original list of words without having to go to the trouble of writing it all out again. One way of extending a list is to use the operation SENTENCE, which takes two inputs and makes a list from them. So SENTENCE "JAM [HONEY JAR] outputs [JAM HONEY JAR].

```
TO ADDWORDS1 :LIST
   MAKE "WORDS SENTENCE :LIST :WORDS
END
```

So we can now extend WORDS with ADDWORDS [ANXIETY REPRESSION [FEAR OF FLYING]]. The problem with this is if the variable WORDS has not previously been assigned a value. The primitive THING? is used to overcome this by testing if a variable has been assigned a value; it outputs true if its input has a value associated with it. We can now improve our list of extra words with ADDWORDS1:

```
TO ADDWORDS1 :LIST
   IF NOT THING? "WORDS THEN MAKE "WORDS []
   MAKE "WORDS SENTENCE :LIST :WORDS
END
```

Using a different list of words, we obtained the following piece of 'poetry' using this procedure:

APPARITION LOUDLY SPOKE SPLENDID PARANOID PLANET TERRIFIED THE WITH GREEN APPARITION FLOATING PARANOID ROBOT MAN FLEW SPOKE FLOATING LOUDLY

One of the more obvious failings of our computerised poetry is its total disregard for English grammar. The poems might make more sense if we could constrain them to some simple syntactical patterns — such as: noun, verb, noun. One way to do this is to have a number of lists, one for each part of speech. We could then choose one word from each list according to our desired sentence structure.

We leave this problem for you to explore and investigate. In the next instalment of the course, we will show you some ways of how to improve the turtle's poetry-writing abilities.

Logo Flavours

Some versions of MIT LOGO do not have EMPTY?, ITEM and COUNT. In all LCSI versions, use:

EMPTYP for EMPTY?
LISTP for LIST?
TYPE for PRINT1

There is a primitive, EQUALP, which tests whether its two inputs are the same. Use this for comparing lists and words in place of the equals sign (=). (The equals sign works for lists on some LCSI versions, but not on others.)

Remember the different IF syntax:
IF EMPTYP :LIST [OUTPUT 0]
On Atari LOGO use SE for SENTENCE, and note that ITEM is not implemented

Exercises

1) Write a procedure to print a list in reverse order (use LAST and BUTLAST). Modify this procedure so that it outputs the reversed list

2) Write a procedure that removes an element from a list. So DELETE "FOOD [DRINK FOOD] outputs [DRINK] and DELETE "WINE [DRINK FOOD] outputs [DRINK FOOD]

Exercise Answers

Answers to the exercises on page 737:

1. Calculation powers:

```
TO POWER :A :N
   IF NOT ( ( INTEGER :N ) = :N ) THEN PRINT
   [WHOLE NUMBER INDICES ONLY] STOP
   IF :N = 0 THEN OUTPUT 1
   OUTPUT :A * POWER :A :N – 1
END
```

2. Converting to hexadecimal:

```
TO HEX.PRINT :NO
   IF :NO < 10 THEN OUTPUT :NO
   IF :NO = 10 THEN OUTPUT "A
   IF :NO = 11 THEN OUTPUT "B
   IF :NO = 12 THEN OUTPUT "C
   IF :NO = 13 THEN OUTPUT "D
   IF :NO = 14 THEN OUTPUT "E
   IF :NO = 15 THEN OUTPUT "F
END

TO HEX :NO
   IF :NO = 0 THEN STOP
   HEX QUOTIENT :NO 16
   PRINT1 HEX.PRINT REMAINDER :NO 16
END
```

3. Testing if a number is even:

```
TO EVEN? :NO
   IF ( ( REMAINDER :NO 2 ) = 0 ) THEN OUTPUT
   "TRUE OUTPUT "FALSE
END
```

4. Finding an area using the Monte Carlo method:

```
TO MC
   DRAW PU MAKE "IN 0
   MC1 1000 10 100
   ( PRINT [AREA IS] ( :IN ) )
END

TO MC1 :NO :XNO :YNO
   IF :NO = 0 THEN STOP
   RANDOM.POINT :XNO :YNO
   IF INSIDE? THEN MAKE "IN :IN + 1
   MC1 :NO – 1 :XNO :YNO
END

TO RANDOM.POINT :XNO :YNO
   SETXY RANDOM :XNO RANDOM :YNO
END

TO INSIDE?
   IF YCOR < XCOR * XCOR THEN OUTPUT "TRUE
   OUTPUT "FALSE
END
```