GILSOFT
Home Computer Software

The Quill

Adventure Writing System

FOR THE COMMODORE 64

The Quill

An Adventure System for the Commodore 64 by Graeme Yeandle

Acknowledgment

Many thanks to Liz and Steve of Birchgrove Business Limited (Cardiff) for their invaluable assistance with the production of this manual.

Contents

What is Adventure?

Adventure can be described as a computerised version of the game Dungeons & Dragons. In Dungeons & Dragons one person is nominated as the dungeon master, and he invents a dungeon for other players to explore and try to retrieve hidden treasures which are usually protected by monsters of various shapes and sizes. Each player notifies his proposed actions to the dungeon master who decides on the outcome, sometimes with the help of dice to introduce a random element.

In Adventure the computer takes the place of the dungeon master and the player or players explore a predefined dungeon. Most Adventures will contain a vocabulary of words which the computer 'understands', a variety of locations which a player may wander around and objects which have to be used in the correct way to enable the Adventure to be solved. The computer will describe a situation to the player and invite him to decide on a course of action. The computer then tells the player the result of his action.

Everyone who plays Adventure has the problem of making the computer understand their commands. The computer will only have a limited vocabulary of perhaps a few hundred words and 'finding the right words' can sometimes be a problem, for example, if you are playing Dungeons & Dragons and the dungeon master tells you "There is a lamp nearby" then if you decide to "PICK UP THE LIGHT" the dungeon master should know what you mean. If the same situation occurs when playing Adventure the computer may understand "GET LAMP" but may not know that LIGHT means, on this occasion, the same as LAMP or that PICK UP means the same as GET. Even so, most players will very quickly get the knack of 'finding the correct words'. However, it should be noted that it is up to the Adventure designer to decide which words are included in the computer's vocabulary.

What is The Quill?

The Quill is an adventure writer which allows someone with no programming experience to create a machine code adventure.

Getting Started

The Quill Adventure system is made up of three parts:-

a) A database which will contain all the information relevant to an adventure.

b) A database Editor which enables data to be inserted into, amended in or deleted from the database.

c) An Interpreter (this is the dungeon master) which uses the data in the database to execute your adventure.

To load The Quill from disc, use LOAD"Q",8,1.

To load The Quill from tape, use SHIFT/RUN STOP

Part 1 of this manual will introduce you gently to The Quill, from simple location descriptions through to complex condition tests and actions. It is strongly recommended that you work all the way through Part 1 before attempting to write your own adventures. Part 2 contains a detailed description of The Quill for reference.

Part 1

The Main Menu

When The Quill has loaded you will be presented with the Editor's Main Menu which gives you a number of options. Some of these options e.g. Bytes Spare, will perform a function and return to the Main Menu while others e.g. Location Text, will give you a sub menu. The RETURN TO BASIC option is an exception to this as it executes the BASIC NEW command which destroys The Quill.

The Input Routine

The Quill uses the Commodore 64's business mode character set and has its own screen editor which is not limited to two screen lines. When you type in on the keyboard the characters that you key are placed into a large input buffer and the contents of the input buffer are then printed at the bottom of the screen. Colour controls and reverse on/off are entered in the usual way and take effect immediately, e.g. CTRL 9 causes subsequent characters to be printed in inverse. CURSOR LEFT and CURSOR RIGHT are used to move the cursor through the input buffer but note that when the cursor is moved over a colour control code the position of the cursor on the screen will not change. When entering text, SHIFT/RETURN may be used to move the cursor, and everything following it, onto a new line. CURSOR UP and CURSOR DOWN are used to move the cursor within the input buffer and will move it up to 40 characters or to the end of the line whichever comes first.

CLR HOME will move the cursor to the beginning of the input buffer while SHIFT/CLR HOME will clear the input buffer. Try typing in a few characters and then pressing SHIFT/CLR HOME, then type in a few more characters and press RUN STOP. While in the Editor RUN STOP will always return you to a menu but never ever press RUN STOP/RESTORE as this will corrupt the Quill.

To insert characters simply position the cursor at the appropriate place and type in the characters to be inserted. It is not possible to type over characters that are already present. If you type in more than 23 lines any subsequent lines will still be present in the input buffer but will not be displayed on the screen therefore this is not recommended. Error messages appear at the bottom of the screen and pressing any key will return you to a menu.

Whenever you press RETURN the Editor checks that what you have typed in is valid i.e. it checks the syntax. If the Editor finds a syntax error it is indicated, by a ? following the error. The Editor always positions the cursor immediately after the ? so that the cursor is in the vicinity of the error. On the Main Menu the only valid options are single capital letters in the range A to R and SHIFT/+. Try typing in 3 and pressing RETURN, then delete the 3 and type in ABC followed by RETURN. In each case a syntax error will be detected.

Bytes Spare

Let's now try one of the options on the Main Menu. If you clear the input buffer (SHIFT/CLR HOME), then type O for Bytes Spare and press RETURN you will get a display which tells you how many bytes are unused in the database.

Permanent Colours

When The Quill is loaded it has a blue background (i.e. Border and Paper) and White Ink. The values of INK, PAPER & BORDER can be changed by selecting option Q on the Main Menu. Note that RUN STOP can be used at any time to return to the menu. Try changing the colours but be careful not to select the same colour for INK as for PAPER.

Save, Verify & Load Database

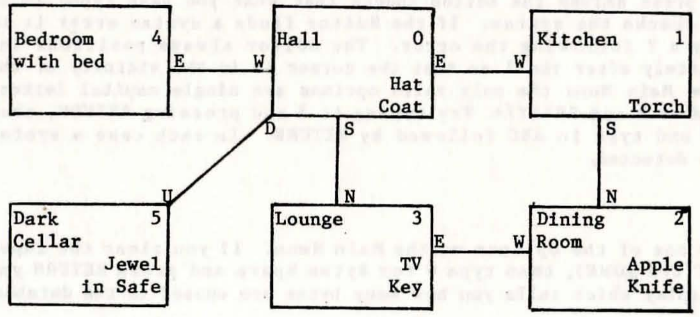
These three options on the Main Menu allow the database to be saved or reloaded and in each case you will be prompted to "Type in name of file". When loading or verifying, the Commodore will search for a file with the name specified and then load or verify it. A failure to verify will be reported as I/O ERROR. Disc users should note that filenames may not start with @: and that the disc error channel is printed to the screen after every disc operation. The RUN STOP key may be used to interrupt a save, verify or load but if it is used to interrupt a load, or an I/O ERROR is detected during a load, then the database will be corrupt. Be very careful with a corrupt database as it can easily cause the Editor & Interpreter to become corrupt. In fact the only safe Editor command with a corrupt database is Load Database and this should be used until a database is loaded successfully.

Setting up an Adventure

The following sections of Part 1 of this manual will give you some practical experience of using The Quill to set up an adventure. Each section depends on entries having been made to the database in earlier sections so you have to work through it step by step. If you wish to break off part of the way through, please save the database so that you can continue later where you left off.

A map of the mini adventure we are going to set up is shown in figure 1 and the objective of the adventure is to find the jewel and place it in the Dining Room. The map shows all of the locations in the adventure and how they are interconnected. The locations have all been given a location number, which is shown in the corner, and the position of various objects is indicated.

Figure 1 - Map of the Adventure



Before we start to set up the adventure make sure that the database is in the state it was when The Quill was loaded. The PAPER and BORDER colours should be set to 6 and INK should be set to 1.

The Location Texts

The descriptions we will use for each location are as follows:-

Location 0

I am in a Hall. The Kitchen is to the East, the Bedroom to the West and the Lounge to the South. Steps lead Down to the Cellar.

Location 1

I am in the Kitchen. The Hall is to the West and the Dining room is to the South.

Location 2

I am in the Dining Room. The Kitchen is to the North and the Lounge to the West.

Location 3

I am in the Lounge. To the North is the Hall while the Dining Room is to the East.

Location 4

I am in the Bedroom. The Hall is to the East and a bed is against the North wall.

Location 5

I am in the Cellar. Steps lead Up to the Hall.

If you type in C on the Main Menu the Location Text menu will be displayed and you will see that location texts can be Inserted, Amended or Printed. P is used to print on the screen while L is used to print on the printer. P or L by itself will start printing with the text for location 0. P and L can also be followed by a location number (locno.) and if that location exists printing will start with the text for that location. The P or L and the locno. must be separated by at least one space and if the locno. does not exist a syntax error will be detected.

Type in P followed by RETURN on the Location Text menu and you will see that a description is already present for location 0. The only reason for this is that the programming of The Quill was much simpler if location 0 was always present. The text that is present is not important but it does show some of the things that can be entered from the keyboard e.g. Reverse.

As a text is already present for location 0 we will have to amend it to the text required for our mini adventure. You will see from the Location Text menu that to amend a location text A followed by a locno. must be entered. Please now enter A 0 and the existing text for that location will be copied to the input buffer and displayed at the bottom of the screen followed by a cursor.

Use SHIFT/CLR HOME to clear the input buffer and then type in the text we need for our location 0 i.e. the Hall. Use SHIFT/RETURN where appropriate and, if you like, put the compass directions in INVERSE. When you have typed it in press RETURN to amend the database. Note that the database is not amended until you press RETURN and that the use of RUN STOP at any time before pressing RETURN will leave the database unchanged.

The texts for the other locations in our mini adventure now have to be inserted by using the I option on the Location Text menu. Notice that the I is not followed by a locno. When you use I the Editor automatically allocates the next location number in sequence. Type in I followed by RETURN and the Editor will tell you which location number has been allocated (1 in this case) and display a cursor at the bottom of the screen. It is important that you realise that a null entry for the location number being inserted has already been made in the database and that if you were to use RUN STOP at this stage (do not do this now) the null entry would still remain.

Type in the text that we need for our location 1 and press RETURN which will change the null entry already inserted to the text you have typed in. You should now be able to insert the texts for locations 2 to 5 and then print them to check that they are correct. If you have made any mistakes then please amend the texts to correct the errors. You might also like to try P followed by a locno. e.g. P 2, to start printing with the text for location 2.

You should have inserted the location texts for locations 0 to 5 and returned to the Main Menu before continuing with the next part.

The Movements

The interconnections between our 6 locations can now be entered into the database and these are placed in the Movement Table. If you select D on the Main Menu the Movement Table menu will be displayed and you will see that entries can be amended or printed. Notice that entries cannot be inserted. This is because when you insert a location text for a location the Editor automatically creates a null entry for that location in the Movement Table. If you type in P on the Movement Table menu you should see that null entries do actually exist for our locations 0 to 5.

Refer back to the map of our mini adventure in figure 1 and you will see for location 0 that:-

```
EAST  goes to location 1
DOWN  goes to location 5
WEST  goes to location 4
& SOUTH goes to location 3.
```

Going back to the Movement Table menu type in A 0 to amend the entry for location 0. When 'Movements from location 0' is printed at the top of the screen, type in (exactly)

```
EST 1 DOWN 5 WEST 9 SOUTH 3  and press RETURN.
```

If you did type it in exactly you should get a syntax error after EST because it is not in the database's vocabulary. If you correct EST to EAST and press RETURN again you should get a syntax error after the 9 because the Editor knows that there is no location 9. Change the 9 to 4 and you should then have EAST 1 DOWN 5 WEST 4 SOUTH 3.

Pressing RETURN now will amend the existing database entry.

Now type in P on the Movement Table menu and you will see that the entry for location 0 reads:-

```
E   TO   1
D   TO   5
W   TO   4
S   TO   3
```

The Editor knows that E is a synonym of EAST (i.e. E means the same as EAST), D is a synonym of DOWN etc. and it always prefers to use the abbreviation or shorter synonym. We will deal with synonyms in more detail when we get to the section on the vocabulary.

If you now wanted to amend the entry for location 0 it would be displayed at the bottom of the screen as "E 1 D 5 W 4 S 3". Type in A 0 on the Movement Menu to have a look; you can then get back to the menu by using RUN STOP or RETURN. If you use RETURN the Editor will copy the input buffer into the database and tell you it has amended the entry even though it hasn't actually changed. Note that you could have typed in the abbreviations initially.

The Movement Table entries we need for our mini adventure are:-

Location 0	E 1 D 5 W 4 S 3
Location 1	S 2 W 0
Location 2	N 1 W 3
Location 3	N 0 E 2
Location 4	E 0
Location 5	U 0

Check these with the map and then amend the Movement Table entries for locations 1 to 5 using the Movement Table menu. You can print the entries to check that they are correct if you wish.

Testing the Adventure

Now that you have entered the location texts and the movements it is time to test the adventure, so select L on the Main Menu. You will be asked whether you require diagnostics and you should reply N & RETURN or just RETURN. Note that the adventure always begins at location 0. You should be able to move to all of our locations using the full words e.g. EAST, or the abbreviations e.g. E. When you are in the bedroom try typing in the following commands:-

	GO WEST	will get the reply I can't go in that direction.
	GET JEWEL	will give I can't.
	LIE ON THE BED	
&	GO TO THE HALL	will give I don't understand...
	REDESCRIBE LOCATION	
or	REDESCRIBE	
or	R	will print the location description again.
	TAKE INVENTORY	
or	INVENTORY	
or	I	will give a list of what you're carrying (nothing).

To return to the Editor use the word QUIT and if you have found any errors in the location texts or the movements then use the Editor to correct them. If you would like to try out the diagnostics then select L again on the Main Menu and reply Y to the prompt. The diagnostics will not help you much at this stage but you should note that the last number in INVERSE is the number of the current location. The first 33 numbers (all 0 at the moment) are the values of, what are called, the user flags and they will be explained in later section.

The Objects

An object is anything that can be manipulated e.g. a key, or moved from place to place or changed from one thing into another e.g. a torch into a lit torch. Most of the objects in our mini adventure are shown on the map in figure 1 but a full list showing the object number, the description needed and the position of the object at the start of the adventure is as follows:-

objno.	Text	Start location
Object 0	A lit torch.	not created
Object 1	A torch.	1
Object 2	An apple.	2
Object 3	A sharp knife.	2
Object 4	A television.	3
Object 5	A coat.	0
Object 6	A deerstalker hat.	0
Object 7	A key.	3
Object 8	A safe.	5
Object 9	A jewel.	not created
Object 10	An open safe.	not created
Object 11	A walking stick.	carried

So, the (unlit) torch starts out at the kitchen, the walking stick starts off being carried and the jewel (which can't be seen because it's in the safe) starts off as not created. Notice that there are two descriptions for the safe and that it is treated as two separate objects. When the safe is opened Object 8 will be destroyed and Object 10 will be created, while the reverse will happen if the safe is closed. Similarly the torch is really two objects which will be swapped over when the torch is switched on or off.

The Object Texts

The descriptions of the objects are entered in exactly the same way as the descriptions of the locations. If you select E on the Main Menu the Object Text menu will be displayed and you will see that it is the same as the Location Text menu except that locno. has been replaced with objno. i.e. object number. If you print the object texts (P) you may not be surprised to find that an Object 0 already exists. Amend the text of Object 0 so that it reads "A lit torch." and use a different colour. Then insert the texts for objects 1-11 so that all the object texts use the same colour. After you have printed the texts, checked and corrected them please return to the Main Menu.

The Object Start Locations

Now that the object texts have been inserted into the database we can go about placing the objects where they will be at the start of the adventure. Type in F on the Main Menu to select the Object Start Location menu and you will see that the entries can be amended or printed. Entries cannot be inserted because when you insert an object text for an object the Editor automatically inserts an entry of 'not created' for that object in the Object Start Location Table. If you print the Object Start Location entries you should see that our 12 objects (0-11) are all 'not created'. To amend the entry for object 7 so that it starts the adventure at location 3 type in A 7 3 on the Object Start Location menu. 252, 253 & 254 are some special locnos which mean not created, worn & carried respectively so to amend the entry for object 11 so that it starts off being carried type in A 11 254.

Refer back to our list of objects and their start locations and amend all the other entries that need to be changed. You can then print the entries again to check that they are correct. Note that if you type A 0 252 (for instance) the Editor will print 'Amended' even though the entry hasn't changed.

Testing Again

It's time to test the adventure again to check that the objects are where they should be and have the correct descriptions so select L on the Main Menu. When you are in the adventure type in INVENTORY or I to check that you are carrying the walking stick and use the map in figure 1 to check that the other objects are at the correct locations.

If you asked for diagnostics you will see that the second number is now set to 1. The Interpreter which is part of The Quill contains 33, of what are termed, user flags and these will be explained in more detail in a later section. However, as a very brief introduction, these flags are numbered from 0 to 32 and may contain values in the range 0-255. When you ask for diagnostics the first 33 numbers printed at the bottom of the screen are the values of these flags. Flag 1 i.e. the second flag, is used to contain a count of the number of objects carried and as one object is carried in this case, Flag 1 has the value 1.

The Vocabulary

One section of the database contains the vocabulary and this will hold an entry for every word that the computer is to understand. The Vocabulary menu is selected by typing in A on the Main Menu and it allows for words to be inserted and deleted, for the vocabulary to be printed and for the synonyms of a word to be displayed. Synonyms were mentioned earlier when we dicovered that the Editor knew that EAST and E meant the same thing. If you print the vocabulary you will see that there are over 30 entries already present and these relate to words which will be needed in most adventures. Each entry consists of up to four letters followed by a number (or word value) and entries with the same word value are synonyms.

The entries in the vocabulary will either hold a whole word e.g. UP, or, if the word has more than four letters, just the first four letters e.g. ASCE(ND). This has the advantage of using up only a little memory and it also reduces the amount of typing the person playing the adventure has to do because he or she will soon learn that only the first four letters of each word are significant. The disadvantage, of course, is that you can't have two words with different meanings which start with the same four letters. This rarely causes problems but note that when you are playing an adventure and want to go NORTHEAST you have to type in NE as the vocabulary says that NORTH is a synonym of N.

Take a look through the vocabulary and you should be able to spot all the words we used when we were doing the Movement Table and in fact the Movement Table can only contain words that are in the vocabulary. Getting back to the Vocabulary menu, type in S ASCEND to see the synonyms of the word ascend (Note that S ASCE is sufficient). Then try inserting the word ORANGE with the value 200 i.e. I ORANGE 200 or I ORAN 200. We haven't an orange in our adventure so delete the entry using D ORAN. Try inserting a

word that is already present e.g. I STOP 62 and deleting a word that isn't present e.g. D COMMODORE. We'll come back to the vocabulary after we've found out what the Interpreter does when a command is typed in.

Decoding the Players command

Each time the player types in a command during an adventure the Interpreter has to decode it. It does this by searching along the command for words which are in it's vocabulary. The word value of the first word recognised is stored in a variable called W1 and similarly the word value of the second word recognised in W2. This means that commands like TURN ON THE TORCH can be reduced to ON TORCH provided the words TURN & THE are not in the vocabulary and that GO TO THE EAST will mean the same as EAST if the words GO, TO & THE are not in the voabulary. Thus it is important to consider which words are excluded from the vocabulary as well as those which are included.

If no words are recognised the Interpreter gives the reply "I don't understand...". If the Interpreter recognises a word or words, but they neither cause movement (due to no entry in the Movement Table) nor cause an action to be performed (to be explained later) then the Interpreter gives the reply "I can't", if the value of W1 is greater than 12 or "I can't go in that direction", if the value of W1 is less than 13. Therefore the words in the vocabulary which relate to directions should have word values in the range 1 to 12.

More Movements

The location descriptions in our adventure include statements of the form "The Hall is to the West" and up to now we have moved to the Hall with the commands WEST, W or GO WEST. We are now going to improve the adventure so that it will also obey commands of the form GO TO THE HALL or just HALL. To do this we will need an entry in the vocabulary relating to each location so insert the following entries into the vocabulary.

HALL	13
KITCHEN	14
DINING	15
LOUNGE	16
BEDROOM	17
CELLAR	18

e.g. to insert the word KITCHEN with a word value of 14 use I KITC 14 on the Vocabulary menu. Print the vocabulary to check the entries are correct and then amend the entries in the Movement Table to include these new words. The Movement Table entries required are:-

```
Location 0 E 1 D 5 W 4 S 3 KITC 1 CELL 5 BEDR 4 LOUN 3
Location 1 2 W 0 DINI 2 HALL 0
Location 2 N 1 W 3 KITC 1 LOUN 3
Location 3 N 0 E 2 HALL 0 DINI 2
Location 4 E 0 HALL 0
Location 5 U 0 HALL 0
```

When you have changed the Movement Table, test the adventure again to check that commands such as GO TO HALL are obeyed correctly. While testing the adventure note that when you are in the Dining Room, GO NE gives the reply "I can't go in that direction" because NE has a word value less than 13, while GO TO THE HALL gives the reply "I can't" because HALL has a word value greater than 12.

More Words

We will shortly get to the stage where we can begin to manipulate the objects in our adventure. Before we can do that though, the words we will use to manipulate the objects have to be inserted into the vocabulary. Please insert the following entries into the vocabulary:-

TORCH	20
APPLE	21
KNIFE	22
TELEVISION	23
TV	23
COAT	24
DEERSTALKER	25
HAT	25
KEY	26
SAFE	27
JEWEL	28
STICK	29
UNLOCK	30
OPEN	30
CLOSE	31
SHUT	31
LOCK	31
LIGHT	32
ON	32
OFF	33
OUT	33
EAT	34
BED	200
HUNGER	201
FINISH	202

The bed is not an object in the adventure and we do not intend to make use of the word BED anywhere else in the database. However, the bed is mentioned in the location description of the bedroom so BED is included in the vocabulary to stop the Interpreter replying "I don't understand..." if a player tries to use the bed.

The Event Table

This table (and the Status Table) form the heart of the database, for it is here that the actions the Interpreter has to take to reply to a player's command are specified. Each entry in the table consists of two word values, a set of conditions and a set of actions. When the adventure is played the Interpreter matches the word values entered by the player (which have been stored in W1 and W2) against each entry in the table. If the word values match and the conditions are satisfied then the actions are

performed.

Select G on the Main Menu to display the Event Table menu and you will see that entries can be inserted, amended or printed. Printing is slightly different to the other menus because you may either enter P by itself, P followed by one word or P followed by two words and printing will start at the appropriate part of the table. The words used must of course be present in the vocabulary. If you type in P now you will find that a few entries, which will be needed for most adventures, are already present.

Action INVEN

This is the action which prints "I have with me:-" etc. The first entry in the Event Table, GET I, has no conditions and a single action called INVEN. What does, this mean? Well if you were playing the adventure and typed in TAKE INVENTORY then, because the word value stored in W1 matched the word value for GET and W2 matched I, action INVEN would be performed. Remember that TAKE & GET and INVENTORY & I are synonyms.

The next entry I * does the same thing. If the first word recognised by the Interpreter is a synonym of I then action INVEN is performed. The * means that the corresponding value in W1 or W2 (W2 in this case) is irrelevant. Please make sure you fully understand the last two sections on the Event Table before reading further.

Action DESC

The third entry in the Event Table R *, means that if R or REDE(SCRIBE) is entered as a command then action DESC will be performed. Action DESC clears the screen and attempts to describe the current location. (If it is dark the Interpreter will print "Everything is dark. I can't see.").

Actions SAVE and LOAD

These are the actions which copy/restore a game position. The last two entries in the Event Table use these actions and mean, for instance, that if LOAD is typed in as a command then action LOAD is performed. Do not be confused by the vocabulary word LOAD and the action word LOAD because they are not related. You can only have an entry in the Event Table of LOAD * if there is an entry for LOAD in the vocabulary. The action word LOAD is independent of the vocabulary.

Actions QUIT, TURNS and END

Action QUIT is the action which asks "Are you sure you want to quit now?" while action TURNS is the action which prints "You have taken x turn(s).". Action END is a very important action as it prints "END OF GAME Do you want to try again?" and if you reply "N" it returns you to the Editor. Note that the only way to return to the Editor, after testing an adventure, is by action END being performed.

Actions GET, DROP and OK

Actions GET and DROP must be followed by an objno. and are used to carry or put down objects while action OK simply prints "OK". In our mini adventure we want the player to be able to take and leave the walking stick using the commands TAKE STICK and DROP STICK respectively. The entries needed in the Event Table for this are:-

TAKE STICK	Conds		
	Acts	GET	11
		OK	
DROP STICK	Conds		
	Acts	DROP	11
		OK	

To insert the first of these into the Event Table go to the Event Table menu, type in I TAKE STICK and press RETURN. This inserts a null entry for TAKE STICK (i.e. with no conditions and no actions) in the table and displays a cursor at the bottom of the screen to allow you to amend the null entry. When the cursor appears, type in GET 11 OK and press RETURN again. Insert the entry for DROP STICK in the same way and then print the Event Table to check the entries. Notice that the Editor prefers to use the vocabulary word GET instead of TAKE as it is a shorter synonym.

Now test the adventure to see the effect of these entries in the Event Table. In particular try the following commands:-

TAKE STICK	when you are already carrying it
DROP STICK	when you are not carrying it
GET STICK	when the stick is at a different location

Condition PRESENT

The entries we need, to be able to GET and DROP the other objects are:-

Words	Conds	Acts
GET APPLE		GET 2 OK
DROP APPLE		DROP 2 OK
GET KNIFE		GET 3 OK
DROP KNIFE		DROP 3 OK
GET COAT		GET 5 OK
DROP COAT		DROP 5 OK
GET HAT		GET 6 OK
DROP HAT		DROP 6 OK
GET KEY		GET 7 OK
DROP KEY		DROP 7 OK
GET JEWEL		GET 9 OK
DROP JEWEL		DROP 9 OK
GET TORCH	PRESENT 0	GET 0 OK
GET TORCH		GET 1 OK
DROP TORCH	PRESENT 0	DROP 0 OK
DROP TORCH		DROP 1 OK

As you can see the entries for the torch are not quite so simple. The problem is that the torch is really two objects so we have to have two entries. The condition PRESENT, which must be followed by an objno., checks whether the object specified is present at the current location. So the first entry for GET TORCH checks whether Object 0 is present and if it is, tries to GET it. If Object 0 is not present the condition is not satisfied so the Interpreter falls through to the next entry, which has no conditions, and tries to get Object 1. When an entry has conditions and actions the conditions are typed in in front of the actions e.g. PRESENT 0 GET 0 OK.

When you have inserted the entries print them to check they are correct and that the entries for the torch are in the correct order. Amending entries in the Event Table is similar to amending entries in the other database tables. However, if there is more than one entry present for the same words e.g. DROP TORCH, then each entry is displayed in turn for amending and you simply press RETURN to leave an entry as it is. Try A DROP TORC on the Event Table menu and keep pressing RETURN until you get back to the menu; both entries should be displayed in turn for possible amending.

Entries in the Event Table can be deleted by removing all the conditions and actions i.e. Amend the entry using SHIFT/CLR HOME to clear the input buffer and then press RETURN. Before we test the adventure again we'll insert a few more entries.

Actions WEAR and REMOVE

These actions enable objects to be worn and removed and must be followed by an objno. Our adventure has two objects that can be worn and the entries needed in the Event Table for this are:-

Words	Conds	Acts
WEAR HAT		WEAR 6 OK
REMOVE HAT		REMOVE 6 OK
WEAR COAT		WEAR 5 OK
REMOVE COAT		REMOVE 5 OK

Insert these entries and then test the adventure again using diagnostics. The reason for using diagnostics is to enable you to monitor the value of Flag 1 (the second flag) which has a value equal to the number of objects carried. When you are in the adventure try these commands in the order shown:-

Command	Response	Flag 1 value
WEAR HAT	I don't have it	1
REMOVE HAT	I'm not wearing it	1
GET HAT	OK	2
TAKE COAT	OK	3
KITCHEN		3
GET TORCH	OK	4
DINING		4

continued next page

Command	Response	Flag 1 value
GET APPLE	I can't carry any more	4
WEAR HAT	OK	3
WEAR HAT	I'm already wearing it	3
GET APPLE	OK	4
REMOVE HAT	I can't. My hands are full	4
DROP HAT	I can't. My hands are full	4
WEAR COAT	OK	3
I	I have with me...	3
DROP HAT	OK	3
REMOVE COAT	OK	4

Also make sure you can GET the key and drop all the objects you are carrying.

As you can see, the actions GET, DROP, REMOVE and WEAR include quite a bit of checking on the objects being manipulated and they can normally be used without being preceded by conditions. These actions are exceptions to the rule as most other actions will need to be preceded by conditions.

Action SWAP

This action is followed by two objnos and simply swaps over the positions of the two objects e.g. if Object 0 is 'not created' and Object 1 is at location 3 then the action SWAP 0 1 would put Object 0 at location 3 and make Object 1 'not created'. In our adventure we are going to use SWAP to switch the torch on and off. The entries we need are:-

Words	Conds	Acts
ON TORCH	PRESENT 1	SWAP 0 1 OK
TORCH ON	PRESENT 1	SWAP 0 1 OK
OFF TORCH	PRESENT 0	SWAP 0 1 OK
TORCH OFF	PRESENT 0	SWAP 0 1 OK

Notice that we have catered for a player giving the command SWITCH OFF TORCH or SWITCH TORCH OFF and that the condition PRESENT 0 means that the torch can only be switched off if the lit torch is present. Insert these entries into the Event table and test them.

The Flags

The Interpreter contains 33 user flags which are really just variables that can hold a value in the range 0 - 255. Like everything else in The Quill, numbering starts at 0 so the flag numbers (flagnos) are in the range 0-32. There are actions which enable the flag values to be changed and conditions which allow the flag values to be tested. Some of the flags have special purposes while others have an auto-decrement feature. The flags with special purposes are flags 0, 1, 30, 31 and 32. Flag 0 is used to tell the Interpreter whether it is light or dark; if it has any value other than zero the Interpreter thinks it's dark. Flag 1 as we have seen before, holds a count of the number of objects carried. Flag 30 is used to hold the players score as a percentage while Flags 31 and 32 are used to hold

the number of turns that the player has taken. Flags 2 to 10 are auto-decrement flags which means they are decreased by 1 under certain circumstances:-

Flag 2 is decreased each time the Interpreter tries to describe a location

Flag 3 is decreased each time the Interpreter tries to describe a location and it's dark

Flag 4 is decreased each time the Interpreter tries to describe a location and it's dark and Object 0 is absent

Flags 5-8 are decreased each time a command is entered

Flag 9 is decreased each time a command is entered and it's dark

Flag 10 is decreased each time a command is entered and it's dark and Object 0 is absent

Flags 11-29 have no special features

Note that a flag cannot be decreased below zero and that Object 0 is a special object because the Interpreter considers it as a source of light.

There is a summary of the flags and a summary of the conditions and actions that can be used, on the last page, of this manual. As mentioned earlier, when diagnostics are requested the values of the flags are printed so that you can monitor them.

Light and Dark

If you take another look at our map in figure 1 you will notice that it says the cellar is dark. To make the cellar dark we will use the condition AT and the actions CLEAR, SET, GOTO and DESC. Condition AT must be followed by a locno. and is satisfied if the player is at that location. The actions CLEAR and SET must be followed by a flagno. and change that flags value to 0 or 255 respectively. Action GOTO is followed by a locno. and causes movement to that location.

If this entry were present in the Event Table (do not insert it).

Words	Cond	Acts
U *	AT 5	GOTO 0 DESC

it would have exactly the same effect as our entry U 0 for location 5 in the Movement Table i.e. if a player gives the command UP when he is at location 5, he is moved to location 0 and the location is described.

The Event Table entries to be inserted to make the cellar dark are:-

Words	Conds	Acts
D *	AT 0	SET 0 GOTO 5 DESC
CELLAR *	AT 0	SET 0 GOTO 5 DESC
U *	AT 5	CLEAR 0 GOTO 0 DESC
HALL *	AT 5	CLEAR 0 GOTO 0 DESC

When a player moves from the Hall to the Cellar, Flag 0 will be set to 255 (making it dark). When he moves back to the Hall, Flag 0 will be cleared to 0 (making it light again). However, if you were to test the adventure now you would find that the Cellar was still light. This is because the Interpreter checks the Movement Table before it checks the Event Table and if the Movement Table causes movement the Event Table is not checked. So remove the movements between the Hall and Cellar from the Movement Table and then test the adventure. When you are testing the adventure monitor the value of flag 0 and try experimenting with moving between the Hall and Cellar and switching the torch on and off.

Action SCORE

The action SCORE prints "You have scored x%" where x is the value of Flag 30. We are going to use a very simple scoring system in our adventure where the player scores 50% for opening the safe and a further 50% for completing the adventure. If the player QUIT's the adventure he will want to know what score he has achieved so amend the Event Table entry for QUIT * so that it has the actions QUIT SCORE TURNS END. It is important that you print this entry (use P QUIT) to check it is correct, because the only way back to the Editor from the Interpreter is by action END being performed.

Opening and Closing the Safe

In our adventure the safe can only be opened if the key is carried, the lit torch is present and, of course, the safe is not already open. To close the safe, the key is not needed but the lit torch must be present. The first time the safe is opened we want to create the jewel and give the player a score of 50% but if the player closes the safe and reopens it then we must not create the jewel again or give another 50%.

We will use Flag 11 to show whether the safe has already been opened. As the flags start off with a value of zero we will say that if Flag 11 has the value 0 the safe has not previously been opened. Then if we SET Flag 11 the first time the safe is opened we will be able to tell that the safe has already been opened. The Event Table entries we need are:-

```
OPEN SAFE        Conds       PRESENT    8
                             CARRIED    7
                             PRESENT    0
                             ZERO       11
                 Acts        DESTROY    8
                             CREATE     10
                             LET        30   50
                             CREATE     9
                             SET        11
                             DESC
```

```
OPEN SAFE        Conds       PRESENT    8
                             CARRIED    7
                             PRESENT    0
                 Acts        DESTROY    8
                             CREATE     10
                             OK

CLOSE SAFE       Conds       PRESENT    10
                             PRESENT    0
                 Acts        DESTROY    10
                             CREATE     8
                             OK
```

Condition CARRIED must be followed by an objno. and is satisfied if that object is carried while condition ZERO must be followed by a flagno. and is satisfied if that flag has the value 0. Action CREATE is used to change an object's location to the current location while action DESTROY changes an object's location to 'not created'. Both CREATE and DESTROY must be followed by an objno. Action LET is followed by a flagno. & a value, and the flag is given that value e.g. LET 30 50 gives Flag 30 a value of 50; a bit like the BASIC command LET FLAG30 = 50.

The order of the two entries for OPEN SAFE is important. The first time the safe is opened the condition ZERO 11 will be satisfied so the actions in the first entry will be performed. If the safe is opened again the condition ZERO 11 will not be satisfied and the Interpreter will fall through to the second OPEN SAFE entry. Insert these entries into the Event Table, then print them (use P UNLOCK SAFE) to check they are correct. Note that the first OPEN SAFE entry ends with action DESC so that the player can see that the jewel has been created. When you have checked the entries test the adventure again and use diagnostics so that you can monitor the values of flags 11 & 30.

The Message Texts

Any messages which are needed in the adventure have to be entered into the Message Text table. The Message Text menu is the same as the Location Text menu with the exeption that locno. has been replaced with mesno. (message number). If you select B on the Main Menu to display the Message Text menu and then print the messages, you will see that a message 0 is already present. The messages we need in our adventure are:-

Message 0

I'm hungry!

Message 1

Ah. That's better!

Message 2

I'm dying of starvation...

Message 3

Well done. You've solved the Adventure.

Amend the existing entry for message 0 and then insert messages 1 to 3.

The Status Table

This table has exactly the same format as the Event Table i.e. each entry has two word values followed by conditions and actions, but the Interpreter uses the two tables in slightly different ways. We have already seen that the Interpreter uses the Event Table after each command given by a player and matches the word values in W1 and W2 with each entry in the table. The Interpreter uses the Status Table in between turns and looks at each entry irrespective of the word values in W1 and W2. Thus the Event Table contains entries which are dependent on the commands entered by the player while the Status Table contains entries which are independent of the commands entered by the player. Note that the Interpreter does not make use of the word values present in the Status Table entries. The vocabulary words used in the Status Table can therefore be used as comments, to remind you what the entries do, or to position an entry at a particular place in the table because the entries are arranged in ascending order of word value. When you write your own adventures you will find that the order of the entries in the Status Table is very important.

Finishing the Adventure

Our mini adventure is solved when a player is at the Dining Room and the jewel is present. It doesn't matter what commands the player uses to get the jewel to the DINING ROOM so we will use an entry in the Status Table to detect when the adventure is solved. The Status Table entry required is:-

FINISH	*	Conds	AT	2	
			PRESENT	9	
		Acts	MESSAGE	3	
			PLUS	30	50
			SID	24	15
			SID	6	240
			SID	1	22
			SID	0	96
			SID	4	33
			PAUSE	20	
			SID	1	29
			SID	0	223
			PAUSE	20	
			SID	1	39
			PAUSE	20	
			SID	1	53
			SID	0	57
			PAUSE	90	
			SID	24	0
			SCORE		
			TURNS		
			END		

Note that the vocabulary word FINISH is used only as a comment.

Select H on the Main Menu to display the Status Table menu and then insert this entry. Action MESSAGE must be followed by a mesno. and simply prints that message. Action PLUS must be followed by a flagno. & a value, and adds that value to the appropriate flag e.g. PLUS 30 50 adds 50 to Flag 30 (the score). Action SID is followed by a register number (regno.) and a value and controls the sound synthesiser e.g. SID 24 15 sets the volume to maximum. The PAUSE action is used to introduce a delay measured in 1/50 second.

When you have inserted this entry in the Status Table test the adventure again and take the jewel to the Dining Room.

Hunger

In our adventure we want the player to become hungry after 6 turns, to remain hungry for the next 7 turns and then to die of starvation. Eating the apple, of course, will ward off the pangs of hunger. We will set Flag 12 when the apple is eaten and we will use Flag 5, which is auto-decremented each turn, to count the 7 turns when the player is hungry. The entries needed in the Status Table are:-

HUNGER	*	Conds	EQ	31	6
			ZERO	32	
		Acts	LET	5	8
HUNGER	*	Conds	NOTZERO	5	
			ZERO	12	
		Acts	MESSAGE	0	
HUNGER	*	Conds	EQ	5	1
			ZERO	12	
		Acts	MESSAGE	2	
			PAUSE	100	
			SID	24	15
			SID	6	240
			SID	1	53
			SID	0	57
			SID	4	33
			PAUSE	20	
			SID	1	39
			SID	0	223
			PAUSE	20	
			SID	1	29
			PAUSE	20	
			SID	1	22
			SID	0	96
			PAUSE	90	
			SID	24	0
			SCORE		
			TURNS		
			END		

Note that the vocabulary word HUNGER is only used as a comment.

The condition EQ must be followed by a flagno. & a value, and is satisfied if the flag specified has the value specified. Condition NOTZERO must also be followed by a flagno. and is satisfied if the flag specified does not have the value 0. Insert these three entries into the Status Table but make sure you insert them in the order shown. The first entry waits until the turns count (Flags 31 and 32) is equal to 6 and it then gives Flag 5 a value of 8. The second entry is the one which prints "I'm hungry!" and it does this when Flag 5 is not zero and Flag 12 says the apple has not been eaten. Flag 5 is auto-decremented each turn. The conditions in the third entry are satisfied when Flag 5 reaches 1 and Flag 12 says the apple has not been eaten. Note that when Flag 5 has the value 1 the conditions in both the second and third entries are satisfied so both the messages "I'm hungry!" and "I'm dying of starvation... " are printed.

Eating the Apple

We have already said that we will set Flag 12 when the apple is eaten. However, we only want to print message 1 if the apple is eaten when the player is hungry i.e. if the apple is eaten before the player is hungry we just want the reply "OK". The entries required in the Event Table are:-

```
EAT APPLE     Conds     PRESENT    2
                        NOTZERO    5
              Acts      DESTROY    2
                        SET       12
                        MESSAGE    1
                        DONE
EAT APPLE     Conds     PRESENT    2
              Acts      DESTROY    2
                        SET       12
                        OK
```

Action DONE simply stops the Interpreter falling through to the next entry in the table. Insert these two entries into the Event (Not Status) table in the correct order.

Number of Objects Conveyable

You may remember when you were testing the adventure earlier, that the player was able to carry only 4 objects. This number can be altered by selecting P on the Main Menu. For our mini adventure change the number of objects conveyable to 1 (one) and then test the adventure for the last time (it won't be quite so easy this time).

Save and Verify Adventure

These two options on the Main Menu allow complete copies of an adventure to be saved in a form that will auto-run when loaded into a COMMODORE using LOAD"name",8,1 or SHIFT/RUN STOP. Note that the files saved are not designed to be reloaded into The Quill.

Deleting Words from the Vocabulary

Note that if all the synonyms of a word are deleted from the vocabulary

then:-

a) Any entries in the Event and Status Tables using those words are also deleted.

b) Any movements in the Movement table using those words are also deleted.

c) Deleting words from the vocabulary can sometimes take a few minutes.

Other Actions

Most of the actions available in The Quill were used in our mini adventure. The other actions are detailed in Part 2 of this manual. One use of the ANYKEY action is to provide an introduction to adventures. If the introduction occupies two screens, then insert it into the database as the texts for locations 0 & 1 and let the adventure proper start at location 2. The following entries in the Status table can then be used to do the introduction:-

```
INTRO    *    Conds     AT        0
              Acts      ANYKEY
                        GOTO      1
                        DESC

INTRO    *    Conds     AT        1
              Acts      ANYKEY
                        GOTO      2
                        DESC
```

More about Event and Status

When the interpreter is processing these tables it considers each entry until it reaches the end of the table or it performs one of the actions INVEN, DESC, END, DONE, OK, SAVE or LOAD. The actions QUIT, REMOVE, GET, DROP, and WEAR can sometimes also stop processing of the Event/Status tables. Full details of all the actions can be found in Part 2 of this manual.

Other Conditions

Our mini adventure made use of only 6 of the conditions available in The Quill. Details of the other conditions can be found in Part 2 of this manual but it is worth mentioning the condition CHANCE here as it can introduce a random element into an adventure. Condition CHANCE must be followed by a percentage, in the range 0-99, e.g. CHANCE 25 would have a 25% chance of being satisfied.

Designing your own Adventure

The suggested procedure of writing your own adventure is:-

a) Read Part 2 of this manual

b) Draw a map of your adventure and allocate locnos. (make your first adventure a small adventure)

c) List the location texts you will use

d) List the Movement table entries you will need

e) List the objects in the adventure showing where they will start the adventure. Allocate objnos

f) List the messages needed in the adventure and allocate mesnos

g) List all the words and synonyms you will use and allocate word values

h) Decide which flags will be used and what they will be used for

i) Write out all the Event table entries showing the conditions and actions to be used. If you have more than one entry with the same word values check that you have the entries in the correct order

j) Write out all the Status table entries and plan the order of the entries. You might need to add further entries to the vocabulary which will be used as comments in the Status table

k) Insert the entries into the database and save the database regularly (you might get a power cut)

l) Thoroughly test your adventure

As you can see you need to do a lot of planning before you start to type your adventure into The Quill.

Selling your Adventure

If you intend to sell adventures then they need to be very thoroughly tested by as many people as possible. In particular:

a) Check the spelling of every word

b) Check it is impossible to get a score of over 100%

c) Try to move in every direction from every location

d) Try to GET, DROP, WEAR and REMOVE each object

e) It should be possible to solve the adventure each time it is played provided the correct commands are used e.g. poison gas which has a 1% chance of appearing and killing the player should be avoided unless you also provide a gas mask.

A great deal of time has been spent testing The Quill but it is possible that a few well hidden bugs still remain within the 12K of machine code. If you do have any problems please let us know so that they can be corrected. We would also be pleased to hear of any comments or criticisms about The Quill or any suggestions for improvements.

If you intend to sell an adventure written with The Quill we would be grateful if you could mention somewhere in it, that it was written with The Quill.

We would be pleased to look at adventures you have written with a view to marketing them. If you are interested please send a 'fully tested' copy together with a map and solution to:-

GILSOFT
30 Hawthorn Road
Barry
South Glamorgan

Part 2

Detailed Description of the Interpreter

The structure of the Interpreter is shown in flowchart 1. The numbers shown on the flowchart relate to the following paragraphs:-

1 Initialise

All the flags are set to zero except for Flag 1 which is set to the number of objects initially carried. The current location and the two word values W1 & W2 are set to zero.

2 Describe Current Location

The screen is cleared. If Flag 2 is non zero it is decremented. If it is dark (Flag 0 is non zero) and Flag 3 is non zero it is decremented. If it is dark and Flag 4 is non zero it is decremented if Object 0 is absent.

If it is light (Flag 0 is zero) or Object 0 is present the location text for the current location is printed. Then, if any objects are at this location, SM1 "I can also see:-" is printed followed by the object texts. SM1 refers to System Message number 1.

If it is dark (Flag 0 is non zero) and Object 0 is absent SM0 "Everything is dark. I can't see." is printed.

3 Consider 1st Entry in Status Table

A pointer is set to the first entry in the Status table.

4 Words & Conditions Satisfied

The 1st word & 2nd word in the entry being considered are matched with W1 & W2 as shown in flowchart 2. Each condition in the entry is checked as follows:-

4.1 AT locno.

Satisfied if the current location is the same as locno.

4.2 NOTAT locno.

Satisfied if the current location is different to locno.

4.3 ATGT locno.

Satisfied if the current location is greater than locno.

4.4 ATLT locno.

Satisfied if the current location is less than locno.

4.5 PRESENT objno.

Satisfied if Object objno. is carried or worn or at the current location.

4.6 ABSENT objno.

Satisfied if Object objno. is not carried and not worn and not at the current location.

4.7 WORN objno.

Satisfied if Object objno. is worn.

4.8 NOTWORN objno.

Satisfied if Object objno. is not worn.

4.9 CARRIED objno.

Satisfied if Object objno. is carried.

4.10 NOTCARR objno.

Satisfied if Object objno. is not carried.

4.11 CHANCE percent.

Satisfied if percent. is less than or equal to a random number in the range 1-100 (inclusive).

4.12 ZERO flagno.

Satisfied if Flag flagno. is set to zero.

4.13 NOTZERO flagno.

Satisfied if Flag flagno. is not set to zero.

4.14 EQ flagno. value

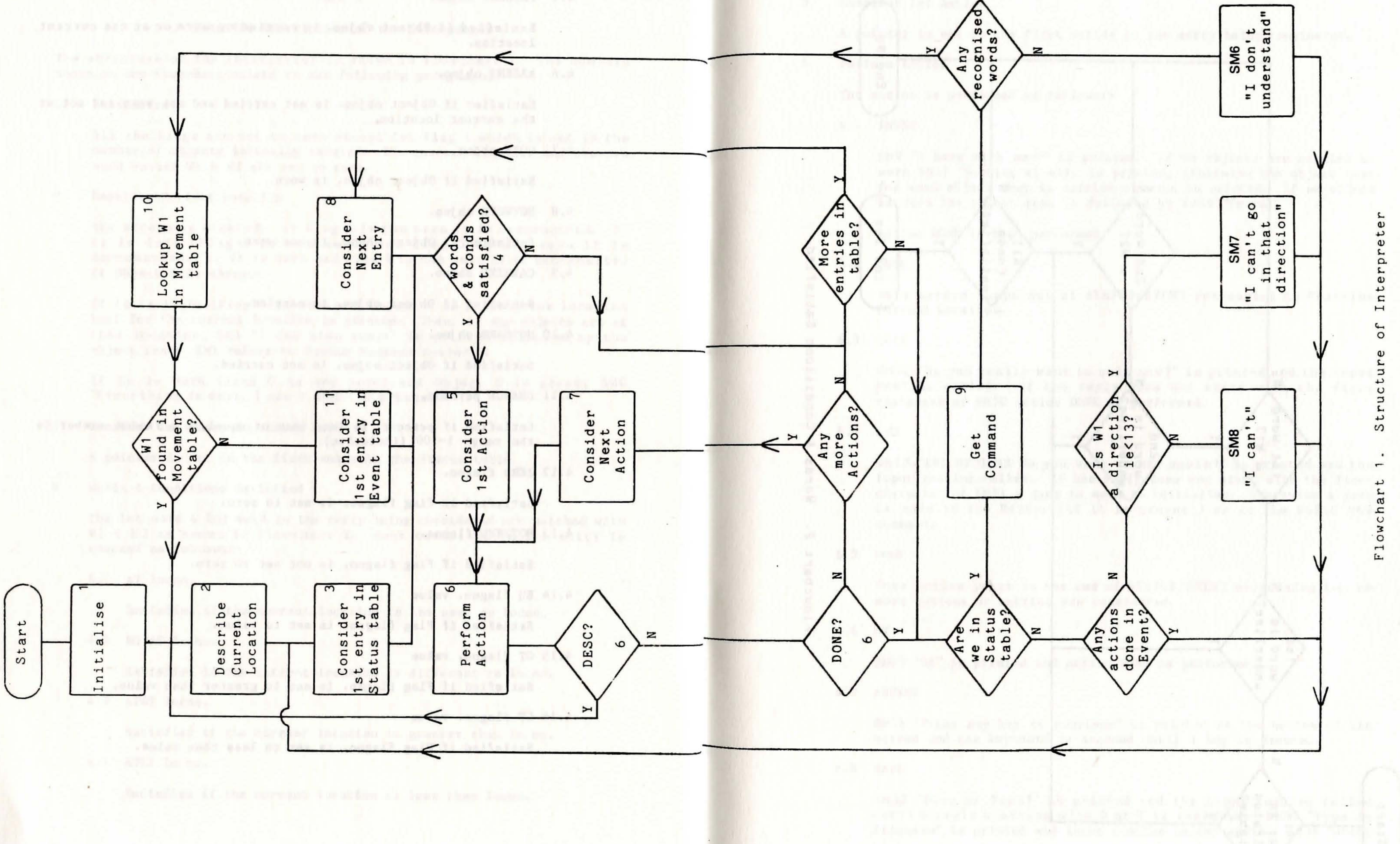
Satisfied if Flag flagno. is set to value.

4.15 GT flagno. value

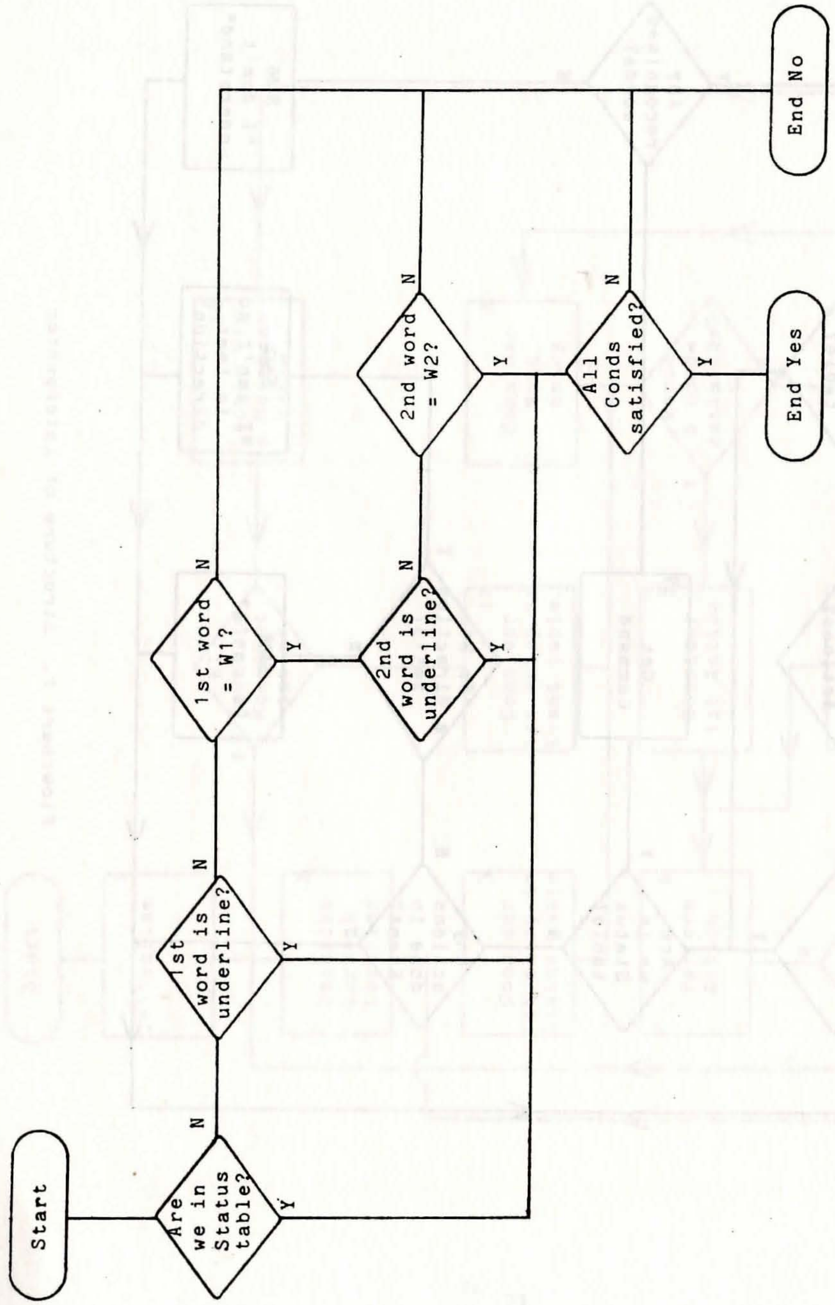
Satisfied if Flag flagno. is set to greater than value.

4.16 LT flagno. value

Satisfied if Flag flagno. is set to less than value.



Flowchart 1. Structure of Interpreter



Flowchart 2. Words & Conditions Satisfied

5 Consider 1st Action

A pointer is set to the first action in the entry being considered.

6 Perform Action

The action is performed as follows:-

6.1 INVEN

SM9 "I have with me:-" is printed. If no objects are carried or worn SM11 "Nothing at all." is printed, otherwise the object text for each object that is carried or worn is printed. If an object is worn its object text is followed by SM10 "(worn)".

Action DONE is then performed.

6.2 DESC

This action jumps out of STATUS/EVENT processing to Describe Current Location.

6.3 QUIT

SM12 "Do you really want to quit now?" is printed and the input routine called. If the reply does not start with the first character of SM30 action DONE is performed.

6.4 END

SM13 "END OF GAME Do you want to try again?" is printed and the input routine called. If the reply does not start with the first character of SM31 a jump is made to Initialise. Otherwise a jump is made to the Editor (if it is present) or to the BASIC NEW command.

6.5 DONE

This action jumps to the end of STATUS/EVENT processing i.e. no more actions or entries are considered.

6.6 OK

SM15 "OK" is printed and action DONE is performed.

6.7 ANYKEY

SM16 "Press any key to continue" is printed at the bottom of the screen and the keyboard is scanned until a key is pressed.

6.8 SAVE

SM32 "Disc or Tape?" is printed and the input routine called until a reply starting with D or T is received. SM33 "Type in Filename" is printed and input routine called again. SM34 "start the tape" is printed if appropriate. The game position is then saved and action DESC is performed. If RUN STOP is pressed during the save or an I/O ERROR occurs, a jump is made to Describe Current Location.

6.9 LOAD

SM32 "Disc or Tape?" is printed and the input routine called until a reply starting with D or T is received. SM33 "Type in Filename" is printed and input routine called again. SM34 "start the tape" is printed if appropriate. The game position is then loaded and action DESC is performed. If RUN STOP is pressed during the load or an I/O ERROR is detected a jump is made to Initialise. If data is loaded which is not a game position an I/O ERROR will normally be detected.

6.10 TURNS

SM17-20 "You have taken x turn(s)." is printed where x is Flag 31 + 256 * Flag 32.

6.11 SCORE

SM21-22 "You have scored x%" is printed where x is Flag 30.

6.12 PAUSE value

Pauses for value/50 secs.

6.13 GOTO locno.

Changes the current location to locno.

6.14 MESSAGE mesno.

Message mesno. is printed.

6.15 REMOVE objno.

If Object objno. is not worn, SM23 "I'm not wearing it." is printed and action DONE is performed.

If the maximum number of objects is being carried (Flag 1), SM24 "I can't. My hands are full." is printed and action DONE is performed.

Otherwise the position of Object objno. is changed to 'carried' and Flag 1 is incremented.

6.16 GET objno.

If Object objno. is worn or carried, SM25 "I already have it." is printed and action DONE is performed.

If Object objno. is not at the current location, SM26 "It's not here." is printed and action DONE is performed.

If the maximum number of objects is being carried (Flag 1), SM27 "I can't carry any more." is printed and action DONE is performed.

Otherwise the position of Object objno. is changed to 'carried' and Flag 1 is incremented.

6.17 DROP objno.

If Object objno. is worn and the maximum number of objects is being carried (Flag 1), SM24 "I can't. My hands are full." is printed and action DONE is performed.

If Object objno. is neither worn nor carried, SM28 "I don't have it." is printed and action DONE is performed.

Otherwise the position of Object objno. is changed to the current location and Flag 1 is decremented if the object was carried.

6.18 WEAR objno.

If Object objno. is worn, SM29 "I'm already wearing it." is printed and action DONE is performed.

If Object objno. is not carried, SM28 "I don't have it." is printed and action DONE is performed.

Otherwise the position of Object objno, is changed to 'worn' and Flag 1 is decremented.

6.19 DESTROY objno.

The position of Object objno. is changed to 'not created' and Flag 1 is decremented if the object was carried.

6.20 CREATE objno.

The position of Object objno. is changed to the current location and Flag 1 is decremented if the object was carried.

6.21 SWAP objno. objno.

The positions of the two objects are exchanged.

6.22 SET flagno.

Flag flagno. is set to 255.

6.23 CLEAR flagno.

Flag flagno. is cleared to 0.

6.24 PLUS flagno. value

Flag flagno. is increased by value. If the result exceeds 255 the flag is set to 255.

6.25 MINUS flagno. value

Flag flagno. is decreased by value. If the result is negative the flag is set to 0.

6.26 LET flagno. value

Flag flagno. is set to value.

6.27 SID regno. value.

SID register regno. is set to value.

6.28 BORDER value.

The Border colour is changed to value.

6.29 PAPER value.

The global Paper colour is changed to value.

6.30 INK value.

The global Ink colour is changed to value.

6.31 CLS

The screen is cleared.

6.32 DROPALL

All objects which are carried or WORN are created at the current location (i.e. all objects are dropped) and Flag 1 is set to 0.

6.33 PLACE objno. locno.

The position of object objno. is changed to locno. and Flag 1 is decremented if the object was carried.

7 Consider Next Action

A pointer is set to the next action in the entry being considered.

8 Consider Next Entry

A pointer is set to the next entry in the table being processed.

9 Get Command

If Flags 5 to 8 are non zero they are decremented. If it is dark (Flag 0 is non zero) and Flag 9 is non zero it is decremented. If it is dark and Flag 10 is non zero it is decremented if Object 0 is absent.

The turns count (Flags 31 & 32) is incremented, one of four messages SM2-5 e.g. "Tell me what to do." is printed and the input routine called. The first four letters of each word in the reply are looked up in the vocabulary and the values of the first two words found in the vocabulary are stored in W1 & W2.

10 Lookup W1 in Movement Table

The entry in the movement table for the current location is searched to see if the word value in W1 is present. If it is, the current location is set to the value following W1 in the movement table.

11 Consider 1st Entry in Event Table

A pointer is set to the first entry in the Event table.

The basic structure of the interpreter has its origins in an article written by Ken Reed and published in the August 1980 issue of Practical Computing. The interpreter in The Quill now has little in common with the article but the terms used to describe it (e.g. Status, Event, Flags) have been retained as an aid to people who are familiar with the article.

Detailed Description of the Database

The database consists of a number of inter-related tables and also contains an area of miscellaneous information e.g. values of permanent colours, number of objects conveyable. The tables present are:-

A The Vocabulary

Each entry in the table uses 5 bytes and contains a word (or the first four characters, if the word is longer than four characters) and a word value in the range 1-254. Words with the same word value are called synonyms. The entries are held in ascending order of word value and within each word value, entries with more spaces come first e.g.

```
U
UP
CLIM
ASCE
```

where entries with the same word value also have the same number of spaces the entry inserted first comes earlier e.g. CLIM(B) was inserted before ASCE(ND).

Note 1. Whenever the editor has to convert from a word value to a word it takes the first word with that value.

Note 2. Word values less than 13 should be reserved for movement words.

B The Message Text table

This table contains the text of any messages which are needed for the adventure. The messages are numbered from 0 upwards and each one uses 3 bytes plus the length of the text.

C The Location Text table

This table, which has an entry for each location, contains the text which is printed when a location is described. Each entry uses 3 bytes plus the length of the text. The entries are numbered from 0 upwards and location 0 is the location at which the adventure starts. Whenever a new location is inserted a null entry for that location is also made in the movement table.

D The Movement Table

This table has an entry for each location and each entry may either be empty (null) or contain a number of 'movement pairs'. A movement pair consists of a word value in the vocabulary followed by a location number and means that any word with that word value causes movement to that location. A typical entry could be SOUTH 6 EAST 7 RETURN 6 NORTH 5 which means that SOUTH or RETURN or their synonyms cause movement to location 6, EAST or it's synonyms to location 7 and NORTH or it's synonyms to location 5. Each entry uses 3 bytes plus 2 bytes for each movement pair.

Note 1. The movement pairs contain the word value not the actual word and if a word value is deleted from the vocabulary then all movement pairs which contain that word value are also deleted.

Note 2. When the adventure is being played it is only the first recognised word (W1) which will cause movement.

Note 3. If any movements are to be performed in the Event or Status tables using the action GOTO then those movements should be excluded from the Movement table.

E The Object Text table

This table, which has an entry for each object, contains the text which is printed when an object is described. Each entry uses 3 bytes plus the length of the text. An object is anything in the adventure which may be manipulated and objects are numbered from 0 upwards. Object 0 is assumed by the Interpreter to be a source of light. Whenever a new object text is inserted an entry of 'not created' is made for that object in the object start location table.

F The Object Start Location table

This table has a 1 byte entry for each object, which specifies the location at which the object is situated at the begining of the adventure. An object can also start the adventure being worn, carried or not created.

G The Event table

This table (together with the Status table) is the main part of the database and each entry contains 2 word values followed by any number of conditions and then (normally) at least one action. When the adventure is played, if there is an entry in the table with the word values entered and the conditions specified are satisfied then the actions are performed. The conditions and actions that may be present and the effect that they have is fully specified in the description of the Interpreter. The order of entries in the table is in ascending order of the first word value. Entries which have the same first word value are held in ascending order of the second word value. Entries with the same first and second word values are held in the order they were inserted into the database (i.e. they must be inserted in the order required). An example of the order of the table, with word values shown in brackets, is as follows:-

```
LOOK(30) UP (9)
LOOK(30) DOWN(10)
LOOK(30) * (255)
GET (100) KEY (16)
GET (100) LAMP(26)
GET (100) LAMP(26)
```

Each entry in the table has an overhead of 6 bytes and each condition and action uses 1, 2 or 3 bytes depending on the number of parameters.

Note 1. If a word value is deleted from the vocabulary then all entries in the Event and Status tables which contain that word value are also deleted.

H The Status table

This table has exactly the same format as the Event table. When the adventure is played the Status table is scanned between turns to see if the Commodore wants anything to happen. The Event table can be considered as the players's table and contains entries which are dependent on the words entered, while the Status table is the computers's table and contains entries which are independent of the words entered by the player. The words in the Status table can however be used to position entries at the required place and/or as a reminder of the purpose of the entries.

I The System Messages

This table contains the messages used by the Interpreter. Each entry uses 3 bytes plus the length of the text. The description of the Interpreter shows when these messages are used.

Detailed Description of the Editor

A Vocabulary

Words may be inserted or deleted, the synonyms of a word may be displayed or the vocabulary may be printed:-

Insert I word No. (No. is in the range 1-254)

If word is not already present in the vocabulary it is inserted with a word value of No.

Delete D word

If word is present in the vocabulary, it and it's word value are deleted. If synonyms of the word deleted are present in the vocabulary no further action is taken. However, if no synonyms are present, then:-

a) all entries in the Event and Status tables which use this word value are also deleted.

b) all movements in the movement table which use this word value are also deleted.

Show Synonyms S word

If word is present in the vocabulary, it and all other words with the same word value are displayed.

Print P or L

Printing is either to the screen using P or to the printer using L.

Points to note:

a) Be careful using delete as it can also affect the Event, Status and Movement tables.

b) Words with a word value of less than 13 are assumed to be movement words by the Interpreter and cause the message "I can't go in that direction," to be printed instead of "I can't".

c) When entries in the Event, Status and Movement tables are affected, the Editor may take a few minutes to delete a word.

B Message Text

Message texts may be inserted, amended or printed:-

Insert I

The next available message number is used and a null entry is made for it in the message text table. An automatic call to the amend routine is then made to allow the user to amend the null entry.

Amend A **mesno.**

The existing text for message **mesno.** is copied to the input buffer and displayed at the bottom of the screen for amending. When RETURN is pressed the existing text is replaced with the contents of the input buffer.

Print P (**mesno.**) or L (**mesno.**)

Printing is either to the screen using **P** or to the printer using L. Printing starts with the text for message **mesno.** or at the begining if **mesno.** is not specified.

Points to note:

a) There is a limit of 255 messages.

C Location Text

Location texts may be inserted, amended or printed:-

Insert I

The next available location number is used and a null entry is made for it in both the movement table and the location text table. Processing then continues with an automatic call to the amend routine to allow the user to amend the null entry already set up in the location text table.

Amend A **locno.**

The existing text for Location **locno.** is copied to the input buffer and displayed at the bottom of the screen for amending. When RETURN is pressed the existing entry is replaced with the contents of the input buffer.

Print P (**locno.**) or L (**locno**)

Printing is either to the screen using **P** or to the printer using L. Printing starts with the text for Location **locno.** or at the begining if **locno.** is not specified.

Points to note:

a) The start of an adventure is always at Location 0.

b) There is a limit of 252 locations.

D Movement table

Movements may be amended or printed:-

Amend A **locno.**

The existing entry for Location **locno.** is decoded, copied to the input buffer and displayed at the bottom of the screen for amending. When RETURN is pressed the input buffer is vetted to be empty or to contain **word locno.** repeated any number of times. **word** must be present in the vocabulary and **locno.** must be present in the location text table. If there are no syntax errors the existing entry is replaced with an encoded copy of the input buffer (i.e. words changed to word values).

Print P (**locno.**) or L (**locno.**)

Printing is either to the screen using **P** or to the printer using L. Printing starts with the entry for Location **locno.** or at the begining if **locno.** is not specified.

Points to note:

a) A location text must be present for a Location before movements can be present.

b) Any words in the Vocabulary may be used in the Movement table.

c) When an entry is decoded (for Amend or Print) the word value is changed into the first word in the Vocabulary with that word value.

E Object Text

Object texts may be inserted, amended or printed:-

Insert I

The next available object number is used and a null entry is made for it in the object text table. An entry of 'not created' is also made for it in the object start location table. Processing then continues with an automatic call to the amend routine to allow the user to amend the null entry already set up in the object text table.

Amend A **objno.**

The existing text for Object **objno.** is copied to the input buffer and displayed at the bottom of the screen for amending. When RETURN is pressed the existing text is replaced with the contents of the input buffer.

Print P (**objno.**) or L (**objno.**)

Printing is either to the screen using **P** or to the printer using L. Printing starts with the text for Object **objno.** or at the begining if **objno.** is not specified.

Points to note:

a) Object 0 is considered by the Interpreter to be a source of light.

b) There is a limit of 255 objects.

F Object Start Location Table

The location at which an object is situated at the start of the adventure may be amended or the object start location table may be printed:-

Amend A **objno. locno.**

The existing entry for Object **objno.** is replaced with **locno.** which must either be present in the location text table or be one of the special locnos. 252 not created, 253 worn or 254 carried.

Print **P** or L

Printing is either to the screen using **P** or to the printer using L

Points to note:

a) An object text must be present for an object before it's start location can be present.

G Event Table

Entries may be Inserted, Amended, Deleted or Printed:-

Insert I **word1 word2**

Word1 and **Word2** must be asterisks or words which are in the vocabulary. The word values of **Word1** and **Word2** (asterisk has a word value of 255) are used to find the correct place in the table for the new entry to be created. If any entries already exist for **Word1 Word2** then the new entry will be created after the existing entries. A null entry is created at the appropriate place and an automatic call made to the amend routine to allow the user to amend the null entry.

Amend A **Word1 Word2**

The first entry in the table with word values of **Word1** and **Word2** is copied to the input buffer and displayed at the bottom of the screen for amending. When RETURN is pressed the input buffer is vetted to be empty, in which case the existing entry is deleted, or to contain any number of valid conditions followed by at least one valid action. If there are no syntax errors the existing entry is replaced with the contents of the input buffer. Any following entries in the table with the same word values (i.e. **Word1** and **Word2**) are then displayed in turn for amending in the same way.

The conditions that may be used are:-

AT	locno.		
NOTAT	locno.		
ATGT	locno.		
ATLT	locno.		
PRESENT	objno.		
ABSENT	objno.		
WORN	objno.		
NOTWORN	objno.		
CARRIED	objno.		
NOTCARR	objno.		
CHANCE	percent		(percent, is in the range 0-99)
ZERO	flagno.		
NOTZERO	flagno.		
EQ	flagno.	value	
GT	flagno.	value	(value is in the range 0-255)
LT	flagno.	value	

The description of the Interpreter will tell you what these conditions do. The actions that may be used are:-

θ	INVEN	
θ	DESC	
ϕ	QUIT	
θ	END	
θ	DONE	
θ	OK	
	ANYKEY	
θ	SAVE	
θ	LOAD	
	TURNS	
	SCORE	
	PAUSE	value (value is 1/50 sec in the range 0-255)
	GOTO	locno.
	MESSAGE	mesno.
ϕ	REMOVE	objno.
ϕ	GET	objno.
ϕ	DROP	objno.
ϕ	WEAR	objno.
	DESTROY	objno.
	CREATE	objno.
	SWAP	objno. objno.
	SET	flagno.
	CLEAR	flagno.
	PLUS	flagno. value
	MINUS	flagno. value (value is in the range 0-255)
	LET	flagno. value
	SID	regno. value
	BORDER	value
	PAPER	value
	INK	value
	CLS	
	DROPALL	
	PLACE	objno. locno.

The description of the Interpreter will tell you in detail what these actions do. However, please note that the actions marked θ will always cause an exit from the Event/Status table entry being processed so any following actions will never be performed. The actions marked ϕ may cause an exit from the table being processed.

Delete

To delete an entry amend it so that no conditions or actions remain.

Print P (word1 (word2)) or L (word1 (word2))

Printing is either to the screen using P or to the printer using L. Printing starts at the first entry with word values of **Word1 Word2**. If **Word1** or **Word2** is not specified then a word value of 0 is assumed. Thus P or L by itself starts at the beginning of the table.

H Status Table

The Status table is handled in exactly the same way as the Event Table.

I Save Database

Saves the database to tape or disc.

J Verify Database

Verifies that a database has been saved correctly.

K Load Database

Loads a database into The Quill.

Very Important

If RUN STOP is pressed or an I/O ERROR is detected during a load then the database held in memory will be corrupt and should not be used as it may corrupt the Editor and Interpreter. Under these circumstances the only Editor command which may be used safely is Load Database and this should be used until a database is loaded successfully.

L Test Adventure

"Do you require diagnostics?" is printed and any reply that doesn't start with "Y" is assumed to be negative. A jump is then made to the Interpreter. If diagnostics are required then whenever the Interpreter's input routine is used the bottom of the screen will display the values of the flags, the value of W1, W2 and the value of the current location. The first of the four lines will contain the values of flags 0-9, the second line flags 10-19 the third line flags 20-29 and the fourth line flags 30-32, followed in inverse, by the values of W1, W2 and the current location. Note that the use of diagnostics will cause the screen to scroll earlier than usual.

Very Important

The only way back to the Editor from the Interpreter is by performing the action END in either the EVENT or STATUS tables.

M Save Adventure

The Interpreter and database are saved to tape with the file name specified and in such a way that the Adventure will auto run when loaded from BASIC using LOAD"name",8,1 or SHIFT/RUN STOP.

N Verify Adventure

Verifies that an Adventure has been saved correctly.

O Bytes Spare

The number of spare bytes is printed.

P Objects Conveyable

The number of objects that can be carried at any one time may be set to any value from 0 to 255. A value less than 10 will normally be used.

Q Permanent Colours

The BORDER, PAPER & INK colours may be set to any valid values.

R System Messages

System message texts may be amended or printed:-

Amend A no.

The existing text for message no. is copied to the input buffer and displayed at the bottom of the screen for amending. When RETURN is pressed, the existing text is replaced with the contents of the input buffer.

Print P (no.) or L (no.)

Printing is either to the screen using P or to the printer using L . Printing starts with the text for message no. or at the beginning if no. is not specified.

Points to note:

a) The description of the Interpreter shows where these messages are used.

b) Messages 30 and 31 are not really messages but contain the positive and negative replies used in the QUIT and END actions. Therefore be very careful changing these as action END is the only way back to the Editor from the Interpreter.

c) These messages may be changed to use "You" instead of "I" if you prefer, i.e. "You're not wearing it", or even into different languages.

d) SM10 "(WORN)" should not include any colour control codes.

(+) Return to BASIC

Jumps to the BASIC NEW command.

Editor Error Messages and their meanings

RUN STOP	RUN STOP was pressed.
I/O ERROR	Disc or tape error. Note that an I/O ERROR during a LOAD means that the database is corrupt.
Database full	Insufficient room in the database for what you were attempting
Limit reached	The maximum number of locations, messages or objects is already present
I/P Buffer full	Entry for Event, Status or Movement table is too large for input buffer (very unlikely to occur). The remedy is to use a smaller entry

Note If an abnormally large entry is inserted in the movement table using abbreviations e.g. N 1 W 6 S 4 etc and the abbreviations are deleted from the vocabulary, the movement entry (when decoded) i.e. NORT 1 WEST 6 SOUT 4 etc could be too big for the input buffer. If this happens an I/P Buffer full message will be produced. The remedy is to reinsert the abbreviations in the vocabulary.

Summary of Conditions, Actions and Flags

Conditions

AT	locno.
NOTAT	locno.
ATGT	locno.
ATLT	locno.
PRESENT	objno.
ABSENT	objno.
WORN	objno.
NOTWORN	objno.
CARRIED	objno.
NOTCARR	objno.
CHANCE	percent
ZERO	flagno.
NOTZERO	flagno.
EQ	flagno. value
GT	flagno. value
LT	flagno. value

Actions

INVEN	
DESC	
QUIT	
END	
DONE	
OK	
ANYKEY	
SAVE	
LOAD	
TURNS	
SCORE	
PAUSE	value
GOTO	locno.
MESSAGE	mesno.
REMOVE	objno.
GET	objno.
WEAR	objno.
DROP	objno.
DESTROY	objno.
CREATE	objno.
SWAP	objno. objno.
SET	flagno.
CLEAR	flagno.
PLUS	flagno. value
MINUS	flagno. value
LET	flagno. value
SID	regno. value
BORDER	value
PAPER	value
INK	value
CLS	
DROPALL	
PLACE	objno. locno.

Flags

Flag 0	If this flag is set to zero it is light. Any other value means it's dark
Flag 1	holds count of objects carried
Flag 2	decreased when a location is described
Flag 3	decreased when a location is described and it's dark
Flag 4	decreased when a location is described and it's dark and object 0 is absent
Flags 5-8	decreased each turn
Flag 9	decreased each turn when it's dark
Flag 10	decreased each turn when it's dark and Object 0 is absent
Flags 11-29	ordinary flags
Flag 30	holds the score
Flag 31	holds turns count LSB
Flag 32	holds turns count MSB