ÁSZ BASIC®

PROGRAMOZÁS KÁRTYÁVAL

Első lépés a programozáshoz.
Játék négyéves kortól mindazoknak,
akik szeretnek játszani és
érteni akarják egy számítógép nyelvét

NOVOTRADE
OFFSET és JÁTÉKKÁRTYA NYOMDA

AJÁNLÁS

A közelmúlt nagy sikere a személyi számítógép, mely Magyarországon is mind több embert hódít meg, családtag lett otthon és az iskolában is társa a diáknak, tanárnak. A nagy siker és az érdeklődés követelően írja elő, hogy minél többet tudhassunk meg erről az eszközről, megismerhessük tulajdonságait, lehetőségeit, megtanuljuk használatát. Ezt a célt szolgálják a szakkönyvek, szakfolyóiratok, az oktatási programcsomagok, melyek lehetőséget adnak arra, hogy a számítógépet, annak használatát, a számítógépek programozását megismerjük, elsajátítsuk.

A számítástechnikai kultúra elterjesztése azonban igényel más módszereket is és, ha azt szeretnénk, hogy ez a kultúra mindenki számára befogadható legyen, akkor örömmel kell venni az új módszereket, javaslatokat. Ilyen jó ötlet, kezdeményezés, a játékos módszer a komoly ismeretek elsajátítására. Ez a könyv egy ilyen lehetőséget ad. Felépítését és módszereit tekintve gyerekek és felnőttek egyaránt hasznosan forgathatják, tanulmányozhatják. Ha megismerkednek a könyvvel, figyelmesen nyomon követik a leírtakat, betartják a játékszabályokat, akkor nemcsak hasznos időtöltéshez jutnak, hanem megismerik a számítástechnika világát, a programozás alapjait. Szokszor elhangzott már, de szükséges újra elmondani, hogy a számítástechnika nem öncél, hanem hasznos eszköz, társ munkánkban, életünkben.

Remélem, hogy e játékos könyv segítségével sokan megismerkednek a számítástechnikával, megtanulják a programozást és az itt szerzett ismereteket életükben, munkájukban jól tudják hasznosítani.

Budapest, 1984. december 20.

PÁRIS GYÖRGY
Minisztériumi főtanácsos
A Tudományszervezés és
Informatikai Intézet igazgatója

KEDVES JÁTSZÓTÁRSAK!

A személyi számítógépek már megjelentek gyáraink-ban, áruházainkban, iskoláinkban. Nincs messze az az idő, amikor ilyen vagy olyan módon kapcsolatba kell kerülnötök VELE.

A személyi számítógép segítséget nyújthat nektek a gondolkodást igénylő feladatok megoldásához. Például: megoldhat matematikai feladatokat, társatok lehet a játékban, szöveget tárolhat, rajzolhat, segítségetekre lehet problémák elemzésében vagy akár a zeneszerzésben is. Azonban nem olyan okos, hogy megértse a gondolkodásotok logikáját és azt a nyelvet, amelyet beszéltek. Sajátos gépi logika szerint működik, és működését csak a programokon keresztül befolyásolhatjátok. Ezért, ha azt akarjátok, hogy a személyi számítógép nektek engedelmeskedjék, el kell sajátítanotok legalább egy programozási nyelvet.

Ilyen programozási nyelv a BASIC. Előnye a többi programozási nyelvvel szemben, hogy használatával minden személyi számítógépre írhattok programot, és megtanulása nem jelent számotokra nehézséget, különösen akkor nem, ha ezt a nyelvet az ÁSZ–BASIC játékkal sajátítjátok el.

ÁSZ BASIC

Az ÁSZ–BASIC játék lehetőséget nyújt a BASIC programozási nyelv – minden számítógépen közös elemeinek – elsajátítására, a programolvasás és a programozás tervezési alapjainak megismerésére, s mindezeket az ismereteket a játékosság, a vetélkedés, a szórakozás keretei között nyújtja.

program

A számítógép működését programokkal lehet megszervezni. *A program a számítógép működését előíró olyan utasítások sorozata, amely egy vagy több feladat teljes körű végrehajtását eredményezi.*

A számítógép sajátos belső gépi nyelven írt program hatására működik. Ezen a nyelven elsősorban a programozó szakemberek tudnak programot írni. A fejlesztők, hogy közelebb hozzák a programkészítést az emberi nyelvhez, úgynevezett magas szintű programozási nyelveket dolgoztak ki.

Az egyik gyorsan elsajátítható magas szintű programozási nyelv a BASIC. A programozás tanítására készült, de napjainkra az egyik legelterjedtebb programozási nyelv. Talán széles körű elterjedtségének következménye, hogy nem alakult ki egységes leírási szabálya. A számítógépeket készítő gyárak több-kevesebb egyedi sajátossággal állítják össze számítógépeik BASIC nyelvét. A BASIC programozási nyelvet ezért célszerű két ütemben megtanulnotok.

Első ütemben sajátítsátok el azokat a közös részeket, melyek minden személyi számítógépben megtalálhatók, s ezzel lehetőségetek nyílik bármelyik gép működését megszervezni.

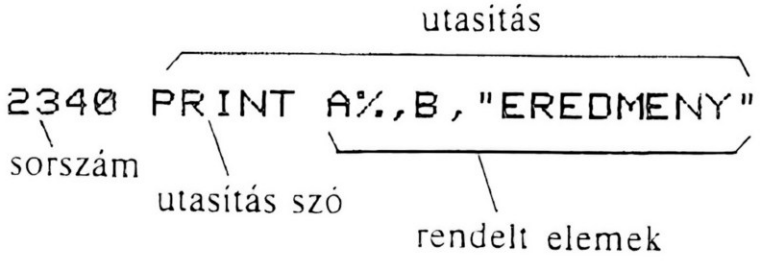
A második ütemben, a meglévő programozási tudásotokat kell az adott típusú gépre jellemző részekkel fejleszteni.

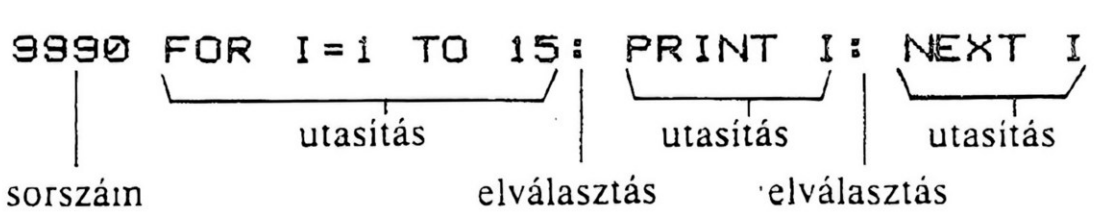
Az ÁSZ–BASIC játék a programozás tanulásának első ütemében segít, a kidolgozott feladatok bármelyik személyi számítógépen végrehajthatók, a programok alapvető átalakítása nélkül. (A játékok programjai Commodore-64 számítógépre készültek.)

programsor

A BASIC program programsorokból épül fel. A programsor lehet egyszerű és összetett. *Az egyszerű programsor sorszámból, utasításszóból és rendelt elemekből áll. Az utóbbi kettőt együtt utasításnak nevezzük. Az összetett programsor egy sorszámból és egymástól kettősponttal elválasztott utasításokból áll.*

egyszerű programsor





összetett programsor

Kedves játszótársak, kezdjetek el játszani!

Az első játékkal játszhattok Fekete Pétert.
(Három-négyéves kortól).

A második játék segítségével megtanulhatjátok a BASIC nyelv utasításszavait és megismerhetitek jelentéstartalmukat.
(Hét-tizenkét évesek részére)

A harmadik játékkal megtanulhattok programot olvasni.
(Tizenkét éves kortól)

Végezetül, ha magatok is készítetek a játékhoz programokat, tanultok meg programot írni.

FEKETE PÉTER JÁTÉK ÁSZ–BASIC KÁRTYÁKKAL

A játékot ketten, hárman vagy négyen játszhatjátok. A játékhoz 18x2 ÁSZ–BASIC és egy Fekete Péter kártya szükséges. (A kártyacsomagokból szedjétek ki a képeket tartalmazó lapokat.)
A játék célja, hogy minél több azonos jelű kártyapárt gyűjtsetek össze.
1. Az osztó keverje meg a kártyákat, és ossza ki a teljes csomagot, a kezdő játékosnak eggyel többet!
2. A kezetekben lévő kártyapárokat tegyétek magatok elé, majd a kezdő játékos a pár nélküli lapjait a következő játékostársának húzásra mutassa fel. A következő játékos húzzon egy lapot, és ha a lap párja a kezében van, a párt maga elé tegye le. Ezután a pár nélküli lapjait a következő játékosnak húzásra mutassa fel.

3. A játékot mindaddig folytassátok, amíg minden páros lapot le nem raktatok. Akinek a kezében a Fekete Péter kártya maradt, az a vesztes.

LERAKÓ JÁTÉK ÁSZ–BASIC KÁRTYÁKKAL

A játékban kettő, három vagy négy játékos vehet részt.
1. Közös megegyezéssel válasszatok az A–K betűkkel jelölt JÁTÉKLAPOK közül, majd a JÁTÉKTÁBLÁT helyezzétek az asztal közepére.
2. Az osztó keverje meg a kártyacsomagot, és osszon a JÁTÉKLAPON feltüntetett számú kártyát. A többi kártyát helyezze húzásra a játéktábla mellé.
3. Ha a kezdő játékosnak kezében van a programnyitó kártya, akkor tegye a játéktáblára a 10. programsorba, az utasítás szó helyére (ha nincs a kezében a nyitókártya, „továbbot" mond), majd dobjon el egy lapot. Ha a kezdőjátékosnak nincs a megfelelő kártya a kezében, akkor a következő játékos húz egy lapot, és átveszi a kezdőjátékos szerepét. A kezdőjátékos szerepének átvállalása mindaddig folytatódik, amíg a kezdőlap lerakásra nem kerül.
4. A következő játékos húzzon egy lapot, és ha kezében van a következő programsor utasításszavát tartalmazó kártya, akkor azt tegye a játéktáblára (ha nincs, mondjon „továbbot"), majd dobjon egy lapot.
5. A játékosok ismételjék meg a 4. lépést az utolsó programsorig. Ha a játékos kezéből elfogytak a lapok és rákerül a sor, húzzon új lapot, a játékból nem szállhat ki.
6. Az a játékos nyert, aki a befejező programsor utasításszavát tartalmazó kártyát letette.
7. Ha játék közben az egyik kártyacsomag elfogy, úgy az osztó keverje meg a másik kártyacsomagot, és húzásra helyezze az asztalra. Ha a második kártyacsomag is elfogy, akkor az osztó keverje meg a dobott lapokat, és azokat tegye húzásra az asztal közepére.

Az ÁSZ kártya használata

Az ÁSZ kártya minden ÁSZ–BASIC kártyát helyettesíthet, természetesen a programsorban annak az utasításszónak megfelelően kell értelmezni, amelyik kártyát helyettesíti.

A lerakott ÁSZ kártyát a következő játékos felveheti, ha a programsornak megfelelő utasításszavú kártyával ki tudja cserélni.

A játék értékelése

1. A nyerő játékos –10 pontot ír. (A pontjaiból levonunk 10 pontot.)
2. A kézben maradt lapok értékeit írjuk össze:
   2 pontot kap az a kártya, amely nem szerepelt a feladatlapon;
   5 pontot kap az a kártya, amely szerepelt a feladatlapon;
   10 pontot az ÁSZ kártya.
   A nyerő játékos kézben maradt lapjainak értékét az eredménybe nem kell beszámolni.

A játékot egyenként vagy öt ismételt játék után lehet értékelni, és a pontok összege alapján a győztest meghatározni. Az a játékos győzött, akinek a legkevesebb pontja van.

PROGRAMOLVASÓ JÁTÉK ÁSZ–BASIC KÁRTYÁKKAL

Ezt a játékot akkor kezdhetitek játszani, ha előzetesen megismerkedtetek a BASIC programozási nyelv alapjaival. A BASIC nyelv sajátossága, hogy két formában is írható vele program: közvetlen végrehajtású és tárolt program.

A közvetlen végrehajtású program esetében minden utasítás a számítógépre történt beírása után azonnal végrehajtásra kerül. Ennek a formának az előnye az eredmény közvetlen értékelésének lehetősége.

A tárolt program alkalmazásakor a számítógép által végrehajtandó feladatra programot készítünk. Ezt a programot a számítógépben elhelyezzük, és csak akkor hajtatjuk végre, amikor a gépi feldolgozás eredményére szükségünk van. Aki elsajátítja a tárolt program készítését, az automatikusan közvetlen végrehajtású programot is tud írni.

BASIC programozási nyelv

A BASIC programozási nyelv sorszámokból, elemekből, utasításokból és parancsokból épül fel. A sorszámok a programok megkülönböztetésére szolgálnak. Az elemek rendeltetése az adatok hordozása, az utasítások írják elő az elemeken végzendő műveleteket, a parancsok pedig a számítógéppel hajtatnak végre a programokkal kapcsolatos feladatokat.

sorszámok

A BASIC programozási nyelvben a tárolt program minden programsorának sorszáma van. A sorszámok 1-től 99999-ig terjedhetnek. A programozás gyakorlatában meghonosodott, hogy a programozók a programsorok sorszámát ötösével vagy tízesével növekvő rendben választják meg. Ezt a gyakorlatot azért célszerű elfogadnotok, mert így lehetőségetek van a programok utólagos bővítésére, javítására, a programsorok átsorszámozása nélkül. *A közvetlen végrehajtású programsoroknak és a parancsoknak nincs sorszáma.*

A BASIC programozási nyelv az ember és a számítógép közötti kapcsolatokat jelekkel hozza létre. A jelek az ember–ember kapcsolat írásformáinak felelnek meg. Ha gondolatotokat írásban akarjátok közölni társatokkal, betűket, számokat, írásjeleket és különböző más jeleket használtok. Így történik ez a számítógéppel való kapcsolattartásnál is. Gondolataitokat, tudásotokat jelekkel közlitek a számítógéppel. A betáplált jeleket parancsaitoknak megfelelően a számítógép a programjaival átalakítja, és jelekkel új ismereteket közöl veletek. Az ember–ember kapcsolatban igen sok jelet különböztethetünk meg. A BASIC programozási nyelv ezeknek a jeleknek csak egy részét alkalmazza. *Az alkalma-*

zott jeleket nevezzük a nyelv elemeinek. Az elemek lehetnek *alapelemek* és *összetett elemek.*

elemek

alapelemek

Betűk:	az ABC kis- és nagybetűi A-tól Z-ig, kivéve az ékezetes és a kettősbetűket
Számjelek:	0 1 2 3 4 5 6 7 8 9 (a számítógépek a nulla számjegyet áthúzva írják ki)
Előjelek:	+ –
Elválasztó jelek:	: utasítások elválasztására, , ; számok, szövegek elválasztására
Írásjelek:	? ! " " .
Aritmetikai műveleti jelek:	+ összeadás – kivonás * szorzás / osztás ↑ hatványozás
Szövegműveleti jel:	+ szövegrészek összefűzése

Logikai műveleti jelek:	AND	és
	OR	vagy
	NOT	nem

Relációs jelek:	<	kisebb
	>	nagyobb
	=	egyenlő
	< =	kisebb vagy egyenlő
	> =	nagyobb vagy egyenlő
	< >	nem egyenlő

Egyéb jelek:	$	dollár
	%	százalék

és még néhány más jel

összetett elemek

Az összetett elemek a BASIC nyelv szabályai alapján az *alapelemekből képződnek*, és formájukat tekintve lehetnek: *állandók, változók, kifejezések és függvények.*

állandók (konstansok)

A BASIC programozási nyelven az *adatok megjelenítésére szám- vagy szövegállandókat alkalmaznak.* (Adat: az emberi ismeretek megjelenési formája.)

Számállandók lehetnek egész és valós típusúak.

Egész típusú számállandó áll: előjelből és számjegyekből (pl.: 16 vagy –32767, a + előjelet nem kell feltüntetni). Az egész típusú számállandó alkalmazását az adatok tárolási helyszükséglete és a program futási ideje indokolja. Viszont a legtöbb számítógépen csak
– 32767<=X<= 32767 értéket vehet fel.

A valós típusú számállandónak kétféle megjelenési formája van. Az első előjelből, egész részből, tizedespontból, törtrészből áll (pl.: 16.5 vagy 9.999999 vagy .33, ha az egész szám nulla, nem kötelező kiírni).

A valós számállandónak ezt a formáját maximálisan kilenc számjegyből álló szám leírására használjátok. Ha az egész és törtrész számjegyeinek száma több mint a kilenc, a számítógép automatikusan a másik, az exponenciális formában fogja ábrázolni a számot. Az exponenciális forma: áll előjelből, egy számjegyből álló egész részből, tizedespontból, törtrészből, E jelből, előjelből és kitevőből (pl.: $1{,}63256 \cdot 10^{-38}$ számot a számítógép 1.63256E–38 formával ábrázolja).

Szövegállandók (angolul stringek) betűkből, számjegyekből és bármilyen jelekből, vagyis az alapelemek összességéből épülhetnek fel. A szövegállandókat idézőjelbe kell tenni (pl.: „ÍRJ BE EGY SZÁMOT")

változók

A változók fogalma a BASIC programozási nyelvben megfelel a matematikából ismert változók fogalmának, olyan nevek, amelyek értékeket vehetnek fel.

A nevek általában több jel hosszúságúak lehetnek, de a gép csak az első két jelet veszi figyelembe. Az első jel mindig betű, a második pedig betű vagy számjegy lehet.

A változók típusaik szerint az állandók típusainak felel-

nek meg. Így *lehetnek szám-, vagy aritmetikai, ill. szöveg-(string) változók.*
A számváltozók valós típusa esetén a név az előzőekben megjelölt formát veszi fel, ha a számváltozó egész típusú, úgy a név után % jelet kell alkalmaznunk. A *szövegváltozók* esetén a név után $-jelet kell kitenni.
(Pl.: A1, A1%, A1$)

kifejezések

A számítógép az ember által közölt jelekkel műveleteket végez. A műveletek leírására szolgálnak a kifejezések. A kifejezések típusaik szerint *lehetnek aritmetikai, relációs, logikai és szövegkifejezések.*
Aritmetikai kifejezések számállandókból, változókból és aritmetikai műveleti jelekből állnak.
Pl.: `(A+B)/2*(C-3)↑2`
Amint a példa is mutatja, az aritmetikai kifejezések leírásának szabályai a BASIC programozási nyelvben bizonyos mértékben eltérnek a metematikában alkalmazott szabályoktól.

Matematikai	BASIC
$a + b$	`A+B`
$ab + 3b$	`A*B+3*B`
ax^2	`A*X↑2`
$\frac{a}{a+b}$	`A/(A+B)`
$x^2 - (3x+2)$	`X↑2-(3*X+2)`
$a - (-b)$	`A-(-B)`

Az aritmetikai kifejezések *egész* és *valós típusúak* lehetnek. Az egész típusú aritmetikai kifejezések minden elemének egész típusúnak kell lennie. Valós típusú aritmetikai kifejezésnek legalább egy eleme valós típusú kell hogy legyen.
A relációs kifejezések a relációjelekkel összekapcsolt állandók, változók, valamint aritmetikai kifejezések.
Pl.: `A>B, A*B=6, A>=B↑2, A$<B$`

A logikai kifejezések a logikai jelekkel összekapcsolt relációs kifejezések.

Pl.: `A=3 AND B<A↑2, B>A OR B=10`

A szöveg- (string) kifejezések a szövegek, ill. szövegváltozók összekapcsolására szolgálnak. Az összekapcsolás + műveleti jellel történik.

Pl.: `"ALMA"+A$+B$`

függvények

A különböző típusú függvények közül a leggyakrabban alkalmazottak a BASIC programozási nyelvben elemként szerepelnek. *A függvények megnevezésből és zárójelbe foglalt független változókból állnak.* A BASIC programozási nyelvben a függvények rendeltetésük szerint három típusba sorolhatók: matematikai, szöveg- és egyéb függvények.

Pl.: `SIN(X), STR$(X), TAB(X)`

A függvények felsorolásától itt eltekinthetünk. Megtaláljátok a BASIC nyelv összefoglalóban, a játékfüzetben.

utasítások

A BASIC programozási nyelv utasításszavai a programban végrehajtandó feladatokat angol nyelvű szavakkal fejezik ki. A szavakhoz meghatározott szabályok betartásával a programnyelv elemei járulnak. A nyelv utasításkészlete és alkalmazásának szabályai igen széles lehetőségeket nyújtanak a programozónak. Ez az ismertetés nem vállalkozhat az utasítások alkalmazásával kapcsolatos minden változat tárgyalására, csak az utasítások alapértelmezésére szorítkozik. Részletesebb ismertetések a szakkönyvekben találhatók.

KIÍRÓ utasítás

Formája: PRINT rendelt elemek

Feladata: kiírás képernyőre.

A PRINT szó után rendelt elemek lehetnek: állandók, változók, kifejezések és függvények.

kiíró PRINT

A PRINT utasítással végrehajtható alapműveletek a következők lehetnek:

Az utasítással végrehajtható művelet	*A programsorok formája*
Számváltozó értékének kiírása	`80 PRINT A1`
Szövegváltozó tartalmának kiírása	`90 PRINT B$`
Szöveg kiírása	`100 PRINT "EREDMENY"`
Kifejezés kiszámított értékének kiírása	`110 PRINT X↑2+3`
Függvény kiszámított értékének kiírása	`120 PRINT SIN(X)`

A számítógép a kiíró utasításra ismereteket szolgáltat az embernek. Az előző példákban megjelenő ismereteket egy-egy adatnak értelmezhetjük. A számítógép képernyőjére a programok rendszerint több adatot is kiírnak. Ilyen esetekben gondoskodni kell arról, hogy a kiírt adatok rendezettsége a képernyőn elősegítse a közölt ismeretek megértését.

A személyi számítógéphez (több típusnál) televíziót lehet csatlakoztatni, így az adatok kiírása a képernyőre történik. Általános, hogy a televízió képernyőjének méretétől függetlenül: 25 sorban és soronként 40 helyre írhatók ki alapelemek, pontosabban jelek.

Egy sorban 40 jel fér el, ez azt jelenti, hogy a kiírt elemek és szóközök száma összesen 40 lehet. Egy sor tehát lehet 40 alapelem (például csillag) és lehet 40 üres hely, vagyis szóköz, vagy az alapelemek és a szóközök bármilyen összetételű változata.

A példáinkban a kiírási területen mindig a 25 sort, soronként a 40 helyet vesszük számításba. Vannak olyan számítógéptípusok, ahol ettől eltérő, általában nagyobb kiírási területtel találkozhattok. Ez a játékok programjainak futtatásában nem jelent számotokra nehézséget.

Az adatok elrendezését szolgálja a vessző és a pontosvessző, mint elválasztó jel.

pontosvessző használata

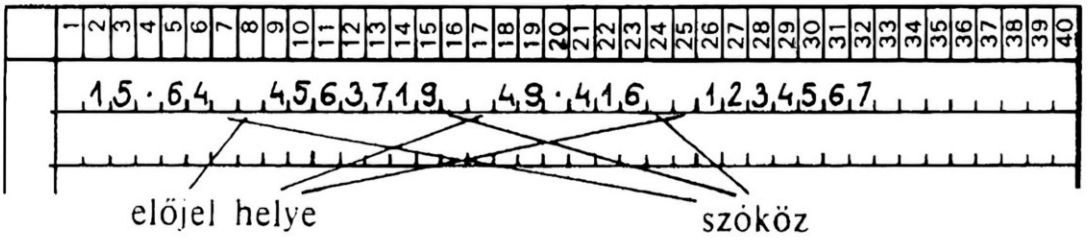
Például ha:

A=15.64, B=4563719, C=49.416, D=1234567

számállandókat egymás után akarjuk kiírni, akkor az elválasztáshoz a pontosvesszőt alkalmazzuk.

A programsor felépítése és a számok kiírása a következő:

```
100 PRINT A;B;C;D
```



szövegrészek összefűzése

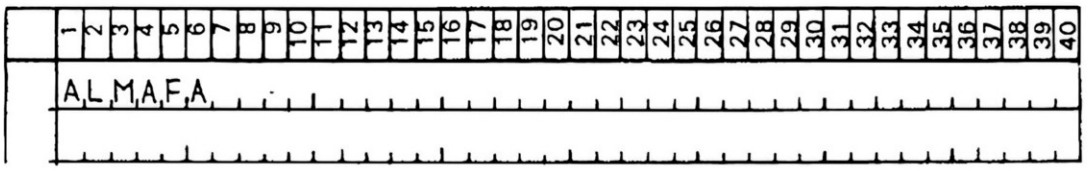
Ugyanezen a módon írhatunk ki szöveget is.

Például ha: A$=„ALMA” és B$=„FA”

kiírását akarjuk elvégeztetni, a programsort az alábbiak szerint fogalmazzuk:

```
80 PRINT A$;B$
```

A szöveg megjelenése a képernyőn pedig a következő:



Nincs szóköz!

vessző használata

Kiírási szabály, hogy a pontosvessző után:

ha számállandó, számváltozó vagy kifejezés áll, az adat kiírása egy szóközzel kezdődik;

ha szövegállandó vagy szövegváltozó áll, az adat kiírásakor szóköz nem jelenik meg.

Az adatok táblázatos formába történő kiírását segíti elő a vesszőnek, mint elválasztó jelnek a használata.

A képernyő kijelzési zónákra osztható fel. A kijelzési zónák a számítógép típusától függően különböző helyszélességgel rendelkeznek. Esetünkben egy zóna szélessége 10-10 hely szélességű, tehát a zónák az 1, 11, 21, 31 pozíciókban kezdődnek.

1	11	21	31
Első zóna	Második zóna	Harmadik zóna	Negyedik zóna

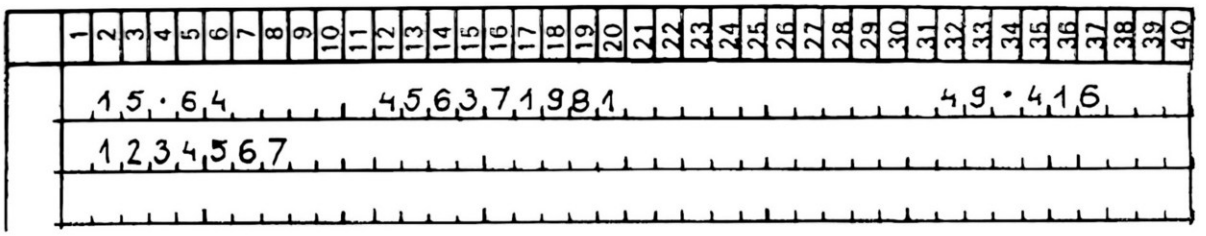
kijelzési zóna

A zónákba történő kiíráskor az előző példát figyelembe véve a programsor felépítése és a számok kiírása a következő szerint módosul:

```
100 PRINT A,B,C,D
```

Növeljük meg a B változó tartalmát kilencjegyűre, legyen B = 456371981, a képernyőn a számok az alábbiak szerint íródnak ki:

szám a kijelzési zónába nem fér el



Tehát olyan esetekben, ha a szám nem fér el a kijelzési zónában (eléri a kijelzési zóna határát) a számítógép a kiírást a következő zóna igénybevételével folytatja, és a következő szám kiírását egy zónával eltolja. Ha nincs több zóna az adott sorban, úgy a következő sorban folytatja a kiírást. (Egy PRINT utasításra maximálisan két sorban írhat ki adatokat a számítógép.)
A vessző alkalmazásával szövegeket is lehet kijelzési

zónákba elhelyezni. Egy szöveg hossza esetünkben nem foglalhat el több helyet kilencnél. Ennél hosszabb szövegek esetében a számítógép a kiíráshoz a következő zónát (zónákat) veszi igénybe.
Legyenek a kiírandó szövegek: „ALMÁS", „KÖRTÉS", „CSERESZNYÉS", „BARACKOS" és változóik: A$, B$, C$, D$.
A programsor felépítése, ha folyamatosan össze akarjuk írni a szövegeket:

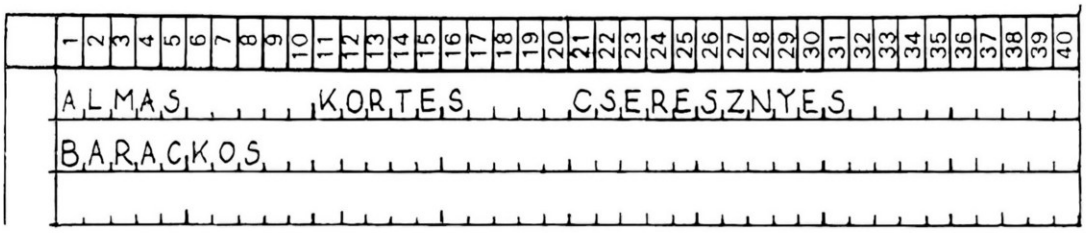
```
50 PRINT A$;B$;C$;D$
```

Ha azt akarjuk, hogy a kijelzési zónákba kerüljenek a szövegek, akkor a programsor felépítése a következő:

```
50 PRINT A$,B$,C$,D$
```

szöveg a kijelzési zónába nem fér el

Számoljátok meg a „CSERESZNYÉS" szöveget. Betűinek száma tizenegy. Tehát a kiírása két zónát vesz igénybe. A képernyőn a következőképpen jelenik meg:



Olyan esetekben, amikor a fenti négy szöveget egy sorban kell elhelyeznünk, mert például valamilyen táblázat fejlécéhez tartoznak, akkor a „CSERESZNYÉS" szöveget rövidítenünk kell.
A szöveget a PRINT utasítással változók nélkül is kiírhatjuk:

```
110 PRINT "EREDMENY"
```

Az idézőjel és a szöveg első eleme közötti szóközökkel állítható be, hogy az adott sorban hol kezdődjék el az „EREDMÉNY" szó kiírása.
Ugyanezt érhetjük el a TAB (X) függvény alkalmazásával is:

TAB (X)

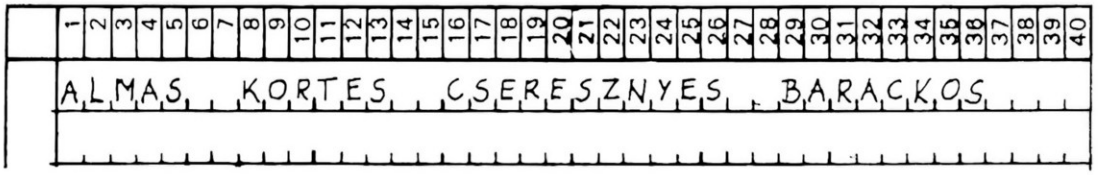
```
110 PRINT TAB(6);"EREDMENY"
```

A TAB(6) hatására az „EREDMÉNY" szöveg a 7 pozícióval kezdődően íródik ki.

A TAB(X) függvény alkalmazásával helyezzük el az előző négy szöveget.
A programsor:

```
50 PRINT A$;TAB(7);B$;TAB(15);C$;
TAB(28);D$
```

Az utasítás hatására kiíródó szövegek elhelyezkedése a képernyőn:



A PRINT utasítást alkalmazzuk a sormelésre. **soremelés**
Például:

```
70 PRINT:PRINT
```

programsorral kettős soremelés műveletét írhatjuk elő. A számítógépek egy részénél ha a kiíró utasításhoz idézőjelben a CLR billentyűt is lenyomjuk, lehetőségünk van a képernyő törlésének programozására. **képernyő törlés**

```
50 PRINT "■"
```

Ezt az eljárást azonban nem minden személyi számítógépnél alkalmazhatjátok. Kiíró utasítással képernyő törlést úgy is végezhettek, ha CHR$(X) függvénnyel először a kurzor jelet (a kiírás helyét jelölő jelzés) a képernyő bal felső sarkába állítjátok, majd a törlés vezérlését rendelitek el. Az X értéke egy-egy vezérlő ASCII kódszám. Sajnos ez is számítógéptípusonként változik. A kódszámok a gépkönyvben megtalálhatók.

BEVITELI utasítás

Formája: INPUT változó, változó, ... vagy „szövegállandó"; változó, változó,...
Feladata: billentyűzetről adatok bevitele a számítógépbe.
Az INPUT szó után a rendelt elemek szám- vagy szövegváltozók lehetnek: **beviteli INPUT**

```
90 INPUT A,B
```

programsorral kettő valós számot;

```
120 INPUT B%,KK%,B3%
```

programsorral három egész számot:

```
150 INPUT A$,S$
```

programsorral kettő szöveget vihetünk be a számítógépbe.

Az INPUT utasítás után magyarázó céllal szöveget helyezhetünk el, ha szükségesnek tartjuk, hogy a számítógép jelezze, hogy milyen adatokat kell beírni a gépbe.

```
200 INPUT" IRJ BE KET SZAMOT";A1,B1
```

programsorban a magyarázó szöveget idézőjel közé kell tenni, és a változóktól pontosvesszővel el kell választani. Olyan esetekben, ha szöveget és számot (Pl.: A TANULÓ NEVE ÉS SZÜLETÉSI ÉVE) kell egymás után bevinni a gépbe, akkor a következőképpen írjuk le a programsort:

```
150 INPUT"A TANULO NEVE ES SZULETESI
EVE";N$,EV%
```

Az INPUT utasításra a számítógép kérdőjellel jelzi a bevitelre való készenlétet. NS változó értékének megfelelő adatok beírása után a gép ismét kérdőjellel jelzi a bevitelre való készenlétet.

Az adatok beírását elvégezhetjük az első kérdőjel után is, de ekkor vesszővel el kell választani az adatokat egymástól.

értékadó LET

ÉRTÉKADÓ utasítás

Formája: LET változó = kifejezés

Feladata: változóhoz értéket rendelni, illetve elvégeztetni a kifejezésben megadott műveleteket.

A LET szó után szám- vagy szövegváltozó, a hozzárendelő jel és szám- vagy szövegállandók, illetve kifejezések állhatnak, szigorúan betartva, hogy a számváltozóhoz szám-, és szövegváltozóhoz szövegállandó, illetve kifejezés rendelhető.

Amennyiben számváltozóhoz szövegállandót, illetve szövegváltozóhoz számállandót akarunk rendelni, úgy VAL és a STRS függvényt kell alkalmaznunk.

A LET szó kiírása nem minden számítógéptípus esetében szükséges. A játékunkban azonban nem nélkülözhetjük a kiírást. Ezért minden programban kiírtuk.
Értékadás egész típusú számváltozónak

```
100 LET A%=12
110 LET B%=A+6
```

Értékadás valós típusú számváltozónak

```
120 LET C=A/B
130 LET D=1.5*C↑2
```

Értékadás szövegváltozónak

```
140 LET A$="ALMA"
150 LET B$="FA"
160 LET C$=A$+B$
```

Értékadás függvénnyel

```
170 LET E=INT(D)
```

A PRINT, az INPUT és a LET utasítások alkalmazásával már írható program. Kíséreljük meg értelmezni.
Program:

```
10 INPUT X
20 LET Y=X↑2+3*X+2
30 PRINT Y
```

Tartalom:
1. X értékének bevitele;
2. x^2+3x+2 értékének kiszámítása és hozzárendelése Y változóhoz;
3. Az eredmény kiírása.

Értelmezzük a rend és a szer szavakból a rendszer szót összeállító programot.

Program:

```
10 INPUT"IRD BE A REND ES A SZER SZAVA
KAT";A1$,A2$
20 LET A$=A1$+A2$
30 PRINT A$
```

Tartalom:
1. A rend és a szer szavak bevitele;

2. A szavak összevonása és hozzárendelése
   A$ változóhoz;
3. A rendszer szó kiírása.

Mit gondoltok, leírható az utóbbi programban megfogalmazott feladat kettő programsorral is?
Igen, hiszen a PRINT utasítással is összevonhatunk szövegeket.

```
40 INPUT"IRD BE A REND ES SZER SZAVAKA
T";A1$,A2$
50 PRINT A1$+A2$
```

magyarázat
REM

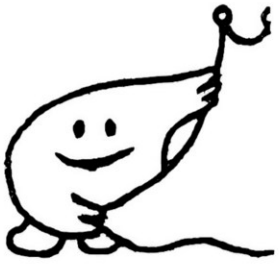
MAGYARÁZAT utasítás
Formája: REM szöveg
Feladata: magyarázó szöveg rögzítése a programban, illetve a programok részekre tagolása.

A REM szó után a magyarázó szöveg írható, de szöveg nélkül is alkalmazható. A hosszú, több száz vagy ezer programsorból álló programban eligazodni elég nehéz, ezért a programsorokat célszerű csoportosítani. Elsősorban ezeknek a csoportosításoknak a megkülönböztetésére alkalmazzuk a REM utasítást, de felhasználható minden olyan magyarázat leírására is, amit a program kidolgozója a felhasználókkal közölni akar. A REM utasításra a programban a gép továbblép, anélkül hogy műveletet végezne el.

VÉGE utasítás
Formája: END
Feladata: a program futásának befejezése.

vége END

Az END utasítás hatására a programfutás befejeződik, és minden változó aktuális értéke kitörlődik a számítógép memóriájából.



Ha kedvetek van hozzá, játsszatok! Az 1–4. JÁTÉK programjait már tudjátok értelmezni. Előzetesen ismerkedjetek meg a harmadik JÁTÉK szabályaival!

A PROGRAMOLVASÓ JÁTÉK SZABÁLYAI

A játékot 2–4 személy játszhatja.

1. Közös megegyezéssel válasszatok JÁTÉKLAPOT. (Ha először játszotok együtt célszerű, ha sorbaveszitek a JÁTÉKLAPOKAT.) Az osztó ossza ki a JÁTÉKLAPOKAT, és a JÁTÉKTÁBLÁT helyezze az asztal közepére.

2. Az osztó keverje meg a kártyákat, és a JÁTÉKLAPON feltüntetett számú kártyát osszon, a többit helyezze húzásra az asztal közepére, a JÁTÉKTÁBLA mellé.

3. Tanulmányozzátok a programot, és ha szükséges, puha ceruzával írjátok az utasításszavakat a programsorok üresen hagyott részeire, majd gondoljátok végig, hogy a kérdésekre mit feleltek.

4. A kezdő játékos, ha kezében van a programnyitó kártya, akkor tegye a JÁTÉKTÁBLÁRA, és hangos szóval értelmezze a programsor tartalmát, amennyiben az értelmezés nem helyes, vegye vissza a kártyát. (Az értelmezés helyességét vagy az osztó dönti el, vagy közösen döntsetek, vitás esetekben nézzétek meg a játék leírását.) Ha a kezdő játékosnak nincs a kezében a nyitó kártya, „továbbot" mond. Végezetül eldob egy lapot.

Olyan esetekben, ha a kezdő játékosnak nincs a kezében a nyitó kártya, a következő játékos húz egy lapot, és átveszi a kezdő játékos szerepét. Mindez addig folytatódik, amíg a kezdő lap lerakásra nem kerül.

5. A következő játékos húzzon egy lapot, és ha kezében van a következő programsor utasításszavát tartalmazó kártya, a programsor hangos értelmezése után tegye a JÁTÉKTÁBLÁRA (ha nincs, mondjon „továbbot"), és dobjon el egy lapot.

6. Ismételjétek az 5. lépést az utolsó programsorig. Amennyiben egyikőtöknek elfogyott a kezéből a lap, ha rákerül a sor, húzzon lapot, a játékból nem szállhat ki.

7. Az a játékos nyer, aki az utolsó programsor utasításszavát tartalmazó kártyát letette a JÁTÉKTÁBLÁRA,

és megfelelt az osztó által feltett kérdésekre. (A kérdésekre adott válaszok helyességét közösen döntsétek el, vitás esetben nézzétek meg a játék leírását.)
8. Ha játék közben az egyik kártyacsomag elfogyott, úgy az osztó keverje meg a másik csomagot, és húzásra tegye az asztal közepére. Ha ez a csomag is elfogyott, akkor az osztó a dobott lapokat keverje össze, és húzásra tegye az asztalra.
9. A játékban az ÁSZ kártya szerepe és a játék értékelése megegyezik a második játéknál leírtakkal.

1. JÁTÉK

Az $y = x^2 + 3x + 2$ függvény értékét számító program:

```
10 REM"FUGGVENY AKTUALIS ERTEKENEK SZAMITASA"
20 PRINT"□"
30 PRINT:PRINT
40 PRINT" ** Y=X↑2+3*X+2 **"
50 PRINT
60 INPUT" IRD BE AZ X AKTUALIS ERTEKET";X
70 LET Y=X↑2+3*X+2
80 PRINT
90 PRINT" A FUGGVENY ERTEKE Y=";Y;"HA X=";X
100 END
```

Tartalom:

1. A programmal végrehajtott feladat;
2. A képernyőtörlés (a törlést az "□" jel szimbolizálja) egy soremelés a PRINT hatására. A képernyő törléshez a PRINT beírás után lenyomni – idézőjelek között – a CLR billentyűt;
3. Kettő soremelés;
4. A program nevének kiírása;
5. Soremelés;
6. Az X értékének bevitele;
7. A függvény aktuális értékének kiszámítása és hozzárendelés Y változóhoz;
8. Soremelés;
9. Eredménykiírás;
10. Vége.

1. Az összetett programsorban az utasításokat kettősponttal választjuk el;
2. A negyedik sor második pozicióján;
3. Azzal, hogy olvasni tudjátok a képernyő tartalmát, gyakorlatilag a programolvasási készségeteket bizonyítjátok. Ezért célszerű, ha a játékban megfelelő figyelmet fordítotok a képernyőn megjelenő írásképnek és eredményadatoknak a programból való kiolvasására. A kiolvasásnál célszerű, ha következetesen dolgoztok.

Először, azt állapítsátok meg, hogy a képernyő tartalma a program futása közben hányszor törlődik. Ezt azért kell megállapítani, mert a törlési utasítások a képernyőre való kiírás szempontjából szakaszokra bontják a programot. A képernyő tartalomvizsgálatát az egyes szakaszokra külön-külön kell elvégezni.

Másodszor, a képernyőtörlést előíró utasítás „PRINT"␣"" programsorától kezdve programsoronként a következő törlési utasításig vagy a program végéig minden sort meg kell vizsgálni. A vizsgálat során minden egyes programsorra vonatkozóan meg kell állapítani, hogy miként járul hozzá a képernyőre történő kiíráshoz.

Harmadszor, a képernyőre kiírást végző utasításoknál meg kell állapítani, hogy a tartalom hányadik sorban és az adott sorban hányadik helyre íródik ki.

Negyedszer, az adatfeldolgozást végző utasítások milyen tartalmat állítanak elő, vagyis mi íródik ki eredményként.

Nézzük végig közösen az 1. játék programsorát.

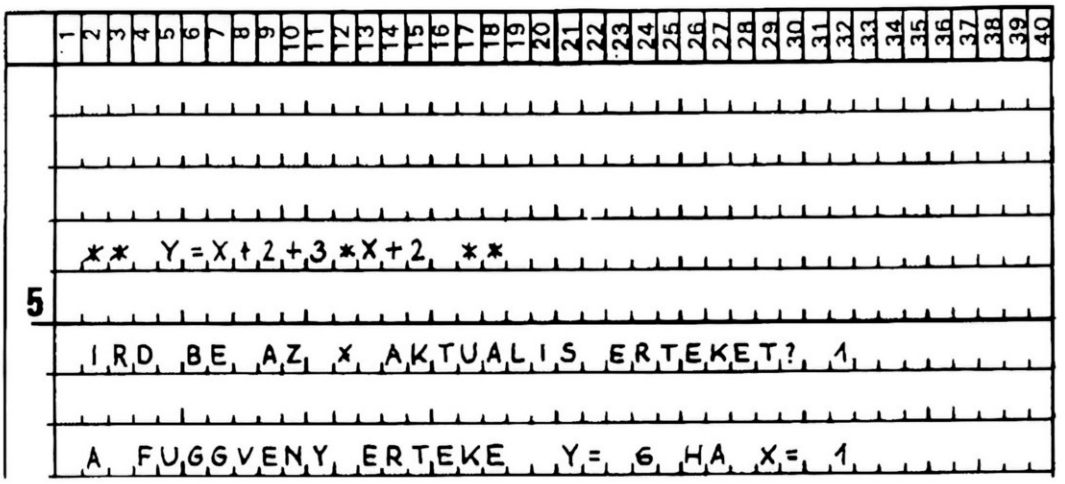
A képernyőtörlési utasítás a 20. programsorban van, és a program végéig másik törlő utasítás nem jelenik meg. A képernyőre történő kiírás szempontjából a program egy szakaszból áll.

A programsorok tartalmát már végignéztük. Kiírást végeznek: a 20., 30., 40., 50., 60., 80., és 90. programsorok. A 20. és 30. programsor a képernyő felső szélétől számítva három sort emel, így a 40. programsor szövege a negyedik sorban íródik ki, a kiírás pedig a második pozíción kezdődik. Az 50. programsor egy sort emel, ezért a

60. programsor szövege a hatodik sorba íródik, és a kiírás a második pozícióból kezdődően történik. A 80. programsor egy sort emel, 90. programsor szövege a nyolcadik sorban a második poziciótól kezdődően jelenik meg. A kiírandó eredményt pedig úgy kaphatjuk meg, ha elvégezzük a számítást:

Ha X = 1, akkor Y = 1^2+3x1+2 = 6

A további játékoknál már ilyen részletes leírást nem fogunk közölni a képernyőtartalom megállapításához. Ellenben bemutatjuk a képernyő írásképét.



2. JÁTÉK

A százalékértéket számító program:

```
10 REM"SZAZALEKERTEK SZAMITAS"
20 PRINT"■":PRINT:PRINT
30 PRINT"    ** SZAZALEKERTEK **"
40 PRINT
50 INPUT"          ALAP";A
60 PRINT
70 INPUT"    SZAZALEKLAB";P
80 LET E=A*P/100
90 PRINT:PRINT:PRINT
100 PRINT"ALAP","SZ.LAB","SZAZALEKERTEK"
110 PRINTA,P,E
120 END
```

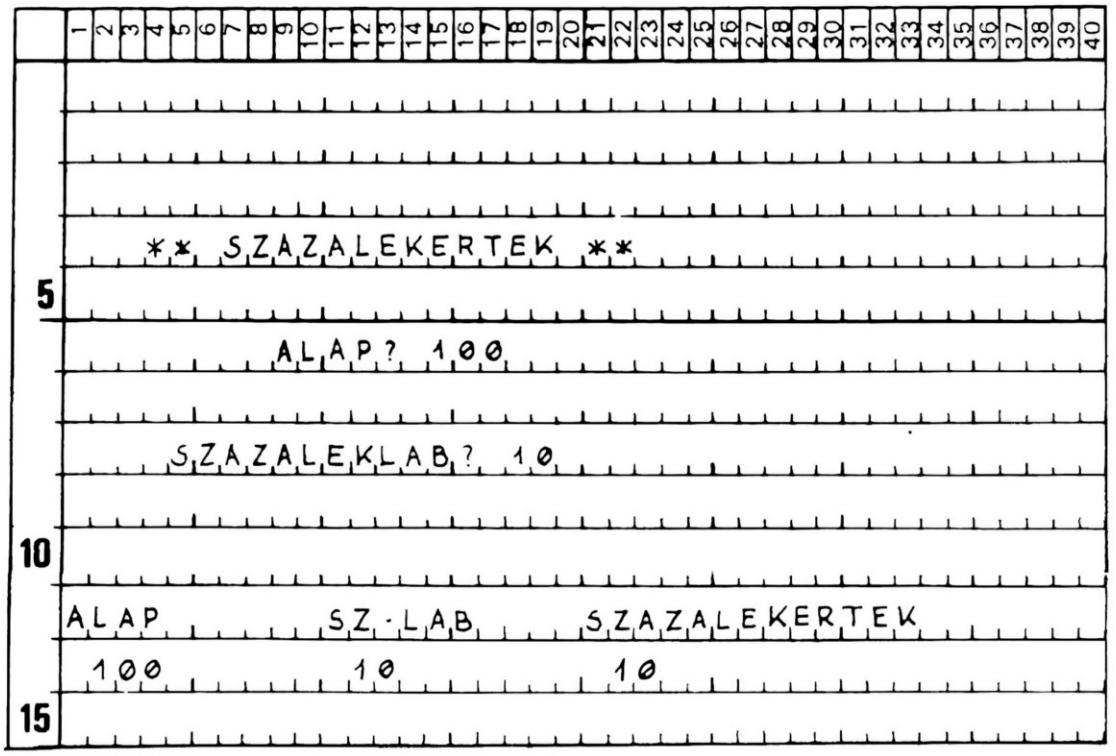
Tartalom:

1. A programmal végrehajtható feladat;
2. Képernyőtörlés és három sor emelés;
3. A program nevének kiírása;

4. Soremelés;
5. Az alap értékének bevitele;
6. Soremelés;
7. A százalékláb értékének bevitele;
8. A százalékérték számítása és hozzárendelése E változóhoz;
9. Három sor emelés;
10. Az eredmény zónaformában történő kiírásához a szövegek elhelyezése;
11. Az eredmény kiírása, a megfelelő zónákba;
12. Vége.

válasz a kérdésekre

1. Kiírná „SZÁZALÉKÉRTÉK SZÁMÍTÁS" szöveget.
2. A „SZÁZALÉKLÁB" és a „SZÁZALÉKÉRTÉK" szöveg együtt négy kijelzési zónát igényelne, egy sorban pedig csak négy zóna helyezhető el.
3.



4. Az 1. játéknál már tapasztalhattátok, hogy a kiíró utasításhoz rendelt elemeknél a pontosvessző után elhelyezkedő számváltozó értéke az előjel helyével kezdődően íródik ki. Ez a szabály a vesszőnél, mint elválasztójelnél is érvényesül. Ez a hely a szövegek esetében nem jön létre.

3. JÁTÉK

A százalékláb-számító program:

```
10 REM"SZAZALEKLAB SZAMITAS"
20 PRINT"␣"
30 LET A1$="SZAZALEK"
40 LET A2$="LAB"
50 LET A3$="ERTEK"
60 PRINT"  ** ";A1$;A2$;" SZAMITAS **"
70 PRINT" IRD BE AZ ALAPOT ES A ";A1$;A3$;"ET"
80 INPUT A,E
90 PRINT
100 LET P=E*100/A
110 PRINT TAB(8);"ALAP";A
120 PRINT TAB(2);A1$;A2$;P
130 PRINT A1$;A3$;E
140 END
```

Tartalom:

1. A program feladata;
2. Képernyőtörlés és soremelés;
3. Az A1$ változónak értékadás;
4. Az A2$ változónak értékadás;
5. Az A3$ változónak értékadás;
6. A program nevének kiírása, szöveg: SZÁZALÉKLÁB SZÁMÍTÁS;
7. A számítógépbe beírandó adatok megnevezésének kiírása, szöveg: ÍRD BE AZ ALAPOT ÉS A SZÁZALÉKÉRTÉKET;
8. Az adatok beírása (a két számot a beírásnál vesszővel kell elválasztani);
9. Soremelés;
10. A százalékláb kiszámítása és hozzárendelése a P változóhoz;
11. Az ALAP megnevezés és értékének kiírása;
12. A SZÁZALÉKLÁB megnevezés és értékének kiírása;
13. A SZÁZALÉKÉRTÉK megnevezés és értékének kiírása;
14. Vége.

válasz a kérdésekre

1. A program nevének kiírásánál ezzel biztosított a programozó a SZÁZALÉKLÁB és a SZÁMÍTÁS szavak között egy üres helyet.
2. REM utasítást. A számítógép a program futása közben a REM utasításra tovább halad.
3.

```
5
   ** SZAZALEKLAB SZAMITAS **
   IRD BE AZ ALAPOT ES A SZAZALEKERTEKET
? 100,10

        ALAP 100
   SZAZALEKLAB 10
SZAZALEKERTEK 10
```

4. JÁTÉK

A százalékszámításban az alapot számító program:

```
10 REM "AZ ALAP SZAMITASA"
20 PRINT "□"
30 LET A0$="ALAP":A1$="SZAZALEK":A2$="LAB"
40 LET A3$="ERTEK":A4$=" SZAMITAS"
50 PRINT TAB(3);A0$;A4$
60 PRINT
70 INPUT " SZAZALEKLAB,SZAZALEKERTEK";P,E
80 LET A=E*100/P
90 LET A=INT(A)
100 LET A5$=A1$+A2$:A6$=A1$+A3$
110 PRINT
120 PRINT A0$;TAB(7);A5$;TAB(21);A6$
130 PRINT:PRINT A;TAB(9);P;TAB(23);E
140 END
```

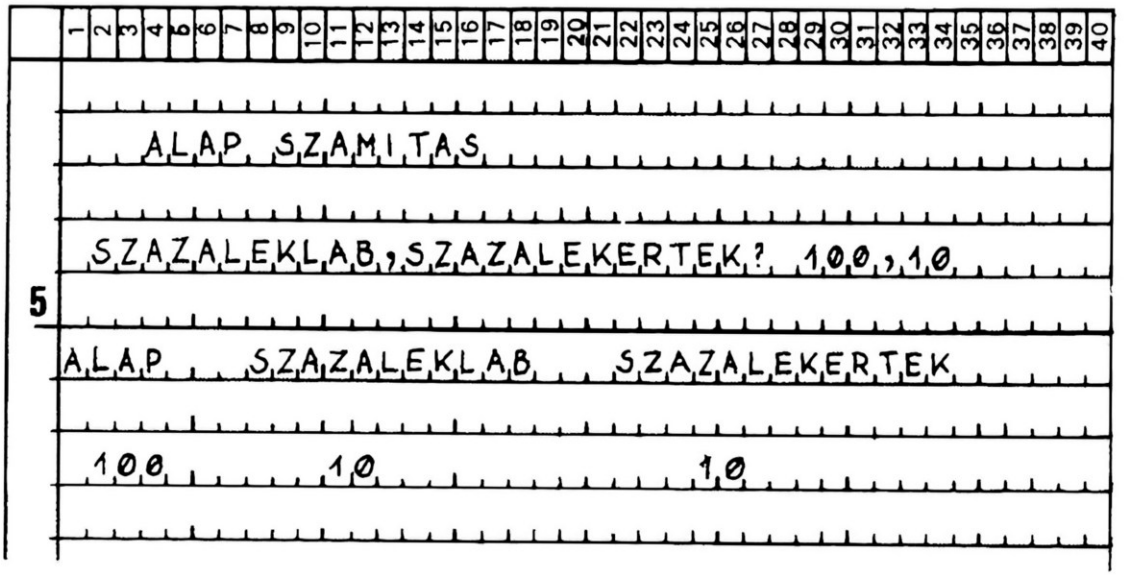
Tartalom:

1. A program feladata;
2. A képernyőtörlés és soremelés;
3. Változókhoz szövegek hozzárendelése;
4. Változókhoz szövegek hozzárendelése;

5. A program nevének kiírása, szöveg: ALAP SZÁMÍTÁS;
6. Soremelés;
7. Az adatok beírása;
8. Az alap kiszámítása és A változóhoz rendelése (az alap lehet törtszám);
9. Az alap törtrészének leválasztása (az INT(X) függvény a törtszám egész részét rendeli a változóhoz);
10. A „SZÁZALÉKLÁB" és „SZÁZALÉKÉRTÉK" szövegek hozzárendelése az A5$ és A6$ változókhoz;
11. Soremelés;
12. Az eredmény kiírásához a megnevezések kiírása;
13. Soremelés és az eredmény adatok kiírása;
14. Vége.

válasz a kérdésekre

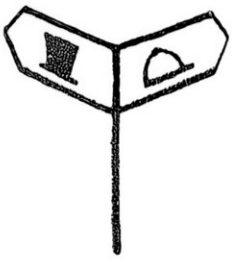
1. Utasításokat választ el.
2. Igen, a program további részében már nincs szükség az A1$ változó tartalmára, ezért új szöveg rendelhető hozzá.
3. Nem, az INPUT utasításnál a szöveget ki kell írni.
4.



FELTÉTELES utasítás

Formája: IF feltétel THEN sorszám vagy utasítás

Feladata: reláció vagy logikai kifejezés igaz, vagy hamis -értékének megfelelően vezérelni a program futását.



feltételes IF THEN

Ha az IF szó után álló feltétel igaz, úgy a THEN szó utáni sorszámon folytatódik a program, illetve, ha ott utasítás szerepel, akkor az utasításnak megfelelő művelet kerül végrehajtásra. Ha a feltétel hamis, úgy a következő programsoron folytatódik a program futása. A feltételben megadott kifejezésnek kiértékelhetőnek kell lennie.

Az utasítással végrehajtható műveletek	*A programsorok formája*
Ha A< B feltétel igaz, a program 100. programsorral folytatódjon.	`50 IF A<B THEN 100`
A=1 és B=2 feltétel esetén legyen A=B↑2	`60 IF A=1 AND B=2 THEN A=B↑2`
Az IGEN-re a program a 10. programsorral folytatódjon	`70 IF A$="IGEN" THEN 10`
Ha X< Y vagy X> Z írja ki Z értékét.	`80 IF X<Y OR X>Z THEN PRINT Z`

ugró GOTO

UGRÓ utasítás

Formája: GOTO sorszám

Feladata: az utasítássorozat végrehajtása a sorszámnak megfelelő programsorral folytatódjék.

A GOTO szó után annak a programsornak a sorszáma áll, ahol a programfutásnak folytatódnia kell. Az utasítás alkalmazható az IF utasítással együtt. Ebben az esetben az ugrás feltételhez kötött.

Az utasítással végrehajtható műveletek	*A programsorok formája*
A program a 100. programsoron folytatódjon	`30 GOTO 100`
Ha A< B, a program a 100. programsoron folytatódjon	`40 IF A<B GOTO 100`
Ha A=10, akkor kerüljön végrehajtásra A=B+C és a programfutás a 100. programsoron folytatódjék	`50 IF A=10 THEN A=B+C: GOTO 100` *Figyelem: Összetett programsor, ezért a GOTO utasítás elé kettőspontot kell tenni.*

ciklus
FOR TO STEP

CIKLUS utasítás

Formája: FOR változó= kifejezés TO kifejezés STEP kifejezés NEXT változó

Feladata: a program futásában ciklusok, ismétlések szervezése.

Az utasításban a FOR szó után a ciklusszámláló kezdeti értékét, a TO szó után a végértékét, a STEP szó után a lépésközt állítjuk be. Ha a lépésköz értéke egy, akkor a STEP szót nem kell kiírni. A NEXT szó hatására a gép a ciklusszámlálást végzi el.

ciklusszámláló
NEXT

A FOR TO STEP és a NEXT szavak közé helyezzük el azokat az utasításokat, amelyeket a ciklusban legalább egyszer végre akarunk hajtani. (Ezek az utasítások tartoznak a ciklus törzsébe.) Ha negatív számmal határoztuk meg a lépésközt, úgy a NEXT utasítás visszafelé számlál.

Az utasítással végrehajtható műveletek	*A programsorok formája*

Egytől tízig add össze az egész számokat!	`10 LET A=0`
	`20 FOR I=1 TO 10`
	`30 LET A=A+1`
	`40 NEXT I`

összetett programsorral

```
10 LET A=0
20 FOR I=1 TO 10: A=A+1: NEXT
```

A változó értékétől
B változó értékéig
0.1 lépésenként végezz
összegzést!

```
30 INPUT A,B
40 LET C=0
50 FORI=A TO B STEP.1
60 LET C=C+1
70 NEXT I
```

A+2 kifejezés értékétől
B+3 kifejezés értékéig a
9-ik oszlopban írjon a
gép egymás alá
csillagokat

```
10 LET A$="*"
20 INPUT A,B
30 FOR I=A+2 TO B+3
40 PRINT TAB(8);A$
50 NEXT I
```

tömbváltozó

A tömbváltozókról még nem volt szó. Mielőtt rátérnénk a HELYKIJELÖLŐ utasítás tárgyalására, értelmezzük a tömbváltozó fogalmát. A számítógéppel megoldandó feladatok között szerepelhetnek olyanok, amelyekben sok adattal kell ugyanolyan műveletet elvégezni. Ilyen feladatoknál nehézkes lenne minden adatot (számot, szöveget) külön-külön a megfelelő változókhoz rendelni. Ezért úgynevezett tömbváltozóhoz rendeljük az adatokat. A tömbváltozó lehet szám- és szövegváltozó, egy-, kettő- vagy háromdimenziós. Egydimenziós tömbváltozóhoz egy adatsorozatot rendelhetünk, a kétdimenzióshoz egy adattáblázatot, a háromdimenzióshoz több adattáblázatot.
Például: egydimenziós tömbváltozó A(9). Az A nevű változóhoz 0,1,2,3,4,5,6,7,8,9 elemekből álló adatsorozat rendelhető a következő programsorokkal:

```
10 FOR I=0 TO 9
20 INPUT A(I)
30 NEXT I
```

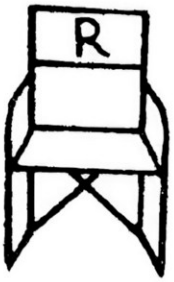
Az A (9,9) kétdimenziós tömbváltozóhoz viszont már tíz sorból és tíz oszlopból álló adattáblázat elemei rendelhetők hozzá.

```
10 FOR I=0 TO 9
20 FOR J=0 TO 9
30 INPUT A(I,J)
40 NEXT J
50 NEXT I
```

Az A (9,9,9) háromdimenziós tömbváltozóhoz tíz adattáblázat elemei rendelhetők hozzá – amelyek mindegyike tíz sorból és tíz oszlopból áll.

```
10 FOR I=0 TO 9
20 FOR J=0 TO 9
30 FOR Z=0 TO 9
40 INPUT A(I,J,Z)
50 NEXT Z
60 NEXT J
70 NEXT I
```

A legtöbb személyi számítógépnél szövegsorozat csak egydimenziós szöveges tömbváltozóhoz rendelhető.



helykijelölő
DIM

HELYKIJELŐLÖ utasítás

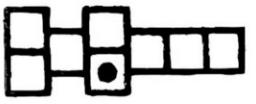
Formája: DIM tömbváltozó (elemek száma, elemek száma, elemek száma), ...

Feladata: a tömbváltozók helyének kijelölése az operatív tárban.

A tömbök méretét csak a tárolóhely mérete korlátozza.

Az utasítással végrehajtandó műveletek	*A programsorok formája*
15 valós szám tömbjének kijelölése	`10 DIM X(14)`
15×15 egész szám tömbjének kijelölése	`20 DIM X%(14,14)`
12 szövegből álló tömb kijelölése	`30 DIM A$(11)`

TÖBB HELYRE UGRÓ utasítás (esetszétválasztás)



több helyre ugró
ON GOTO

Formája: ON kifejezés GOTO sorszám, sorszám, ...
Feladata: a kifejezés értékének megfelelő sorszámú programsorra való ugrás.

Ha a kifejezés értéke 0 vagy több mint a sorszámok száma, úgy a programfutás a következő programsoron folytatódik.

Az utasítással végrehajtható művelet	*A programsorok formája*
Az N értéke: ha 1 akkor 100. ha 2 akkor 200. ha 3 akkor 300. programsoron folytatódjék a programfutás	`40 ON N GOTO 100,` `200,300`
Az INT(N)+ABS(X) értéke: ha 1 akkor 150. ha 2 akkor 250. ha 3 akkor 350. programsoron folytatódjék a programfutás.	`50 ON INT(N)+ABS(X)` `GOTO 150,250,350`

FÜGGVÉNY DEFINIÁLÓ utasítás

függvény definiáló
DEF FN

Formája: DEF FN függvény név (változó)-kifejezés
Feladata: A programban egy kifejezésnek függvényként való meghatározása.

Olyan programsorokban alkalmazzuk, amelyekben egy kifejezést többször kell utasításokhoz rendelni.
Például X↑3+2*X↑2 kifejezést függvényként definiálni a következőképpen lehet:

```
30 DEF FNQ(X)=X↑3+2*X↑2
```

függvény név

Az így definiált függvény a programban a következőképpen írható be:

```
50 INPUT X
60 LET A=FNO(X)
```

Elég volt a tanulásból játsszatok!

5. JÁTÉK

Számtani sorozat n-ik elemét és a sorozat összegét számító program:

```
10 REM "SZAMTANI SOROZAT N-EDIK ELEMENEK ES"
20 REM "AZ OSSZEGENEK S(N) KISZAMITASA"
30 PRINT"␣":PRINT
40 PRINT"      *** SZAMTANI SOROZAT 1 ***"
50 PRINT
60 INPUT"  ELSO ELEM A(1)";A1
70 INPUT"  DIFFERENCIA D";D
80 INPUT"  ELEMEK SZAMA";N
90 LET AN=A1+(N-1)*D
100 LET SN=(A1+AN)*N/2
110 PRINT:PRINT
120 PRINT" A(1)=";A1,"N=";N
130 PRINT" A(N)=";AN,"S(N)=";SN
140 PRINT:PRINT
150 INPUT"  TOVABB(I/N)";E$
160 IF E$="I" THEN 20
170 END
```

Tartalom:

1. A program feladata;
2. A feladat leírásának folytatása;
3. Képernyőtörlés és kettő sor emelés;
4. A program nevének kiírása;
5. Soremelés;
6. Az első elem értékének beírása;
7. A két elem közötti különbség beírása;
8. A sorozat elemszámának beírása;
9. A n-ik elem kiszámítása és hozzárendelése az AN változóhoz;
10. A sorozat összegének kiszámítása és hozzárendelése az SN változóhoz
11. Kettő sor emelés;

12. A sorozat első elemének és az elemek számának kiírása;
13. Az n-ik elem és a sorozat összegének kiírása;
14. Kettő sor emelés;
15. Rákérdezés a programfutás megismétlésére és a válasz beírása;
16. A válasz elemzése, és I billentyű leütése a program futás átadása a 30. programsorra Minden más billentyű lenyomására a programfutás folytatódik a 170. programsoron;
17. Vége.

válasz a kérdésekre

1. A programfutás az 170. programsoron folytatódik hiszen a 160. programsorban a feltétel nem igaz, vagyis hamis.
2. Vannak olyan géptípusok, amelyek kiírásnál a szöveg után automatikusan pontosvesszőt olvasnak, ezért hiányát nem jelzik ki hibának, és kiírják a következő számváltozó értékét. Más géptípusoknál viszont hibát jelez a gép. Ezért minden esetben célszerű a pontosvessző használata.
3.

```
      ***  SZAMTANI  SOROZAT  1  ***
 5  ELSO ELEM A(1)? 1
    DIFFERENCIA D? 2
    ELEMEK SZAMA? 3
10  A(1)= 1   N= 3
    A(N)= 5   S(N)= 9
    TOVABB(I/N)? N
```

6. JÁTÉK

A mértani sorozat n-ik elemét és a sorozat összegét számító program:

```
10 REM "MERTANI SOROZAT N-EDIK ELEMENEK ES"
20 REM "OSSZEGENEK S(N) KISZAMITASA"
30 PRINT"□":PRINT
40 PRINT"   —— MERTANI SOROZAT SZAMITAS ——"
50 PRINT:PRINT
60 INPUT"  A(1), Q, N";A1,Q,N
70 LET AN=A1*Q↑(N-1)
80 LET SN=A1*(Q↑N-1)/(Q-1)
90 PRINT"□":PRINT
100 PRINT" A(1)=";A1,"N=";N
110 PRINT" A(N)=";AN,"S(N)=";SN
120 PRINT:PRINT
130 INPUT"  TOVABB I/N";E$
140 IF E$="I" THEN 30
150 END
```

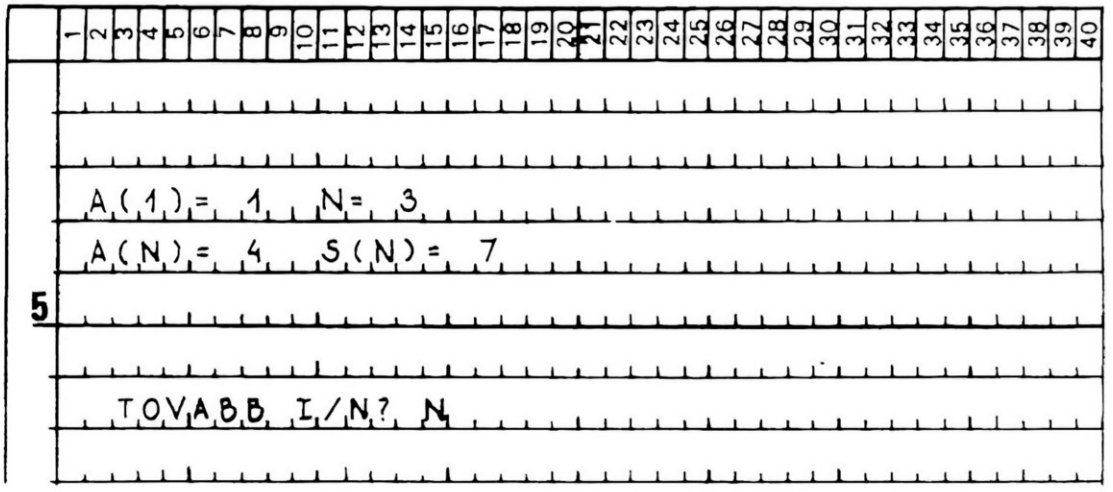
Tartalom:

1. A program feladata;
2. A feladat leírásának folytatása;
3. Képernyőtörlés és kettő sor emelés;
4. A program nevének kiírása;
5. Kettő sor emelés;
6. Az A1, Q és az N értékeinek bevitele;
7. Az An értékének kiszámítása;
8. Az Sn értékének kiszámítása;
9. Képernyő törlés és kettő sor emelés;
10. Az első elem és az elemek számának kiírása;
11. A sorozat n-ik elemének és összegének kiírása;
12. Kettő sor emelés;
13. FOLYTASSUK I/N kérdés kiírása;
14. I betű leütése esetén a programfutás a 2. pontnál folytatódik;
15. Vége.

válasz a kérdésekre

1. A1 szer Q az N mínusz 1-ediken,
   A1 szer Q az N mínusz 1-ediken osztva Q mínusz 1-el.

2.



A számtani sorozat első elemét és a sorozat összegét két tetszőlegesen megadott elemből számító program:

7. JÁTÉK

```
10 REM"SZAMTANI SOROZAT ELSO N ELEMENEK ES OSSZEGENEK"
20 REM"SZAMITASA, HA ADOTT KET TETSZOLEGESEN VALASZTOTT ELEM"
30 PRINT"␣":PRINT
40 PRINT"    *** SZAMTANI SOROZAT 2 ***
50 PRINT:PRINT
60 INPUT" N,I,A(I),K,A(K)";NV,I,AI,K,AK
70 IF 1<NV AND 0<I AND I<K THEN 100
80 PRINT " HIBAS ADAT"
90 GOTO 60
100 LET D=(AK-AI)/(K-I)
110 LET N=1
120 LET AN=AI-(I-1)*D
130 LET SN=AN
140 PRINT" N","A(N)","S(N)"
150 PRINT N,AN,SN
160 FOR N=2 TO NV
170 LET AN=AN+D
180 LET SN=SN+AN
190 PRINT N,AN,SN
200 NEXT N
210 PRINT
220 INPUT" TOVABB(I/N)";T$
230 IF T$="I" THEN 10
240 END
```

Tartalom:

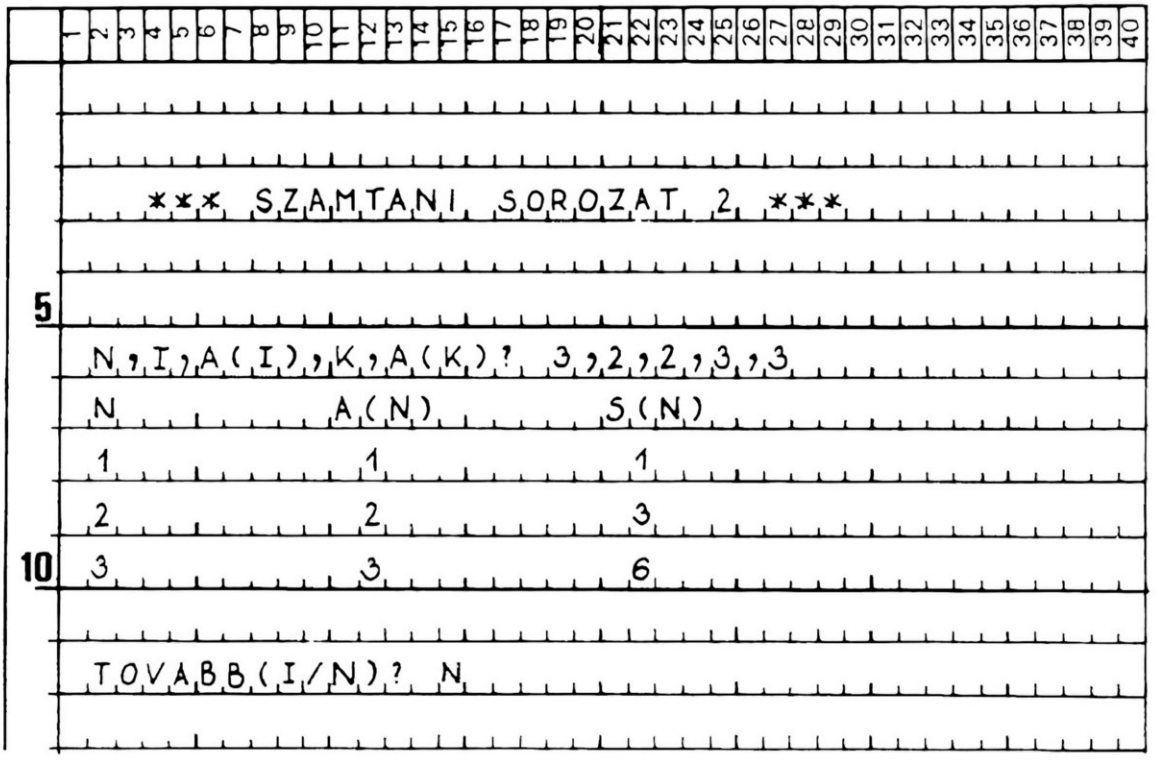
1. A program feladata;
2. A feladat leírásának folytatása;
3. Írjátok ti be beírtad?
4. beírtad?

5. beírtad?
6. beírtad?
7. A bevitt adatok vizsgálata, a programfutás a 100. programsoron folytatódik: ha az elemek száma nagyobb, mint egy, ha az I index nullánál nagyobb, ha K index nagyobb, mint I index. Minden más esetben a programfutás a 80. programsoron folytatódik;
8. A kiírja a „HIBÁS ADAT" szöveget;
9. A programfutást a 60. programsorra ugratja. Ezzel az az adatbevitel megismétlését kéri a program;
10. A differencia kiszámítása és hozzárendelése D változóhoz;
11. beírtad?
12. A sorozat első elemének a kiszámítása és hozzárendelése An változóhoz;
13. Tekintve, hogy a sorozat első elemét számítottuk, az összeg egyenlő az első elemmel;
14. A táblázat fejlécének a kiírása;
15. Az első elem adatainak kiírása;
16. Ciklusszervezés a sorozat többi elemeinek a kiszámítására;
17. Az n-ik elem számítása;
18. Az n-ik elemnél a sorozatösszeg kiszámítása;
19. Az n-ik elem adatainak a kiírása;
20. A ciklus számlálás;
21. beírtad?
22. beírtad?
23. beírtad?
24. beírtad?

válasz a kérdésekre

1. A 170. 180. és 190. programsorok
2. Az elem értékét úgy, hogy az előző elem (n–1) értékéhez hozzáadja a két tag közötti különbséget. Az elemek összegét pedig úgy, hogy az előző elemhez tartozó összeghez hozzáadja az elem értékét.
3. A kezdő elemnél az összeg megegyezik az első elem értékével, és az elemek összegének kiszámításához erre az értékre szükség van.

4. Ha az adatbevitel: 3, 2, 2, 3, 3, és 220. programsorban N betűt ütöttél le, ezt olvasod le a képernyőről.



Függvények helyettesítési értékeit számító program: **8. JÁTÉK**

```
10 REM "FUGGVENYEK HELYETTESITESI ERTEKEINEK SZAMITASA"
20 DIM X(20)
30 PRINT"␣":PRINT TAB(10);"* FUGGVENYEK *"
40 DEF FNO(X)=X*X
50 PRINT:PRINT" SZAMITASI LEHETOSEGEK:"
60 PRINT"1- EGY FUGGVENY EGY HELYEN"
70 PRINT"2- TØBB FUGGVENT EGY HELYEN"
80 PRINT"3- EGY FUGGVENY TOBB HELYEN":PRINT
90 INPUT"   VALASZTOTT ESET SZAMA";A%
100 IF 4>A% AND A%>0 THEN 130
110 PRINT"NINCS ILYEN ESET"
120 GOTO 90
130 ON A% GOTO 140,190, 250
140 PRINT"␣":PRINT"Y=X*X FUGGVENYT EGY HELYEN"
150 INPUT"IRD BE X ERTEKET";X
160 LET Y=FNO(X)
170 PRINT:PRINT,"X=";X,"Y=";Y
180 END
190 PRINT"␣":PRINT" Y1=2*X-3,Y2=-X+6, Y3=X↑3+X↑2+X"
200 PRINT"FUGGVENYEKET EGY HELYEN"
210 INPUT" IRD BE AZ X ERTEKET";X
220 LET Y1=2*X-3:Y2=-X+6:Y3=FNO(X)*X+FNO(X)+X
230 PRINT:PRINT"X=";X," Y1"," Y2"," Y3":PRINT,Y1,Y2,Y3
```

```
240 END
250 PRINT"␣":PRINT" Y=X↑2 FUGGVENYT TOBB HELYEN"
260 INPUT"HELYEK SZAMA (2-20)";N
270 IF 20<N OR N<2 THEN PRINT TAB(25);"HIBA":GOTO260
280 PRINT:PRINT"IRD BE AZ X(I) ERTEKEKET"
290 FOR I=1 TO N
300 INPUTX1(I)
310 NEXT I
320 PRINT:PRINT," X"," Y"
330 FOR I=1 TO N
340 LET X=X1(I):Y=FNO(X)
350 PRINT,X,Y
360 NEXT I
370 END
```

Tartalom:

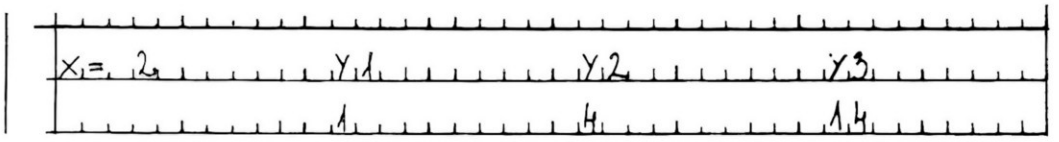
1. A program feladata;
2. 20 elemből álló számsorozat helyének kijelölése;
3. Képernyőtörlés, soremelés, a program nevének kiírása;
4. Az X*X kifejezés definiálása függvényként;
5.
6.
7.
8.
9.
10. A bevitt szám ellenőrzése. A szám 1, 2 vagy 3 lehet ha a feltétel teljesült, a 130. programsoron folytatódik a program futása;
11. Ha negatív szám, nulla vagy 3-nál nagyobb szám került bevitelre, akkor kiíródik a képernyőre a NINCS ILYEN ESET szöveg;
12. A programfutás a 90. programsoron folytatódik;
13. A programfutás elágazik a választott számítási változat szerint;
14. A képernyőtörlés, soremelés, a számítási változat megnevezésének kiírása;
15. Az X változó értékének bevitele;
16. A definiált függvény értékének hozzárendelése Y változóhoz;
17.
18. Az 1– változat feladatának elvégzésével a programfutás befejeződött;

19.
20.
21.
22. A függvényértékek kiszámítása, a definiált függvény alkalmazása;
23.
24.
25.
26. Az $y=x^2$ függvény számítási helyeinek egyenkénti bevitele;
27. Ellenőrzés, ha 20-nál nagyobb és 2-nél kisebb a bevitt szám, a programfutás a 260. programsoron folytatódik, miközben kiíródik a HIBA szöveg.
Az adatbevitelt ismételni kell;
28.
29. Az X értékének bevitelét szervező ciklus feje;
30. X értékeinek bevitele, ciklustörzs;
31. Ciklusszámlálás;
32. Az eredmény megnevezéseinek kiírása;
33. A számítást és kiírást szervező ciklus feje;
34. A számítási helyek X1(I) értékeinek hozzárendelése az X változóhoz, a függvények értékének kiszámítása és hozzárendelése Y változóhoz;
35. Az eredmény kiírása;
36. Ciklusszámlálás;
37.

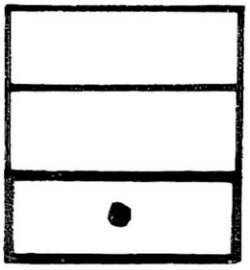
válasz a kérdésekre

1. A programban négy rész különböztethető meg;
A program feje, 10 – 130 programsor,
az egy függvény egy helyen számítási egység,
140 – 180 programsor,
a több függvény egy helyen számítási egység,
190 – 240 programsor,
az egy függvény több helyen számítási egység,
250 – 370 programsor.

2.



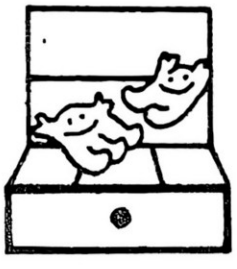
3. Mert az FNO(X) függvény, így: FNO(X1(I)) nem írható le. Ezt a számítógép nem tudja értelmezni



adattároló
DATA

ADATTÁROLÓ utasítás
Formája: DATA adat, adat,...
Feladata: az adatok tárolása a programban.
Olyan esetben alkalmazzuk, ha a program kidolgozásakor már ismertek a feldolgozandó adatok.

Az utasítással végrehajtható műveletek	*A programsorok formája*
Tárolja a program 5,15, 30,3.15,0.35 számokat	`40 DATA 5,15,30, 3.15,.35`
Tárolja a program ALAP, SZÁZALÉKLÁB, SZÁZALÉKÉRTÉK, KISZÁMÍTÁSA szövegeket	`50 DATA "ALAP","SZA ZALEKLAB","SZAZALEK ERTEK","KISZAMITASA"`
Tárolja a program 15, ES, VALAMINT, 13 számokat és szövegeket vegyesen	`60 DATA 15,"ES", "VALAMINT",13`



adatbeolvasó
READ

ADATBEOLVASÓ utasítás
Formája: READ változó, változó...
Feladata: a programban tárolt adatok változókhoz való rendelése.
A DATA szó után álló adatok típusának megfelelően és sorrendben kell a READ szó után a változókat írni.

Az utasítással végrehajtható művelet	*A programsorok formája*
Rendeljen az A%, B%, A1, BETA, AA változókhoz adatokat.	`70 READ A%,B%,A1, BETA,AA`

Az adatok az ADATTÁROLÓ utasítás 1. példa programsorában vannak.

Rendeljen A$, B$, C$, A1$, változókhoz adatokat.	`80 READ A$,B$,C$, A1$`

Az adatok az ADATTÁROLÓ utasítás 2. példa programsorában vannak.

Rendeljen X1, X1$, C2$ X3% változókhoz adatokat.	`90 READ X1,X1$,C2$, X3%`

Az adatok az ADATTÁROLÓ utasítás 3. példa programsorában vannak.

ADATBEOLVASÁST ISMÉTLŐ utasítás

Formája: RESTORE

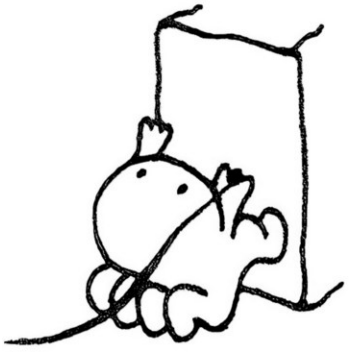
Feladata: az adatbeolvasás ismételt végrehajtása.

Az ADATTÁROLÓBAN lévő adatelemet, ha az ADATBEOLVASÓ utasítással már változókhoz rendeltük, ismételten változóhoz csak az ADATBEOLVASÁST ISMÉTLŐ utasítás alkalmazásával rendelhetjük. A RESTORE hatására az utána következő programsorban levő READ változóihoz a programban elsőnek álló DATA adatelemek rendelődnek.

adatbeolvasást ismétlő RESTORE

Az utasítással végrehajtható művelet	*A programsorok formája*

Az 5,15,30, ES adatelemeket kettő változóhoz rendeljük hozzá	`10 DATA 5,15,30,"ES"`
	`20 READ A%,B%,C,Y$`
	`:`
	`:`
	`80 RESTORE`
	`90 READ S1,S2,S3,Q$`



szubrutin hívó
GOSUB

SZUBRUTINHÍVÓ utasítás
Formája: GOSUB sorszám.
Feladata: a program futását a szubrutin első programsorára léptetni.
Szubrutinoknak nevezzük a programban többször felhasználásra kerülő programrészeket. A SZUBRUTINHÍVÓ utasítás csak a VISSZA utasítással párban alkalmazható, ezért itt a példáktól tekintsünk el.

vissza
RETURN

VISSZA utasítás
Formája: RETURN
Feladata: a szubrutin programrész végétől a program futását visszaléptetni a SZUBRUTINHÍVÓ utasítást tartalmazó programsor utáni programsorra.

Az utasítással végrehajtható művelet

Szubrutinnal írjon ki a gép képernyőre, a 11–30 oszlopba egymás alá két csillag sort, két sortávolsággal

A programsorok formája

```
5 GOTO 50
10 FOR I=1 TO 20
20 PRINT TAB(I+9);"*";
30 NEXT I
40 RETURN
50 PRINT "■"
60 GOSUB 10
70 PRINT:PRINT
80 GOSUB 10
90 END
```

A 20. programsor végén, ha nem tesszük ki a ; jelet, akkor a csillagokat függőlegesen írja a képernyőre a gép.

több szubrutinra ugró
ON GOSUB

TÖBB SZUBRUTINRA UGRÓ utasítás
Formája: ON kifejezés GOSUB sorszám, sorszám...
Feladata: a meghatározott szubrutinra való ugrás vezérlése.

A VISSZA utasítással együtt kell alkalmazni, úgy mint a SZUBRUTINHÍVÓ utasításnál.

állj
STOP

ÁLLJ utasítás
Formája: STOP
Feladata: a programjavítások elősegítése.
Akkor alkalmazzuk, amikor a program futását a számunkra kérdéses programsornál meg akarjuk állítani.

Ezzel a legtöbb számítógéptípusnál alkalmazható BASIC utasítások ismertetését befejeztük. Az ismertetett és kártyákra rögzített 18 utasításnál a BASIC nyelv több utasítással rendelkezik, de ezek különböző számítógéptípusoknál eltérő szabályok szerint funkcionálnak.

PROGRAMKIDOLGOZÁS TERVEZÉSE

A 7. és a 8. játéknál tapasztalhattátok, hogy a program folyamatában új jelenség jelentkezett. Megnőtt a programsorok – ezen belül is az összetett programsorok – száma és a program egészének áttekinthetősége csökkent. Ez a jelenség a számítógépes programozás természetes velejárója. A program méretének és az összetett programsorok számának növekedése egy bizonyos határ után még a gyakorlott programozóknál is oda vezetett, hogy nem tudták áttekinteni a kidolgozott programjukat.

Ezért szükségessé vált, hogy a programozás folyamatára különböző módszereket dolgozzanak ki. Ezek a módszerek megnövelik a programok áttekinthetőségét, és lehetőséget adnak arra, hogy egy programon egy időben több programozó is dolgozzon, ugyanakkor a programok ellenőrzése, a hibák kijavítása egyszerűsödik.

strukturált programozás

Napjainkban a programozásra kidolgozott módszerek közül a fenti követelményeket legjobban az úgynevezett *strukturált programozás módszere* elégíti ki.

A módszer nagyméretű programok kidolgozására és ellenőrzésére készült. Elsősorban a hivatásos programozók alkalmazzák munkájukban, azonban úgy gondoljuk, hogy helyesen járunk el, ha Titeket már most megismertetünk a strukturált programozási módszer alapjaival és az ÁSZ–BASIC kártyajátékba folyamatosan bekapcsoljuk a módszer elemeit.

Kezdjünk hozzá!

A célunk kezdetben nem a nagyméretű programok kidolgozásának tanítása, hanem hogy a módszer segítségével tanuljátok meg elemezni a már kidolgozott programokat és tudjátok meghatározni a program tartalmát.

Ha kezetekbe vesztek egy programot, a legfontosabb feladat a program egészének áttekintése anélkül, hogy az egyes programsorok tartalmát egyenként értelmeznétek. Ha nem ezt teszitek, könnyen úgy járhattok, hogy a „fától nem látjátok az erdőt", vagyis a programsorokból nem tudjátok megállapítani, hogy milyen feladatot végez el a program.

A strukturált programozással kidolgozott program olyan részekből áll, amelyek nagyobb nehézség nélkül áttekinthetők. Ezáltal a program egészének áttekintéséhez elegendő a részek helyének és szerepének tisztázása a programban. A részek helyének és szerepének tisztázásával viszont eljuthatunk a program felépítésének, rendezettségének megismeréséhez, majd ezen keresztül meghatározhatjuk a program feladatát.

A strukturált programozás módszere *ugyanakkor a számítógéppel végrehajtandó feladatoknak olyan kidolgozását teszi lehetővé, amelyekkel megtervezhető a program felépítése, részeinek helye a programban és a részek szerepe.*

A programnak ez a kétirányú megközelítése:

1. részek szerepe – helye a programban – a program rendezettsége – feladata;
2. a program feladata – rendezettsége – részek helye – – szerepe

lehetőséget ad arra, hogy elolvassátok a programokat, illetve a programokhoz csatolt leírásokat. Megállapíthassátok, hogy az adott programok mit tudnak szolgáltatni, és ha szükséges, bizonyos átalakítást is végezzetek a programokon.

Először ismerjétek meg, hogy a strukturáltan kidolgozott programból hogyan lehet következtetni a program feladatára és összetételére.

A strukturált programozásban az önálló szerepkört betöltő részeket moduloknak nevezzük. **modul**

A programokban a modulok sokféle szerepet tölthetnek be, ezért a szerep lényegét – vagyis azt a feladatot amelyet a modul a programban végrehajt – a modul

kezdő programsorában le kell írni, mint ahogyan azt a programok feladatainak leírásánál láttátok.
Ennek a programsornak a programban kettős rendeltetése van:

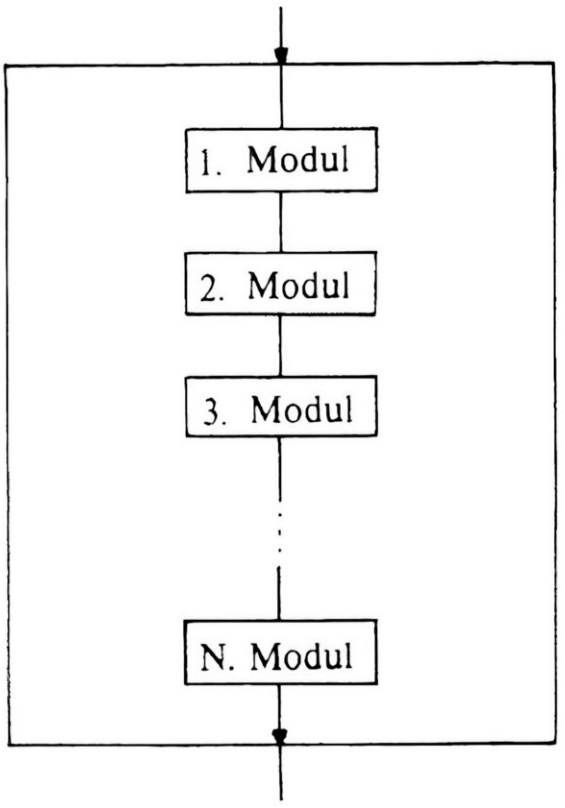
1. a modul szerepkörének egyértelmű azonosítása;
2. a modul kezdetét jelöli, a programfutás közben a modul végrehajtása csak itt kezdődhet el.

A valóságban a számítógép végrehajt olyan utasítást is, amellyel a programfutás beugratható a modul közepébe, de ez a megoldás a strukturált programozásban tilos!
A modul helye a programban kétféle lehet: főprogram és alprogram. Az alprogramoknak három típusát különböztetjük meg: beviteli, adatfeldolgozó és kiíró alprogramokat, illetve ilyen szerepeket betöltő modulokat.

alap vezérlőstruktúrák

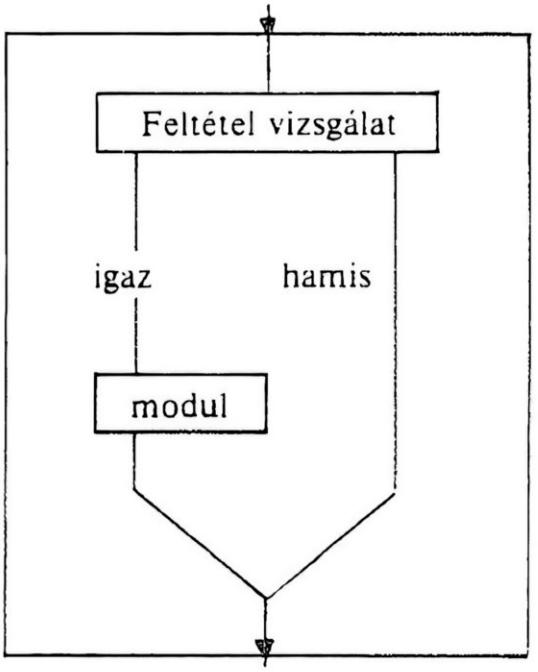
A modulok rendezettségét a programban a fő- és az alprogramok közötti kapcsolatok határozzák meg. E kapcsolatoknak öt alaptípusát, más néven alap vezérlőstruktúráját különböztethetjük meg.

soros vezérlőstuktúra

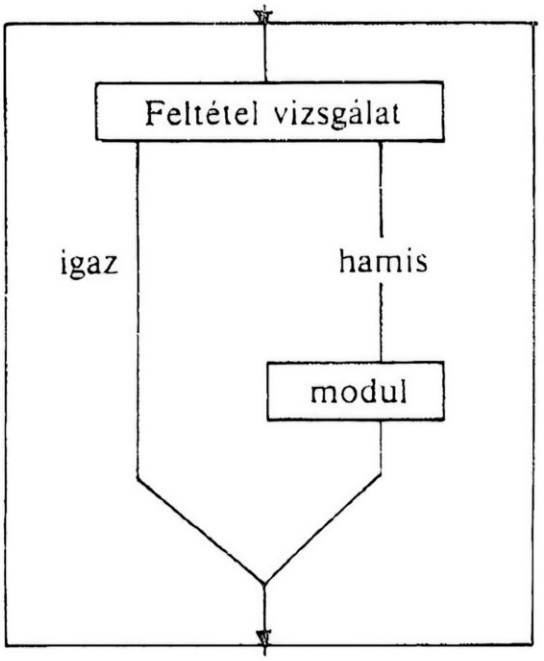


A soros vezérlőstruktúra a modulok egymás utáni végrehajtását vezérli

Két formáját különböztetjük meg:
Első az *IF THEN vezérlőstruktúra*



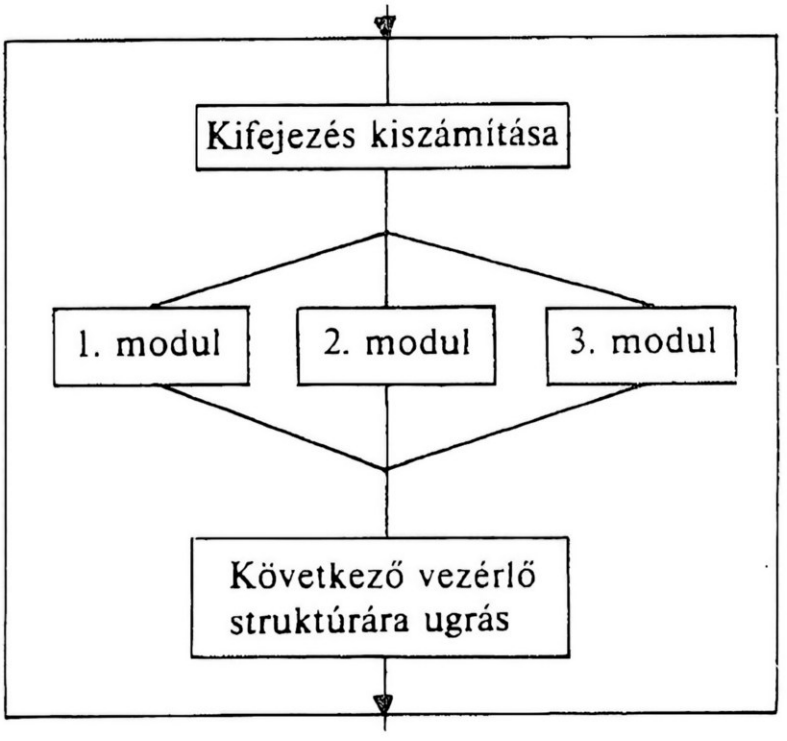
Ha a feltétel igaz, végrehajtásra kerül a modul, ha a feltétel hamis, úgy a következő vezérlőstuktúrára lép a program.



A feltételeket fordítottan is beállíthatjuk, ha a feltétel hamis, akkor kerül a modul végrehajtásra.

A második
az ON GOTO vezérlőstruktúra

A kifejezés aktuális értékének megfelelő modul kerül végrehajtásra

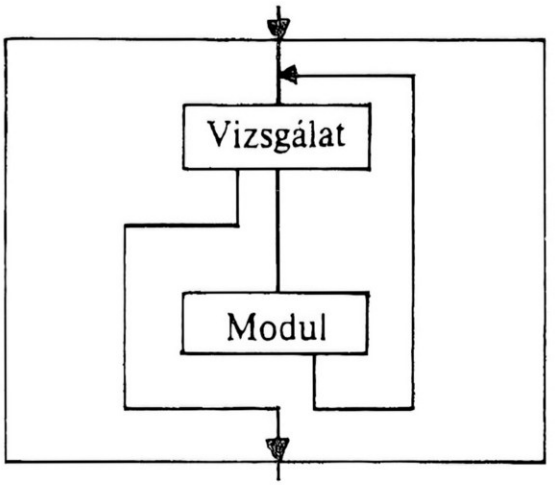


Ugyanezt a műveletet hajtja végre az ON GOSUB utasítás is.

Az ugrás a következő vezérlőstruktúrára ennél az utasításnál a RETURN utasításra következik be.

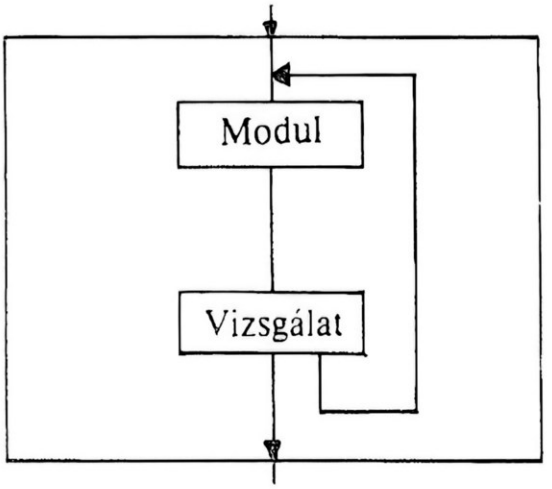
ciklus vezérlő-struktúrák

Két formáját különböztetjük meg.



Első a *modul előtt vizsgáló vezérlőstruktúra*

A vizsgálatot az IF THEN utasítás szerkezettel végezhetjük el.
A ciklus mindaddig folytatódik, amíg a feltétel igazról hamisra vált.



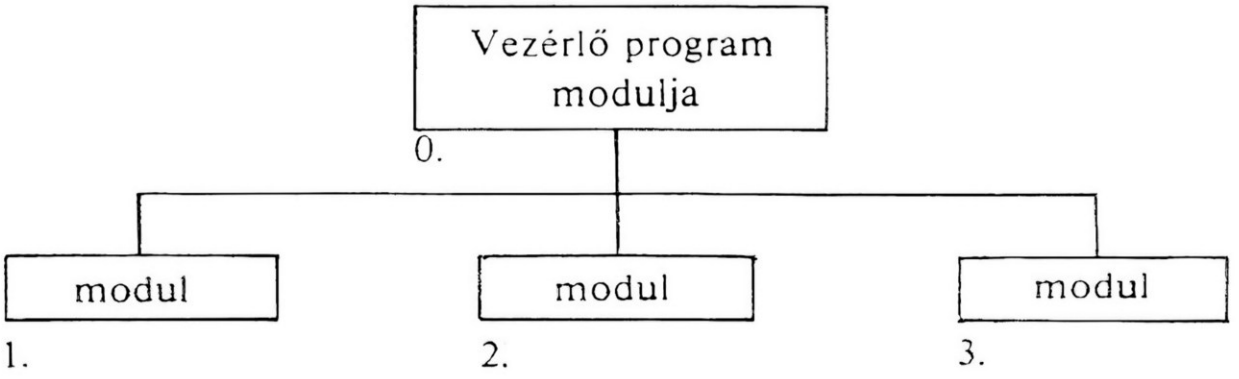
A második a *modul után vizsgáló vezérlőstruktúra*

A vizsgálatot a FOR NEXT, illetve az IF THEN utasítással, utasításokkal végezhetjük el.

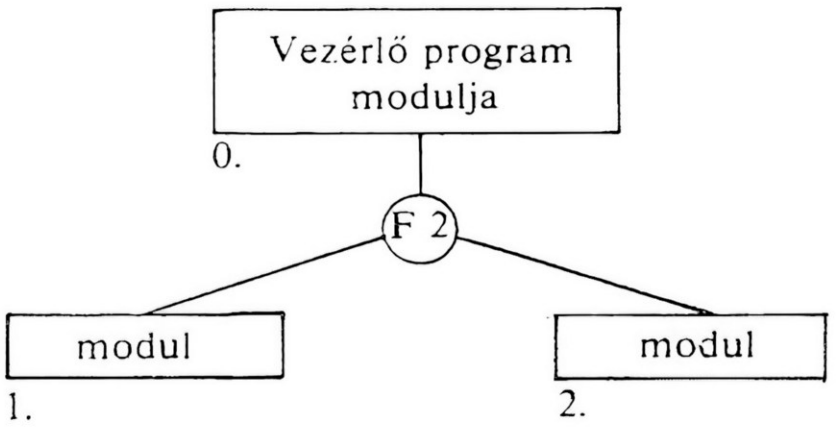
A vezérlőstruktúrák alkalmazása amellett, hogy kemény követelményeket állít a programozóval szemben, haszonnal is jár. Jelentősen megkönnyíti a program megtervezését, növeli az áttekinthetőséget, lehetőséget nyújt a lépésenkénti elemzésre, és végül az így kidolgozott programok bővíthetősége egyszerűbb.
A strukturált programban egy modul egyben önmagára és az alárendelt modulokra vonatkozó vezérlőstruktúra is. Ezt az elrendezettséget a program tervében ábrázolni is kell.

vezérlő-struktúrák ábrázolása

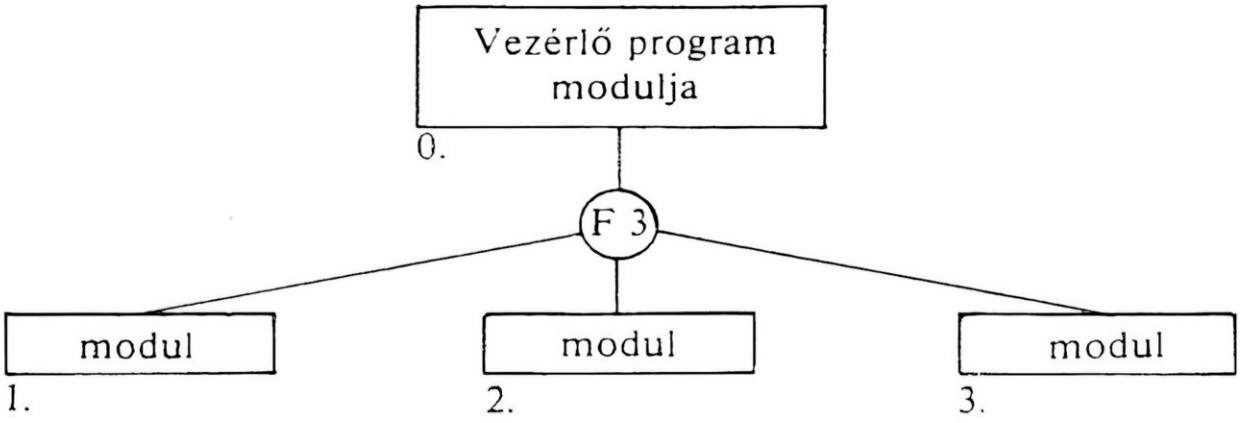
A soros strukturát a szerkezetben alacsonyabb szintű modulok egymásmellettiségének ábrázolásával fejezzük ki.



A választás vezérlőstruktúrát körbe írt F betűvel és sorszámmal, illetve a magyarázattal határozzuk meg.



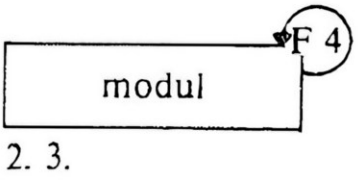
F2: Ha a feltétel igaz, akkor az 1. modul, ha a feltétel hamis, akkor a 2. modul kerül végrehajtásra.



F3: A vezérlőprogram által beállított számnak megfelő modul kerül végrehajtásra.

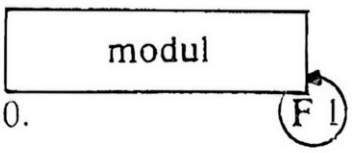
A ciklus vezérlőstruktúrákat a ciklust szimbolizáló jelbe írt F betűvel és sorszámmal jelöljük. A jelet a modult szimbolizáló négyszög jobb oldalán helyezzük el. Ha szükséges, a jelet magyarázattal egészíthetjük ki.

Modul előtt vizsgáló vezérlőstruktúra



F4: Vizsgálat, hogy a modul végrehajtásra kerül-e?

Modul után vizsgáló vezérlőstruktúra



F1: Vizsgálat, hogy a modul ismételten végrehajtásra kerül-e?

9. JÁTÉK

A strukturált program elemzését a 9. játék programján mutatjuk be.

```
10 REM "FUGGVENYEK HELYETTESITESI ERTEKEINEK SZAMITASA"
20 DIM X(20),Y(20)
30 PRINT"■":PRINT TAB(10);"* FUGGVENYEK *"
40 REM "BEVITEL"
50 PRINT"1- Y=X*X FUGGVENYT EGY HELYEN"
60 PRINT"2- Y1=2*X-3, Y2=-X+6, Y3=X↑3+X↑2+X F-EKET EGY HELYE
70 PRINT"3- Y=X↑2 FUGGVENYT TOBB HELYEN"
80 INPUT" A VALASZTOTT ESETSZAMA";A%
90 IF A%>3 OR A%<1 THEN PRINT"NINCS ILYEN ESET":GOTO80
100 ON A% GOTO 110,110,130
110 INPUT"IRD BE X ERTEKET";X
120 GOTO170
130 INPUT"IRD BE A HELYEK SZAMAT (2-20)";N
140 IF 20<N OR N<2 THEN PRINT TAB(25);"HIBA":GOTO130
150 PRINT:PRINT"IRD BE AZ X(I) ERTEKEKET"
160 FOR I=1 TO N:INPUT X(I):NEXT I
170 REM "SZAMITAS"
180 ON A% GOTO 190,200,230
190 LET Y=X*X: GOTO 240
200 LET Y1=2*X-3
210 LET Y2=-X+6
220 LET Y3=X↑3+X↑2+X: GOTO 240
230 FOR I=1 TO N: Y(I)=X(I)↑2: NEXT I
240 REM "KIIRAS"
250 ON A% GOTO 260,270,280
260 PRINT,"X=";X,"Y=";Y: GOTO 300
270 PRINT"X=";X," Y1"," Y2"," Y3":PRINT,Y1,Y2,Y3:GOTO 300
280 PRINT," X"," Y"
290 FOR I=1 TO N: PRINT,X(I),Y(I): NEXT I
300 PRINT:PRINT
310 INPUT"TOVABB (I/N)";E$
320 IF E$="I" THEN 30
330 END
```

A program pontosan ugyanazt a feladatot hajtja végre, mint a 8. játék programja, felépítésében azonban strukturált.

a programrészek

A program első ránézésre öt, egymástól jól elkülöníthető részből áll:

az első	10–30	programsor
a második a	40–160	programsor
a harmadik a	170–230	programsor
a negyedik a	240–290	programsor
az ötödik a	300–330	programsor

a program rendszere

Az első és az ötödik rész a program egészével összefüggő programsorokat tartalmaz. Ezek a programsorok együttesen alkotják a főprogram modult. A második, harmadik és negyedik részek programsorai az alprogramok moduljai.

A főprogram modulja amellett hogy tájékoztat bennünket a program feladatáról, kijelöli a tömbváltozók helyeit, kiíratja a program nevét, valamint vezérli a játék ismételhetőségét, egyben meghatározza az alprogramok végrehajtási sorrendjét, soros vezérlőstruktúrában.

Miről lehet a soros vezérlőstruktúrát felismerni?

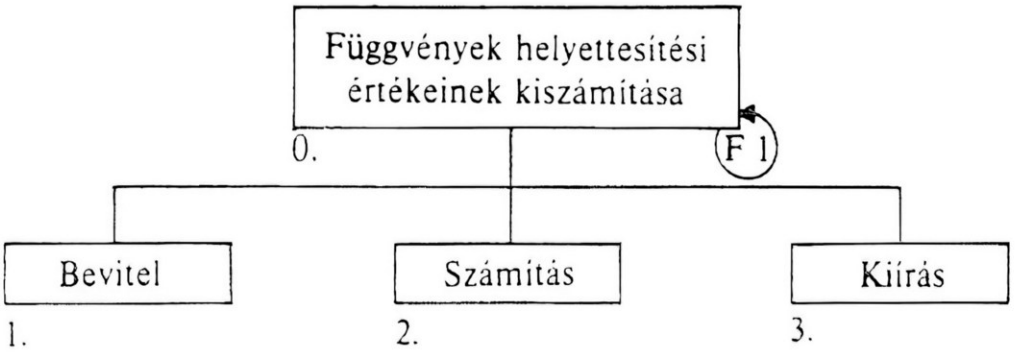
Arról, hogy az alprogramok a program végrehajtása szempontjából egymás után következnek. Az egyes alprogrammodulok szerepe is leolvasható a programból:

adatbevitel	(BEVITEL)
adatfeldolgozás	(SZÁMÍTÁS)
eredménykiírás	(KIÍRÁS)

Mindezek alapján már megrajzolhatjuk a program szerkezeti ábráját. (54. oldal)

Ezzel tisztáztuk a program egészének feladatát és felépítését. Azt viszont, hogy az egyes modulok hogyan hajtják végre feladataikat, töltik be szerepüket, csak a modulok részletes elemzésével ismerhetjük meg. A mo-

dulok elemzése azonban már egyenként, a többitől függetlenül elvégezhető. Ezzel biztosítva van a program lépésenkénti áttekinthetősége.



F1: A program futásának ismétlése.

Figyelembe véve, hogy a modulok önálló részek a programban, egy másik, de ugyanazt a szerepet betöltő modulra bármikor kicserélhetőnek kell lenniök, ezért leírásukra bizonyos szabályokat kellett bevezetni. A modulról olyan részletes leírást kell készíteni, amely a lehető legpontosabban meghatározza a modulban végrehajtandó műveleteket és mindazokat a kapcsolatokat, amelyekkel a program többi moduljaihoz csatlakozik. Írjuk le egyenként a modulokat.

0.modul főprogram

0. FÜGGVÉNYEK HELYETTESÍTÉSI ÉRTÉKEINEK SZÁMÍTÁSA

Szerepe:

1.Főprogram a BEVITEL, a SZÁMÍTÁS és a KIÍRÁS alprogramok soros struktúrában való vezérlése;
2.Műveleteket végez.

Műveletek:

1. A program feladata;
2. Tömbváltozóknak helykijelölés;
3. Képernyőtörlés, soremelés, a program nevének kiírása;

30. Kettő soremelés;

31. A program megismétlésére kérdezés;
32. A válasz alapján a programfutás vezérlése;
33. A program befejezése.

Változók:
X (I)-a független változó aktuális értékeinek helyei,
Y (I)-a függvények helyettesítési értékeinek helyei,
E$-„I" vagy „N"

Bemenő adat	*Kimenő adat*
E$	nincs

1.modul
alprogram

1. BEVITEL

Szerepe:
A független változók aktuális értékeinek bevitelével kapcsolatos műveletek végrehajtása.
Műveletek:
4. A modul neve;
5. Az első számítási változat kiírása;
6. A második számítási változat kiírása;
7. A harmadik számítási változat kiírása;
8. A felhívás kiírása a számítási változat közötti választásra és a választott változat számának bevitele;
9. A bevitt szám ellenőrzése, ha a bevitt szám nem 1, 2 vagy 3, akkor kiírja: „NINCS ILYEN ESET", és a vezérlést a 8. művelethez visszaadja;
10. A kiválasztott változatnak megfelelő programfutás megszervezése;
11. Az első és második számítási változathoz az X aktuális értékének beírása;
12. A programfutás ugratása a következő modul bemenetére;

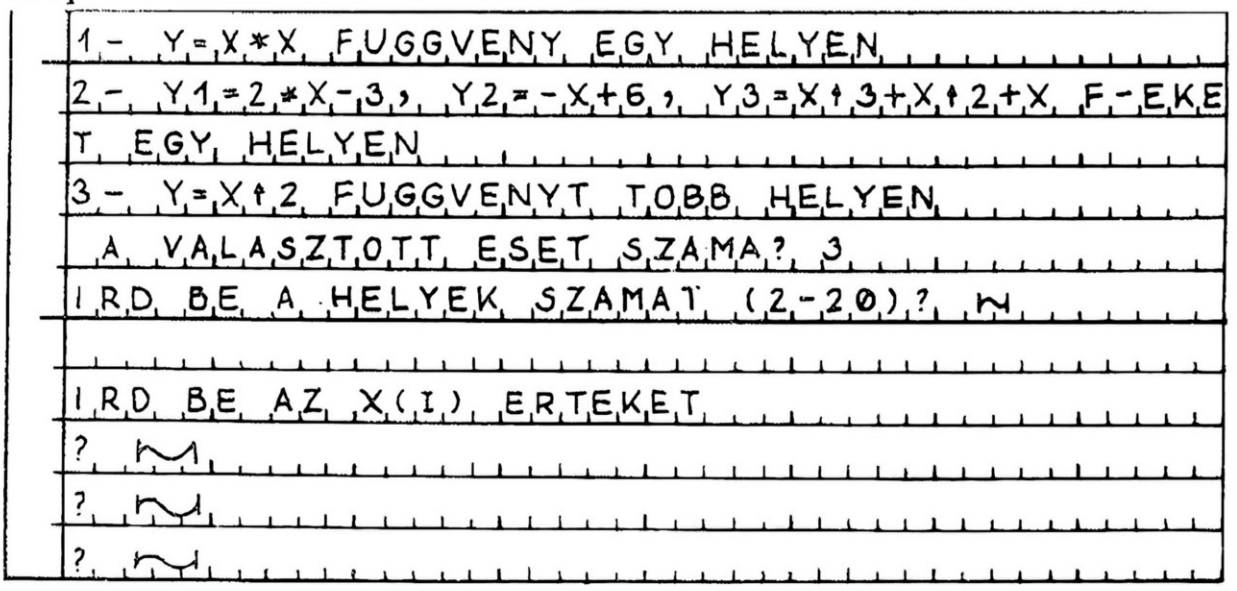
13. A harmadik számítási változathoz a számítási helyek számának bevitele. A számítási helyek 2–20 között lehetnek;
14. A számítási helyek beírásának ellenőrzése és hibás bevitelnél „HIBA" kiírása, a vezérlés vissza a 13. művelethez;
15. A X(I) aktuális értékeinek beírásához a felszólító szöveg „ÍRD BE AZ X(I) ÉRTÉKEKET" kiírása;
16. Ciklusszervezés az X(I) értékeinek beviteléhez. A ciklus I=1-től N-ig tart.

Változók:
X, X(I)–független változók aktuális értéke,
A%–a változat száma,
N–a helyek száma

Bemenő adat	*Kimenő adat*
A%, X(I), X, N	A%, X(I), X, N

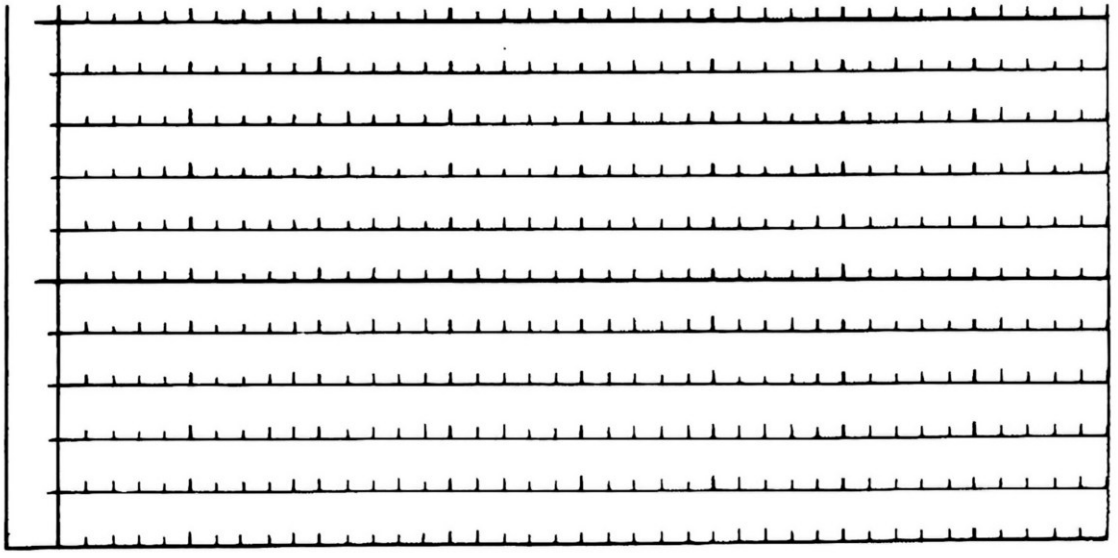
A modulhoz írásképek is csatolhatók.
Közöljük az 1. modul írásképét a harmadik változat kiválasztására. Ti pedig az első változatot dolgozzátok ki!

Y = X ↑ 2 FÜGGVÉNY TÖBB HELYEN változat írásképe



Azért nem írtuk be a sorok számát, mert a modulból nem határozható meg, hogy a kiírás melyik sorokra történik, azt az előző modulok behatárolják. Jelen esetben a főprogram.

Y=X*X FÜGGVÉNY EGY HELYEN változat írásképe



A második változat írásképe megegyezik az első változatéval.

2.modul alprogram

2. SZÁMÍTÁS

Szerepe:
A függvények helyettesítési értékeit kiszámító műveletek végrehajtása.
Műveletek:
17. A modul neve;
18. Az 1. modulban kiválasztott számítási változatnak megfelelően a programfutás megszervezése;
19. Az Y=X*X, ugrás a 3. modul bemenetére;
20. Y1 = 2*X–3
21. Y2 = –X+6
22. Y3 = X ↑ 3+X ↑ 2+X, ugrás a 3. modul bemenetére;
23. Ciklusszervezés az Y(I)=X(I) ↑ 2; számítására.

Változók:
A%, N, X, X(I),
Y, Y1, Y2, Y3, Y(I)–a függvények helyettesítési értékei

Bemenő adat	*Kimenő adat*
A%, N, X, X(I)	A%, N, Y, Y1, Y2, Y3, Y(I)

3. modul alprogram

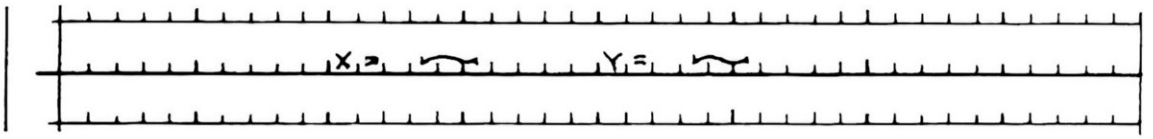
3. KIÍRÁS

Szerepe:
Képernyőre a számítási eredmények kiírását végző műveletek végrehajtása.
Műveletek:
24. A modul neve;
25. Az 1. modulban kiválasztott számítási változatnak megfelelő programfutás megszervezése;
26. Az első változat eredményeinek kiírása és ugrás vissza a főprogramba;
27. A második változat eredményének kiírása és ugrás vissza a főprogramba;
28. A harmadik változathoz a táblázat fejének kiírása;
29. A ciklus szervezése az eredménykiíráshoz.

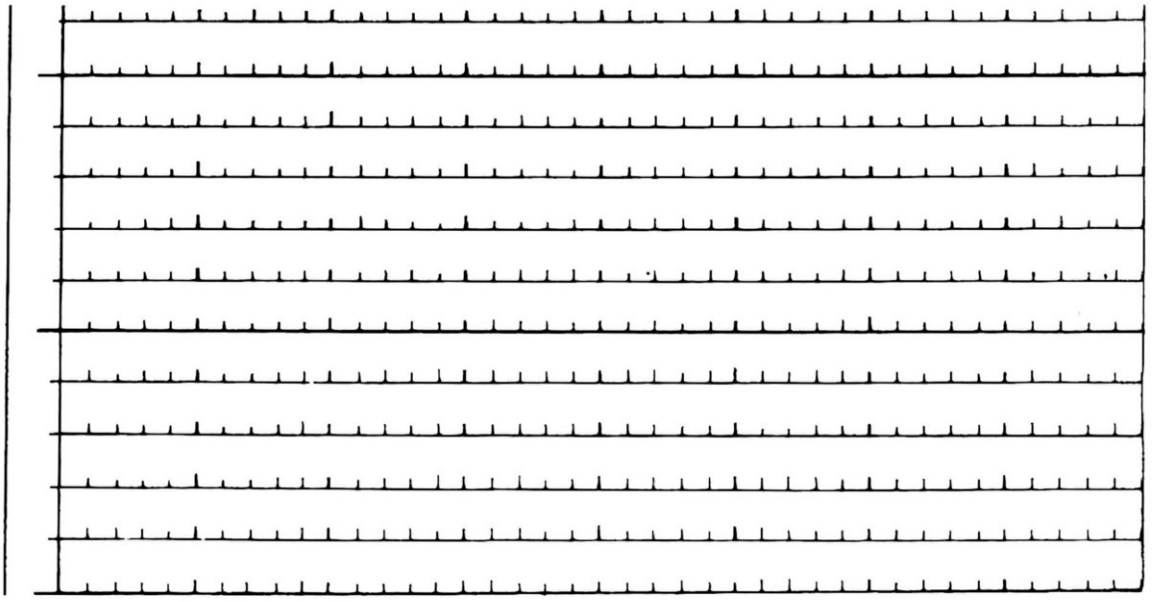
Változók
A%, N, X, X(I), X, Y1, Y2, Y3, Y(I)

Bemenő adat	*Kimenő adat*
A%, N, X, X(I), Y, Y1, Y2, Y3, Y(I)	X, X(I), Y, Y1, Y2, Y3, Y(I)

Az első változat írásképe



Irjátok le a második és a harmadik változat írásképét!



Ezzel befejeztük a 9. játék programjának leírását, a strukturált programozási módszer tervezési menete szerint.
A programokat kidolgozó szakemberek természetesen nem a programból, hanem a feladatból előzetesen elkészítik a program tervét, majd a program terve alapján modulonként kidolgozzák a programot. Így fogjátok ti is kidolgozni a programjaitokat, de most még nem tartunk itt. Először el kell sajátítanotok a strukturált programok olvasását, majd a módosítását.
Kezdjetek ismét játszani!

Egy kicsit azonban *változtassunk a játékszabályokon.* Eddig a program végén hangzottak el a kérdések. Ettől kezdve azt javasoljuk, hogy minden MAGYARÁZÓ (REM) utasítás kártyát csak akkor tehesse le a soron következő játékos, ha jellemezni tudja a program rendszerét vagy az előző modult, illetve annak részeit. A kér-

déseket az osztó tegye fel a JÁTÉKLAPON kidolgozott kérdésekből, majd ha többször játszottatok, a programtervből. A modul programsorait ilyen esetben azonban nem szükséges magyarázni, ezzel felgyorsul a játék.

10. JÁTÉK

Összevont program a százalékszámítás három esetére (a 2., a 3. és a 4. játék feladatainak összevonásával):

```
10 REM "SZAZALEKERTEK, SZAZALEKLAB, ALAP SZAMITAS"
20 PRINT"□":PRINT
30 PRINT"     *** SZAZALEKSZAMITAS ***
40 DATA" ALAP"," SZAZALEKLAB"," SZAZALEKERTEK"
50 DATA" ADOTT"," ES"," A"," AZ"
60 READ SA$,SP$,SE$,S1$,S2$,S3$,S4$
70 REM"SZAMITAS"
80 PRINT:PRINT"   SZAMITASI LEHETOSEGEK":PRINT
90 PRINT" 1- ";S1$;S4$;SA$;S2$;S3$;SP$
100 PRINT" 2- ";S1$;S4$;SA$;S2$;S3$;SE$
110 PRINT" 3- ";S1$;S3$;SP$;S2$;S3$;SE$:PRINT
120 INPUT" VALTOZAT SZAM";V
130 IF 0<V AND V<4 THEN160
140 PRINT" NINCS ILYEN VALTOZAT":GOTO120
150 PRINT"□":PRINT:PRINT
160 ON V GOTO 170,210,250
170 REM"SZAZALEKERTEK SZAMITAS"
180 INPUT"  ALAP,SZAZALEKLAB";A,P
190 LET E=A*P/100
200 GOTO 280
210 REM"SZAZALEKLAB SZAMITAS"
220 INPUT"  ALAP,SZAZALEKERTEK";A,E
230 LET P=E*100/A
240 GOTO 290
250 REM"ALAP SZAMITAS"
260 INPUT"  SZAZALEKLAB,SZAZALEKERTEK";P,E
270 LET A=INT(E*100/P)
280 REM"EREDMENY KIIRAS"
290 PRINT
300 PRINT TAB(5);SA$;"=          ";A
310 PRINT TAB(5);SP$;"=  ";P
320 PRINT TAB(5);SE$;"=";E:PRINT
330 INPUT"    TOVABB (I/N)";E$
340 IF E$="I" THEN 70
350 END
```

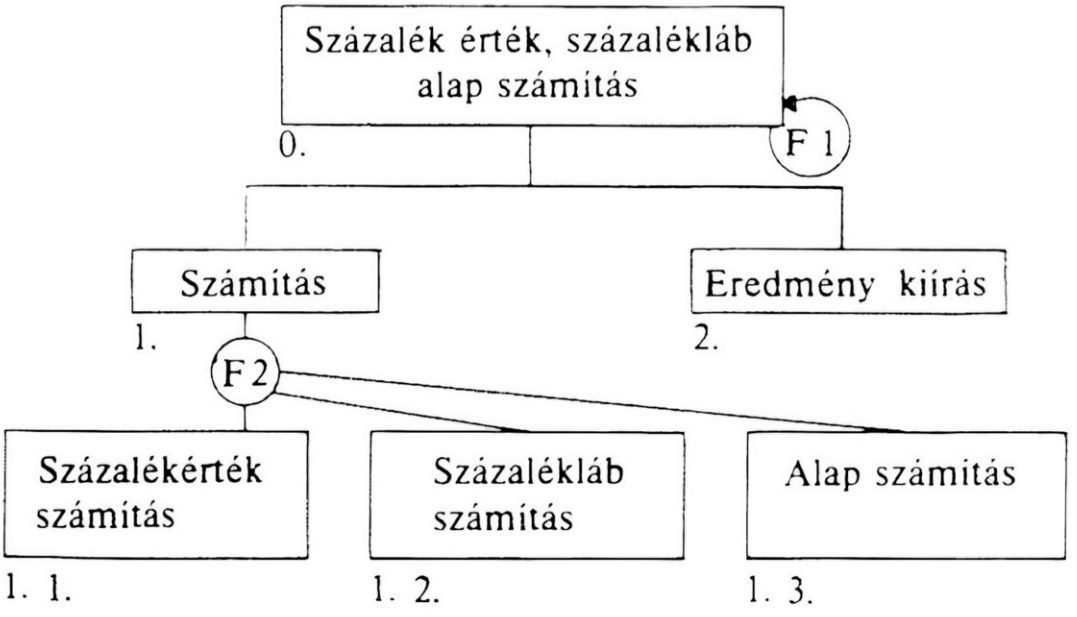
A tartalom megjelenítésére a jelen és a további öt játék esetében a strukturált programozásnak egy egyszerűsített – a 9. játéknál bemutatott – tervezési módszerét alkalmazzuk.

Ez a tervezési módszer a programoknak olyan leírását teszi lehetővé, amely egyrészt segítséget nyújt programolvasó készségetek ellenőrzéséhez, illetve hogy kérdéseket tehessen fel az osztó, másrészt a játék közbeni esetleges vitákat segítségével eldönthetitek.

a program rendszere

SZÁZALÉKSZÁMÍTÁS

Feladat:
A százalékérték, a százalékláb és az alap kiszámítása.
Szerkezete:



F1: A programismétlés vezérlése.
F2: Számítások vezérlése a kívánt változat szerint.

Bemenő adat:
Számítási változat
Alap
Százalékláb
Százalékérték
} közül mindig az ismert kettő

Kimenő adat:
Alap
Százalékláb
Százalékérték

0.modul

0. SZÁZALÉKÉRTÉK, SZÁZALÉKLÁB, ALAP SZÁMÍTÁSA

Szerepe:

1. Főprogram, 1. és 2. modul soros vezérlése;
2. Szöveg tárolása a kiíráshoz;
3. A programismétlés vezérlése.

Műveletek:

1. A program feladata;
2. Képernyőtörlés, két soremelés;
3. A program nevének kiírása;
4. Szövegek tárolása;
5. Szövegek tárolása;
6. Szövegek változókhoz való rendelése;

33. A program megismétlésére kérdezés és a felelet bevitele;
34. A program futásának vezérlése a felelet szerint;
35. Vége.

Változók:

SA$=„ALAP", SP$=„SZÁZALÉKLÁB", SE$=„SZÁZALÉKÉRTÉK", S1$=„ADOTT", S2$=„ÉS", S3$=„A", A4$=„AZ", E$=„I", vagy bármilyen betű

Bemenő adat:	*Kimenő adat:*
E$	SA$, SP$, SE$, S1$, S2$, S3$. S4$

1. modul

1. SZÁMÍTÁS

Szerepe:

A számítási változatok közötti választás vezérlése.

Műveletek:

7. A modul neve;
8. Szövegkiírás: SZÁMÍTÁSI LEHETŐSÉGEK;
9. Szövegkiírás: 1–ADOTT AZ ALAP ÉS A SZÁZALÉKLÁB;
10. Szövegkiírás: 2–ADOTT AZ ALAP ÉS A SZÁZALÉKÉRTÉK;
11. Szövegkiírás: 3–ADOTT A SZÁZALÉKLÁB ÉS A SZÁZALÉKÉRTÉK;
12. A változat számának bevitele;
13. A bevitt változat számának ellenőrzése;
14. Hibás szám esetén kiírás és a programfutás áthelyezése a 120. programsorra
15. Képernyőtörlés és három sor emelés;
16. A választott változatnak megfelelő alprogramnak a vezérlés átadása.

Változók:

V-a változat száma

Bemenő adat:	*Kimenő adat:*
V	nincs

1.1. SZÁZALÉKÉRTÉK-SZÁMÍTÁS

Szerepe:

1. A százalékérték-számítás;
2. Vezérlés átadása a 2. modulnak.

Műveletek:

17. A modul neve;
18. Az alap és a százalékláb bevitele;
19. A százalékérték kiszámítása;
20. Ugrás a 2. modulra.

Változók:
A-alap, P-százalékláb, E-százalékérték

Bemenő adat:
A, P

Kimenő adat:
A, P, E

1.2.modul

1.2. SZÁZALÉKLÁB-SZÁMÍTÁS

Szerepe:
1. A százalékláb-számítás;
2. A vezérlés átadása a 2. modulnak.

Műveletek:
21. A modul neve;
22. Az alap és a százalékérték bevitele;
23. A százalékláb kiszámítása;
24. Ugrás a 2. modulra.

Változók:
A-alap, P-százalékláb, E-százalékérték

Bemenő adat:
A, E

Kimenő adat:
A, P, E

1.3.modul

1.3. ALAPSZÁMÍTÁS

Szerepe:
1. Az alap számítása;
2. A vezérlés átadása a 2. modulnak.

Műveletek:
25. A modul neve;
26. A százalékláb és a százalékérték bevitele;
27. Az alap kiszámítása, a számított érték egész részének hozzárendelése az A változóhoz.

Változók:
A-alap, P-százalékláb, E-százalékérték

Bemenő adat:
P, E

Kimenő adat:
A, P, E

2. EREDMÉNY KIÍRÁSA

Szerepe:
1. Az eredmény kiírása;
2. A vezérlés visszaadása a főprogramnak.

Műveletek:
28. A modul neve;
29. Soremelés;
30. Az alap kiírása;
31. A százalékláb kiírása;
32. A százalékérték kiírása.

Változók:
A-alap, P-százalékláb, E-százalékérték

Bemenő adat:
A, P, E

Kimenő adat:
Alap
Százalékláb
Százalékérték

11. JÁTÉK

Számsorozattal megadott osztályzatok átlagát számító program:

```
10 REM" OSZTALYZATOK ATLAGANAK SZAMITASA"
20 PRINT"␣":PRINT:PRINT
30 PRINTTAB(5)" ** SZOVEGMUVELETEK 1 **":PRINT
40 REM"BEIRAS"
50 INPUT" OSZTALYZAT SZAMSOROZATTAL";A
60 IF A<0 OR A>5555555 THENPRINT"CSAK HET TAGOT":GOTO50
70 REM"SZAMITAS"
80 LET A1$=STR$(A)
90 LET B1=LEN(A1$)
100 LET D1=0
110 FOR V=1 TO B1
120 LET C1$=MID$(A1$,V,1)
130 LET C1=VAL(C1$)
140 LET D1=D1+C1
150 NEXT V
160 LET E=D1/(B1-1)
170 IF E<1.51 THEN A=1:GOTO 220
180 IF E<2.51 THEN A=2:GOTO 220
190 IF E<3.51 THEN A=3:GOTO 220
200 IF E<4.51 THEN A=4:GOTO 220
210 LET A=5
220 REM"EREDMENY KIIRAS"
230 PRINT:PRINT" OSZTALYZAT ATLAG";A
240 PRINT
250 INPUT" TOVABB";E$:IF E$="I"THEN20
260 END
```

a program rendszere

SZÖVEGMŰVELETEK I.

Feladata:
A számsorozattal megadott osztályzatok (maximálisan hét elemből álló) átlagának kiszámítása.
Szerkezete:

F1: A programismétlés vezérlése

Bemenő adat:	*Kimenő adat:*
Osztályzatok	Az osztályzatok átlaga

A program rendszere alapján tisztázható a feladat végrehajtásának módja: a számsorozatot szöveggé kell átalakítani, majd elemekre bontani, és az így kapott elemeken az átlagszámítást elvégezni.

0. modul

0. OSZTÁLYZATOK ÁTLAGÁNAK SZÁMÍTÁSA

Szerepe:
1. Főprogram, az 1., 2., 3. modul soros vezérlése;
2. A programismétlés vezérlése.

Műveletek:
1. A program feladata;
2. Képernyőtörlés, három sor emelés;
3. A program nevének kiírása;
24. Soremelés;
25. A programismétlés szervezése;
26. Vége.

Változók:
E$="I" vagy bármely más betű

Bemenő adat:	*Kimenő adat:*
E$	nincs

1.modul

1. BEÍRÁS

Szerepe:
A számsorozat bevitelének szervezése.
Műveletek:
4. A modul neve;
5. Osztályzatok beírása;
6. A beírás ellenőrzése.

Változók:
A-számsorozat

Bemenő adat:
A

Kimenő adat:
A

2.modul

2. SZÁMÍTÁS

Szerepe:
1. A számsorozat elemekre bontása;
2. Az osztályzatátlag kiszámítása.
Műveletek:
7. A modul neve;
8. A számsorozat átalakítása szöveggé az STR$ (A) függvénnyel és hozzárendelés A1$ változóhoz;
9. A szöveg elemeinek megszámlálása LEN(A1$) függvénnyel, szükséges az összegező ciklus beállításához;
10. Az osztályzatösszeg változó beállítása 0-ra;
11. Ciklus vezérlőstruktúra megszervezése;
12. Elemekre bontás a MID$(A1$, V, 1) függvénnyel.

Megjegyzés:
A függvény úgy működik, hogy az A1$ szövegből, a V ciklusváltozó értékének megfelelő elemet kiválasztja. Ha a V értéke eggyel nő, akkor balról jobbra haladva minden egyes osztályzat kiválasztásra kerül.
13. Visszaalakítás számmá a VAL(C1$) függvénnyel;
14. Az osztályzatok összeadása;
15. Ciklusszámlálás;
16. Az osztályzat átlagának számítása. Elegendő, ha a számított átlag század pontosságú, ezért el kell végezni a kerekítést;
17. Ha a szám kisebb 1,51-nél, akkor 1-nek feleljen meg, és ugrás a 3. modulra;
18. Ha a szám kisebb 2,51-nél, akkor 2-nek feleljen meg, és ugrás a 3. modulra;
19. Ha a szám kisebb 3,51-nél, akkor 3-nak feleljen meg, és ugrás a 3. modulra;
20. Ha a szám kisebb 4.51-nél, akkor 4-nek feleljen meg, és ugrás a 3. modulra;
21. A szám feleljen meg 5-nek.

Változók:
A–számsorozat, A1$ – a számsorozat mint szöveg, B1 – a szöveg elemeinek száma, D1 – az osztályzatok összege, C1$ – a szöveg 1 elemének értéke, C1 – egy osztályzat, V – ciklusváltozó, E – munkaváltozó, A – átlagosztályzat

Bemenő adat:
A-számsorozat

Kimenő adat:
A-átlagosztályzat

3. EREDMÉNYKIÍRÁS

Szerepe:
Az osztályzatok átlagának kiírása.
Műveletek:
22. A modul neve;
23. Az osztályzatátlag kiírása.

Változók:
A-átlagosztályzat

Bemenő adat:	*Kimenő adat:*
A	Osztályzatok átlaga

A 11. játék programjának leírásában tapasztalhattátok, hogy a programsorok többségét a korábbi játékokban már megtanultátok elolvasni, ugyanakkor a SZÁMÍTÁS modul új feladat elé állított benneteket. Az ilyen feladatok a kidolgozóktól már megkövetelik, hogy az adott program műveleteit lépésről lépésre megtervezzék, és az egyes lépéseket folyamatukban leírják, vagyis a program működését előzetesen végiggondolják.
A programok tervezésénél a szakemberek rendszerint nem írják le a modulok minden egyes programsorának tartalmát. Elegendőnek tartják azoknak a főbb műveleteknek a meghatározását, amelyek az adott modul szerepét megvalósítják. A következő játékoknál már mi is így írjuk le a programok tartalmát. Javasoljuk, hogy ti pedig képzeljétek magatokat a kidolgozó szerepébe.

12. JÁTÉK

Készíts programot tízes számrendszerből más számrendszerekbe történő átszámításra (16-os számrendszerig).

feladattisztázás

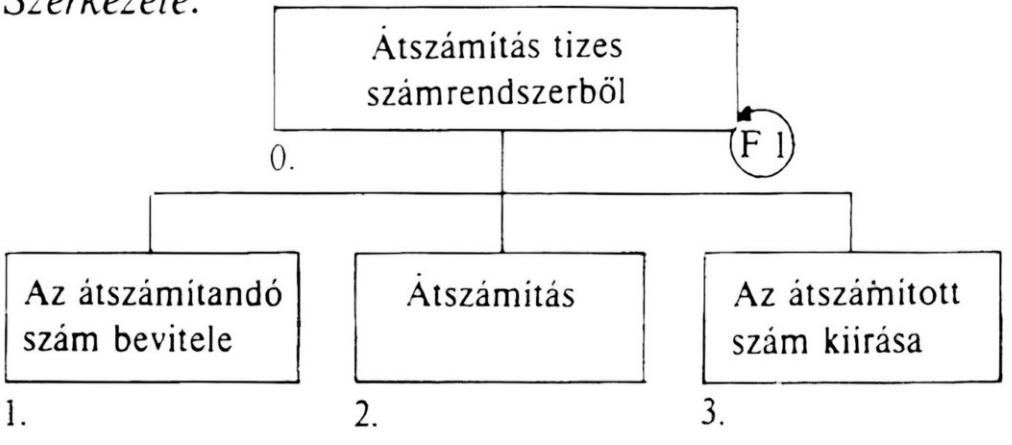
Tízes számrendszerből bármely számrendszerbe való átszámítás folyamata:

1. Kiszámítjuk az átszámítandó szám és az alapszám hányadosát, és vesszük annak egész részét;
2. Ezt a számot megszorozzuk az alapszámmal, és kivonjuk az átszámítandó számból;
3. Az így kapott szám az új számrendszerben az egyes helyértékű, majd az ismétlésnél a következő helyértékű szám;
4. Megvizsgáljuk, hogy az átszámítandó szám és az alapszám hányadosának egész része egyenlő-e 0-val, ha nem, akkor e számot átszámítandó számként vesszük figyelembe, és a műveleteket az 1. lépéstől megismételjük;
5. Az 1–4 lépést mindaddig folytatjuk, amíg a vizsgált hányados értéke 0 nem lesz.

a program rendszere

ÁTSZÁMÍTÁS

Feladata:
Számoknak tízes számrendszerből 2–16-ig számrendrendszerekbe való átszámításának végrehajtása.
Szerkezete:



F1: A programismétlés vezérlése.

Bemenő adat:	*Kimenő adat:*
Az átszámítandó szám	Az átszámítandó szám
Az alapszám	Az alapszám
	Az átszámított szám

0.modul

0. ÁTSZÁMÍTÁS TÍZES SZÁMRENDSZERBŐL

Szerepe:

1. Fő program, az 1., 2. és 3. modul soros struktúrában való vezérlése;
2. A számításhoz szükséges alapadatok előkészítése;
3. A programismétlés vezérlése.

Műveletek:

1. A program feladata;
2. Helykijelölés 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, Á, B, C, D, E, F számjegyek tárolása;
3. Képernyő-előkészítés és a program nevének kiírása;
4. A programismétlés szervezése.

Változók:

S$(0–15)-számjegyek, E$=„I" vagy más betű

Bemenő adat:	*Kimenő adat:*
E$	S$(0–15)

1.modul

1. AZ ÁTSZÁMÍTANDÓ SZÁM BEVITELE

Szerepe:

Az átszámítandó szám és az alapszám bevitele.

Műveletek:
1. Az átszámítandó szám bevitele, nem lehet kisebb 0-nál és nagyobb 999999-nél;
2. Az új számrendszer alapszámának bevitele és ellenőrzése.

Változók:
NI–átszámítandó szám, NA–alapszám

Bemenő adat:
NI, NA

Kimenő adat:
NI, NA

2. ÁTSZÁMÍTÁS

Szerepe:
Az átszámítás végrehajtása.
Műveletek:
1. Az NI változó értékét N változó vegye fel, a NI változó értéke maradjon meg a kinyomtatáshoz;
2. Egy szövegváltozónak adjunk üres helyértéket;
3. A feladattisztázásnál ismertetett lépések végrehajtásának megszervezése.

Változók:
N–munkaváltozó, IH–munkaváltozó, IM–munkaváltozó, NA–alapszám, NI–átszámítandó szám, S$(0–15)–számjegyek, SE$–átszámított szám

Bemenő adat:
NI, NA, S$(0–15)

Kimenő adat:
SE$

3. AZ ÁTSZÁMÍTOTT SZÁM KIÍRÁSA

Szerepe:
Az átszámítás eredményének kiírása.
Műveletek:
1. Az átszámítandó szám kiírása;
2. Az alapszám kiírása;
3. Az átszámított szám kiírása.

Változók:
NI–átszámítandó szám, NA–alapszám,
SE$–átszámított szám

Bemenő adat:	*Kimenő adat:*
NI, NA, SE$	Átszámítandó szám Alapszám Átszámított szám

```
10 REM"ATSZAMITAS TIZES SZAMRENDSZERBOL"
20 REM"2-TOL 16-OSIG SZAMRENDSZEREKBE"
30 DIM S$(15)
40 DATA 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
50 FOR I=0 TO 15
60 READ S$(I)
70 NEXT I
80 PRINT"␣":PRINT TAB(5);"**ATSZAMITAS **"
90 PRINT:PRINT
100 REM"AZ ATSZAMITANDO SZAM BEVITELE"
110 INPUT" ATSZAMITANDO SZAM 10 SZR-BEN";NI
120 IF 0=<NI AND NI<1000000 THEN 150
130 PRINT"NEM NEGATIV ES NEM T0BB HATJEGYUNEL"
140 GOTO  90
150 INPUT" ALAPSZAM 2-16 SZR-BEN";NA
160 IF 1<NA AND NA<17 THEN 190
170 PRINT:PRINT"NEM MEGENGEDETT BEVITEL":PRINT
180 GOTO 150
190 REM"ATSZAMITAS"
200 LET N=NI
210 LET SE$=" "
```

```
220 LET IH=INT(N/NA)
230 LET IM=N-NA*IH
240 LET SE$=S$(IM)+SE$
250 LET N=IH
260 IF N<>0 THEN 220
270 REM"AZ ATSZAMITOTT SZAM KIIRASA"
280 PRINT:PRINT" AZ ATSZAMITANDO SZAM";NI
290 PRINT"                ALAPSZAM";NA
300 PRINT"AZ ATSZAMITOTT SZAM ";SE$:PRINT
310 INPUT" TOVABB (I/N)";E$
320 IF E$="I" THEN 80
330 END
```

13. JÁTÉK

Készíts programot tetszőleges szövegben előforduló betűk gyakoriság vizsgálatára!

feladat-tisztázás

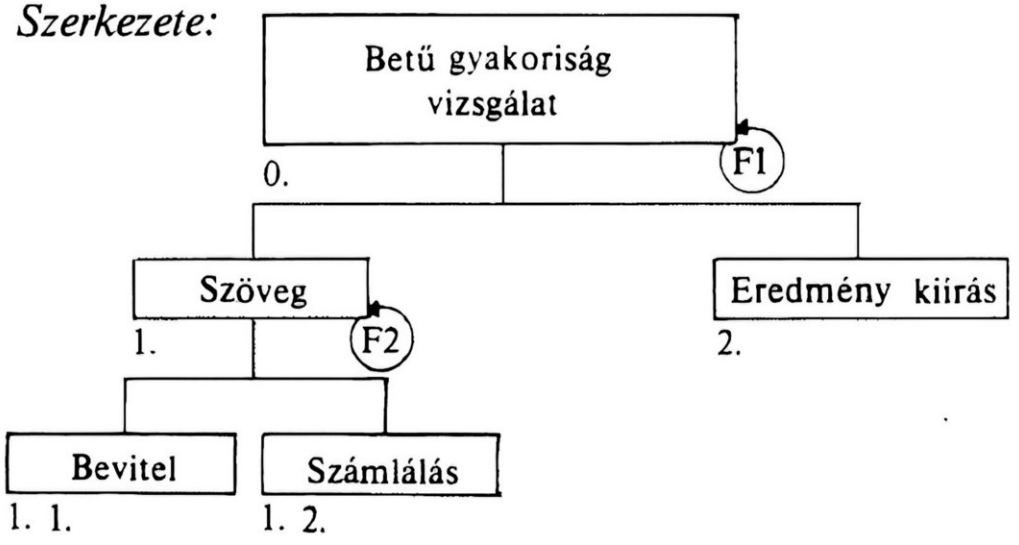
A szöveg vizsgálatához az ASCII. kódokat használjuk fel.

A betűk kódjai az A betűvel kezdődően 65 és a Z betűvel befejezve 90. A 65–90 között minden betűnek van tehát egy kódszáma. (A Commodore típusú számítógépeken.) A kódot az ASC szövegfüggvény állítja elő. Szöveg elemenkénti vizsgálata MID$ függvénnyel valósítható meg, azáltal hogy az ASC függvénybe foglalva, skatulyázva, argumentumaként MID$ függvényt helyezünk el.

a program rendszere

SZÖVEGMŰVELETEK 2

Feladata:
A betűk előfordulásának számlálása.
Szerkezete:



F1: Programismétlés vezérlése.
F2: Szövegismétlés vezérlése.

Bemenő adat:
Szövegek

Kimenő adat:
Karakterek jelek száma
Betűk száma
Betűk gyakorisága

0. modul

0. BETŰGYAKORISÁG-VIZSGÁLAT

Szerepe:
1. Főprogram, az 1. és 2. modul soros struktúrában való vezérlése;
2. A betűszámlálás előkészítése;
3. A programismétlés vezérlése.

Műveletek:
1. A betűszámláláshoz helyek kijelölése;
2. A képmező előkészítése;
3. A betűket számláló tömbváltozó feltöltése 0-val (erre azért van szükség, mert a program megismétlése esetén a számlálást újra kell kezdeni);
4. A program ismétlésének szervezése.

Változók:
K(0–26)–betűket számláló tömbváltozó,
L–karakterszámláló, E$–„I" vagy más betű

Bemenő adat:
E$

Kimenő adat:
K(0–26), L

1. SZÖVEG

Szerepe:
1. Az 1.1. és 1.2. modul soros struktúrában való vezérlése;
2. Ciklusszervezés a szöveg vége vizsgálat alapján.

Műveletek:
1. Szöveg vége vizsgálat /* jelre, minden más szöveg esetén ciklusszervezés.

Változók:
S$–szöveg

Bemenő adat:
S$

Kimenő adat:
nincs

1.1. BEVITEL

Szerepe:
Szövegbevitel.

Műveletek:
1. Szövegbevitel szervezése.

Változók:
S$–szöveg

Bemenő adat:
S$

Kimenő adat:
S$

1.2. SZÁMLÁLÁS

Szerepe:

1. Karakter számlálás;
2. Betűk számlálása;
3. Betűgyakoriság számlálás.

Műveletek:

1. A szövegkarakterek számának meghatározása;
2. Karakterszámlálás;
3. Ciklusszervezés a szöveg karakterszámának megfelelően;
4. A szövegelemek ASCII. kódjainak megállapítása és a betűk hozzárendelése a számláló tömbváltozó megfelelő eleméhez. (A szöveg egy-egy elemének kódját úgy állítjuk elő, hogy kiválasztjuk a MID$ függvénnyel a ciklus változó szerinti elemét, és az ASC függvénnyel meghatározzuk kódját. Ha ebből a kódértékből kivonunk 64-et, akkor 1–26-ig sorszámmal a betűket hozzárendelhetjük a számláló tömbváltozó azonos indexű elemeihez. Ha a függvény más számot állít elő, akkor az nem betű);
5. Az összes betű számlálása.

Változók:

IV–szöveg elemszáma, L–összes karakter, I–ciklusváltozó, J–munkaváltozó, K(1–26)–betű előfordulások, K(0)–összes betű, S$–szöveg

Bemenő adat:	*Kimenő adat:*
S$	L, K(0), K(1–26)

2. EREDMÉNYKIÍRÁS

Szerepe:
Az eredmény kiírása.
Műveletek:
1. Az összes karakter és betű kiírása;
2. A betűk előfordulásának kiírása két oszlopban, az ASCII. kódok sorrendjében, CHR$ függvénnyel.

Változók:
L–összes karakter K(1–26) betű előfordulások,
K(0)–összes betű

Bemenő adat:
L, K(0), K(1–26)

Kimenő adat:
Összes karakter
Összes betű
Betűként az előfordulások száma

```
10 REM"BETU GYAKORISAG VIZSGALAT"
20 DIM K(26)
30 PRINT"␣"
40 PRINT"      ** SZOVEGMUVELETEK 2 **":PRINT
50 PRINT"A SZOVEG BEIRAS BEFEJEZESE /* JELLEL"
60 LET L=0
70 FOR I=0 TO 26:K(I)=0:NEXT I
80 REM"SZOVEG"
90 REM"BEVITEL"
100 INPUT" SZOVEG";S$
110 REM"SZAMLALAS"
120 LET IV=LEN(S$)
130 LET L=L+IV
140 FOR I=1 TO IV
150 LET J=ASC(MID$(S$,I,1))-64
160 IF 0<J AND J<27 THEN K(J)=K(J)+1:K(0)=K(0)+1
```

```
170 NEXT I
180 IF S$<>"/*" THEN 90
190 REM"EREDEMENY KIIRAS"
200 PRINT"␣":PRINT" OSSZES KARAKTER";L;" OSSZES BETU";K(0)
210 PRINT:PRINT" A BETUK ELOSZLASA"
220 FOR I=1 TO 26 STEP 2
230 PRINT TAB(2);CHR$(I+64);K(I);TAB(12);CHR$(I+65);K(I+1)
240 NEXT I
250 INPUT" TOVABB";E$:IF E$="I" THEN 30
260 END
```

14. JÁTÉK

Készíts programot a gyufajátékra. Az egyik játékos a számítógép legyen!

a játék leírása

1. Az *A* játékos tetszés szerinti gyufaszálat helyez az asztalra;
2. A *B* játékos húz először 1–3 szálat, majd az *A* játékos is elvesz 1–3 között a gyufákból;
3. A játékosok mindaddig folytatják a 2. lépést, amíg van az asztalon gyufa. Az a játékos veszített, aki az utolsó szálat vette el.

feladat-tisztázás

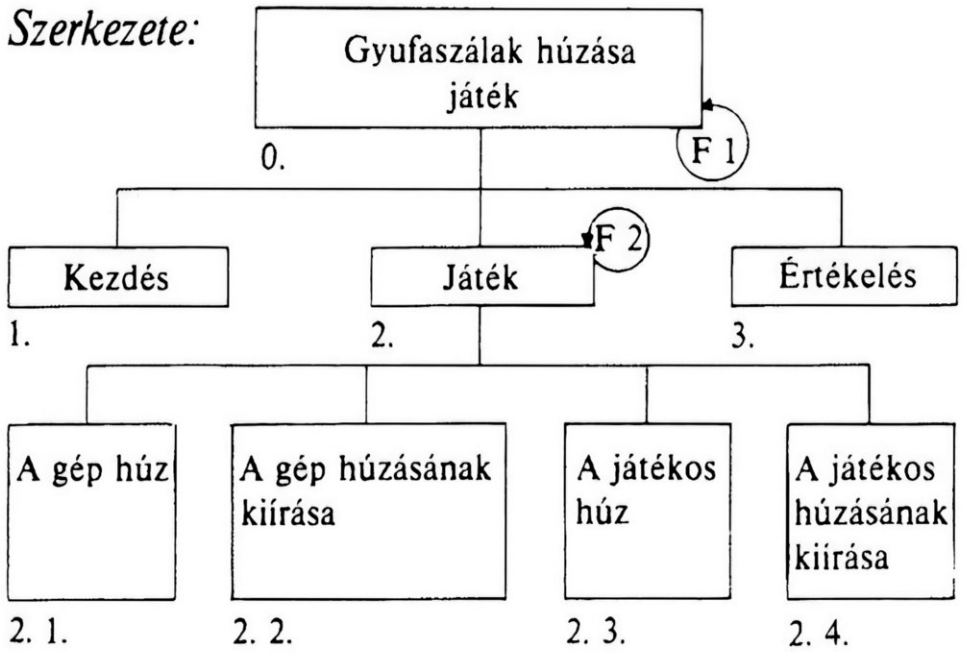
1. Kétjegyű szám bevitele;
2. A számítógép véletlenszámmal előállítja a húzását;
3. A húzás és a maradék kiírása;
4. A játékos húzásának bevitele;
5. A játékos húzásának és a maradéknak a kiírása;
6. A 3–5 lépés ismétlése, míg a szám értéke nulla, vagy negatív nem lesz;
7. A játék értékelése.

GYUFAJÁTÉK

Feladat:
A gyufajátékban az A játékos megszemélyesítése.

a program rendszere

Szerkezete:



F1: a játék ismétlése
F2: a húzások ismétlése

0.modul

0. GYUFAJÁTÉK

Szerepe:
1. Főprogram, az 1., 2. és 3. modul soros struktúrában való vezérlése;
2. A programismétlés szervezése.

Műveletek:
1. Képernyő előkészítés;
2. Programismétlés.

Változók:
B$–„I" vagy más betű

Bemenő adat:
B$

Kimenő adat:
nincs

1.modul

1. KEZDÉS

Szerepe:
1. A játék szabályainak kiírása;
2. A kezdő szám beírása.

Műveletek:
1. A játékszabályok kiírása, szöveg:
   „ÍRJ KÉTJEGYŰ SZÁMOT. ENNYI GYUFÁVAL JÁTSZUNK. FELVÁLTVA 1–3 SZÁLAT VEHETÜNK EL. AKI AZ UTOLSÓ SZÁLAT HÚZTA, AZ VESZÍTETT.";
2. A kezdő szám bevitele, ellenőrzéssel.

Változók:
X–kezdő szám

Bemenő adat:	*Kimenő adat:*
X	X

2.modul

2. JÁTÉK

Szerepe:
A húzások ismétlésének szervezése.

Műveletek:
1. A kezdő szám elemzése, ha értéke nulla, a vezérlés átadása a 3. modulnak,
2. A húzások ismétlésének vezérlése.

Változók:
X– a gyufaszálak száma

Bemenő adat:	*Kimenő adat:*
X	Nincs

2.1.modul

2.1. A GÉP HÚZ

Szerepe:
A gép húzásának előállítása.
Műveletek:
1. A véletlen szám előállítása;
2. Az előállított szám beállítása 1–3 érték közé.

Változók:
S–véletlen szám, L–húzott gyufa száma

Bemenő adat:	*Kimenő adat:*
Nincs	L

2.2.modul

2.2. A GÉP HÚZÁSÁNAK KIÍRÁSA

Szerepe:
A gép húzásának és a maradék gyufa számának kiírása.
Műveletek:
1. A gép húzásának kivonása a gyufák számából;
2. AS változó beállítása, a gép húzásának jelölése. Erre azért van szükség, hogy a játék értékelésénél megállapíthassa a gép az utolsó húzót;
3. Vizsgálat annak érdekében, hogy a gépi húzás eredményeként a gyufák száma nem kevesebb mint nulla. Ez akkor következhet be, ha az utolsó húzásnál már kevesebb a gyufaszálak száma, mint

a véletlen számmal előállított húzás értéke. Ha a szám negatív, akkor a húzott számot korrigálni kell úgy, hogy a gyufák száma nulla legyen;

4. A húzott szám és a maradék gyufa számának kiírása, szöveg: „SZÁLAT HÚZTAM. MARADT".
5. Ha gép húzta az utolsó szálat, a vezérlés átadása a 3. modulnak.

Változók:
L–húzott gyufa szám, X–gyufák száma, A$–az utolsó húzó megjelölése, A$ =„G"

Bemenő adatok:	*Kimenő adatok:*
L, X	X, A$

2.3.modul

2.3. A JÁTÉKOS HÚZ

Szerepe:
A játékos húzásának bevitele.
Műveletek:
1. Bevitel ellenőrzéssel.

Változók:
K–a játékos húzott száma

Bemenő adat:	*Kimenő adat:*
K	K

2.4. A JÁTÉKOS HÚZÁSÁNAK KIÍRÁSA

Szerepe:
1. A maradék szál kiírása;
2. Ellenőrzés az utolsó húzásra.

Műveletek:
1. A játékos húzásának kivonása a gyufák számából;
2. A$ változó beállítása a játékos húzásának jelölésére;
3. Vizsgálat, hogy a játékos húzott-e több szálat, mint amit lehetett. Ha igen, akkor szövegkiírás: „NINCS MÁR ENNYI SZÁL", és a szám beállítása;
4. A maradék szál kiírása, szöveg: „MARADT".

Változók:
K–játékos húzása, X–gyufák száma, A$–az utolsó húzó megjelölése. A$=„J"

Bemenő adat:	*Kimenő adat:*
K, X	X, A$

3. ÉRTÉKELÉS

Szerepe:
A játék végeredményének értékelése.

Műveletek:
1. A$ változó, ha „G" akkor a gép veszített, szöveg: „VESZTETTEM";
2. A játékos veszítésének kiírása, szöveg: „AZ UTOLSÓ SZÁLAT VETTED EL. VESZTETTÉL."

Változók:
A$

Bemenő adat:	*Kimenő adat:*
A$	A vesztes kiírása

```
10 REM"GYUFASZALAK HUZASA JATEK
20 PRINT"□":PRINT
30 PRINT "          ***  GYUFAJATEK  ***":PRINT
40 REM"KEZDES"
50 PRINT "IRJ KETJEGYU SZAMOT. ENNYI GYUFAVAL JATSZUNK. ";
60 PRINT"FELVALTVA 1-3 SZALAT VEHETUNK EL. ";
70 PRINT"AKI AZ UTOLSO SZALAT HUZZA AZ VESZITETT.":PRINT
80 INPUT" SZAM";X:IFX<10 OR X>99THENPRINT"KETJEGYUT!":GOTO80
90 PRINT"□"
100 REM"JATEK"
110 IF X=0 THEN 300
120 REM"A GEP HUZ"
130 LET S=RND(X)
140 IF S<.33 THEN L=1: GOTO 170
150 IF S<.67 THEN L=2: GOTO 170
160 LET L=3
170 REM"A GEP HUZASANAK KIIRASA"
180 LET X=X-L: A$="G"
190 IF X<0 THEN L=L+X: X=0
200 PRINT L;"SZALAT HUZTAM.   MARADT"X: IF X=0 THEN 280
210 REM"A JATEKOS HUZ"
220 INPUT" A HUZASOD";K: IF K<1 OR K>3 THEN220
230 REM"A JATEKOS HUZASANAK KIIRASA"
240 LET X=X-K: A$="J"
250 IF X<0 THEN PRINT"     NINCS MAR ENNYI SZAL": X=0
260 PRINT"                 MARADT"X
270 GOTO 100
280 REM" ERTEKELES"
290 IF A$="G" THEN PRINT"VESZTETTEM":PRINT:GOTO 310
300 PRINT"AZ UTOLSO SZALAT VETTED EL. VESZTETTEL.":PRINT
310 INPUT"MEGISMETELJUK(I/N)";B$: IF B$="I" THEN 20
320 END
```

15. JÁTÉK

Készíts olyan programot, amelyben a számítógépnek a gyufajátékban stratégiája van!

feladat-tisztázás

A gyufajátékot, ha kellő figyelemmel játszottátok, akkor ismeritek már a nyerő játékos stratégiáját. A játékos

akkor nyeri meg a játékot, ha arra törekszik, hogy a húzása után a gyufaszálak száma a 4 többszöröse plusz 1 legyen, majd minden következő húzásnál 4-re egészítse ki a játékostársa húzását.
Erre a stratégiára kell a számítógépet felkészíteni. A stratégia érvényesítésének 4 változata lehetséges. A gyufaszálak számának megválasztásával a 4 többszöröséhez:
1. egy szálat,
2. kettő szálat,
3. három szálat, vagy
4. nulla szálat adunk hozzá, és minden változatban 4-re egészítjük ki a játékostárs húzását.

El kell dönteni, hogy melyik változatot választjuk a számítógép stratégiájának. Nyerjen a számítógép a 2., a 3. és a 4. változatban, a játékostársnak csak az 1. változatban legyen nyerési esélye.

új modul beillesztése a 14. játék programjába

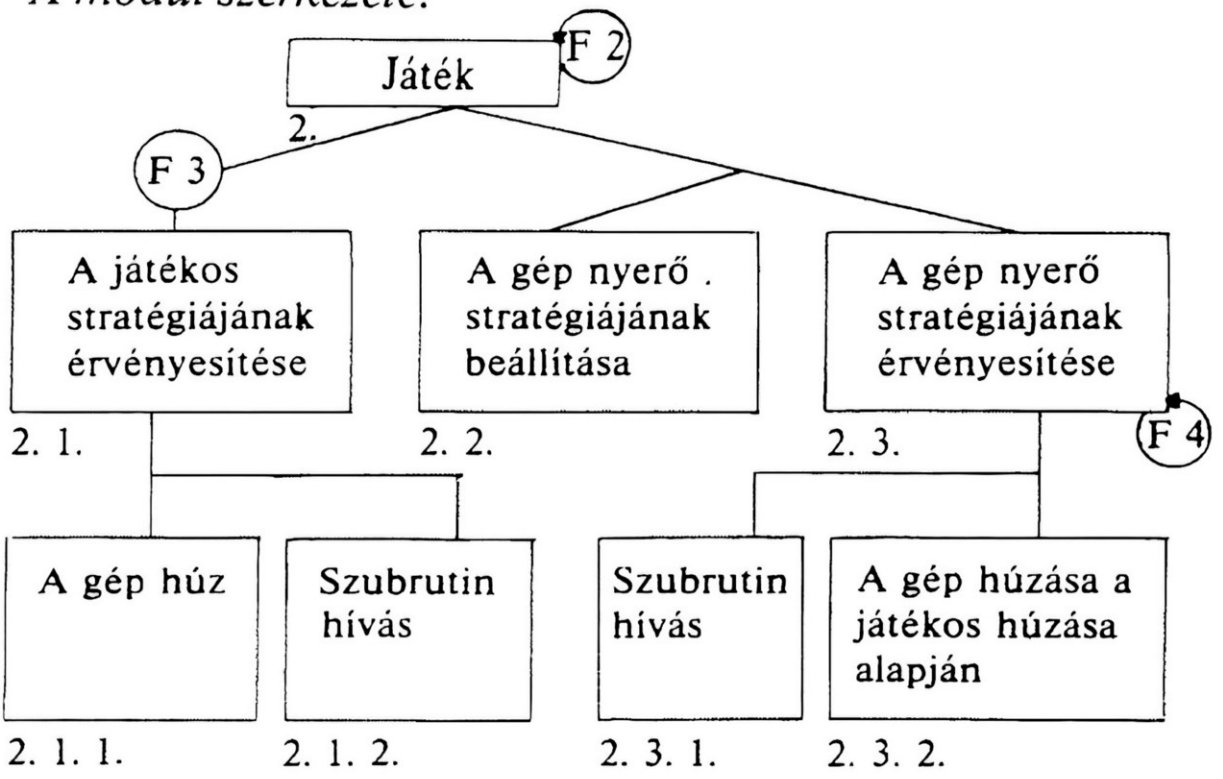
A programot a 14. játék programjának felhasználásával készíthetjük el. Átgondolva a program kidolgozásának feladatát, szembetűnik, hogy a FŐPROGRAM, a KEZDÉS és az ÉRTÉKELÉS modulok szerepe nem változik, csak a főprogramban a modul nevében kell elhelyezni STRATÉGIÁVAL szöveget. Megváltozik azonban a 2. játék modul feladata, ezért e modul programját ki kell dolgozni.

2.modul

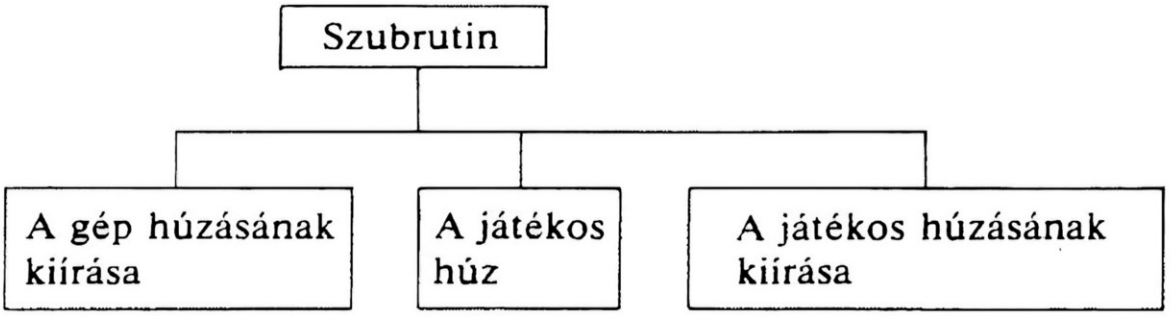
2. JÁTÉK

Szerepe:
1. Vizsgálat a játék befejezésére;
2. Vizsgálat arra, hogy a játékos, vagy a gép stratégiája érvényesül-e;
3. A vizsgálat eredményétől függően a vezérlés átadása a 2.1. vagy a 2.2. modulnak;
4. A 2.2. és a 2.3. modul soros vezérlése.

A modul szerkezete:



F2: a JÁTÉK modul megismétlése;
F3: vizsgálat az érvényesülő stratégiára;
F4: a húzások ismétlése.



Műveletek:

1. Vizsgálat, ha a gyufaszálak elfogytak, a vezérlés átadása a 3. modulra;
2. A követhető stratégia kiszámítása;
3. Vizsgálat, ha a gép stratégiája érvényesül, a vezérlés átadása 2.2. modulra, ha a játékosé 2.1. modulra;
4. A vezérlés visszavétele a 2.1. modultól.

Változók:

X–gyufák száma, A–változat beállítása

Bemenő adat:
X

Kimenő adat:
X, A$–az utolsó húzó, a szubrutinban található

2.1. A JÁTÉKOS NYERŐ STRATÉGIÁJÁNAK ÉRVÉNYESÍTÉSE

Szerepe:
1. A 2.1.1. és a 2.1.2. modulok soros vezérlése;
2. A vezérlés visszaadása a 2. modulnak.
Műveletek:
1. Ugrás a 2. modul kezdetére.

Változók:
nincs

Bemenő adat:
nincs

Kimenő adat:
nincs

2.1.1. A GÉP HÚZ

Szerepe:
A gép húzásának beállítása.
Megjegyzés:
Felépítése megegyezik a 14. játék azonos nevű moduljával.

2.1.2. SZUBRUTINHÍVÁS

Szerepe:
A program futásának áthelyezése a szubrutinra.
Műveletek:
Ugrás a szubrutinra.

Változók:
nincs

Bemenő adat:
nincs

Kimenő adat:
nincs

2.2.modul

2.2. A GÉP NYERŐ STRATÉGIÁJÁNAK BEÁLLÍTÁSA

Szerepe:
A gép nyerő stratégiájának érvényesítéséhez az első húzás végrehajtása.
Műveletek:
1. Ha A=.75, akkor L=3
2. Ha A=.5, akkor L=2
3. Ha A=.25, akkor L=1

Változók:
A–munkaváltozó, a változat beállítása, L–a gép húzása

Bemenő adat:
A

Kimenő adat:
L

2.3. A GÉP NYERŐ STRATÉGIÁJÁNAK ÉRVÉNYESÍTÉSE

Szerepe:
1. A 2.3.1. és a 2.3.2. modulok soros struktúrában való vezérlése;
2. A játék befejezéséig a modul ismételt végrehajtásának vezérlése.

Műveletek:
1. Vizsgálat a gyufaszálak számára, ha van szál, ugrás a modul elejére, ha nincs, ugrás a 3. modulra.

Változók:
X–gyufák száma

Bemenő adat:
X

Kimenő adat:
A$–az utolsó húzó, a szubrutinban van

2.3.1. SZUBRUTINHÍVÁS

Szerepe:
Ugyanaz, mint a 2.1.2 modulnak.

2.3.2. A GÉP HÚZÁSA A JÁTÉKOS HÚZÁSA ALAPJÁN

Szerepe:
A gép húzásának kiszámítása.
Műveletek:
1. A játékos húzása alapján a gép húzásának kiszámítása.

Változók:
L–a gép húzása, K–a játékos húzása

Bemenő adat:	*Kimenő adat:*
K	L

szubrutin

SZUBRUTIN

Szerepe:
A gépi húzás kiírása, a játékos húzása és a húzás kiírása művelet végrehajtása.
Megjegyzés:
A szubrutin a 14. játék 2.2., 2.3., és a 2.4. modulok programsorait tartalmazza. A szerkezeti ábrán a szubrutinnak ezeket az alprogramjait még megkülönböztettük, a programban azonban ilyen megkülönböztetést nem tettünk.
Műveletek:
1. A megnevezés leírása;
2. A gép húzása és kivonása a gyufák számából, vizsgálat túlhúzásra;
3. A gép húzásának kiírása, vizsgálat a játék végére, ha X=0 akkor ugrás a szubrutin végére;

Megjegyzés:
A strukturált programozásban a modulból való kilépés, csak a modul utolsó soránál megengedett. Erre minden programozónak törekednie kell. A BASIC nyelv azonban nem minden esetben biztosítja e szabály maradéktalan betartását.
4. A játékos húzása és vizsgálat túlhúzásra;
5. A maradék gyufaszálak számának kiírása.

Változók:
X–gyufák száma, L–a gép húzása, K–a játékos húzása, A$–az utolsó húzó

Bemenő adat:
X, L

Kimenő adat:
X, A$, K

```
10 REM"GYUFSZALAK HUZASA JATEK  STRATEGIAVAL"
20 PRINT"■":PRINT
30 PRINT "          ***  GYUFAJATEK  ***":PRINT
40 REM"KEZDES"
50 PRINT "IRJ KETJEGYU SZAMOT. ENNYI  GYUFAVAL JATSZUNK. ";
60 PRINT"FELVALTVA 1-3 SZALAT VEHETUNK EL.";
70 PRINT"AKI AZ UTOLSO SZALAT HUZZA AZ VESZITETT":PRINT
80 INPUT" SZAM ";X:IF X<10 OR X>99 THEN PRINT"KETJEGYUT!":GOTO 80
90 PRINT"■":PRINT
100 REM"JATEK"
110 IF X=0 THEN PRINT:GOTO 400
120 LET A=(X-1)/4-INT((X-1)/4):PRINT
130 IF A<>0 THEN 300
140 REM"A JATEKOS NYERO STRATEGIAJANAK ERVENYESITESE"
150 REM "A GEP HUZ"
160 LET S=RND(X)
170 IF S<.33 THEN L=1: GOTO 200
180 IF S<.67 THEN L=2: GOTO 200
190 LET L=3
200 REM "SZUBRUTIN HIVAS"
210 GOSUB 230
220 GOTO 100
230 REM"SZUBRUTIN"
240 LET X=X-L: A$="G": IF X<0 THEN L=1: X=0
250 PRINT L;"SZALAT HUZTAM.   MARADT";X: IF X=0 THEN 290
260 INPUT" A HUZASOD";K: IF K<1 OR K>3 THEN 260
270 LET X=X-K:A$="J":IF X<0 THEN PRINT"NINCS MAR ENNYI SZAL":X=0
```

```
280 PRINT"                    MARADT";X
290 RETURN:REM"VEGE A SZUBRUTINNAK"
300 REM"A GEP NYERO STRATEGIAJANAK BEALLITASA"
310 IF A=.75 THEN L=3: GOTO 340
320 IF A=.5 THENL=2: GOTO 340
330 LET L=1
340 REM"A GEP NYERO STRATEGIAJANAK ERVENYESITESE"
350 REM "SZUBRUTIN HIVAS"
360 GOSUB 230
370 REM "A GEP HUZASA JATEKOS HUZASA ALAPJAN"
380 LET L=4-K:PRINT
390 IF X>0 THEN 340
400 REM"ERTEKELES"
410 IF A$="G" THEN PRINT" VESZTETTEM": GOTO 440
420 PRINT"AZ UTOLSO SZALAT VETTED EL. VESZTETTEL."
430 PRINT:PRINT
440 INPUT"MEGISMETELJUK I/N";B$:IF B$="I" THEN 20
450 END
```

Az eddigi játékokban elsősorban a program olvasását sajátítottátok el. Már rendelkeztek olyan alapismeretekkel, amelyekkel megkezdhetitek a programok írását is. De mielőtt hozzákezdenétek, ha lehetőségetek van rá, a játékok programjait vigyétek fel számítógépre. Természetesen ennek érdekében a rendelkezésetekre álló számítógép kezelési utasítását előzetesen tanulmányoznotok kell. Kezelési utasítás minden számítógép mellett található.

parancsok

A számítógépben a programok az ember által megszabott parancsok alapján kerülnek végrehajtásra, ezért a programok futtatásához, illetve a programokkal kapcsolatos műveletek elvégzéséhez ismernetek kell a fontosabb parancsokat.

A parancsok a programok végrehajtásával összefüggő vezérlő feladatokat látnak el. A parancsokat az különbözteti meg az utasításoktól, hogy nem rendezik őket programsorokba, ezért sorszámot sem kapnak. (A félreértés elkerülése végett megjegyezzük, hogy a parancsok egy része utasításként, illetve egyes utasítások parancsként is használhatók.)

A fontosabb parancsok a következők:

LISTÁZZ parancs **listázz**
Formája: LIST **LIST**
Feladata: a program kiírása a képernyőre.

Vezérlő műveletek:	*Parancs:*
A program 10. programsorának kiírása	LIST 10
A 10. programsortól a program kiíratása	LIST 10–
Az első programsortól 100. programsorig kiíratás	LIST–100
A 10 –100 programsorok kiírása	LIST 10–100

INDÍTS parancs **indíts**
Formája: RUN **RUN**
Feladata: a programfutás indítása

Vezérlő műveletek:	*Parancs:*
Programindítás	RUN
Programindítás a 300. programsorról	RUN 300

FOLYTASD parancs **folytasd**
Formája: CONT **CONT**
Feladata: a programba épített ÁLLJ utasítás hatására megállított programfutás újraindítása.

VISSZA parancs **vissza**
Formája: RETURN **RETURN**
Feladata: a programsor, illetve parancs beírása az operatív tárba és újabb programsor, ill. parancs fogadására a gépet felkészíteni.
Miután begépeltük a gépbe a programsort, illetve parancsot, a RETURN billentyű lenyomásával érvé-

nyesítjük a VISSZA parancsot. Van olyan számítógép, ahol ezt a műveletet a NEW LINE feliratú billentyű lenyomása hajtja végre.

töröl
NEW

TÖRÖLJ parancs
Formája: NEW
Feladata: az operatív tárban levő programok és adatok kitörlése.
A TÖRÖLJ parancs beírása után új program írásához kezdhetünk hozzá, vagy a háttértárolóból hívhatunk be programot.

program kiírás
SAVE

PROGRAMKIÍRÁS parancs
Formája: SAVE „programnév", tároló szám
Feladata: a program kiíratása az operatív tárból a háttértárolók valamelyikébe.

Vezérlő művelet:	*Parancs:*
A FÜGGVÉNY nevű programot a 8. floppy disk egységre írd ki.	SAVE „FÜGGVÉNY", 8 (A Commodore–64 számítógépen)

program betöltés
LOAD

PROGRAMBETÖLTÉS parancs
Formája: LOAD „programnév", tároló szám
Feladata: a program betöltése: a háttértárolóból az operatív tárba

Vezérlő művelet:	*Parancs:*
A FÜGGVÉNY nevű programot a 8. floppy disk egységről töltsd be.	LOAD „FÜGGVÉNY", 8 (A Commodore–64 számítógépen)

programok beírása a számítógépbe

1. Kapcsoljátok be a számítógépet a kezelési utasítás szerint;
2. Ha megjelent a számítógép üzemkészségére utaló szöveg a képernyőn, soronként kezdjétek a kiválasz-

tott programot beírni. A programsorok végén üssétek le a RETURN (NEW LINE) billentyűt;

3. Ha minden sort pontosan leírtatok, és a villogó jel a VÉGE utasítást tartalmazó programsor alatt van, írjátok be a RUN parancsot, és nyomjátok le a RETURN billentyűt;
4. Hajtsátok végre a képernyőre kiírt teendőket.
5. Ha a programot tárolni akarjátok, vigyétek ki valamelyik háttértárolóra a PROGRAMKIÍRÁS parancscsal;
6. Amennyiben a RUN parancs után a számítógép képernyőjén HIBA felirat jelenik meg, pontatlanul írtátok be a programot. Ilyen esetben tanulmányozzátok a kezelési utasításban a program javításának szabályait, és az ott leírtak szerint járjatok el.

A PROGRAMOK KIDOLGOZÁSA

Sokat játszottatok, a számítógéppel is találkoztatok, és már szeretnétek ti is programokat kidolgozni. A programok kidolgozásához az ÁSZ–BASIC játékban megfelelő kiinduló ismereteket szereztetek, ezért bátran fogjatok hozzá.
Kezdetben egyszerű, egymodulos programokat írjatok.

A programkidolgozás menete a következő lehet:

a program-kidolgozás menete

1. Tervezzétek meg a programot:
   - határozzátok meg a program feladatát,
   - tisztázzátok a feladat végrehajtásának menetét,
   - írjátok le a modult;
2. Dolgozzátok ki a programot;
3. Ha lehetőségetek van, írjátok be a számítógépbe, ha nincs, kérjétek meg jól programozó társatokat, hogy ellenőrizze a programokat;
4. Dolgozzátok ki a JÁTÉKLAPOT 4 példányban;
5. Írjátok le a JÁTÉK-ot.

egymodulos feladatok

Kezdetben négy egymondatos feladat kidolgozását ajánljuk:

1. Állítsatok össze szavakat szótagokból számítógéppel!
2. Testvéretek a kertben különböző gyümölcsöt szedett a kosarába, mindegyiket megszámolta. Számolja ki a számítógép, hogy összesen mennyi gyümölcs van a kosárban!
3. Végeztessetek kockadobást a számítógéppel!
4. Szerkesszetek szavakból mondatot a számítógépen! Célszerű, ha az A, B, C, D JÁTÉKLAP programjain ellenőrzitek a megoldásaitokat!

Ha már sikeresen tudtok kidolgozni egymodulos programokat, hozzákezdhettek a programok strukturált tervezéséhez.

Először módosítsátok a kidolgozott programokat, majd ha ezt már megfelelően tudjátok végezni, kezdjetek hozzá saját programjaitok kidolgozásához.

program módosítási feladatok

A programok módosítására három feladatot dolgoztunk ki.

1. Ha beírtátok a számítógépbe a 10. játék programját, és játszottatok vele, tapasztalhattátok, hogy ismételt végrehajtásnál nem törli a képernyőt. Dolgozzátok át a programot úgy, hogy programismétlésnél a képernyő törlésre kerüljön.

Megjegyezzük: az ADATBEOLVASÓ utasítás (READ) az első programfutásnál a változókhoz hozzárendeli a szövegeket, és a programismétlésnél nem ismétli meg a hozzárendelést, csak akkor, ha előzetesen kiürítettük a változókat.

A feladatot háromféleképpen is végrehajthatjátok:
Első felcserélitek a programsorok sorszámait, az alábbiak szerint:

20–50 30–60 40–20 50–30 60–40

A második az ADATBEOLVASÁSI ISMÉTLŐ utasítás alkalmazásával:

```
340 IF E$="I" THEN RESTORE:GOTO 20
```

A harmadik:

```
340 IF E$="I" THEN PRINT"■":GOTO 70
```

2. A 11. játéknál a 60. programsorban az adatbevitel ellenőrzése csak a sorozat tagjainak számára vonatkozik, ellenben a sorozat elemei olyan számjegyek is lehetnek, amelyek nem osztályzatok (0, 6, 7, 8, 9).
Fejlesszétek a programot úgy, hogy az osztályzatsorozat egyes elemei is ellenőrzésre kerüljenek!
A megoldás egy változatát az E FELADATLAP-on találhatjátok.
3. A program kidolgozásánál különös figyelmet és aprólékos munkát igényel a ciklusok tervezése. Gondos tervezésnél is rendszerint hiba csúszik a kidolgozó elgondolásába, ezért a programozók ellenőrző kiírásokat végeztetnek a számítógéppel. A ciklusban minden egyes ismétlésnél kiíratják a ciklustörzsben végrehajtásra kerülő műveletek eredményeit.
Ezeket a kiíratő utasításokat a program kidolgozásának végeztével a programozók kitörlik a programból.
Írjatok be kiíró utasításokat tartalmazó programsorokat a játékok programjaiba.
Példát az F JÁTÉKLAP-on találtok.
A kiíró utasítítások elhelyezésével kapcsolatos kerdéseket is feltehettek. Például: Mit ír ki a számítógép, ha . . . programsor után KIÍRÓ utasítást helyezünk el, és a bemenő adat ez . . . , meg . . . ez?
Ha már szorgalmasan gyakoroltátok a programok módosítását, kezdjetek hozzá a kidolgozásukhoz.
Az induláshoz felsorolunk öt feladatot és a megoldásaik egy-egy változatát a G–K-ig JÁTÉKLAP-okon találjátok.
Ha más változatot dolgoztatok ki a feladatok végrehajtására, mint ami a JÁTÉKLAPOKON van, az még nem jelenti, hogy a kidolgozásotok hibás. A feladatoknak többféle megoldásuk van. Azért javasoljuk, hogy a kidolgozott programjaitok helyességét vagy számítógépen, vagy jól programozó társatok segítségével ellenőrizzétek.

program-kidolgozási feladatok

1. Gyuri a gimnázium I/a osztályába jár. Házi feladatként kapta, hogy ellenőrizze számításokkal a $(a+b)(m+n) = am+bm+an+bn$ összefüggés azonosságát az alábbi adatokkal

A	5	10	10
B	2	−6	22
M	3	18	8
N	6	−4	−5

majd

$(2a+3b)\times(2m-5n)$
$(5a-3b)\times(m+n)$
$(a^2+3ab+b)\times(2m-\frac{n}{3})$

szorzásokat végezze el, mindegyiket más adatsorral. Készítsétek el a házi feladatot megoldó programot!

2. Készítsetek játékot egy, a számítógéppel előállított szám meghatározására, kitalálására! A számítógéppel előállított szám 0 és 16000 között lehet.

3. John az iskolában a következő feladatot kapta:
1978 végén Uppertown és Lowertown (Nowhedewil külvárosai) lakosainak a száma megegyezett. A következő három évben azonban a lakosok száma jelentősen nőtt.

	1979.	1980.	1981.
Upertown	9 %	13 %	7 %
Lowertown	9 %	15 %	4 %

Melyik városrésznek volt több a lakosa 1981 végén? Számítsátok ki!

4. A vetélkedő játékok többségére kidolgozható valamilyen nyerő eljárás; stratégia.

Ilyen eljárást alkalmaztunk a GYUFAJÁTÉK programoknál.
Fogalmazzátok meg a stratégiát úgy, hogy mindig a számítógép nyerjen!

5. Gondolom, játszottatok már olyan játékot, amikor is egymás mellé letettetek három-négy tárgyból bizonyos mennyiséget (pl. gyufaszál 10 db, pénzérme 7 db, babszem 15 db), majd az *A* és *B* játékos felváltva elvett tetszőleges tárgyat, legalább egyet, és csak egyfajta tárgyat vehetett ei egy húzásra.
A játékot az vesztette el, aki az utolsó tárgyat vette el.
Készítsétek el a játék programját!

A programok készítéséhez sok segítséget kaphattok *Donald Alcock:* Ismerd meg a BASIC nyelvet;
Donald D. Spencer: Játékok BASIC nyelven című könyvekből.

Jó játékot és sikeres kidolgozásokat kívánunk!

TARTALOM

Egy kártyacsomag tartalma:

ÁSZ–BASIC	2 lap
DATA	2 lap
DEF	2 lap
DIM	2 lap
END	2 lap
FOR	4 lap
GOSUB	3 lap
GOTO	3 lap
IF	5 lap
INPUT	4 lap
LET	5 lap
NEXT	4 lap
ON	2 lap
PRINT	7 lap
READ	2 lap
REM	4 lap
RESTORE	2 lap
RETURN	3 lap
STOP	2 lap

A FEKETE Péter játékhoz további egy Fekete Péter kártyalap.

A játékot készítette Dr. Seebauer Imre
Megjelent az Offset és Játékkártya Nyomda, gondozásában 1985
Felelős vezető: Burger László igazgató
Tervező-szerkesztő: Seebauer Gabriella
Grafika: Illés Anna
Borítóterv: Haász István