL: IVAN GERLIČ – PA MARIBOR
3Ø PRINT "IZRAČUNAL TI BOM VSOTO, PRODUKT IN KOLIČNIK TVOJIH
      ŠTEVIL!"
35 PRINT
4Ø PRINT "ŠTEVILO A=": INPUT A : PRINT A
5Ø PRINT "ŠTEVILO B=": INPUT B : PRINT B
6Ø PRINT : PRINT
7Ø PRINT TAB (1); "A+B"; TAB (6); "A B"; TAB(6); "A/B"
8Ø PRINT
9Ø PRINT TAB (1);A+B;TAB (4); A B; TAB(4); A/B
1ØØ END
```

Izpis na zaslonu bo naslednji:

```
IZRAČUNAL TI BOM VSOTO, PRODUKT IN KOLIČNIK TVOJIH ŠTEVIL!

ŠTEVILO A =
? 5Ø RET
ŠTEVILO B =
? 1Ø  RET

A + B          A  B          A/B
  6Ø           5ØØ            5

B #
```

10.5. Preskoki z računalnikom

Povrnimo se k problemu in programu na sliki 46. Mogoče ste takrat pomislili na to, da vam ni všeč večkratno izpisovanje 2Ø in 3Ø stavka. To lahko rešimo z novim ukazom in sicer GO TO. Ukaz GO TO nam omogoča brezpogojni preskok na katerikoli stavek v programu. Primer:

```
 GOTO

(Pojdi na)  .
            .
            .
      1ØØ PRINT   "
      11Ø PRINT   "
      12Ø GO TO   15Ø
      13Ø ..........
      14Ø ..........
      15Ø ..........
      16Ø ..........
      17Ø ..........
            .
            .
            .
            .
```

Ko računalnik naleti na ukaz GO TO, npr. s številom 150, mu le ta ukazuje, da brezpogojno preskoči vse ukaze v programu (v našem primeru stavka 130 in 140) in nadaljuje v stavku s številko 150.

Primer programa:

LIST

```
1Ø REM   GO TO PRIMER
2Ø PRINT "TO": GO TO 6Ø
3Ø PRINT "ZELO": GO TO 5Ø
4Ø PRINT "PROGRAM": END
5Ø PRINT  "ENOSTAVEN" GO TO 4Ø
6Ø PRINT "JE" GO TO 3Ø
```

RUN

```
TO
JE
ZELO
ENOSTAVEN
PROGRAM
```

No sedaj pa še izboljšana in razširjena verzija programa s slike 46:

```
1Ø PRINT TAB (1);"A";TAB(6);"B";TAB (6);"A+B";TAB(6);"A B"
2Ø PRINT "..........."
3Ø READ A, B
4Ø PRINT TAB (0);A;TAB(3);B;TAB(3);A+B;TAB(3); A B
5Ø GO TO 3Ø
6Ø DATA   5, 1Ø, 15, 2Ø, 25, 3Ø, 35, 4Ø, 45, 5Ø, 55, 6Ø, 65, 7Ø, 75, 8Ø,
          85, 9Ø, 95, 1ØØ
7Ø END
```

Po ukazu RUN, dobimo na zaslonu naslednji rezultat:

A	B	A+B	A B
5	10	15	50
15	20	35	300
25	30	55	750
35	40	75	1400
45	50	95	2250
55	60	115	3300
65	70	itd.	

Poglejmo še naslednji primer programa:

LIST

```
1Ø   REM      NESKONČNA ZANKA
2Ø   PRINT    "PROSIM DVE ŠTEVILI"
3Ø   PRINT    "PRVO ŠTEVILO JE: ":INPUT A: PRINT A
4Ø   PRINT    "DRUGO ŠTEVILO JE:":INPUT B : PRINT B
5Ø   PRINT  :  PRINT
6Ø   PRINT    "NJUNA:"
7Ø   PRINT TAB (7);"–VSOTA JE:"; A+B
8Ø   PRINT TAB (7);"–RAZLIKA JE:";A–B
```

```
9Ø   PRINT TAB (7);"–PRODUKT JE:";A B
1ØØ  PRINT TAB (7);"–KOLIČNIK JE:";A/B
11Ø  PRINT : PRINT : PRINT
12Ø  GO TO 2Ø
13Ø  END
```

RUN

```
PROSIM DVE ŠTEVILI!
PRVO ŠTEVILO JE:
?  2Ø  E
DRUGO ŠTEVILO JE:
?  2   E

NJUNA:
      – VSOTA JE: 22
      – RAZLIKA JE: 18
      – PRODUKT JE: 4Ø
      – KOLIČNIK JE: 1Ø

PROSIM DVE ŠTEVILI!
PRVO ŠTEVILO JE:
                itd.
```

Slika 50

Kot vidimo, nam v tem programu računalnik izračuna vsoto, razliko, produkt in količnik zadanih števil in ko napiše rezultate, preide na stavek 12Ø GO TO 2Ø, ki mu ukazuje, da zopet skoči na stavek 2Ø. Ko je računalnik to opravil, pričenja seveda svoje delo znova. Če podrobno pogledamo program, lahko vidimo, da ubogi računalnik ne bo prišel nikoli do stavka 13Ø, kar pomeni, da bo neprenehoma računal omenjene računske operacije.

Kako ga zaustaviti, ne da bi uničili program? Nekateri računalniki imajo poseben kontrolni znak, npr. C^C itd., drugi se lahko zaustavijo s pritiskom BREAK (seveda v toku izpisa rezultatov), itd. Lahko pa se poslužimo tudi naslednjega postopka (pri HR 84):

– Ko pride računalnik do stavka 3Ø ali 4Ø in ko po ukazu INPUT A čaka na vnos določenega števila, mu namesto številke vtipkamo določen abecedni znak ali niz znakov, kot npr. A, B, CCC, AXY, itd.
  Računalnik znaka ne prepozna kot številka, zato izpiše: VARIABLE NOT FOUND ali ERROR NO.8 IN LINE 3Ø . . . , kar pomeni: Spremenljivke nisem našel oziroma prepoznal.
– Če v programu ni ukaza INPUT, pritisnemo istočasno tipki CTL in C (control C).
– Odtipkamo nov program ali pa z ukazom RUN ponovno startamo istega.

Program po sliki 50 lahko izvedemo tudi tako, da nam računalnik izvede le želeno računsko operacijo. To opravimo z ukazom ON GO TO. Oglejmo si to na naslednjem primeru:

ON GOTO
(Razvejani GOTO)

```
1Ø   PRINT "PROSIM DVE ŠTEVILI!"
2Ø   PRINT "PRVO ŠTEVILO JE:" :INPUT A
3Ø   PRINT "DRUGO ŠTEVILO JE:" INPUT B
4Ø   PRINT "ALI ŽELITE NJUNO:"
5Ø   PRINT "1–VSOTO"
6Ø   PRINT "2–RAZLIKO"
7Ø   PRINT "3–PRODUKT"
8Ø   PRINT "4–KOLIČNIK"
9Ø   PRINT "IZBERITE RAČUNSKO OPERACIJO IN GLEDE"
95   PRINT "NA TO ODTIPKAJTE 1, 2, 3, ALI 4!"
1ØØ  INPUT N
11Ø  ON N GOTO 12Ø, 13Ø, 15Ø, 17Ø
12Ø  PRINT "A + B="; A+B
125  GOTO 1Ø
13Ø  PRINT "A–B="; A–B
14Ø  GOTO 1Ø
15Ø  PRINT "A B="; A B
16Ø  GOTO 1Ø
17Ø  PRINT "A/B="; A/B
18Ø  GOTO 1Ø
19Ø  END
```

Po ukazu RUN, dobimo na zaslonu naslednji potek programa:

```
RUN
PROSIM DVE ŠTEVILI!
PRVO ŠTEVILO JE:
? 56
DRUGO ŠTEVILO JE:
? 7
ALI ŽELITE NJUNO:
1–VSOTO
2–RAZLIKO
3–PRODUKT
4–KOLIČNIK
```

```
IZBERITE RAČUNSKO OPERACIJO IN GLEDE NA TO
ODTIPKAJTE 1, 2, 3, ALI 4!
? 3
A B = 392

PROSIM DVE ŠTEVILI!
                    itd.
```

Kot vidimo, nam je program izvedel le želeno računsko operacijo, to je množenje. Omenjeni ukaz bomo torej uporabljali, ko bomo želeli nadaljevati izvajanje glede na to, kakšen je rezultat dotedanjih operacij. Namesto kontrolne spremenljivke imamo lahko tudi poljuben aritmetični izraz, kot npr. ON SQR(N) GOTO 10,20,30,40. Toda ukaz ON GOTO je razmeroma malo uporabljen pri programiranju. Zato si oglejmo naslednjega.

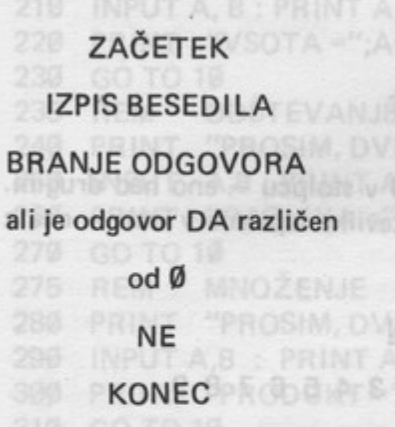
I F . . . . . T H E N
(če . . . potem . . .)

Ukaz IF . . . THEN je zelo uporaben v programiranju, saj nam omogoča pogojne skoke, kot npr.:

To pomeni, da bo računalnik preskočil v stavek 1Ø šele takrat, ko bo izpolnjen pogoj A ≠ Ø kar pomeni A različen od Ø. Ko je A različen od Ø, bo računalnik preskočil v stavek 1Ø. Če ta pogoj ni izpolnjen, računalnik ta stavek preskoči in izvršuje naslednjega.

Za primer in boljšo predstavo, si oglejmo diagram poteka in program zelo enostavne naloge in sicer: Napišimo program, ki na zaslonu izpisuje besedilo: VTIPKAJTE ŠTEVILO, PROSIM in bere vtipkano število, dokler ne vtipkamo števila nič!

Diagram poteka:

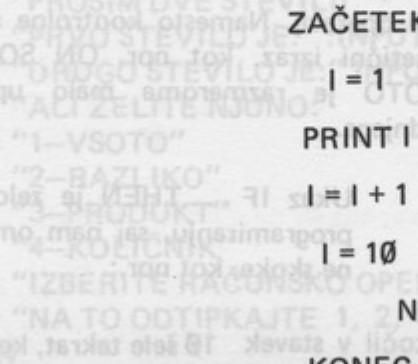



```
1Ø PRINT "VTIPKAJTE ŠTEVILO,PROSIM"
2Ø INPUT A
3Ø IF A≷ Ø THEN (GOTO) 1Ø
4Ø END
```

V 3Ø. stavku smo opazili novi, do sedaj še neomenjen znak ≷ . Seveda jih je v programiranju še več in njihovo ime je: **relacijski operatorji** ali **znaki relacije**. Prikažimo ji, da jih lahko v nadaljevanju uporabimo.

>	večje
=	enako
<	manjše
≤	manjše ali enako
≥	večje ali enako
≠	neenako oz. različno

Poskušajmo napisati program za izpis naravnih števil od 1 do 1Ø. Najprej diagram poteka:



LIST

```
 5 REM   ŠTEVILA OD 1 DO 1Ø
1Ø LET I = 1
2Ø PRINT I
3Ø LET I = I + 1
4Ø IF I = 1Ø THEN 2Ø
5Ø END
```

RUN

```
1
2
3
4
5
6
7
8
9
10

B#
```

Kot vidimo, nam program izpiše števila od 1 do 10 v stolpcu — eno nad drugim. Kako pa bi morali spremeniti program, da bodo številke izpisane v vrsti — ena z drugo? Oglejmo si ta primer!

LIST

```
 5 REM   ŠTEVILA OD 1 DO 1Ø
1Ø LET I = 1
2Ø PRINT I;
3Ø LET I = I + 1
4Ø IF I  = 1Ø THEN 2Ø
45 PRINT
5Ø END
```

RUN

```
1 2 3 4 5 6 7 8 9

B#
```

Torej, spremenili, oziroma dopolnili smo le stavek 2Ø. Kot vidimo, so posamezni znaki "ločil" v programiranju izredno pomembni, zato jih moramo pazljivo in pravilno uporabljati.

Sedaj pa že toliko znamo, da lahko napišemo svoj prvi, nekoliko zahtevnejši program za izračunavanje enostavnih matematičnih operacij tako, da si uporabnik lahko izbere računsko operacijo, ki jo želi izvajati s svojima številom. Veselo na delo!

```
1Ø   REM    PROGRAM ZA IZRAČUNAVANJE ENOSTAVNIH
2Ø   REM        MATEMATIČNIH OPERACIJ
35   PRINT : PRINT
3Ø   REM        SESTAVIL: IVAN GERLIČ, PA MARIBOR 1984
4Ø   PRINT      "ALI ŽELIŠ IZBRANI ŠTEVILI:"
5Ø   PRINT TAB (1Ø);"1–SEŠTETI"
6Ø   PRINT TAB (1Ø);"2–ODŠTETI"
7Ø   PRINT TAB (10);"3–MNOŽITI"
8Ø   PRINT TAB (10);"4–DELITI"
9Ø   PRINT TAB (10);"5–KVADRIRATI"
1ØØ  PRINT "ODGOVORI Z 1,2,3,4, ALI 5!"
11Ø  INPUT X : PRINT X
12Ø  REM        ODLOČITEV – IZBIRA
13Ø  IF X = 1 THEN 2ØØ
14Ø  IF X = 2 THEN 24Ø
15Ø  IF X = 3 THEN 28Ø
16Ø  IF X = 4 THEN 32Ø
17Ø  IF X = 5 THEN 36Ø
18Ø  GO TO 1ØØ
185  PRINT  : PRINT
19Ø  REM    SEŠTEVANJE
2ØØ  PRINT  "PROSIM, DVE ŠTEVILI LOČENI S TIPKO RET!"
21Ø  INPUT A, B : PRINT A; "   ";B
22Ø  PRINT  "VSOTA =";A+B
23Ø  GO TO 1Ø
235  REM    ODŠTEVANJE
24Ø  PRINT  "PROSIM, DVE ŠTEVILI LOČENI S TIPKO RET!"
25Ø  INPUT  A,B : PRINT A; "   ";B
26Ø  PRINT  "RAZLIKA =";A – B
27Ø  GO TO 1Ø
275  REM    MNOŽENJE
28Ø  PRINT  "PROSIM, DVE ŠTEVILI LOČENI S TIPKO RET!"
29Ø  INPUT A,B  : PRINT A; "   ";B
3ØØ  PRINT  "PRODUKT =";A B
31Ø  GO TO 1Ø
315  REM    DELJENJE
32Ø  PRINT  "PROSIM, DVE ŠTEVILI LOČENI S TIPKO RET!"
33Ø  INPUT A, B: PRINT A; "   ";B
```

```
340  PRINT "KOLIČNIK =";A/B
350  GO TO 10
355  REM   KVADRIRANJE
360  PRINT "PROSIM ENO ŠTEVILO!"
370  INPUT C : PRINT C
380  PRINT "KVADRAT ="; C 2
390  GO TO 10
400  END
```

Zanimive in uporabne pa so tudi druge oblike IF stavka, torej pogojnega skoka v programu. Oglejmo si jih!

IF . . . . THEN	IF . . . GO TO	IF . . THEN . . ELSE
– IF A 10 THEN 100	– IF C = 0 GO TO 100	
– IF A = 1 THEN 80		
–IF D = E AND F = 10		

Zanimiv in zelo uporaben je ukaz IF . . . THEN . . . ELSE, zato si oglejmo primer programa!

```
10   PRINT "IZRAČUNAM TI LAHKO KVADRAT ALI"
20   PRINT "KVADRATNI KOREN KATEREGA KOLI ŠTEVILA!"
30   PRINT "PROSIM ŽELENO ŠTEVILO":INPUT N
40   PRINT "ODTIPKAJ ŽELENO OPERACIJO:"
50   PRINT TAB (5);"1–KVADRIRANJE"
60   PRINT TAB (5);"2–KVADRATNI KOREN":INPUT A
70   IF A=1 THEN 80 ELSE PRINT "KVADRATNI KOREN OD";N;"=";N!
          (1/2): GOTO 90
80   PRINT "KVADRAT OD";N;"=";N!2
90   PRINT : PRINT : PRINT : PRINT : PRINT : PRINT
100  GO TO 10
```

10.6. Računalniške zanke, kaj je to?

F O R . . . N E X T
S T E P

V primeru ukaza GO TO smo se srečali z brezpogojnim skokom na določen programski stavek in tudi z **neskončno zanko**, saj se je program izvajal (sl. 50) tako dolgo, dokler ga "nasilno" nismo ustavili oziroma prekinili. Če pa že vnaprej vemo, kolikokrat želimo ponavljati program, imamo v BASIC-u za to izredno ugoden ukaz **FOR . . . NEXT**. Kjer se pojavi ukaz FOR, mora v programu slediti tudi drugi del ukaza, to je NEXT, saj tvorita skupaj zaključeno **zanko**! Vsi programski ukazi v notranjosti zanke, se lahko ponavljajo poljubnokrat!

```
10 .............
20 .............
30 FOR A=0 to 10
```

```
4Ø . . . . . .
5Ø . . . . . .
6Ø NEXT A
7Ø . . . . . .
```

Slika 51

Na ukaz FOR je vezan še ukaz STEP, kar pomeni KORAK. Izgled takšnega stavka bi bil npr.:

```
1Ø FOR X=Ø TO 5Ø STEP 1Ø
```

To pomeni: Za X =Ø do 5Ø s korakom 1Ø – torej ukaz, ki pravi računalniku, da naj šteje od Ø do 5Ø po 1Ø. **Če za ukazom FOR ne zapišemo in določimo koraka (npr. STEP 1Ø), potem za računalnik to pomeni, da naj šteje s korakom 1.**

Primer enostavnega programa:

LIST

```
1Ø FOR X=Ø TO 5Ø STEP 1Ø
2Ø PRINT X
3Ø NEXT X
4Ø END
```

RUN

```
 1
 1Ø
 2Ø
 3Ø
 4Ø
 5Ø
B #
```

Kadar želimo izpis v ravnini, pa bo program izgledal tako:

LIST

```
1Ø FOR X = Ø TO 5Ø STEP 1Ø
2Ø PRINT X;
3Ø NEXT X
4Ø STOP
```

RUN

```
Ø  1Ø  2Ø  3Ø  4Ø  5Ø
B #
```

Korak je lahko tudi negativen, kar pomeni, da v našem programu pišemo števila od 5Ø proti Ø. Oglejmo si to:

LIST

```
1Ø FOR X = 5Ø TO Ø STEP – 1Ø
2Ø PRINT X
3Ø NEXT X
4Ø END
```

RUN

```
 5Ø
 4Ø
 3Ø
 2Ø
 1Ø
 Ø
B
```

Če želimo korak 1, lahko ukaz STEP izpustimo. Izpustiti pa ga ne smemo takrat, kadar je korak negativen (−1)!

Sedaj nam ni več težko rešiti problema na sliki 42 po enostavnejši poti. Oglejmo si tak program, ki bo krajši in bo prav tako izrisal poševno črto iz trez zvezdic (***).

"***"

LIST

```
10 FOR X=1 TO 10
20 PRINT TAB (X);"***"
30 NEXT X
40 END).
```

RUN

Pa je krajši, ali ne?

Bi znali s tem programom narisati dve poševni črti, eno iz treh zvezdic, drugo iz treh osmic? Nič lažjega, ali ne?

LIST

```
10 FOR X=1 TO 10
20 PRINT TAB (X);"  *  *  *";"  *  *  *";"888"
30 NEXT X
40 END
```

RUN

B

Z že znanim ukazom FOR . . . NEXT naredimo še en zanimiv program in sicer tabelo kvadratov, kubov in kvadratnega korena števil od 1 do 20. Manj poznan nam je samo še znak za kvadratni koren naravnih števil. Ukaz za kvadratni koren v BASIC-u je: SQR(X).

S Q R (X)
(kvadratni koren)

LIST

```
10 PRINT TAB (1);"x";TAB (5);,,X 2";TAB(5);"X 3";TAB(5);"SQR(X)"
20 FOR  = 35 TO 0 STEP (−2)
30 PRINT "=";
40 NEXT : PRINT
45 FOR X=1 TO 20
50 PRINT TAB(0);X;TAB(3);X 2;TAB(5);X 3;TAB(5);SQR(X)
60 NEXT X
70 END
```

RUN

X	X 2	X 3	SQR(X)
1	1	1	1
2	4	8	1.41421

3	9	27	1.73205
4	16	64	2

itd.
B

Program, ki je pred nami, ni zanimiv samo po matematično uporabni plati, temveč tudi zato, ker pri podrobnejšem pregledu vidimo, da gre za dve zaporedni zanki. Sedaj pa še vprašanje. Ali lahko obstaja "zanka v zanki" in ali se lahko zanke prepletajo – križajo? Prvi odgovor je DA, drugi NE! Znotraj programa lahko obstaja več zank, toda le-te morajo biti tako razporejene, da se ne prepletajo.

```
FOR A  ... itd.
FOR B  ... itd.

NEXT B
NEXT A
```

PRAVILNO

```
FOR A  ... itd.
FOR B  ... itd.

NEXT A
NEXT B
```

NAPAČNO

Slika 52

Poglejmo si primer dvojne zanke:

LIST

```
10 FOR N=1 TO 3
20 PRINT "NOTRANJA ZANKA ŠT.:";N
30 FOR X=1 TO 10
40 PRINT " ";
50 NEXT X
60 PRINT
70 NEXT N
80 END
```

RUN

```
NOTRANJA ZANKA ŠT.:1
NOTRANJA ZANKA ŠT.:2

NOTRANJA ZANKA ŠT.:3

B #
```

Sedaj pa še primer programa s trojno zanko:

LIST

```
10 FOR Z=1 TO 3
20 PRINT "Z= ";Z
30 FOR  L=1 TO 4
40 FOR N=1 TO 10
50 PRINT ", "
60 NEXT N
65 PRINT
70 NEXT L
80 NEXT Z
90 END
```

RUN

```
Z=1

Z=2

Z=3

B #
```

Za preverjanje osvojenega znanja samostojno razrešite naslednji program in vrišite zanke!

LIST RUN

```
10 FOR B=1 TO 3
20 FOR L=1 TO 2 B
30 FOR N=1 TO 2 (L+B)
40 PRINT " ";
50 NEXT N
60 PRINT
70 NEXT L
75 PRINT
80 NEXT B
90 END
```

9.7. Naredimo program za mlajšega bratca

Sedaj pa si zadajmo še naslednjo nalogo: želimo izdelati program za mlajšega bratca, ki ga bo vodil v seštevaju enomestnih števil. Nič lažjega, boste rekli, in napisali naslednji program:

```
5   DIM X(5)
10  PRINT  TAB (10);"VAJA SEŠTEVANJA"
20  PRINT "5+6=" : INPUT X
30  IF X=11 THEN GO TO 60
40  IF X  11 THEN PRINT "NEPRAVILNO, ŠE ENKRAT!"
50  GO TO 20
60  PRINT "DA, PRAVILNO!"
70  PRINT  "7+3=" :INPUT X
80  IF X = 10 THEN GO TO 110
90  IF X   10 THEN PRINT "NEPRAVILNO, ŠE ENKRAT!"
100 GO TO 70
110 PRINT  "DA, PRAVILNO!"
120 PRINT  "4+9=" : INPUT X
130 IF X =13 THEN GO TO 160
140 IF X = 10 THEN PRINT "NEPRAVILNO, ŠE ENKRAT!"
150 GO TO 120
160 PRINT "DA, PRAVILNO!"
170 PRINT "7+2=" : INPUT X
180 IF X = 9 THEN GO TO 210
190 IF X = 10 THEN PRINT "NEPRAVILNO, ŠE ENKRAT!"
200 GO TO 170
210 PRINT "DA, PRAVILNO!"
220 PRINT "8+8=" : INPUT X
230 IF X = 16 THEN GO TO 260
240 IF X 10 THEN PRINT "NEPRAVILNO, ŠE ENKRAT!"
250 GO TO 220
```

```
26Ø PRINT "DA, PRAVILNO! DOVOLJ JE SEŠTEVANJA. ADIJO!"
27Ø END
```

D I M
(Rezerviranje polj)

Program je resnično enostaven in menda vam ga ni težko napisati in razumeti. Toda nepoznan nam je še ukaz DIM v stavku 5! Kaj pomeni? Ukaz DIM rezervira prostor v spominu računalnika za določeno število spremenljivk. Oglejmo so to kar na našem primeru:

DIM X (5)

kar pomeni računalniku, da mora v spominu rezervirati prostor za niz z imenom X, ki ima pet elementov, in sicer: X_1, X_2, X_3, X_4 in X_5. V našem primeru smo to uporabili zato, da smo si rezervirali pet mest za pet rezultatov (INPUT X) naših petih računov seštevanja. Seveda tega za manjše število spremenljivk ni potrebno narediti, saj nam računalnik v primeru, ko z DIM stavkom ne določimo, sam določi največjo vrednost indeksov in sicer 10.

V računalništvu se naš primer imenuje **enodimenziolano polje** ali vektor. Ko je polje določeno, lahko uporabljamo njegove elemente prav tako, kot ostale spremenljivke. Vrednost indeksov moramo seveda navesti in sicer v oklepajih (DIM X(5)) X(1), X("),X(3) . . .).

Seveda lahko na podoben način določimo tudi **dvodimenzionalno polje ali matriko**. Primer:

DIM X (1,2)

Stavek tvori dvodimenzionalno polje realnih spremenljivk (matriko) dimenzije 2 x 3:

X(0,0)	X(0,1)	X(0,2)
X(1,0)	X(1,1)	X(1,2)

Povrnimo se zopet na naš program elementarne matrike za mlajšega bratca. Program bo zanj zanimiv mogoče samo dva do trikrat, pozneje pa ne več, ker si bo rezultate računov seštevanja že zapomnil. Program moramo torej narediti tako, da bo računalnik ob vsakem računu podal našemu bratcu novi — naključno izbrani števili. To je mogoče izpeljati, le poznati je potrebno dva nova ukaza, in sicer RND in INT. Oglejmo si ju pobliže, nato pa izpopolnimo naš program!

R N D
(generiranje naključnih števil)

Ukaz RND pomeni računalniku, da prične z generiranjem naključnih števil med 0 in 1; torej nam bo računalnik dajal niz izmišljenih — naključnih števil, kot npr. naslednji primer:

LIST

```
1Ø FOR N=1 TO 1Ø
2Ø PRINT RND(Ø)
```

RUN

```
 .845Ø57
 .334546
```

```
3Ø NEXT N                    .643649
4Ø END                       .774261
                             .210709
                             itd.
                             B
```

Kot vidimo, so števila resnično med Ø in 1. Za računalnik HR 84 so značilne naslednje variante ukaza RND(X):

X = Ø vsakokrat se generira novo slučajno število med Ø in 1
X 0 generira se isto slučajno število
X 0 vrne zadnje generirano število

Običajno uporabljamo varianto X = Ø.

Toda, kot vidimo, nam RND (Ø) generira števila med Ø in 1 in to na šest decimalk natančno (npr. 0.210709). S takimi števili (decimalnimi) naš mlajši bratec še ne zna računati; decimalna števila moramo spremeniti v cela. To naredimo z ukazom INT, ki ima obliko: INT(X)

I N T
(zaokrožitev realnih števil)

Primeri:

INT (-2) = -1
INT (4Ø) = 4Ø
INT (Ø) = Ø
INT (-4.2) = -5
INT (-12) = 12
INT (.3) = Ø
INT (1.3) = 1
INT (2.8) = 2

Sedaj nam ni več problem zapisati našega programa za generiranje desetih enomestnih naključnih števil. Oglejmo si ga:

```
1Ø FOR N = 1 TO 1Ø
2Ø PRINT INT (RND(Ø) * 1Ø)
3Ø NEXT N
4Ø END
```

Razložimo 2Ø. stavek:

RND(Ø) * 1Ø vsako naključno število pomnožimo z 1Ø, da dobimo npr.: 0,372816 * 1Ø = 3.72816

INT (RND(Ø) * 1Ø) na zgornji način pridobljeno naključno število zaokrožimo v celo število kot npr.:

INT (3.72816) = 3

Že znamo dovolj, da izboljšamo program seštevanja za mlajšega bratca. Oglejmo si ga!

```
 5
 1Ø
 2Ø                              1Ø)
 3Ø                              1Ø)
 4Ø
 5Ø
 6Ø  INPUT X9Ø IF X = C GO TO 11Ø ELSE PRINT "ŠE ENKRAT! NEPRAVILNO!
 7Ø
 8Ø                          GO TO 7Ø
1ØØ
11Ø
12Ø
13Ø
14Ø
15Ø
16Ø
```

Razumete program! Mislim, da vam ne dela težav. Toda tudi ta program nam še ni všeč in to oblikovno. Želimo, da se nam po vsakem računu zapis predhodnega računa zbriše, izpis "BRAVO '" PRAVILNO!" pa ostane samo toliko časa, da ga lahko preberemo!

GO SUB...
...RETURN

Nič lažjega, če se spoznamo z novim ukazom, ki nam omogoča večkratno uporabo istega **podprograma** – ukaz GO SUB ... RETURN. Naš program z nekaj "kozmetičnimi" izboljšavami, bo sedaj izgledal takole:

```
 5 DIN X (5)
1Ø PRINT TAB (8) ; "* * * * VAJA SEŠTEVANJA* * * *
15 PRINT : PRINT : PRINT
2Ø PRINT "KAKO PA TI JE IME? ? " : INPUT A$
25 PRINT "POZDRAVLJEN-A" ; A$ ; " ! VESELI ME, DA SVA SE"
3Ø PRINT "SPOZNALA ! ! !"
35 PRINT "MENI JE IME HR–84!" : PRINT
4Ø PRINT "SEDAJ PA POGLEJVA, KAKO ZNAŠ SEŠTEVATI!"
45 PRINT : PRINT
5Ø FOR N=1 TO 5
55 LET A = INT (RND (Ø) * 1Ø)
6Ø LET B = INT (RND (Ø) * 1Ø)
65 LET C = A + B
7Ø PRINT A; " + " ; B; " = ";
8Ø INPUT X
9Ø IF X = C GO TO 12Ø ELSE PRINT "ŠE ENKRAT"; A$ "! NEPRAVILNO !!"
1ØØ GO TO 7Ø
```

```
11Ø PRINT "BRAVO"; A$; "!!! PRAVILNO !"
12Ø FOR Z = 1 TO 1500 : NEXT Z
125 GO SUB 180
13Ø NEXT N
14Ø FOR Q = 0 TO 39 : PRINT "*"; : NEXT Q
15Ø PRINT "ADIJO" ; A$; ", IN PRIDI ŠE KAJ !!"
16Ø FOR W = 0 TO 39 : PRINT "* ; : NEXT W
17Ø END
18Ø FOR M = 1 TO 24 : PRINT : NEXT M : RETURN
```

Oglejmo si sedaj novitete tega programa.

1. V uvodu, to je od tavka 2Ø do stavka 4Ø, smo dodali pozdravni del, ki te pozdravi, se predstavi in nato v vseh komentarjih uporablja vnešeno ime (glej stavke 9Ø, 11Ø in 15Ø),
2. V staku 12Ø smo pripravili pavzo, da podprogram, ki prične v 125. stavku počaka, da lahko preberemo pohvalo o pravilnosti rešitve! To smo dosegli s tem, da smo v tem času dali računalniku "šteti" od 1 do 150.
3. V stavku 125 pa se že prične ukaz (GO SUB), ki pove računalniku, da preskoči v programski stavek 18Ø, ki vsebuje celoten podprogram za brisanje zaslona med posameznimi novimi računi. To smo dosegli s tem, da smo ukazali računalniku, naj od 1. do 24. vrstice piše prazne vrstice. Z ukazom RETURN smo poslali računalnik nazaj na mesto v glavnem programu, kjer je prej ostal. Tako se torej program nadaljuje v 13Ø stavku!

Z GO SUB . . . RETURN lahko torej na različnih mestih glavnega programa uporabimo podprogram (podprogram lahko uporablja iste spremenljivke kot glavni program) z določenim številom stavkov, ki bi jih morali drugače večkrat pisati. Zelo uporabno, ali ne?

```
150 FOR N = 1 TO 2
160 PRINT "   00   00   00 00     00   00      00" : NEXT N
170 FOR N = 1 TO 2
180 PRINT "  00   00   00   00   00   00      00"
190 NEXT N
200 PRINT "   00   00   00    00   00   00      00"
210 FOR N = 1 TO 2
220 PRINT "   00   00   00   00    0000000      00" : NEXT N
230 GO SUB 380
240 GO SUB 390
250 PRINT : PRINT : PRINT
260 GO SUB 400
265 PRINT : PRINT
310 PRINT B (24); "00" : PRINT
320 PRINT "      0 00 0 0 000 000    00 0 000 000"
330 PRINT "      0 0  0 0 0 0 0 0    0  0 0 0 0 0"
340 PRINT "      0 00 00  000 000    00 0 000 0 0"
350 PRINT "      0  0 0 0 0 0 0 0     0 0 0   0 0"
360 PRINT "      0 00 0 0 0 0 0 0    00 0 0   000"
370 PRINT : PRINT : PRINT
375 FO SUB 400
377 GO SUB 380
378 GO SUB 390
379 GO TO 10
380 FOR X = 0 TO 23 : PRINT : NEXT Z : RETURN
400 FOR Q = 1 TO 3
410 FOR W = 0 TO 39
420 PRINT "*";
430 NEXT W : NEXT Q
440 RETURN
```

Kratka razlaga — komentar programa:

- stavki od 10 do 220 izrišejo: HR 84 iz črk O;
- stavek 230 pokliče časovni podprogram (zadrži izpis HR 84 nekaj deset sekund)
- stavek 240 pokliče podprogram za izbrisanje zaslona
- stavek 250 omogoči tri prazne vrste
- stavek 260 pokliče podprogram za izrisovanje treh črt z "*";
- stavki od 265 do 360 izrišejo zapis: ISKRA ŠIPO
- stavek 370 omogoči tri prazne vrste
- stavek 375 pokliče podprogram za izris treh črt z "*"
- stavek 377 pokliče časovni podprogram
- stavek 378 pokliče podprogram za izbrisanje zaslona
- stavek 379 tvori neskončno zanko izpisa HR 84 in ISKRA ŠIPO; zaustavimo ga lahko istočasno le z istočasnim pritiskom tipk CTL in C!
- stavek 380 časovni podprogram
- stavek 390 podprogram za izbrisanje zaslona
- stavki 400 do 440 podprogram za izrisanje treh črt z "*".

10.8. Za konec pa še

Tako, prišli smo do konca naših prvih korakov v BASIC-u z računalnikom. Z znanjem, ki ste ga pridobili, lahko napišete že kar dobre programe in upamo, da jih tudi boste, saj je namen te knjižice dati najosnovnejše pojme o računalništvu in jeziku BASIC, s tem pa vas motivirati za nadaljnje delo na tem zanimivem področju. Mnogo boste morali še spoznati, toda prve, kar trdne korake v računalniški svet ste že naredili, še posebej, če ste vse programe preverili z računalnikom. Pri nadaljnjem delu vam želimo še mnogo uspehov in predvsem prijetno preživetega časa z vašim novim prijateljem RAČUNALNIKOM HR 84!

Za slovo pa še program za vajo in preverjanje osvojenega znanja!

1. igra: zadevanje števil

```
  5 REM * * * IGRA: ZADEVANJE ŠTEVIL * * *
 10 REM * * * PRIREDIL: IVAN GERLIČ PA MARIBOR * * *
 20 DIM I(15) : DIM S (15)
 25 DIM X (15)
 27 GO TO 1120
 30 PRINT "* * * * IGRA : ZADEVANJE ŠTEVILA * * * * : PRINT : PRINT
 40 PRINT TAB ! (10) ; "NAPOTKI:"
 50 PRINT "RAČUNALNIK SI BO IZMISLIL ŠTEVILO MED"
 60 PRINT "0 IN 1000. TVOJA NALOGA JE, DA UGOTOVIŠ"
 70 PRINT " TO ŠTEVILO TAKO, DA 15-KRAT IZBIRAŠ"
 80 PRINT " ŠTEVILO, ZA KATEREGA MISLIŠ, DA SI GA"
 90 PRINT " JE RAČUNALNIK IZMISLIL! "
100 PRINT "RAČUNALNIK TI BO ODGOVORIL ALI JE TVOJE "
110 PRINT "ŠTEVILO PREVELIKO ALI MEJAHNO!!"
120 PRINT : PRINT
130 PRINT "POZOR, PRIČENJAVA ! ! !"
140 LET X = 1
150 GO SUB 1000
160 LET P = N
170 FOR I = 1 TO 15
180 PRINT I; ".-IZBIRAJ ŠTEVILO!"
190 INPUT S
200 X = X + 1
210 IF S = P GO TO 300
220 IF S P GO TO 260
230 PRINT TAB (7); "ŠTEVILO JE VEČJE OD"; S : PRINT
240 GO TO 270
250 GO TO 310
260 PRINT TAB (7): "ŠTEVILO JE MANJŠE OD" ; S : PRINT
270 NEXT I
275 FOR N = 1 TO 3 : PRINT SHR $(7) ; "  ": NEXT N
280 ORUBT "NA ŽALOST SI ŽE IZKORISTIL SVOJIH 15" : PRINT
285 PRINT "POSKUSOV! ŠTEVILO, KI SI GA ISKAL JE: " ; P : PRINT
290 GO TO 310
```

```
300 FOR N = 1 TO 15 : PRINT CHR$ (7) ; " ": NEXT IN
305 PRINT "BRAVO !!!! UGOTOVIL SI ŠTEVILO"; P; " V ": PRINT
307 PRINT " "; I; ". POSKUSIH !!" : PRINT
310 PRINT "ŽELIŠ POSKUSITI ŠE ENKRAT (DA/NE)?
320 INPUT A$
330 IF A$ = "DA" GO TO 1120 ELSE PRINT" ČE NE PAČ NE!
    PA KDAJ DRUGIČ!"
340 END
1000 LET N = INT (RND( = ) * 1000)
1010 RETURN
1120 FOR Q = 0 TO 23 : PRINT : NEXT Q
1130 GO TO 30
```

11. ABECEDNI PREGLED ZNAČILNIH UKAZOV JEZIKA BASIC

ABS	Absolutna vrednost določene spremenljivke: izračuna absolutno vrednost števila ali izraza (npr. .3.14 3)
ADV	Npr. ADV(x) vrne naslov spremenljivke x v BASIC RAM.
A$	Spremenljivka, ki označuje niz znakov – niz (A$ = GOGI . . .)
ACS	Arkus kosinus (v radianih!)
AND,OR,NOT	Logične operacije IN, ALI, NE
ASC	Zankovna funkcija, ki določa ASC II zaporedno število. Vrnjena vrednost funkcije je ASCII številčna vrednost prvega znaka v nizu (npr. ASC(B$))
ASN	Arkus sinus (v radianih!)
AT	Pozicija na zaslonu za naslednji ukaz PRINT
ATN	Arkus tangens (v radianih!)
BIN	Alternativna notacija za števila: za BIN sledi niz Ø in 1, ki predstavlja dano število dvojiški obliki
BREAK	Prekinitev izvajanja programa
BEEP	Ukaz za izvajanje zvočnega efekta (npr. BEEP x, y x = trajanje signala v sekundah, y = višina poltonov nad C–jem ali pod, če je y negativen). Nekateri računalniki imajo namesto ukaza BEEP ukaz SOUND s podobno funkcijo
BORDER	Odreja barvo robu zaslona
BRIGHT	Odreja „svetlobo" znaka, ki ga odtipkamo
CIRCLE	Riše krog (pa tudi lok), kot npr. CIRCLE x, y, z krog s središčem v x, y in polmerom z, itd.
CHR$	Generiranje števila ali izraza, ki ga predstavlja. Npr. CHR$ (65) A. Torej je to znakovna funkcija, ki določa ASCII znak, iz njegove številke
CLEAR	Briše vse spremenljivke in s tem sprosti pomnilnik

CLS	Briše zaslon — Display-File
CONT	Cont — continue: nadaljuje prekinjeni program
COS	Kosinus (v radianih!)
COPY	Tiska oziroma prepiše vsebino z zaslona na tiskalnik
DATA	Označuje začetek spiska podatkov, ki jih naj računalnik bere, ko pride do ukaza READ
DEF IN	Definicija samostojno definirane funkcije
DIM	Rezervira prostor v spominu, za niz spremenljivk. Npr. DIM C(3) pomeni računalniku, da mora rezervirati prostor v spominu za niz z imenom C in imeti 3 elemente, ki se bodo imenovali C_1, C_2 in C_3. Pri večini računalnikov lahko na isti način dimenzioniramo tudi niz števil
DRAW	Risanje črte (lahko je: DRAW x, y ali DRAW x, y, z)
EDIT	Omogoča brisanje in popravljanje želenih programskih ukazov
END	Fizični konec programa — ko računalnik pride do njega, preneha z delom. Je vedno zadnji stavek v programu.
ENTER	Ukaz za prenos. Pri različnih računalnikih je lahko različen, kot npr.: RETURN, NEWLINE
ERR, ERL, EOF	Funkcije, ki nam sporočijo vrsto napake.
EXIT	Izhod iz BASIC-a v monitor računalnika.
	FOR X = 1 TO 10 STEP 2
EXP	e^x
FAST	Startanje hitrejšega izvajanja programa
FIX	Funkcija za računanje celega dela realnega števila (npr. FIX (6.37) —6)
FLASH	Določa, ali se bodo vnešeni znaki svetlikali — ali bodo utripali ali ne.
FOR . . .NEXT	Skupaj s TO in NEXT obdeluje določen del programa. Sem spada še STEP, ki določuje korake. Običajna oblika npr.
	: FOR X = 1 TO 10 STEP 2
	.
	.
	.
	NEXT X
FN (FUNCTION)	FN za katerim sledi črka, kliče funkcijo, ki jo je definiral uporabnik (glej DEF).
GRAPH (GRAPHICS)	Omogoča poziv grafičnih simbolov.
GOSUB	Klic podprograma; npr. GOSUB 120 ukazuje računalniku, da začne izvajati podprogram, ki se začenja s stavkom 120.
GO TO	Brezpogojni skok na določeni programski ukaz (npr. GO TO 120)

HEX — Funkcija, ki pretvori šestnajstiški znakovni niz v njegov desetiški ekvivalent

IF . . THEN — Pogojni skok na določeni programski ukaz (npr. IF A=B THEN PRINT A ali GO TO 2Ø ali ELSE PRINT B. Razlaga za ELSE za gornji primer: če je izpolnjen pogoj se izpolni ukaz za THEN, če pa ni se izvede ukaz za ELSE).

INKEY $ — Čitaj ukaz s tipkovnice (samo enega)

INPUT — Čitanje enega ali več znakov oz. nizov)navadno ločenih z ENTER, RETURN . . .). To je torej ukaz za računalnik, da prekine potek programa in čaka na vnos vrednosti spremenljivk.

INPUT LINE — Vsa vtipkana vrsta se prebere in shrani v zadano spremenljivko (npr. INPUT LINE A$). Včasih tudi LINPUT.

INVERGE — Kontrola inverzije barv (INVERGE Ø — znaki se tiskajo v normalni obliki z barvo črnila na barvi papirja; INVERGE 1 — znaki se tiskajo v obrnjeni — inverzni obliki).

INK — Daje barvo znakom, ki jih odtipkamo.

INT — INT — INTEGER: celoštevilčni del nekega števila (vedno zaokrožen navzdol). Npr. INT (PI) = 3, ker je PI = 3,14

LEFT $ — Izvajanje prvih nekaj črk iz besede, ki je navedena pod spremenljivko, npr. A$ (A$ = GORAZD; LEFT $ (A$, 3) GOR)

LEN — Dolžina niza — stringa (določitev). Funkcija nam vrne dolžino niza kot celoštevilčno vrednost.

LET — Določanje vrednosti spremenljivke. Pri večini računalnikov ima ukaz LET X = 2 isti pomen kot X = 2.

LINE — Spajanje točk, katerih koordinate so podane (izriše npr. premico, krivuljo . . .). Ta ukaz imajo le kvalitetnejši računalniki.

LIST — Izpis programa na zaslon.

LLIST — Izpis programa na tiskalnik.

LN — Naravni logatirem (baza e)

LOAD — Včitavanje programa iz zunanjega pomnilnika (diskete, kasete . . .). Npr.: LOAD " " ali pa LOAD "BASEN" (BASEN je ime programa, ki ga pri nekaterih računalnikih moramo navesti, pri drugih pa zadoščajo že narekovaji za snemanje programa iz kasete. Pri disketi je obvezno zapisati ime programa!).

LPRINT — LPRINT Tiskalnik kot izhodni element.

MERGE — Včitavanje programa iz zunanjega pomnilnika. Ne uniči vrst in različic stavkov starega programa.

MID $ — Izvaja zadnjih nekaj črk besede, ki je navedena pod spremenljivko, npr. A$ (A$ = GORAZD; ŠROMT MID $ (A$, 3) AZD).

NEW — Izbriše delovni spomin in spremenljivke — torej izbriše stari program in pripravi spomin za vnašanje novega.

ON GO TO — Pogojni skok na določene programske vrstice (npr. ON 5Ø GO TO 1ØØ, 15Ø, 2ØØ)

ON GO SUB Pogojni klic podprogramov glede na vrstice (npr. ON A GO SUB
ON EROR GO TO
ali
ON EROR THEN: Programska kontrola napak ob izvajanju. V primeru napake se program nadaljuje v podani vrstici.

OPEN Stavek za odpiranje datoteke.

PAPER Podobno kot INK, le da nadzira barvno ozadje zapisa

PAUSE Prekine delo programa za čas (sekunde), ki smo ga zadali, ali pa do tedaj, ko pritisnemo eno od tipk tipkovnice.

PEEK Vsebina specifičnega pomnilniškega mesta.

PI 3.14159265 . . .

PLOT Izris črnega ali barvnega kvadratka po koordinatah, ki smo jih zadali.

POKE Premika BYTE iz enega mesta na drugega.

POS Vrne položaj konca trenutnega izpisa tekoče vrste.

PREND Prikaže zasedenost spomina s programom in mejo še prostega prostora zanj.

PRINT Izpis navedene vrednosti; spremenljivke, niza itd.

RANDOMIZE RANDOMIZE – RND: generiranje slučajnih števil med Ø in 1.

READ Vezan na ukaz DATA.

REM V program, omogoča vnos komentarjev, ki pa se v toku programa ne izvajajo.

REND Nastavi novo končno mejo razpoložljivega pomnilnika.

RENUM (RENUMBER) Preštevilčenje stavkov programa, tako da dobijo številke 10, 20, 30 . . . Nekateri računalniki imajo ukaz RESEQ namesto RENUM.

RESTORE Omogoča, da naslednji stavek READ prične spet čitati podatke z začetka seznama prvega DATA stavka v programu.

RESUME Odpravi stanje napake, po katerem se je znašel program po napaki.

RETURN Vezan na ukaz GO SUB.

ROOM Vrne število prostih zlogov do konca razpoložljivega pomnilnika.

RUBOUT Briše en znak v smeti levo od kazalca.

RUN Pričetek programa od prvega stavka dalje.

SAVE Shranjevanje programa na elemente zunanjega pomnilnika. Navadno mu sledi ime programa, npr.: SAVE "BASEN"

SCROLL Premaknitev vsebine na zaslonu za eno vrsto navzgor.

SGN — Predznak specifične spremenljivke. Daje 1, če je x > Ø; Ø, če je x = Ø in – 1, če je x < Ø.

SIN — Sinus kota (v radianih!)

SLOW — Upočasni izvajanje programa.

SPACE $ — Funkcija, ki nam generira zadano število praznih mest (SPACE $ (N)).

SPC — Npr. SPC (A) povzroči izpis A presledkov v stavku PRINT.

SQR — Kvadratni koren

STEP — Korak v ukazu FOR . . . NEXT

STOP — Logični konec programa. Za razliko od ukaza END je lahko STOP kjerkoli v programu. Navadno se uporablja kot preventiva za nezaželen tok izvajanja programa.

STR $ — Spreminjanje števila v niz – string. Npr. STR $ (X) pretvori število x v znakovni ASC II niz, primeren za izpis.

STRING $ — Funkcija za generiranje enakih znakov (oblika: STRING $ (a, b)).

TAB — Tabulatorska funkcija: ukaz je vezan na PRINT in sicer: TAB (X); " ", ki premakne zapis na zaslonu (ali tiskalniku) za toliko, kot ga določa zapis v oklepajih.

TAN — Tangens kota (v radianih!).

UNPLOT — Inverzna funkcija od PLOT

USR — Poziv podprograma s strojnim jezikom – omogoča uporabniku sklicati iz programa BASIC podprogram v strojni kodi (assemblerju), ki ga vsebuje sistem (monitor), ali pa ga je izdelal sam.

VAL — Spreminja niz – string v število. Npr. VAL (X$) pretvori znakovni niz X$ z ASCII zapisom števila v vrednost števila VAL.

VAL$ — Spreminja število v niz – string.

VERIFY — Enako kot LOAD, le da se podatki ne shranjujejo v RAM, temveč se le primerjajo s tistimi, ki so že v njemu. Uporanljamo ga za verificiranje shranjenih programov v zunanji pomnilnik.

12. LITERATURA

1. BRATKO Ivan, RAJKOVIĆ Vladislav, Uvod v računalništvo, DZS Ljubljana 1981
2. CAREY David, How it works . . . Ladybird Books LTD 1979
3. CURRAN Susan, CURNOW Ray, First Steps in Basic, Windward 1980
4. CURRAN Susan, CURNOW Ray, Learning with your computer, Windward 1980
5. DWYER Thomas, CRITCHFIELD Margot, Basic and the Personal Computer, Addison – Wesley Publishing Company 1978
6. ĐURIĆ Branislav, Mini in mikro računari, Tehnička knjiga Beograd 1980
7. FRIEDRICHS G., SCHOFF A., Mikroelektronika in družba – v dobro ali zlo, DZS Ljubljana 1983
8. GERLIČ Ivan, ABC računalništva, ZOTKS Ljubljana 1984
9. GRUNDLER Darko, Uvod v Mikroprocesore, Tehnička knjiga Zagreb 1982
10. HARTNELL T., Programme zum Lernen und Spielen, SYBEX, Düsseldorf 1983
11. HICKMAN Ian, Get More from Your Personal Computer, Newnes Technical Books, London
12. Jugoslovanski standard DK 681.3:001.4 (JUS I.AO.010/1983, JUS I.AO.012/1984, JUS I.AO.013/1984, JUS I.AO.015/1983, JUS I.AO.022/1983, JUS I.AO.024/1982)
13. KUŠČER Samo, Računalniški slovarček, Življenje in tehnika 4/1982
14. RELJIĆ Željko, MAKANEC Branimir, BASIC – kompjuterski jezik, Tehnička knjiga Zagreb.
15. REVIJI: BIT št. 1 in 2/1984 in Moj mikro št. 1 in 2/1984
16. RIBARIČ Slobodan, Arhitektura mikroprocesora, Tehnička knjiga, Zagreb 1982
17. RISTOVIĆ Dejan, Računari u vašoj kući, Specialno izdanje časopisa Galaksija, Beograd 1984
18. SCHÄRF Julius, Basic für Aufäanger, R. Oldenbourg Verlag, Wien 1977
19. STOJKOVIĆ V., TOSIĆ D., Zbirka zadataka iz programiranja – programski jezik BASIC, PFV Beograd 1982
20. ŠPILER Jure, Basic, Samozaložba, Ljubljana 1984
21. ŠUHEL Peter, VIRANT Jernej, Mikroračunalnik, DDU Univerzum, Ljubljana 1978
22. VICKERS Steven, Basic programming ZX Spectrum, Sinclair Research Ltd., Cambridge 1983
23. WOLTERS F. Martin, Ključ za kompjuter, ZAK Beograd.

12. LITERATURA

1. BRATKO Ivan, RALKOVIČ Vladislav, Uvod v računalništvo, DZS Ljubljana 1981
2. CAREY David, How it works . . . Ladybird Books LTD 1979
3. CURRAN Susan, CURNOW Ray, First Steps in Basic, Windward 1980
4. CURRAN Susan, CURNOW Ray, Learning with your computer, Windward 1980
5. DWYER Thomas, CRITCHFIELD Margot, Basic and the Personal Computer, Addison – Wesley Publishing Company 1978
6. ĐURIĆ Branislav, Mini in mikro računari, Tehnička knjiga Beograd 1980
7. FRIEDRICHS G., SCHORF A., Mikroelektronika in družba – v dobro ali zlo, DZS Ljubljana 1983
8. GERLIČ Ivan, ABC računalništva, ZOTKS Ljubljana 1984
9. GRUNDLER Darko, Uvod v Mikroprocesore, Tehnička knjiga Zagreb 1982
10. HARTNELL T., Programme zum Lernen und Spielen, SYBEX, Düsseldorf 1983
11. HICKMAN Ian, Get More from Your Personal Computer, Newnes Technical Books, London
12. Jugoslovanski standard DK 681.3:001.4 (JUS I.AO.010/1983, JUS I.AO.012/1984, JUS I.AO.013/1984, JUS I.AO.016/1983, JUS I.AO.022/1983, JUS I.AO.024/1982)
13. KUŠČER Samo, Računalniški slovarček, Življenje in tehnika 4/1982
14. RELJIĆ Željko, MAKANEC Branimir, BASIC – kompjuterski jezik, Tehnička knjiga Zagreb.
15. REVIJI: BIT št. 1 in 2/1984 in Moj mikro št. 1 in 2/1984
16. RIBARIĆ Slobodan, Arhitektura mikroprocesora, Tehnička knjiga, Zagreb 1982
17. RISTOVIĆ Dejan, Računari u vašoj kući, Specialno izdanje časopisa Galaksija, Beograd 1984
18. SCHARF Julius, Basic für Aufsänger, R. Oldenbourg Verlag, Wien 1977
19. STOLKOVIĆ V., TOSIĆ D., Zbirka zadataka iz programiranja – programski jezik BASIC, PFV Beograd 1982
20. ŠPILER Jure, Basic, Samozaložba, Ljubljana 1984
21. ŠUHEL Peter, VIRANT Jernej, Mikroračunalnik, DDU Univerzum, Ljubljana 1978
22. VICKERS Steven, Basic programming ZX Spectrum, Sinclair Research Ltd., Cambridge 1983
23. WOLTERS F. Martin, Ključ za kompjuter, ZAK Beograd.

VSEBINA

UVOD 3

1. UVODNA RAZMIŠLJANJA IN OSNOVNI POJMI 5

2. RAZVOJ RAČUNALNIŠTVA 8

3. O INFORMACIJAH IN NJIH PREDSTAVITVI 16

4. ANATOMIJA IN DELOVANJE RAČUNALNIKA 18

4.1. Pomnilnik 22

4.2. Aritmetično–logična in krmilna enota 23

4.3. Vhodno–izhodne enote

5. MIKROPROCESORJI IN MIKRORAČUNALNIK 28

6. ANATOMIJA MIKRORAČUNALNIKA 30

6.1. Vhodno–izhodna enota 33

6.2. Zunanji pomnilnik hišnih računalnikov 35

6.3. Dodatna oprema

7. MIKRORAČUNALNIKI V SVETU IN PRI NAS 36

8. MIKRORAČUNALNIŠKI SISTEM PARTNER 38

8.1. Kratka predstavitev delovne organizacije ISKRA DELTA 38

8.3. Zgradba PARTNER-ja 40

8.4. Uporaba PARTNER-ja 43

8.5. Instalacija in vključitev 46

8.6. Vzdrževanje in šolanje 47

9. POGOVOR Z RAČUNALNIKOM 48

9.1. Programski jezik 48

9.2. Še nekaj o tehniki programiranja 51

9.2.1. Definiranje naloge 51

9.2.2. Analiza naloge 52

9.2.3. Diagram poteka in program 52

9.2.4. Vnos programa in kontrola 53

9.2.5. Testiranje programa 53

9.2.6. Preizkusna obdelava in shranjevanje programa 54

9.2.7. Diagram poteka – malo za šalo, malo zares

10. PRVI KORAKI V BASIC Z RAČUNALNIKOM ISKRA HR–84 57

10.1. Pa začnimo!

10.2. 65

10.3. Spremenljivke 72

10.4. Premikanje zapisov 75

10.5. Preskoki z računalnikom 82

10.6. Računalniške zanke, kaj je to? 86

10.7. Naredimo program za mlajšega bratca

VSEBINA

UVOD 3

1. UVODNA RAZMIŠLJANJA IN OSNOVNI POJMI 5

2. RAZVOJ RAČUNALNIŠTVA 8

3. INFORMACIJA IN NJIH PREDSTAVITVI 16

4. ANATOMIJA IN DELOVANJE RAČUNALNIKA 18

4.1. Pomnilnik 22

4.2. Aritmetično–logična in krmilna enota 23

4.3. Vhodno–izhodne enote

5. MIKROPROCESORJI IN MIKRORAČUNALNIK 28

6. ANATOMIJA MIKRORAČUNALNIKA 30

6.1. Vhodno–izhodna enota 32

6.2. Zunanji pomnilnik nižjih računalnikov 35

6.3. Dodatna oprema

7. MIKRORAČUNALNIKI V SVETU IN PRI NAS 38

8. MIKRORAČUNALNIŠKI SISTEM PARTNER 38

8.1. Kratka predstavitev delovne organizacije ISKRA DELTA 38

8.3. Zgradba PARTNER-ja 40

8.4. Uporaba PARTNER-ja 43

8.5. Instalacija in vključitev 46

8.6. Vzdrževanje in solanje 47

9. POGOVOR Z RAČUNALNIKOM 48

9.1. Programski jezik 48

9.2. Še nekaj o tehniki programiranja 51

9.2.1. Definiranje naloge 51

9.2.2. Analiza naloge 52

9.2.3. Diagram poteka in program 52

9.2.4. Vnos programa in kontrola 53

9.2.5. Testiranje programa 53

9.2.6. Preizkusna obdelava in shranjevanje programa 54

9.2.7. Diagram poteka – malo za šalo, malo zares 54

10. PRVI KORAKI V BASIC Z RAČUNALNIKOM ISKRA HR–84 57

10.1. Še začnimo!

10.2.

10.3. Spremenljivke 65

10.4. Premikanje zapisov 72

10.5. Preskoki z računalnikom 75

10.6. Računalniške zanke, kaj je to? 82

10.7. Nerednno program za malšega brata 86

IZDAJATELJ: Iskra–DELTA V SODELOVANJU Z ZOTKS

ZALOŽILA: SOZD ISKRA

ORGANIZACIJA TISKA IN TISK: PARALELE LJUBLJANA – DU DOMŽALE

LJUBLJANA, 1. oktober 1984

NAKLADA: 6000 izvodov, 1. natis