

A Number Game for the 6502

by Steve Wozniak
 Apple Computers, 770 Welch Rd., Suite 154
 Palo Alto CA 94304; (415) 326-4248

DESCRIPTION

MASTERMIND is a number guessing game. In this version the computer generates a 5-digit, random number where the digits are limited to 0 through 7. The user attempts to guess the number in the fewest possible tries. After each guess the computer informs the user as to how many digits were correct (contained in the random number) and in the correct position by printing '+' signs, and how many digits are correct and in the wrong position with '-' signs. The user is not informed as to exactly which digit positions the '+' and the '-' signs correspond. A skilled player can usually win in 6 or 7 tries.

THE PLAY

1. Computer prints 'READY?'
2. User hits any key when ready (not echoed). A random number is generated by counting from the 'READY?' message to key depression.
3. Computer prints '01' for move number
 ↑
Cursor left here.
4. User enters his 5-digit guess following the move number. Remember that valid digits are 0 through 7 only. Entering any invalid character cancels the line and the computer repeats step 3 for the current move. This may be used to cancel errors.
5. Immediately after entering the 5th digit of his guess, the computer will print a number of '+' signs (for correct digits in correct positions) followed by a number of '-' signs (for correct digits in incorrect positions). This 'score' is indicated to the right of the guess and does not indicate the specific digit positions involved. Play resumes at 3 for the next move number except for a win. Examples follow:

Scoring Examples for Random Number 12154

Move	Guess	Score
01	33366	(none correct)
02	00018	(line cancelled due to invalid digit, 8)
02	00011	-- (2 digits right, in wrong positions)
03	11234	++-- (4 digits right, 2 in correct positions, 2 in wrong positions)
04	11325	+--- (4 digits right, 1 in correct position)
05	13216	+-- (3 digits right, 1 in correct position)
06	44444	+ (1 digit right in correct position)
07	55555	+ (1 digit right, in correct position)
08	12154	+++++ YOU WIN (win message)

READY? (for next game)

RUNNING ON APPLE-I SYSTEMS

The source and object listings provided will run on APPLE-I systems. The program loads in locations \$300 through \$3AE and uses the following page 0 locations for variables:

SF2	TRIES	(no. of tries - 0 to 99 BCD)
SF3	RNDL	} Binary
SF4	RNDH	
SF5	RND2L	(Temp.)
SF6-SFA	N	(5 digits of unpacked random no.)
SFB-SFF	GUESS	(User guess)

RUNNING ON OTHER SYSTEMS

1. The LDA STROBE instruction at loc. \$313 (on supplied listing) senses for a key down condition after the READY? message. This is used to generate a random number corresponding to the delay before a key is depressed. The code is written for a negative value to indicate 'key down' (b₇ = 1) and a positive value (b₇ = 0) to indicate no key down. This instruction is followed by a JSR CHARIN instruction whose only purpose is to clear the strobe. The character returned is not used. Thus the user must provide an address for the LDA STROBE instruction corresponding to his hardware and must insure that the CHARIN subroutine clears the strobe.

2. A CHARIN subroutine must be provided to read one ASCII character with b₇ set. Do not use the CHARIN subroutine provided as it uses APPLE-I I/O assignments. The character read must be returned in the A-Reg. The Y-Reg may be altered by CHARIN but not the X-Reg. If b₇ is returned clear (=0) then the EOR #\$B0 instruction at loc. \$34E (on supplied code) must be changed to an EOR #\$30.

3. A COUT subroutine must be provided which outputs one ASCII character (passed in A-Reg). If the user output device requires line feeds with carriage returns then the COUT routine must intercept the carriage return character (\$8D) and output the necessary CR-LF combination. All computer generated text has b₇=1. No registers (A,X,Y) may be altered.

4. PRBYTE subroutine must be provided which outputs one byte (passed in A-Reg) in hexadecimal (printing 2 digits). No registers need be preserved. The following routine will do:

PRBYTE	PHA	Save for LSD
	LSR	
	LSR	
	LSR	MSD to LSD position
	LSR	
	JSR PRHEX*	Output MSD first
	PLA	Restore A
PRHEX	AND #\$F	Mask LSD
PRHEX*	ORA #\$B0	Add "0"
	CMP #\$BA	} May be skipped if used for } MASTERMIND only, since } only BCD digits supplied.
	BCC TOCOUT	
	ADC #\$6	
TOCOUT	JMP COUT	Output ASCII and return

