A „Szöveges kalandjátékok készítésének gyakorlati problémái” című szakdolgozat szövege

Készítette: Olessák Róbert, a Kandó Kálmán Műszaki Főiskola számára, 1997-ben

1.) TARTALOMJEGYZÉK

2.) ÖSSZEFOGLALÓ

Jelen írás két fő részből fog állni. Az első részben magát a kalandjáték fogalmát tárgyaljuk, illetve általánosságban beszélünk annak megvalósítási formáiról és lépéseiről. Főképpen a hagyományos szöveges kalandjátékról esik szó benne, külön koncentrálva a magyar nyelvű szövegértés nehézségeire (főnevek és igék ragozása, kötetlen szórend, ékezetes betűk stb.). A második rész azzal foglalkozik röviden, hogy az ismertetett alapelvek milyen formában kerültek alkalmazásra a J. R. R. Tolkien: A babó című regénye nyomán íródott kalandjáték elkészítése során. Az említett program dokumentációja és kezelési útmutatója zárja le a tanulmányt.

Játékokkal foglalkozni nem feltétlenül tartozik a lenézendő vagy megvetendő tevékenységek közé: minden élőlény egészséges állapota, ha játszik, és különösen pedig a legnagyobb matematikusok mindig is rendkívül játékos elmék voltak. Itt van pl. mindjárt Neumann János; vagy a John Conway angol matematikus által kiagyalt Életjáték – sejtautomata, mely kiált a számítógépes megvalósítás után. Állítom, hogy a kalandjáték is ugyanebbe a kategóriába tartozik!

3.) SUMMARY

Present writing will consist of two main parts. In the first part, we treat the idea of adventure itself, respectively talk about its forms and steps of realization. The question of traditional text adventures comes up mainly, especially concentrating the difficulties of Hungarian textual analysis (inflection of nouns and verbs, absolutely free order of words, accented letters etc.). The second part shortly deals with that how the recited principles were applied while making the adventure The Hobbit adapted from J. R. R. Tolkien's novel, the same entitled. The documentation and users' manual of the mentioned program closes the study.

Playing games is not necessarily one of those activities we can look down and despise: every living creatures' healthy condition if playing, and especially the greatest mathematicians were always extraordinarily playful minds. For example, John Neumann at once; or the Life Game thought out by English mathematician John Conway – a cell automaton crying for computer realization. I state positively that adventure comes under the same category, too!

4.) KIDOLGOZÁS

4.1.) A kalandjáték-készítés általános tudnivalói

4.1.1.) A kalandjáték fogalmának tisztázása

Az isteni szikra két egyetemista (bizonyos Crowther és Woods nevezetű úriemberek) fejéből pattant ki 1977-ben, amikor is előálltak PDP-10 számítógépre készített – és akkoriban még teljesen eredetinek számító – játékprogramjukkal, melynek az „Adventure” (magyarul: „Kaland”) címet adták. Egy kígyókkal, manókkal és más mesebeli lényekkel benépesített földalatti labirintusban elszórt kincseket kellett a játékosnak megtalálnia. Maga a program a korabeli diákság kedvenc szórakozásává lépett elő; a ráakasztott elnevezés pedig – az azóta eltelt két évtized során – olyan fogalommá terebélyesedett, melyről ma már szinte mindenkinek más-más dolgok jutnak eszébe. Számítógépjátékok tízezrei viselik büszkén vagy kevésbé büszkén ezt a besorolást, de ha némelyiket közülük egymás mellé raknánk, hát bizony meg sem fordulna a fejünkben a gondolat, hogy ezeknek bármi közük is lehet egymáshoz. Az egyik képernyőn egy kicsinyített színes csatateret látunk mintegy madártávlatból, vezényszóra ide-oda rohangáló figurákkal tarkítva; a másikat egy térhatású, penészes falú börtön hatalmas sötét alagútja tölti be; a harmadikon mondatokba és bekezdésekbe tördelt, sűrűn szedett apróbetűs szöveg olvasható. Pedig valamennyi ugyanarról a közös tőről fakad: amiről ebben az írásban szó lesz, az mindezek őstípusa, mely mindazonáltal nem pusztult ki és nem tűnt el teljesen ma sem, csupáncsak erősen háttérbe szorította őt ezer ágon fejlődő leszármazottainak hihetetlen burjánzása – pontos meghatározás szerint ez az ún. *hagyományos szöveges kalandjáték*. A „kalandjáték” szót tehát mostantól kezdve ebben az értelemben használom. Még ha másfajta programot szándékozik készíteni valaki, akkor is erről az alapról kell elindulnia; s noha főképpen ezekről lesz szó, természetesen a téma alapelvei mindenfajta grafikus játékra is vonatkoznak – elvégre a rajzos külső is csak egy burok, amit ha lehántunk róla, a program belsejében hasonlóan kell nyilvántartanunk a dolgokat. (S minekutána egy színvonalas szöveges játékot megírni némileg nehezebb programozói feladat, mint egy grafikust, ez egyúttal jó iskolát is jelent a későbbiek számára.)

Mihelyt egy ilyen programot elindítunk, rendszerint valamilyen különös helyszínen, egy fantáziavilág közepén találjuk magunkat. Kapunk egy bekezdésrevaló leírást vagy valamilyen képet tartózkodási helyünk közvetlen környezetéről – esetleg némi magyarázatot is arról, miképpen kerültünk oda –, és elindulunk fölfedezni világunk ismeretlen titkait. A klasszikus modellben a játék egyfajta párbeszédes üzemmódban működik: mi valamit mondunk a számítógépnek (kérjük vagy utasítjuk valaminek a megtételére), ő pedig üzenetek útján értesít bennünket lépéseink eredményéről. De cselekvéstől függetlenül is történhetnek események. Maga a játék lényegében egy kitalált világ életének szimulációja: ez a világ, hogy könnyebben

ábrázolható legyen, határozott, különálló helyiségekre oszlik. Egy-egy ilyen helyszín lehet pl. egy erdei tisztás, egy lépcsőforduló, egy szoba egy házban vagy egy tó vizén ringatózó csónak – általánosságban a terep bármely jellegzetes pontja, mely a folyamatos mozgás számára megállóhelyet, a látványnak kapaszkodót biztosít; egyedül a készítők gondosságán múlik, milyen részletességgel tagolják az egész helyszínt kisebb vagy nagyobb helyiségek sokaságára. A különálló helyeket átjárók kötik össze egymással; megállapodás szerint ezeket égtájakkal szokás azonosítani, így rendszerint tíz-tizenkét irányban haladva kerülhetünk egyikből a másikba (a nyolc égtáj, kiegészítve a FEL/LE/KI/BE lehetőségeivel). Pl. egy folyópartról észak felé indulva egy erdő pereméhez érkezünk. Egy helyszínről többnyire három-négy kijárat vezet (a többi irányban pl. fal van, vagy más gátló tényezők), s a világ térképét megkaphatjuk úgy, ha ezeket az utakat, mint csomópontokat összekötő hálózatot, lerajzoljuk egy papírra. Hogy ne lehessen rögtön az egészet bejárni, ajtók, kapuk és más nyílászárók feszülnek itt-ott két-két helyiség között (ezek nyitva, csukva és zárva lehetnek), de gyakorlatilag bármilyen típusú akadály is elképzelhető: kőomlás, amit el kell takarítani, ellenséges szereplő, aki elállja az utat, vagy akár egy vizesárok, egy kerítés vagy bármi. A helyiségeket helyhez kötött díszletek és mozdítható tárgyak töltik meg – utóbbiakat föl lehet venni, le lehet rakni, el lehet cipelni máshová és a legkülönbözőbb célokra alkalmazni őket. Tipikus eszközök pl. a lámpa, mellyel sötétben világítani tudunk, vagy a lezárt ajtók kulcsai – de itt is akármi elképzelhető. Ugyancsak jelen vannak itt a hozzánk hasonló (csak a gép által vezérelt) élőlények, szereplők. A legváltozatosabb módokon kommunikálhatunk velük: beszélgethetünk, megtámadhatjuk őket, tárgyakat csereberélhetünk; lehetnek helyhez kötöttek és passzívak, de mozoghatnak is egyik helyiségből a másikba, sőt, hozzánk hasonlóan „saját akaratukból” cselekedhetnek is ezt-azt; lehetnek ellenségesek vagy barátságosak. Legegyszerűbb viselkedésünk egy ilyen világban az lehet, hogy céltalanul csavargunk kusza labirintusának terein, miközben érdeklődve figyeljük előre beprogramozott vagy véletlenszerűen változó életét. Ez minden kalandjáték váza, s különbséget mindössze az tesz egyik vagy másik közt, hogy a helyiségek, akadályok, tárgyak és szereplők egymáshoz való viszonyát mennyire összetetten és sokoldalúan képes megjeleníteni számunkra.

Még néhány szót arról, miért is jelent egy igényes (!!!) szöveges játék elkészítése keményebb megpróbáltatást, mint grafikus testvéreié. Egy igazán intelligens programnak igék és főnevek százait, ezreit kell megértenie és helyesen alkalmaznia, köztük egy rakás, semmilyen szabályba sem sorolható kivétellel: a játékos ezeket tetszése szerint variálhatja egymással, aminek következtében a parancsoknak sok tízezer vagy akár több millió (vagy még ennél is több!) kombinációja és variációja is elképzelhető bennük, s ez ráadásként még a helyszínek és helyzetek számával is igencsak fölszorzódik (nem mindegy, hogy hol adjuk ki őket, vagy hogy melyik lépésünk előzi meg a másikat) – olyan rengeteg, hogy egyetlen játék(os) sem képes akárcsak töredékéig kiaknázni őket. Ha egy kalandjáték fenn kívánja tartani a valószerűség látszatát (és ezen keresztül a játékosok érdeklődését…), úgy ezek a lehetőségek közül minél többre kell valamilyen értelmes válasszal szolgálnia, azt az illúziót keltve, mintha birtokában lenne valamennyinek – a programozónak lehetőség szerint minden számbajöhető kombinációt figyelembe kéne vennie, elvarrnia minden elképzelhető szálat. Később majd látni fogjuk, hogy az utasítások kielemzése, a szóragozás és a szövegértés nehézségei is micsoda egy bonyolult problémával keserítik életünket. Ezzel szemben egy grafikus játék irányítása ikonokon, menükön keresztül

zajlik, s a mindössze maroknyi alapfunkció már önmagában töredékére zsugorítja a kipróbálható lehetőségek számát, melyeket csupán listaszerűen végig kell venni. (A tényleges megjelenítés problémája pedig már nem is a programozóra, hanem a rajzolóra tartozik…) A cselekvési térnek éppen ez a beszűkülése magyarázza egyébként azt is, hogy igazán intelligens játékokat miért nem lehet grafikusan, hanem kizárólag szöveges alapúként elkészíteni. (Esetleg olyan kereszteződések formájában, mint amilyen pl. a Maniac Mansion volt, a Times of Lore vagy a Tir Na Nog.) A szöveges játékok legnagyobb varázsát pontosan az nyújtja, hogy sohasem érezhetjük bennük úgy, hogy kimerítettük összes rendelkezésre álló lehetőségeiket; elvégre örökké tömegével maradnak még olyan párosítások, amelyeket egyszerűen nem jutott eszünkbe leírni (különösen, ha a programozók gondoskodtak róla, hogy ébrentartsák bennünk a reményt, mindenféle utalások képében) – egy grafikus játéknál ez a tágasság érzése csak sokkal szerényebb mértékben van jelen. (Azonkívül manapság már egy szöveges játékkal szemben is alapvető elvárás, hogy grafikus képernyőt kezeljen – legalább a megtervezett betűkészlet és az állóképek erejéig.)

4.1.2.) Kalandjáték és mitológia

Ha valaki szereti a sört, annak nyilván nagyobb örömet okoz elfogyasztani egy korsónyi Soproni Ászokot, mint a számítógépbe pötyögni, hogy IGYÁL SÖRT. Természetesen a számítógépes játékok nem arra valók, hogy helyettesítsék az igazi dolgokat (helytelenül teszi, aki erre használja őket), hanem hogy valami olyasmit nyújtsanak, amelyet máshol nem talál meg az ember. Mert mi is tulajdonképpen egy kalandjáték? Világmodell: olyan, mint a Karácsonyfa a gyertyáival, amely a csillagos égre emlékeztet (a boák a Tejutat, az üveggömbök a bolygókat jelképezik rajta stb.). Az általunk elképzelt világnak egy – ugyancsak általunk – létrehozott, kicsinyített mása. És minél gazdagabb, árnyaltabb és részletesebb ez a világ, annál nagyobb örömet jelent foglalkozni vele.

Minden egyes embernek – hacsak nem kíván maga is elveszni, eltévedni vagy szétforgácsolódni a nagyvilágban – alapvetően sürgető igénye van rá, hogy az őt körülvevő világot egy egységes egészként lássa át. Mivel pedig ezt a valóságban kivitelezni – tekintettel rá, hogy a Mindenség milyen elképzelhetetlenül roppant tömegű, szövevényes és bőséges valami – nyilvánvalóan reménytelen és lehetetlen vállalkozás volna, így nem marad más hátra, mint ennek mintájára egyszerűsített, stilizált fantáziaképeket, szimbólumokat teremteni magunkban. Olyanokat, melyek – végesek lévén – befogadhatóan és megemészthetően utalnak vissza a Végtelenségre. (Maga a Nyelv, amit beszélünk, a leggyönyörűbb példája ennek!)

A mitológia a régiek által elképzelt hétrétegű világegyetem ábrázolása volt: a regék és mítoszok többsége az egyes csillagképeket és az égitestek mozgását (és azok ránk gyakorolt hatását, a napszakok és évszakok változásait) írja le jelképes, metaforikus megfogalmazásban. A magyar népmeséket elemző Jankovics Marcell írásaiból kiderül, hogy részben ugyanez a hagyomány folytatódik a népmesék világában is: amikor Borsszem Jankó megüli a hatökrös szekeret, az eredetileg a Göncöl hét csillagának megjelenítése volt. (A hetedik, a leghalványabb, a szekér rúdjának a tövében, a Jankó.) Az általunk oly előszeretettel magunkénak vallott Csodaszarvas mondája sem kizárólag a miénk, hanem főként Szibériában, de gyakorlatilag az egész

északi félgömbön általánosan elterjedt motívumot alkot; s hogy az éghez kötődik, azt mi sem bizonyítja jobban, mint hogy e terület legtávolabbi vidékein is – egymás létéről semmit sem tudva – megszólalásig hasonló történeteket szőttek az emberek az északi égbolt bizonyos csillagképei köré. (A Cassiopeia, a Perseus és az Auriga együtt egy szaladó szarvashoz hasonlít, s az ezt körülvevő alakok a történet további hősei. A különböző népek sajátos változatai közti különbségek pedig jórészt visszavezethetők arra, ahogy az eltérő földrajzi körülmények miatt másképpen látszanak a csillagok az égen.) Természetesen nem puszta szórakozásból fektettek tekintélyes mennyiségű szellemi munkát egy teljes, összefüggő jelképrendszer kialakításába, hanem mert elemi szükségük volt rá: írás-olvasás, naptár, iránytű, térkép és más, tájékozódást segítő találmányok híján ezekkel írták le a körülöttük levő világ naponta, havonta és évenként ciklikusan visszatérő változásait. (Egyes szibériai sámándobokon például olyan elrendezésben lyukakat fúrtak, hogy azokon keresztül az égbolt megfelelő csillagaira nézve, a dob valamelyik díszítése mutatta az északi irányt – úgy használták ezeket, mint tengerész a szextánst.) Nemcsak romantikus álmodozás hozta őket létre, de a maguk környezetében komoly gyakorlati jelentőséggel bírtak. A bűvös számok, alkímiai, asztrológiai jelképek, a népi babonában csökönyösen meglapuló számmisztika – mindezek talán egy történelem előtti, összefüggő csillagászat és matematika darabokra hullott, törmelékes maradványai.

A szöveges kalandjátékoknak kezdetben még megvolt az az egyedülálló varázsuk, hogy az archaikus világnak erre az elveszített rendjére emlékezhetett általuk az ember (bár persze nem közvetlenül, hanem csak áttételesen: magát az előbbi alapelvet, a csillagképek ábrázolását hiába keresné bennük bárki – de például a kelta mitológia elemeit ezerszer is fölhasználták bennük); a hömpölygő szöveg és a mellette illusztrációként alkalmazott képek látványa régi könyvek hangulatára emlékeztetett; a szellemes útvesztők, a jól eltalált leírások – és egyáltalán: a Nyelv, a Beszéd központi szerepe – pedig tagadhatatlanul némi irodalmi ízt adtak neki. Azóta lassanként eluralta őket a rajzfilm meg az üzlet, s kiadóik ma már Walt Disney-szerű giccsfigurákkal keresnek dollármilliókat. Amikor még jóformán csak szövegek voltak bennük, a játékosnak (a szerzőnek meg aztán pláne!) igencsak meg kellett tornáztatnia a képzeletét, hogy maga előtt lássa a mögöttük rejlő világot; és ezt a világot mindenki egy picit másmilyennek látta, attól függően, hogy mennyire és mivel egészítette ki azt önmagában, belül (amitől aztán részben a magáénak is érezhette őket). Ha egy 16-millió színű, térhatású mozgókép jelenik meg a képernyőn, az – a tökéletes illúzió lehengerlő erejével bírva – jóval csekélyebb belső részvételt igényel a nézőtől, aki ennek megfelelően nem is csinál egyebet, mint lustán és közönyösen kattintgat az egérrel ide-oda. Ha egy hangzás tökéletes, már nincs mit kiegészíteni rajta, s a hallgatóságra szinte nincs is szükség hozzá: az első pár perc izgalma után méla undorba és unalomba fullad az egész.

Mivel a kalandjáték, mint olyan, is csak egy jelképes ábrázolás, ezért egyértelműen a dekadencia jelének tekinthető, ha a játékosok attól vannak elragadtatva, hogy a kép- és hanghatásoknak köszönhetően milyen élethűen jelennek meg benne a főhősök meg a helyszín – ahelyett, hogy a *lényegével* törődnének a játéknak! A készítők pedig a divathullámot meglovagolva szorgalmasan csomagolják számukra a látványosan ragyogó semmit…

Jelenleg szinte alig akad olyan területe a PC-s világnak, ahol az eredeti szándék még érvényesülni képes; de ez is inkább csak a technikai körülményeknek (pontosabban

azok korlátainak…) köszönheti létét; az Interneten keresztül játszható kalandjátékokról van szó. A TELNET egy viszonylag gyors és egyszerű szöveges párbeszédes kapcsolatot létesít egy központi gép és a hozzá bejelentkező terminálok között – ezt kihasználva rengeteg szöveges játékot írtak és írnak, amelyek egy (rendszerint Linux-os alapú) szolgáltató gépen éjjel-nappal futnak, és a világ bármelyik pontjáról bárki beléjük léphet. Legkellemesebb vonása ezeknek a programoknak az, hogy rajtunk kívül is még akárhány illető részt vehet bennük egyidejűleg, a játék többi szereplőjét is igazi hús-vér emberek irányítják: nincs szükség többé a számítógép által jól-rosszul szimulált értelemre, hiszen valódi értelem (mégpedig egyszerre akár több száz!) jelenik meg helyette, a szereplőkön, mint szócsöveken át. A különböző játékosok együttműködhetnek, segíthetik egymást, csapatokat alkothatnak, vagy akár versenghetnek is egymással vagy egymás ellen. Ezekben az élet megszakítás nélkül zajlik – hogyha mi kilépünk belőle, az addigi helyzet akkor is létezik tovább. Található köztük néhány színvonalas alkotás (pl. Holy Mission, Isengard). Hátrányuk viszont, hogy ezeknek a parancsértelmezőjét – lustaságból, vagy ki tudja, miért – szinte minden esetben rendkívül elhanyagolják: sokszor csak a legprimitívebb, „egy ige meg egy főnév” típusú utasítások kiértékelésére képesek, nem lehet egyben több parancsot beírni és így tovább. A hiányzó minőséget itt a mennyiséggel pótolják: a nagyszámú szereplőre méretezve rendszerint sokezer helyszínt és tárgyat zsúfolnak össze beléjük; jóllehet ötleteik többsége nem igazán eredeti. Szintén eléggé rossz tulajdonságuk még az is, hogy sablonos módon valamennyi inkább szerepjáték, mint kaland – ami azt jelenti, hogy különféle ellenségek leöldösése árán begyűjtött tapasztalati pontok segítségével lehet fölfelé lépkedni a ranglétrán, és ez a „fejlődés” képezi a játék tulajdonképpeni lényegét és célját. Léteznek belőlük magyar nyelvű alkotások is – egyszerűen borzalmasak. Nem azért, mert a TELNET-en nem lehet keresztülzavarni az ékezetes betűket, ez még rendben is volna; hanem a szövegek hemzsegnek a fogalmazási és helyesírási hibáktól – a parancsok szintaktikájáról nem is szólva… (Egyik legszerencsétlenebb húzás volt például az angol HELP helyettesítése a SÚGÓ-val és hasonlók…)

4.1.3.) A számítógép beszélni tanul

A kalandjáték „lelke” – azaz legfontosabb, legbonyolultabb és legtöbbet foglalkoztatott része – a beadott mondatokat értékelő programrész (parancsértelmező, szövegértelmező, értelmező, fordító, vagy angolul „parser”, „interpreter” stb.). Hiába a legkáprázatosabb szobor-kompozíció, ha az egy bevakolt szobában, a látogatók elől elzárva hever, úgyhogy kívülről senki sem férhet hozzá. Számunkra is egyik leglényegesebb kérdés, hogy a programunkban tárolandó világba miképpen kapjon betekintést a játékos. Egy kritikus megjegyzés szerint a kalandjátékok a „néma gyereknek anyja sem érti a szavát” közmondáson alapulnak, és ez eleinte valóban így is szokott lenni: ha valaki csak úgy leül egy ilyen játék elé, az leginkább egy üres beviteli mezővel találja magát szembe, és fogalma sincs róla, mit várnak el tőle. A más helyeken általánosságban kialakult szokások ugyan segítenek kissé eligazodni egy új programban is, de legjobb egy teljesen laikus felhasználót feltételezni. Sohasem tudhatjuk előre, hogy a – mienkétől esetleg teljesen elütő észjárású – kalandozónak miféle ötletei támadnak, ezért a játékok intelligenciájának fokmérője, hogy ugyanannak a szándéknak

a megnyilvánulásait milyen sokféle különböző formában képesek azonosítani. Például, ha egy ház kijárata egy délkeletre nyíló ajtó, akkor nem elég a programot csak a DK, DÉLKELET, MENJ DÉLKELETRE… stb. típusú parancsok fogadására fölkészíteni, elvégre a szemfüles játékos teljes lelki békével nyilatkozhat úgy is, hogy MENJ KI A KERTBE, HAGYD EL A HÁZAT, TÁVOZZ, LÉPJ ÁT AZ AJTÓN… – és még hosszan sorolhatnánk ugyanannak a mozgásnak a különféle megfogalmazásait. Minél több ilyen azonos értékű kifejezéssel elboldogul, annál színvonalasabb lesz a programunk, és a játékosok is annál jobban fogják élvezni! Régebben, amikor még a 16-, 48-, 64-kbyte-os mikroszámítógépek jelentették – a játékpiacon – az általánosan elfogadott szintet, elegendő volt azok teljesítőképességére hivatkozni, ami a játékok lehetőségeit is eleve behatárolta. Ma viszont, amikor egy PC nemritkán akár 32-64 MB RAM-ot is rejteget, s a processzorok sebessége is hihetetlen értékeket produkál, ráadásul mindez folytonosan egyre tovább bővül (és ki tudja, hová vezet mindez?!…), a fizikai korlátok egyszerűen megszűnnek létezni: egy kalandjáték színvonalát most már egyedül a készítője lustasága, vagy fantáziája, szókincsének bősége (vagy prózaibb okok közül: a készítésre rászánt idő…) határozza meg. De tény, hogyha egy ÜSD LE A RABLÓT mellett azt is magunkénak akarjuk tudni, hogy LÁSD EL A BAJÁT A RABLÓNAK, az nemcsak egyszerűen a szótár bővítését jelenti, hanem efféle szóvirágok kezelésével a kód is egyre bonyolultabbá válik…

Na de túlságosan előre szaladtunk (ez lett volna a bevezető ugyanis…): hogyan is szólunk egyáltalán egy ilyen játékhoz? Valamit ugyebár cselekedni óhajtunk benne, és ezt a közvetítő programnak az értésére próbáljuk adni: szándékunkat egy megállapodás szerinti nyelv szabályain keresztül kell megfogalmaznunk. Legjobb (volna), ha ez a nyelv mind közelebb áll(na) a természetes beszélt nyelvhez – de ahhoz, hogy ezt maradéktalanul megvalósítsuk, egy olyan „számítógépet” kellene építenünk, melyben – mint régi szélhámos sakkautomatákban – benne ül egy ember… Megközelíteni azonban lehet – pontosabban azt a látszatot kelteni, mintha megközelítettük volna.

Általános szabály, hogy mindinkább egyszerűnek, spontánnak és magától értetődőnek látszik valami a felhasználó szemszögéből nézve, annál nagyobb apparátust kénytelen a programozó megmozgatni hozzá, hogy egyáltalán működjön a dolog. A kívülálló számára természetes a nyelv, amelyen beszél, a legcsekélyebb szellemi erőfeszítés nélkül helyezi egymás mellé a szavakat. Fogalma sincs róla, milyen egy irdatlanul bonyolult rendszernek a birtokában van, s hogy a szerencsétlen programozó mennyit izzad vele, míg végül valami, ehhez úgy-ahogy hasonlót nagynehezen kiprésel magából. Különösképp érvényes ez a magyar nyelvre. Világszerte a kalandjátékok túlnyomó többségét angol nyelven írták. Ez nemcsak a nyugati régió egyik legáltalánosabb nyelve, de egyszersmind a legegyszerűbb nyelvtani szerkezetűek közé is tartozik – ezért is olyan könnyűszerrel megtanulható, mint köztudott róla. Az angol nyelv nem (vagy csak elvétve) használ ragokat, jeleket, maguk a szavak többnyire változatlanok maradnak, helyette a köztük fennálló viszonyt az erősen kötött szórend és az elöljárók mutatják meg. A főneveknek nincsenek nemeik, és még csak a magázódást sem ismeri. Összetett szavak képzése is úgy történik, hogy egyszerűen egymás mellé raknak két darab névszót, s legtöbbször még csak egybe sem írják őket (vagy ha mégis, akkor kötőjellel). Semminemű ékezetet nem használnak, az összetett hangzókkal sincs különösebb gond. Egyszóval, akárcsak az akácfa vagy a káposztalepke, az angol nyelv is alapvető igénytelenségének köszönheti azt, hogy világszerte elterjedhetett… Nézzük meg, hogyan hangzik angolul, ha valakit megkérünk rá, hogy nyissa ki az ajtót: OPEN

THE DOOR (esetleg még hozzátehetjük, hogy PLEASE – de ennek most semmi jelentősége nincsen). Az egész felszólítás két egyszerű szóból tevődik össze (a névelővel most nem foglalkozunk, azokat szimplán át lehet lépni): OPEN és DOOR, melyek – bármilyen összefüggésben is használjuk őket – minden esetben változatlanok maradnak, nem számítva az időnként a végükre kerülő „-s" betűt vagy „-ing" végződést, így egy szótárból kikeresve szemvillanás alatt azonosítani lehet őket. Ráadásul a mondat szórendje is szigorúan kötött (nem mondhatjuk pl. azt, hogy DOOR OPEN), tehát bizonyosan tudjuk róluk azt is, hogy az első az ige és a második a főnév. (Itt most csak a kalandjátékok irányításához felhasznált nyelvről esik szó.) Bővítsük most egy határozóval ezt a mondatot! Tegyük fel, hogy kulccsal szeretnénk kinyitni az ajtót. Ez esetben így hangzik: OPEN THE DOOR WITH KEY. A szórend most is kötött, és mindössze annyi a különbség, hogy a WITH elöljárót követő főnevet, mint eszközhatározót vesszük figyelembe, nem pedig, mint tárgyat.

Ugyanez magyarul már jóval keményebb diót jelent. Hogyan is fordítanánk le az előbbi példákat úgy, hogy értelmesen összefüggő mondatokat kapjunk? Valahogy így: NYISD KI AZ AJTÓT (ha netántán magázódunk a programmal, akkor a NYISD helyett NYISSA áll…), vagy ha nem parancsolgatni akarunk, akkor mondjuk KINYITOM AZ AJTÓT. De mi van akkor, ha NYISD KI AZ AJTÓT KULCCSAL helyett mi inkább úgy fogalmaznánk, hogy NYISD KI KULCCSAL AZ AJTÓT? Vagy: KULCCSAL NYISD AZ AJTÓT KI? Esetleg: AZ AJTÓT A KULCCSAL NYITOM KI?… Folytathatnánk a sort: valamennyi értelmes, és – a hangulati árnyalatoktól eltekintve – ugyanazt jelentik. A szórend itt már semmiféle támpontot nem nyújt, hiszen gyakorlatilag korlátlanul szabad – a szavak értelmét egymástól elszakítva, önmagukban kell megtalálnunk. De miképpen döntsük el, hogy egyáltalán melyik szóról van szó? Hiába szerepel a szótárunkban a NYIT ige, ha egyszer nekünk olyanokat írnak, hogy NYISD, NYISSA, KINYITOM, NYITOM… – és még vagy százféle különböző ragozott alak. Még ha attól eltekintünk is, hogy az igekötővel egybe- avagy különírjuk-e az igét, akkor is mennyi lehetőség marad! A főnevek ragozása ugyancsak egy cifra eset. Már maga a legalapvetőbb tárgyrag sem pusztán csak egy „-t" betűből áll: magánhangzós végű szavaknál ékezetet kap az utolsó betű (pl. alma – almát), néha kimarad egy hangzó (bokor – bokrot, tükör – tükröt), hosszú magánhangzók rövidülhetnek (ég – eget, tűz – tüzet), azután meg ott vannak a különféle kötőhangok is (-at, -et, -ot, -öt), minden szóhoz más-más fajta. Minden egyes határozónak saját ragja van: -ba, -be, -ra, -re, -n, -on, -en, -ön, -ból, -ből, -ról, -ről, -tól, -től, -nak, -nek, -hoz, -hez, -höz, -val, -vel, -vá, -vé, -kor, -ért, -ig… stb. stb. Ebből is van vagy negyven-ötven féle! (A KULCCSAL ráadásul még egy különleges eset is, hiszen -val, -vel esetén a „-v-" helyett duplázódik az utolsó betű, s miután „cs"-ről van szó, még csak nem is az „s", hanem a „c" betű…) Amennyiben emellé még a többesszám és birtokos eset jeleit is engedélyezzük, úgy az előbbi mennyiség ezek számával szorzódik (!!!), tehát végeredményben minden egyes főnévnek többszáz féle különböző ragozott alakja lehet! Ha a szótárunk mondjuk ezer szóból áll, akkor a kereséskor ez több százezer szóval való összehasonlítást jelent, ami már önmagában véve is egy képtelenség; nem beszélve arról, hogy lehetetlen volna ennyi alakot mind-mind letárolni. Megabyte-okat foglalna el, és percekig tartana megtalálni benne valamit…

A legegyszerűbb, korai, igénytelen játékokban még kikerülték ezt a problémát, ahelyett, hogy megbirkóztak volna vele. Valamiféle hallgatólagos megállapodással kialakítottak maguknak egy ún. „csonka magyart" – ezalatt a nyelvnek egy roppant

alacsony szintre csökkentett változata értendő. Az angol forráshoz igazodva, azonos sorrendű szópárokat várt el bevitel gyanánt a program, és kizárólag a szavak ragozatlan szótári alakját lehetett alkalmazni benne. Az idézett parancsok így hangzottak ezen a nyelven: NYIT AJTÓ, ill. NYIT AJTÓ KULCS… Némelyest fifikásabb, de éppilyen gépies és felszínes megoldásnak bizonyult az is, amikor a szavak első néhány betűje alapján kerestek a szótárban (többnyire az első háromtól az első ötig terjedt ez a hossz). Például egy program az első négy betűt vette figyelembe, s ezáltal a NYISS ABLAKOT utasításból ennyit látott csak: NYIS ABLA. A szótárban ennek alapján egységesen négybetűs jelsorozatok voltak tárolva, az ennél rövidebb szavaknak – vagy amelyek töve megváltozott, mint a NYIT ige esetében a „t” és az „s” betű – természetesen föl kellett venniük a ragozott alakjait is. (A CSOMÓ és a CSOMAG szavakat már nem is tudta volna megkülönböztetni egymástól.) Amennyiben igényesebb programokat szeretnénk írni, nem elégedhetünk meg ezekkel a primitív mankókkal! Meg kell tanítanunk beszélni azt a buta ócskavast…

Szükség lesz tehát egy algoritmusra, mely a szótárban külön-külön meglévő szótövek és ragok alapján dolgozik, és mindkét irányú működésre képes: egyrészt a begépelt szavakat szétbontja szótőre és ragokra (ezeket a programon belül majd sorszámmal fogjuk azonosítani), vagyis értelmezi azokat – másrészt fordítva: a megadott szónak előállítja a megadott típusú ragozott alakját. (Erre azért lesz szükség, hogy parancsunkra a számítógép válaszolni is tudjon, azaz ki tudja írni helyesen a képernyőre a neveket.) Figyelembe kell vennie bizonyos szavak különleges tulajdonságait is (épp az imént említettük őket: kulccsal, tükröt, nyisd…), és különbséget kell tennie az igék és főnevek teljesen eltérő ragozása közt. Fontos követelmény még, hogy megfelelően gyors legyen: elvégre a legegyszerűbb mondatok is min. három-négy szóból állanak, de mint később látni fogjuk, ennek akár a sokszorosát is kaphatjuk – a szótár pedig nyugodtan állhat akár többezer szóból is.

4.1.4.) A szavaktól a mondatok felé

A kalandjátékok utasításaira általában jellemző, hogy az alany hiányzik belőlük ill. rejtett (első vagy második személyű), s csaknem kivétel nélkül valamilyen igei állítmányra épülnek, amihez gyakran különféle tárgyak, jelzők és határozók csatlakozhatnak. Utóbbiak önmagukban egyszerűek, egy-két szóból állanak, s ami még nagyon jelentős egyszerűsítés, hogy alá-, fölérendelt *tagmondatok* sosem fordulhatnak elő. (Legalábbis ezidáig nem volt rá még példa…) Tipikusan egyszerű parancs pl. az alábbi: VEDD FEL A KULCSOT. Ebben föllelhető mindkét alapvető szófaj: az ige és a főnév – tulajdonképpen az összes többi szófaj fölfogható úgy is, mintha ezek speciális esetei lennének. Az ige fontos tartozéka (vagy inkább: része) még az igekötő is, hiszen ha azt mondjuk, hogy VEDD FEL, VEDD LE vagy VEDD MEG, azok teljesen eltérő cselekvésre utalnak. E kettő együttesen dönti el, hogy mit akarunk csinálni, a főnevek és társaik pedig a cselekvésünk célját határozzák meg, azáltal, hogy az elérhető tárgyak és szereplők közül konkrétan kijelölnek valamit. Ez nem mindig egyértelmű: egyrészt létezhet a játékban több hasonló nevű tárgy is, másrészt használhatunk különféle gyűjtőneveket. Előbbi esetre példa lehet az, ha többféle különböző kulcs van: ilyenkor mindegyiknek van valamilyen sajátos jelzője, mely megkülönbözteti az összes többitől (pl. „kis kulcs” vagy „nagy kulcs” stb.); a főnevek elé melléknevek járulnak (VEDD

FEL A NAGY KULCSOT). Gyűjtőnév esetén pedig az illető szó ugyan pontosan megjelöli a tárgyat, ám egyszerre akár többet is – legjobb példa erre a MINDEN szónak az alkalmazása: egy VEGYÉL FEL MINDENT utasítás egyszerre vonatkozik a közelünkben látható valamennyi tárgyra. Melléknevek szintjén is elképzelhető hasonló, pl. így: VEDD FEL AZ ÖSSZES KULCSOT. (Hogy miért is célszerű a melléknevet is speciális főnévként nyilvántartani, az kitűnik abból, hogy alkalomadtán ezek is a főnevekhez hasonlóan ragozódhatnak. Egy jólnevelt kalandjáték ugyanis, ha nem egészen biztos a dolgában, további kérdéseket tesz fel: megkérdezi tőlünk, hogy *melyik kulcsot* szándékozunk fölvenni. Erre mi felelhetjük azt is, hogy a NAGY KULCSOT – vagy egyszerűen ennyit: NAGYOT. A melléknév ilyenkor kifejezetten úgy viselkedik, mintha főnév lenne; fordított esetben pedig egyes főnevek is állhatnak a melléknévi jelzők helyén.) Mivel igencsak ritka eset (úgy értem, egy kalandjátékban…) az, hogy teljesen azonos tárgyakból legyen több példány ugyanazon a helyen, s ráadásul ezek közül többre, de nem valamennyire akarunk hivatkozni egyidejűleg, ezért számnevek csak elvétve fordulnak elő; ilyenkor, mint sejthető, a melléknévhez hasonlóak, csak funkciójuk más. Ugyancsak fehér holló egy másikfajta jelzős szerkezet, nevezetesen a birtokos jelző: egyfelől erre is csak viszonylag ritkán kerül szükség, másfelől meg akkor is kiválthatjuk más, egyszerűbb jelzőkkel a szerepét. Pl. ha az iménti nagy kulcs egy őr kezében van: egy VEDD EL AZ ŐR KULCSÁT helyett a FOGD A NAGY KULCSOT utasítás sokkal egyszerűbb. Jóllehet ezek kezelése eléggé bonyolult eljárást igényel, mindenesetre sokat emel egy játék színvonalán, ha – mint értékes egzotikumot – ilyesmit is beleépítünk. Annál nagyobb igény lehet ezzel szemben (márminthogy a játékosok oldaláról) egy másik, teljes főnévi értékű szófajra, a névmásra: ha leírjuk, hogy BESZÉLGESS *VELE* vagy NÉZZ *RÁ*, miközben a legutóbbi parancsunk egy KÖSZÖNJ A MADÁRIJESZTŐNEK volt, ez kétségkívül azt jelenti, hogy ezúttal is a madárijesztőre gondolunk – az utolsóként előfordult főnevet helyettesíthetjük valamilyen névmással. Érthetetlen, hogy ezt is csak alig néhány program ismeri, amikor pedig a megvalósítása egyszerű (hisz' mindössze egy szócseréről, behelyettesítésről van szó), és igen nagy könnyebbség, ha nem kell a hosszadalmas MADÁRIJESZTŐ szót mindig újra és újra bepötyögni, hanem helyette ezt a „kézreálló" pár betűt…

A határozók már nem önálló szófajok, hanem csak mondattani egységek – lényegében a főnevek eltérő ragozású alakjai. Miután a szórend szabad, és más támpontunk nincsen, egyedül a ragozásuk alapján tudjuk csoportosítani a különféle kifejezéseket. Túlságosan körülményes és fölösleges is volna azonban valamennyi raghoz külön határozót rendelni, így legcélravezetőbb öt csoportba sorolni őket (az egy csoportba tartozó ragok nemigen szoktak előfordulni egy parancsban együtt, és körülbelül hasonló dolgot jelentenek, így nyugodtan tekinthetjük azonosnak őket): 1.) tárgy: -t raggal vagy rag nélkül; 2.) helyhatározó: -n, -ban, -nál, -ról, -ból, -tól; 3.) célhatározó: -ra, -ba, -nak, -hoz; 4.) eszközhatározó: -val; 5.) egyéb: -ért, -ig… stb. Pl. az ADJ PÉNZT A KOLDUSNAK utasítás három mondatelemet tartalmaz: egy igét, egy tárgyat és egy célhatározót (az, hogy ezt a nyelvtanban részeshatározónak nevezik, ne zavarjon túlzottan bennünket…). A ragokon túl használhatunk névutókat is a főnév jelentésének módosítására, így helyhatározó lehet az is, hogy ASZTAL ALATT, vagy a KULCCSAL helyett eszközhatározó, hogy KULCS RÉVÉN vagy KULCS SEGÍTSÉGÉVEL stb. stb.

Összefoglalva tehát, minden egyes begépelt parancs lényegében a következő részekből tevődik össze: az ige (a beleértett igekötőjével együtt), valamint a felsorolt

ötféle tárgy ill. határozó. A parancsot akár fölfoghatjuk úgy is, mint egy rekordot, aminek ez a hatféle mezője van. A programon belül persze a szavak nem szövegként jelennek meg, hanem minden egyes dolgot valamilyen sorszám azonosít. Pl. ha a „kinyitni" ige a 13-as sorszámot kapta az előzetes tervezés során, akkor ez azt jelenti, hogy az ige változójába egy 13-as szám kerül. Amelyik fajta eleme hiányzik a mondatnak, az természetesen a 0-ás értéket kapja. Ez gyanúsan egyszerűnek látszik, holott valójában egyáltalán nem az: ugyanis ez a végeredményül kapott egyszerűség a valóságban egy hosszú és bonyolult értelmezési folyamatot takar, melynek folyamán a különféle szókapcsolatok és más szerkezetek fokozatosan a lehető legtömörebb viszonyokká bomlanak le. Ahhoz, hogy ez megvalósuljon, tucatnyi eltérő algoritmust kell egymás után lefuttatni, amelyek mindegyikének feladata a megfelelő szavak vagy szópárok cseréje, *helyettesítése* valamilyen másik, egyszerűbb szóval. Ezek a cserék táblázatokban való keresések útján történnek. Efféle algoritmus lehet például az, amelyik az igék és igekötők különálló párosát egyetlen összetett igére cseréli ki (a NYISD és a KI szavakat törli, és helyükre egy KINYITNI szót ír – ha pedig nemlétező párosítást talál, pl. azt írtuk, hogy NYISD ÖSSZE, ami nincs benne a táblázatban, akkor hibát jelez, és az ige értelmezhetetlen). Vagy egy másik, mely a névmás helyére elegánsan becsempészi az utolsóként alkalmazott főnevet. Egy olyan, mely a „melléknév + főnév" szókapcsolatot egyetlen új főnévvel helyettesíti be (a NAGY és a KULCS szavak sorszámait kiiktatva, helyettük egy másik, a NAGY KULCS szó sorszámát illeszti a mondatba). És így tovább; minél több ilyen funkcióval ellátjuk, annál intelligensebb lesz a programunk. (És annál nehezebb lesz megírni…) Természetesen az sem mindegy, milyen sorrendben hajtjuk végre ezeket az egyes szabályokat a mondatban. Pl. ha azt találjuk leírni, hogy NAGYOT, akkor az először is felbomlik a – kissé idegenszerűen hangzó – NAGY AZT szókapcsolatra, majd az AZ helyére bekerül a KULCS főnév (mint névmás-helyettesítés!), s csupán ezután lehet belőle NAGY KULCSOT. Egy szöveges kalandjáték értelmező programját elkészíteni körülbelül hasonló feladat, mintha mondjuk egy PROLOG fordítóprogramot óhajtanánk írni…

A *parancs* és a *mondat* kifejezéseket az eddigiekben egy kissé összekeverten használtuk, úgyhogy lassanként ideje lenne szétválasztanunk őket egymástól: míg egy parancs mindig egyetlen meghatározott cselekvésre utal, vagyis lényegében tagmondat (a fentiekben hozott példák mind parancsok voltak), addig egy mondat nyugodtan akár több parancsból is állhat. (Már amelyik program hajlandó ezt tudomásul venni…) Vagy az ellenkező irányból megközelítve a dolgot: egy mondat az a szövegmennyiség, amit a játékos egy szuszra begépel a bevitel során, s a parancs ennek egy-egy önállóan végrehajtható része. A mondaton belüli parancsok elválasztására írásjeleket (pont, vessző és társaik), valamint az ÉS szócskát vagy annak szinonimáit (MAJD, TOVÁBBÁ, VALAMINT… stb.) használjuk. A játék a mondatban lépésről lépésre halad előre; mihelyt egy parancsot végrehajtott, utána kezdi el értelmezni a következőt. Pl. egy tipikus összetett mondat így hangzik: VEDD FEL A KULCSOT ÉS NYISD KI VELE AZ AJTÓT. Ebben az esetben a két tagmondatot akár egy-egy külön mondatként is le lehetne írni, de előfordul olyan is, hogy szorosabb összefüggés kapcsolja őket össze, miáltal szétválaszthatatlanok lesznek – ez akkor állhat fenn, mikor a második parancsból kihagyunk valamit, amit már tartalmaz az első. Pl.: GYÚJTSD MEG A GYERTYÁT ÉS A MÉCSEST. Itt a második parancsból kimaradt az ige. Ahhoz, hogy ezt értelmezni tudjuk, egy aprócska trükköt kell alkalmazni a programban, nevezetesen

amíg a mondatnak nincs vége, addig az aktuális parancs változóit sohasem töröljük, hanem csak az újabbik parancs szavai mindig felülírják a régit – így tehát a teljes első parancs változatlanul megmarad, mindössze a MÉCSES, mint tárgy fogja felülírni az előző tárgyat, a GYERTYÁ-t. Ezáltal egyetlen állítmány mögé tetszőleges számú tárgyat fölsorakoztathatunk. Sokkalta nehezebb volna – éppenhogy a lépésenkénti végrehajtás miatt! – ugyanezt az ellenkező irányból is megcsinálni, valahogy ilyképpen: NYISD KI ÉS CSUKD BE AZ AJTÓT. Ebben az esetben ugyanis a programnak szinte „előre kéne gondolkodnia": amikor odáig jut, hogy NYISD KI, még fogalma sincsen róla, hogy mire is vonatkozik ez a dolog – jóllehet a keresett főnév benne van a mondatban, csak jóval később volna esélye megtalálni azt. Az eredmény: a számítógép megkérdi, hogy „Mit akarsz kinyitni?". Ennek feloldására be lehetne vezetni egy ún. öröklési rendszert, ami abból állna, hogy még mielőtt hozzálátna a legelső lépés végrehajtásához, a program végigértelmezné a teljes mondatot, és mindegyik tagmondat állapotát eltárolná valahol, egy rekordokból álló láncban úgy, hogy az aktuális rekord mindig örökölné – az iméntiekben leírt módon – az előzőnek azt a részét, ami nem változott. Azután ugyanezt végigcsinálná fordított irányban, az utolsótól az első felé haladva is – és ami valahonnan hiányzik, azt mindig a másikból vett megfelelő részekkel pótolná és kiegészítené benne ekkor is. Eközben az első parancsba, ahonnan hiányzik a tárgy, beíródna az utána következő tagmondat tárgya (ami esetleg szintén a következőből örökölte azt, és így tovább). Csupán ezt követően hajtódnának végre a parancsok.

Amivel aztán el is érkeztünk az értelmezés következő állomásához: mi történjék, amikor valamilyen szó végképp hiányzik a mondatból. Erre több megoldás is kínálkozik. Lehet egyszerűen a felhasználó képébe vágni, hogy márpedig ez nem egy értelmes mondat, és legközelebb szíveskedjék jobban megválogatni a szavait… Ennél valamivel udvariasabb (és intelligensebb) megoldás, amikor a homályos részekre vonatkozó kérdéseket teszünk fel, és további kiegészítéseket várunk. Pl. ha csak annyit kaptunk parancsként, hogy NYISD, akkor feltesszük neki a következő kérdést: „Mit akarsz kinyitni?" – ami után következő bevitelként kétféle felelet várható. A játékos vagy begépel egy teljesen új mondatot, nagylelkűen elfelejtve az előzőt – vagy pedig a kérdésünkre válaszol, és mindössze a hiányzó tárgyat adja meg. Hogyan készítsük fel a játékot a helyes reagálásra mindkét esetben, amikor e kettőt tulajdonképpen nem is igazán lehet megkülönböztetni egymástól? Hát úgy, hogy nem is különböztetjük őket meg. Az előbb láttuk, hogy az esetleg összefüggő részmondatok miatt a parancsok változóit mindig a mondat végén töröljük, amikor egy új mondatba kezdünk. Egyszerűen annyit kell tennünk, hogy mikor valamilyen kérdést intézünk a kalandorhoz, olyankor az újabb bevitel megkezdése előtt *nem* töröljük ezeket a változókat, azaz nyitva hagyjuk a mondatot – akármit begépel a tisztelt felhasználó, az a jelenleg meglévő parancsot fogja felülírni, éppúgy, mintha az a következő parancs volna. Teljesen mindegy, hogy töredékeket kapunk-e tőle, vagy egy új mondatba kezd. Így akár több, eltérő tartalmú kérdés is követheti egymást, és mégis valamennyi válasz ugyanazt a parancsot fogja fokozatosan gazdagítani – szabályos kis párbeszéd alakulhat ki a program és a felhasználója között. Komolyabb probléma akkor adódik, ha már bevezettük az előzőkben vázolt örökléses rendszert – ilyenkor ugyanis a kérdés feltevése előtt meg kell jegyezünk az egész mondat állapotát, majd a kiegészítés megtétele után ismételten végigfuttatni rajta oda-vissza a parancsokat összefűző eljárást.

Harmadik, és egyben legintelligensebb megoldása a hiányos mondatok esetének, ha a program maga automatikusan megkísérel behelyettesíteni a kimaradt részek helyére valamit, és ha csak ez végképp nem sikerül neki, akkor teszi fel a kérdést. Pl. a NYISD megadása esetén sorjában végignézi az összes elérhető tárgyat, s csak ha nincs közöttük egyetlen olyan sem, ami nyitható, akkor kérdi meg a kezelőjét, hogy voltaképpen mire is gondolt. (Vagy ha többet talált, akkor azt, hogy melyiket.) Hasonlít ez némiképpen a gyűjtőnevek kezeléséhez. Leggyakoribb gyűjtőnév a korábban már megemlített MINDEN szó: ilyenkor a parancs végrehajtását egy külső ciklusba kell ágyazni – sorban egymás után behelyettesíteni a kívánt szó helyére az összes elérhető tárgynevet, és mindegyikkel egyesével végrehajtani ugyanazt a parancsot. (Érdemes elgondolkodni rajta, hogy mi történik akkor, ha netán a huncut játékos egyazon parancson belül *többször* is alkalmazott valamilyen gyűjtőnevet. Pl. így: MUTASS MEG MINDENT MINDENKINEK. Ez az, ami a programozó idegeit végképp próbára teszi…)

4.1.5.) Helyiségek összefüggő labirintusa

Említettük, hogy a kalandjáték életének teljes színtere egyes, különálló helyiségekre tagozódik. Ez valahogy úgy néz ki, hogy tartózkodunk valahol, pl. egy szobában, és érzékelésünk azokra a dolgokra korlátozódik, amelyek szintén ugyanazon a helyen vannak – egészen addig, míg át nem haladunk valamelyik szomszédos területre. Hogyha a jelenlegi szobában hever egy asztal, akkor azzal, mint tárggyal, elméletileg megtehetünk bármit (megvizsgálhatjuk és hasonlók), de ha valahol másutt leírjuk, hogy ASZTAL, akkor már az alábbihoz hasonló válasz érkezik: „Nincs itt semmiféle asztal.” Miképpen lehet ezt a világot a számítógépen keresztül leképezni a játékosok számára? Minden helyiség egyértelműen azonosítható és a többitől megkülönböztethető kell legyen, és a játékban szereplő valamennyi objektumot (tárgyat, élőlényt, beleértve minket is!) pontosan el kell helyezni valahol a térképen, úgy, hogy annak helyzete változtatható is legyen. Le kell írni a csatlakozó helyiségek viszonyát, a mozgást gátló akadályokkal és más tényezőkkel együtt.

Természetesen hűek maradunk eddigi alapelvünkhöz: mindenhol sorszámok és táblázatok sarjadnak a lépteink nyomán. Egy jól szervezett kalandjáték-világ valójában igen összetett rendszert alkot, annak megfelelően, amit a filozófiában is előszeretettel hangoztatni szoktak: minden összefügg benne minden egyébbel… Emiatt is olyan nehéz eldönteni, hogy egyáltalán hol kezdjen neki az ember. A parancsok és cselekvések végrehajtásának kidolgozásába bele sem foghatunk addig, amíg a játék teljes leendő világának leíró adatai holmi adatbázis-szerűen nincsenek letárolva a gépben. Egy nagyobb lélegzetű műnél ez hosszú hónapokig tartó tervezést és előkészületeket igényel, mielőtt még egyetlen sort is leírnánk bármilyen programozási nyelven. Igazából ez a munka legkellemesebb része, hiszen – játékról lévén szó – jórészt álmodozással, ötleteink lejegyzésével és összerendezésével, az epizódok és jelenetek kiagyalásával telik el. Azonban fabatkát sem érne az egész, ha nem mindjárt abban a formában kezdenénk el az adatok gyűjtését, ahogyan azt majd a későbbi programban felhasználni fogjuk…

Legelső feladatunk, hogy fölépítsük a helyiségek rendszerét. Ehhez legjobb, ha fogunk egy darab üres papírost, és térképet rajzolunk a képzeletünkben fölsejlő világról.

Ez a térkép egy csöppet elvont megjelenéssel bír a szokásos atlaszokhoz képest: a helyszíneket vázlatos körök jelképezik, az egyikből a másikba vezető utakat a közéjük húzott vonalak. Mindegyik helynek valamilyen nevet adunk (pl.: „Erdei tisztás”), ezt a megfelelő karika belsejébe írjuk; s hogy két-két helyiség között melyik irányban lehet közlekedni, azt a köztük feszülő vonal végére biggyesztett nyíllal ábrázolhatjuk. A nyíl tövébe írjuk azt is, hogy mi módon kell mozognunk a jelzett helyváltoztatáshoz: ennek jelzésére – megállapodás szerint, de persze nem kötelező jelleggel – az iránytű nyolc égtája valamint a két függőleges irány szolgál. Pl. ha a lakásunk szobáit szeretnénk ábrázolni, ami öt helyiségből áll, akkor a papírra öt darab kört húzunk, a szobák állásának megfelelő elrendezésben, beléjük írkálva egy-két szavas elnevezéseiket; ha mondjuk a hálószoba a nappalitól nyugatra fekszik, akkor a „Nappali” feliratú buboréktól a „Hálószoba” feliratúig rajzolunk egy nyilat, tövében egy NY betűvel, s miután nyilván visszafelé is vezet út, a vonal ellenkező végét is nyíllal látjuk el, a másik tövébe egy K betűt írva. Ha a két szoba közt ajtó is nyílik, a vonal közepe táját kiegészítjük egy kisebb téglalappal, belevezetve az „Ajtó” szót… Azután ezt a módszert végigvisszük a termek egész hálóján. (Tág kiterjedésű, szabadtéri helyeknél élhetünk azzal a trükkel, hogy önmagába visszamutató nyilakat rajzolunk – ameddig a játékos a jelzett irányban gyalogol, folyton ugyanabban a térben marad.) Amerre egy helyről nem vezetnek nyilak, ott fal van, vagy más hasonló áthághatatlan akadály. Nagyobb térképeket célszerű több részre bontva elkészíteni, hogy áttekinthetőek legyenek.

Idáig minden pofonegyszerű, bárkinek eszébe juthatott volna mindez – csak kár, hogy a számítógép egy összefirkált papírlappal semmit sem tud kezdeni. Le kell fordítanunk a számok nyelvére az egészet. Kezdjük el ezt azzal, hogy valamennyi helyünknek egytől növekvő rendben sorszámokat osztunk: a hálószoba lesz az 1-es, a nappali a 2-es stb. – írjuk be ezeket a megfelelő buborékokba, a névszövegek mellé. A választható irányokhoz is rendeljünk hasonlóan sorszámot; a felsorolt tíz égtáj mellé fölvehetjük még a KI és BE irányokat is, úgyhogy összesen tizenkét irányunk lesz; 1-től 12-ig számozzuk meg őket – mondjuk az 1-es jelentse az északot, és így tovább. Ezt követően a térképet már táblázattá lehet alakítani: olyan elrendezésben, ahol baloldalt függőlegesen a helyiségek sorszámait tüntetjük fel sorban, vízszintesen pedig az irányok helyezkednek el. A táblázatban minden helyhez tartozik egy sor, ezeket úgy kell kitölteni, hogy a sor mindegyik rubrikájába annak a másik helyszínnek a száma kerül, ahová a hozzá tartozó irányban az adott szobából kimozdulva érkezünk. Ha a jelzett irány sehová sem vezet (lezárt irány – fal), akkor ide nullát írunk, ha önmagába kanyarodik vissza, akkor a saját helyszín száma kerül ide is. Ha arra vagyunk kíváncsiak, hogy a nappalitól (ez most ugye a 2. helyiség) észak felé (1. irány) mi található, akkor egyszerűen kiolvassuk a táblázat 2. sorának 1. oszlopát, ahol tegyük fel, egy 4-es számot találunk (legyen ez az előszoba száma): a nappalitól északra az előszoba vár ránk. Ezt számítógéppel kezelni igen egyszerű lesz később.

4.1.6.) Barangolás a térképen

Mihelyt ezzel megvolnánk, ne felejtsünk még egy második táblázatot is készíteni, az egyes utakat lezáró akadályok nyilvántartására. Mondanom sem kell, hogy ezek is tipikus sorszámokat kapnak, ám itt már nem árt némi rendszert is vinni a dologba: ha valamiből több hasonló van, úgy célszerű azokat egy csoportba venni, még ha távol

esnek is egymástól (pl. 1-től valameddig számozzuk az ajtókat, utána jönnek sorjában az ablakok stb.); ezáltal a későbbiekben megkönnyítjük ezek kezelését. Az egy-egy akadályhoz tartozó sorokban itt a következő adatokat szükséges feltüntetnünk: két-két rubrika az általuk elválasztott két-két helyiség számára (pl. a nappaliból a hálóba nyíló ajtó esetén egy 1-es és egy 2-es), valamint egy harmadik mező, ami pedig az akadály jelenlegi állapotát tükrözi. Ezúttal is számokkal kell élnünk: legalább három értéket használjunk (célszerűen 0-tól 2-ig számozva) annak jelzésére, hogy az ajtó nyitva, csukva vagy zárva van-e – de ennél jóval több lehetőség is elképzelhető (lehet mondjuk félig behajtva vagy betörve az ajtó, vagy függöny esetében behúzva vagy elhúzva, akármi az eszünkbe juthat!). Végül hagyjunk még egy negyedik oszlopot is saját megjegyzéseink részére: ide lehet beírni az akadály nevét (ajtó, kapu stb.), és esetleg még azt is, hogyan kezelődjék később, azaz miképpen nyitható (milyen cselekvést kell véghezvinni az állapota megváltoztatásához – mondjuk milyen kulccsal nyitható egy zárt ajtó). Listánkba fölvehetünk olyan „elvont” akadályokat is, mint egy kerítés, vagy az utunkat álló eleven őrszem – csak vigyázzunk ez utóbbi esetén, mert ő az ajtókkal ellentétben nem két, hanem egyszerre csak egy helyiségben létezik, s így másként kell majd kezelni később. Egyáltalán, az akadálylistába fölvenni azokat a tényezőket érdemes, amelyeknek állapota változik – ha egy falon sohasem juthatunk át semmilyen módon, azt egyszerűbb inkább a másik táblában a megfelelő irányhoz, mint valami fiktív, valójában nemlétező helyszínszámot beírni.

Most már megtehetjük első – egyelőre csupán képzeletbeli – sétánkat kezdetleges szobáink kis színpadán. Azt is látjuk, hogy – noha még csak két táblázatunk van – máris mennyi mindent kéne ellenőriznie egy vezérlő programnak olyan, primitívnek látszó művelet elvégzéséhez, mint egy meghatározott irányban való haladás. Először is – jelenlegi pozíciónk tudatában – a bejárási táblázatból ki kell olvasnia az elérendő célhelyiség számát. Meg kell néznie, az akadályok listájában szerepel-e olyan objektum, melynek összekapcsolt két helyisége éppen az általunk igénybe vett két szobával egyezik-e meg (ráadásul duplán is kell vizsgálódnia, mert hátha az ellenkező irányból jövünk, a 2-esből az 1-esbe, s nem pedig az 1-esből a 2-esbe), majd ha talált egy ilyet, akkor tájékozódni róla, nyitva van-e az akadály. Ha nem járható az az út, figyelmeztető üzenetet kapunk („Az ajtó csukva van.”), ha igen, akkor további ellenőrzések következnek: létezik-e egyáltalán abban az irányban bármi. Nullás célhelyiség esetén: „Nem mehetsz abba az irányba.” – de lehetnek további fiktív sorszámok is, melyek például külön jelölik a falat, a sövénykerítést, a vizesárkot stb. Sőt, akár még halálos irányokat is kiképezhetünk – mondjuk, hogy egy veszélyes hegyi úton az északnyugati irány végzetes lezuhanással végződik… Csak ha minden stimmel, akkor szabad elvégezni a tényleges mozgatást. (Hogy hogyan, azt is hamarosan meglátjuk.)

Egy kalandjátékban szereplő bármilyen élőlény alapvetően négyféle mozgási formát követhet a helyszínen történő kalandozása folyamán. Ezek: a közvetlen haladás, a közvetett haladás, egy távoli célpont megközelítése és az ún. „mágikus” bejárás. Közülük az első három mindegyike az őt megelőzőre épül, és egyre magasabb szintű cselekvési módokat tesznek elérhetővé.

1.) Közvetlen (vagy magyarul: „explicit”) haladás esetén olyan mozgási utasításról van szó, mely pontosan meghatározza az illető által követendő haladási *irányt*. Az ezt kiváltó parancsok például a következőképpen hangzanak: DÉLKELET vagy MENJ BE A KUNYHÓBA stb. (Ez utóbbit az teszi közvetlenné, hogy szerepel benne a BE szócska! Máskülönben a következő csoportba tartozna – és akkor is, ha a

BE irányt nem vettük volna fel az irányok közé az előző fejezetben.) Hogy ezek megvalósítása miképpen történik, azt épp az imént magyaráztam el, a továbbiakról pedig a következő fejezetben esik némi szó.

2.) A közvetett („implicit”) mozgás ezzel szemben a kalandozó részéről egy olyan kívánságot jelent, amelyben nem fogalmazza meg konkrétan, hogy melyik irányban szeretné a következő lépést megtenni – hanem csak bizonyos utalásokat tesz, az elérendő *célpont* megjelölésével. Ez már nehezebb feladat a programunk számára, hiszen őrá hárul az irány kiválasztásának felelőssége: a mondatban megadott célról el kell döntenie, melyik irányban fekszik tőlünk. Ha sikerült eldöntenie, akkor a lépés már egy egyszerű közvetlen mozgássá redukálható. Ilyenek rendkívül sokfélék lehetnek, és ráadásul valamennyiük egyéni bánásmódot igényel. HASZNÁLD AZ AJTÓT típusú felszólítás, pl. MÁSSZ KERESZTÜL A CSAPÓAJTÓN esetében az akadályok táblázatából ki kell keresni a helyszínről kivezető csapóajtót (ha van olyan), azután ennek másik helyiségét kell megkeresni a bejárási táblázat megfelelő sorában – ezzel megvan a követendő irány (ha esetleg több irány is vezet ugyanoda, a program dönthet, melyiket használja); LÉPD ÁT A KÜSZÖBÖT: majdnem ugyanaz, mint az előző, azzal a különbséggel, hogy miután küszöbe többféle eszköznek is lehet, ha ugyanott van mondjuk egy ajtó és egy kapu, akkor megintcsak el kell döntenünk azt is, hogy melyiket koptassuk; MÁSSZ FÁRA: egyértelműen a FEL irány helyettesítése; UGORJ: valószínűleg a LE irányt jelenti, de helyzettől függően egészen más is lehet… Fölösleges sorolni a példákat tovább, a dolog lényege ennyiből is látható.

3.) Az eddigi két mozgásfajta egyszerű volt abból a szempontból, hogy mindkettő egyetlen lépés megtételére korlátozódott – valamelyik szomszédos terembe ruccanhatott át segítségükkel a játékos. Távoli célpont megközelítése esetén ugyanakkor már egy teljes *útvonalat* szükséges végigjárnia, méghozzá úgy, hogy ennek kiválasztását is legnagyobb mértékben a számítógépre hagyja: MENJ A KASTÉLYHOZ, KERESS EGY TISZTÁST, KÖVESD A VÁNDORT típusú parancsok ilyenek. Mint valami taxiba szálló utazó, egyszerűen közli velünk, hová óhajt elkeveredni, és attól fogva a játék idegenvezetésére bízza magát. Ez a legmagasabb szintű, legelvontabb és legnehezebben kivitelezhető haladási forma, a kalandjátékok túlnyomó többsége teljes mértékben nélkülözi. Van azért egy kicsike előnye is a közvetett mozgással szemben, az, hogy nem kell olyan sokféle eltérő kivétellel vacakolni, hanem egyetlen általános eljárásba bele lehet foglalni az egészet. Az viszont a maga nemében szép egy darab… Ismerjük ugyebár a teljes világ térképét a rajta elszórt akadályokkal egyetemben, a saját pozíciónkat és azt, hogy hová kell majd eljutnunk: meg kell találnunk az egyiktől a másikhoz vezető legrövidebb (?!) utat. Ha egy ember rápillant egy térképre, a megoldás szinte önmagától adódik – de egy számítógépnek ugyanezt a diót már sokkal nehezebb lesz föltörnie. Bonyolítja (vagy egyszerűsíti?) a helyzetet az is, hogy esetleg még magáról a célhelyiségről sincs pontos tudomásunk; utóvégre, ha egy játékban vagy harminc ajtó lézeng itt-ott elszórtan, és a tisztelt játékos megsúgja nekünk, hogy MENJÜNK AZ AJTÓHOZ – akkor mégis melyikre gondolt?! Nyilván arra, amelyiket leghamarább elérjük – ha legrövidebb útvonalat keresünk, akkor ez egyszersmind a legközelebb lévő. Nincs más megoldás: el kell indulni, lépésről lépésre tapogatózva előre a helyiségek végtelen óceánjában, s minden új helyszínre lépve ellenőrizni, hogy találtunk-e a leírásnak megfelelő tárgyat. Persze nem összevissza, hanem módszeresen: sorban végigpróbáljuk mind a tizenkét irányt, s amelyik járható, arra lépünk tovább; ha végképp zsákutcába futottunk, úgy mindig visszalépünk az ezt megelőző helyre, és a

következő iránnyal kísérletezünk. Ha visszaértünk a kiindulásra, akkor kudarcot vallottunk: a célpontot nem tudjuk elérni. Namost ehhez három dolog szükségeltetik: először is maximalizálni kell a megtehető legnagyobb távolságot, mondjuk 30-40 lépésben (minden újabb lépéssel hatványozódik a lehetőségek száma! – ennyi pedig bármekkora térképhez bőségesen elég lesz); másodszor egy listában nyilván kell tartanunk az eddig megtett útvonalat (melyik helyszínről merre haladtunk tovább), hogy legyen hová visszalépnünk és tudjuk, merre kell majd mennünk; harmadszor pedig – ugyancsak e lista alapján – folyton ellenőriznünk kell, nehogy egy, már érintett helyszínre másodszor is rálépjünk (ilyenkor úgy tekintjük az arrafelé nyíló irányt, mintha akadály zárná el – egyedül így kerülhetjük el, hogy önmagába visszatérő hurkokat írjunk le). Ezzel a módszerrel szépen meg is találjuk a keresett célpontot, csak éppenséggel egyetlen bökkenő lesz vele: korántsem biztos, hogy a legrövidebb útvonalon jutottunk el oda! Ahhoz, hogy amazt is fölleljük, kénytelenek vagyunk a keresést először egyetlen lépésre szűkítve elvégezni, s ha nem találjuk, akkor fokozatosan (mindig egy-egy lépéssel) hosszabbra engedni a pórázt, minden ilyen bővítés után újra és újra végigkeresve – egészen addig, míg el nem jutunk a célba, vagy el nem érjük a bűvös határt. Lassú géppel jobb nem belekezdeni… A KÖVESD ige (egy közelben mozgó illetőnek a nyomába szegődni) a közönséges MENJ-től annyiban eltér, hogy hatósugarát már eleve csak 2-3 lépésre kell zsugorítani – azonkívül, ha sikeres volt, addig ismételni, míg az „üldözött” személy meg nem állapodik valahol. Ha igazán *nagyon* intelligens programmal akarjuk elkápráztatni a közönségünket, ezt az egészet esetleg még megpakolhatjuk olyan finomságokkal, hogy gyaloglás közben az emberkénk reflex-szerűen nyissa ki az útjába eső ajtókat (majd ha áthaladt rajtuk, csukja is be maga után őket…), sőt, ha zárva vannak, először próbálja végig bennük a nála levő kulcsokat, majd kopogtasson rajtuk, és így tovább…

4.) A negyedik mozgás, az ún. „mágikus bejárás” (DaCosta), nem egy nagy szám, semmi köze az előző háromhoz, és jól hangzó nevével ellentétben mindössze azt jelenti, hogy a szokásos útvonalakat mellőzve, egycsapásra egyik helyről a másikra kerülünk. Tulajdonképp semmi hagyományos – irányokhoz kötődő – mozgás nem történik ekkor. Mindez történhet valamilyen bűbájos varázsige kimondásával, esetleg hasonló módon, mondjuk egy varázsló elteleportál bennünket máshová. Ide sorolható az az eset is, amikor valamilyen, a valóságban hosszabb műveletsort igénylő tevékenységet a játék – dramaturgiai megfontolásból – egy lépésbe sűrítve hajt végre. Például megfizetünk egy kocsist, hogy fuvarozzon el minket, s erre a program tényleges utaztatás helyett mindössze pár mondatban ecseteli számunkra hosszú utunk viszontagságait – aztán a következő lépésben már éppen kiszállunk a kocsiból a célnál. Jellegéből adódóan ezt a legkönnyebb megvalósítani.

4.1.7.) Lakberendezővé változunk

A játék alapkövét letettük, világunk színterének hálózata megvan – de rajtunk kívül senki másnak nem telne benne túl sok öröme, hogy az üres helyszínek közt mászkál. A Teremtés csak félkész állapotban van: még föl kell húznunk a falakat, bevakolnunk őket, s a szobákat berendezni díszletekkel, tárgyakkal és élőlényekkel. E három dolog nem is különbözik egymástól annyira: ugyanabba a számozási rendbe bele lehet vonni az összeset, s hogy aztán melyik objektum viselkedik mozdíthatóként vagy

élőként, azt már csak a konkrét programműködés dönti el, a megfelelő sorszámokra reagáló eljárásaink. Azért mindenesetre alaposan megkönnyítjük vele a saját dolgunkat, ha ezt sem végezzük teljesen logikátlanul. Tapasztalataim alapján elmondhatom, hogy úgy a legcélravezetőbb a sorszámokat megválasztani, hogy legelsőként vesszük a legmagasabb rendű, legtöbb szereppel bíró „tárgyakat”: az élőlényeket. Amiképpen az ajtók sorolásakor tettük, 1-től valameddig legyenek sorszámozva, és úgy, hogy mi magunk (azaz a főhős, a játékos!) is csupán egyek legyünk közülük, semmilyen kitüntetett szereppel ellátva – ez majd később fog jól jönni, ha azt akarjuk megvalósítani, hogy egyszerre több szereplő bőrébe is bele tudjunk bújni. Közvetlen utánuk következzenek az olyan, mozdítható tárgyak, eszközök, amelyeket föl tudunk venni és magunkkal tudunk vinni útjaink során – alighanem ezekből lesz a legtöbb. Folytassuk a leltárt a mozdíthatatlan, de azért még jelentős szerepet játszó díszletekkel (pl. egy asztal, amelyet megvizsgálva találhatunk rajta valamit, vagy szekrény, amit ki lehet nyitni stb.), utánuk azokkal, melyeknek már nincsen központi szerepük, ténylegesen csupán díszítő és hangulatfestő háttérelemek gyanánt vannak jelen a játékban (pókhálók a sarokban, fal, föld, erdő, víz…) – s a legvégére kerüljenek a rendkívül nagy példányszámban tenyésző, teljesen különleges nyilvántartású elemek (pl. ajtók). Az egyes csoportok között hagyhatunk kisebb hézagokat, kihasználatlan területeket, mert ha később netán kiderülne, hogy kifelejtettünk egy tárgyat, vagy még okvetlenül be szeretnénk szúrni valamit, akkor ne kelljen utólag átsorszámozni az egészet… Ügyeljünk ezenkívül arra is, hogy a gyűjtőnévként összevonható, hasonló funkciójú elemek (pl. a különféle kulcsok) lehetőleg egyetlen, nagy tömbben hézagok nélkül legyenek – nem mindegy ugyanis, ha a játékos egy határozatlan értékű KULCS szót használ valahol, akkor összevissza kell-e szaladgálni a különféle főnevek között, vagy csupán az x-diktől az y-dikig terjedő területet egy rendezett ciklussal végigfutni, közelebbi meghatározás végett.

Az egyes tárgyak kezelése egy gondosan és körültekintően kialakított sorszámozással rendkívüli módon leegyszerűsödik. Pl. ha éppen egy NYISD AJTÓT utasítás végrehajtása közepette vagyunk, s már eldöntöttük, melyik ajtóról van szó, de az ajtó feltárulkoztatásához történetesen a 28-as számú bronzkulcsra van szükség, akkor annak ellenőrzése, hogy a kalandozó ezt próbálja-e használni e nemes célra, mindössze annyiból fog állni, hogy megvizsgáljuk a jelenlegi parancs eszközhatározó rekeszét: ha ez nulla, akkor kiírjuk, hogy „Az ajtó zárva van.”; ha egyenlő 28-cal (ami éppen akkor fordulhat csak elő, ha a parancshoz hozzáfűzte a BRONZKULCCSAL szót is!), akkor sikeres lesz a művelet, kinyílik az ajtó; míg ha valami más érték van ott, úgy egyszerűen annyit üzenünk neki, hogy ezt az ajtót nem nyitja az a valami, amelyet próbál. (Természetesen előtte azért meg kell bizonyosodni arról, hogy tényleg *zárva* van-e az az ajtó, nem pedig csak – mondjuk – csukva.) Amennyiben nagyon sok ajtónk és hozzájuk való kulcsunk van, és mindegyikük hasonló módon működik, akár ezeket is lehet egy újabb táblázatba rendezni, hogy ne kelljen külön-külön foglalkozni valamennyivel…

És pontosan erről van itt szó! A tárgysorszámot indexként (vagy keresendő adatként) alkalmazva listák és táblázatok valóságos dáridóját hozhatjuk létre, melyek a tárgyak különböző tulajdonságainak, előre megadott helyzetekben megfelelően történő viselkedésének leírását szolgálják. Jelen esetben ezek közül csak egyet említünk: a legfontosabbat, mely az objektumok térképen való elhelyezkedésének záloga. Általában ez egy olyan táblázat, amelyben a tárgysorszámot közvetlenül indexként használjuk (akárcsak a bejárási táblázatnál a helysorszámot), és valamennyi tárgyhoz tartozik

benne pontosan egy-egy adat – annak a helyiségnek a sorszáma, ahol a tárgy éppen megtalálható. Egy-egy tetszőleges tárgyunk elhelyezkedése ebből szempillantás alatt kiolvasható. Ha a korábbi példáinkban felbukkant bronzkulcs teszemazt a hálószoba padlóján hever, akkor ez azt jelenti, hogy eme táblázat 28. eleme a 2-es értékre van beállítva. Ha valahol a 0-ás értékkel találkozunk, az szokás szerint azt jelenti: a hozzá tartozó tárgy „eltűnt" – semerre sem látható. Egyes díszítőelemek (pl. fal vagy padló) olyan rengeteg helyen előfordulnak egyszerre, hogy nem érdemes ezeket annyiszor külön fölvenni tárgyként, hanem csupáncsak egyszer, s elérésük speciális módon történik: mindegyiknek saját listája van a hozzá tartozó helyekről, vagy éppen a fal esetében erre egyáltalán nincs is szükség; elvégre a bejárási táblázatban előzetesen már bevezettünk egy sajátos ál-sorszámot arra az esetre, ha valamelyik irányban fal zárná le az utunkat, így most is csak elegendő megvizsgálni, előfordul-e vajon ez az érték a pillanatnyi helyiségben valahol (s ha igen, akkor létezik itt a FAL nevű tárgy). Szintén sajátos kisebbséget alkotnak az ajtók, kapuk és más hasonló akadályok: ezek mindegyike két helyiségben található egyidejűleg, s ezekért a pozíciókért mindössze az akadálylistába kell visszanyúlni egyszer (ha ügyesek voltunk, itt is a tárgysorszámmal indexelhetünk). Nem tartozik érték ebben a táblában a gyűjtőnevekhez – azokat úgyis egy másik, konkrét tárgyra fogjuk beváltani, ha pedig nem sikerülne ez, akkor maga a gyűjtőnév-tárgy sem elérhető. Az élőlényekhez ellenben tartozik még egy másik, szintúgy roppantmód fontos táblázat, amelyben ugyancsak a tárgysorszámmal indexelhetünk (többek között ezért is volt érdemes a lista elejére tenni őket): az ajtókhoz hasonlóan önekik is állapotuk van! Célzatosan a nulla érték mutassa azt, hogy valaki él, virul, egészséges és éber – az ennél magasabb számok pedig különféle rendkívüli állapotokat jelölnek (pl. alszik, elájult, részeg, megőrült vagy meghalt).

Mindebből már megválaszolásra került az az, előző fejezetben nyitva hagyott kérdés: miként mozdulunk el egyik helyiségből a másikba. Értelemszerűen a kalandor játszotta szereplőnkhöz is tartozik a táblázatnak egy olyan eleme, amelyik megmondja, hogy hol van – pusztán csak ezt kell megváltoztatni a megfelelő új értékre (amelyet a bejárási táblázatból olvastunk ki, az iránynak megfelelő oszlopban).

4.1.8.) Hogyan találjunk meg valamit?

De mielőtt még hozzászoknánk a gondolathoz, hogy milyen egyszerű az életünk, szembesítsük magunkat egy megdöbbentő kérdéssel, mely az egész rendszernek hirtelen egy nagyságrenddel nagyobb mélységet és bonyolultságot ad! Mi történjék akkor, ha bármelyik tárgyunk egyben helyiség is lehet? Talán első hallásra nem egészen világos a dolog, ezért megmagyarázom: egy reális világban gyakorlatilag tetszőleges tárgy hozzákapcsolódhat egy másikhoz, különféle módokon – például rajta áll egy asztalon, bent csücsül egy szekrényben, vagy éppenséggel a mi kezünkben van. Ilyenkor annak a tárgynak a gazdája tulajdonképpen egy helyiség, ahol a tárgy tartózkodik. De ugyanakkor a gazda-tárgy is egy másik, tágabb helyiségben létezik, ahonnan – mintegy kívülről – nézve mind a két tárgyunk látható. Ha a hordozót, mint helyiséget tekintjük, onnan nézve kifelé nemigen látunk sokmindent, viszont befelé, „mélyebbre" korlátlan bepillantást nyerhetünk. Eszerint föl kell készítenünk a programot rá, hogy mindegyik objektum hordozhassa a másikat – enélkül még csak azt sem tudnánk megoldani, hogy

egyáltalán kézbe vegyünk egy kavicsot. Ezeket a tárgy-helyiségeket természetesen meg kell különböztetnünk a valódiaktól.

Ha alaposabban megvizsgáljuk a kérdést, rájöhetünk, hogy bár ennek a kapcsolódásnak a valóságban töméntelen módozata elképzelhető, mégis három olyan alapvető kategóriára tudjuk leszűkíteni őket, amelyek egy kalandjáték szempontjából is nélkülözhetetlennek látszanak: vagy *nála* van valakinél, vagy *rajta* van valamin, vagy pedig *benne* van valamiben az illető holmi. Ha ezeket 1-től 3-ig sorszámmal látjuk el, azután 0-ás sorszámmal hozzávesszük az alapesetet, amikor közvetlenül egy helyiségben leledzik, az pontosan négyféle lehetőség. A helyiségek sorszámait leggazdaságosabb 2-byte-os egész számként nyilvántartani, ami így legfeljebb 65536-féle értéket vehet fel; minthogy sem tárgyból, sem helyiségből soha nincsen 16384 darab (rendszerint párszáz szokott lenni…), a 16 bit legfelső 2 bitjét nyugodtan fenntarthatjuk ennek az információnak a tárolására. A dolog ekkor úgy áll, hogy 0-tól 16383-ig számozódnak a valóságos helyiségek, efölött pedig a fiktív, tárgy-jellegű pozíciók helyezkednek el. Ha egy tárgysorszámhoz 16384-et hozzáadunk, megkapjuk azt a helyet, ahol a nála lévő valamiket keresgélni lehet (ennek csak lényeknél van értelme); 32768-at adva hozzá, a külső felülete bukkan elő; 49152-vel növelve meg, pedig a belsejébe mutatunk. Pl. ha egy zseblámpa sorszáma a 45-ös, a beléje csavart izzóé pedig 73, akkor az azt jelenti, hogy a 73. tárgy a 45 + 49152 = 49197. helyiségben tartózkodik. Ezek az elvont helyiségek így szervesen és zavartalanul simulhatnak bele a korábban kialakított rendszerbe, egyetlen dolgot kivéve: ha valahol egy ilyen értékre bukkanunk, akkor nem elégedhetünk meg vele, mint normális helyszínnel, hanem tovább kell kutatnunk azt is, hogy az a tárgy, amihez kötődik, merrefelé található. Ennek a szervezésnek a révén számos tárgyat eldughatunk a kíváncsi játékosok orra elől, akik – mint holmi húsvéti tojásra – így csak hosszas vizsgálódás után lelnek végre rá. (Amikor megvizsgálunk valamit, a program kiírja vele együtt a hozzá kapcsolódó tárgyakat.)

Csakhogy ez így még mindig túlságosan egyszerű volna, úgyhogy azért még bonyolítunk rajta egy keveset… Utóvégre is, egyetlen ember sem egy Sámson vagy egy Herkules, és mindennek van valahol határa! Az olyannyira áhított valósághűség érdekében a tárgyak befogadóképességét korlátozni kell – egy szatyorba talán mégse lehessen belezsúfolni többet, mint amennyi belefér vagy amit a fülei elbírnak. Ha valamit oda szeretnénk tenni, ahol már nincs számára hely, vagy többet próbálunk fölemelni a karjaink erejénél, figyelmeztetést kell kapnunk a játéktól, hogy nem tehetjük meg (pl. „Nem bírsz már el annyi súlyt.”). De ennek a luxusnak komoly ára van: minden egyes tárgyunkhoz további három tulajdonságértéket kell bevezetnünk hozzá; ezek: a saját súlya, a teherbírása és a pillanatnyi terheltsége. Az első kettő ugyebár egy állandó érték (nem muszáj, hogy valóságos mértékegység legyen, tetszőleges viszonyszámként alkalmazható); a harmadik meg folyton változik, de nem nőhet magasabbra a másodikénál. Egy tárgy terheltségét úgy kaphatjuk meg, hogy saját súlyához az összes hozzá csatlakozó tárgy terheltségét hozzáadjuk – azért a terheltségét, és nem pedig a súlyát, mert így az az érték már összegezve tartalmazza a kapcsolódó tárgy további kapcsolódásait. Következik ebből, hogy mikor egy tárgy „üres”, azaz terheletlen, akkor a terheltsége nem nulla, hanem pontosan a tulajdon súlyával egyezik meg (a teherbírást is ennek figyelembevételével szükséges megállapítani, és lényegében csak emiatt van szükség magára a súlyra).

Többnyire egy kalandjáték leggyakoribb eljárásainak egyike egy bizonyos tárgy elérhetőségének a meghatározása: mi megadjuk neki a helyiséget, amelyen belül vizsgálni óhajtjuk, és a keresett tárgy sorszámát – ő visszaadja azt, hogy a helyszín „felségterületén" hány darabot talált belőle. (Egynél több is lehet, hogyha gyűjtőnevet adtunk meg, vagy több ugyanolyan tárgy is szerepel a játékban.) Ha létezik ilyen tárgy, akkor azt mondjuk, hogy azon a helyszínen jelen van, vagy arról a helyről elérhető stb. Rendesen egy jelenlévő tárgyhoz kapcsolódó további tárgyak is elérhetőek szoktak lenni. Természetesen egy szereplő kizárólag jelenlévő holmikkal cselekedhet bármit is (néhány kivételtől eltekintve, pl. amikor kérdezünk valakit egy témáról), ezért mielőtt még bármilyen parancsot is kiadni merészelne, legelső teendőnk az lesz, hogy a benne lévő *összes* főnév elérhetőségét ellenőrizzük. Ha pl. azt mondja nekünk, hogy HAJTSD FEL A SZŐNYEGET egy szőnyeg nélküli szobában, akkor mi keményen visszavágunk neki azzal, hogy „Nincs itt semmiféle szőnyeg." – az utasítás végrehajtásáig (vagy egyáltalán: az ige elővételéig – lehet, hogy nincs is benne ige!) még csak el sem jut a program, hanem mindenfajta cselekvés nélkül, hatástalanul továbbugrik a következő parancsra. (Következésképpen nyugodt szívvel írhatja akár azt is, hogy EDD MEG A SZŐNYEGET, vagy csak egyszerűen SZŐNYEG – ugyanazt a feleletet kapja rá úgyis.) Ezzel csírájában elfojtunk mindenféle illegális tárgyra való hivatkozást, s mire a végrehajtáshoz érünk, addigra garantáltan megvan az összes főnév – tehát ezzel később már nem kell törődnünk a programban. Fontos még, hogy az is mindig jelen van, ami az általunk irányított hőshöz, mint „tárgyhoz" kapcsolódik – vagyis a nálunk lévő cuccok, rajtunk viselt ruhadarabok. Ugyanide tartozik az is, hogy a többi szereplők mind-mind féltékenyen őrzik előlünk a náluk lévő dolgokat, ergo ha egy ördögnél lévő bármilyen tárggyal próbálnánk meg babrálni valamit – kivéve, ha csupáncsak megszemléljük azt –, akkor: „Az ördög nem engedi." (De ha az ördög elaludt vagy meghalt, akkor zavartalanul elvehetünk tőle bármit!)

Látjuk viszont, hogy a szőnyeg most már nemcsak a padlón, hanem akár egy asztal tetején is heverhet – mi több, még annak a tetején is állhat egy láda, s a ládában egy törpe, a kezében egy táskával, ami egy erszényt rejteget, és elképzelhető, hogy mi éppen az erszényben csöndben meglapuló aranypénzt szeretnénk kiemelni onnan… Hogy a helyzet még bonyolultabb legyen, a törpe dühösen becsaphatja az orrunk előtt a táskát, és akkor a benne lévő erszény, a pénzdarabbal együtt, eltűnik a szemünk elől; ha erre mi bosszúból rácsukjuk a törpére a láda fedelét, akkor már az egészből semmit sem látunk, csupán egy asztali szőnyeg tetején fekvő, lecsukott ládát… (De ha a ládában benne lennénk mi is, akkor megintcsak látnánk a törpét, eltekintve attól, hogy odabent sötét van.) Magyarul, a tárgyak tetszőleges mélységig egymásba ágyazódhatnak, s hogyha *benne* van egy tárgyban valami, akkor még azt is figyelnünk kell közben, hogy nyitva van-e a hordozó vagy csukva. Ahhoz, hogy a pénzérme elérhetőségét megtudjuk, először is az erszény elérhetőségét kellene ismernünk, ami viszont a táskáétól függ, és így tovább – ezt a keresést tehát egyedül egy *rekurzív eljárással* tudjuk megoldani, olyannal, mely minden egymásba ágyazódás esetén újra és újra önmagát hívja meg. Ha eközben bármelyik szinten negatív válasszal tér vissza, akkor az végigvonul a teljes láncon, és az eredményünk elutasító lesz; ha nagynehezen mégiscsak elvergődünk a gyökérelemig, azaz egy valóságos helyiségig (és az megegyezik a keresett helyiséggel!), akkor megvan a tárgy, és elérhető.

Az igazsághoz hozzátartozik, hogy ennek az egész tárgy-kezelésnek van egy elég komoly korlátja. (Nem a rekurzív keresésre gondolok, hanem az előző fejezetben

kifejtett táblázatos nyilvántartásra!) Csak akkor alkalmazható hatékonyan, ha mindegyik tárgyból egy vagy több – de konkrétan előre meghatározott számú darab létezik. Pl. egy fáról leszedünk egy almát – az egy darab. Leszedünk egy másikat – most már kettő van a kezünkben. De ha leszedünk százat, akkor száznak kéne lennie – ami viszont ebben a rendszerben kizárólag úgy oldható meg, ha már eleve, a játék indításakor is létezik valahol mind a száz alma, pontos, külön sorszámokkal ellátva. Ha azt szeretnénk, hogy bármelyik tárgyból lehessen hasonlóan száz darab, akkor már eleve mindegyikből legalább ennyit kellene a játékba beleterveznünk. Párszáz tárgy helyett így lenne több tízezer – iszonyatos pazarlás a tárral és az idővel! Egyszóval, ez a kezelési rendszer, bár egyszerű, de túlságosan merev. Az esetek túlnyomó többségében ugyan tökéletesen beválik, de ha valamivel igényesebbek vagyunk, akkor meg kell próbálnunk ugyanezt valahogy dinamikusabban megvalósítani. Erre megoldás lehet, hogyha egyetlen, statikus táblázat helyett egy amolyan változó hosszúságú puffer-táblázatot készítünk. Ennek minden eleméhez jóval több adat tartozna, mint eddig: lenne egy, mely megadja, hogy milyen tárgyról van szó (tehát az eddig használt tárgysorszám), s egy másik, mely azt jelölné, hány darab van belőle. A harmadik volna a helyszín, amelyen megtalálható – ez annyiban módosulna, hogy a fiktív tárgy-helyiségek alapját most már nem a tárgysorszám jelentené, hanem egy, a pufferon belüli másik elemre mutatna tovább. (Negyedikként pedig hozzájönne még az aktuális terheltsége – a súly és a teherbírás elég, ha továbbra is a tárgysorszámhoz kapcsolódóan van meg, ugyanis azok állandó értékek. Ellenben ne felejtsük, hogy az összeadandókat itt még a darabszámmal is szorozni kell!) Ha új tárgy bukkanna elő valahonnét, először mindig helyet kéne neki foglalni a pufferban, s hogyha betelne az egész terület, akkor lehetne sorjában eldobálni a régebbieket. A száz alma így mindössze egyetlen bejegyzést foglalna el, amiben 100 volna a darabszám (amikor további almákat gyűjtünk a már meglévők mellé, azokat egyszerűen hozzá kellene adni ehhez). Ha kettesével szétpakolnánk őket ötven különböző helyiségbe, akkor már ötven bejegyzésünk lenne, 2-es darabszámokkal. Amikor keresünk egy tárgyat, a közvetlen indexelés helyett végig kellene olvasnunk a puffer teljes méretét, vizsgálva, hogy melyik elemben egyezik a tárgysorszám a kívánttal; amikor egy tárgyat törlünk, az az összes őhozzá kapcsolódót is magával rántaná. Vigyázni kéne, hogy a fontosabb tárgyak ne tűnjenek el.

4.1.9.) „Sokasodjatok és növekedjetek!”

Az eddigiek folyamán egy meglehetősen kihalt és sivár valóságot építettünk föl – egyelőre úgy fest az egész, mint az őstengerek, a bennük úszkáló egysejtűek nélkül. Hiányzik belőle valami, ami elevenné teszi azt: a játék szereplői még nincsenek bekapcsolva az események vérkeringésébe. Az élőlényeket nemcsak az különbözteti meg a többi tárgytól, hogy a sorszámozás legelején foglalnak helyet, hanem hogy sokkal változatosabb módokon lehetséges érintkezésbe kerülni velük, illetve a mi közreműködésünk nélkül is mindenféle „akciókra” ragadtatják magukat. Szélsőséges esetben ezek akár valóságos személyek is lehetnek – ld. a második fejezetben megemlített hálózatos kalandjátékokat –, máskülönben nekünk kell a program segítségével emulálnunk őket. Az egyes élőlényeket leginkább az különbözteti meg egymástól, hogy különféle közeledési kísérleteinkre eltérő módon reagálnak. Vannak

bizonyos alapvető magatartás-formák, amelyeket egy jó kalandjátéknak föl szükséges ismernie, és általános jelleggel lekezelnie őket: agresszív cselekedetek (megütni vagy megtámadni valakit), a beszélgetés különböző formái (köszönni, beszédbe vegyülni általában, vagy pontosan meghatározott dolgokat mondani, esetleg faggatózni jól körülírt témákról) és a tárgy-csere jellegű dolgok (megmutatni, odaadni vagy elkérni egy tárgyat, netalántán pénzt). Ezeket valamennyi szereplőre nézve külön-külön ki kell dolgozni, úgy, hogy látszólag egyéni módon reagáljanak.

Sajnos, a legtöbb kalandjáték úgy készül, hogy már eleve egy szilárdan lebetonozott nézőpontból mutatja a benne zajló eseményeket. Ez azt jelenti, hogy létezik egy e célra kiválasztott főhős, és mi mindent egyedül az ő szemein keresztül látunk. Kétségtelenül ez a legkényelmesebb megoldás, ám sokkal izgalmasabb a játék, hogyha kívülről is megfigyeljük benne önmagunkat közben. Programozástechnikailag legjobb, ha – a manapság divatos objektumorientált programozáshoz hasonlóan – a játék szereplőinek mindegyike egy önálló, zárt egységet alkot, s valamennyiükhöz tartozik egy-egy, a magatartásukat generáló program vagy programrész, amiről a vezérlő főprogram gyakorlatilag semmit nem tud, csupáncsak az általuk jelzett cselekvési szándékaikat látja. Ebben a felállásban – a főprogram szempontjából legalábbis – maga a játékos, „a” főhős is csak „egy a sok közül”, a többiekkel egyenrangú lény, aki hasonló korlátokkal és lehetőségekkel rendelkezik, akárcsak balsorsában osztozó társai; más kérdés, hogy – valódi léténél fogva – ezekkel a lehetőségekkel hasonlíthatatlanul sokoldalúbban képes élni és cselekedni, mint ők. Ez egyúttal azt is jelenti, hogy bármilyen lépésre képesek vagyunk, azt akárki más is megtehetné helyettünk ugyanebben a helyzetben – elméletileg –, más szóval a többieket is aktívan bevonhatjuk a játék feladványainak teljesítésébe; ahelyett, hogy magunk tennénk meg valamit, megkérünk rá valaki mást. (Pl. NYISD KI AZ AJTÓT helyett így: KÉRD MEG AZ ŐRT, HOGY NYISSA KI AZ AJTÓT.) Ettől egyrészt mindjárt változatosabb lehetőségeink nyílnak, hiszen ha többször próbálkozunk valamivel, nem kell mindig ugyanazon a módon megfogalmaznunk, amit szeretnénk; másrészt pedig meglehet, vagy a program netán *el is várja* tőlünk, hogy másokat is megmozgassunk a végső siker érdekében: lehetnek benne olyan epizódok, amelyek feloldása kizárólag csapatmunka árán lehetséges. Például áll valahol egy őrszem, aki egy bizonyos ponton nem enged átlépni senkit – de ha egy szereplőnkkel ügyesen elterelük a figyelmét a feladatáról, akkor másvalaki észrevétlenül elsurranhat a háta mögött közben… Egycsapásra érdekesebbé válnak ily módon a játékunk feladványai! Arra is lehetőséget nyújt ez a rendszer, hogy akár egyszerre több szereplő bőrébe is belebújjunk, állandóan váltogatva azt, hogy melyiküket alakítjuk éppen. Egy játék résztvevőinek viselkedése az alábbi forrásokból származhat:

1.) Közvetlenül a billentyűzetről begépelt utasítások határozzák meg számára a következő lépést. Tipikusan a főszereplő magatartása ilyen. Egy főhős személyének kiválasztása nem is olyan bonyolult, mint hinnénk: adott a szereplők listája, amelyek közül választani lehet, s mindössze egy változóban folyton nyilván kell tartani annak sorszámát, akik jelen pillanatban éppen vagyunk. Amikor a játékos azt mondja, hogy ÉN, MAGAMAT, ENGEM… stb., ezeket a személyes névmásokat kicseréljük az itt tárolt sorszámra, üzenetek kiíratása közben pedig annak neve helyett a „te” (Ön), „magad” (maga)… stb. alakokat kell – persze helyesen felragozott formában – használni. Na ez utóbbival szokott inkább gond lenni – szeretjük az üzeneteket kész szövegként, rögzítetten tárolni, és nemigen fűlik a fogunk hozzá, hogy mindenféle

képlékeny kifejezésekkel bonyolítsuk őket. Elvégre ez nemcsak egyetlen szónak a cseréjét jelenti, de a teljes szövegkörnyezetet is hozzá kell igazítani ehhez. (Pl. „Valaki kinyitja az ajtót.” helyett „Kinyitod az ajtót.” – az alany mellett az állítmány is megváltozik stb.) Ha azt akarjuk elérni, hogy egyes tevékenységekre csak bizonyos „kiválasztottak” legyenek képesek, nem kell mást tenni, mint előtte ellenőrizni, hogy éppen ők vagyunk-e mi.

2.) Ösztönös sugallatait egy külön e célra készített, irányító programrésztől kapja. Ez kétféle módon történhet: vagy adott helyzetben egy előre lerögzített cselekménysort hajt végre (pl. van valahol egy éjjeliőr, aki bizonyos időközönként egy határozott útvonalat végigjár, és ellenőrzi, minden rendben van-e arra – napközben meg lefekszik aludni, és ki lehet figyelni az őrségváltás idejét stb.), avagy pedig részben véletlenszerűen állítja elő ezeket a program (egyes alakok a játékban szeszélyesen kóborolnak ide-oda, miközben időnként az „eszükbe jut” valami – mondjuk becsukni egy ajtót vagy eldobni egy tárgyat stb.). Esetenként ez az előbbi ponttal kombinálódhat, olyképpen, hogy a saját figuránkon keresztül megkérünk rá valaki mást, hogy tegyen meg számunkra valamit (az említett KÉRD MEG AZ ŐRT… kezdetű példa!), s ha jó kedvében találjuk a fickót, akkor lehet, hogy még engedelmeskedik is… Nem könnyű – de igen hálás! – feladat minden egyes szereplőnkhöz különböző programokat írni, és bizony komoly előzetes tervezést igényel összhangba hozni számos olyan eseményt, amelyek egyidőben, de a játéktér különböző pontjain mennek majd végbe…

3.) Valahonnan „kívülről”, a számítógép számára ismeretlen külvilágból érkeznek a parancsai – rendszerint már előértelmezett, azaz a legrövidebb, legtömörebb alakra faragott formában. Ez az eset akkor állhat elő, amikor egy olyan programot írunk, amivel egyszerre több felhasználó is üzekedhet – az egymással összeköttetésben álló felek a hálózaton keresztül kommunikálnak egymással. Vegyük észre, hogy ez alig különbözik az első ponttól, szinte csak annyiban, hogy itt egyszerre több kalandorhoz kell hozzárendelt sorszámokat nyilvántartani.

Ezenkívül gondoskodni kell még arról, hogyha egy szereplő valamilyen módon „önkívületi” állapotba került (meghalt vagy elájult), akkor az eredeti irányítás ne érvényesüljön nála, és ne is kommunikálhasson a környezetével többet (de bizonyos idő elteltével azért újra magához térjen – kivéve, ha meghalt). Az eddigiek konkrét megvalósítása nehéz, de nem túlságosan bonyolult programozói feladatot jelent (meg lehet írni egy általános szervező programot, ami minden lehetséges bemenetet lekezel), sokkal keményebb mogyoró azonban ennek a fordítottja, vagyis az egyes résztvevők tájékoztatása a különféle történések hatásairól. A gyakorlatban ez annyit tesz, hogy a központi résznek a játék világában végbemenő *minden egyes* eseményt figyelnie kell, és erről az összes szereplő felé valamilyen üzenetet továbbítania – mégpedig *minden egyes* szereplő felé más-más tartalmú üzenetet! Ráadásul az egyes események is – mintegy következmény gyanánt – további eseményeket vonhatnak maguk után. Egy nagyon egyszerű példával illusztrálva a dolgot: ha valahol egy Tompika nevű szereplő pl. kinyit egy ajtót (hogy továbbra is ennél az oly sűrűn fölbukkanó motívumnál maradjunk), akkor nem elég az ajtó állapotát nyitottra változtatni, hanem az illetőt értesíteni is kell róla: „Kinyitod az ajtót.” (Az egyszereplős kalandjátékok meg is elégednek ennyivel!) A vele azonos helyszínen tartózkodókat ugyanakkor már arról kell tájékoztatni, hogy „Tompika kinyitja az ajtót.” – míg végül a távol maradóknak abszolúte semmit nem szabad megszimatolniuk erről az egész cselekményről. Sőt, még egy negyedik csoport is van: az ajtó *túloldalán* állók nem tudhatják, ki cselekszik, így mindössze annyit

észlelnek belőle, hogy „Valaki kinyitja az ajtót.” Ezenkívül a helyzetet még tovább bonyolíthatja, ha teszemazt a kétféle helyiség közül valamelyikben *sötét* van: ekkor tudniillik az ott-tartózkodók *nem láthatják*, amidőn Tompika kinyitja az ajtót, tehát akkor ezeket nem is szabad értesíteni róla… Viszont megeshet, hogy a másik szobából, az immáron nyitott ajtón keresztül beárad a fény, aminek következtében ezek a bácsik nemcsak hogy az ajtó kinyitásáról szereznek tudomást, de ráadásul még megpillantják azokat a dolgokat is, amiket eddig a teremben honoló sötétség eltakart a szemük elől. (Tehát mindjárt a helyszínleírást is ki kell nekik írni, és az egész szoba kezelése megváltozik.) És akkor akár ez még folytatódhat azzal, hogy mondjuk a hirtelen világosságban valaki fölismeri az ellenségét, és azonnal rátámad… Vagy nála van egy tekercs film, ami fényt kap és tönkremegy stb. Egy ilyen egyszerű döntés is, mint amilyen egy ajtónak a kinyitása, alkalmasint a történések egész láncolatával (vagy inkább: hálójával) járhat együtt – amiről pedig szintén mindig tudósítanunk kell az érintett felek mindegyikét. Egy dinamikusan szerveződő kalandjátékban lépten-nyomon előfordulhatnak hasonló jellegű problémák, ahol rengeteg, egymástól látszólag független külső körülmény együttes hatását kell figyelembe venni.

4.1.10.) Az idő kerekéhez kötve

Egyáltalában nem mindegy, hogy egy kalandjátékban miféle ütemhez igazítva telik-múlik az idő. A legegyszerűbb megoldás, különösen az egyszereplős kalandjátékokra leginkább jellemző, hogy mindig egy-egy sikeres parancs végrehajtása lépteti tovább az egész rendszert egyetlen időegységgel, mintha valamiféle, változékony hosszúságú órajelet adna neki ezzel. Ameddig gépelünk, a következő mondatunkat szerkesztjük, addig a játékban áll az idő – mihelyst azonban leütjük az ENTER-t (és be is írtunk neki valamit), egycsapásra mindenki öregebbé válik, mondjuk egy negyed órával. Tulajdonképpen nem is időhöz, hanem lépésszámhoz vannak szinkronizálva egy ilyen játék történései – pl. olyan időzítéseket alkalmaznak bennük, hogy – mondjuk egy kopogtatást követően – öt lépés múlva kinyitja nekünk az ajtót valaki. Tetszőleges ideig lehet töprengeni a pillanatnyi helyzet megoldása fölött, és hiába ugrik a nyakunkba egy vérszomjas démon, nyugodtan elmehetünk megvacsorázni, mielőtt végképp az arcunkra fagyna az az utolsó vigyor… A program belső számlálóit, amelyek az eljövendő események titkos előhírnökei, a végrehajtott utasítások csökkentik rendszeresen eggyel. És ez nagyon helyesen van így! Egy elsősorban képzeletre és gondolkodásra épülő játéknál ez egy ideális helyzet – tökéletesen megengedhető és elfogadható, hogy ki-ki a saját belső ritmusa szerint haladjon előre a feladatok megoldásában. (Hogy is nézne ki, ha egy gyakorlott gyors- és gépíró illetéktelen előnyre tenne szert a lassúkezűvel szemben…)

Csakhogy egy többszereplős kalandjátékban már egyszerűen tarthatatlanná válik ez az állapot. Ha a program mondjuk hat különböző játékos parancsait fogadja párhuzamosan, akik mind eltérő ritmus szerint gépelnek, akkor mégis melyikükhöz alkalmazkodjon a többi? Amikor az egyik semmit nem csinál, hanem csak karbatett kézzel és összeráncolt homlokkal bámulja a képernyőt, akkor is kötelessége a programnak, hogy pontosan a történések idejében haladéktalanul értesítse őt a másik lépéseiről, ha az éppen akkor halad el mellette, vagy ugyanazon a helyszínen tevékenykedik. Az efféle programokban eszerint nem tehetünk mást: valós idejű

időzítéseket vagyunk kénytelenek alkalmazni bennük. Ez viszont maga után vonja, hogy a bevitel és kiírás funkcióját az eddigieknél sokkal élesebben el kell határolnunk egymástól – mert mi történjék, hogyha véletlenül éppen akkor érkezik valakihez egy fontos kiírandó üzenet, amikor ő saját mondata szerkesztésének a kellős közepénél tart; esetleg föl sem pillantva a képernyőre, keresi a megfelelő billentyűt? Nem lehet csak úgy otrombán kettévágni a félig begépelt mondatot, és a közepébe belenyomtatni a szöveget! (De lehet: a TELNET-es játékok sajnos pontosan ezt teszik…) Okvetlenül kétfelé kell bontanunk a képernyőt, úgy, hogy külön legyen egy felület a kiírás és szintén külön a bevitel számára (utóbbinak két-három sor is elég lesz). Így azok már nem zavarják egymást – vagy mégis?! Mit tehetünk akkor, ha a megjelenítés mezejében egy különösen hosszú leírás kezd el szépen, megfontoltan kibontakozni az ismeretlenség homályából, mialatt mi szorgosan a parancsunkat gépeljük? Bevett gyakorlat, hogyha egy szöveg hosszabb annál, mint amennyi a képernyőre egy adagban kifér, akkor oldalanként meg-megszakítva, minden oldal végén egy billentyű-lenyomásra várakozva fokozatosan léptetjük azt tovább (<More> vagy magyarul <Tovább> funkció). De még ha ez az eset nem is forog fenn, akkor is az ablak görgetése több másodpercig is eltarthat, és rendkívül illúzióromboló lenne, ha erre az időre hirtelen megakadna az alsó sorokban a bevitel. Másik probléma: mi van, ha a kiírás-ablakban egy félig megjelenített szöveg éppen ENTER-rel való továbbléptetésre vár, de mi nem törődünk vele, és zavartalanul csak a mondatunkra figyelünk – hol várakozzon addig a szöveg hátralevő része, és ha kiírás közben újabb üzenetek érkeznek, azokat miképpen várakoztassuk? Meg kell oldanunk tehát azt is, hogy a kétféle funkció ne csak térben, de időben is egymástól teljesen független és párhuzamos legyen: mialatt gépelünk, *tényleg* aközben folyjon odaát a kiírás! A még kiíratlan, de a küldőtől már átvett „szűz” szöveget pedig addig is egy átmeneti pufferban kell tárolnunk, ahol egy bizonyos határig gyűlhetnek és halmozódhatnak a sorok és a mondatok, de ha a puffer betelt, akkor haladéktalanul ki kell görgetnünk őket a képernyőre – akár tetszik a felhasználónak, akár nem. Különösen mókás tud lenni, amikor a szereplőt a játékban egy súlyos baleset érte, de ő még valahol tíz oldallal följebb tart a szövegek olvasásában, miközben a többiek már réges-régen értesültek róla, hogy ájultan hever a földön, és apránként kipakolják a hátizsákjából az értékesebb cuccokat…

Egyetlen megoldás létezik a legsimább párhuzamosság elérésére, az, ha a program központi, vezérlő része sohasem „ragad le” valamilyen szubrutinnál, hanem egy örökös végtelen ciklusban megállás nélkül kering három alapvető tevékenység: a beviteli mező szerkesztése, az események végrehajtása (beleértve természetesen a saját és a többiek által kiadott utasítások végrehajtásait is) és a szövegkiírás alapvető fázisai közt. Ehhez az szükséges, hogy valamennyi funkciót apró, szétválasztható és önállóan végrehajtható kis lépésekre tagoltan valósítsuk meg. A szövegkiírás esetében ilyen építőkocka lehet pl. az átmeneti puffer egyetlen sorának kiléptetése a képernyőablakba, vagy a beviteli rutin esetében egyetlen lenyomott billentyű beolvasása a billentyűzet-pufferből (ha van olyan), és annak megfelelően a beviteli mező módosítása. A program tehát úgy fog működni, hogy folyton figyeli, történnie kell-e valamilyen eseménynek, s ha igen, akkor végrehajtja azt, és a neki megfelelő szöveget a puffer alsó végéhez hozzácsapja (ez egyetlen szemvillanás alatt megvan); majd rögtön továbbadja a vezérlést a kiíró rutinnak, amelyik a szövegpuffer legfelső sorát kigörgeti a képre (hacsak nem üres a puffer éppen akkor); ezt követően a beviteli szubrutin lép színre, mely pedig egyetlen karaktert beolvas, ha képes; végezetül pedig vissza az elejére, és ez így megy tovább

megállás nélkül… Nem kis feladat elérni, hogy ez az egész így egyben kellőképp összehangoltan fusson – különösen, hogyha grafikus képernyőt használunk a megjelenítéshez, ami már eleve alaposan lelassítja a kiírásokat. (Ajánlott az ASSEMBLY nyelv használata például.)

Többek között emiatt is számít, hogy mind az értelmező szubrutin, mind pedig a program más egyéb végrehajtó eljárásai igen-igen serényen végezzék a dolgukat – mert miért ne fordulhatna elő, hogy húsz különböző játékos egyszerre adja ki, egyenként hatszáz betűből és harminc parancsból álló mondatait? Amiből mindjárt egy másik szempont is következik, nevezetesen hogy az egyes játékosok teendőit is hasonló pufferokban kell előzetesen nyilvántartani – elvégre mindnyájan kiadhatnak több parancsból álló mondatokat is, amiből egyelőre még csak az első vagy a második hajtódik végre, amelyeket addig is tárolni kell valahol, de ő gonosz mosollyal az ajkán tüstént begépeli máris a következő adagot, és lehet, hogy ezt egyszerre akár többen is megteszik…

És ne gondoljuk azt sem, hogy mindezek a szörnyűségek kizárólag hálózatos kalandjátékokban fordulhatnak elő! Ha egy olyan kalandot készítünk, amiben ugyanazt az egyetlen számítógépet használó játékos több szereplőt is irányíthat egymással párhuzamosan, mondjuk ESC-pel vagy TAB-bal kapcsolgatva az egyik vagy a másik között (esetleg osztott képernyőn megjelenítve egymás mellett egyidejűleg akár többet is), miközben valós idejű megjelenítést alkalmaztunk benne, az pontosan ugyanakkora galibákat teremthet, mintha ezren küldözgetnék az utasításaikat a komputernek egyszerre! De megéri a belefektetett munkát a dolog, mert rendkívül látványos lesz a végeredmény.

Ezáltal tehát elértük, hogy – elméletileg – akárhány játékos tényked̄ését le tudjuk kezelni egyidejűleg, miközben sem a végrehajtás, sem a folyamatosan a háttérben zajló szövegkiírás nem zavarja egymást, és főképpen pedig a bevitel mindeközben zökkenőmentes marad. De nyitva maradt még egy probléma: a különböző játékosok különböző sebessége. Ha valaki gépír, mint a villám, annak a parancsai is fokozott ütemben hajtódnak végre, amire még ráfoghatjuk, hogy megérdemli ezt a kis előnyt, még ha a többiek nemhogy reagálni rá, de még csak figyelemmel követni se nagyon bírják közben; de ha egy több parancsból álló mondatot írt be valaki – erre ugyebár egy lassúbb illető is képes –, akkor már igazán tisztességtelenül manőverező vágtázásba kezdenek az utasításai – ennek egymást követő lépéseit a program mindenfajta késleltetés nélkül, teljesen egybefolyva, mondhatni szinte egyszerre vágja a többiek képébe. Fokozott mértékben áll ez a számítógép által irányított szereplőkre (ún. NPC-k: ez a „Non Player Character” – „nem játékos szereplő” – angol nyelvű kifejezés rövidítése): őnáluk tudniillik abszolúte semmiféle gépírásról nincsen szó – az illető fickó cselekvési szándékai a másodperc törtrésze alatt alakulnak ki egy külön e célra berendezett „műhelyben”. Ennek eredményeképpen aztán ezek olyan követhetetlen ámokfutásba kezdenek, hogy szinte látni se nagyon fogják őket a többiek, amint nagyritkán föltünedeznek egy-két tizedmásodpercre itt-ott… A megoldás kézenfekvő: minden egyes élőlénynek – függetlenül attól, valódi-e vagy NPC – osszunk ki valamilyen maximális sebességértéket, amivel haladhat. Ez pl. úgy néz ki, hogy mindenkihez tartozik egy-egy számláló, melyeket bizonyos időközönként csökkentünk, s ha nullára csökkent, akkor következik lépésre az az illető. Ezeket egytől néhány másodpercig terjedő tartamokra célszerű beállítani, miáltal mindenkinek személyre szabott sebessége lehet – bizonyos játékosokat vagy NPC-ket tetszőlegesen

fölgyorsíthatunk vagy lelassíthatunk, amivel a helyzetek nehézségét is lehet némiképpen állítani. Az említett értékeket akár dinamikusan is változtathatjuk: pl. minél jobban meg van pakolva súlyos tárgyakkal valaki vagy minél fáradtabb és kimerültebb, annál jobban lelassul a mozgása stb.

4.1.11.) Néhány jótanács és további lehetőségek

Az eddigiek során eléggé nagyvonalúan és magas szinten kezeltük a témát – ha ugyanezt részletekbe menően kéne kifejteni, egy egész könyvet sem volna nehéz megtölteni a kalandjáték készítéséről. (F. DaCosta megtette már ezt, de ő csak az egésznek az alapjaival foglalkozott.) Most befejezésül mindössze néhány hasznos jótanácsot szeretnék felsorolni azok számára, akik komolyabban szeretnének foglalkozni ilyesmivel, illetve egy kicsit eltöprengeni rajta, hogy mire is lehetne még fölhasználni a leszűrt tapasztalatokat.

Először is, a kalandjátékok világában kezdettől fogva léteznek bizonyos kialakult szokások és elvárások, amiknek nem árt, ha az újdonsült jövevények is megpróbálnak eleget tenni. Nem elég egy helyiségekből álló térképrendszert fölépíteni, és megtölteni tárgyakkal és élőlényekkel – ahhoz, hogy ez igazán kézzelfoghatóan megjelenjen a játékosok számára, az is szükséges, hogy az információknak lehetőleg minél bőségesebb tömegével elhalmozzuk őket azok milyenségét illetően. Mint ahogy a való világban, itt is a szemét használja leggyakrabban, azaz mindent, ami csak elérhető, vizsgálni és tanulmányozni igyekszik: kétségtelenül a mozgási parancsok mellett a leggyakrabban alkalmazott funkciójuk a VIZSGÁLD ige. Úgy kell megoldanunk, hogy ezzel minél többmindent elérjen, és lehetőleg minden tárgyról, amelyet megvizsgál, hosszú és részletes leírásokat bocsássunk a rendelkezésére. Ha mondjuk egy helyszínnek a leírásában hangsúlyozottan szerepel, hogy „az erdőben, egy hatalmas fenyőfa tövénél állsz”, akkor nem túlzottan szerencsés, ha egy NÉZD MEG A FENYŐFÁT próbálkozásra valami ilyesmi lesz a válasz: „Nem értem azt, hogy fenyőfát.” – vagy: „Nincs itt semmiféle fenyőfa.” Ilyesmi a legjobb körökben is előfordul néha, elvégre *mindent* belezsúfolni még a legbővebb memóriába sem lehetséges, de törekednünk kell rá, hogy minél ritkábban forduljon elő. Ha pedig már fölvettük a szótárba, és vizsgálni is engedi a program, akkor ne csak valami olyasmi kétszavas közhelyet vessünk oda neki, hogy „Szép nagy fa.” – hanem aprólékosan írjuk le, ahogy a kérge pikkelyei repedésekkel tarkítva borítják a törzsét, amelyet egyébként tömény, bódító illat vesz körül, földből kiálló gyökerei körül arasznyi vastagon fedik a talajt a többéves, bomló fenyőtüskék, itt-ott kibuggyant belőle a ragacsos gyanta, és egészen elszédülünk, amikor a magasban eredő legalsó ágaira nézünk… Ennek nemcsak öncélú szórakoztató szerepe van a játékban, hanem így burkoltan utalunk esetleges további tárgyakra, melyeket a játékos észrevesz, ha egy kicsit is kísérletező hajlamú. Pl. az említett avart fölpiszkálva egy odahullott fenyőtobozt találhat, amire majd a játék valamelyik másik pontján lesz szüksége, vagy a törzsről lekapart gyantát ragasztóként alkalmazhatja máshol stb. A lényeg az, hogy nem szabad mindent direktben az orra elé tárni, hanem el kell rejteni az értékesebb dolgokat, és a játékosra bízni, hogy apránként elmélyedve fölfedezze őket. Láttuk, miképpen lehet a tárgyakat egymáshoz kapcsolni és egymásba ágyazni, így most már nem a szoba közepére helyezünk majd mindent egyetlen halomban, hanem szétosztva mindenféle asztalokkal, polcokkal és

szekrényekkel töltjük meg a falakat, melyek mindegyike akár egy egész külön kis világot is tartalmazhat elrejtve magában. Ez a helyiségek elrendezésére is vonatkozik, azaz ne egyetlen kijelölt útvonalon lehessen végighaladni rajtuk, sőt, még csak egyáltalán ilyen „útvonalak" se létezzenek a játékban, hanem olyan legyen az egész, mint egy nagy, kerek, zegzugos arborétum, váratlanul egymásba visszakanyarodó ösvényekkel, és egyedül a játékoson múljon, hogy merrefelé mozog rajtuk – ne akarjuk a szájába rágni semmilyen előzetes elképzelésünket. (Vagy ha igen, akkor azt úgy tegyük, hogy ő ezt ne vegye észre…) A játékban éppenhogy az elrejtett, nem látható dolgok jelentik a legfőbb vonzerőt – ha *tudjuk*, hogy ott vannak, és mégse találjuk meg őket…

Nagyon sokan elrontják a játékot azzal, hogy mindenáron valamilyen történetet akarnak elmesélni általa; holott egy kalandjáték egyáltalán nem erről szól! Az persze nem baj, ha van ilyen – sőt, egy jól kitalált kerettörténet sokat emel a játék színvonalán –, de vegyük tudomásul, hogy ez egy másodlagos dolog: kicsit szigorúan azt is mondhatnánk, hogy a bevezetőnél tovább nem szabad(na) menni ezzel. Miért? Ha mi elejétől a végéig kimódoljuk a cselekményt, majd a szereplőket arra kényszerítjük, hogy lépésről lépésre ezen az úton haladjanak végig, azzal elvesszük tőlük azt az illúziót, amiért voltaképpen leültek a számítógép elé: hogy a játék tőlük függ, hogy szabadon ők alakítják az eseményeket. Ha így teszünk, azzal nyíltan uralkodni akarunk a főhősön, amivel bizony könnyen elvehetjük a kedvét a játéktól – az ilyesmi csakis ellenérzéseket szülhet. Az efféle túlszerkesztést leggyakrabban attól való félelem motiválja, hogy a játékos esetleg átsiklik a kedvenc ötleteink fölött, egyszerűen nem veszi észre azt, amit mi a legnagyobb poénnak szántunk – s ennek elkerülése végett mindenáron megpróbáljuk bebiztosítani a dolgot, mintegy szájbarágni a következő lépést. És ezzel – akaratlanul is – agyoncsapjuk az egészet: kapkodva lerántjuk a leplet az összes titokról, míg végül szegény kalandor ott áll teljesen kiábrándultan. Egy kalandjáték tartalmát nem egy történetnek az elmesélése adja, hanem hogy tulajdon képzeletünkből merítve fölépítünk egy miniatűr világot, ami a megfelelő eszközökkel azt a látszatot kelti magáról, hogy él – s ennek a célnak kimondottan jót tesz, ha az őt alkotó részletek és epizódok lazán szétszórtan hevernek, és minél kevesebb összefüggést mutatnak egymással a felszínen. (Hosszú ideig eltart, mire az ember megsejti a látszólagos káosz mögött mélyen meghúzódó rendet.) A jól sikerült játék nem egy vagy több, előre elrendelt útvonalat jelent, hanem úgyszólván egyszerre terjeszkedik valamennyi irányban.

Minden lehetséges eszközzel akadályozni kell, hogy világunkról pontos, részletes térkép készüljön! Álljon minél nagyobb számú helyiségből az egész, és azok is minél bonyolultabban kapcsolódjanak egymáshoz (pl. ne csak vízszintesen, hanem függőlegesen is keresztül-kasul, akár egy többszintes barlangrendszer esetében) – úgyhogy ha valami vakmerő nekiállna ábrázolni őket, hát minimum A1-es papírra legyen hozzá szüksége… Minél több változó körülmény legyen! (Pl.: sötét helyek, ahol világítani kell, de a lámpában egy idő múlva kimerül az elem; véletlen utakon bolyongó, csavargó szereplők; pénz, amivel vásárolni lehet, de igencsak szűkében mérjük számára; vízalatti zugok, melyekben csak rövid ideig tartózkodhatunk megfulladás nélkül; napszakok váltakozása – éjszakára bezárnak a boltok, becsukják a városkaput – stb.) Szélsőséges esetben akár még az is elképzelhető, hogy napszakok szerint változik a helyszínek leírása, úgyhogy ha éjszaka járunk ugyanazon a helyen, akkor meglehet, egészen más dolgokat találunk ott, mint nappal… Csak részletesebb

vizsgálódás árán fölfedezhető rejtekutak kellenek (festmény mögötti rejtekajtó, csapóajtó az ágy alatt, létra a kútban…), lelakatolt ajtók, egyirányú átjárók, helyüket változtató őrszemek. Sok-sok csukott ajtó legyen, és más hasonló akadályok, s még az sem kizárt, hogy némelyik ajtót netán *sohasem* lehet kinyitni, így a kalandor örökké csak találgathat, mi is lehet mögötte. Roppantul lényeges, hogy minél több olyan epizód szerepeljen benne, melynek nemcsak egyféle megoldása van! Ha mondjuk be szeretnénk jutni egy fallal körülkerített városba, akkor cselekedhetünk úgy is, hogy bizonyos összeget fizetünk a főkapu őrének, aki erre kinyitja nekünk a főkaput; esetleg ha elég erősek vagyunk, megölhetjük és elvehetjük tőle a kulcsokat; valahol, egy másik ponton átmászhatunk a kőfalon; vagy akár egy földalatti alagúton keresztül is behatolhatunk, hogy aztán az egyik lakóház pincéjében bukkanjunk elő… stb. stb. Sose felejtsük el, hogy minél nagyobb szabadságot kap tőlünk a játékos, annál jobban élvezi majd a játékot – igaz, másik fele a dolognak, hogy ez sajnos nagyságrendekkel megnehezíti számunkra a játékprogram megtervezését és elkészítését…

Említettük: a kalandjáték bizonyos szempontból a valóságos világ szimulációjára törekszik. Rendkívül sokat árt ennek a szándéknak, hogy bizonyos konkrét célok elérésére irányuló vonalakba rendeződik a cselekmény – azt a hamis illúziót keltve, mintha az egész világnak egy bizonyos célja és értelme lenne… Ha a játékos elérte a végső célját, akkor egyszercsak mintha elvágták volna: nincs tovább, leállt, filmszakadás, dráma… Attól kezdve már nem is érdekelheti tovább az egész. Segíthet ezen valamennyire a sokféleség, az, hogy mindennek többféle megoldása van; így legalább van értelme mindig újrakezdeni, és az újabb és újabb változatokat próbálni, fölfedezni az elrejtett, mellékes kis részleteket is. De a legjobb (és persze legnehezebben elképzelhető) megoldás az volna, ha egyáltalán nem lenne sem eleje, sem pedig vége az eseményeknek. Ez esetben ténylegesen is föl lehetne használni a valóság jelenségeinek ábrázolására. Lehetne például egy programot írni, ami – valóságos tényekre és tapasztalatokra építve – megjelenítené egy erdő mindennapos életét. Az alapszíntér sokezer, egymáshoz nagyon hasonló helyszínből állna, amely lassan és fokozatosan, de nem szűnő következetességgel folyton-folyvást változna: bizonyos ösvényeket lassanként benőne a csalán, míg előbb-utóbb teljesen eltűnnének, s eközben máshol új, friss csapások képződnének az állatok lábai nyomán – nem lehetne egy állandó térképet rajzolni a világunk fölépítéséről, mert mire elkészülnénk ezzel, addigra talán már teljesen megváltozna a helyiségek elrendezése; sőt, maguk a helyiségek is eltűnnének idővel, hogy azután hasonlóak, de nem pontosan ugyanolyanok bukkanjanak fel valahol egy másik pozícióján. Éppígy cserélődnének a bennüket megtöltő tárgyak és élőlények is: természetesen ehhez min. hetekig vagy hónapokig kellene futtatni a programot, de ennyi idő alatt az oszlopos fák is szépen kiöregednének és kidőlnének (ha szerencsés a főhős, akkor éppen olyankor tartózkodik egy ilyen érdekes esemény helyszínén, amikor az megtörténik, s így néha-néha közvetlenül figyelemmel kísérheti azt), hogy aztán újabb hosszú ideig korhadjanak a földön, mialatt a friss nyiladékban fejlődik a következő nemzedék. A keletkezett tisztásokat benőnék a cserjék és a virágok, rajtuk madarak és rovarok csapataival, eső után előbújnának a csigák és szeptemberben bőgnének a szarvasok… Természetesen az idő múlása is figyelemmel kísérhető lenne – alaposan felgyorsítva, pl. egy másodperc alatt múlna el ott egy perc, egy perc alatt egy óra, következésképpen 24 perc alatt egy nap; a napszakok váltakozásai nyomán változnának a helyiségek leírásai és a bennük található élőlények összetétele; kb. fél nap leforgása alatt pedig már egy teljes hónap

peregne le a szemeink előtt – hat nap alatt telne el egy év… Ezalatt mind a négy évszak megtenné a magáét: téli álomba vonulnának a medvék, leesne a hó és a fák lombjai, februárban malacokat ellene a vaddisznó, majd ezután a következő tavasszal újra kezdődne minden. Ebben a játékban mászkálva nem azon mesterkedne a kalandor, hogy valamilyen célhoz kerüljön mindegyre közelebb, hanem elsősorban a nagyságával és a gazdagságával nyűgözné le őt: bármekkora kirándulóutakat tenne is benne, sohasem fedezhetné föl egészen, és képtelen volna egyszer s mindenkorra emlékezetben tartani annak a seregnyi állat- és növényfajnak a nevét, amelyek életciklusa mind-mind parányi részét nyújtaná a végleges képnek – köztük akár olyan ritkaságok is, amivel jó, ha egy életben egyszer találkozik az ember, vagy a teljes év folyamán mindössze néhány óráig vagy napig található szabadon… Sohasem lehetne megunni, elvégre örökké más-más lenne benne minden, és a kissé részletesebb megismeréshez is rendkívül hosszú ideig kéne foglalkozni vele. Bár őszintén megvallva, a valóságos természetjárás azért valamivel izgalmasabb időtöltés ennél… Nem is ez volna benne az igazi kihívás – hanem egy ilyen programot *elkészíteni*!

Ez volna az egyik út, amelyen tovább lehetne lépni innen; a másik pedig a játékok szöveges voltát használná ki, és magának a Nyelvnek a szimulációjára törne. Történtek is már ez irányban különféle kisebb-nagyobb próbálkozások; de az elfuserált fordító- és beszélgetőprogramok – minden beléjük fektetett szorgalmas és kemény munka ellenére is – igen szánalmas látványt nyújtanak… Nem csoda, ugyanis ez még az előbbinél is jóval hatalmasabb feladat volna, és nem tudom, hogy egyáltalán megvalósítható-e. Roppant érdekes adalékok ehhez a témához Noam Chomskynak, az amerikai nyelvésznek és filozófusnak magyarul csak nemrég megjelent *Mondattani szerkezetek* valamint *Nyelv és elme* című írásai – előbbiben az emberi beszédnek egy pontos matematikai modelljét igyekszik fölállítani, összetevőkkel és ún. nyelvi transzformációk segítségével; utóbbiban azt foglalja össze, milyen hatással volt és lehet a nyelvészet az emberi elme kutatására. Számtan és nyelvtan összekapcsolására mások is tettek már kísérletet, pl. a magyar Kiss Dénes is – de az övé jóval kevésbé módszeres és alapos, mint a Chomskyé, inkább amolyan ösztönösen ide-oda csapongó.

4.2.) Második rész: A babótábor a számítógépbe megy

4.2.1.) A program működésének leírása

Ehhez a szakdolgozathoz program is készült – J. R. R. Tolkien: A babó című regénye alapján, azonos címmel. Még annak idején, C64-en írtam egy – hivatalosan is forgalmazásra került – kalandjátékot (A Gálya volt a címe), mely egy, a lehetőségekhez képest eléggé intelligens szövegértelmezőt tartalmazott (az ismertetett szóragozó algoritmusok alkalmazásával), benne volt a távoli célpont megközelítése, a rekurzív tárgy-keresés, többszáz helyiséggel és tárggyal és hosszú és szövevényes cselekménnyel volt ellátva, és jelentős újdonsága pedig az volt, hogy egyszerre (pontosabban csak felváltva) két főhőst lehetett irányítani benne egymástól függetlenül: ennek a programnak az elkészítése majdnem kereken két évig tartott. (Igaz, hogy grafika is volt benne, amelyet szintén én magam rajzoltam meg hozzá, azonkívül két nyelven írtam

egyszerre, magyarul és angolul…) Nyilvánvalóan egy diplomamunka elkészítéséhez rendelkezésre álló pár hónap alatt nem lehet hasonló színvonalú végeredményt elvárni, ezért A Babó c. játék ennél valamivel egyszerűbb. Illetőleg maga az alapvető irányítási rendszer jóval fejlettebb és bonyolultabb, de a játék tartalma töredéke sincs a másiknak.

A szöveg – írásos – ábrázolásának alapegysége a betű, s nyilván én is innen indultam el. A hagyományos ábécé és az erre alapuló ASCII-kódrendszer eléggé megnehezíti a szavak kezelését bizonyos körülmények között: már eleve is vegyesen vannak benne a magánhangzók és a mássalhangzók, s miután az ASCII-rendszer alapvetően az angol ábécére épül, a PC-s karakterkészletből a magyar ékezetes betűk egy része hiányzik is, a többi pedig, ami megvan, teljesen összevissza, elszórtan helyezkedik el. A fontosabb írásjelek is teljesen ésszerűtlenül lézengenek erre-arra… Így egy karakterről eldönteni, hogy pl. az ábécébe tartozik-e egyáltalán, eléggé macerás dolog, olyan finomságokról, mint a szótár szavainak ábécé-sorrendbe való rendezése, nem is beszélve. Ezért én egy teljesen saját kódrendszert alkalmaztam helyette: ez úgy néz ki, hogy az elején vannak a magánhangzók, azok is szétválasztva (előbb az alacsony, majd a magas hangrendűek), ill. a nullás karakter a szóköz, s mögöttük jönnek a mássalhangzók, aztán egy ugyanilyen blokkban a nagybetűk, a számok, végül pedig az írásjelek zárják a sort – mindösszesen éppen 128 karakter. Annak eldöntése, hogy egy karakter betű-e, szám vagy írásjel, vagy hogy egy betű magánhangzó vagy mássalhangzó-e, egyetlen összehasonlítással történik mindenféle táblázatokban való keresgélések helyett.

A szótár szavai eszerint az új ábécé szerint blokkokba és szegmensekbe csoportosítva helyezkednek el. Egy blokkba tartozik az összes olyan szó, aminek az első két betűje egyezik (kis- és nagybetűk közt nem tesz különbséget), egy szegmensbe pedig az összes olyan blokk, aminek a legelső betűje egyezik. Van még egy speciális, ún. vegyesblokk is, amibe az esetleg nem betűvel (hanem mondjuk számmal stb.) kezdődő, valamint az egybetűs szavak kerülnek. Ezzel a módszerrel bármilyen hatalmas szótár is villámgyorsan kezelhetővé válik: ha egy ismeretlen szót szeretnénk azonosítani, az első két betűje máris kijelöli a blokkot, amiben keresni kell – azonos blokkba pedig még egy sokezer szavas szótár esetén is max. néhány tucatnyi kerülhet. Ezt a pár darab szótárbéli szót az értelmező program minden lehetséges módon végigragozza, s az alakokat sorban egymás után összehasonlítja a beadott szóval. (Erre azért van szükség, mert a szavaknak néha eléggé sokféle ragozott alakja lehet.) Az ellenkező irányból ugyanez az út nem volna járható, vagyis egy ismeretlen szó végéről nem lehetséges lehasítani semmiféle ragot, miután képtelenek volnánk megállapítani, hol végződik a szótő és kezdődik a rag. (Pl. a KÖVET szó egyaránt lehet a KŐ tárgyesete vagy a KÖVETNI ige alakja – de ha ezt általánosítani próbálnánk, akkor a program esetleg a LÖVET igéből is levezetne egy nemlétező LŐ főnevet stb.) Minden egyes szót egy-egy sorszám azonosít, ezt kapjuk vissza végeredményül, valamint a ragozásnak a típusát. A szavak mellett szerepel a szótárban mindegyiknek a sorszáma, továbbá még egyéb információk is: a szófaja (ige, főnév, melléknév…), hogy milyen típusú ragozást igényelnek és hasonlók. A különböző szófajok a szótárban abszolút vegyesen helyezkednek el, a kezdőbetűk törvényének engedve, de a sorszámozás alapján már különválnak egymástól: az elején a főnevek (azon belül is az élőlények, tárgyak stb. – ld. a korábbi fejezeteket), majd a melléknevek, az igék, az igekötők, s végül a speciális, ragozatlan szavak jönnek (névelők, kötőszók stb.). A parancsok igéből, tárgyból, helyhatározóból és eszközhatározóból állnak, és lekezeli a program a

„melléknév + főnév” típusú jelzős szerkezeteket a gyűjtőnevek pontosítására (pl. szerepel benne tizennégy különböző színű csuklya: zöld, sárga… stb.). A legutóbbi főnév helyettesítésére névmások is alkalmazhatók (külön van az élők és élettelenek számára is egy-egy: ŐT és AZT), és egy mondaton belül több parancs is állhat (ezeket lépésenként hajtja végre). A hiányos mondatokat is lekezeli a program: ha egy fontos főnév vagy ige hiányzik, akkor kiegészítendő jellegű kérdéseket tesz föl arra vonatkozólag. A játék teljes szótára kb. 1000 szót tartalmaz – jóllehet ezek többsége nem igazán van kihasználva, a jelenlévő főnevek és tárgyak túlnyomó része inkább csak leíró, díszlet jellegű.

Az egyes szófajok kezelése és megkülönböztetése úgy történik, hogyha kiértelmeztük a soron következő szót, akkor megvizsgáljuk, hogy az ige-e vagy igekötő – s ha igen, akkor még mielőtt elraknánk a megfelelő változó rekeszébe, ott helyben megpróbáljuk az „ige + igekötő” szókapcsolatot egy összetett igévé alakítani. Ha melléknevet találtunk, úgy azt azonnal eltesszük a melléknév változójába. Innen csak akkor vesszük elő legközelebb, ha már egy főnév is érkezett, s akkor – az igekötőkhöz hasonlóan – megpróbáljuk ezt a jelzős szókapcsolatot is átalakítani. Ezek a lehetséges szópárok valamennyien egy-egy táblázatban fordulnak elő, ahol egyrészt el vannak tárolva az igék és igekötők (ill. melléknevek és főnevek) összerendelt párjai, másrészt az ezekhez tartozó új, kicserélendő igék (ill. főnevek). A táblázat első részével sorra összehasonlítjuk a keresett szavak sorszámait, s ha valahol egyezést találunk, akkor az ahhoz tartozó új szóra cseréljük ki a régit. Hogy a beazonosított főnév tárgy, helyhatározó vagy eszközhatározó lesz-e, azt annak ragozása dönti el, s ennek megfelelően a szükséges változóba kerül. A névmások behelyettesítésekor egy kisebbfajta trükköt kellett alkalmazni, mert ha csak úgy egyszerűen kicserélnénk azt a legutóbbi főnévre, akkor például az ADD ODA A KULCSOT NEKI utasítás esetén (ahol a NEKI a legutóbbi élőlény neve helyett állhat) az elsőként megtalált KULCS főnév tárolódna el legutóbbiként (felülírva a korábbiakat), majd az is helyettesítődnék be, s így helytelenül az ADD KULCSOT KULCSNAK parancsot kapnánk belőle végeredményül… Ennek elkerülése érdekében a névmásokat két példányban kell tárolni: külön van a régi, ahonnan a behelyettesítéskor mindig kiolvasunk, valamint az új, ahová pedig mindig beírjuk az éppen megtalált főnév sorszámát; a parancs végére érve azután az újból áttöltjük az ott tárolt értéket a régibe, s így a következő parancs már azt veszi majd át. Ezenkívül ráadásul kétféle különböző névmást vezettem be az élő (Ő, ŐT, NEKI, ŐNEKI, ŐVELE… stb.) és az általános (nemcsak élettelen, hanem mindent, így az élőket is magába foglaló) objektumok számára (AZT, ANNAK, VELE, AZZAL… stb. alakok). Sőt, még egy harmadik típusú személyes névmás is létezik, ez pedig magát a játékost, tehát az általa irányított jelenlegi szereplőt helyettesíti (ÉN, ENGEM, TE, TÉGED, MAGAD, MAGAM… stb. alakok). Az eddigiekben elmondottak lehetővé teszik számunkra, hogy a megfogalmazott utasításaink teljesen szabad szórendűek legyenek – elvégre is a végrehajtás nem kezdődik meg mindaddig, míg a teljes parancsot be nem olvasta a program, és valamennyi szófaj és mondatelem egymástól független módon kezelődik. (Leszámítva, hogy a melléknévnek a főnévnél előbb, az igekötőnek pedig az igénél hátrébb muszáj állnia.)

Ez a program tipikusan egy többszereplős kalandjáték: akár tizennégy főhőst is alakíthatunk benne, amennyiben kedvünk tartja (az eredeti történet főszereplőit), s mindig követhetjük a többiek szemével is azt, hogy mit csinál az egyik. Ezt ráadásul úgy oldottam meg, hogy egy, kettő vagy négy, egymástól teljesen független ablakra

oszthatjuk a képernyőt, s valamennyi ilyen ablakban egy-egy külön szereplőt irányíthatunk: ezeknek az ablakoknak a kezelése teljesen párhuzamosan zajlik – akár mind a négyben egyidejűleg (!!!) is történhet különböző szövegeknek a folyamatos kiíratása. Mindeközben szakadatlanul szerkeszthetjük a képernyő legalján fönntartott beviteli mezőben a következő mondatunkat, s azt bármelyik ablak számára el lehet küldeni. Mindebből következik, hogy a program természetesen valósidejű időzítéseket alkalmaz. A másodpercenként kb. 18.2-szer végrehajtódó IRQ megszakításra rászinkronizálva egy megszakítás-alprogram kb. 1/3 másodpercenként generál egy órajelet az összes szereplő számára, s valamennyi szereplőhöz tartozik egy számláló, ami azt tartja nyilván, hány db. ilyen órajel múlva következik lépésre. Az egyes szereplők számlálói 2-5 másodperces időközökre vannak beállítva, személyenként különbözőképpen (pl. a testes Bombur mozgása lassúbb, mint a fürge Bilbóé). Minden szereplő számára fönntart a program egy-egy saját puffert, ahová a végrehajtandó utasításai kerülnek (ilyenkor még szövegesen, kiértelmezetlenül), s innen veszi elő egyesével őket. Ha az aktuális szereplőnknek egy friss mondatot begépelünk, miközben az előzőt még nem hajtotta végre teljesen, akkor a végrehajtásra váró lépései elvesznek, felülírja őket az új mondat. Mind a négy ablakban lévő szereplőnknek külön utasításokat adhatunk, s azután hátradőlve leshetjük, ahogyan egymással versengve végrehajtják azokat…

Az ablakokba történő kiíratás ugyancsak – többszörösen – pufferelten történik, részben ez teszi lehetővé, hogy egymással párhuzamosan használhassuk őket. Az ablakok itt valóban ablakot jelentenek, amelyeket a hozzájuk tartozó – jóval nagyobb méretű – szövegpufferok területén fölfelé-lefelé egyaránt mozgatni lehet (így pl. visszanézhetjük a korábban kiírt szövegeinket is). Négy ablak létezik és négy puffer, ezek közül bármelyik puffer bármelyik ablakban megjelenhet (egyszerre többen is akár). A szövegpufferok tartalmát különféle mutatók tartják karban: létezik egy-egy a szöveg kezdetének és végének meghatározására, arra, hogy hol helyezkedik el éppen az ablak a szövegen, és hogy a szöveg mely pontjától kezdődik annak még érintetlen, „szűz” része, ami még sohasem volt a képernyőn, s ezért azt külön felszólítás nélkül is ki kell íratni. Egy-egy másik változó őrzi a pufferban lévő foglalt és szabad terület méretét. Egy üzenet kiírása a pufferba úgy történik, hogy először is fölszabadítjuk számára a szükséges helyet, majd onnantól bemásoljuk a karaktereket, s ez hozzáépül az alsó „szűz” részhez (hozzáigazítjuk a megfelelő mutatókat). Ezután kezd el ez soronként innen kiaraszolni a képernyőre – ha több volna, mint amennyi egyszerre kifér, akkor a jobb alsó sarokban egy kis szamárfül jelenik meg, és ENTER-re folytatódik a kiírás (de bizonyos idő elteltével magától is továbblépteti a szöveget). Az ablakok megjelenítésére a program grafikus képernyőt használ, a ma már alapnak számító 640*480*256 színű SVGA üzemmódban (ill. ha ez nem elérhető, akkor a 320*200*256 színű módban). Az ablakokban levő szöveg görgetése nem karakterenként, hanem képpontokként történik! (Állíthatóan 1,2,4… stb. pixelenként egy lépésben.) A program karakterkészlete 8*16 pontos karakterekből áll. Hogy mind a négy ablakban párhuzamosan történhet a kiírás, az annak köszönhető, hogy egyszerre mindig csak egy pixelsornyit mozdítunk rajta, majd vesszük a következő ablakot és azon is, és így tovább. Ehhez még hozzájön a beviteli sornak a szerkesztése is, valamint a szereplők beidőzített lépéseinek a végrehajtása – pontosan a 4.1.10. fejezetben elmondottaknak megfelelően. Hogy ne döcögjön és normális tempóval haladjon mind a négy szöveg, ehhez igen nagy sebességre van szükség, ami ASSEMBLY nyelvű

programozást és a 386-os kódú 32-bites regiszterek használatát vonja maga után. 386-osnál ócskább gépen (vagy ha nincsen hozzá VGA) emiatt nem grafikus, hanem csak sima szöveges karakteres képernyőn jeleníti meg a program. (Tehát elméletileg még egy Hercules monitoros XT-n is működnie kell – bár kipróbálni sajnos nem volt lehetőségem.) Egyébként is, az egész program színtiszta ASSEMBLY-ben készült…

A helyiségek és a térkép tárolása pontosan a 4.1.5. fejezetben ismertetett bejárási táblázat formátumának megfelelően történik. A játékban mintegy 120 helyiség található. A tárgyak térképen való elhelyezése és nyilvántartása szintén a 4.1.7. és 4.1.8. fejezetekben leírt statikus sorszám- és rekurzív táblázatkezelés elve alapján zajlik. Ha egy helyiségben másodszor járunk, utána már a teljes leírás helyett csupán pár szavas nevét írja ki nekünk a program. A tárgyak befogadóképessége is súly szerint korlátozva van.

Az egyes igékhez különböző belső eljárásai vannak hozzárendelve a programnak, ezeknek kezdőcímei egy táblázatba foglalva várják, hogy az ige sorszámával kiindexeljük és végrehajtsuk őket. Még az ige végrehajtása előtt ellenőrzi azonban, hogy a megadott főnevek helyesek-e és elérhetőek, s ha minden stimmel, akkor ugrik tovább a megfelelő eljárásra. Itt még további ellenőrzéseket végez, s ha sikerült végrehajtania, akkor a tájékoztató jellegű üzeneteket egy speciális kiíró eljáráson keresztül vezeti át az ablakok puffereibe – ez a szubrutin mindig végigteszteli mind a négy ablakot és azok „tulajdonosait”, s amelyik ezek közül ugyanazon a helyszínen tartózkodik, annak számára írja ki a megfelelően átalakított üzeneteket. (Pl. „Felveszed a lámpát.” helyett „Bilbó felveszi a lámpát.” stb.) Ha olyan főnév szerepel benne, amelyik több helyszínen is jelen van egyszerre (pl. egy ajtó két szobát köt össze), akkor a vele történt eseményről valamennyi helyen tudósít – de úgy, hogy a mondatban szereplő további főnevek közül, ha valamelyik nem elérhető, akkor helyette a „valami” ill. „valaki” szót használja a program (attól függően, hogy élőlényről van-e szó). Pl. ha kopogtatunk egy ajtón, akkor annak túloldalán a „Valaki kopog az ajtón.” üzenetet kapjuk. Akkor is ezek a határozatlan névmások kerülnek elő, ha a kiírás egy sötét helyen történik; és minden ilyen esetben a főnévvel együtt az ige ragozása is módosul („felveszed” helyett „felveszel” – tárgyas helyett alanyi ragozást kap – stb.). Ha a hivatkozás az ablakban irányított szereplőre vonatkozik, az ő neve helyett a „te” névmás megfelelően ragozott alakjai („téged”, „neked”… stb.) jelennek meg; és hasonlóképpen bánik a „maga” (ha a cselekvő egyezik a tárggyal) ill. a „magad” (ha mindhárom főnév – tehát a cselekvő, a szemlélő és a tárgy – is ugyanaz) személyes névmásokkal is. Külön érdekessége a játéknak, hogy az eredeti regényben szereplő varázslatos Gyűrű az őt viselőnek *láthatatlanságot* ad – ez a szereplő tehát egyszerűen bármit megtehet a játékban anélkül, hogy a többieket erről értesítenie kéne. (Következésképpen bizonyos akadályokon is átmehet, pl. ha valahol egy őrszem senkit nem enged át, ott őneki akkor is szabad bejárása kell legyen, mert nem láthatják, amikor áthalad; vagy elveheti bárkitől a nála levő tárgyakat stb.)

A végrehajtást követően a vezérlő főprogram minduntalan visszatér a beviteli-megjelenítési végtelen főciklusba, ahonnan csak az egyes szereplőknek adott következő órajelek mozdítják majd ki a programot. (Kicsit emlékeztet ez a működés a Windows eseménykezelőire: mindig a kívánt esemény bekövetkezte indítja el a hozzá tartozó lekezelő eljárást.)

4.2.2.) Használati útmutató

Mihelyt a programot elindítottuk, mindjárt a grafikus képernyőn elhelyezkedő négy darab ablakkal nézünk farkasszemet. Az egyes ablakokban a hozzájuk tartozó puffer szövege foglal helyet (hangulatos gótbetűs karakterkészlettel), a körülöttük elhelyezkedő egyéb információkból azonban egyszerre mindig csak a kiválasztott ablakhoz tartozót láthatjuk. Ezek: fölül az állapotsor, alul a beviteli mező, jobboldalt pedig az elmaradhatatlan szövegkocka. Ez utóbbi a szokványos stílusban mutatja az ablaknak a puffer szövegében való elhelyezkedését, és mindig pontosan vele együtt mozog. A beviteli mezőben szerkeszthetjük bármikor, folyamatosan a következőleg kiadásra szánt mondatunk szövegét. Ezt a billentyűzeten való gépeléssel tehetjük meg; a programban használt billentyűzetkiosztás alapállapotban 99%-ban a magyar ékezetes billentyűzet pontos kiosztásának felel meg (ékezetes betűk az írásjeleken, Y és Z felcserélve stb.), ám amennyiben ez nekünk nem tetszene, úgy a CONTROL + CAPS LOCK billentyűkombinációval tetszőlegesen átkapcsolhatunk a hagyományos angol billentyűzet kiosztására is. Ilyenkor az ékezetes betűket az ALT váltóbillentyűvel együtt nyomva érhetjük el (pl. ALT + A = Á… stb.), ill. az Ö, Ő, Ü, Ű karaktereket a CONTROL + O, I, U, Y kombinációkkal; ezek nagybetűs megfelelői a kisbetűsek „alatt” lévő gombokon vannak hasonlóképpen elhelyezve (pl. ALT + Z = nagy Á, CONTROL + L = nagy Ö stb.); magyar billentyűzet esetén mindezek a kombinációk teljességgel hatástalanok. Magyar billentyűzeten a nagybetűket egyszerűen SHIFT-elve csalogathatjuk elő; amellett a CAPS LOCK is használható a maga szokásos funkciójában, ami ezúttal már az ékezetes billentyűkre is érvényes lesz – ám minderre egyszerűen nincsen szükség, tudniillik a program egyáltalában nem tesz különbséget a kis- és nagybetűvel megfogalmazott utasítások között. Az angol billentyűkiosztásról a magyarra ugyancsak a CONTROL + CAPS LOCK segítségével térhetünk vissza. A NUM LOCK billentyű is megőrizte elfogadott szerepét.

A szöveg szerkesztése közben a kurzorvezérlő JOBBRA/BALRA billentyűkkel tetszőlegesen előre-hátra haladhatunk a mondatunk karakterei között, a CONTROL + JOBBRA/BALRA révén pedig ugyanezt nem betűnként, hanem teljes szavanként ugrálva tehetjük meg. Hogy a szöveg belsejében beszúrunk-e vagy felülírunk, azt az INSERT kapcsoló megfelelő állapota dönti el. Max. 256 karakternyi hosszúságú szöveget írhatunk be egy szuszra – ennyi viszont nem fér ki a beviteli sorban, ezért ilyenkor a program megfelelően előre-hátra görgeti a szöveget, és csak a szükséges részét láthatjuk belőle. A HOME és END gombokkal a teljes szöveg elejére/végére ugrunk. A DELETE az ismerős módon letörli a kurzor pozícióján álló karaktert, a BACKSPACE pedig egyet balra lépve töröl; kellemes újdonság viszont, hogy a CONTROL + BACKSPACE kombinációval itt is kiterjeszthetjük a törlés szerepét egy egész szóra nézve (amiben a kurzor áll vagy balra tőle található, plusz a hozzá fűzött írásjeleket is). A CONTROL + INSERT a kurzortól balra, a CONTROL + DELETE pedig a jobbra fekvő teljes szövegmennyiséget tünteti el végérvényesen. Az ENTER-rel tudjuk elküldeni az elkészült mondatot, a CONTROL + ENTER pedig a mostani és a legutóbb elküldött mondat szövegét *felcseréli* egymással – ennek ismételt megnyomására a szövegek újfent visszacserélődnek. Ezáltal nemcsak az utolsó mondat ismétlésére nyílik lehetőség, de váltogathatjuk és párhuzamosan szerkeszthetjük mind a két mondatot, majd a végén határozhatunk, hogy melyiküket dobjuk el.

Sőt, nemcsak ezt a kettőt tudjuk ilyen szimultán módon alakítani, hanem egyenesen mind a négy ablakhoz külön-külön inputsorok vannak rendelve! Ezek közül természetesen mindig csak a jelenlegi ablakhoz tartozót láthatjuk itt alant. Az aktuális ablak a színében különbözik az összes többitől, és a TAB ill. SHIFT + TAB billentyűkkel lapozgathatunk előre-hátra az egyes ablakok között; az F1…F4 által pedig közvetlenül valamelyik ablak kijelzésére válthatunk. Az ablakok cseréjével együtt az állapotsor, beviteli sor és a szövegkocka is rendesen a kiválasztottnak megfelelően ugrál. Említettem, hogy a négy ablak mellett négy puffer is létezik: a pufferokhoz tartoznak a szereplők, s az ablakokban jelennek meg a pufferok… Azt, hogy a jelenlegi ablakban melyik puffert óhajtjuk megjeleníteni, a CONTROL + TAB ill. SHIFT + CONTROL + TAB révén lapozhatjuk, vagy közvetlenül az F5…F8 billentyűkkel kapcsolhatjuk be; ezáltal tetszőlegesen beállíthatjuk akár mind a négy ablakba is ugyanazt a puffert (és akkor mindenütt ugyanazt a szöveget látjuk…), de jobb inkább nem piszkálni, mert könnyen összekuszálhatjuk vele a dolgokat. Az ESCAPE használatával ugyanakkor a pillanatnyi ablakot kinagyíthatjuk a teljes képernyő nagyságára (vagy újabb megnyomásával vissza); a CONTROL + ESCAPE és SHIFT + CONTROL + ESCAPE segítségével pedig a képernyő négyféle felosztását váltogathatjuk: azt, hogy egyetlen ablak foglalja el az egész képernyőt, kettő legyen vízszintesen, kettő függőlegesen, vagy egyszerre tekintsük át mind a négyet. Akármekkora méretű ablakokkal „dolgozzunk” is, a szöveg mindig annak méreteihez igazítva, megfelelő hosszúságú sorokra széttördelve jelenik meg majd benne, úgy, hogy soha szó egy sor végén meg ne törjön (ha nem fér ki, akkor a következő sor elején folytatódik); és mindegy, hogy hány ablakot látunk át egyszerre, akkor is mind a négyben zavartalanul folyhat a kiírás – TAB és társai is ugyanúgy működnek mind ekkor. (Legfeljebb elsőre egy kissé furcsa lesz a látvány.) Ablakok váltogatásakor az iméntiekben említett CONTROL + ENTER sikerrel alkalmazható az inputsorban található szövegnek egyik ablakból a másikba történő átvitelére is: először kicseréljük vele a mondatot, s így a jelenlegi a „tartalékba” tolódik, majd a másik ablakba átlépve újból megnyomjuk – mire a háttérből előkerül az eltárolt mondatunk.

Az ablakokat nem véletlenül hívják ablaknak, hiszen a puffernak mindig csak egy részét látjuk bennük. Ezt az ablakot a pufferon belül szabadon föl-le tologathatjuk a kurzor FÖL/LE vezérlőbillentyűivel, illetve ugrálhatunk is benne a PAGE UP/PAGE DOWN segítségével; a CONTROL + HOME és CONTROL + END a puffer elejére és végére lépteti az állást. A CONTROL + PAGE UP/PAGE DOWN ugyancsak egy-egy oldalnak megfelelő mennyiséget halad, de nem ugrik, hanem folyamatosan görgeti el azt. A CONTROL + FÖL/LE pedig folyamatosan görgeti az ablakot a szöveg elejéig/végéig. (Próbáljuk ki: ezekkel akár egyidejűleg mozgathatjuk *mind a négy* ablakot is különböző irányokba! Olyan, mintha a páternoszter fülkéi haladnának…) Igazi különlegessége a programnak, hogy a szövegek görgetése nem karakteresen, hanem finoman, képpontokra bontva történik! Ennek a sebességét (pontosabban a finomságát) állíthatjuk is: 1, 2, 4, 8, 16, 32, 64 vagy 128 pixeles lépésközökkel is vonulhat a látvány. Az F11-gyel lehet gyorsítani (durvítani), az F12-vel pedig lassítani (finomítani). Minthogy betűink 16 képpont magasak, ez 1 képpontos mozgás esetén azt jelenti, hogy 16 fázisban tolódik ki a képre egyetlen karaktersornak megfelelő mennyiség (ez a legfinomabb, egyenletes mozgás – de az esetek többségében a leglassúbb is egyben); 16 képpontosnál éppen karakterenként halad; míg a 128 pixeles esetében már egyenesen 8 karaktersoronként ugrál (ezt csak akkor van értelme

használni, ha valami különösen lassú gépünk van, ami egyszerűen nem bírja szuflával a görgetést…). A megjelenítés a képernyőfrissítés elektronsugár-visszafutásához van szinkronizálva (ami a legtöbb monitornál 50-70 Hz), így ennek és a processzor gyorsaságának megfelelően állíthatjuk be magunknak a legkellemesebb verziót. Az átmenetileg történő lassításra van még egy lehetőség, az, hogyha PAUSE állapotba tesszük a gépünket (mi mással, mint a PAUSE billentyűvel): ennek meglétét a szövegkocka invertálódó benyomódása jelzi (kikapcsolni ismételt megnyomással lehetséges): ilyenkor valamennyi mozgás kb. az eredeti egytizenhatod-részére lassul, és például a túl gyorsan távozó szöveget kiválóan el lehet olvasni közben.

A kiírás során igen lényeges szerepe van még a SCROLL LOCK kapcsolónak is: ameddig ez bekapcsolt állapotban van, addig a gép automatikusan tördeli nekünk a hosszú szöveget, vagyis ha nem fér ki egyben a képernyőre, akkor egy oldal után megtorpan, és egy ENTER lenyomására várakozik a továbbléptetése érdekében; addig is az ablak jobb alsó sarkában megjelenő kis behajtott szamárfül jelzi ezt (ill. karakteres képernyőnél a <Tovább> felirat). Míg ellenben kikapcsolt SCROLL LOCK mellett semmi ilyesmi nem történik: a szövegek megállás nélkül vágtáznak keresztül a képernyőn – ilyenkor jön jól az iménti PAUSE funkció. Ha az ablakunk ilyen várakozó állapotban van, akkor addig nem tudunk új parancsot elküldeni, amíg előzetesen tovább nem léptettük a szöveget (az ENTER ilyenkor foglalt erre az új szerepkörre). De más módokon is léptethetjük tovább: a KURZOR LE vagy a PAGE DOWN gombokkal – esetleg egy CONTROL + END-del azonnal a legaljára pottyanunk. (Akkor is megszűnik a várakozás, hogyha fölfelé haladunk, ám ekkor már nekünk magunknak kell legörgetnünk majd.) Ha sokáig nem nyúlunk hozzá, az ablak időnként önmagától is araszolgat egy-egy sort (kb. tizenöt másodpercenként egyet). Ha viszont nem kívánunk egyesével vacakolni mind a négy ablakkal, akkor vagy kapcsoljuk ki a SCROLL LOCK-ot, vagy használjuk a BREAK (= CONTROL + PAUSE) billentyűt, amely az összes ablakot egyszerre alapállapotba hozza (leugrik az aljára, és mindenfajta késleltetést megszüntet).

Két különleges szerepű billentyű maradt ki csupán a felsorolásból: az egyik a PRINT SCREEN (= CONTROL + SYS REQ), amelyik egy SCRNxxxx.TXT file képében elmenti a jelenlegi ablak pufferének teljes szövegét a programkönyvtár MENTÉS alkönyvtárába (ahol xxxx egy 0000-tól 9999-ig terjedő sorszám, úgyhogy rengeteg ilyen lehet); a másik pedig a CONTROL + ALT + DELETE – nem kell megijedni, nem fogja RESET-elni a gépet, mindössze a játékból való azonnali kilépésre szolgál (visszatérés DOS-ba). A program emellett még egy beépített, önműködő képernyővédő funkcióval is rendelkezik: ha öt percig nem nyúlunk a billentyűzethez, akkor lekapcsolja a képernyőt, amelyen egy – a Norton Commander-éhez hasonló – csillagmezőt látunk a következő lenyomásig tündökölni.

Most már tehát be tudjuk gépelni a programnak a kiadni óhajtott utasításainkat – igen ám, de hogyan is fogalmazzuk meg őket? Felszólító módban megfogalmazott mondatokat vár el tőlünk a játék, úgymint MENJ ÉSZAKRA vagy GYÚJTSD MEG A LÁMPÁT A GYUFÁVAL stb. A parancsok szórendje majdnem teljesen szabad, tehát az előző példát akár beírhatjuk a következőképpen is: A GYUFÁVAL A LÁMPÁT GYÚJTSD MEG… Vagy bármilyen más sorrendben – egyetlen megkötés, hogy az igekötőnek az igénél hátrébb kötelező állnia valahol, s a melléknévnek pedig a főnévnél előbb (nem mondhatjuk pl. azt, hogy MEG GYÚJTSD vagy CSUKLYÁT ZÖLD, de ezeknek amúgy se lenne értelme, ezért nem is jutna eszébe senkinek sem leírni

hasonlót). Mivel ez nagyrészt az előző fejezetben már elhangzott, itt csak röviden összefoglalnám a továbbiakat. Tehát több hasonló tárgy megkülönböztetésére melléknévi jelzőket alkalmazhatunk (FOGD MEG A NAGY KULCSOT vagy DOBD EL A CIFRA KULCSOT stb.), a legutóbbi főnevek pedig névmásokkal helyettesíthetők (AZT, ŐT és MAGAD és ezek másféle ragozású alakjai). Egyetlen mondaton belül akárhány különböző parancs is szerepelhet (ponttal, vesszővel vagy ÉS-sel elválasztva), és természetesen az összes felsorolt mondatelemet kombinálhatjuk is egymással. Pl. így: VEDD FEL A NAGY KULCSOT ÉS NYISD KI VELE A KŐAJTÓT. Ha valami hiányzik a mondatból, akkor a gép célzatosan rákérdez: VEDD LE hatására pl. úgy, hogy „Mit akarsz levenni?" – ezekre a kérdésekre elegendő már csak a hiányzó mondatelemeket megadni (pl. KABÁTOT). De valami teljesen új mondatot is beírhatunk akár. Természetesen, ha nem ért meg egy szót, azt is jelzi a program. A mozgási parancsok és néhány más sokszor használt ige is a kezdőbetűivel rövidíthető. (Pl. ÉSZAK, mint É, VIZSGÁLD, mint V, VEDD FEL, mint VF, RAKD LE, mint R, LELTÁR vagy LISTA, mint LI vagy rövidebben I… stb.) Az egyes – általunk is irányítható – szereplők közül úgy választhatunk ki valakit, hogy egyszerűen, ige és ragozás nélkül leírjuk a nevét. A választható szereplők listája: Bilbó, Thorin, Dwalin, Balin, Fíli, Kíli, Dori, Nori, Ori, Óin, Glóin, Bifur, Bofur és Bombur. Hogyha teszemazt valamelyik ablakban leírjuk, hogy NORI, akkor attól kezdve abban az ablakban (illetőleg az ahhoz az ablakhoz tartozó pufferban) őt fogjuk irányítani. De őrajtuk kívül is még további illetőket bevonhatunk a játék menetébe, azáltal, hogy különféle kéréseket vagy parancsokat intézünk hozzájuk (pl.: MONDD MEG GANDALFNAK, HOGY ADJA NEKEM A TÉRKÉPET), s így közvetve manipulálhatjuk őket is. Ezeknél a parancsoknál lényeges, hogy a kérő utasítás (MONDD, KÉRD, UTASÍTSD) és a tényleges tennivaló között valamilyen elválasztójelet hagyjunk: például egy vesszőt, egy idézőjelet vagy egy HOGY szócskát – mint az az idézett példában is látható. Mi több, akár még több ilyen parancsot is egymásba ágyazhatunk, mondjuk valahogy így: MONDD MEG ÓINNAK, HOGY SZÓLJON GLÓINNAK, HOGY KÉRJE MEG BIFURT, HOGY NYISSA KI AZ AJTÓT… Az emberkék majd szépen egymásnak adogatják tovább a mondatot, míg végül az le nem bomlik a végső mondanivalóig, ami már közvetlenül is végrehajtható.

5.) IRODALOM

Abonyi, Zsolt. (1996). *PC hardver kézikönyv*. ComputerBooks, Budapest.

Agárdi, Gábor. (1995). *IBM Gyakorlati Assembly haladóknak*. LSI Oktatóközpont és A Mikroelektronika Alkalmazásának Kultúrájáért Alapítvány.

Chomsky, Noam. (1995). *Mondattani szerkezetek – Nyelv és elme*. Osiris-Századvég, Budapest.

DaCosta, F. (1986). *A kalandprogram írásának rejtelmei*. Műszaki Könyvkiadó, Budapest.

Gidófalvi, Zoltán Dr. (1995). *Programozás MASM Assembly nyelven*. Műegyetemi Kiadó.

Jankovics, Marcell. (1996). *Ahol a madár se jár*. Pontifex Kiadó.

Kiss, Dénes. (1993). *ŐSnyelv – nyelvŐS?* Antológia Kiadó, Lakitelek.

Kiss, László és Schmidt, Endre. (1988). *1001/2 játék*. LSI Alkalmazástechnikai Tanácsadó Szolgálat, Budapest.

László, József. (1995). *A VGA-kártya programozása Pascal és Assembly nyelven*. ComputerBooks, Budapest.

Tolkien, J. R. R. (1992). *A babó*. Ciceró.

A cikk a világhálón: **http://istennyila.hu/hun/program/0002/0000.htm**

A szerző honlapja: **http://istennyila.hu/**

Olessák Róbert (1997-2011)