There probably are faster formats for storing the information than the data file the PET assembler writes. But I know my simple method works. After the operating system writes the file header and file name, the file begins with the starting address chosen for the code. The number "999" is used for an END marker.

Lines 100 to 106 get the line of assembly language in. Use the GET command instead of INPUT to allow the instruction to be written in usual assembly language. (Otherwise, you would somehow have to get instructions like LDA 17 in as one string; usually, internal spaces in a string cause trouble if not enclosed in quotes.) ASCII 20, the "delete" command, must be treated specially, as must ASCII 13, the "return" key.

Once the line is in, it's split into a mnemonic part, L$, and an operand part, C$. C$ may be empty, for instructions such as NOP and TXA. The PET scans the list of legal mnemonics, and if it finds the string L$, it puts its number in the array into the variable MN. (If you enter an illegal mnemonic, the computer prints an error message and the program goes back to wait for more input.) Entering the command END closes the tape file (very important!) and ends the program's execution.

Once the computer finds the mnemonic, its category is looked up and put into the variable CA, and the base opcode is stored in OP. (These variables are distinct from arrays with the same name.)

The rest of the assembler program examines the instruc-

Assembler Program Listing

```
1   REM 6502 ASSEMBLER--COPYRIGHT 1978 MARK ZIMMERMANN
10  DATA ADC,97,1,AND,33,1,ASL,2,3,BCC,144,8,
      BCS,176,8,BEQ,240,8,BIT,36,7

12 DATA BMI,48,8,BNE,208,8,BPL,16,8,BRK,0,0,BVC,80,8,
      BVS,112,8,CLC,24,0

14 DATA CLD,216,0,CLI,88,0,CLV,184,0,CMP,193,1,
      CPX,224,4,CPY,192,4

16 DATA DEC,198,2,DEX,202,0,DEY,136,0,EOR,65,1,
      INC,230,2,INX,232,0

18 DATA INY,200,0,JMP,76,6,JSR,32,9,LDA,161,1,
      LDX,162,5,LDY,160,5

20 DATA LSR,66,3,NOP,234,0,ORA,1,1,PHA,72,0,
      PHP,8,0,PLA,104,0,PLP,40,0

22 DATA ROL,34,3,ROR,98,3,RTI,64,0,RTS,96,0,
      SBC,225,1,SEC,56,0,SED,248,0

24 DATA SEI,120,0,STA,129,1,STX,134,2,STY,132,2,
      TAX,170,0,TAY,168,0

26 DATA TSX,186,0,TXA,138,0,TXS,154,0,TYA,152,0

28 REM DATA STATEMENTS ABOVE FOR M$ (MNEMONICS),
      OP (OP CODES), & CA (CATEGORIES)

30 DIM M$(55),OP(55),CA(55):FOR I=0 TO 55:
     READ M$(I),OP(I),CA(I):NEXT I

35 REM CATEGORIES DETERMINE POSSIBLE ADDRESSING
     MODES

40 PRINT "BEGIN AT";:INPUT BE:AD=BE

45 REM BE IS BEGINNING ADDRESS OF ASSEMBLED CODE
     & AD IS CURRENT ADDRESS

50 PRINT "DATA FILE NAME";:INPUT NA$

55 REM ASSEMBLED CODE IS WRITTEN ONTO TAPE

60 OPEN 1,1,1,NA$

65 REM DATA FILE STARTS WITH BE AND ENDS WITH 999

70 PRINT #1,BE

100 PRINT:C$="":A$="":PRINT "? ";

101 REM GET ASSEMBLY LANGUAGE LINE--20=DEL--13=RETURN

102 GET A$:IF A$="" GOTO 102

103 PRINT A$;:L=LEN(C$)

104 IF ASC(A$)=20 AND L<2 THEN C$="":A$="":
      GOTO 102:REM DEL FIRST CHAR.

105 IF ASC(A$)=20 THEN A$="":C$=LEFT$(C$,L-1):
      GOTO 102:REM DEL A CHARACTER

106 IF ASC(A$)<>13 THEN C$=C$+A$:A$="":
      GOTO 102

108 L$=LEFT$(C$,3):IF L<=4 THEN C$="":L=0:
      GOTO 120

109 REM L$ HAS MNEMONIC, C$ HAS OPERAND

110 C$=RIGHT$(C$,L-4):L=L-4:REM L IS OPERAND
      LENGTH

120 MN=-1:REM MN IS MNEMONIC NUMBER--NOW SEARCH
     LIST

130 FOR I=0 TO 55:IF M$(I)=L$ THEN MN=I:
      I=55

140 NEXT I

145 IF L$="END" GOTO 2000

150 IF MN=-1 THEN PRINT "WHAT?": GOTO 100

155 REM MN=-1 MEANS MNEMONIC NOT FOUND

160 CA=CA(MN):OP=OP(MN):REM LOOK UP CATEGORY
      AND BASE OPCODE

200 IF CA=0 THEN POKE 997,OP:BY=1:GOTO 1000

205 REM OPCODES ARE STORED IN 997,998,999
      BEFORE WRITING TO TAPE

207 REM BY=INSTRUCTION LENGTH IN BYTES

208 REM 1000 WRITES TO TAPE

209 REM CA=0 ARE "IMPLIED"  INSTRUCTIONS

210 IF C$<>"A" GOTO 220

213 IF CA<> 3 THEN PRINT "NO":GOTO 100:
     REM CATEGORY 3 FOR ACCUMULATOR OPERAND
```