```
NEW
OK
10 FOR I=-32767 TO -1
20 PRINT I, UNS(I)
30 NEXT I
-32767          32769
-32766          32770
-32765          32771
^C
BREAK AT LINE 20
OK
```

Use <control-S> to suspend execution so you can look at the values, and use <control-C> when you wish to exit from this program.

POKE

In addition to letting you to examine memory locations with the PEEK function, XYBASIC allows you to put data directly into memory with the POKE command. If your system uses memory mapped I/O, you can use POKE to perform output. To modify location 4 to contain 0, just say

```
POKE 4,0
```

Don't try this unless you are sure modifying location 4 will not endanger your system! POKE is very dangerous -- you can easily modify the operating system or the XYBASIC interpreter itself with careless POKEing, with unpredictable consequences -- so be extremely careful!

Example:
Herman Wayl has just bought a new memory board for his 8080 and wants to test it. It is an 8K board which occupies address space 16K to 24K. Herman tests it for bit failures with the following program.

```
NEW
10 FOR I = 16*1024 TO 24*1024 - 1        'SET UP LOOP PARAMETER
20 N = 0                                 'CLEAR TEST VALUE BITS
30 GOSUB 100                             'CHECK FOR ERROR
40 N = #FF                               'SET TEST VALUE BITS
50 GOSUB 100                             'CHECK FOR ERROR
60 NEXT I                                'AND TRY NEXT LOCATION
70 PRINT "TEST CONCLUDED"                'DONE
80 END
100 REM SUBROUTINE TO TEST LOCATION I WITH VALUE N
110 POKE I, N
120 IF PEEK (I) = N THEN RETURN          'VALUE IS CORRECT
130 PRINT "FAILURE AT LOCATION"; I; ":"; PEEK (I); "SHOULD BE"; N
140 RETURN
```

This program POKEs a 0 into a memory location on the new board, then PEEKs to check that the data reads back correctly and PRINTs an appropriate error message if it does not. Then the test is repeated with #FF (all 1's). After testing all memory locations, the program prints a concluding message.