A MANUAL FOR EAMON ADVENTURE DESIGNERS

EAMON is a computerized fantasy role-playing game that was originated by Donald Brown and developed thru version 6.2 by John Nelson. This version, 7.1, was developed by Tom Zuchowski from v6.2. This manual will help you design your own Eamon scenarios.

When a player embarks on an adventure, his character is taken from the CHARACTERS file, and its record in CHARACTERS is marked empty. Its name and CHARACTERS record number is recorded in a file named THE.ADVENTURER, and the player is prompted to insert the adventure disk into the disk drive. The character's attributes are put into the FRESH.MEAT file, and EAMON.NAME is opened to obtain the name of the adventure. The MAIN.HALL program then RUNs the program that has the same name as the adventure.

Once the adventure is over, control is returned to the Master diskette. If the character died, THE.ADVENTURER is deleted, & MAIN.HALL is run. If the character survived, he is replaced in the CHARACTERS file. The MAIN.PGM rewrites him into his own character record now. Normally this is simply writing of the new information of the character into his old record. If your program has the ability to quit for a while and come back later, it is possible that a new character might be in the old record. In this case you should search the CHARACTERS file for a free record, and write him in there (for more info list NEW.CHARACTER on the Master diskette). Next, the character's name and record # must be entered in a file called THE.ADVENTURER, and then MAIN.HALL is run. This is all done by the MAIN.PGM.

USE OF THE DUNGEON DESIGN DISKETTE

IMPORTANT NOTE: There are 2 edit & list programs on this disk. The programs DUNGEON.EDIT and DUNGEON.LIST cannot be used on a version 7 Eamon. Use EAMON.EDIT.V7.1 and EAMON.LIST.V7.1 for version 7 databases, and DUNGEON.EDIT and DUNGEON.LIST for all others. These programs check the version before entering the database to prevent mistakes, so don't worry about it. See the Utilities Manual for more information.

EAMON.INIT.V7.1

The first step in creating your adventure is to initialize it. The EAMON.INIT.V7.1 program on the DDD (Dungeon Design Diskette) will generate the EAMON.NAME file at the specified prefix. It is not important what number you assign to your adventure; it will be be assigned to the next valid number once you have submitted it to the EAG library.

The next step is to transfer the programs LEADIN, REV.DATE, EAMON.NNN.INTRO, MAIN.PGM, MAKE.FAST.START, & MAKE.ARTS.MONS to the adventure disk/subdirectory. Optionally, if using large drives, you can transfer the files EAMON.EDIT,V7.1, EAMON.LIST.V7.1, and WORK.PREFIX to the subdirectory as well in order to put your tools in with the adventure being worked on.

EAMON.EDIT.V7.1

You should have your dungeon 90% designed before beginning data entry. Decide what rooms you have, how they connect, and what monsters, treasures, and effects are in each room. You should know whether the names of your rooms, monsters, & artifacts will fit in the name size you selected during dungeon intialization. Normally the default file sizes will be more than adequate; in any event the EAMON.EDIT.V7.1 program monitors name lengths to prevent file damage, and the RESIZE.FILES program can be used to change the name lengths.

For each of the 4 things you can enter (ROOM, ARTIFACT, EFFECT, and MONSTER), you can either add a new one to the list, or edit one already there. You cannot delete, but you can replace through editing. You MUST NOT go beyond 200 of any single thing (total 800 for all 4 types). To keep from running out of disk space, it is a good idea to keep track of how much disk space your adventure will use. A rough and conservative rule of thumb for the default lengths is 2/3 disk block per room, monster, & artifact, & 1/2 block per effect.

Adding and Editing is very similar for the 4 data types. Every time you enter more than a single key, the entry is done thru a special input routine. The old Escape-key editing features do not work. Instead, the following control keys do things:

ESC: accepts ALL of the text, both before & after cursor.

RETURN: accepts all of the text from beginning to cursor. Any text following cursor is deleted.

CTRL-B: moves cursor to beginning of text on screen.

CTRL-E: moves cursor to end of text on screen.

CTRL-D: deletes character under cursor.

CTRL-I: inserts a space where cursor is, moving text from cursor to end one space right.

<-,-> (FORWARD & BACKWARD ARROWS): move cursor back or forward one character in text.

Your entries must fit within the space underlined, and you must NEVER use quote marks ("). You may use commas and colons in descriptions ONLY. Do NOT leave any trailing spaces on the names of items ("LION" & "LION " look different to Applesoft). Do not begin artifact names with a number; use eight-inch knife instead of 8-inch knife.

The EAMON.EDIT.V7.1 program has default data for all fields for rooms, artifacts and monsters. This makes it faster to key in data when you want most fields to be the default.

You can learn a lot by listing other adventures' programs and data files. Be fair to the adventurer; an adventurer can generally lick about 3 times his own Hardiness in opponents, with allies subtracting their Hardiness from the opposition. People don't like unbeatable odds, but if it is too easy, it will be boring. Try to strike a balance. Don't use no-warning death traps; it's not fair and won't win you any friends.

EAMON.LIST.V7.1

Once you have entered all of your data, you will want to see it, to catch errors and get a good overview. The EAMON.LIST.V7.1 program on the DDD can list all of your rooms, artifacts, effects, and monsters, naming the "links" you have set up (the name of the room where it is found, the name of the artifact that a monster uses as a weapon, etc.)

Here is a summary of the steps to follow to create an adventure of your own:

1. Be sure you have everything you need:

   a. A theme and a plot.
   b. A map of rooms
   c. A list the monsters
   d. A list of the artifacts
   e. A BACKUP COPY of the DDD (Dungeon Designer's Diskette)

2. Initialize your adventure. Run EAMON.DDD.V7.1 and select the initialize function. Note that it requires that the disk/directory to be initialized must already be formatted.

3. After initialization, you'll need to copy these Applesoft programs from the DDD to your adventure disk/directory:

```
  LEADIN
  REV.DATE
  EAMON.NNN.INTRO
  MAIN.PGM
  MAKE.FAST.START
  MAKE.ARTS.MONS
```

If using a large drive, you can also copy the tools:

```
  EAMON.EDIT.V7.1
  EAMON.LIST.V7.1
  WORK.PREFIX
```

It is not necessary to transfer these 3 files but it can be done for convenience' sake, if desired. If you do this, it will not be possible to return to the Designer menu, but it will not be necessary, either.

4. Enter the adventure data using EAMON.EDIT.V7.1, adding all of your rooms, artifacts, effects, and monsters. This may take days, so whenever you get tired, select the Quit option. All of your items will be saved and you can pick up where you left off.

5. Make your program changes to MAIN.PGM. These will be all of the things that you want to be special about your adventure.

6. The ProDOS Eamon MAIN.PGM expects to find the artifacts and monsters data in a VAR file named FAST.START. Before running/testing your adventure, you must run the program MAKE.FAST.START to generate the FAST.START file from the EAMON.ARTIFACTS and EAMON.MONSTERS text files. The companion program MAKE.ARTS.MONS can generate the two text files from the FAST.START file. Thus you can delete these files from the finished adventure to save disk space.

7. Test your adventure: Using the ProDOS Eamon Master, select a character at the Main Hall and send him on an adventure. Select your adventure and you can then begin your test. From that time on, to repeat a test you should be able to simply RUN MAIN.PGM. (Or if MAIN.PGM is already in memory, simply type RUN). A helpful hint: to restart an adventure that has crashed, type POKE 51,0:GOTO 100.

8. When you are sure your adventure is ready, have a friend test it and/or send it to the EAG for testing. When the adventure

is completely ready, send a copy of it to:

```
        Eamon Adventurer's Guild
        7625 Hawkhaven Dr.
        Clemmons, NC 27012
```

It will be tested for bugs and assigned an "official" Eamon Adventure number, and then may be distributed by you or by the EAG to various public-domain outlets.

If you have any questions or problems designing an adventure, write to the above address. We will do our best to help you get back on the right track.

The EAG is also on-line at GEnie, so you can get your adventure to us by uploading it there, and you can ask questions in the Eamon Category in the A2 RT.

ADAPTING THE MAIN.PGM

You can use the MAIN.PGM exactly as it exists on the DDD, and can simply SAVE it onto your adventure disk.

Lines 100-900 are the main loop that is run thru every turn. Every-turn Special programming should be put in lines 500-900.

Lines 3000-3999 are the movement commands. If you want to magically move the player, set R2 to the number of the room to enter and GOTO 3500. If something happens that makes the monsters reconsider their attitudes towards the player, GOSUB 3600 will allow them to change sides. If you have negative room connections numbers, put the special code at lines 3050-3249

Lines 4600-4699 are the synonym checker. Place your synonyms for actual artifacts here. Set SY$ = the synonym to be checked for, and SY = the # of the artifact that you are checking for. For example, if a Secret Panel that happens to be Artifact #3 is hidden in Room #2, and a hint in the room description says that the wall looks funny, you can put a line like this in your program:

```
 4620 IF RO = 2 THEN SY$ = "WALL":SY = 3: GOSUB 4680
```

If the player then types EXAMINE WALL, the secret panel will 'appear' in the room.

Lines 7000-7999 are the attack commands and subroutines. Lines 7700-7999 kill monster M. If you want dead bodies, see the 'Dead Bodies' discussion in the 'Artifacts' section. To enable 'dead bodies', remove the REM from the front of line 7735 and change 'X' to the number of the FIRST dead body artifact.

Lines 13000-13999 are the Power Spell. It is rather useless in its standard form; make all the changes you want, or even completely replace it if you want to.

Lines 16000-16999 are the Say command, useful for 'words of power', etc.

Lines 28000-28999 are the Use command, which does nothing at all in the basic MAIN.PGM, but is included for your own special programming.

Lines 31000-31999 read in monsters and artifacts from the disk, as well as doing other initializing. To add a new command, change line 31910 (add 1 to the number for each new command), 31930 (add the new verbs--no spaces permitted), and 290 (add the new line numbers to GOTO). If you want
special things to happen at the start, do it in lines 31150-31890

Lines 32000-32999 are the closing routines. If DI < > 0 then the player died. Lines 32100 thru 32290 are for your additions.

Some variables:

```
A%(x,y): Artifact data: 'x' is the artifact #; 'y' is:
  0='Seen' flag
  1=Value
  2=Type
  3=Weight
  4=Room
  5 thru 8: see ARTIFACTS below
AC: Armor
AR: Armor worn
```

```
AE: Armor expertise
A$(*): Artifact names
BA: Gold in bank
BV$(*): Battle verbs
C: Number of command given
C$(*): Valid commands
CH: Charisma
CP: columns (40 or 80)
CZ$: last command
D2: damage to defender
DF: defender
DI: 1=player died
D$:  CTRL-D
EA: player armor factor
ED$: 'EAMON.DESC'
ER$: 'EAMON.ROOMS'
F: 'found' flag used by search routines
F(*): damage to each side in combat
F1, F2: top of free memory
FR: fumble roll/friend rating
GO: player's gold
HI: Hit in combat
L: line counter for screen pause
LA: record length of EAMON.ARTIFACTS
LM: record length of EAMON.MONSTERS
LR: record length of EAMON.ROOMS
LL: string length
LS: artifact number of current light source
LT: light level of room (includes artificial light)
M%(x,y): monster data:  'x' is monster number, 2nd is:
  0='Seen' flag
  1=Hardiness
  2=Agility
  3=# members in group
  4=Courage
  5=Room
  6=Weight
  7=Armor
  8=Weapon #
  9=# Dice
 10=# dice sides
 11=Friendliness
 12=Original size of group
 13=Damage
M$(*): Monster names
MC: counter for group monsters in battle
MR%: Monster morale
NA: Number of artifacts
NC: Number of commands
NE: Number of Effects
NL: natural light level of room
NM: Number of monsters
NR: number of rooms
NW: Number of weapons
NZ: # of artifacts in EAMON.ARTIFACTS
OF: Attacker
R2: room being moved to
R3: room just exited
RB$(*): battle response verbs
RD%(*): room connections
RE: Player record in char file on Master disk
RL: Random number 1-100
RN$: name of current room
RO: # of Room player is in
S$: Object of command
S2%(*): Current spell ability
```

SA%(*): Total spell ability
SE$: Sex (M/F)
SH: shield worn
SL: string length
SM$(*): smile verbs
SP: Counter for speed spell
SU: spell succeeded
SY: artifact # of synonym match
SY$: synonym
T(*): hardiness of each side in combat
TP: value of loot
UP: logical flag if weap. ability up
V$: Verb of command
V%(*): Flag to print room desc.
WA%(*): Player's weapon ability
WA$(*): Player's weapons (exiting pgm)
WD%(*,*): weapon data (exiting pgm)
WP%(*): Weapon pointer (Exiting pgm)
WT: Weight player is carrying

EAMON DATA FILES: ROOMS, ARTIFACTS, EFFECTS, & MONSTERS:

The record sizes of the data files are variable and are determined by the name sizes that you select when initializing your adventure. The default name lengths are: Rooms-38; Artifacts-30; Monsters-30. The description size is fixed at the Dos limitation of 238 characters. The name lengths can be changed using the program RESIZE.FILES.

ROOMS

Each room record contains 9 pieces of data: First is the room name; it will be printed as: "YOU ARE (room name)", so use names like "IN THE HALL". Next is the room description which is not preceded by anything, so it must be a full and complete description. Next you must give the numbers of the rooms that you can get to from that room in each direction. Room numbers from 1 to 199 are normal connections. Use zero for walls, cliffs, etc. The special code of -99 indicates an exit back to the Main Hall. Negative numbers can also be made to have special results by altering the MAIN.PGM. There is no code in the MAIN.PGM for this; you will have to program it yourself. Room numbers greater than 500 are doors. The difference between 500 and the room code is used to point to an artifact. This artifact must be a door or gate. Doors can be closed or locked. The door will specify the room beyond, if it is locked, and its strength (see Door artifacts). Lastly, you must specify if the room is lit or dark. Room names may be in lower-case if desired.

ARTIFACTS

An artifact is any non-living thing that is in the dungeon. In addition to what you might normally think of as the artifacts, (gold, silver, statues), you must also have as an artifact all weapons carried by monsters, and (if enabled) a dead body for every monster (dead monsters are optional - see Dead Bodies below).

For each artifact, you will need a name (such as "GOLD COIN") and a full description. Next give the room that it is in, its value, its type, and its weight. The room is usually a positive number; however, if the item isn't in the dungeon yet (such as a dead body) you should assign a room of zero. See 'Location of Artifacts' below for more information.

Twelve types of artifacts are available, each with their own data fields. All 8 of the artifact fields are accessible for editing. The fields labeled USER are not used by the standard Eamon 7.x MAIN.PGM, but can be used by you for your own special programming. The available types of artifacts are:

```
 Type          Format
  0. Gold..........0
  1. Treasure......0
  2. Weapon........1
  3. Magic Weapon...1
  4. Container......2
  5. Lightable......3
  6. Drinkable......4
  7. Readable.......5
  8. Door/Gate......6
  9. Key...........7
 10. Bound Monster..8
 11. Wearable.......9
```

All of the formats have the same information in Fields 1-4: value, type, weight, and location. Fields 5-8 contain the following:

Format 1	Format 2	Format 3
5-Complexity	5-Key #	5-Counter
6-Weapon Type	6-Strength	6-(USER #6)
7-# of Dice	7-Open/Closed	7-(USER #7)
8-# Dice Sides	8-(USER #8)	8-(USER #8)

Format 4	Format 5	Format 6
5-Heal Amt	5-1st Effect	5-Room beyond
6-# of Uses	6-# of Effects	6-Key #
7-Open/Closed	7-Open/Closed	7-Strength
8-(USER #8)	8-(USER #8)	8-Hidden?

Format 0 & 7	Format 8	Format 9
5-(USER #5)	5-Monster#	5-Armor Class
6-(USER #6)	6-Key#	6-Type
7-(USER #7)	7-Guard#	7-(USER #7)
8-(USER #8)	8-(USER #8)	8-(USER #8)

Gold: has a set value, like gold pieces.

Treasure: has a value that varies, depending on the character's charisma

Weapon: spear, axe, club, bow, or sword. See the 'Monsters' section for more info on dice & sides.

Magic Weapon: is magic within the current adventure only. If taken back to the Main Hall, it reverts to an ordinary weapon. It cannot break except by magical means, which is the extent of its magic power. If you want other magic effects, you will have to program them. The v7.x MAIN.PGM classes any weapon brought by the player that has a maximum hit potential (#dice * #sides) of 25 or greater as magical.

Container: can contain other artifacts. It may be opened, at which time the artifacts are searched for any having a room number of 500 plus the number of the artifact being opened. Any artifact meeting this criteria is moved into the room by the program and the message, 'YOU FOUND SOMETHING', is printed. If it has a key #, then it is locked unless the player has the key specified in Field 5. A container can be smashed open if it absorbs enough hits to overcome its strength number. 'Closed' = 0 & 'Open' = 1.

Lightable: used to illuminate dark rooms, using the Light command. The counter decrements on each turn, and when it reaches zero, the artifact extinguishes and may not be relit using standard programming.

Drinkable: heals the user by the amount specified in Field 5. It may only be used the number of times set in Field 6 before it is exhausted. If Field 5 is set to a negative number then it is a poison. Field 7 is for sealed or closed bottles, jars, etc; be sure that streams, bowls, etc. are 'Open', or the player will have to OPEN them before drinking! (Closed = 0 & Open = 1). Potions, etc. may be given to friendly monsters, who will take a sip and then return it to the player.

Readable: can be read by the player. The program will automatically read the effects from the Effects portion of the EAMON.DESC file. These effects must be added by the designer, with Field 5 containing the # of the first Effect, and Field 6 the # of Effects to be read. This also has an 'Open?' field for books, scrolls, etc.; be sure that things like signs, etc. are open (closed = 0 & open = 1).

Door: the number of this artifact + 500 is put in the room data (see ROOMS). Field 5 tells the program where the door goes. Field 7 is how strong the door is; locked doors may be broken down by beating on them. If Field 6 is anything other than 0, then the door is locked, and the player must have the specified key that has the same artifact number as is in this field. There is a 'Hidden?' field that is used for secret doors; If a door is hidden then travel in that direction gets the message YOU CAN'T GO THAT WAY until the door is either Examined or Opened. Thus you must mention the secret panel somehow in the room desc (see 'synonym checking' in ADAPTING YOUR OWN MAIN.PGM).

Key: used for opening containers, doors or freeing bound monsters. The artifacts that need keys know the correct key, so no additional data is needed. Note that 'key artifacts' don't have to always be keys, but can also be things like magic amulets, crowbars, etc.

Bound Monster: used to simulate a bound monster. Field 5 is the actual monster to be freed. If Field 6 contains anything but 0, that key is needed. Field 7 specifies a Guard monster who should be in the same room. Freeing the Bound Monster causes the artifact to disappear from the room and the actual monster to take its place.

Wearable: can be worn by the player. 'Clothing Type' is not presently implemented by the MAIN.PGM, but the following protocol is recommended as a standard:

```
 Armor Class:
   0: clothing
   1: shield, helmet, gauntlets
   2: leather
   4: chain mail
   6: plate armor

 Clothing Type:
   0: clothing (incl. armor, shields)
   1: overclothes (incl. coats, capes, disguises)
   2: shoes, boots
   3: gloves
   4: headwear
```

Dead Bodies: If you want dead bodies, you must make a complete set of 'dead body' artifacts of artifact type 1. These 'dead bodies' do not need to have exactly the same artifact number as monster number, but they MUST be in the same order, and there MUST be as many bodies as monsters. To enable 'dead bodies', remove the REM from the front of line 7735 and change 'X' to the number of the FIRST dead body artifact.

Location of Artifacts

Artifacts normally have room numbers ranging from 1 to the number of rooms in a dungeon, but there are additional codes recognized by the program:

```
 #+500: Inside a container artifact
 #+200: Embedded in a room description
    -1: Carried by the player
  -999: Worn by the player
```

Embedded artifacts are those that are not listed in the room but can be acted upon by the adventurer. For example you could describe a room as containing a statue, and embed a statue artifact. The statue will not be listed in the room until it is examined or acted upon by the player. Thus you can hide things from the adventurer. Hidden artifacts can no longer be found using the Look command. You must embed them in the room description.

If you want to embed an artifact in room number 17 and not list it as a separate artifact, code its room code as 217. Remember that Embedded artifacts MUST be mentioned in the
description of the room or of another artifact that is in the same room or it can never be found by the player, who must Examine it to cause it to 'appear' in the room.

Containers: For example, if you want to place an artifact inside artifact #9, code its room code as 509. Note that artifact #9 MUST be a container or the artifact can never be found!

EAMON.EDIT.V7.1 will also allow you to build your own artifact types, or modify existing artifact types, formats, or even monster or room data. However, you will find that use of the USER fields will often allow you to do the special stuff that you want without going to the trouble of generating new artifact types.

Artifact names MUST be in upper-case, though the artifact descriptions may be in lower-case if desired.

EFFECTS

This is a description that can be used for your own special stuff. To read in and print your special Effect, use this code:

```
     R = (Effect #) + 400: GOSUB 45
```

Effects may be in lower-case if desired.

MONSTERS

Monsters make up all living (or animate) things in the dungeon. Monsters are similar to characters; however they have full armor expertise and know all weapons equally well. Monsters have been extensively changed from past versions of Eamon. It is now possible to have 'group' or multiple monsters, such as '4 RATS'. Actually, all monsters are now of the 'group' type, but single monsters are a 'group' of 1.

For each monster you will need a name, description, and this data:
  Hardiness: resistance to injury
  Agility: speed in battle. An agile monster will be a more successful fighter
  Number of Members: number of monsters in this group; single monsters are a 'group' of 1
  Courage: tendency to flee from a fight when injured or to pursue the player when he flees. Courage of less than 100 is the % chance that the monster will stay and fight. Courage of 200 will always pursue the player.
  Room: the number of the room in which it will be found. May be zero if activated by special programming
  Weight: This data is not presently used by Eamon; this field may be used by the author for special characteristics
  Armor: hits absorbed or stopped per blow. Equivalent to player armor. Armor class is deducted from agility during battle; the weight of its armor slows it down
  Weapon number: the number of the artifact that it is using for a weapon. A zero means that it has natural weapons such as claws. A negative number means that it is unarmed.
  # Dice: half of the number used to compute battle damage by the monster if it uses natural weapons.
  # Sides: the other half of the damage chances. For example, if # Dice = 1 and # Sides = 8, then there is a random chance that the monster will be able to land between 1 and 8 damage points (1 * 8). If the 2 numbers are, for example, 2 and 6, then the possible damage range is between 2 and 12.
  Friendliness: 1 = enemy; 2 = neutral; 3 = friend. A random chance can be assigned by adding the percent chance of friendliness to 100. For example, 140 indicates a 40% chance that the monster will be friendly.

The v7.x Eamon does not support offense & defense odds. The ability to hit is 50% + 2 times the difference in monster agility and armor between attacker and defender. For example, assume that the attacker has an agility of 20 and armor of 2, and the defender has agility of 15 and armor of 6. The more agile the monster, the greater his ability in battle, and the heavier his armor, the more his agility is compromised. In the above example, the attacker has an effective agility of (20 - 2) - 18 and the defender has an effective agility of (15 - 6) = 9, and the overall hit chance is 50 + 2(18 - 9) = 68%. If their roles are reversed, the hit chance is 50 + 2(9 - 18) = 32%.

Monster names MUST be in upper-case, though the monster descriptions may be in lower-case if desired.

ARRAY SEARCH SYNTAX AND NOTES

Note: This search routine is automatically installed and used by the MAIN.PGM, and you can skip the following section if you wish. If you need to do a search for your own special commands, simply GOSUB 4700 (for monsters), or 4804 (for artifacts), and the variable S$ will be searched for you. Take a look at the other commands to see how it is done. Or when all else fails, you can still do an ordinary array search using Applesoft.

The v7.x Eamon uses machine-code augmentation of the array searches. This subroutine is reached by way of the Ampersand option of Applesoft. The machine-code routine, SEARCH.ROUTINE, is fully relocatable. If the Ampersand is already in use, the Ampersand hooks set at line 33005 may be replaced by CALL statements. Feel free to contact the Eamon Adventurer's Guild for further assistance concerning this routine.

The array search routine can search a string array for a match to a search string, or it can search one dimension of an integer array for a match to a search value.

The string search syntax is:

```
   & S,S$,A$,C%
```

```
where: S = command to perform a string search
       S$ = the name of the search string
       A$ = the name of the array to be searched
       C% = the starting location in the array
              also returns with the value of a match

  1) The first 'S' is a command and may not be changed
  2) S$ in the above example may be any simple string name
  3) A$ in the above example may be any string array name.
     Note that the parentheses are NOT included.
  4) C% MUST be a simple integer variable.
     If C% returns unchanged then there was no match found
```

To reenter the routine to search for a second match, use the command

```
   & R
```

Here is a modified example from the MAIN.PGM:

```
 4705 WH = RO:HA = -1 : REM POSSIBLE LOCATIONS OF MONSTER
 4710 F = 0 : REM 'FOUND' FLAG
      C% = 0 : REM START AT ARRAY LOCATION 0
      D% = 0 : REM 'LAST MATCH LOCATION' VARIABLE
      & S,S$,M$,C% : REM SEARCH FOR A MATCH FOR S$ in M$() ARRAY
      IF NOT C% THEN RETURN : REM NO MATCH IF C% RETURNS WITH 0
 4720 D% = C% : REM SAVE LOCATION OF MATCH
      IF M%(C%,5) < > HA AND M%(C%,5) < > WH THEN 4770
 4750 F = F + 1: IF F = 1 THEN M = C% : REM FIRST MATCH
 4760 IF S$ = M$(C%) THEN M = C%: RETURN : REM EXACT MATCH
 4770 & R : REM SEARCH FOR ADDITIONAL MATCHES
      IF C% < > D% THEN 4720 : REM IF ANOTHER MATCH THEN 4720
 4780 IF F > 1 THEN PRINT "WHICH "S$" DO YOU WANT?" : F = 0
 4790 RETURN
```

The integer array search syntax is:

```
   & A,D%,A%(X,Y),C%,B%
```

where: A = command to perform an integer array search
       D% = search value
       A%(X,Y) = starting location in array
       C% = Starting location in array (same as 'X')
            returns with array location of match
            returns with zero if no match
       B% = the last location in the array

 1) The first 'A' is a command and may not be changed
 2) D% must be any simple integer variable
 3) A%(X,Y) may be any integer array. If it is a multiple
    array, only the first term ('X') may be searched
 4) C% must be any simple integer variable
 5) B% must be any simple integer variable and must equal the DIM
    size of the first term ('X').

Here is an example from the MAIN.PGM:

```
 140  B% = NM : REM SIZE OF MONSTER ARRAY
      D% = RO : REM SEARCH VALUE (ROOM # IN THIS EXAMPLE)
      FOR M = 1 TO NM : REM USE LOOP TO INCREMENT STARTING LOC.
      C% = M : REM STARTING LOCATION OF SEARCH (START AT 'M')
      & A,D%,M%(C%,5),C%,B% : REM PERFORM SEARCH
      IF NOT C% THEN M = 999: NEXT: GOTO 160 : REM NO MATCH
 145  M = C% : REM 'BUMP' LOOP UP TO LOCATION OF MATCH
      PRINT M$(M)"IS HERE."
 159  NEXT M : REM SEARCH REST OF ARRAY
```