Debreceni Egyetem
Informatika Kar

SZAKDOLGOZAT

LOGIKAI JÁTÉKPROGRAMOK A KÖZÉPISKOLÁBAN

Témavezető:
Dr. Nagy Benedek
egyetemi adjunktus

Készítette:
Asztalos Szilárd
informatika tanári szak

Debrecen
2008.

BEVEZETÉS

Az első „Neumann-elveken működő” számítógépek megjelenésével még nem gondoltak arra, hogy azzal milyen változásokat idéznek elő nemcsak a tudományos-technikai életben, hanem idővel a családok és főként a fiatalság körében. Az ötödik generációs számítógépeket már nem csak tudományos célokra, adatbázis-kezelésre, szimulációkra, „információfeldolgozásra” használták; tovább szélesedett a felhasználási kör: multimédiás felhasználás és a játékok. A játékok között pedig többnyire szimulátorok, a legkülönfélébb sportok, akció játékok az elterjedtek. Csak kevés hányadát képezik a különböző készségeket fejlesztő játékok, ha nem számítjuk ide a stratégiai játékokat, melyeknek – valljuk be – azért mégis van némi készségfejlesztő szerepe.

Ám a legkisebb korosztályként a számítógép elé kerülve nem lehet ezzel „indítani” a képességek fejlesztését. El kell sajátítaniuk azokat az alapvető gondolkodási műveleteket, melyekre a későbbi összetett algoritmikus gondolkodást igénylő feladatok, problémák épülnek. Hiszen az iskolás évek kezdetekor egy olyan sajátos szemléleti módjuk van, mellyel egyetlen felnőtt sem látja a világot: a *gyermeki.* Az indulattelt feszültségnek és a játékos megoldásnak az a gazdagsága, mely az átélésüket jellemzi. Ez a feszültség náluk átalakul az áttételek hosszú útjain érdeklődéssé, cselekvési formává, megismerési indítékká. Ennek az érdeklődésnek egy példája: Eső után guggol a víztócsa előtt, nádszállal vagy fűszállal kavargatja. Nem csodát vár, nem is sző hozzá ábrándokat. Csak lebilincseli a látvány. De ha Mi szülők elég figyelmesek vagyunk, az ehhez hasonló egyszerű játékokba csempészhetünk tanulást. Mert a játékban a gyerek nem saját személyében vesz részt: mozdonyként pöfög, oroszlánként üvölt, egy szerepet játszik, egy bábut mozgat önmaga helyett, vagy Ő MAGA a bábu.

Ez a tartalmas és gazdag perspektíva pedig a gyermekkor közepén kezd igazán kibontakozni: 10-14 éves korban. Ezt a korszakot kell a leginkább kihasználni, megalapozni a gondolkodásmódjukat. Elsősorban azért, mert ekkor kezd háttérbe szorulni életükben a játék. Ha kezükbe is akad egy-egy játékuk, szinte csak gépiesen használják. Az a tevékenység és azok a szituációk, problémák, melyeket ő maga állít fel születése óta, egyre kevésbé tudja levezetni feszültségeit, egyre kevésbé érdeklik az általa kreált helyzetek és azok játékai. Főleg napjainkban, amikor a legtöbb gyermek a számítógépek világába menekül. Hiszen ott adottak a problémák, szituációk: el kell pusztítani minden ellenfelet, minél több gólt kell rúgni az

ellenfél kapujába, stb. Ám itt jön a szerepe az egyszerű készségfejlesztő játékoknak. És mi az a készség, amely egy ember életében a legnagyobb hangsúlyt kapja? Igen, a **logika**.

A tanítási gyakorlataim alatt az általános iskolákban nem csupán 5-8. osztályig állt módomban tanítani, hanem lehetőségem nyílt az alsóbb osztályokban is tanítani, valamint immáron 6 éve tanítok egy középiskolában, és ezért olyan problémákat próbáltam ezen korosztály elé tárni, amely egy-egy számítógépes játékprogramba illesztve nem csak szórakoztat, de a logikus gondolkodást is fejleszti egyben; legyen az 8, 14, vagy 18 éves. A logika persze magában foglalja a helyzetfelismerő készséget, a megfelelő stratégia kialakításához szükséges analitikus gondolkodásmódot is, melyeknél nem mellékes a minél rövidebb időn belüli „szituációnak és céloknak megfelelő" döntéshozatal sem.

A programok elkészültekor sikerült is tesztelésre bocsátanom a programokat ebben a korosztályban, és az visszajelzések is pozitívak voltak. Ezért érzem úgy, hogy a szakdolgozatom témája számomra ideális, és annak megvalósítása is sikerült.

A programok alapjául azért választottam a Turbo Pascal 7.0 programozási nyelvet, mert az alábbi játékprogramok elkészítéséhez tökéletesen megfelel. A programok pedig változatosak: van köztük könnyen érthető, nehezebb szintű; megjelenítését tekintve szöveges és grafikus képernyőn egyaránt (a megjelenítést tekintve mindig a legcélszerűbb grafikai módot választottam, egyszerre tartva szem előtt a grafikai megjelenítést és a játék kezelhetőségét). A programozási nyelv megválasztása azon ok miatt is esett a Pascalra, mert ez a legelterjedtebb, középiskolában alkalmazott nyelv. Egy Pascal program egyszerű felépítésű, könnyen érthető, és ugyanakkor képes arra, amit egy középiskolás diák megtanulhat és alkalmazhat. Ennél fogva egy ilyen nyelven íródott programon keresztül bemutathatóak az egyes megoldások, számítási módszerek, átvezetések, algoritmusok, eljárások, függvények. Hiszen a program futásakor – a játék alkalmazása során – látja az eredményt, látja a működés folyamatát, s a forráskódot böngészve és a játékot futtatva remek prezentációs anyagot jelentenek ezek a programok a parancsok-utasítások alkalmazási okának megértésére, sorrendiségük fontosságára, az egyszerűségre való törekvés megértésére, valamint hogy felfedezze az összefüggést egy nagy felbontású grafikus felület, és egy „kis felbontású állapottér" között. Megfigyelheti a mozgás-mozgatás miértjét és az egyes események bekövetkeztének okát. Számos oktatási célú programozási feladat található a világhálón, azonban ezek többnyire középszintű érettségire készítik fel a diákokat. Viszont gondolnunk kell azokra a diákokra is, akik emelt szinten szeretnének érettségi vizsgát tenni,

akik számára már csak egy komplex feladat jelent tanulási lehetőséget. Az, hogy egy komplex prorgam egy játékprogram, csak felkelti az érdeklődését a programozás iránt. Hiszen a programozás egy 14-18 éves diák számára unalmassá válhat matematikai és egyéb tudományos problémák orvoslása során: a cél az érdeklődésük felkeltése. S mi lehet jobb, egy játékprogram készítésénél? Nincs szükség 10-20 éves pedagógiai munkásságra ahhoz, hogy rájöjjünk: a mai diákokat nem lehet lekötni, csak látványos feladatokkal. Habár ezek a játékprogramok nem 3 dimenziós megjelenésűek, nincs bennük látványos effekt. A programozás örömét az okozza a programozó fiatalok számára, hogy a játékok programozói megvalósítását ők maguk fedezik fel…

A LOGIKÁRÓL

Röviden…

A „logika” szavunk a mai magyar nyelvben mindennapi kifejezésnek számít. Számos változatban használjuk. Ezek közül néhány:

„Van benne logika.”
„Logikusan érvel.”
„Ez a dolgok logikája.”
„Logikátlan eljárás.”
„Ellentmond a józan logikának.”

Ezekben a megnyilvánulásokban a logika szó használatát mindenki érti. Általában következetességet, a gondolkodás rendezettségét dicsérjük, vagy annak hiányát tesszük szóvá. Nyelvi megnyilvánulásainkban releváns szerepe van annak, hogy logikailag hogyan minősítjük őket. Ám hol van szükség a logikára? Logikára szükség van mindenütt, ahol következtetéseket végzünk. Következtetésekre pedig szükség van a tudományban, az oktatásban és a mindennapi életben. Egy tudomány által létrehozott információmennyiségnek csupán kis töredéke explicit információ (azaz amely közvetlenül felfogható). Az információk többségét ezekből gondolati munkával nyerhetjük. Még a grammatikai szabályok alkalmazása is következtetést, bár rendkívül egyszerű következtetést igényel! Íme egy egyszerű példa:

Premisszák: A magánhangzóra végződő igék múlt idejének jele két „t” betű.
A „lő” ige magánhangzóra végződik.
Konklúzió: A „lő” ige múlt ideje „lőtt”.

A kiinduló információkat hordozó állításokat a következtetés premisszáinak, előzményeinek nevezzük, míg a napvilágra hozott rejtett információt tartalmazott állítást a következtetés konklúziójának, zárótételének nevezzük. De mit is jelent tudományosan megközelítve a logika?

A logika szűkebb értelemben a formailag szabatos következtetés, bizonyítás, érvelés tudománya (formális logika), tágabb értelemben hozzá számítják a tudományos

megismerés módszertanát, az igazság elméletét, és a *logika* bizonyos alkalmazásait is. Alapvető feladata a formailag helyes következtetés meghatározása és törvényeinek feltárása.

Tágabb értelemben hozzászámítjuk a tudományos módszerek és a megismerő gondolkodás normatív törvényeinek kutatását is.

Ám a logikára többféle definíciót ismerünk. Íme néhány:

Formális logika: szűkebb értelemben vett logika.

Hagyományos vagy tradicionális **logika**: általában a logikának a szimbolikus logika kialakulása előtti korszaka (fejlődési foka), de főleg a 17-19. század logikai tanainak elnevezése.

Matematikai logika: matematikai diszciplína, mely egyrészt a logika matematikai alkalmazásaival, másrészt a logika matematikai eszközeinek, módszereinek tisztán matematikai vizsgálatával és általánosításaival foglalkozik. Gyakran a szimbolikus logika terminussal egyezően használják.

Szimbolikus logika: a logika tudományának modern fejlődési foka, melyet G. Frege alapozott meg. Fő jellegzetességei: a logikai grammatika kidolgozása, a kvantorok bevezetése, a szemantikai értékek elmélete, a következményreláció szabatos értelmezése.

Tehát a logika, mint következtetés során az adott információkból olyan további információkat kapunk, amely kiinduló adatainkban árnyalva már megvolt. Az információkat állításokkal – grammatikailag kijelentő mondatokkal – kifejezve, a kiinduló adatokat tartalmazó állítások a következtetés premisszái (előzményei), az eredményt kifejező állítás pedig a következtetés konklúziója (zárótétele). A következtetés helyességének vizsgálatához föl kell tárni a benne szereplő állítások logikai szerkezetét. Ez a fogalom annak fölismerésén alapul, hogy bizonyos szavak az információ strukturálására szolgálnak (ezek az ún. logikai szavak, mint pl. „nem”, „és”, „vagy”, „ha ... akkor”, „minden”, „azonos”. stb.), más kifejezések pedig az információhordozás szempontjából lényeges kategóriákba sorolhatók; e kategóriák közül legfontosabb – a kijelentő mondat kategóriája mellett – az individuumnév és a predikátum kategóriája. Ezek az észrevételek vezettek el a logikai grammatika megalkotásához.

Ez tulajdonképpen a nyelvi kifejezések logikai szempontból való kategorizálása (osztályozása) és összekapcsolási szabályaik meghatározása, a logikai következményreláció szabatos definiálása érdekében.

A logikai szerkezet föltárásakor a természetes nyelven adott állításokat rend szerint át kell fogalmaznunk úgy, hogy láthatóvá váljanak az ún. logikai szavak (a modern logikában ezeket, az egyértelműség kedvéért, speciális szimbólumokkal helyettesítik), és fölismerhető legyen az egyéb kifejezések kategóriákba sorolása.

A következtetés formai helyessége kizárólag a benne szereplő állítások (a premisszák és a konklúzió) logikai szerkezetén múlhat, azaz csakis az előforduló logikai szavak jelentésén és az egyéb kifejezések kategóriákba sorolásán, az utóbbiak jelentése nem befolyásolhatja. Ezért minden helyes következtetés sematizálható a nem logikai alkatrészek betűkkel (változókkal) való helyettesítésével, amely megfelel az eredeti kategorizálásnak, ismét helyes következtetés tartalmilag is helyes a következő értelemben: a premisszák igazsága szükségszerűen maga után vonja a konklúzió igazságát. E tényt igazolja a logikai szemantika. A mindennapi életben gyakran következtetünk, és ennek alapján előrejelzéseket (jóslásokat) teszünk; ezek néha beválnak, néha nem. A negatív esetekben – föltételezve premisszáink igazságát – következtetésünk nem volt formailag helyes. A logikai szerkezet feltárása kisebb vagy nagyobb mélységig történhet. A logika fejlődésének egyik fő mozgató rugója éppen az a törekvés, hogy a természetes nyelv változatos kifejezéseinek minél nagyobb részét bevonja a logikai elemzés körébe. Ez maga után vonja a logikai szemantika bővítését, kiterjesztését. Ezért a logika nem befejezett, hanem szüntelen fejlődésben lévő tudomány.

Története

Az antik görög filozófiában és matematikában felfedezték a bizonyítás és az érvelés egyes fogásait, törvényeit. Az első rendszeres logikai elméletet Arisztotelész alkotta meg (a kategorikus szillogizmusok elmélete, szillogizmus), ezért őt tekintik a logika atyjának. Munkásságát a peripatetikus iskola filozófusai folytatták. Valamivel később a sztoikusok megalkották az állításlogika első elméletét (sztoikus logika). Arisztotelész analitikának nevezte a logikát, és a filozófiai vizsgálódás előtudományának tekintette. A sztoikusok viszont a filozófia részének minősítették, és dialektikának nevezték. Maga a „logika” terminus az i.sz. 3. században aphrodisziaszi Alexandrosz kommentárjaiban jelenik meg a mai értelemben.

Az i.sz. 5. században a peripatetikus és a sztoikus logika eredményeit összeolvasztották; ekkor tájt lett a logika nyelve a latin (a korábbi görög helyett). Később, a római birodalom

bukása után a 8. század végén éled újra az európai műveltség; lassanként előkerülnek az antik logika forrásművei.

A 12. századtól a skolasztika tudósai számos probléma vizsgálatával gazdagították a logikát, de nem fejlesztették ki az ehhez szükséges jelöléstechnikát. A 11. században jelennek meg először élő nemzeti nyelveken – úgy, mint angol, vagy francia nyelven – írt logikai értekezések. Leibniz vetette föl elsőként a logika „matematizálásának" szükségességét, egy univerzális szimbólumnyelv (lingua characteristica universalis) és egy következtetésellenőrző kalkulus (calculus ratiocinator) kidolgozásának igényét.

A reneszánsz korában – már a 15. századtól – megkezdődött a logika tekintélyének hanyatlása, egyebek kötött azért, mert a logika és a matematika kapcsolata – amely a logika születése idején lényeges volt – meglazult az új matematikai módszerek akkori logikai megalapozatlansága miatt.

A 17. századtól alakult ki az iskolás-hagyományos logika, amely az ókor és a középkor logikai eszméit és eredményeit fölhigította és eltorzította, ismeretelmélettel és pszichologizálással keverte; az váltotta ki a filozófia és a tudomány részéről az ellenszenvet a „meddő és semmire sem jó" formális logikával szemben. Kant a formális logikát befejezett tudománynak tekintette, és kifejtette a transzcendentális logika eszméjét. Hegel teljes lét- és ismeretelméletét logikának nevezte (dialektikus logika).

A 19. században angol matematikusok figyeltek fel arra, hogy a logika hagyományos szerkezetei és törvényei algebrai keretekben tanulmányozhatók (G. Boole, A. De Morgan, logikai algebra, osztálykalkulus). A logika modern megújulása a szimbolikus logika kialakulása G. Frege nevéhez, s az 1879-ben megjelent Begriffsschrift Fogalomírás c. művéhez kapcsolódik.

Az új logikai elmélet a 20. században először a matematikai alkalmazásokban – a matematikai bizonyításelméletben –, később pedig a tudományok módszertanában jutott kiemelkedő szerephez. Napjainkban minden rendszerező-törvényalkotó tudománynak értékes segédeszköze.

A logika tudománya

ÁLTALÁNOS MEGÁLLAPÍTÁSOK

A logika azon módszereknek és elveknek a tudománya, amellyel megkülönböztethető a jó (helyes) gondolkodás, ill. érvelés a rossztól (a helytelentől). Egy kutató számára a legfontosabb amit tanulhat az az, hogy hogyan kell helyesen gondolkodni és érvelni. A logika a helyes gondolkodás sok nélkülözhetetlen elemét foglalja össze. A helyes gondolkodáshoz azonban a szándék is szükséges: „sine ira et studio", azaz előítéletek és részrehajlás nélkül, kizárólag a gondolatmenethez tartozó információkat és a logika szabályait kell figyelembe vennünk.

A logika egyben a (racionális) ész tisztelete is. A logikát helyesen tudományként és művészetként kell megközelítenünk. Habár az érzelmekre való hagyatkozás néha hatékony, hosszútávon a gondolkozásra való hagyatkozás hatékonyabb.

A logika tanulmányozása élesíti a logikai érzéket. A logika tanulásának előnye a megnövekedett képesség

- a gondolatok tiszta és tömör megfogalmazására,
- fogalmak pontos definiálására,
- figyelmesebb olvasásra, az olvasottak jobb megértésére,
- érveléseink jobb megszervezésére, az érvelések szigorú szabályainak betartására,
- az érvelések kritikai elemzésére.

A logika tárgyalása előtt azonban tisztában kell lenni néhány alapfogalommal[*]:

Állítás: a modern logika egyik alapfogalma: a kijelentő mondat egyértelmű információtartalma. A modern logika alapvető feladatai közé tartozik a következményreláció legáltalánosabb törvényszerűségeinek föltárása. A következményreláció állítások közötti olyan kapcsolat, melyben a premisszák igazsága szükségszerűen maga után vonja a konklúzió igazságát. Az állítás nyelvi formája a kijelentő mondat, a logikai vizsgálat számára is így válik hozzáférhetővé. Ám egy kijelentő mondat gyakran csak a használat körülményeinek figyelembe vételével hordoz egyértelmű információt, igaz vagy hamis volta csak ezzel együtt válik egyértelművé. Egy állítást mindig megillet a két igazságérték - az igazság, ill. a hamisság - egyike (de csak egyike), s ez a kijelentő mondatokra nem mindig teljesül, ha

használatuk körülményeit nem vesszük tekintetbe. Ezért az állítás és a kijelentő mondat fogalma nem azonos (az első logikai, a második grammatikai fogalom). Ehhez járul még, hogy egy állítás (egyazon természetes nyelven belül is) több különböző kijelentő mondattal is kifejezhető lehet. Ezért: ha egy logikai elemzésben bizonyos kijelentő mondatokat állítások hordozóinak tekintünk, akkor kontextusukat (használati körülményeiket) rögzítettnek kell föltételeznünk. Az állítás fogalmának használata két szempontból is rendkívüli elméleti jelentőségű: egyrészt lehetővé teszi a logika számára, hogy független legyen bármely természetes nyelvtől, másrészt explicitté teszi az általánosságnak a logikában alkalmazott szintjét.

Kijelentés: gyakran az állítás szinomímája a modern logika irodalmában. A két kifejezés között azonban finom megkülönböztetést tehetünk: a kijelentést a kijelentő mondat jelentése vagy intenziója, az állítás pedig az az információtartalom, amelyet a mondat jelentése és használatának kontextusa együttesen határoz meg. (Pl. az „ő tegnap elutazott" mondat csak a használat kontextusával - az „ő" és a „tegnap" konkrét referenciájának megadásával - együtt fejez ki határozott információt.) Ez a megkülönböztetés főleg az intenzionális logikában jelentős. A propozíció többnyire a kijelentés ezen értelmében szerepel a szakirodalomban.

Logikai igazság: olyan állítás, amely logikai szerkezete (a benne szereplő logikai szavak jelentése) alapján igaz. A szemantikus fölépítésű formalizált logikai rendszerekben logikai igazságok; vagy érvényesek azok a formulák, amelyek minden megengedett interpretációban igazak.

Logikai művelet: a hagyományos logikában gondolkodási tevékenység, pl. azonosítás, megkülönböztetés, analízis, szintézis, általánosítás, konkretizálás, ítéletalkotás, meghatározás, következtetés stb.

Logikai rendszer: tágabb értelemben valamely összefüggő logikai elmélet, pl. az arisztotelészi logika.

A szimbolikus logikában: a következményrelációnak valamely formalizált nyelvre alapozott elmélete. Két logikai rendszer ekvivalens, ha nyelvünk közös (vagy legalább formuláik között kölcsönösen egyértelmű fordítás létesíthető) és a következményreláció terjedelme a két rendszerben azonos. Az ekvivalens logikai rendszereket többnyire egyazon rendszer különböző fölépítési módjainak tekintik. Ilyen értelemben beszélhetünk pl. a klasszikus elsőrendű logika rendszeréről, amelynek számos ekvivalens fölépítési módja

ismeretes. A logikai rendszer szintaktikai, ha kizárólag szintaktikai szabályokra alapozott (logikai kalkulus), és szemantikai, ha a következményreláció definíciója szemantikai szabályokra (is) hivatkozik (szemantikus rendszer).

A logika a helyes gondolkodás érdekében az érvelések helyességét vizsgálja. Az érvelés olyan speciális gondolkozás, amelyben feltevésekből (premisszák) következtetéseket (konklúzió) vonunk le. A következtetés (inferencia) során egy megállapítást (propozíció, ami lehet premissza vagy konklúzió) teszünk a következtetéshez felhasznált megállapítások segítségével. Az érvelés (argumentáció) egy vagy több következtetést tartalmaz. Az érvelés más megfogalmazásban propozíciók olyan együttese, amelyek közül egyről, a konklúzióról a többit felhasználva azt állítjuk, hogy igaz.

Az érvelések helyességének vizsgálatakor az érvrendszerben először is meg kell találni

- az egyes érveléseket (argumentációkat), és
- a premisszákat és a konklúziókat.

Fontos, hogy az érvelésekbe értendők a „józan paraszti ész”-ből adódó következtetések, a „köztudottság” is, amit az érvelésből általában elhagynak, bár annak szerves részét képezi.

A JÁTÉKRÓL

Története

Egy ember életének első korszakában, a gyermekkorban, a játék kapja a legnagyobb szerepet. Hiszen még mielőtt megtanulunk járni, a játék már kitölti mindennapjaink egész idejét, mivel ez a legfőbb tevékenységünk rövid gyermekkorunkban. A gyermek pedig mindenben a játékot keresi: legyen egy szobában a szőnyegen üldögélve, vagy odakint a természetben. Ám mióta van ez így? Vajon az „ősember” alakította ki ezt? Öntudatlanul vagy tudatosan? Esetleg mástól leste el? Vagy nem csupán az emberi élet egyik sajátosságáról van szó? Esetleg valami többről…

A válasz: igen. A játék természeti jelenség, mely kezdettől irányította a világ folyását: az anyag kialakulását, élő struktúrákká szerveződését, valamint az ember magatartását.

Játékunk története az idők kezdetéig nyúlik vissza. Az ősrobbanás energiája volt az, amely az anyagot felkavarta, hogy soha többé nyugalomba ne juthasson. Rendező erők törekedtek a széthullókat összetartani, a véletlent igába hajtani. Ami azonban e módon létrejött, az nem a kristály merev, hanem az élő dinamikus rendje. A Véletlen kezdettől fogva elválaszthatatlan ellenpárja a Szabálynak.

A Véletlen és a Szabály a Játék elemei. Ami valaha az elemi részecskék, atomok és molekulák szintjén kezdődött el, az most agysejtjeinkben folytatódik. A játék nem az ember alkotása, mégis „az ember valamennyi állapota közül épp a játék és csak a játék az, ami teljessé teszi őt”.

Nem a játékból ered-e minden képességünk? Először izmaink és tagjaink játéka válik céltalan kapálózásból pontosan koordinált mozgásfolyamokká. Majd az érzékszervek játéka alakul játszi kíváncsiságból mélyreható tudássá, a színekkel, formákkal, hangokkal való játék halhatatlan műremekké. Játékkal indul a szerelem: a szemek játéka, a tánc, a gondolatokkal és érzelmekkel folytatott kölcsönös játék. A szanszkrit nyelv a szerelmesek összeolvadását „kridaratnam”-nak, a „játékok drágakövé”-nek nevezi.

Minden játéknak megvannak a maga szabályai. Ezekkel határolja el magát a külvilágtól, a valóságtól, és állítja fel saját értékrendjét. Aki részt akar venni benne, annak el kell fogadnia a szabályokat – különben játékrontóvá válik. A társasjátékokban előzetesen megállapított formák irányítják a játék menetét és szabják meg értékrendszerét. Öntevékeny játékmenetre

ösztönözhet az is, hogy egy véletlen esemény következménye visszahat forrására. Így kezdte a maga játékát az élet is, és így játszanak gondolataink és képzeleteink is.

A játék értelme és öröme

Amikor *a gyermekpszichológus figyeli, feljegyzi, értelmezi azt az izgalmas folyamatot, ahogyan a csecsemő a bölcsőből átkerül a járókába, majd kilép a rács mögül, és nekiindul a világnak, saját fejlődését is vizsgálja, saját múltját is kutatja - önéletrajzát írja. Ítéleteinek és viselkedésmódjának, életvezetésének és kötődéseinek az előképét próbálja feltárni. Felnőttkori vágyainak és gondolatainak a forrásvidékét igyekszik bejárni. Mintha azokat a benyomásokat, hatásokat kutatná, amelyekből énje, lelki életének az a szabályozó, műveletkész szintézise kialakult.

Ennek a vizsgálódásnak egyik ígéretes lehetősége a játék elemzése. Királyi út a gyermeki éntől a szintézisig vezető fejlődés megismeréséhez, írja Freudra utalva Erickson, s így magyarázza a gyermekpszichológia fokozott érdeklődését a játék iránt.

A társadalmi fejlődés más szintjén, hasonló összefüggés alapján módosul a játék a mi értékrendszerünkben is. A munkaidőnek, a házi munkára fordított időnek jelentős csökkenése, a higiéniai viszonyok módosulása, a bűntudatot terjesztő vallási nyomás enyhülése nagyobb teret enged a játéknak a felnőtt életében is. S ez nem annyira a kifejezett játékra fordított idő növekedésében érvényesül, hanem inkább a szabadidő-tevékenység különféle formáinak, ágazatainak a játékos színezetében.

A játék pszichológiai vizsgálatát a gyermektanulmányi és pedagógiai nézőpont mellett így önismereti igényünk, valamint a szabad idő felhasználásával összefüggő társadalmi, közműveltségi tervezés is indokolja.

12 megközelítés a játékra

A pszichológiai elméletképzésnek mintegy 70-80 éve kitüntetett problémája a **játék** magyarázata. Számos jelentős gondolkodó vetette fel a játék biológiai értelmének kérdését. Abból indultak ki, hogy a játék az egész emlősvilágban, sőt csaknem az egész állatvilágban megfigyelhető viselkedés, s így feltehetően valamilyen biológiai funkciót kell betöltenie. Számos magyarázat született, mindegyik valamilyen tapasztalat általánosításával értelmezte a játékot.

A hagyományos magyarázat a munka és a játék szembeállításának sablonjából, a naiv világkép örökéből származik. A játék eszerint pihenés, a fáradt ember felüdülése. Tudjuk, a játéknak ilyen indoka is lehet. De általánosításának a cáfolásához elegendő az a megfigyelés, hogy a fáradt, kimerült gyerek játszani az elalvásig képes.

Az első tudományos igénnyel fellépő magyarázat a fölösleges energia elvezetését látta a játékban. Magyarázata számos játék-megnyilvánulásra érvényes, főként a szabad időben végzett játékos művészkedésre.

A következő értelmezés a fejlődéstani elméletképzés egyik nagy illúziójához, a biogenetikai törvényhez tapadt. Ez az analógiákból szőtt gondolat azt feltételezi, hogy az egyén fejlődése a törzsfejlődés szakaszainak rövidített megismétlése. Így a gyerek játékában az emberiség őstörténetének és történetének letűnt korait idézné fel. A feltevés, mint analógia összecseng a nyolcéves gyerek ősember-játékával. De nagyon nehéz lenne a villanyvasúttal és különféle elektronikus játékszerekkel foglalatoskodó gyerek tevékenységét ezzel az elmélettel értelmezni.

Sokáig általánosan elfogadott gondolat volt, hogy a játék gyakorlás, amelyben a gyerek, illetve a kölyökállat felkészül a fajtájának megfelelő felnőtt életre. A nevezetes példa: a kismacska játékosan ugrik a pamutgombolyagra, s így gyakorolja a mozdulatot, ahogyan felnőtt macskaként majd az egérre ugrik. A magyarázat vezető tétele: a fiatal állat nem azért játszik, mert fiatal, hanem azért fiatal, hogy játsszon. Játék nélkül nem tudna felkészülni a felnőtt létre. De látott-e valaki olyan macskát, amelyik azért nem tudott kandúr korában egeret fogni, mert cica korában nem volt alkalma játszani a pamutgombolyaggal - mondja a vitában egy svájci pszichológus, rámutatva a magyarázat önkényes túlzásaira.

Az ehhez kapcsolódó következő elmélet a felnőttségre való felkészülés katartikus jellegét emeli ki. Az embergyerek káros adottságokkal, veszedelmes ösztönökkel jelenik meg a világban, ezektől megtisztul a játék folyamatában. A játék katartikus hatása nem vitatható. De távolról sem olyan általános, ahogyan ezt ez a magyarázat feltételezi. Sőt előfordulhat, hogy a játékos tevékenység nem elvezeti, hanem megerősíti az egyébként káros törekvést. Millar joggal mondja, hogy hiába játszik a gyerek ólomkatonáival, ettől nem szabadul meg az agressziótól.

Másféle magyarázatot kínál a pszichoanalízis. Nem lép fel azzal az igénnyel, hogy a játékról sajátos összefüggő magyarázatot adjon, csupán azt hangsúlyozza, hogy a játék az

elhárított vágyak áttételes teljesülésének kitüntetett tartománya. Fiktív világában megnyilvánulhatnak az akadályozott indulatok, szabadabban fogalmazódhatnak meg a rejtett konfliktusok, a katarzis lehetősége adva van, jóllehet sajátos, nem is könnyen realizálható feltételekhez körötten, amint ezt a játék terápiás felhasználása mutatja.

A fejlődéstani gondolathoz tapad az a magyarázat is, hogy a játékban veleszületett, de az élőlény számára már fölöslegessé vált cselekvésmódok nyilvánulnak meg. A játék ebben az elgondolásban biológiailag üres járatú viselkedés, jelentését vesztett működés, csökevény a viselkedésben) a csökevényes viselkedések, például a libabőr élettani analógiájára.

E bölcseleti hangsúlyú magyarázattal szemben a viselkedéstan vezető felfogása, a tanuláselmélet éppen ellentétes irányban halad. Nem tekinti a játékot sajátságosan elemezhető pszichológiai jelenségnek. Feltevése szerint a gyerek éppen úgy megtanul játszani, mint ahogyan megtanul - veleszületett fiziológiai apparátusának bázisán - a léthez szükséges minden egyéb cselekvést. Ezt a gondolatot a motivációelmélet kiegészítette azzal, hogy a játékra ugyan nem érvényes a jutalmazásnak az az indítékmódja, amelyre az élőlények a valóságban való eligazodást és a létszükségleti helyzetmegoldásokat megtanulják. De léteznek másodlagos jutalmazásértékű indítékok, amelyek az embergyereket is vezethetik a játékban.

S ami még lényegesebb: az élőlény saját reakciója, mint inger, új reakciót vált ki, s így indul el a játéknak megfelelő öngerjesztéses megismétlésnek a lánca. Ez a magyarázat, amelyet Millar is kiemel, a gyermeklélektan egyik legfontosabb magyarázó elvét, a Baldwintól származó cirkuláris reakciót eleveníti fel. Egy nagyon korai és nagyon sokáig megfigyelhető viselkedésmódról van szó. A csecsemő ad egy hangot, később végez egy gesztust. A közelében levő felnőtt ezt megismétli, mire a gyerek utánozza a hallott hangot vagy a látott gesztust. Ez a reakció az én alakulásának fontos összetevője, s egyben a játékos tevékenység sajátos módja. Beletartozik tehát a játék körébe, de távolról sem annak egyértelmű magyarázat.

Éppen így lényeges vonását emeli ki a játéknak a Pavlov felfogását követő magyarázat. A játék eszerint az új ingerekkel kiváltott, ismeretlen helyzetekben mutatkozó orientációs reflex működésének a megfelelője. Az inger helyzeti újdonsága idézi elő, s így az új helyzetben való eligazodás próbálkozásait, az új adekvát cselekvést megelőző tájékozódást képviseli. A játéknak egy körét, az explorációs játékot e gondolat nélkül nem értelmezhetnénk. Millar is ezt az utat követi a játékos fürkészés magyarázatában.

A pszichológia történetében újnak tekinthető magyarázatok egyike a játékot a gyermeki dinamikával azonosítja (Buytendijk), s feleleveníti azt a koncepciót, hogy a gyerek azért játszik, mert gyerek, ez a sajátos megnyilvánulási módja, csak ezt teheti. Gazdag megfigyelési anyaggal alátámasztott felfogás, általános érvényét csak az teszi kétségessé, hogy a gyermeki dinamikának nem minden megnyilvánulása játék. A gyermeki ismeretszerzésnek más útjai is vannak. Ez érdeklődés nemcsak játékos tevékenységhez vezet.

A dinamikai elmélethez tartozik még az a magyarázat is, amely a játékban a gyermeki tevékenység szabad megnyilvánulását, a még nem gátolt, nem szabályozott spontaneitást látja. A megfigyelések ezt is igazolnák. Éppen csak a feltevés általánosításának mondana ellent a nem játékos, spontán megnyilvánulások gyakorisága, főként a kötődésekben, az érzelmi odafordulásban.

A személyiség érvényesülését a játékban, mint spontán tevékenységben, minden mélylélektani irányzat hangsúlyozza. Az áttételes vágyteljesítés mellett ebbe az értelmezési körbe tartozik a kompenzációs magyarázat. A játék valóban alkalmat nyújt a hátrányos helyzet „kiegyenlítésére” s ilyen hátrányos helyzet maga a gyerekség ténye, a kiszolgáltatottság, a függő helyzet, amely a játékban a felnőttségnek, a fölénynek a kellékeivel és stílusfordulataival kompenzálható. Ez az Adler gondolatmenetén alapuló magyarázat is megragad valami reálisat a játéktól, de távolról sem fejezi ki annak általános értelmét.

Susanna Millar gondosan felsorolja, részben elemzi is e sokféle magyarázatot, de érvényességükkel szemben bizalmatlan. Nem azt állítja, hogy az életre való felkészülés vagy a kompenzáció, vagy a vágyteljesítő indulatelvezetés idegen lenne a játéktól, sőt viszonylag gazdagon szemlélteti mindazokat a jelenségeket, amelyekből egy-egy játékelmélet indul. Azzal a feltevéssel szemben bizalmatlan, hogy a játék körébe tartozó sokféle viselkedésmód, tevékenységi fordulat egyetlen magyarázattal értelmezhető lenne, bármilyen szellemes konstrukcióval próbálkoznánk is.

Az egységes biológiai jelentés meghatározására törekvő magyarázatokkal szemben Millar utal arra, hogy a játék paradox viselkedés, meglepő ellentéteket foglal magába, s ezeknek egységbe foglalása teszi illuzórikussá egységes biológiai jelentésének a feltevését. A játszó gyerek roppant erőfeszítéseket tesz azért, hogy a játék folyamán megismerjen valamit, amit már ismer. Véget nem érően gyakorol egy gesztust, egy viselkedési fordulatot, holott már tudja. Társas feltételekhez szabja a játékos cselekvést, holott annak, amit társasnak tekint, nincs intézmény alapja. Közben színlel, tettet, úgy tesz, mintha; és mindebben mégsincs

színlelés, sem csalás. S még ennél is lényegesebb Millar szerint, hogy az öröm átszínezi mindazt, ami játék, de mindaddig, amíg játék, kielégülést nem hozhat. Az ismertnek a megismerése, a tudottnak a gyakorlása, a kielégülés nélküli öröm keresése azonban - s ez Millar szövegéből is kiderül - nem valódi paradoxonok, nem összeegyeztethetetlenek.

Az ellentmondások látszólagosak:

1. A megismerő játék ismeretlenje a gyerek számára sejtett és jelzett, de nem abban a mélységben ismert, amelyben a játékot motiváló kíváncsiság rejlik.
2. Éppen így a gyerek, már valahogyan tudja azt, amit gyakorol, de még messze van a cselekvési biztonságnak és könnyedségnek attól a fokától, amelyen a funkcióöröm megélhető.
3. A biológiai létfenntartás szempontjából, az elsődleges szükségletek szintjén a játék nem hozhat kielégülést. De az ember szintjén nem ez az egyetlen szükségletrendszer, amely kielégülést hozhat. A biológiai szükségletekhez sokféle kvázi-szükséglet kapcsolódik, feszültséget kiváltó, így cselekvés-meghatározó, feszültségoldó viselkedéshez vezető igények, amilyen a díszítésnek, a művészi kivitelezésnek, a szimmetrikusnak vagy az aszimmetrikusnak az igénye, akár magának a játékosnak az igénye.

Millar nem megy végig ezen a Lewin kísérleteihez visszanyúló gondolatmeneten, de habsúlyozza, hogy a játékviselkedés paradoxiái látszat-paradoxiák, s így nem kell feltétlenül lemondani arról, hogy a játékot, mint viselkedést magyarázzuk, elméletbe foglaljuk. Aggályai inkább a kritikai spekuláció fogódzói, amelyek végül mégis rákényszerítik az elméletképzés útjára, amelynek folyamán, mint látni fogjuk, Millart még az egységes elmélet kialakítása is megkísérti. Ebben az elméletképzésben *Piaget* gondolatmenetéhez kapcsolódik.

Piaget elmélete

Jean Piaget *magyarázórendszere külön világ a gyermeklélektanban. Varázslatos intellektuális építmény, ismeretelméleti fejlődéslélektan. A természetes és kísérleti helyzetben megfigyelt gyermeki viselkedésmódok, fejlődési változások értelmezésére alkalmas fogalomrendszer.

A fejlődés éveit Piaget elemzései periódusokra, szintekre tagolják. Ehhez a rendszerhez viszonyítjuk a megfigyelt jelenségeket, s a szakaszokhoz való tartozás kritériumainak segítségével lehántjuk róluk a véletleneket, és fejlődési jelentőségüket meghatározhatjuk.

Piaget gyermeklélektani gondolatrendszerének alaptétele, hogy a fejlődés egyenes vonalban ugrások nélkül halad a reflextől az érett logikai műveletekig. A logikai formák rámintázódnak az első helyzetmegoldási viselkedési módokra, sőt szerkezetükben idegrendszeri struktúráknak felelnek meg. Ezt a fejlődési folyamatot Piaget a helyzetmegoldó alkalmazkodás szempontjából vizsgálja. Kimutatja - talán inkább feltételezésként fogalmazza meg-, hogy az alkalmazkodás két funkciónak az együttes hatása. Az egyik az asszimiláció: az új tartalmat úgy értem meg, hogy bevonom egy már kialakult cselekvési sémába, hasonlítom az ismerthez. A másik funkció az akkomodáció: a meglevő cselekvési sémámat hozzáillesztem, hozzáidomítom az új benyomáshoz. Az asszimilációval a világot alakítom magamhoz, az akkomodációval magamat alakítom a világhoz. Az előbbinél hasonítok, az utóbbinál idomulok.

Piaget rendszerében a játék legegyszerűbb formája, ennek a fejlődésnek a kezdete, az érzékszervi-mozgásos értelem szintjének megfelelő gyakorló játék. Sokszoros megismétlése valamilyen gesztusnak, mozdulatnak, érzékszervi-mozgásos viselkedési egységnek (egy tág gurításának, pörgetésének, dobásának, az ujjak mozgatásának, a kar lengetésének, egy rikkantó hangadásnak stb.) Nincsen benne sajátos játéktechnika (labdázás, babázás, kocsitolás, autóhúzás), vagy ha ez a tárgy révén előadódik is, nem indítéka a játéknak. Nincsen benne szimbolikus behelyettesítés sem, a gyerek feltehetően nem képzel bele valamit. Ennek a játéknak a miértje, indítéka a funkcióöröm, valamint az „én csinálom” öröme. Az élet első másfél évének játékfajtája ez. Minden új készségnek a szükségszerű gyakorlása, minden új funkciónak a bejáratása. Ezért rendkívül labilis. Csak addig érdekes, amíg a bejáratásra szükség van, márpedig a kicsik gyorsan fejlődnek, gyorsan tanulnak, rövid idő alatt eljutnak a begyakorlást jelző telítődéshez. Egyre kevesebb alkalom kínálkozik, mert a gyerek egyre kevesebb játékos begyakorlást igénylő új működést tanulhat meg.

Ez a fajta játék még messze van a tiszta asszimilációtól, a funkcionális asszimilációnak felel meg. Működteti, ritualizálja a funkciót, elpróbálja és megtanulja minden helyzetben, minden tárgyon minden vonatkozásban, amelyre alkalma nyílik. Szerkezetét tekintve Piaget a cirkuláris reakcióból vezeti le: a gyerek egy gesztuság külső közvetítő segítségével utánozza (utánozza azt, aki őt utánozta!). S így állandó körkörös menetben egyre csiszoltabb, egyre

kidolgozottabb cselekvéssorok indulnak meg. Piaget, s nyomán Millar nagyon hangsúlyozza, hogy ezt a játékmódot még nem a tárgya, hanem a működésének, a funkciónak az öröme teszi játékká. Nem a labda vagy a gumiállat a játék, amit a szőnyegen mászó nyolchónapos szorongat, vagy a fogódzkodva álldogáló, esetleg imbolygó tizenöt hónapos már húz vagy lök, hanem a szorongatásnak, a húzásnak, a lökésnek a funkciója. A második életév második felében ez a játékfajta fokozatosan háttérbe szorul, s jóllehet még sokáig fennmarad, és szokásnak álcázva átszövi életünket, elsődleges jelentőségét a játékban elveszti. Egy másik játékfajta bukkan fel, s ezzel kezdetét veszi Piaget és az őt követő Millar szerint a játék jellegzetes világa, a szimbolikus játék. Ahogyan a gyakorló játék az érzékszervi-mozgásos értelem fejlesztője, úgy a szimbolikus játék a belső szemléleti képekben, képzeletekben folyó értelem kimunkálását segíti elő.

Az ebbe az élménykörbe sorolható képzeleti játékokban a dolgok tetszés szerint, illetve a gyerek meglévő sémái szerint alakulhatnak, és ezekhez hasonulhatnak. Piaget írja:

> *Négy év három hónapos kislánya odaáll mellé merev tartásban - miközben ő dolgozik -, és harangzúgásra emlékeztető hangot ad, bim-bamozik. Kéri a gyereket, hagyja abba. A gyerek változatlanul folytatja, mire ő a kezét a szájára teszi. A gyerek ellöki a kezét, és fölháborodva, még mindig merev testtartással mondja: "Ne nyúlj hozzám. Én egy templom vagyok!".*

A játéknak ez a szép példája nem tiszta asszimiláció; az utánzó mozzanat is hangsúlyos benne. Mutatja azonban ennek a játékszakasznak a „mintha” jellegét, a „hidd el” tudatát, s megmutatja mindenekelőtt Piaget magyarázatmódjának egyik fontos elvét: az interiorizálást, a cselekvésben szerzett tapasztalás befelé fordítását, belső megélését. A gyerek a cselekvés szintjén alakítja a torony - harang - templom képzeletkört; a játékban cselekvésesen, s ennek emléknyomaiból, cselekvéses vázlatából munkálja ki a valóságot még őrző, de már műveleti gondolkodásra alkalmas belső szemléleti képet, s ennek még szűrtebb változatát, a képzetet.

Ebből és az ehhez hasonló Piaget-elemzésekből jut Millar arra a figyelemre méltó következtetésre, hogy a játék ezen a szinten a tapasztalatok újra kódolása, helyzeteknek, történéseknek a megértése, ismereteknek az elrendezése a játékos áttételek kódnyelvén.

A „mintha-tudat” lehetővé teszi, hogy viszonylag szórványos hasonlóságok alapján bevonjon valami újat a tapasztalatai maradványokat őrző sémákba, s ezzel benyomásait a saját életterében adekvát rendszerré szervezze. Millar itt Piaget nyomán emeli ki, hogy a szimbolikus játék azt a szerepet töltheti be a gyerek gondolkodásában, azzal a különcséggel, hogy egy gyerek nemcsak felidézi, megnevezi, hanem meg is éli azt, amit ezen a módon végiggondol.

A „hidd-el” játék minden labilitása, kiszámíthatatlansága mellett is a megismerésnek az útja: megragad valamit a külvilágból és azt egy tapasztalathoz, viszonyítja. Igaz, ez a tapasztalás nem a logikailag ellenőrzött ismeretszerzés rendjébe tartozik, logika előtti szabályozás, amely éppen a játék élménye révén teljes értékű.

A szimbolikus játékkal természetesen még nem ér véget a játék fejlődése. Az 5-6 éves korral egyre gyakoribbá válnak a konstrukciós játékok, a társas játékok, s ezzel a játék már elsődlegesen a szocializáció részévé válik.

Piaget gondolatrendszerében a játéknak van egységes funkciója. Az adott fejlődési szakasznak megfelelően szolgálja az értelmi fejlesztést: az érzékszervi-mozgásos intelligenciát az első szakaszban; a szemléletes helyzethez kötött, szituatív intelligenciát a másodikban, s végül a konkrét műveletnek megfelelő értelmi működést a harmadik szakaszban. A fejlődés egész menetében egységes terméke és eszköze az értelmi fejlődésnek, gyakorlórendszere a mozgásnak, majd a képzetalkotásnak, s a szocializációnak.

Az explorációs játék

Az érzékszervi-mozgásos korszaknak, az élet első 18 hónapjának a játékait talán elsődlegesen az exploráció jellemzi. Valószínűleg nemcsak a szoros értelemben vett játéknak, hanem ebben az életkorban a gyerek egész viselkedésének ez a fő vonása. Azzal magyarázható, hogy új helyzetben való reakcióként jelenik meg az orientációs reflex mintájára. Válasz, amelyben a gyerek rámozdul a helyzet új elemére, érinteni próbálja, mint aki ismeretlennel ismerkedik. S ugyanez a rámozdulás folytatódik, amikor letaposgatja, fogdossa, forgatja a tárgyat.

Millar összefoglalása alapján az explorációhoz, a fürkésző ismerkedéshez vezető viselkedést mindenekelőtt az újnak a vonzereje vezérli. Ami új, ez önmagában is elegendő ennek a viselkedésnek a kiváltásához. De még inkább érvényesül ez az indíték, ha a változás

egyhangú, unalmas helyzetsorban következik be. Az unalomban (a változatlanságban) megjelenő új az, ami leginkább elindítja a fürkésző viselkedést.

„Rejtélyesnek” mondja Millar azt a gyereknél megfigyelhető jelenséget, hogy az ismétlődést éppen úgy szeretik, mint a változatosságot. Tudjuk, ez a meglepő tapasztalat már nem egy esetben kalandos elméletképzéshez is vezetett (Freud például többek közt az ismétlés magyarázatára iktatta be rendszerébe a „halálösztönt”). De az ismétlés preferenciája, mint jelenség, vitathatatlanul fennáll. Talán úgy tekinthetjük, mint az újra való rámozdulás kiegészítését: a megismétlés ismerősségével biztonságot nyújt a tájékozódásban. Az élet e korai szakaszában minden változás nagy változás, hiszen a tárgy, amit elrejtenek, olyan mintha nem is létezne többé, a mama, aki kimegy, olyan mintha örökre tűnne el. A változás rendkívül veszélyével szemben a változást meg-megszakító ismétlés biztonságot nyújt.

Az újnak, az érdekesnek a viselkedés-meghatározó jellegével szemben felhozható az az érv is, hogy a 11-14 hónapos kisgyerek óráig tud játszani egy dobozzal, egy spulnival, egy lánccal. Rég ráunt talán és még mindig játszik vele. Erre viszont Millar jegyzi meg, hogy egy ilyen egyszerű tárgy a kisgyereknek számunkra elképzelhetetlenül hosszú ideig ígér új és új változatokat: a lökés- és nyomáserőnek, a lendületnek a különféle szintjein végrehajtott cselekvésváltozásokat, a gesztusok beállításának, irányításának minden lehetséges szögben elképzelhető változatait stb.

Ez a következtetés fontos tételt implikál, amelyet Millar inkább jelez, mint kifejt: nem minden fürkésző és mozgásos viselkedés tekinthető játéknak. Az új tárgy félelmének a leküzdése még nem játék, jóllehet ez is történhet explorációs-manipulációs tevékenységgel. Ha az új tárgy félelmetes, szorongató, akkor az exploráció jó ideig ennek eloszlatására irányul, a tárggyal való megbarátkozásnak egy bizonyos fokán túl azonban ugyanazok a tevékenységi fordulatok már nemcsak a félelem eloszlatását célozzák, hanem az ismerkedés örömének a felkeltői.

Ahhoz, hogy egy tevékenységet játéknak mondjunk az élet bármely szintjén, kritériuma az is, hogy örömteli legyen, szórakoztató, az egyhangúságnak, az unalomnak az ellensúlyozását is magába foglalja.

Az exploratív játékmódnak addig van értelme, amíg a fürkésző, letapogató ismerkedés az életben való tájékozódás műveletrendszere. Amint a jelekkel való behelyettesítés általánossá válik, a szimbolikus funkció átszövi a magatartást, az exploratív viselkedés az alkalmazkodásnak jóllehet fontos, mégis rövid bevezető folyamatára szorítkozik, s ami még

lényegesebb, egyre inkább automatizálódik. Sokáig megmarad még ugyan játéknak, de távolról sem a legfontosabb játékformának.

Játékpszichológia

„Tanulják meg a számtant, és adjanak miniatűr szerszámokat azoknak a hároméves gyerekeknek, akikből építészeket akartak nevelni!”

Arisztotelész [*]is úgy gondolta, hogy olyan játékra kell a gyerekeket megtanítani, amit felnőttként komolyan csinálhatnak. A nagy reformer nevelők nyomán – tizenhetedik századbeli Comeniustól kezdve, a tizennyolcadik század végén, a tizenkilencedik század elején élő Rousseau-n, Pestalozzin át Fröbelig – a pedagógusok egyre inkább elfogadták azt a nézetet, hogy a nevelésnek számolnia kell a gyerekek természetes érdeklődésével és fejlettségi szintjével. Ez a nézet Fröbelnél érte el a tetőpontot, aki a tanulás folyamatában a játék kivételes jelentőségét emelte ki. Érdeklődése a gyermekek iránt boldogtalan ifjúságra vezethető vissza, Schelling romantikus filozófiája iránt érzett csodálata pedig a szabadság és az önkifejezés gondolatát szinte dogmává fejlesztette benne. A gyermekek iránt érzett rokonszenve és nevelői tapasztalata képessé vette arra a felismerésre, hogy azok a játékok, amelyeket a gyerekek élveznek, és azok játékszerek, amik vonzzák őket, felhasználhatók figyelmük lekötésére, valamint képességeik és tudásuk fejlesztésére. Gyakorlati szempontból igen értékesek voltak gondolatai. De a gyermeknevelésről Fröbel korában még keveset tudtak és az is rendszerezetlenül, így koncepciója, amely szerint a játék a „gyerekkorban sarjadozó rügyek kibontása”, nem ad megfelelő magyarázatot mai kérdéseinkre a játékkal kapcsolatosan.

Az első játékelméletek kialakítása a tizenkilencedik század közepére és végére maradt, amikor érezhetővé vált az evolúciós elmélet hatása. Eszerint az élőlények tevékenysége elsősorban a testi szükségletek kielégítésére, vagyis az életfenntartásra és a fajfenntartásra irányul, ilyen körülmények között milyen funkció tulajdonítható a játéknak? Eleinte egyszerűnek tűnő választ adtak: a játék célja a munka ellensúlyozása. Ez a gondolat, alkalomszerűen már egyes tizenhetedik századi írásokban is felmerült, általában azonban két tizenkilencedik századi német gondolkodó Schaller és Lazrus nevéhez fűződik, akik egy-egy könyvet írtak a játékról. Schaller szerint a játék a kimerülőfélben levő erőforrásokat táplálja

újra. Lazarus a játékot a munkával és a tétlenséggel állította szembe, a tevékeny pihenést frissítőnek tartotta.

Pihentethet egy futballmérkőzés, a célba dobás, akár a kertkapu festegetése is. Ez a tétel viszont nem áll a fiatal szervezetek játékára. Semmilyen fárasztó munka nem előzi meg azt a játékot, amelyben a kiscica a gomolyagra veti magát. A kölyökkutyák addig kergetőznek, amíg el nem fáradnak, azután elölről kezdik az egészet.

A JÁTÉPROGRAMOK

Ma a legnagyobb példányszámban eladott játékok többségében FPS (first person shooter, azaz saját szemszögű lövöldözős játék), szimulátorok (autó, repülőgép, stb.), sportjátékok (foci, kosárlabda, jégkorong), vagy kalandjátékok. Azonban ezek forráskódja igencsak komplex, így nem kivitelezhető bemutatásuk programozás órán. Éppen ezért az általam írt játékprogramok jellegüknél fogva beilleszthetőek egy emelt szintű érettségire készülő diák feladatai közé, vagy a szakmacsoportos informatikát tanuló diákok magas óraszámú számítógép-programozás elméleti vagy gyakorlati órán bemutatott programok közé. Ezen programok során elszakadnak az addig megszokott „készíts programot mely kiírja a…” típusú programoktól, ill. a nyilvántartó vagy statisztikai programok száraz világából, és belépnek azon világ kulisszái mögé, melyben a mai középiskolások – sőt, már általános iskolások, óvodások – többsége majd’ mindennap élnek napi több órát is.

Azonban mind a hat játékprogram logikai vagy logikai-ügyességi játék. A probléma vagy számítógép által adott (Memologik), vagy két, ember irányította játékos játszhatja egymással (Recogni). Felhasználói felületük általában grafikus, csupán az egyikük szöveges felületű.

Választásom során számos játékprogramot megnéztem Interneten, s azért döntöttem ezek mellett, mert amellett, hogy a játék használata fejleszti a diákok gondolkodásmódját, logikáját, addig programozói szemszögből nézve nem túl bonyolultak ahhoz, hogy oktatási célra használjuk fel ezeket.

Az első két játékprogram – a Mátrix és a Worms (ez utóbbi Kígyóként ismert Interneten) – részben ügyességi, részben logikai, taktikai játék. A Mátrix esetében egyszerre 3-4-5 ellenfél mozgására kell figyelnie a játékosnak, s szem előtt tartania a számára lehetséges kibúvókat jelentő csapdákat, illetve telefonfülkéket. A csapdába csalva az ellenfelet az meghal (ugyanígy a telefonfülke esetében is), míg a telefonfülke a játékos számára egyfajta lépéselőnyt jelenthet: lévén a fülkébe jutva a Mátrix egy másik, véletlenszerű pontjára kerül. Ezeket szem előtt tartva alakíthatja a soron következő lépéseit, s egyedüli túlélőként a Mátrixban, győzhet.

A Worms esetében csak egyetlen dolgot kell szem előtt tartania a játékosnak: a másik játékost. S annak megfelelően kialakítania taktikusan az útvonalát, hogy minél kisebb területre szorítsa be – persze saját magának minél nagyobbat fenntartva – ezáltal az ellenfele hamarabb fog „falba” ütközni. Habár ez így elsőre egyszerűbbnek tűnik, s kevésbé követeli

meg a játékos(ok)tól azt, hogy számos tényezővel számolva építsen fel egy taktikus útvonalat, ám míg a Mátrixnál egy-egy lépés során gyakorlatilag végtelen a rendelkezésre álló idő, addig itt egy jóval pörgősebb játékról van szó, hiszen a fal egyre közelebb van. Tehát míg a Mátrixnál több tényezővel kell számolni több rendelkezésre álló idő alatt, addig a Worms esetében csak egyetlen tényezővel, ám nagyságrendekkel kevesebb idő alatt, s mindezek felett még a rendelkezésre álló terület is egyre kevesebb!

A három korongjáték (Recogni, Discs, Amőba) pusztán logikai játék. Az Amőba és a Recogni gyakorlatilag azonos célú kétszemélyes játék, csupán a korong lehelyezésére vonatkozó szabályok különböznek. Azonban a Discs esetében már a korong(ok) felvétele a cél, és csupán egy egyszemélyes játékról van szó. Ezen túl a Discsben jó előre meg kell tervezni a korongok felvételét: nem feltétlenül célravezető mindig azon korongra való kattintás (felvétel), amelyiknek éppen abban a játékállásban van a legtöbb szomszédja. Hiszen megeshet, hogy ha abban a játékállásban egy kevesebb szomszédú korongot felszedve egy olyan állásba kerül a játékos, melyben egy korong jócskán több szomszédhoz jut, mint az előzőben a legtöbb szomszéddal rendelkező! S mivel a pontok hatványozódnak, így több pontot lehet elérni egy esetleges „áldozattal”. Így nyeri el ez a játék a logikai mivoltát.

A Memologik talán az, amelyik a legelgondolkodtatóbb a játékmenet során, vért is lehet izzadni, ahogy közeledik a mágikus 15-dik lépés, lévén annyi a rendelkezésre álló tippek száma: ha a játékos 15 tippen belül nem találja el a hat rendelkezésre álló szín közül azt a négyet (a megfelelő sorrendben), amelyek kiteszik a feladványt, veszített. S a segítsége csupán annyi, hogy látja az előző tippjeit, valamint azt, hogy egy adott tippjében hány szín szerepel a megoldásban, és hány darab a megfelelő helyen. A tapasztalatom során igencsak bosszankodtak a diákok, amikor nem sikerült kitatálniuk a véletlenszerű feladványt… őszintén szólva nem kis kárörömöt okozva nekem… De természetesen annak messze jobban örültem, amikor megfejtettek egy-egy feladványt, különösképpen amikor kiderült egy-egy gyenge tanulmányi eredménnyel bíró diákról, hogy megvan a logikai érzéke. Csupán csak azért gyenge a tanulmányi eredménye, mert van jobb dolga is, mint egy könyvet lapoznia, és megtanulnia az évszámokat, a Present Perfect Continous Tense-t. De miután már felfedeztük, hogy megvannak a képességei, azok után már csak annyi teendője a szaktanároknak, hogy ösztönözzék a diákot a tanulásra valamilyen figyelemfelkeltő, kíváncsiságot kiváltó feladattal. Mint például egy számára érdekes játékprogram programozói megvalósításával…

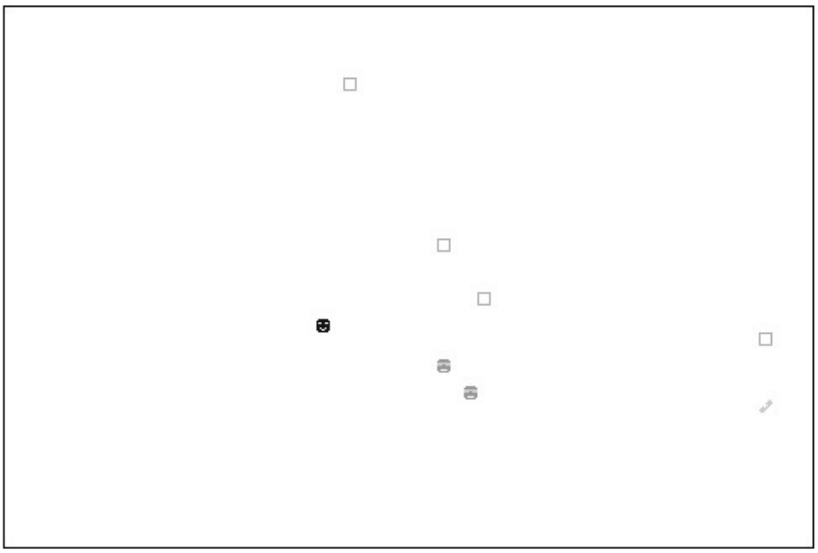
Mátrix

A gyerekeket leginkább érdeklő játékok azok, melyek *inkább ügyességi játékok*. Ebbe könnyen bele lehet csempészni egy kis logikai vonatkozást is. Ám még közelebb lehet vinni a gyerekekhez egy olyan játékot, amelynek van valamilyen vonatkozása egy azt megelőzően megjelent játékhoz vagy esetleg *film*hez.

Az első ilyen logikai játékprogram egy filmhez kötődik, mely főleg a fiatalabb korosztályt célozta meg merész ötletei miatt, melyeket sikerült beépíteni egy logikai játékba: ez a Mátrix.

A JÁTÉK CÉLJA

A filmben látottak alapján a Mátrix egy virtuális világ, melyben Ügynökök üldözik a „szabályokat nem betartók”-at. A játék lényege és célja megegyezik a filmben látottakkal: a játékosnak arra kell törekednie, hogy minél tovább legyen képes életben maradni a Mátrixban úgy, hogy közben az Ügynökök – ők a tulajdonképpeni ellenfelek – ne kapják el (ebben az esetben a játékos elveszíti a játékot). Íme egy játékrészlet:



Megtett lépések száma: 80

Természetesen a Mátrixot nem lehet elhagyni, határai vannak, és csak azokon belül mozoghatunk. Egy kis könnyítés is bekerült a játékba: hasonlóan a filmhez, telefonfülkék

vannak elhelyezve a Mátrixban, melyekbe besétálva a Mátrix egy másik, véletlenszerűen generált pontjára kerülünk, mintegy teleportként működve.

A JÁTÉK ELVI VÁZLATA

A játék magja egy mátrix, azaz egy kétdimenziós tömb. Ennek elemei logikai változók, mely azt takarja, hogy a mátrixnak azon pontján van-e ott valami. Például a mátrix határai – az első és utolsó sor ill. az első és utolsó oszlop – „igaz”-ra vannak definiálva: hiszen a határ van ott. Ha ebbe belesétálunk, elveszítjük a játékot (ugyanígy elveszítjük, ha csapdába lépünk, vagy egy Ügynök elkap minket). Természetesen az aknák, a telefonfülkék, és az Ügynökök, mind-mind teljesen véletlenszerűen generáltak a játszma kezdetekor.

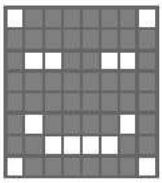
A Mi emberünk, az Ügynökök, a telefonfülkék és a csapdák viszont már rekord szerkezetűek: tartalmazza a mátrixbeli vízszintes és függőleges koordinátákat és a hozzátartozó színt.

Ezek után már csak le kell képezni mindezt a grafikus képernyőre, ami egyszerű matematikai művelet. Mivel a Mátrix egy 62 x 42 –es mátrix, míg a képernyő 640 x 480 –as felbontású (habár nincs kihasználva a teljes képernyő, mert egyetlen mező csupán egy 8x8-as terület a képernyőn), ezért a mátrixot némi szorzás és összeadás útján le lehet képezni grafikus képernyőre. Íme a játékos „képének” megjelenítése:

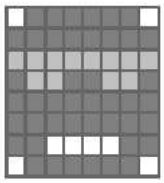
```
Procedure FaceU ( x , y , color : Byte ) ;
Begin
  Setcolor ( color ) ;
  Setfillstyle ( Solidfill , color ) ;
  Bar( x * w + hx , y * w + hy , x * w + hx + 7 , y * w + hy + 7 ) ;
  Putpixel ( x * w + hx , y * w + hy ,Black ) ;
  Putpixel ( x * w + hx + 7 , y * w + hy , Black ) ;
  . . .
End;
```

Először az átadott oszlop és sor koordinátákat kell megszorozni a játékos fejének képernyő-pixelben kifejezett szélességével. Ehhez hozzá kell adni a grafikus képernyőn való eltolást, mivel a megjelenített mátrix középen van elhelyezve.

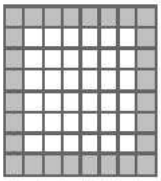
Persze a leképezés során figyelni kell arra, mit hogyan jelenítünk meg: más a jelképe a csapdáknak, telefonfülkéknek, Ügynököknek, és mi magunknak. Egy-egy ilyen „arckép” egy 8 x 8 –as terület a képernyőn. A következőképpen néznek ki kinagyítva a különféle alakzatok (a képek szürkeárnyalatos színképben vannak megjelenítve):



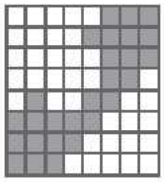
A Mi képünk



Az Ügynökök képe



Egy csapda képe



Egy telefonfülke képe

A megjelenített mátrix fordulóról fordulóra változik: ha lépünk egyet, az Ügynökök is lépnek, mégpedig a lehető legrövidebb utat megkeresve.

MOZGÁSKOORDINÁLÁS

A „Mi” mozgásunk nyolc irányban történhet: balra, jobbra, fel, le, és átlósan is négy irányban. Mindez a numerikus padon történik. Persze nem fontos lépnünk egy-egy fordulóban. Ha a nyolc billentyű között lévőt ütjük le (az **5** billentyűt), ugyanott maradunk, és az Ügynökök lépnek. Ezután persze meg kell vizsgálni, hova is léptünk:

```
For i:=1 To maxports Do
 If (pph[i].x=p.x) And (pph[i].y=p.y) Then
    Begin
     Repeat
      p.x:=Random(60);
      p.y:=Random(40);
      Until (p.x>0) And (p.x<61) And (p.y>0) And (p.y<41) And (matrix[p.x,p.y]=False);
      FaceU(p.x,p.y,p.c);
    End;
```

Pl. a fenti ciklus azt vizsgálja meg, beleléptünk-e *MAXPORTS*-ban megadott számú telefonok valamelyikébe (ennek értéke szintenként változik).

Az Ügynökök lépései szintén nyolc irányban történhetnek, de mindig a lehető legrövidebb úton történik. Ám miután lépnek, egy sor vizsgálat következik: Hova kerültek az Ügynökök? Ha oda lép, ahol Mi magunk állunk, akkor elkapott, és vége a játéknak. Ha egy csapdába lépett, meghal és így kiesik a játékból. Ha két Ügynök egy helyre lépett, az egyikük meghal, és egy csapda keletkezik a helyén, a másik viszont tovább léphet. Mindez persze rengeteg feltételt jelent a program forráskódjában, amely csak első látszatra tűnik ilyen

egyszerűnek. *(Ez utóbbinál meg lehetett volna tenni, hogy két Ügynök „ütközésekor” mindkettő meghaljon, azonban szerintem ez túlságosan megkönnyítené a játékot.)*

NEHÉZSÉGI SZINTEK

Minden nehézségi szinten változik a csapdák, telefonfülkék és Ügynökök száma. Az első szinthez képest a másodikon több Ügynök van, míg kevesebb telefonfülke és csapda. Viszont mivel a 3-4. nehézségi szinten az Ügynökök mozgási lehetőségeivel is nehezedik a játék (átlósan is képes lépni), ezért a második szintet képest a harmadikon több csapda és telefonfülke található. Azonban a negyedik szinten – a harmadikhoz képest – ismét kevesebb csapda és kevesebb fülke található. Íme:

Első szint (kezdő):	Ügynök – 3
	Csapda – 5
	Telefonfülke – 4
Második szint (haladó):	Ügynök – 5
	Csapda – 3
	Telefonfülke – 1
Harmadik szint (mester):	Ügynök – 3
	Csapda – 6
	Telefonfülke – 3
Negyedik szint (profi):	Ügynök – 3
	Csapda – 4
	Telefonfülke – 1

Mint már említettem, az Ügynökök a lehető legrövidebb úton közelítenek felénk. Ez a legrövidebb út nehézségi szinttől függ. Az első és második nehézségi szinten az Ügynökök többnyire csak le-fel ill. balra-jobbra tesznek lépéseket, kivételt képezve, ha a függőleges és vízszintes koordináták különbsége egyenlő (az Ügynökök és a Mi koordinátáink különbsége): ebben az esetben átlósan lép.

A harmadik és negyedik nehézségi szinten viszont már mindig átlós irányban fognak lépni, mindkét (a függőleges és vízszintes) irányú távolságot egyidejűleg csökkentve. A játék első-második, majd harmadik-negyedik nehézségi szinten történő kipróbálása után ez már nagyon érezhető különbség.

Persze a játék lényege nem az Ügynökök „lerázásán” van, hanem hogy csapdába csaljuk őket. *Így fel is merült a játék megkérdőjelezése: Mitől lesz logikai játék?*

Lépéseinket úgy kell megválasztani, hogy az Ügynökök követvén minket csapdába kerüljenek, vagy egymásnak ütközzenek. Először vizsgáljuk meg az elsőt: hogy kerülhet egy csapdába? Nézzünk meg ehhez egy helyzetet a játékból:

(Az általunk irányított figura a sötétebbik.) Ebből már látszik, hogy ha Mi lefelé haladunk, akkor az Ügynök követvén minket a legrövidebb útvonalon – azaz lefelé – pont belesétál egy csapdába. Ezáltal Ő meghal, és mivel nincs több Ügynök, ezért meg is nyerjük a játékot.

Vizsgáljuk meg azt is, mikor „ütközhetnek” egymásnak az Ügynökök. Ehhez is nézzünk meg egy helyzetet a játékból (*Melléklet, 1-3.ábra*).

Az első ábrán az látszik, hogy ha lefelé lépünk, akkor a felül lévő Ügynök is lefelé fog lépni, mivel számára az a legrövidebb út felénk. Azonban a másik – a jobboldali Ügynök – balra fog lépni (neki az jelenti a legrövidebb utat felénk). Ez pedig azt jelenti, hogy egy mezőre, azonos koordinátákra fognak lépni, és egyikük meghal (az, amelyik „utóbb” lép). Ez történt a második ábrán. A harmadikon már „eltávolodunk” egy kissé a létrejött csapdától. Viszont a 3-4. nehézségi szinten már más lenne a kialakuló helyzet, ugyanis ott a felénk vezető utat mindig átlósan teszik meg az Ügynökök, azaz egyszerre csökkentik mindkét, a vízszintes és a függőleges koordináták közti különbséget.

Tehát olyan ez a játék, mint a sakk: minden lépésünket végig kell gondolni, hogy milyen következményei lehetnek, hova kerülhetnek az Ügynökök, hogyan lehetne őket csapdába csalni. A játék akkor fog a győzelmünkkel véget érni, ha minden Ügynök meghal… biztonságban maradva a Mátrixban.

KIEGÉSZÍTŐK

Egy-egy számítógépes játék hangulatát feldobják a különféle kiegészítők. Ilyen például a már bemutatott nehézségi szintek, melyek lényege már bemutatásra került. Hiszen mindenki számára más-más élvezetet nyújtanak a különböző nehézségi szintek. Találja meg mindenki a saját magának megfelelő szintet!

Emellett még szerepel a játékban egy olyan lehetőség, mellyel színbeállításokat érhetünk el: a saját és az Ügynökök színét határozhatjuk meg egy színskáláról.

SZEREPE PROGRAMOZÁS ÓRÁN

A program egyik részét képezik azok a saját eljárások, melyek a különféle alakzatok kirajzolásáért felelősek; ilyen maga a játékos, az ügynökök, a telefonfülkék, és a csapdák.

A programozás egyik fontos eleme az eljárások, függvények. Ez a programozói eszköz szolgálja az újrafelhasználhatóságot – ugyanarra a szerepre, vagy egy kissé módosított szerepre, ez utóbbinál kell paraméterezni – és az áttekinthetőséget, ill. komolyabb programozói esetekben mint kizárólagos, egyszerű és elegáns megoldási módszerként szolgálnak. A középiskolai programozási órákon az eljárások és függvények többnyire az újrafelhasználhatóságot jelentik, ám ebben az esetben a program ezen része remek példaként szolgál az áttekinthetőségre: a játéktérben egy-egy, a képernyőn megjelenő objektum kirajzolásáért felelősek a FaceU/FaceA/FaceP/FaceT/FaceOff eljárások. Ezek kivitelezése ha eljárás nélkül valósul meg, a program forráskódja kevésbé áttekinthető, túlságosan „száraz”. Ezen túl – ám ugyanakkor erre épülve – a relatív koordinálás alapjaira is kiváló példaként szolgál.

Igen érdekes rész a grafikus felületen megjelenő játéktér, és a „nem látható, valós játéktér” összerendelése: lévén a logikai játéktér egy mindössze 62x42-es mezőből álló mátrix, míg a grafikus felületen egyetlen mező 8x8 pixel méretű, ennélfogva a grafikus felületen egy 488x328-as méretű területen lehet mozogni: a két játéktér összerendelése mindösszesen egy egyszerű matematikai művelet, ami szintén kiváló lehetőség arra, hogy a diákok észrevegyék: az amit a képernyőn látnak, az nem feltétlenül egyezik meg a valós kivitelezéssel, az csupán csak egy selyemköntös…

Egy másik oktatási célzatú programrész a billentyűzetről történő irányítás, azaz hogyan valósul meg a játéktérben való mozgás. Amennyiben a diákok már rendelkeznek a többirányú

elágazó utasítás fogalmával és ennek Pascalban való kivitelezésével, úgy nagyon egyszerűen bemutatható ezzel a programrésszel a legtöbb játék alapjául szolgáló „mozgás” megoldása. Hiszen a legtöbb száraz programozás órán az elágazó utasítás matematikai módszereknél alkalmazzák, itt azonban látványos hatása is van: a mozgás, a program valós használata.

Ehhez kapcsolódnak azok a feltételek, amelyek a játékos játéktérben tartásáért felelősek, azaz hogy a játékos ne hagyhassa el a játékteret; valamint azok a történések, melyek az egyes objektumok fedésekor jönnek létre (pl. a játékos a telefonfülkébe jut, csapdába lép, stb.), és természetesen a győzelem fogalma…

Worms

Ez a játék hasonlít a már igen ismert „kukacos” játékhoz, mely nemcsak a számítógépeken hódított a „kezdetekben”, hanem ma már megjelent a mobiltelefonokban is: Kígyó néven vált ismertté. Ez azonban egy picit különbözik tőle. A Kígyó játék pusztán ügyességi játék, és mindössze egyetlen játékos játszhatja, a Worms azonban ettől továbblép: két játékos játszhatja, valamint tekintettel arra, hogy taktikusan és logikusan kell felépíteni az utunkat – hogy minél kisebb területre zárjuk az ellenfelet – már egy kis logikai vonatkozás is észrevehető benne (persze azért még inkább ügyességi játéknak megmaradva).

A JÁTÉK CÉLJA

Mindkét játékos a saját kukacát (angolul WORMS, innen a játék neve) irányíthatja a pályán négy irányban: fel, le, balra, jobbra. A kukac nyomvonala megmarad, és ha abba bármely játékos belemegy, elveszíti a játékot (ui. a pályát sem szabad elhagyni). Ennél fogva úgy kell felépíteni az útvonalunkat, hogy a másik játékost sarokba tudjuk szorítani: minél kisebb területre behatárolni. Tehát az útvonalat **minél célszerűbben, logikusabban, de minél gyorsabban** kell megválasztani.

Persze előfordulhat az is, hogy mindkét játékos egyszerre fogy ki a pálya adta területből, vagy esetleg pontosan a fejükkel ütköznek egymásnak: ebben az esetben döntetlen lesz a játék eredménye.

A JÁTÉK ELVI VÁZLATA

Hasonlóan a Mátrix nevű játékhoz, itt is a játék magja egy mátrix. Ennek elemei logikai változók, melyek értéke igaz vagy hamis annak megfelelően, hogy lépett-e oda valamelyik játékos. Persze teljesen mindegy melyik játékos „járt már ott”, hiszen akár a másik, akár a saját nyomvonalunkba sétálunk bele, elveszítjük a játékot.

A mátrix elemeinek értéke – logikai típusú mátrix – induláskor hamisra vannak állítva, de kivételt képez a négy határoló vonal és a két kukac kezdeti pozíciója, mely pozíciók igazra vannak állítva. Ezek után már csak „le kell képezni” ezt a kétdimenziós tömböt a grafikus képernyőre, amely tulajdonképpen ugyanúgy történik, mint a Mátrix nevű játékban, ezért ezt már nem tartom szükségesnek újra felvázolni.

Maga a játék pedig tulajdonképpen egy hátultesztelő ciklus (*Repeat … Until*), ebbe van elhelyezve a játék magva, mely mindössze egy billentyűfigyelés Case utasításba foglalva és két feltétel a grafikai megjelenítésekkel (ez utóbbi egy pár soros eljárás). Tehát csupán néhány sor jelenti a tényleges játékot. A ciklus kilépési feltétele pedig a következő:

```
Until (bill=#27) Or death1 Or death2;
```

A *bill=#27* az **ESCAPE** billentyű leütésére vonatkozik: ha esetleg a játékot azonnal abba szeretnénk hagyni. A *death1* és *death2* pedig az egyes ill. kettes játékos halálát jelentő logikai változó. E három eset bármelyikének bekövetkeztekor a játéknak vége.

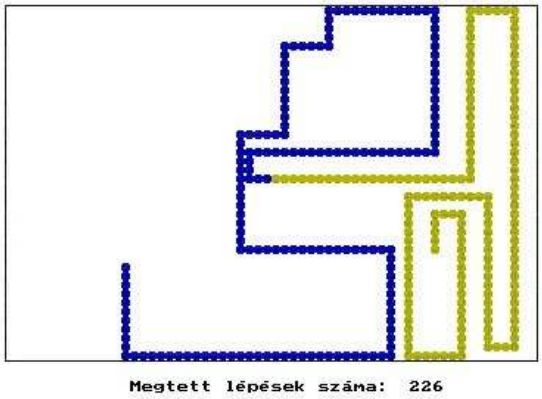
MOZGÁSKOORDINÁLÁS

A játék irányítása a billentyűzetről történik: az egyik játékos a kurzormozgató nyilakkal, míg a másik az ugyanígy elhelyezett **W-S-A-D** billentyűkkel.

Induláskor a két kukac ellentétes irányban indul el a pálya közepéről. A program egy ciklusban tulajdonképpen figyeli, történik-e billentyűleütés. Ha igen, megnézi melyik billentyű volt az, és annak megfelelően változtatja meg annak a kukacnak az irányát. Ezt az irányt persze el kell tárolni egy változóban (ez a *lastbill1* és *lastbill2*), hogy a kukac tartsa azt az irányt, amíg irányváltás nem történik. Ezen változóknak megfelelően módosul a kukacok fejének pozíciója:

```
Case lastbill1 Of
    'w'  : Dec(p1y);
    's'  : Inc(p1y);
    'a'  : Dec(p1x);
    'd'  : Inc(p1x);
End;
Case lastbill2 Of
    #72 : Dec(p2y);
    #80 : Inc(p2y);
    #75 : Dec(p2x);
    #77 : Inc(p2x);
End;
```

De hogyan tudjuk úgy felépíteni az útvonalunkat, hogy az ellenfelet sarokba szorítsuk, vagy legalábbis minél kisebb területre? Ennek a lényege, hogy próbáljuk meg minél szorosabban követni az ellenfelünket, de figyelve arra, hogy „ne vághasson elénk”, azaz ne tudjon elénk kerülni. Ha sikerült bezárni egy akkora területre, ami kisebb, mint amit mi magunk a hátunk mögött hagytunk, nyert ügyünk van: már csak húzni kell az időt, hiszen ő hamarabb „fogy el” a saját területéből, amibe bezártuk. Egy ilyen szituációt szemléltet az alábbi játékállás kimerevített képe:



Ezen látszik, hogy a sötét kukaccal lévő játékos sikeresen „csapdába zárta” a világos kukaccal lévő játékost, akinek így jóval kevesebb területe maradt. A sötét játékos tovább képes mozgásban maradni a pályán, így ő fogja megnyerni a játékot (hacsak el nem rontja azzal, hogy véletlenül „belerohan” a falba).

NEHÉZSÉGI SZINTEK

Mivel mindkét játékost billentyűzetről lehet irányítani, így a gépi intelligencia kimarad a programból. Azonban a két játékos mozgását még lehet nehezíteni: a kukacok mozgásának „sebességével”.

Mivel a játék csak egy hátultesztelő ciklus, így a nehézségi szintek kialakítása leegyszerűsödik. Az *Until* utasítás előtt (még a ciklusban) be van ágyazva egy *Delay(ms)* parancssor, melyben az *ms* változó értékét lehet változtatni a játék **Nehézségi szint** menüjében, ami az ezredmásodpercben megadott késleltetési idő. Ez jelenti a tulajdonképpeni mozgás sebességét.

KIEGÉSZÍTŐK

Ez egy olyan játék, mely a tulajdonképpeni egyszerűsége, de ugyanakkor a játék élménye miatt ragadja meg az embert (ill. emberpárost). Az egyszerűsége miatt nem is lehet igazán kiegészítőkkel ellátni. Az egyetlen, ami a játékban elérhető, az a kukacok színválasztása.

SZEREPE PROGRAMOZÁS ÓRÁN

Ez a játék tartalmazza azon programozói megoldást, amely a Mátrix c. játékban is megtalálható: a logikai játéktér, mely nem látható közvetlenül, ill. a valós, grafikus felületű játéktér összerendelése. Így ez a – már korábban bemutatott – módszer ennél a játékprogramnál tovább gyakoroltatható a diákokkal.

Azonban itt a kukacok mozgása már másképp valósul meg: folyamatos mozgás történik, még olyankor is, amikor nem történik billentyűleütés. Csupán akkor történik valami változás a játékmenetben, ha valamilyen billentyűleütés történik (ill. ütközéskor). Ugyebár – ezzel szemben – a Mátrix játék esetében folyamatos vezérlésre van szükség a játékos részéről, s ha nem történik billentyűleütés, úgy addig nem is történik semmi. Erre két megoldás is kínálkozik: az egyik szerint a megfelelő billentyű leütésekor egy nyilvántartott irány megváltozására kerül sor, míg a másik szerint a leütött billentyű eltárolása valósul meg egy „utoljára leütött billentyű” változóban. Itt a Worms játékban ez utóbbit építettem be a programba, lévén az első módszer egyszerűbb, azt a diákokkal közösen szépen fel lehet építeni. Ez a megoldás érdekesebb, ám nehezebb is. Természetesen mindkét megoldást célszerű felvezetni és bemutatni az órai munka során, és a kettő közötti különbségeket, előnyöket-hátrányokat megvitatni: minél több módszert ismernek meg a diákok, annál szélesebb látókörrel rendelkeznek, és annál fogékonyabbak lesznek arra, hogy ki is dolgozzák azokat, és képesek legyenek összehasonlítani azokat, és a legmegfelelőbbet kiválasztani.

A játéktér ebben a játékban egyszerűbb, mint a Mátrix esetében: itt mindösszesen egy logikai mátrix is elegendő, s ezáltal értékes memóriakapacitás takarítható meg. Habár a mai számítógépek korában – átlagosan 1 GB belső memóriával – teljesen érdektelen, hogy 6400 byte vagy 25600 byte, ami azért valljuk be, messze áll az akár egy-két milliárd bájtos belső memóriánktól; ám a diákoknak nem árt szem előtt tartani egy igen fontos programozói célt: egy programnak nem csupán az adott célt kell szolgálnia, hanem minél jobban kell gazdálkodnia az erőforrásokkal. Ha ezt a szempontot nem tanulja meg idejekorán, úgy egy komplex program esetében kudarcot vallhat, és a felhasználóktól csak panaszokat fog hallani, lsd. számos „nagynevű” cég „nagynevű” terméke. (Természetesen van még egy olyan programozói szempont is, hogy a programnak „bolondbiztos”-nak kell lennie, ám jelen esetben most az egyszerűségen volt a hangsúly.)

DISCs

A legelterjedtebb játékok a korongjátékok. Ezek valójában az egyszerűségük miatt ragadják meg az embert. Persze az is szerepet játszik, hogy a korongjátékok már jóval a számítógépek megjelenése előtt is elterjedtek voltak, ez alatt pedig nem néhány évet értünk, hanem több évszázadot! Ilyen régóta ismert korongjáték a malom, reverzi, stb.

Azonban a számítógépek megjelenésével még szélesebbre tárult a kapu a korongjátékok előtt. Ezeknek az egyik képviselője az általam csak **DISCs**-re keresztelt játék, a korong angol megfelelője alapján.

A JÁTÉK CÉLJA

Egy 12 x 8 –as táblán vannak elhelyezve korongok négyféle színben: piros, zöld, világoskék és sötétkék. Egy-egy színű korongot levéve a felette levők lejjebb csúsznak. Azonban ha leveszünk egy bizonyos színű korongot, akkor a mellette lévő azonos színű korongok is eltűnnek. Így akár egyetlen levétellel 2, 3, 4 … 10, vagy akár még több korong leszedhető; majd az ezek felett lévők lecsúsznak az üresen maradt helyekre. A cél: minél kevesebb levétellel leszedni az összesen 12 x 8 = 96 korongot.

A pontozás is ennek megfelelően alakul: egyetlen korongért 10 pont jár. Azonban két korongért már nem 2 x 10 = 20 pont jár, hanem 30 pont! S ez alapján ugyanígy 10 korongért nem 10 x 10 =100 pont jár, hanem 550 pont! Íme a pontozás:

1 korong - 10 pont	10 korong - 550 pont
2 korong - 30 pont	11 korong - 660 pont
3 korong - 60 pont	12 korong - 780 pont
4 korong - 100 pont	13 korong - 910 pont
5 korong - 150 pont	14 korong - 1050 pont
6 korong - 210 pont	15 korong - 1200 pont
7 korong - 280 pont	16 korong - 1360 pont
8 korong - 360 pont	17 korong - 1530 pont
9 korong - 450 pont	18 korong - 1610 pont

Azaz levéve egy korongot ad érte 10 pontot, ha talál egy azonos színű szomszédot, akkor azért a „második” korongért már 2x10 azaz 20 pontot, ha van egy harmadik szomszéd, azért a „harmadik” korongért már 3x10 azaz 30 pontot ad, és így tovább. Úgy is lehet fogalmazni, hogy $\sum_{i=1}^{db} i \times 10$, ahol a **db**-nak megfelel a **pointsadd** változó, **i**-nek a **k**

ciklusváltozó. Ezen pontozási módszerért felel a következő rész (`pointsadd` a levett korongok száma):

```
For k:=1 To pointsadd Do Inc(points,k*10);
```

Nézzünk meg ehhez két játékállást, amely mutatja egy levétel előtti és utáni állapotot (**Melléklet, 4-5. ábra**). Ha megfigyeljük a pontállást, érthető a játék a pontozási módszere: a lila korongra leszedve (ahol az egérmutató áll), 15 korongot lehet levenni. Erre „alapesetben” mindössze 15 x 10 = 150 pontot kellene kapni, azonban a kialakított pontozásnak megfelelően 1200 pontot kapunk rá. Egyetlen levétellel eltávolított 15 korongért!

De miért logikai játék ez a táblás korongjáték? A válasz a játék céljában rejlik. Ha csak „vaktában” elkezdjük leszedni a korongokat, akkor általában csak 2000-3000 pontot lehet elérni (gondoljunk bele: ha egyesével szedjük le a korongokat, az csak 96 x 10 = 960 pont). De ha figyelünk arra, hogy egy korongcsoport leszedésekor milyen helyzetbe fognak kerülni a felette lévők – hiszen azok lecsúsznak az üresen maradt helyekre – akkor ki tudjuk választani a legkedvezőbb lépést. Fontos tehát a logika és kalkuláció a játék folyamán. Egy kis logikával és némi kalkulációval a pontjaink száma elérheti akár a 6000-t is!

A JÁTÉK ELVI VÁZLATA

A játék egy 12 x 8 –as mátrixra épül. Ennek elemei rekordok, amelyek színkódokat (a négyféle szín egyikét) és a grafikus *x-y* koordinátákat tárolják. A tényleges játék egy ciklusba van ágyazva, melynek kilépési feltétele, ha a korongok elfogynak (vagy egy **ESCAPE** billentyűt ütünk).

Az az ellenőrzés, hogy a mellette lévő azonos színűeket eltávolítsa, annak a kidolgozásának a legegyszerűbb és egyben legelegánsabb módja egy **rekurzív eljárás**. Mindig ellenőrzésre kerülnek a felette-alatta-balra-jobbra lévő korongok, majd az ezek mellett elhelyezkedők is. Ez az eljárás tehát nem más, mint egy négy feltételes (a négy irány miatt) rekurzív eljárás, mindössze pár sorban. Az eljárás *CHECK* néven szerepel a programlistában.

Miután a korongok lekerültek a tábláról, a felette lévőknek le kell csúszniuk az üresen maradt helyekre. Ennek kivitelezése szintén úgy a legegyszerűbb, ha **rekurzív eljárás**sal oldjuk meg. Ennek elve annyiban különbözik, hogy csak egyetlen feltételt kell benne megszabni, miszerint lefelé van-e még üres maradt hely? Ezután már csak az adott korongot kell eltávolítani, és letenni az új helyére.

NEHÉZSÉGI SZINTEK

A játék elve miatt nehézségi szintek alkotása nem kivitelezhető. Azt hiszem elegendő nehézséget jelent az, hogy mivel a tábla véletlenszerűen van generálva, így gyakorlatilag nem találkozunk két egyforma táblával.

SZEREPE PROGRAMOZÁS ÓRÁN

Ezen játékprogram kivitelezése egy, az eddigiektől eltérő mozgatást, és egy még érdekesebb ellenőrzést tartalmaz.

A mozgatás jelensége talán nem is szembetűnő, csak ha belegondolunk a következőbe: amint leszedünk egy korongot és az eltűnik – valamint a vele négyszomszédos azonos színű korongok, és az azokkal is négyszomszédos azonos színű korongok, és így tovább – a helyükön maradó űrbe lepotyognak az űr felett lévő korongok. Ezen mozgatás annyiban tér el az előző játékoknál megtapasztaltakkal, hogy a „leesésnek” mindaddig kell folytatódnia, amíg lejjebb már nem tud esni. Ez azért érdekes, mert a Mátrixnál és a kukacos játéknál mindaddig történt a mozgás-mozgatás, amíg bele nem ütközött valamilyen akadályba a játékos, akkor azonban „meghalt”.

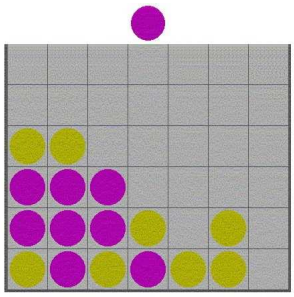
Azonban a játék igazi szerepe egy olyan programozói eszköz, ami a középiskolákban nem igazán jelenik meg, ám egy szakkörön, amennyiben jó képességű diákokból áll a csoport, úgy felfedeztethető a rekurzivitás fogalma a diákokkal, és remekül be is mutatható ezen programrésszel! A rekurzivitás, mint tudjuk, igen hatékony és egyszerű programozói eszköz. Jelen esetben ott lép be a programba, amikor le kell ellenőrizni egy korongot, hogy a szomszédai azonos színűek-e, mert akkor azokat a korongokat is le kell venni, ám természetesen azoknak is tovább kell vizsgálni a szomszédjait, s mindaddig folytatni az ellenőrzést, amíg a színazonosság fennáll a szomszédoknál.

Recogni

Egy másik ismert táblás korongjáték azonban már számítógép nélkül is játszható, ez a játék már jóval a számítógépek megjelenése előtt létezett. Egy kicsit hasonlít a mindenki által csak amőbának nevezett játékhoz, melyben két játékos játszva egymás ellen megpróbál egymás mellett – vízszintesen, függőlegesen vagy átlósan – a saját alakzatából (ill. színéből) ötöt kirakni. Ez az „öt” persze csak a megállapodáson múlik, hiszen lehet az akár három is.

A JÁTÉK CÉLJA

Én ebben a játékban azt a bizonyos öt egymás mellett lévő korongot négyre csökkentettem. Azonban nem csak ennyiben különbözik ez a játék az amőbától. Az amőba során a mátrixban bárhová le lehet helyezni a saját alakzatunkat (a továbbiakban inkább a korong szót használom, mert ebben a játékban a két játékos különböző színű korongokkal játszik). Viszont ebben a játékban úgy történik a korong lehelyezése, mintha az aljára állított táblába felülről csúsztatnánk be a korongot, és az lecsúszik a legaljára. Persze ha abban az oszlopban alul már van egy másik korong, akkor annak a tetejére esik rá. Íme egy szemléltető kép a játékból:



Ebből az is látszik, hogy ha a sötét színű koronggal játszó játékos abba az oszlopba teszi le a korongot, ahol éppen a mutató áll, meg is nyeri a játszmát, mivel vízszintesen sikerül kiraknia négy korongot. Tehát az a játékos nyer, akinek sikerül a saját színű korongjaiból négyet egymás mellé helyezni (vízszintesen, függőlegesen vagy átlósan).

A JÁTÉK ELVI VÁZLATA

Ennek a táblás korongjátéknak is az alapja egy mátrix, mely 7 x 6 –os dimenziójú, és elemei a két játékos színeinek megfelelő színkód, mely a *P1* és *P2* konstansban van tárolva,

ill. 0, amennyiben az adott elem helyén nincs korong. A játék pedig egy hátultesztelő ciklus (*Repeat…Until*), melynek kilépési feltétele a *theend* változó, melynek értéke ha nem nulla, vége a játéknak (ennek szerepére később visszatérek). A ciklus nagyon egyszerű: fordulónként változtatja az aktuális játékost (színt), s annak megfelelően hajt végre két eljárást: az egyik a korong lehelyezése, a másik a tábla ellenőrzése.

A korong lehelyezése a táblába a tetején lévő korong pozíciójának megfelelően történik. Amely oszlop felett áll a korong, abba az oszlopba helyezi le a korongot alulra a le-kurzormozgató billentyű hatására.

A tábla ellenőrzése egy többszörös feltétel. Ennek oka a játék célja: vízszintesen, függőlegesen, jobbra-le átlósan, vagy balra-le átlósan van-e négy azonos színű korong egymás mellett. Ez négy darab két-két egymásba ágyazott *For* ciklus, melyben egy harmadik van a tényleges ellenőrzésre. Ha a *counter* változó értéke eléri a négyet, az adott színű játékos nyert.

Ez alapján már látszik, hogy a program kidolgozása igen egyszerűen valósult meg.

MOZGÁSKOORDINÁLÁS

A játék irányítása végképp egyszerű: a balra-jobbra kurzormozgató nyilakkal lehet a tábla tetején lévő koronggal a megfelelő oszlop felé mozogni, és a le-kurzormozgató billentyű hatására a korongot lehelyezni – egy egyszerű billentyűleütés-figyelés.

SZEREPE PROGRAMOZÁS ÓRÁN

A program egyik része már megtalálható az előzőben tárgyalt Discs játékban, mégpedig a korong „leesése”, lévén mindaddig kell zuhannia az adott korongnak, amíg el nem éri a tábla alját, ill. az abban az oszlopban lévő legfelső korongot. Így a programnak ez a része ismétlésként felhasználható, folytatva az előző Discs játék tanórai felhasználását.

Itt azonban egy újabb programozói módszerrel ismerkedhetnek meg a programozópalánták, ez pedig a tábla ellenőrzése. Erre kétféle lehetőség van: az egyik szerint a teljes tábla átvizsgálása megtörténik – hogy van-e valahol négy szomszédos korong egyvonalban – míg a másik módszer szerint csupán a legutoljára lerakott korongból kiindulva a négy irányban – függőlegesen, vízszintesen, és a két átló vonalában – megtalálható-e négy korong egyvonalban. Én az előbbi módszert építettem be a programba, habár az utóbbi a hatékonyabb, egyszerűbb, és rövidebb. Ugyanis szerintem a diákok számára a felfedezés

öröme jelenti a leghatékonyabb módszert a tanulás terén, ugyanakkor fenntartja az érdeklődést is a téma iránt; valamint rájönnek arra is, hogy olykor érdemes újabb és újabb algoritmusok után kutatniuk, annak ellenére hogy találtak egy működőt.

Miután megértették a „teljes tábla ellenőrzési algoritmust”, utána könnyedén rávezethetőek az egyszerűbb ellenőrzésre, s ennek megfelelően az már kiadható otthon megoldandó feladatnak, házi feladatnak. Lévén csak egy adott programrészt kell módosítaniuk.

Memologik

Kevesek által ismert játék a soron következő korongjáték, mert a leírtak alapján kissé nehezen érthető; inkább a játék tényleges bemutatása után lesz igazán érthető. Ez egyike azon játékoknak, mely igazán megdolgoztatja az embert: egyike a legnehezebb és legkomolyabb logikai játékoknak.

Habár ez a játék nem igényel számítógépes játékprogramot a kivitelezéséhez, hiszen ez létezik egyszerű „papír” formájú változatban is, ennek ellenére az én ismeretségi körömben nem találkozott még senki ezzel a játékkal.

A JÁTÉK CÉLJA

Hatféle színben vannak korongok, ebből csak négy szerepel a játékban egymás mellett kirakva. Ezek persze véletlenszerűen vannak generálva a játék kezdetekor, és ezt a négy korongot kell a helyes sorrendben elhelyezkedve kitalálni 15 lépésben

Minden egyes tipp után az kiértékelésre kerül, és a számítógép megmondja, hogy a tipped megadott színű korongok közül mennyi van megfelelő színnel a megfelelő helyen, és azt is, hogy mennyi van megfelelő színnel, de rossz helyen. Az előbbit fekete körök számával mutatja meg, míg az utóbbit fehér körök számával mutatja meg.

A játék megértését könnyíti, ha bemutatok egy játékállást (**Melléklet, 6. ábra**). Ebben a legutóbbi tipp szerint a négy színű korongból három szín a megfelelő helyen van. Az azt megelőző két tipp szerint pedig a kék-szürke-lila (3-0-2) az. Az utolsó helyen pedig biztos, hogy nem fekete, narancs vagy világoskék van (5, 1, 4): az csakis kék, szürke vagy lila lehet (3, 0, 2). Azonban kék (3-as) nem lehet, ennek oka pedig a következő: az első tippben a kék (3-as) az utolsó helyen van, és egy fekete kör jelzi, hogy a négyből egy jó helyen van. De biztos, hogy nem a kék (3-as), mert a második tippben is ott van az utolsó helyen, de annak az ellenőrzésében nincs fekete kör. Így csak szürke vagy lila (0, 2) lehet. Azt viszont ebben a szituációban csak megtippelni lehet, mert előzőleg egyiket sem helyeztük az utolsó pozícióba (egyébként a megoldás mint kiderült, kék-szürke-lila-lila (3-0-2-2) volt).

A játék során kialakulhat egy-egy stratégia a játékostól függően. Célszerű az első egy vagy két lépésben csak két-két színt szerepeltetve megadni a tippeket, pusztán csak azért, hogy tudjuk, mely színek szerepelnek a sorozatban. Ez után jöhet a sorrend kialakítása.

Ha azonban a játékban szeretnénk már rögtön a megfelelő helyekre is koncentrálni, akkor már az első lépésben 4 különböző színt kell szerepeltetni. Időben és lépésben gyorsabb módszer, azonban összetettsége miatt komoly fejtörést okoz: komolyabb stratégiát igényel, és precízebb analitius gondolkodást.

A JÁTÉK ELVI VÁZLATA

A játék magja itt is egy mátrix, mely a játék futtatásakor látszik, hogy 15 sorból áll, azonban az oszlopainak száma a könnyebb kezelhetőség miatt nem négy (habár ennyi korong van a megoldásban), hanem tíz, mivel belevettem még az ellenőrzést is.

A játék egy egyszerű *Repeat …Until* ciklus, melynek kilépési feltétele, ha meghaladjuk az adott lépésszámot (15 a megengedett tippjeink száma), vagy ha kitaláltuk a sort (ill. ha leütjük az **ESCAPE** billentyűt, kilépés miatt). A ciklus előtt persze a „szokásos előkészítések”: a mátrix nullázása, a rejtvény véletlenszerű generálása, kirajzolása.

A ciklusban minden egyes billentyűleütés során az annak megfelelő színnel rajzolja azt a korongot, majd a sor végén ellenőrzésre kerül: előbb a „megfelelő színű korong a megfelelő helyen”, ami egy *For* ciklus során végzett ellenőrzés, majd a „megfelelő színű korong, de rossz helyen”, ami ugyanúgy értékelődik ki, mint az előző. A problémát nem is ez a kettős ellenőrzés jelenti – végül is mindkettő egy-egy három soros *For* ciklus előtte néhány feltétellel – hanem inkább a billentyűk kezelése, ami első ránézésre elég összetettnek tűnik. Ugyanis ezt nem csak a szokásos CRT unit-beli eljárásokkal és függvényekkel oldottam meg, hanem egyszerűbbnek tűnt, ha inkább néhány saját eljárásban és függvényben foglalom össze.

Egy apró könnyítés pedig bekerült a játékba a „kezdők” számára, amely a játék leírásában (amely a főmenüből érhető el) is szerepel: ha esetleg a játékos feladná a játékot – képtelen azt megfejteni – akkor az **M** billentyű hatására a megoldást megjeleníti a legfelső – nulladik – sorban.

SZEREPE PROGRAMOZÁS ÓRÁN

A programban felbukkan egy olyan típus, amelyet a középiskolai órákon ritkán vesznek elő a pedagógusok, ez a felsorolás típus. Habár egyszerű a használata, ám mégis célszerű csak az utolsó évre hagyni az oktatását a diákok számára, hisz a legtöbb probléma megoldható a beépített típusokkal, legyen az egyszerű, összetett, vagy mutató típus. Ez a program azonban

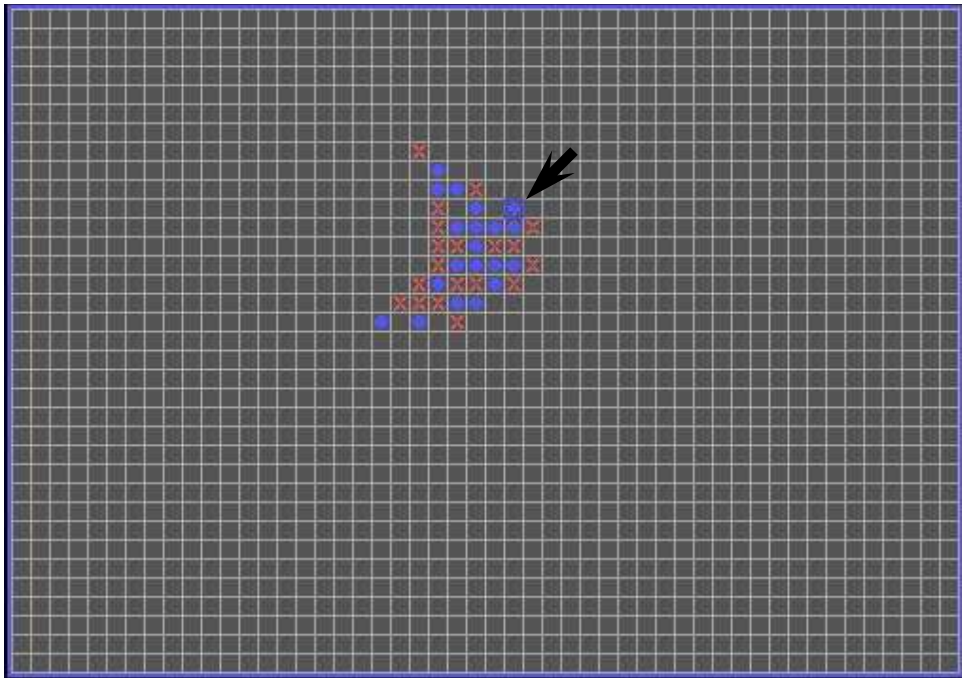
érdekeségképpen bevezeti a felsorolás típust, s ezáltal a diákok elkezdhetnek vele ismerkedni: ez a típus ebben a programban a lehetséges színeket képviselik, számok formájában.

Egy másik jellegzetessége a programnak, hogy több helyen használ logikai típusú változót, legyen az egy függvény (vagy eljárás) paramétere, vagy a függvény (vagy eljárás) típusa, vagy csupán egy egyszerű változó. Ezek főleg a megoldás kezelését oldják meg, valamint a „cheat”-et, azaz a megoldás megmutatását, vagy az ellenőrzés során felhasznált változóknál jut szerephez.

Az eddigi programozás óráim során a diákok roppant mód élvezték azt, amikor egy bejelentkezési képernyőt készítettek: a begépelt jelszó karakterei helyett rendre egy-egy csillag jelenik meg. Szintén hasonló helyzet a grafikus képernyőn történő billentyűzetről való beolvasás, ami karakterenként oldható meg. Ezek egyedi módon megoldható problémák, melyet akár saját függvényként is meg lehet írni. Egy ilyen beolvasás van ebben a programban is, amely figyel arra, hogy mely billentyűket fogadja csak el, illetve hogy a megfelelő színnel jelenik meg a beolvasott szín. Erről egy saját függvény gondoskodik, mely mindezekre ügyel.

Amőba

Egy mindenki által ismert játék, melyet már az alsós általános iskolások is ismernek, az az amőba. Így ez egy olyan logikai játék, mely mellett nem lehet csak úgy „elmenni” szó nélkül. Mivel egy széles körben és igen jól ismert játékról van szó, ezért úgy érzem, nem szükséges részletesen bemutatni. Íme egy megnyert játszma képe *(az utolsó lehelyezett alakzat a kék kör)* **:**



A JÁTÉK CÉLJA

Felváltva teszik le a játékosok a saját színű alakzatukat. A játék során arra kell törekedniük, hogy egymás mellé sikerüljön 5-öt (a sajátjukból) elhelyezni, legyen az vízszintesen, függőlegesen vagy átlósan. Akinek sikerül, az nyeri meg a játékot.

A JÁTÉK ELVI VÁZLATA

A játék magja itt is egy mátrix, melynek elemei háromféle értéket vehetnek fel: a kétféle szín kódjának valamelyikét, ill. ha nincs ott semmi, akkor a nullát. Az egyetlen körülményes dolog a játék megvalósításában az alakzatok ellenőrzése. Ám ez sem több, mint a négy

iránynak megfelelően (vízszintes, függőleges, és a két átlós irány) négy *For* ciklus, bennük egy feltételes *For* ciklussal, mely számolja az egymás mellett lévő azonos alakzatokat. Ha az eléri az ötöt, akkor az a játékos nyert.

A játék magja tulajdonképpen megegyezik a **Recogni** magjával (s hasonlít a Discs-hez is, bár ott a játék célja kissé eltér). Csupán néhány módosításra volt szükség, minthogy itt az alakzatokat nem „le”-venni kell a tábláról, hanem „fel”-helyezni. Éppen ezért nem is részletezem a program kivitelezését, mert többnyire megegyezik a **DISCs**-ben leírtakkal.

Emellett a program forráskódja – annak ellenére, hogy az elve nem a legegyszerűbb a hat játékprogram közül – a legrövidebbre sikerült. Ez csupán az **Amőba** játékszabályainak egyszerűségén múlik, hiszen valljuk be, ehhez a játékhoz nincs szükség számítógépre, csak egy négyzetrácsos papírlapra, és egy ceruzára.

Azonban ma, a számítógépek széleskörű elterjedése a legegyszerűbb játékokat is felkarolja: a legegyszerűbb, „papíron játszhatóakat”, és a táblás korongjátékokat is.

SZEREPE PROGRAMOZÁS ÓRÁN

Habár az amőba egészen más játék az előzőekhez képest – kivéve a Recognit, amiben a korongok letétele szigorúbb szabályokhoz kötött – ám programozói szemszögből az előző játékprogramok után már nem mutat semmi újat. A korong letétele még egyszerűbb mint a Recogni esetében, az ellenőrzés algoritmusa pedig teljes mértékben megegyezik az ott bemutatottal. Ezen kívül az Amőba szerepe a programozás órákon végső soron megegyezik a Recognival, így ezen program nem is iskolai feladatnak felel meg, hanem a egy otthoni feladatnak, vagy egy összefoglaló, ismétlő feladatnak egy tanítási órán. **Több** tanítási órán. Hisz egy 2x45 perces programozás órába nem fér bele…

ÖSSZEFOGLALÁS

„Készíts programot, mely egy üdvözlő szöveget ír ki a képernyőre, majd vár egy billentyűleütésre!” Valahogy így kezdődött az első programozásgyakorlat órám, amit 10. évfolyamos informatika szakos diákoknak tartottam. Természetesen ezt megelőzte néhány olyan tanítási óra – mind programozáselméleti, mind programozásgyakorlati – melyen a programozás alapjait tanulták meg a diákok: a programozás elemei, utasítások, változók, kifejezések. Természetesen élvezték, s mint azt a televízióban oly sokat hallhattuk: „igen, megcsináltuk”. Azonban egyre inkább olyan programok születtek ezeken a gyakorlati órákon, melyek – feltételes utasításokra és ciklusokra támaszkodva – a legtöbb jóindulattal sem nevezhetőek látványos programnak. „Készíts programot, mely kíírja a tökéletes számokat”. S ezzel kezdetét vette a gyakorlati órák azon időszaka, mely során egyre inkább unalmasnak találták a programozást. A helyzet határozottan javulni látszott, amikor grafikus képernyőn vette kezdetét a ciklusok, feltételek, képletek, kifejezések felhasználása. Látványos programok születtek. Azonban újra megtört a jég, amikor képbe kerültek az eljárások és függvények. Sokáig törtem a fejem, hogy hogyan nyerjék vissza a diákok az érdeklődésüket a programozás iránt, hogy hogyan élvezhetnék újra a programok alkotását, míg végül arra jutottam, hogy mi lenne ha olyan programot készítenénk, melynek futása során a felhasználó részévé válik a programnak – a program teljes futása során! S erre a legkézenfekvőbb megoldás a játékprogramok világa. Adottak a szabályok, amikhez programkódot kell készíteni, és jöhet a még élvezetesebb tesztelés… Sőt, a szabályokkal tetszőlegesen játszadozhat, s újra tesztelheti a változásokat. Előjön a grafikus felület használata, számítások, képletek, kifejezések, eljárások, függvények, paraméterek, ciklusok, egymásba ágyazott ciklusok, feltételek, feltételrendszerek, rekurzivitás, s természetesen – hogy a „rekordokat” vagy a beállításokat nyilván lehessen tartani, a fájlkezelés. Olyan programkódok kivitelezése során mélyítik el a programozás alapjai során tanultakat, melyek komplexitásuk mellett még látványosak, élvezetesek is: játékok. S mindemellett: ügyességi, stratégiai, logikai játékokról van szó, melyek fejlesztik is a diákok ezen képességeit. Hazamegy az iskolából, s amikor a szülei megkérdezik, hogy „Milyen napod volt?”, akkor a válasz a megszokott „Unalmas, főleg a másfél órányi programozás” helyett az lesz: „Nagyon LooooL, gyere nézd meg milyen zsííír játékot készítettünk, hazahoztam pendrive-on”. S nem ez lenne a célunk? Hisz mit lehet hallani mostanság a híradókban? Tanárverés itt, tanárverés ott. Akár Alsóbögyörővalagpusztá-

ról, Mucsaröcsögéről, vagy egy nagyvárosról van szó, tele van a tévé ehhez hasonló hírekkel. (S természetesen az Internetről már hamarabb tájékozódik róla az ember.) Az én – talán naív? – meglátásom szerint minden fél hibás. Hibás a diák, hisz ő tette. Hibás a szülő, hisz ő nevelte. Hibás a környezete, hisz ott nő fel. Hibás a pedagógus, hisz hagyta, hogy idáig fajuljon a helyzet. Ez utóbbi esetben azért, mert nyilvánvalóan nem egyik pillanatról a másikra lett olyan a diák, hogy „megtette", hanem egy hosszas folyamatról van szó, s már idejében észre lehet venni hogy gond van vele. Vagy azért hibás a pedagógus, mert nem tudja lekötni a diákot. Hiszen ha a diák „ott van az órán", ha aktív résztvevője s nem csupán egy lekötetlen szemlélő aki „az órai rend zavarásával" tölti ki szabad 45 percét, akkor nem fog olyat tenni, ami a híradóba kerül. A mai diákok ingerküszöbe pedig mára igen magas lett. Tehet erről a tévé, az Internet, a környezete, s persze tehet a szülő. Ilyen magas ingerküszöb esetén pedig komolyabb, látványosabb, érdekesebb, izgalmasabb eszközökre van szükség a tanítás során, mely a programozásgyakorlati órák során egy-egy olyan program lehet, amivel nap-mint-nap találkozhat az otthoni számítógépén vagy az Interneten: a játékok világa ez. Ha elmegy a haverjához, aki éppen „kiakasztja" az egyik kedvenc internetes játékprogramját, nem azzal fog felvágni, hogy „én akkor is jobb vagyok", hanem azzal, hogy „én ezt el is készítem neked bármikor". A sikerélményt nem a játék fogja adni, hanem a játék programkódjának elkészítése.

KÖSZÖNETNYILVÁNÍTÁS

Az ember egész életén át tanul. Éppen emiatt igen hosszú lenne az a lista, amelyen fel kellene sorolnom mindazon neveket, akiknek megköszönöm a segítséget a záródolgozatom megszületéséhez. Ám természetesen csak azokat említem meg, akik a leginkább hozzájárultak segítségükkel munkámhoz.

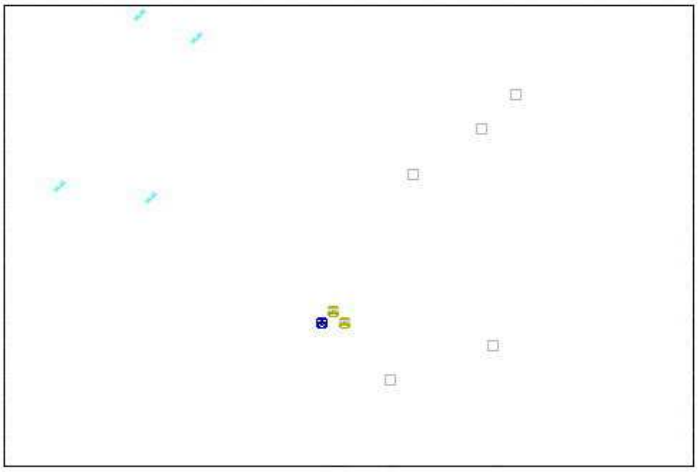
Ezúton köszönöm záródolgozat-vezető tanáromnak, Dr. Nagy Benedek egyetemi adjunktusnak, hogy elvállalta ezt a szerepet, segítette munkámat, tanácsokkal látott el, ráébresztett a kijavítandó hibákra, vagy bármilyen módon segítette a dolgozatom elkészültét.

Köszönöm kollégáimnak a Zay Anna Gimnázium, Egészségügyi Szakközépiskola és Kollégiumban, Asztalos Viktor és Király József tanár uraknak, hogy ötleteikkel hozzájárultak a programok kiválasztásához, és folyamatos unszolásukkal elkészültem a dolgozattal.

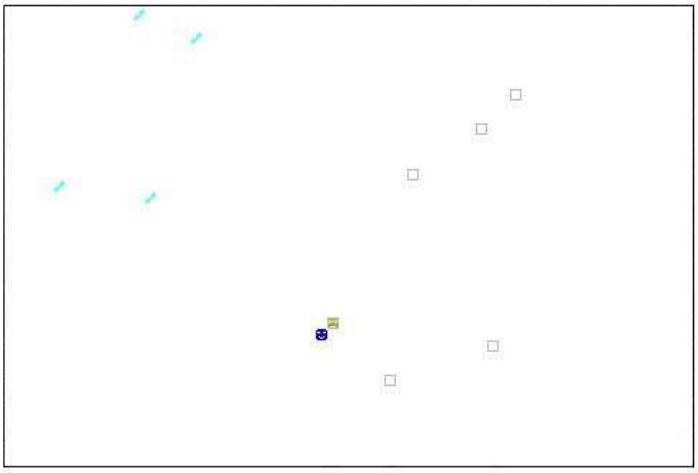
Valamint köszönöm diákjaimnak és volt diákjaimnak, hogy órai részvételükkel, és elsősorban véleményezésükkel és aktív munkájukkal ezt a témát választottam a záródolgozatomnak, és a programok tesztelésével elkészült azok végleges változata. A legfontosabb, hogy mindezekkel folyamatos eszközfejlesztésre ösztönöznek…

MELLÉKLET

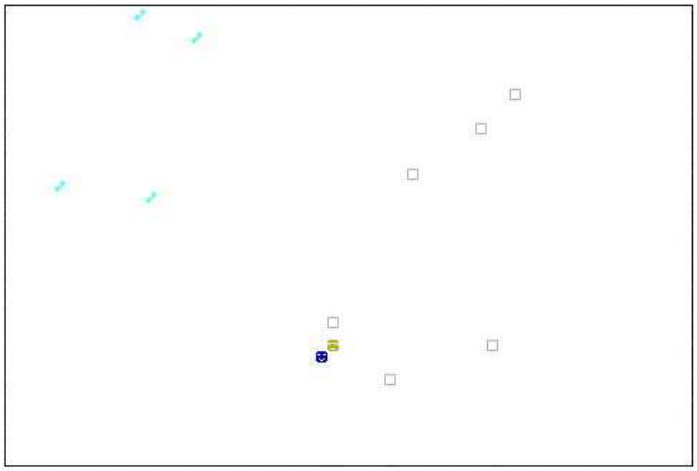
Ábrák



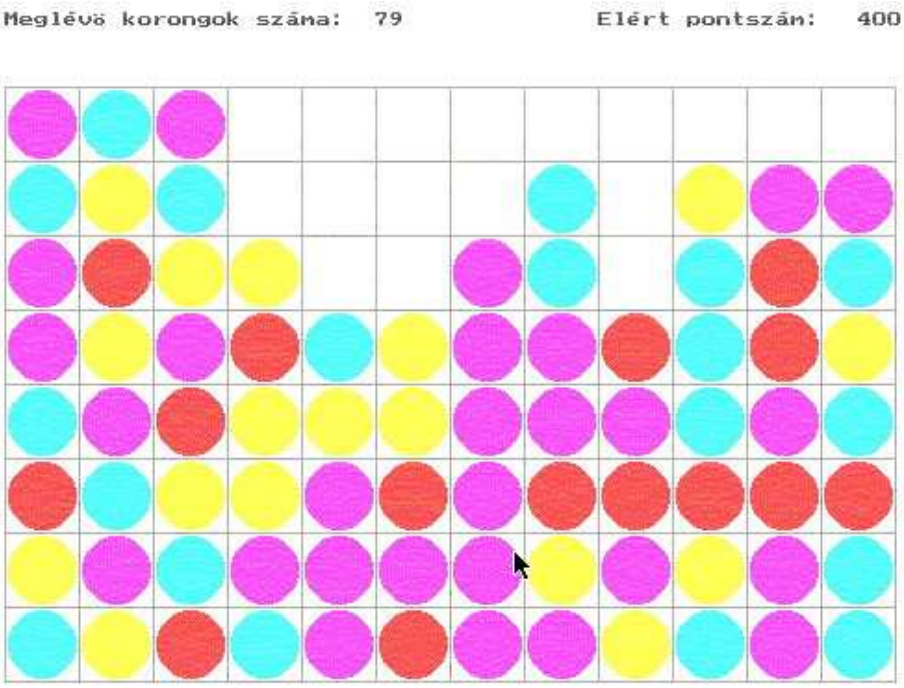
1. ábra



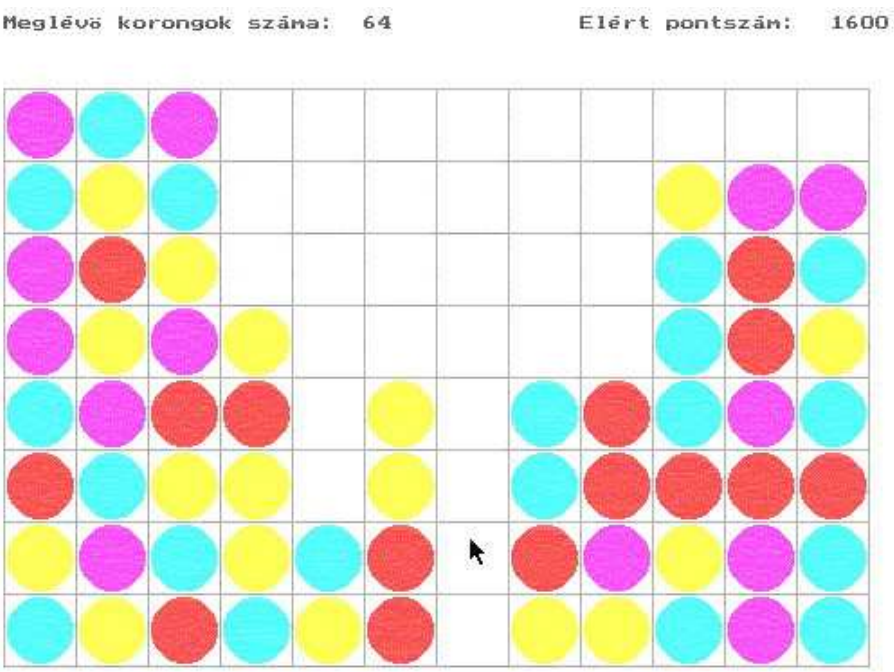
2. ábra


Megtett lépések száma: 29

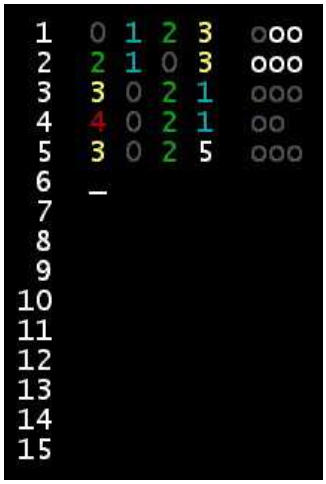
3. ábra


Meglévő korongok száma: 79
Elért pontszám: 400

4. ábra


Meglévö korongok száma: 64
Elért pontszám: 1600

5. ábra


1 0 1 2 3 ooo
2 2 1 0 3 ooo
3 3 0 2 1 ooo
4 4 0 2 1 oo
5 3 0 2 5 ooo
6
7
8
9
10
11
12
13
14
15

6. ábra

Forráskódok

MÁTRIX

```
PROGRAM Matrix;
USES Crt,Graph,Wincrt;
{ nehezsegi szinttol fuggetlenul a max. ertekek: }
CONST a=6;      { ugynokok }
      traps=6;  { csapdak }
      ports=4;  { telefonfulkek }
TYPE man=Record { a jatekterben levo ugynokok, csapdak, telefonfulkek }
       x,y,c:Byte; { pozicio, szin }
       d:Boolean;  { eletben van-e }
     End;
VAR bill:Char;
    quit,mi:Boolean;
    gd,gm,code:Integer;
    { nehezsegi szintnek megfelelo max. ertekek }
    maxtraps,maxports,maxa,lev:Byte;
    i,j:Word;
    p:Man;                    { a jatekos }
    pa:Array[1..a] Of Man;      { ugynokok }
    ptr:Array[1..traps] of Man; { csapdak }
    pph:Array[1..ports] of Man; { telefonfulkek }

 { menu kiirasa }
 PROCEDURE Menuout;
 BEGIN
   Setcolor(Lightgreen);
   Settextstyle(Triplexfont,Horizdir,1);
   Setusercharsize(4,1,1,1);
   Outtextxy(130,60,'Mátrix');
   Settextstyle(Defaultfont,Horizdir,1);
   Setusercharsize(1,1,1,1);
   Outtextxy(250,180,'1 - Betöltés a Mátrixba');
   Outtextxy(250,200,'2 - Nehézségi szint');
   Outtextxy(250,220,'3 - Szín');
   Outtextxy(250,240,'4 - A Mátrix szabályai');
   Outtextxy(250,260,'5 - Kilépés');
 END;

 { bevezeto }
 PROCEDURE Intro;
 BEGIN
   Delay(1000);
   Cleardevice;
   Setrgbpalette(3,0,0,0); Setcolor(3);
   Setfillstyle(Solidfill,3);
   Setbkcolor(3);
   For i:=0 To 63 Do Begin
                       Setrgbpalette(3,0,i,0);
                       Setbkcolor(3);
                       Delay(25);
                     End;
   Setrgbpalette(5,0,63,0); Setcolor(5);
```

```
  Menuout;
  Delay(200);
  For i:=63 Downto 0 Do Begin
                  Setrgbpalette(3,0,i,0);
                  Delay(25);
                 End;
END;

{ kivezeto }
PROCEDURE Outro;
BEGIN
  For i:=0 To 63 Do Begin
                  Setrgbpalette(0,0,i,0);
                  Delay(25);
                 End;
  Setcolor(Lightgreen); Setfillstyle(Solidfill,Lightgreen);
  Bar(0,0,640,480);
  Setrgbpalette(Black,0,0,0);
  Setcolor(Black);
  For i:=0 To 238 Do Begin
                   Line(0,i,640,i);
                   Line(0,480-i,640,480-i);
                   Delay(3);
                  End;
  Delay(500);
  For i:=0 To 318 Do Begin
                   Line(i,0,i,480);
                   Line(640-i,0,640-i,480);
                   Delay(3);
                  End;
  Delay(500);
END;

{ kepernyotorles }
PROCEDURE Erase;
VAR r:Word;
BEGIN
 Setcolor(Black);
 For r:=0 To 320 Do Begin
                   Rectangle(r,r,640-r,480-r);
                   Delay(4);
                  End;
END;

{ a jatek }
PROCEDURE Game;
CONST sizex=61; { a jatekter meretei }
      sizey=41;
      w=8;      { objektumok merete }
      hx=70;
      hy=20;
VAR stepped:Boolean;
    pox,poy,agents:Byte;
    steps:Word;
    stepss:String[4];
    matrix:Array[0..sizex,0..sizey] Of Boolean; { a jatekter }

  { objektum torlese }
```

```
  Procedure FaceOff(x,y:Byte);
  Begin
    Setcolor(Black);
    Setfillstyle(Solidfill,Black);
    Bar(x*w+hx,y*w+hy,x*w+hx+7,y*w+hy+7);
  End;
  { a jatekos kirajzolasa }
  Procedure FaceU(x,y,color:Byte);
  Begin
    Setcolor(color);
    Setfillstyle(Solidfill,color);
    Bar(x*w+hx,y*w+hy,x*w+hx+7,y*w+hy+7);
    Putpixel(x*w+hx,y*w+hy,Black); Putpixel(x*w+hx+7,y*w+hy,Black);
    Putpixel(x*w+hx,y*w+hy+7,Black); Putpixel(x*w+hx+7,y*w+hy+7,Black);
    Putpixel(x*w+hx+1,y*w+hy+2,Black); Putpixel(x*w+hx+2,y*w+hy+2,Black);
    Putpixel(x*w+hx+6,y*w+hy+2,Black); Putpixel(x*w+hx+5,y*w+hy+2,Black);
    Putpixel(x*w+hx+2,y*w+hy+5,Black); Putpixel(x*w+hx+3,y*w+hy+6,Black);
    Putpixel(x*w+hx+4,y*w+hy+6,Black); Putpixel(x*w+hx+5,y*w+hy+5,Black);
  End;
  { az ugynok kirajzolasa }
  Procedure FaceA(x,y,color:Byte);
  Begin
    Setcolor(color);
    Setfillstyle(Solidfill,color);
    Bar(x*w+hx,y*w+hy,x*w+hx+7,y*w+hy+7);
    Putpixel(x*w+hx,y*w+hy,Black); Putpixel(x*w+hx+7,y*w+hy,Black);
    Putpixel(x*w+hx,y*w+hy+7,Black); Putpixel(x*w+hx+7,y*w+hy+7,Black);
    Setrgbpalette(Cyan,12,12,12);
    Setcolor(Cyan);
    Line(x*w+hx,y*w+hy+2,x*w+hx+7,y*w+hy+2);
    Line(x*w+hx,y*w+hy+3,x*w+hx+2,y*w+hy+3);
    Line(x*w+hx+5,y*w+hy+3,x*w+hx+7,y*w+hy+3);
    Putpixel(x*w+hx+2,y*w+hy+6,Black); Putpixel(x*w+hx+3,y*w+hy+6,Black);
    Putpixel(x*w+hx+4,y*w+hy+6,Black); Putpixel(x*w+hx+5,y*w+hy+6,Black);
  End;
  { csapda kirajzolasa }
  Procedure FaceT(x,y,color:Byte);
  Begin
    Setcolor(color);
    Setfillstyle(Solidfill,color);
    Rectangle(x*w+hx,y*w+hy,x*w+hx+7,y*w+hy+7);
  End;
  Procedure FaceP(x,y,color:Byte);
  { telefonfulke kirajzolosa }
  Begin
    Putpixel(x*w+hx,y*w+hy+6,color);

Putpixel(x*w+hx+1,y*w+hy+5,color);Putpixel(x*w+hx+1,y*w+hy+6,color);Putpixel(x*w+hx+1,y*w+hy+7,col
or);
    Putpixel(x*w+hx+2,y*w+hy+4,color);Putpixel(x*w+hx+2,y*w+hy+5,color);
    Putpixel(x*w+hx+2,y*w+hy+6,color);Putpixel(x*w+hx+2,y*w+hy+7,color);
    Putpixel(x*w+hx+3,y*w+hy+5,color);Putpixel(x*w+hx+3,y*w+hy+6,color);

Putpixel(x*w+hx+4,y*w+hy+2,color);Putpixel(x*w+hx+4,y*w+hy+4,color);Putpixel(x*w+hx+4,y*w+hy+5,col
or);

Putpixel(x*w+hx+5,y*w+hy+1,color);Putpixel(x*w+hx+5,y*w+hy+2,color);Putpixel(x*w+hx+5,y*w+hy+3,col
or);
```

```
    Putpixel(x*w+hx+5,y*w+hy+4,color);
    Putpixel(x*w+hx+6,y*w+hy,color);Putpixel(x*w+hx+6,y*w+hy+1,color);
    Putpixel(x*w+hx+6,y*w+hy+2,color);Putpixel(x*w+hx+6,y*w+hy+3,color);
    Putpixel(x*w+hx+7,y*w+hy+1,color);Putpixel(x*w+hx+7,y*w+hy+2,color);
   End;

 BEGIN
  { jatekter generalasa }
  For i:=0 To sizex Do For j:=0 To sizey Do matrix[i,j]:=False;
  For i:=0 To sizex Do matrix[i,0]:=True;
  For i:=0 To sizex Do matrix[i,sizey]:=True;
  For i:=0 To sizey Do matrix[0,i]:=True;
  For i:=0 To sizey Do matrix[sizex,i]:=True;
  Erase;
  { jatekter kirajzolasa }
  Setcolor(White);
  Rectangle(0*w+hx+5,0*w+hy+5,sizex*w+hx,sizey*w+hy);
  { csapdak generalasa }
  For i:=1 To maxtraps Do
    Begin
      Repeat
        ptr[i].x:=Random(60);
        ptr[i].y:=Random(40);
        ptr[i].d:=False;
        ptr[i].c:=8;
      Until (ptr[i].x>0) And (ptr[i].x<61) And (ptr[i].y>0) And (ptr[i].y<41) And (matrix[ptr[i].x,ptr[i].y]=False);
      matrix[ptr[i].x,ptr[i].y]:=True;
      FaceT(ptr[i].x,ptr[i].y,ptr[i].c);
    End;
  { telefonfulkek generalasa }
  For i:=1 To maxports Do
    Begin
      Repeat
        pph[i].x:=Random(60);
        pph[i].y:=Random(40);
        pph[i].d:=False;
        pph[i].c:=4;
      Until (pph[i].x>0) And (pph[i].x<61) And (pph[i].y>0) And (pph[i].y<41) And
(matrix[pph[i].x,pph[i].y]=False);
      matrix[pph[i].x,pph[i].y]:=True;
      FaceP(pph[i].x,pph[i].y,pph[i].c);
    End;
  { a jatekos generalasa }
  Repeat
    p.x:=Random(60);
    p.y:=Random(40);
  Until (p.x>0) And (p.x<61) And (p.y>0) And (p.y<41) And (matrix[p.x,p.y]=False);
  matrix[p.x,p.y]:=True;
  FaceU(p.x,p.y,p.c);
  p.d:=False;
  pox:=p.x;
  poy:=p.y;
  { a nehezsegi szintnek megfeleloen az ugynokok generalasa }
  agents:=maxa;
  For i:=1 To maxa Do
    Begin
      Repeat
        pa[i].x:=Random(60);
```

```
      pa[i].y:=Random(40);
      pa[i].d:=False;
    Until (pa[i].x>0) And (pa[i].x<61) And (pa[i].y>0) And (pa[i].y<41) And (matrix[pa[i].x,pa[i].y]=False);
    matrix[pa[i].x,pa[i].y]:=True;
    FaceA(pa[i].x,pa[i].y,pa[i].c);
  End;
steps:=0;
{ jatek, a fordulok }
Repeat
  bill:=Readkey;
  { mozgatas }
  If bill=#0 Then Begin
                  bill:=Readkey;
                  Case bill Of
                    #72 : Begin
                          poy:=p.y;
                          pox:=p.x;
                          Dec(p.y);
                        End;
                    #80 : Begin
                          poy:=p.y;
                          pox:=p.x;
                          Inc(p.y);
                        End;
                    #75 : Begin
                          pox:=p.x;
                          poy:=p.y;
                          Dec(p.x);
                        End;
                    #77 : Begin
                          pox:=p.x;
                          poy:=p.y;
                          Inc(p.x);
                        End;
                    #71 : Begin
                          poy:=p.y;
                          pox:=p.x;
                          Dec(p.y);
                          Dec(p.x);
                        End;
                    #73 : Begin
                          poy:=p.y;
                          pox:=p.x;
                          Inc(p.x);
                          Dec(p.y);
                        End;
                    #79 : Begin
                          pox:=p.x;
                          poy:=p.y;
                          Dec(p.x);
                          Inc(p.y);
                        End;
                    #81 : Begin
                          pox:=p.x;
                          poy:=p.y;
                          Inc(p.x);
                          Inc(p.y);
                        End;
```

```
              End;
            End
          Else Begin
               pox:=p.x;
               poy:=p.y;
             End;
matrix[pox,poy]:=False;
FaceOff(pox,poy);
{ telefonfulkebe valo esetleges lepes ellenorzese }
For i:=1 To maxports Do
  If (pph[i].x=p.x) And (pph[i].y=p.y) Then
    Begin
      Repeat
        p.x:=Random(60);
        p.y:=Random(40);
      Until (p.x>0) And (p.x<61) And (p.y>0) And (p.y<41) And (matrix[p.x,p.y]=False);
      FaceU(p.x,p.y,p.c);
    End;
{ csapdaba valo lepes vagy Ugynokkel valo talalkozas ellenorzese -> halal }
If matrix[p.x,p.y]=True Then p.d:=True
                Else Begin
                       matrix[p.x,p.y]:=True;
                       FaceU(p.x,p.y,p.c);
                     End;
{ Ugynokok lepese a jatekos fele kozelitve 1 mezovel }
For i:=1 To maxa Do
  If pa[i].d=False Then
    Begin
      matrix[pa[i].x,pa[i].y]:=False;
      FaceOff(pa[i].x,pa[i].y);
      stepped:=False;
      If (Abs(pa[i].x-p.x)>Abs(pa[i].y-p.y)) And (mi=False) Then
          Begin
            If pa[i].x>p.x Then Dec(pa[i].x)
                        Else Inc(pa[i].x);
            stepped:=True;
          End;
      If (Abs(pa[i].x-p.x)<Abs(pa[i].y-p.y)) And (stepped=False) And (mi=False) Then
          Begin
            If pa[i].y>p.y Then Dec(pa[i].y)
                        Else Inc(pa[i].y);
            stepped:=True;
          End;
      If ((Abs(pa[i].x-p.x)=Abs(pa[i].y-p.y)) And (stepped=False)) Or (mi=True) Then
       If pa[i].x>p.x Then
                  If pa[i].y>p.y Then Begin
                                        If pa[i].x>1 Then Dec(pa[i].x);
                                        If pa[i].y>1 Then Dec(pa[i].y);
                                      End
                               Else Begin
                                        If pa[i].x>1 Then Dec(pa[i].x);
                                        If pa[i].y<(sizey-1) Then Inc(pa[i].y);
                                      End
                Else
                  If pa[i].y>p.y Then Begin
                                        If pa[i].x<(sizex-1) Then Inc(pa[i].x);
                                        If pa[i].y>1 Then Dec(pa[i].y);
                                      End
```

```
                                Else Begin
                                      If pa[i].x<(sizex-1) Then Inc(pa[i].x);
                                      If pa[i].y<(sizey-1) Then Inc(pa[i].y);
                                    End;
        If (pa[i].x=p.x) And (pa[i].y=p.y) Then p.d:=True
          Else
            If matrix[pa[i].x,pa[i].y]=True Then Begin
                                      pa[i].d:=True;
                                      Dec(agents);
                                      matrix[pa[i].x,pa[i].y]:=True;
                                      FaceT(pa[i].x,pa[i].y,8);
                                    End
                                  Else Begin
                                      matrix[pa[i].x,pa[i].y]:=True;
                                      FaceA(pa[i].x,pa[i].y,pa[i].c);
                                      For j:=1 To (i-1) Do
                                        If (pa[i].x=pa[j].x) And (pa[i].y=pa[j].y) Then
                                          Begin
                                            pa[j].d:=True;
                                            pa[i].d:=True;
                                            Dec(agents,2);
                                            matrix[pa[i].x,pa[i].y]:=True;
                                            FaceT(pa[i].x,pa[i].y,8);
                                            matrix[pa[j].x,pa[j].y]:=True;
                                            FaceT(pa[j].x,pa[j].y,8);
                                          End;
                                    End;
      End
    Else Begin
          matrix[pa[i].x,pa[i].y]:=True;
          FaceT(pa[i].x,pa[i].y,8);
        End;
  { a sikeres fordulo elkonyvelese }
  If p.d=False Then Begin
                  Inc(steps);
                  Setcolor(White);
                  Outtextxy(240,430,'Megtett lépések száma:');
                  Setcolor(Black); Outtextxy(430,430,stepss);
                  Str(steps,stepss);
                  Setcolor(White); Outtextxy(430,430,stepss);
                End;
Until (bill=#27) Or (p.d=True) Or (agents=0);
{ halal eseten a jatek elvesztese }
If p.d Then Begin
            FaceOff(p.x,p.y);
            Setcolor(pa[1].c);
            Outtextxy(285,410,'Meghaltál !');
            For i:=1 To sizex-1 Do Begin
                            For j:=1 To sizey-1 Do Begin
                                            Setcolor(pa[1].c);
                                            Rectangle(i*w+hx,j*w+hy,i*w+hx+7,j*w+hy+7);
                                            Delay(1);
                                          End;
                            Delay(1);
                          End;
          End;
{ az Ugynokok halala eseten gyozelem }
If (agents=0) Then Begin
```

```
                Setcolor(p.c);
                Outtextxy(285,410,'Nyertél !');
                For i:=1 To sizex-1 Do Begin
                                For j:=1 To sizey-1 Do Begin
                                                FaceOff(i,j);
                                                FaceU(p.x,p.y,p.c);
                                               End;
                                Delay(1);
                               End;
              End;
 If bill<>#27 Then bill:=Readkey;
 Erase;
END;

{ szinvalasztas }
PROCEDURE Colors;
VAR billb:Byte;
BEGIN
 Erase;
 Setcolor(White);
 Outtextxy(130,120,'A Te színed');
 Setcolor(14);  Outtextxy(120,160,'1 - Sárga');
 Setcolor(9);   Outtextxy(120,180,'2 - Kék');
 Setcolor(1);   Outtextxy(120,200,'3 - Sötétkék');
 Setcolor(4);   Outtextxy(120,220,'4 - Vörös');
 Setcolor(10);  Outtextxy(120,240,'5 - Zöld');
 Setcolor(2);   Outtextxy(120,260,'6 - Sötétzöld');
 Setcolor(12);  Outtextxy(120,280,'7 - Piros');
 Setcolor(11);  Outtextxy(120,300,'8 - Világoskék');
 Setcolor(15);  Outtextxy(120,320,'9 - Fehér');
 bill:=Readkey;
 Val(bill,billb,code);
 Case billb Of
   1 : p.c:=14;
   2 : p.c:=9;
   3 : p.c:=1;
   4 : p.c:=4;
   5 : p.c:=10;
   6 : p.c:=2;
   7 : p.c:=12;
   8 : p.c:=11;
   9 : p.c:=15;
 End;
 Setcolor(p.c);
 Outtextxy(130,360,'Rendben.');
 Setcolor(White);
 Outtextxy(410,120,'Az ügynökök színe');
 Setcolor(14);  Outtextxy(390,160,'1 - Sárga');
 Setcolor(9);   Outtextxy(390,180,'2 - Kék');
 Setcolor(1);   Outtextxy(390,200,'3 - Sötétkék');
 Setcolor(4);   Outtextxy(390,220,'4 - Vörös');
 Setcolor(10);  Outtextxy(390,240,'5 - Zöld');
 Setcolor(2);   Outtextxy(390,260,'6 - Sötétzöld');
 Setcolor(12);  Outtextxy(390,280,'7 - Piros');
 Setcolor(11);  Outtextxy(390,300,'8 - Világoskék');
 Setcolor(15);  Outtextxy(390,320,'9 - Fehér');
 bill:=Readkey;
 Val(bill,billb,code);
```

```
 Case billb Of
   1 : For i:=1 To a Do pa[i].c:=14;
   2 : For i:=1 To a Do pa[i].c:=9;
   3 : For i:=1 To a Do pa[i].c:=1;
   4 : For i:=1 To a Do pa[i].c:=4;
   5 : For i:=1 To a Do pa[i].c:=10;
   6 : For i:=1 To a Do pa[i].c:=2;
   7 : For i:=1 To a Do pa[i].c:=12;
   8 : For i:=1 To a Do pa[i].c:=11;
   9 : For i:=1 To a Do pa[i].c:=15;
 End;
 Setcolor(pa[1].c);
 Outtextxy(410,360,'Rendben.');
 Delay(500);
 Erase;
END;

{ nehezsegi szint valasztasa }
PROCEDURE Levels;
BEGIN
 Erase;
 Setcolor(Lightgreen);
 Outtextxy(260,180,'1 - Kezdö');
 Outtextxy(260,200,'2 - Haladó');
 Outtextxy(260,220,'3 - Mester');
 Outtextxy(260,240,'4 - Profi');
 bill:=Readkey;
 Case bill Of
   '1' : Begin
           maxtraps:=5;
           maxports:=4;
           maxa:=3;
           mi:=False;
           lev:=1;
         End;
   '2' : Begin
           maxtraps:=3;
           maxports:=1;
           maxa:=5;
           mi:=False;
           lev:=2;
         End;
   '3' : Begin
           maxtraps:=6;
           maxports:=3;
           maxa:=3;
           mi:=True;
           lev:=3;
         End;
   '4' : Begin
           maxtraps:=4;
           maxports:=1;
           maxa:=3;
           mi:=True;
           lev:=4;
         End;
 End;
 Erase;
```

```
 END;

 { jatekleiras }
 PROCEDURE Description;
 BEGIN
  Erase;
  Setcolor(Lightgray);
    Outtextxy(265,10,'Játékleírás'); Line(265,20,355,20);
    Outtextxy(290,455,'Fömenü');     Line(290,465,335,465);
    Outtextxy(100,60,'Irányítás'); Line(100,70,170,70);
      Setcolor(p.c);
        Outtextxy(40,100,'Te:');
        Outtextxy(60,140,'Home');Outtextxy(130,140,Chr(24));Outtextxy(175,140,'PgUp');
        Outtextxy(80,190,Chr(27));Outtextxy(180,190,Chr(26));
        Outtextxy(60,240,'End');Outtextxy(130,240,Chr(25));Outtextxy(175,240,'PgDn');
        Setcolor(Lightgray);
        Outtextxy(50,150,'Bal-Fel');Outtextxy(125,150,'Fel');Outtextxy(160,150,'Jobb-Fel');
        Outtextxy(65,200,'Balra');Outtextxy(165,200,'Jobbra');
        Outtextxy(50,250,'Bal-Le');Outtextxy(125,250,'Le');Outtextxy(160,250,'Jobb-Le');
      Setcolor(pa[1].c);
        Outtextxy(40,320,'Ügynökök:');
        Setcolor(Lightgray);
        Outtextxy(70,340,'A Mátrix irányítja');
    Outtextxy(400,60,'A játék célja'); Line(400,70,502,70);
      Outtextxy(270,100,'  Most Te játszhatsz a Mátrixban az ügynökök');
      Outtextxy(270,120,'ellen! A játék célja, hogy az ügynökök meg -');
      Outtextxy(270,140,'haljanak, ill. minél tovább maradjunk életben');
      Outtextxy(270,160,'a Mátrixban.');
      Outtextxy(270,180,'  Egy ügynök meghal, ha egy másik ügynökbe');
      Outtextxy(270,200,'ütközik ( de csak az egyik hal meg, és azon');
      Outtextxy(270,220,'a ponton egy új csapda jön létre ! ), vagy');
      Outtextxy(270,240,'ha egy csapdába ill. telefonfülkébe ütközik.');
      Outtextxy(270,260,'  Ha Te ütközöl egy csapdába, természetesen Te');
      Outtextxy(270,280,'is meghalsz. De ha egy telefonfülkébe sikerül');
      Outtextxy(270,300,'menekülnöd, a Mátrix egy másik pontjára jutsz,');
      Outtextxy(270,320,'idöt nyerve ezzel az ügynökökkel szemben.');
      Outtextxy(270,340,'  A Mátrixot persze nem hagyhatod el, mert');
      Outtextxy(270,360,'abban az esetben szintén meghalsz.');
      Outtextxy(270,380,'  Csak akkor nyerhetsz, ha egyedül maradsz');
      Outtextxy(270,400,'életben a Mátrixban !');
  bill:=Readkey;
  Erase;
 END;

{ foprogram }
BEGIN
 Randomize;
 { alapertekek egy gyors alapjatekhoz }
 maxtraps:=3;
 maxports:=1;
 maxa:=5;
 mi:=False;
 lev:=2;
 p.c:=14;
 For i:=1 To a Do pa[i].c:=9;
 gd:=9;
 gm:=2;
 Initgraph(gd,gm,'');
```

```
 Delay(1000);
 Intro;
 Repeat
   Cleardevice;
   Menuout;
   bill:=Readkey;
   Case bill Of
     '1' : Game;
     '2' : Levels;
     '3' : Colors;
     '4' : Description;
     '5' : quit:=True;
   End;
 Until quit;
 Outro;
 Closegraph;
END.
```

WORMS

```
PROGRAM Worms;
USES Crt,Graph,WinCrt;
VAR bill:Char;
    quit:Boolean;
    ms,p1c,p2c:Byte;
    gd,gm,code:Integer;

 { menu kiirasa }
 PROCEDURE Menuout;
 BEGIN
   Settextstyle(Triplexfont,Vertdir,2);
   Setcolor(p1c);
   Outtextxy(20,10,'Worms');
   Outtextxy(20,210,'Worms');
   Outtextxy(20,390,'Worms');
   Setcolor(p2c);
   Outtextxy(600,10,'Worms');
   Outtextxy(600,210,'Worms');
   Outtextxy(600,390,'Worms');
   Settextstyle(Defaultfont,Horizdir,1);
   Setcolor(p1c);
   Outtextxy(250,180,'1 - Játék');
   Outtextxy(250,220,'3 - Szín');
   Outtextxy(250,260,'5 - Kilépés');
   Setcolor(p2c);
   Outtextxy(250,200,'2 - Nehézségi szint');
   Outtextxy(250,240,'4 - Játékleírás');
 END;

 { kepernyotorles }
 PROCEDURE Erase;
 VAR r:Word;
 BEGIN
  Setcolor(Black);
  For r:=0 To 320 Do Begin
                      Rectangle(r,r,640-r,480-r);
                      Delay(4);
                     End;
```

```
END;

{ a jatek }
PROCEDURE Game;
CONST sizex=61; { a jatekter meretei }
      sizey=41;
      w=6;      { a jatekter mezoinek merete }
      hx=150;   { eltolas a grafikus feluleten }
      hy=70;
VAR lastbill1,lastbill2:Char;
    death1,death2:Boolean;
    i,j,p1x,p1y,p2x,p2y,steps:Byte;
    stepss:String[4];
    pt:Array[0..sizex,0..sizey] Of Boolean;

  { a kukac egyetlen mezonyi reszenek kirajzolasa }
  PROCEDURE Worm(x,y,color:Byte);
  BEGIN
    Setcolor(color);
    Setfillstyle(Solidfill,color);
    Bar(x*w+hx,y*w+hy,x*w+hx+5,y*w+hy+5);
    Putpixel(x*w+hx,y*w+hy,Black);
    Putpixel(x*w+hx+5,y*w+hy,Black);
    Putpixel(x*w+hx,y*w+hy+5,Black);
    Putpixel(x*w+hx+5,y*w+hy+5,Black);
  END;

BEGIN
Repeat
 { a jatekter feltoltese, grafikus kirajzolasa }
 For i:=0 To sizex Do For j:=0 To sizey Do pt[i,j]:=False;
 For i:=0 To sizex Do pt[i,0]:=True;
 For i:=0 To sizex Do pt[i,sizey]:=True;
 For i:=0 To sizey Do pt[0,i]:=True;
 For i:=0 To sizey Do pt[sizex,i]:=True;
 Erase;
 Setcolor(White);
 Rectangle(0*w+hx+5,0*w+hy+5,sizex*w+hx,sizey*w+hy);
 p1x:=round((sizex-1)/2);
 p1y:=round((sizey-1)/2);
 pt[p1x,p1y]:=True;
 Worm(p1x,p1y,p1c);
 p2x:=round((sizex-1)/2+1);
 p2y:=round((sizey-1)/2);
 pt[p2x,p2y]:=True;
 Worm(p2x,p2y,p2c);
 death1:=False;
 death2:=False;
 lastbill1:='a';
 lastbill2:=#77;
 Delay(1000);
 { a jatekmenet }
 Repeat
   If Keypressed Then Begin
                      bill:=Readkey;
                      { iranyitas }
                      Case bill Of
                        'w' : lastbill1:=bill;
```

```
                    's' : lastbill1:=bill;
                    'a' : lastbill1:=bill;
                    'd' : lastbill1:=bill;
                    #0  : Begin
                          bill:=Readkey;
                          Case bill Of
                            #72 : lastbill2:=bill;
                            #80 : lastbill2:=bill;
                            #75 : lastbill2:=bill;
                            #77 : lastbill2:=bill;
                          End;
                        End;
                  End;
                End;
 { mozgatas }
 Case lastbill1 Of
   'w' : Dec(p1y);
   's' : Inc(p1y);
   'a' : Dec(p1x);
   'd' : Inc(p1x);
 End;
 Case lastbill2 Of
   #72 : Dec(p2y);
   #80 : Inc(p2y);
   #75 : Dec(p2x);
   #77 : Inc(p2x);
 End;
 { utkozes figyelese }
 If pt[p1x,p1y]=True Then death1:=True
                 Else Begin
                        pt[p1x,p1y]:=True;
                        Worm(p1x,p1y,p1c);
                      End;
 If pt[p2x,p2y]=True Then death2:=True
                 Else Begin
                        pt[p2x,p2y]:=True;
                        Worm(p2x,p2y,p2c);
                      End;
 If (death1=False) And (death2=False) Then Inc(steps);
 Setcolor(White);
 Outtextxy(240,330,'Megtett lépések száma:');
 Setcolor(Black);
 Outtextxy(430,330,stepss);
 Setcolor(White);
 Str(steps,stepss);
 Outtextxy(430,330,stepss);
 Delay(ms);
Until (bill=#27) Or death1 Or death2;
{ eredmeny kiirasa }
If (death1=True) And (death2=true) Then Begin
                                         Setcolor(White);
                                         Outtextxy(285,400,'Döntetlen !');
                                       End;
If death1 Then Begin
               Setcolor(p1c);
               Outtextxy(20,150,' Egyes');
               Outtextxy(20,170,'játékos');
               Outtextxy(20,190,'meghalt !');
```

```
            For i:=1 To sizex-1 Do Begin
                        For j:=1 To sizey-1 Do Begin
                                    If pt[i,j]=True Then Worm(i,j,p1c);
                                    Delay(1);
                                   End;
                        Delay(1);
                       End;
           End;
 If death2 Then Begin
            Setcolor(p2c);
            Outtextxy(540,150,'Kettes');
            Outtextxy(540,170,'játékos');
            Outtextxy(540,190,'meghalt !');
            For i:=1 To sizex-1 Do Begin
                        For j:=1 To sizey-1 Do Begin
                                    If pt[i,j]=True Then Worm(i,j,p2c);
                                    Delay(1);
                                   End;
                        Delay(1);
                       End;
           End;
 If death1=True Then Setcolor(p1c);
 If death2=True Then Setcolor(p2c);
 Outtextxy(290,370,'V'); Delay(250);
 Outtextxy(310,370,'É'); Delay(250);
 Outtextxy(330,370,'G'); Delay(250);
 Outtextxy(350,370,'E'); Delay(250);
 Setcolor(Lightred);
 Outtextxy(297,420,' ESC - Fömenü');
 Rectangle(302,417,330,430);
 Outtextxy(255,440,'SPACE     - Uj játék');
 Rectangle(220,437,330,450);
 Repeat
   bill:=Readkey;
 Until (bill=#27) Or (bill=#32);
Until bill=#27;
Erase;
END;

{ nehezsegi szint kivalasztasa }
PROCEDURE Level;
BEGIN
 Erase;
 Setcolor(p1c);
 Outtextxy(260,180,'1 - Kezdö');
 Outtextxy(260,220,'3 - Mester');
 Setcolor(p2c);
 Outtextxy(260,200,'2 - Haladó');
 Outtextxy(260,240,'4 - Profi');
 bill:=Readkey;
 Case bill Of
   '1' : ms:=200;
   '2' : ms:=150;
   '3' : ms:=100;
   '4' : ms:=60;
 End;
 Erase;
END;
```

```
{ szinvalasztas }
PROCEDURE Colors;
VAR billb:Byte;
BEGIN
 Erase;
 Setcolor(p1c);
 Outtextxy(150,140,'Egyes játékos');
 Setcolor(14); Outtextxy(120,160,'1 - Sárga');
 Setcolor(4);  Outtextxy(120,170,'2 - Vörös');
 Setcolor(9);  Outtextxy(120,180,'3 - Kék');
 Setcolor(1);  Outtextxy(120,190,'4 - Sötétkék');
 Setcolor(10); Outtextxy(120,200,'5 - Zöld');
 Setcolor(2);  Outtextxy(120,210,'6 - Sötétzöld');
 Setcolor(6);  Outtextxy(120,220,'7 - Barna');
 Setcolor(7);  Outtextxy(120,230,'8 - Szürke');
 Setcolor(5);  Outtextxy(120,240,'9 - Lila');
 bill:=Readkey;
 Val(bill,billb,code);
 Case billb Of
   1 : p1c:=14;
   2 : p1c:=4;
   3 : p1c:=9;
   4 : p1c:=1;
   5 : p1c:=10;
   6 : p1c:=2;
   7 : p1c:=6;
   8 : p1c:=7;
   9 : p1c:=5;
 End;
 Setcolor(p2c);
 Outtextxy(450,140,'Kettes játékos');
 Setcolor(14); Outtextxy(420,160,'1 - Sárga');
 Setcolor(4);  Outtextxy(420,170,'2 - Vörös');
 Setcolor(9);  Outtextxy(420,180,'3 - Kék');
 Setcolor(1);  Outtextxy(420,190,'4 - Sötétkék');
 Setcolor(10); Outtextxy(420,200,'5 - Zöld');
 Setcolor(2);  Outtextxy(420,210,'6 - Sötétzöld');
 Setcolor(6);  Outtextxy(420,220,'7 - Barna');
 Setcolor(7);  Outtextxy(420,230,'8 - Szürke');
 Setcolor(5);  Outtextxy(420,240,'9 - Lila');
 bill:=Readkey;
 Val(bill,billb,code);
 Case billb Of
   1 : p2c:=14;
   2 : p2c:=4;
   3 : p2c:=9;
   4 : p2c:=1;
   5 : p2c:=10;
   6 : p2c:=2;
   7 : p2c:=6;
   8 : p2c:=7;
   9 : p2c:=5;
 End;
 Erase;
END;

{ jatekleiras }
```

```
 PROCEDURE Description;
 BEGIN
  Erase;
  Setcolor(Lightgray);
  Outtextxy(265,10,'Játékleírás');
  Outtextxy(290,460,'Fömenü');
  Outtextxy(100,60,'Irányítás');
  Setcolor(p1c);
  Outtextxy(40,100,'Egyes játékos:');
  Outtextxy(70,120,'W - Fel');
  Outtextxy(70,140,'S - Le');
  Outtextxy(70,160,'A - Balra');
  Outtextxy(70,180,'D - Jobbra');
  Setcolor(p2c);
  Outtextxy(40,220,'Kettes játékos:');
  Outtextxy(70,240,Chr(24));  Outtextxy(90,240,'- Fel');
  Outtextxy(70,260,Chr(25));  Outtextxy(90,260,'- Le');
  Outtextxy(70,280,Chr(27));  Outtextxy(90,280,'- Balra');
  Outtextxy(70,300,Chr(26));  Outtextxy(90,300,'- Jobbra');
  Setcolor(Lightgray);
  Outtextxy(400,60,'A játék célja');
  Outtextxy(290,100,'Két játékos játszhat egymás ellen egy-egy');
  Outtextxy(280,120,'kukaccal. Az a játékos nyer, aki tovább');
  Outtextxy(280,140,'képes mozgásban maradni a megrajzolt');
  Outtextxy(280,160,'pályán. Természetesen aki nekimegy a');
  Outtextxy(280,180,'falnak, elveszíti a játékot. Persze abban');
  Outtextxy(280,200,'az esetben is, ha nekimegy a már megtett');
  Outtextxy(280,220,'útvonalnak.');
  Outtextxy(290,240,'Tehát az a játékos nyer, aki ügyesebben,');
  Outtextxy(280,260,'logikusan felépítve az útvonalát csapdába');
  Outtextxy(280,280,'zárja az ellenfelét.');
  Outtextxy(290,300,'Persze kialakulhat döntetlen helyzet is,');
  Outtextxy(280,320,'abban az esetben, ha egymásnak ütköznek a');
  Outtextxy(280,340,'fejükkel, illetve egyszerre ütköznek a');
  Outtextxy(280,360,'falnak, vagy egyszerre "fogynak ki" a saját');
  Outtextxy(280,380,'területükböl.');
  Outtextxy(360,420,'Kellemes szórakozást !');
  bill:=Readkey;
  Erase;
 END;

{ foprogram }
Begin
 gd:=9;
 gm:=2;
 Initgraph(gd,gm,'');
 p1c:=14;
 p2c:=9;
 ms:=150;
 Repeat
   Menuout;
   bill:=Readkey;
   Case bill Of
     '1' : Game;
     '2' : Level;
     '3' : Colors;
     '4' : Description;
     '5' : quit:=True;
```

```
  End;
 Until quit;
 Erase;
 Closegraph;
End.
```

DISCS

```
PROGRAM DISCs;
USES Crt,Dos,Graph,WinCrt;
VAR gd,gm:Integer;
    bill:Char;
    quit:Boolean;

  { menu kiirasa }
  Procedure MenuOut;
  Var i,j,color:Word;
  Begin
   Cleardevice;
   For i:=1 To 12 Do
     For j:=1 To 9 Do
       If Not ((i In [4,5,6,7,8,9]) And (j In [3,4,5,6,7])) Then
         Begin
           color:=Random(4)+1;
           Setcolor(color);
           Setfillstyle(Solidfill,color);
           Fillellipse(i*50-10,j*50-10,20,20);
         End;
   Setcolor(White);
   Settextstyle(Triplexfont,Horizdir,2);
   Outtextxy(280,170,'DISCs');
   Settextstyle(Defaultfont,Horizdir,1);
   Outtextxy(250,230,'1 - Játék');
   Outtextxy(250,250,'2 - A játék szabályai');
   Outtextxy(250,270,'3 - Kilépés');
  End;

  { kepernyotorles }
  Procedure Erase;
  Var r:Word;
  Begin
   Setcolor(Black);
   For r:=0 To 320 Do Begin
                       Rectangle(r,r,640-r,480-r);
                       Delay(4);
                      End;
  End;

  { a jatek }
  Procedure Game;
  Const o=30;
  Type disc=Record { a korong tipusa }
             gx,gy:Word; { grafikus pozicioja }
             color:Byte; { szin }
            End;
  Var m:Array[1..12,1..8] Of Disc; { a jatekter }
      tempcolor,i,j,k,spots:Word;
      points,pointsadd:Word; { pontszam; az adott forduloban leszedett korongok szama }
```

```
spots_s,points_s:String[4];

 { egy korong kirajzolasa }
 PROCEDURE Spot(x1,y1:Word;c:Byte);
 BEGIN
  Setcolor(c);
  Setfillstyle(Solidfill,c);
  Fillellipse(x1,y1,Round(o/2)-2,Round(o/2)-2);
 END;

 { ellenorzes }
 PROCEDURE Check(x2,y2:Byte);
 BEGIN
  { a felso ellenorzese }
  If (y2-1>0) And (m[x2,y2-1].color=tempcolor) Then Begin
    m[x2,y2-1].color:=Black;
    Spot(m[x2,y2-1].gx,m[x2,y2-1].gy,Black);
    Dec(spots);
    Inc(pointsadd);
    Check(x2,y2-1);
  End;
  { az also ellenorzese }
  If (y2+1<9) And (m[x2,y2+1].color=tempcolor) Then Begin
    m[x2,y2+1].color:=Black;
    Spot(m[x2,y2+1].gx,m[x2,y2+1].gy,Black);
    Dec(spots);
    Inc(pointsadd);
    Check(x2,y2+1);
  End;
  { a bal oldali ellenorzese }
  If (x2-1>0) And (m[x2-1,y2].color=tempcolor) Then Begin
    m[x2-1,y2].color:=Black;
    Spot(m[x2-1,y2].gx,m[x2-1,y2].gy,Black);
    Dec(spots);
    Inc(pointsadd);
    Check(x2-1,y2);
  End;
  { a jobb oldali ellenorzese }
  If (x2+1<13) And (m[x2+1,y2].color=tempcolor) Then Begin
    m[x2+1,y2].color:=Black;
    Spot(m[x2+1,y2].gx,m[x2+1,y2].gy,Black);
    Dec(spots);
    Inc(pointsadd);
    Check(x2+1,y2);
  End;
 END;

 { az uresen maradt helyre a felette levo korongok lecsuszasa }
 PROCEDURE Push(x2,y2:Byte);
 BEGIN
  If (y2-1>0) And (m[x2,y2].color=Black) Then Begin
    m[x2,y2].color:=m[x2,y2-1].color;
    Spot(m[x2,y2].gx,m[x2,y2].gy,m[x2,y2].color);
    m[x2,y2-1].color:=Black;
    Spot(m[x2,y2-1].gx,m[x2,y2-1].gy,m[x2,y2-1].color);
    Push(x2,y2-1);
  End;
 END;
```

```
    { az ures helyek ellenorzese }
    PROCEDURE Checkspots;
    VAR k,l:Byte;
    BEGIN
      For k:=1 To 12 Do For l:=1 To 8 Do If m[k,l].color=Black Then Push(k,l);
    END;

Begin
 Erase;
 { a jatekter kirajzolasa }
 For i:=1 To 12 Do For j:=1 To 8 Do
  Begin
    m[i,j].color:=Random(4)+1;
    m[i,j].gx:=i*o+60;
    m[i,j].gy:=j*o+60;
    Spot( m[i,j].gx , m[i,j].gy , m[i,j].color );
    Setcolor(Darkgray);
    Rectangle(m[i,j].gx-Round(o/2),m[i,j].gy-Round(o/2),m[i,j].gx+Round(o/2),m[i,j].gy+Round(o/2) );
  End;
 spots:=96;
 spots_s:='96';
 points:=0;
 points_s:='0';
 Setcolor(Lightgray);
 Outtextxy(80,40,'Meglévö korongok száma: ');
 Outtextxy(280,40,spots_s);
 Outtextxy(400,40,'Elért pontszám: ');
 Outtextxy(540,40,points_s);
 Outtextxy(10,10,'ESCAPE - Kilép');
 i:=6;
 j:=4;
 { a kivalasztoteglalap kirajzolasa }
 Setcolor(White);
 Rectangle( m[i,j].gx-Round(o/2) , m[i,j].gy-Round(o/2) , m[i,j].gx+Round(o/2) , m[i,j].gy+Round(o/2) );
 REPEAT
  bill:=Readkey;
  { a kivalasztoteglalap mozgatasa }
  If bill=#0

   Then Begin
   Setcolor(Darkgray);
   Rectangle( m[i,j].gx-Round(o/2) , m[i,j].gy-Round(o/2) , m[i,j].gx+Round(o/2) , m[i,j].gy+Round(o/2) );
   bill:=Readkey;
   Case bill Of
     #72 : If j>1  Then Dec(j);
     #80 : If j<8 Then Inc(j);
     #75 : If i>1  Then Dec(i);
     #77 : If i<12 Then Inc(i);
   End;
   Setcolor(White);
   Rectangle( m[i,j].gx-Round(o/2) , m[i,j].gy-Round(o/2) , m[i,j].gx+Round(o/2) , m[i,j].gy+Round(o/2) );
   End
   { korong levetele }
   Else Begin
         If (bill=#32) And (m[i,j].color<>Black)
           Then Begin
               pointsadd:=1;
```

```
                tempcolor:=m[i,j].color;
                { korong levetel }
                m[i,j].color:=Black;
                Spot(m[i,j].gx,m[i,j].gy,m[i,j].color);
                Dec(spots);
                { ellenorzes }
                Check(i,j);
                Checkspots;
                Setcolor(Black);
                Outtextxy(280,40,spots_s);
                Str(spots,spots_s);
                Setcolor(Lightgray);
                Outtextxy(280,40,spots_s);
                Setcolor(Black);
                Outtextxy(540,40,points_s);
                { szerzett pontok hozzaadasa }
                For k:=1 To pointsadd Do Inc(points,k*10);
                Str(points,points_s);
                Setcolor(Lightgray);
                Outtextxy(540,40,points_s);
                End;
        End;
 UNTIL (bill=#27) OR (spots=0);
 If spots=0 Then Begin
                 Setcolor(White);
                 Outtextxy(280,40,'Gratulálok !');
                 bill:=Readkey;
               End;
 Erase;
 Menuout;
End;

{ jatekleiras }
Procedure Description;
Var i:Word;
Begin
 Erase;
 Setcolor(Lightgray);
 Outtextxy(265,10,'Játékleírás'); Line(265,20,355,20);
 Outtextxy(0,40,' Próbáld meg a lehetö legkevesebb lépésben leszedni a korongokat !');
 Outtextxy(0,60,'Az irányításhoz használd a kurzormozgató nyilakat és a szóközt !');
 Outtextxy(0,80,'Egy - egy levétellel egy - egy korong szedhetö le , de a mellette');
 Outtextxy(0,100,'lévö azonos színüek is eltünnek vele együtt . Ezután a felette lévö');
 Outtextxy(0,120,'korongok " lepotyognak " az üresen maradt helyekre .');
 Outtextxy(0,140,' Természetesen minél több korongot tudsz leszedni egyetlen levétel során');
 Outtextxy(0,160,'annál több pontot kapsz egyszerre .');
 Outtextxy(0,180,' Pl. egyetlen leszedett korongért 10 pontot kapsz , viszont ha egy');
 Outtextxy(0,200,'kattintással két korongot sikerül leszedned , azért nem 2 * 10 = 20');
 Outtextxy(0,220,'pont jár , hanem 30 !');
 Outtextxy(0,240,' Ime a pontozás :');
 Outtextxy(60,280,'1 korong - 10 pont'); Outtextxy(360,280,'10 korong - 550 pont');
 Outtextxy(60,300,'2 korong - 30 pont'); Outtextxy(360,300,'11 korong - 660 pont');
 Outtextxy(60,320,'3 korong - 60 pont'); Outtextxy(360,320,'12 korong - 780 pont');
 Outtextxy(60,340,'4 korong - 100 pont'); Outtextxy(360,340,'13 korong - 910 pont');
 Outtextxy(60,360,'5 korong - 150 pont'); Outtextxy(360,360,'14 korong - 1050 pont');
 Outtextxy(60,380,'6 korong - 210 pont'); Outtextxy(360,380,'15 korong - 1200 pont');
 Outtextxy(60,400,'7 korong - 280 pont'); Outtextxy(360,400,'16 korong - 1360 pont');
 Outtextxy(60,420,'8 korong - 360 pont'); Outtextxy(360,420,'17 korong - 1530 pont');
```

```
   Outtextxy(60,440,'9  korong  -  450 pont'); Outtextxy(360,440,'18  korong  -  1610 pont');
   bill:=Readkey;
   Erase;
   Menuout;
  End;

{ foprogram }
BEGIN
 gd:=9;
 gm:=2;
 Initgraph(gd,gm,'c:\program files\fpc');
 Randomize;
 Menuout;
 Repeat
   If keypressed Then bill:=Readkey;
   Case bill Of
     '1' : Game;
     '2' : Description;
     '3' : quit:=True;
   End;
 Until quit;
 Erase;
 Closegraph;
END.
```

RECOGNI

```
PROGRAM Recogni;
USES Crt,Graph,WinCrt;
CONST EMPTY=0;
      P1=10;
      P2=9;
      P0=254;
      QUIT=255;
VAR bill:Char;
    esc:Boolean;
    active,theend,i:Byte;
    matrix:Array[0..6,0..5] Of Byte; { a jatekter }
    gd,gm,arrow:Integer;

  { menu kiirasa }
  Procedure Menuout;
  Begin
   Cleardevice;
   For i:=1 To 4 Do Begin
                    Setcolor(Lightblue);
                    Setfillstyle(Solidfill,Lightblue);
                    Fillellipse(60,i*60+100,26,26);
                    Setcolor(Lightgreen);
                    Setfillstyle(Solidfill,Lightgreen);
                    Fillellipse(580,i*60+100,26,26);
                   End;
   Setcolor(Lightgreen);
   Outtextxy(280,160,'R c g i');
   Setcolor(Lightblue);
   Outtextxy(280,160,' e o n ');
   Outtextxy(250,220,'1 - Játék');
   Outtextxy(250,240,'2 - Játékleírás');
```

```
 Outtextxy(250,260,'3 - Kilépés');
End;

{ kepernyotorles }
Procedure Erase;
Var r:Word;
Begin
 Setcolor(Black);
 For r:=0 To 320 Do Begin
                     Rectangle(r,r,640-r,480-r);
                     Delay(4);
                    End;
End;

{ a korong letetele }
Procedure Spot(x,y,c:Integer);
Var tempcolor:Integer;
Begin
  Case c Of
    P1 : tempcolor:=10;
    P2 : tempcolor:=9;
    EMPTY  : tempcolor:=0;
  End;
  Setfillstyle(Solidfill,c);
  Setcolor(c);
  Circle(x*60+140,y*60+110,26);
  Floodfill(x*60+140,y*60+110,c);
End;

{ a jatekter kirajzolasa }
Procedure Matrixout;
Var i,j:Integer;
Begin
  Setfillstyle(Solidfill,Lightgray);
  Bar(106,76,533,443);
  Setfillstyle(Solidfill,Darkgray);
  Bar(110,80,529,439);
  Setcolor(Lightgray);
  For i:=0 To 6 Do
    For j:=0 To 5 Do Begin
                       Rectangle(i*60+140-30,j*60+110-30,i*60+140+30,j*60+110+30);
                       matrix[i,j]:=0;
                      End;
  Setcolor(Black);
  Line(110,80,530,80);
  Setfillstyle(Solidfill,Black);
  Bar(106,76,533,80);
End;

{ a kivalasztokorog (felul) }
Procedure Chooserspot(c:Byte);
Var tempcolor:Byte;
Begin
  Case c Of
    P1 : tempcolor:=10;
    P2 : tempcolor:=9;
  End;
  Setcolor(tempcolor);
```

```
  Setfillstyle(Solidfill,tempcolor);
  Fillellipse(arrow*60+140,50,25,25);
End;

{ korong torlese }
Procedure Deletespot;
Begin
  Setcolor(Black);
  Setfillstyle(Solidfill,Black);
  Fillellipse(arrow*60+140,50,25,25);
End;

{ egy fordulo }
Procedure Turn(color:Byte);
Var bill:Char;
    j:integer;
Begin
  While (arrow>=0) And (matrix[arrow,0]<>0) Do Dec(arrow);
  If arrow<0 Then Begin
                  Inc(arrow);
                  While matrix[arrow,0]<>0 Do Inc(arrow);
                 End;
  Chooserspot(color);
  { mozgatas }
  Repeat
    bill:=Readkey;
    If bill=#0 Then bill:=Readkey;
    Case bill Of
      #75 : Begin
              Deletespot;
              Dec(arrow);
              While (arrow>=0) And (matrix[arrow,0]<>0) Do Dec(arrow);
              If arrow<0 Then Begin
                                Inc(arrow);
                                While matrix[arrow,0]<>0 Do Inc(arrow);
                              End;
              Chooserspot(color);
            End;
      #77 : Begin
              Deletespot;
              Inc(arrow);
              While (arrow<=6) And (matrix[arrow,0]<>0) Do Inc(arrow);
              If arrow>6 Then Begin
                                Dec(arrow);
                                While matrix[arrow,0]<>0 Do Dec(arrow);
                              End;
              Chooserspot(color);
            End;
    End;
  Until (bill=#80) Or (bill=#27);
  { korong letetele }
  If bill=#80 Then Begin
                    j:=5;
                    While matrix[arrow,j]<>0 Do Dec(j);
                    matrix[arrow,j]:=color;
                    Spot(arrow,j,color);
                   End
              Else theend:=QUIT;
```

```
  Deletespot;
End;

{ ellenorzes }
Procedure Check(color:Byte);
Var i,j,k,counter:Word;
Begin
  { vizszintesen }
  For i:=0 To 3 Do
    For j:=0 To 5 Do Begin
                     counter:=0;
                     For k:=0 To 3 Do If matrix[i+k,j]=color Then Inc(counter);
                     If counter=4 Then theend:=color;
                   End;
  { fuggolegesen }
  For i:=0 To 6 Do
    For j:=0 To 2 Do Begin
                     counter:=0;
                     For k:=0 To 3 Do If matrix[i,j+k]=color Then Inc(counter);
                     If counter=4 Then theend:=color;
                   End;
  { atlosan, jobbra lefele }
  For i:=0 To 3 Do
    For j:=0 To 2 Do Begin
                     counter:=0;
                     For k:=0 To 3 Do If matrix[i+k,j+k]=color Then Inc(counter);
                     If counter=4 Then theend:=color;
                   end;
  { atlosan, balra lefele }
  For i:=3 To 6 Do
    For j:=0 To 2 Do Begin
                     counter:=0;
                     For k:=0 To 3 Do If matrix[i-k,j+k]=color Then Inc(counter);
                     If counter=4 Then theend:=color;
                   End;
  counter:=0;
  { dontetlen }
  For i:=0 To 6 Do For j:=0 To 5 Do If matrix[i,j]=EMPTY Then Inc(counter);
  If counter=0 Then theend:=P0;
End;

{ a jatek }
Procedure Game;
Var r:Word;
Begin
  Erase;
  Matrixout;
  active:=P1;
  arrow:=3;
  theend:=0;
  { fordulok }
  Repeat
    Turn(active);
    Check(active);
    If active=P1 Then active:=P2
                 Else active:=P1;
  Until theend<>0;
  Setcolor(White);
```

```
  { eredmeny kiirasa }
  Case theend Of
    P0 : Begin
          Setcolor(Lightgray);
          Outtextxy(260,60,'Döntetlen!');
         End;
    P1 : Begin
          Setcolor(Lightgreen);
          Outtextxy(260,60,'Nyert a zöld!');
         End;
    P2 : Begin
          Setcolor(Lightblue);
          Outtextxy(260,60,'Nyert a kék!');
         End;
  End;
  If theend<>QUIT Then bill:=Readkey;
  Setcolor(Darkgray);
  For r:=0 To 320 Do Begin
                      Rectangle(r,r,640-r,480-r);
                      Delay(4);
                     End;
  Erase;
 End;

 { jatekleiras }
 Procedure Description;
 Var r:Word;
 Begin
  Erase;
  Setcolor(Lightblue);
  Outtextxy(265,10,'Játékleírás');
  Outtextxy(290,460,'Fömenü');
  Outtextxy(100,60,'Irányítás');
  Outtextxy(70,120,Chr(27));
  Outtextxy(70,140,Chr(26));
  Outtextxy(70,160,Chr(25));
  Outtextxy(90,120,'- Balra');
  Outtextxy(90,140,'- Jobbra');
  Outtextxy(90,160,'- Korong le');
  Outtextxy(400,60,'A játék célja');
  Outtextxy(270,100,'Két játékos játszhat egymás ellen különbözö');
  Outtextxy(260,120,'színben. Az a játékos nyer, akinek elöbb');
  Outtextxy(260,140,'sikerül kiraknia 4 korongot a saját színében');
  Outtextxy(260,160,'a pályán vízszintesen, függölegesen vagy át-');
  Outtextxy(260,180,'lósan.');
  Outtextxy(270,200,'Felváltva teszik le a korongjaikat a játéko-');
  Outtextxy(260,220,'sok, és mindig lecsúszik a tábla legaljára.');
  Outtextxy(260,240,'Tehát csak oszlopot tudsz választani, sort');
  Outtextxy(260,260,'nem !');
  Outtextxy(270,280,'Persze kialakulhat döntetlen helyzet is,');
  Outtextxy(260,300,'abban az esetben, ha a tábla megtelik.');
  Outtextxy(360,360,'Kellemes szórakozást !');
  bill:=Readkey;
  Erase;
 End;

{ foprogram }
BEGIN
```

```
 gd:=9;
 gm:=2;
 Initgraph(gd,gm,'');
 Repeat
   Menuout;
   bill:=Readkey;
   Case bill Of
     '1' : Game;
     '2' : Description;
     '3' : esc:=True;
   End;
 Until esc;
 Erase;
 Closegraph;
END.
```

MEMOLOGIK

```
PROGRAM Memologik;
USES Crt;
CONST HELPVAR:Boolean=True;
      TIPP=15;
    { a tippek eltarolasahoz es ellenorzesehez szukseges matrix }
VAR matrix:Array [0..TIPP,1..10] Of -1..5;
    quit,done,showsoln:Boolean;
    i,j,step:Integer;
    bill:Char;

  { a lehetseges szinek }
  Function Color(c:Integer):Integer;
  Var d:Integer;
  Begin
    Case c Of
      -1 : d:=Black;
       0 : d:=DarkGray;
       1 : d:=Cyan;
       2 : d:=Green;
       3 : d:=Yellow;
       4 : d:=Red;
       5 : d:=White;
    End;
    Color:=d;
  End;

  { kiiras }
  Procedure Draw(i:Integer);
  Var j:Integer;
  Begin
    Textcolor(White);
    Write(i:4, ' ' );
    { tippek kiirasa }
    For j:=1 To 4 Do Begin
                       Textcolor(Color(matrix[i,j]));
                       Write(matrix[i,j]:2);
                     End;
    Write('  ');
    { egyezesek kiirasa }
    For j:=6 To 9 Do Begin
```

```
                Textcolor(Color(matrix[i,j]));
                Write('o');
               End;
  Writeln;
End;

{ a matrix sorainak kiirasa }
Procedure Drawmatrix(showsoln:Boolean);
Var i,j:Integer;
Begin
  Gotoxy(1,1);
  Clreol;
  If showsoln Then Draw(0);
  Gotoxy(1,2);
  For i:=1 To TIPP Do Draw(i);
End;

{ ellenorzes }
Function Check(i:Integer):Boolean;
Var j,k,m:Integer;
    mark:Array[1..4] of Integer;
    solved,flag:Boolean;
Begin
  For j:=1 To 4 Do mark[j]:=0;
  { egyezes vizsgalata }
  For j:=1 To 4 Do If matrix[i,j]=matrix[0,j] Then mark[j]:=2;
  { csak a tartalom ellenorzese }
  For j:=1 To 4 Do If mark[j]<>2 Then
    Begin
      flag:=True;
      For m:=1 To 4 Do
        If (matrix[i,j]=matrix[0,m]) And (mark[m]=0) And flag Then Begin
                                                    mark[m]:=1;
                                                    flag:=False;
                                                   End;
    End;
  k:=6;
  { ellenorzesek rogzitese }
  For j:=1 To 4 Do If mark[j]=2 Then Begin
                                matrix[i,k]:=0;
                                k:=k+1;
                               End;
  For j:=1 To 4 Do If mark[j]=1 Then Begin
                                matrix[i,k]:=5;
                                k:=k+1;
                               End;
  Repeat
    Inc(k);
  Until k>9;
  solved:=True;
  For j:=1 To 4 Do solved:=solved And (mark[j]=2);
  Check:=solved;
End;

Function Readuser(s:Integer):Char;
Var j,d:Integer;
    bill,bill1:Char;
Label quit;
```

```
Begin
  Textcolor(White);
  j:=1;
  { az adott tippsorozat bekerese }
  REPEAT
    { csak a megfelelo billentyuk elfogadasa }
    Repeat
      Gotoxy(5+j*2,s+1);
      If j<=4 Then Write('_');
      Gotoxy(5+j*2,s+1);
      bill:=Readkey;
      d:=Ord(bill)-Ord('0');
      readuser:=bill;
      Case bill Of
        #0      : bill1:=Readkey ;
        #27,'m' : Goto quit;
        #8      : Begin
                    If j>1 Then j:=j-1;
                    Write(' ');
                  End;
      End;
      If j>4 Then Goto quit;
    Until bill In ['0'..'5'];
    matrix[s,j]:=d Mod 6;
    Textcolor(Color(matrix[s,j]));
    Write(bill);
    j:=j+1;
  UNTIL j>5;
  quit:
End;

{ a jatek }
Procedure Game;
Begin
  Repeat
    { a jatek felulete }
    Clrscr;
    Textcolor(Lightgray);
    Gotoxy(64,23);
    Write('ESC - Kilép');
    Gotoxy(66,24);
    Write('M - Megoldás');
    Textcolor(White);
    Gotoxy(18,24);
    Write('Találd ki a négy színt ! [');
    For i:=0 To 5 Do Begin
                       Textcolor(Color(i));
                       Write(i:1);
                     End;
    Textcolor(Darkgray);
    Write(']');
    Gotoxy(55,4);
    Write('FEKETE KÖRÖK');
    Gotoxy(47,5);
    Write('ennyi szín van megfelelő helyen');
    Textcolor(White);
    Gotoxy(55,8);
    Write('FEHÉR KÖRÖK');
```

```
   Gotoxy(47,9);
   Write('ennyi szín jó, de rossz helyen');
   Randomize;
   showsoln:=False;
   { a matrix generalasa }
   For i:=0 To TIPP Do For j:=1 To 10 Do matrix[i,j]:=-1;
   For j:=1 To 4 Do matrix[0,j]:=Random(6);
   step:=1;
   { a jatek forduloi }
   Repeat
     Drawmatrix(showsoln);
     bill:=Readuser(step);
     If bill='m' Then showsoln:=True;
     done:=Check(step);
     Drawmatrix(showsoln);
     If Not (bill='m') Then Inc(step);
   Until (step>TIPP) Or done Or (bill=#27);
   Drawmatrix(True);
   Gotoxy(1,22);
   Clreol;
   { eredmeny kiirasa }
   Textcolor(White);
   If done Then Begin
                  Gotoxy(37,17);
                  Write('Nyertél !');
                End
           Else If step>=TIPP Then Begin
                                     Gotoxy(27,17);
                                     Write('Nem nyert ! Nincs több lehetöséged !');
                                   End;
   Gotoxy(32,19);
   Write('Még egy játék? (i/n)');
   bill:=Readkey;
   helpvar:=False;
  Until bill In [#27,'n'];
  Clrscr;
End;

{ jatekleiras }
Procedure Description;
Begin
  Clrscr;
  Gotoxy(35,2); Writeln('Játékleírás');
  Gotoxy(15,4); Writeln(' Teszteld egyszerre a memóriádat és a logikádat!');
  Writeln;
  Writeln('  Hatféle szín van, és négy helyen van elhelyezve négy korong. A Te dolgod');
  Writeln('kitalálni, hogy melyik helyen milyen színű korong van.');
  Writeln('  Minden tipped ellenőrzésre kerül, és a sor végén lévő fekete és fehér körök');
  Writeln('jelzik a korongok állapotát. A fekete körök jelzik, hogy hány korong van');
  Writeln('megfelelő helyen. A fehér körök jelzik, hogy hány korong van benne a sorban,');
  Writeln('de rossz helyen.');
  Writeln;
  Writeln('  Addig kell próbálkznod, amíg el nem találod a helyes színek helyes helyét.');
  Writeln('        De az is lehet, hogy kifogysz a lehetőségekből, és veszítesz...');
  Gotoxy(35,16); Writeln('Irányítás');
  Writeln;
  Writeln('  A 0-tól az 5-ös billentyűig jelöli a hat szám a hat féle színt. Ezekkel');
  Writeln('tudod kiválasztani azt a színt, amelyiket szeretnéd. Ha esetleg törölni');
```

```
   Writeln('szeretnéd, a BACKSPACE billentyűvel megteheted. Kilépni pedig az ESC-pel lehet.');
   Writeln(' Ha esetleg a megoldást szeretnéd látni, üsd le az M bilentyűt!');
   Gotoxy(32,23); Write('Jó szórakozást !');
   bill:=Readkey;
   Clrscr;
  End;

{ foprogram }
BEGIN
 Clrscr;
 { menu }
 Repeat
   Textcolor(Lightgray);
   Gotoxy(35,6);  Write('Memologik');
   Gotoxy(33,10); Write('1 - Játék');
   Gotoxy(33,11); Write('2 - Játékleírás');
   Gotoxy(33,12); Write('3 - Kilép');
   bill:=Readkey;
   Case bill Of
    '1' : Game;
    '2' : Description;
    '3' : quit:=True;
   End;
 Until quit;
END.
```

AMŐBA

```
PROGRAM Amoba;
USES Crt,Dos,Graph,Wincrt;
VAR gd,gm:Integer;
   bill:Char;
   quit:Boolean;

  { kepernyotorles }
  Procedure Erase;
  Var r:Word;
  Begin
   Setcolor(Black);
   For r:=0 To 320 Do Begin
                      Rectangle(r,r,640-r,480-r);
                      Delay(2);
                     End;
  End;

  { menu kiirasa }
  Procedure MenuOut;
  Begin
   Erase;
   Setcolor(Lightred);
   Line(361-6,173-6,361+6,173+6);
   Line(361-5,173-6,361+5,173+6);
   Line(361+6,173-6,361-6,173+6);
   Line(361+5,173-6,361-5,173+6);
   Setcolor(Lightblue);
   Setfillstyle(Solidfill,Lightblue);
   Fillellipse(265,173,7,7);
   Settextstyle(Triplexfont,Horizdir,2);
```

```
  Outtextxy(280,160,'Amöba');
  Settextstyle(Defaultfont,Horizdir,1);
  Outtextxy(250,220,'1 - Játék');
  Outtextxy(250,240,'2 - A játék szabályai');
  Outtextxy(250,260,'3 - Kilépés');
End;

{ a jatek }
Procedure Game;
{ a korong tipusa }
Type disc=Record
          gx,gy:Word; { grafikus pozicio }
          color:Byte;
        End;
Const o=10;  { babu meret }
     mx=50; { palyameret, oszlop }
     my=35; { palyameret, sor }
Var m:Array[1..mx,1..my] Of Disc; { a jatekter }
   win,color,tempcolor:Byte;
   turns,i,j,k:Word;

    { egy korong kirajzolasa }
    PROCEDURE Spot(x1,y1:Word;c:Byte);
    BEGIN
      Setcolor(c);
      Setfillstyle(Solidfill,c);
      { a 9-es szinu jatekos egy korongot tesz le, a 12-es egy X-et }
      Case c Of
        9  : Fillellipse(x1,y1,Round(o/2)-2,Round(o/2)-2);
        12 : Begin
               Line(x1-3,y1-3,x1+3,y1+3);
               Line(x1-2,y1-3,x1+2,y1+3);
               Line(x1+3,y1-3,x1-3,y1+3);
               Line(x1+2,y1-3,x1-2,y1+3);
             End;
      End;
    END;

    { ellenorzes }
    PROCEDURE Check;
    VAR spots,i,s,o:Byte;
    BEGIN
      { vizszintesen }
      spots:=0;
      For s:=1 To my Do For o:=1 To mx-4 Do If m[o,s].color<>0 Then
        Begin
          tempcolor:=m[o,s].color;
          spots:=1;
          For i:=1 To 4 Do If m[o+i,s].color=tempcolor Then Inc(spots);
          If spots=5 Then win:=tempcolor;
        End;
      { fuggolegesen }
      spots:=0;
      For o:=1 To mx Do For s:=1 To my-4 Do If m[o,s].color<>0 Then
        Begin
          tempcolor:=m[o,s].color;
          spots:=1;
          For i:=1 To 4 Do If m[o,s+i].color=tempcolor Then Inc(spots);
```

```
          If spots=5 Then win:=tempcolor;
        End;
      { atlosan jobbra lefele }
      spots:=0;
      For s:=1 To my-4 Do For o:=1 To mx-4 Do If m[o,s].color<>0 Then
        Begin
          tempcolor:=m[o,s].color;
          spots:=1;
          For i:=1 To 4 Do If m[o+i,s+i].color=tempcolor Then Inc(spots);
          If spots=5 Then win:=tempcolor;
        End;
      { atlosan balra lefele }
      spots:=0;
      For s:=1 To my Do For o:=mx Downto 5 Do If m[o,s].color<>0 Then
        Begin
          tempcolor:=m[o,s].color;
          spots:=1;
          For i:=1 To 4 Do If m[o-i,s+i].color=tempcolor Then Inc(spots);
          If spots=5 Then win:=tempcolor;
        End;
    END;

Begin
 Erase;
 { a jatekter kirajzolasa }
 Setcolor(Darkgray);
 Setfillstyle(Solidfill,Darkgray);
 Bar(65,65,565,415);
 For i:=1 To mx Do For j:=1 To my Do
   Begin
     m[i,j].color:=Black;
     m[i,j].gx:=i*o+60;
     m[i,j].gy:=j*o+60;
     Setcolor(Lightgray);
     Rectangle(m[i,j].gx-Round(o/2),m[i,j].gy-Round(o/2),m[i,j].gx+Round(o/2),m[i,j].gy+Round(o/2) );
   End;
 Outtextxy(10,10,'ESCAPE - Kilép');
 Rectangle(6,6,123,21);
 win:=0;
 color:=9;
 turns:=0;
 i:=25;
 j:=17;
 { a mezo kivalasztasahoz szukseges vilagos teglalap }
 Setcolor(color);
 Rectangle( m[i,j].gx-Round(o/2) , m[i,j].gy-Round(o/2) , m[i,j].gx+Round(o/2) , m[i,j].gy+Round(o/2) );
 { egy-egy jatekos forduloja }
 REPEAT
   IF keypressed THEN
   BEGIN
   bill:=Readkey;
   If bill=#0

    Then Begin
    { a kivalasztoteglalap mozgatasa }
    Setcolor(Lightgray);
    Rectangle( m[i,j].gx-Round(o/2) , m[i,j].gy-Round(o/2) , m[i,j].gx+Round(o/2) , m[i,j].gy+Round(o/2) );
    bill:=Readkey;
```

```
   Case bill Of
     #72 : If j>1  Then Dec(j);
     #80 : If j<my Then Inc(j);
     #75 : If i>1  Then Dec(i);
     #77 : If i<mx Then Inc(i);
   End;
   Setcolor(color);
   Rectangle( m[i,j].gx-Round(o/2) , m[i,j].gy-Round(o/2) , m[i,j].gx+Round(o/2) , m[i,j].gy+Round(o/2) );
   End
   { korong letetele }
   Else Begin
         If (bill=#32) And (m[i,j].color=Black)
          Then Begin
              m[i,j].color:=color;
              Spot(m[i,j].gx,m[i,j].gy,color);
              tempcolor:=color;
              { ellenorzes }
              Check;
              If win=0 Then Begin
                            If color=9 Then color:=12
                                    Else color:=9;
                            Inc(turns);
                           End;
              Setcolor(color);
              Rectangle(64,64,566,416);
              Rectangle(63,63,567,417);
              If turns=(mx*my) Then win:=1;
              End;
       End;
  END;
 UNTIL (bill=#27) Or (win In [1,9,12]);
 { az eredmeny megallapitasa es kiirasa }
 If win<>0 Then Begin
               Case win Of
                 1 :  Begin
                       Setcolor(Lightgray);
                       Outtextxy(230,20,'    Döntetlen !');
                      End;
                 9 :  Begin
                       Setcolor(win);
                       Outtextxy(230,20,'Nyert a kék játékos !');
                      End;
                 12 : Begin
                       Setcolor(win);
                       Outtextxy(230,20,'Nyert a piros játékos !');
                      End;
               End;
               bill:=Readkey;
              End;
 Menuout;
End;

{ jatekleiras }
Procedure Description;
Begin
 Erase;
 Setcolor(Lightblue);
 Outtextxy(265,80,'Játékleírás'); Line(265,90,355,90);
```

```
   Outtextxy(50,120,'  Próbálj a saját figurádból 5 db-ot kirakni egymás mellé ! Ez');
   Outtextxy(50,140,'lehet persze vízszintesen, függölegesen , vagy akár átlósan is .');
   Outtextxy(50,160,'Az irányításhoz használd a kurzormozgató nyilakat és a szóközt .');
   Outtextxy(50,180,'Természetesen a két játékos felváltva teszi le a saját figuráját .');
   Outtextxy(50,200,'Mindig éppen amilyen színü a keret , az a játékos következik .');
   Outtextxy(50,220,'  Elöfordulhat az is , hogy a tábla megtelik : ebben az esetben ');
   Outtextxy(50,240,'a játék eredménye döntetlen !');
   bill:=Readkey;
   Menuout;
  End;

{ foprogram }
BEGIN
 gd:=9;
 gm:=2;
 Initgraph(gd,gm,'');
 Randomize;
 Menuout;
 Repeat
   If keypressed Then bill:=Readkey;
   Case bill Of
     '1' : Game;
     '2' : Description;
     '3' : quit:=True;
   End;
 Until quit;
 Erase;
 Closegraph;
END.
```

IRODALOMJEGYZÉK

* Madarász Tiborné – Pólos László: A logika elemei (1999)

* Ruzsa Imre: A logika tudománya (1998)

* Mérei Ferenc – V. Binét Ágnes: Gyermeklélektan (1993)

* Nemes Lívia: Az értelmi fejlődés és a gondolkodás szakaszai Piaget műveiben (1958)

* Kiss Árpád: Tanítás és értelmi fejlődés (1947)

* http://www.ludology.org

* http://prog.hu

* http://www.oki.hu/oldal.php?tipus=cikk&kod=2006-02-3l-kajtar-szamitogep

* http://www.oki.hu/oldal.php?tipus=cikk&kod=2000-11-eu-Budai-Egesz

* http://www.oki.hu/oldal.php?tipus=cikk&kod=1999-07-in-Feher-Szamitogep