Below is just a brainstorm of me trying to come up with my own method to draw shapes.
: take the screen address that would be stored somewhere as a variable (SCRADD) ,copy the value to a temp variable (LINEVAR) , load and store each byte of the tables (x is offset of shape table) on that line with the y register being an offset: LDA (SHAPE,X) . STA (LINEVAR),Y. then when Y = the width of the shape, add $0400 to LINEVAR, reset y to 0, clear the screen and repeat. When you want to move the sprite, left or right, just add somthing to the SCRADD value and save it as the new SCRADD value. When moving up or down just subtract $0400 from the SCRADD value and save it. Here is psudo code to think it out. NOTE: this code wont run this is just to help me think (its not done yet)

```
        SCRADD EUQ 2000
        SWIDTH EQU 8
        SHEIGHT EQU 3
        LINEVAR EQU SCRADD
        LDX #$00
        LDY #$00
DRAW:
        LDA (SHAPE,X)
        STA (LINEVAR),Y
        INX
```

Below is my first attempt at making a sprite drawing script (following the book) filename;SPRTD, ddata

```
                ORG $800
                OBJ $800
        SHPL EPZ $1C
        SHPH EPZ SHPL+$1
        DEPTH DFS $1
        SLNGH DFS $1
        LNGH DFS $1
        HORIZ DFS $1
        TEMP DFS $1
        TVERT DFS $1
        HIRESL EPZ $1A
        HIRESH EPZ HIRESL+$1
START:
        JSR SSETUP
        JSR DRAW
        RTS
SSETUP:
        LDA #SHIP
        STA SHPL
        LDA /SHIP
```

```
        STA SHPH
        LDA #$08
        STA DEPTH
        LDA #$09
        STA HORIZ
        LDA #$03
        STA SLNGH
        STA TEMP
        RTS
DRAW:
        LDY TVERT
        JSR GETADR
        LDX #$00
        LDA TEMP
        STA SLNGH
DRAW2:
        LDA (SHPL,X)
        STA (HIRESL),Y
        INC SHPL
        INY
        DEC SLNGH
        BNE DRAW2
        INC TVERT
        DEC DEPTH
        BNE DRAW
        RTS
GETADR:
        LDA YVERTL,Y
        CLC
        ADC HORIZ
        STA HIRESL
        LDA YVERTH,Y
        STA HIRESH
        RTS
YVERTL:
        HEX 0000000000000000
        HEX 8080808080808080
        HEX 0000000000000000
        HEX 8080808080808080
        HEX 0000000000000000
        HEX 8080808080808080
        HEX 0000000000000000
        HEX 8080808080808080
```

HEX 2828282828282828
HEX A8A8A8A8A8A8A8A8
HEX 2828282828282828
HEX A8A8A8A8A8A8A8A8
HEX 2828282828282828
HEX A8A8A8A8A8A8A8A8
HEX 2828282828282828
HEX A8A8A8A8A8A8A8A8
HEX 5050505050505050
HEX D0D0D0D0D0D0D0D0
HEX 5050505050505050
HEX D0D0D0D0D0D0D0D0
HEX 5050505050505050
HEX D0D0D0D0D0D0D0D0
HEX 5050505050505050
HEX D0D0D0D0D0D0D0D0

YVERTH:

HEX 2024282C3034383C
HEX 2024282C3034383C
HEX 2125292D3135393D
HEX 2125292D3135393D
HEX 22262A2E32363A3E
HEX 22262A2E32363A3E
HEX 23272B2F33373B3F
HEX 23272B2F33373B3F
HEX 2024282C3034383C
HEX 2024282C3034383C
HEX 2125292D3135393D
HEX 2125292D3135393D
HEX 22262A2E32363A3E
HEX 22262A2E32363A3E
HEX 23272B2F33373B3F
HEX 23272B2F33373B3F
HEX 2024282C3034383C
HEX 2024282C3034383C
HEX 2125292D3135393D
HEX 2125292D3135393D
HEX 22262A2E32363A3E
HEX 22262A2E32363A3E
HEX 23272B2F33373B3F
HEX 23272B2F33373B3F

SHIP:

HEX 8000008200008200

```
HEX 008A0000AAD580AA
HEX 9582AAD58AA8D5AA
```

Below is some working code that lets the user move a pixel with the w,a,s, and d keys and Q exits the program

Working code

BTTNPLOT

```
JSR     $FB40 ;LORES GR
        LDY #05
        LDX #05
KEYIN   LDA #00
        JSR $FD1B
        CMP #$C1 ; A
        BEQ AMOV
        CMP #$C4 ; D
        BEQ DMOV
        CMP #$D3 ;S
        BEQ SMOV
        CMP #$D7 ;W
        BEQ WMOV
        CMP #$D1 ;Q
        BEQ EXIT
        JMP KEYIN
CSAVE JSR $FF4A ; save all registers
        JSR $F832 ; clear low res screen
        JSR $FF3F ; restore all registers
        LDA #44 ;COLOR15
        STA $0030
        RTS
AMOV JSR CSAVE
        DEY
        TXA
        JSR $F800
        JMP KEYIN
DMOV JSR CSAVE
```

```
        INY
        TXA
        JSR $F800
        JMP KEYIN
SMOV JSR CSAVE
        INX
        TXA
        JSR $F800
        JMP KEYIN
WMOV JSR CSAVE
        DEX
        TXA
        JSR $F800
        JMP KEYIN
EXIT    BRK
        END
################
```

Here is better version. This one, checks to see if either the X or Y registers are equal to 0 (The lowest plot value) or equal to 39 (The highest plot value). It does not check all of this every time, for example for AMOV, it only checks to see if the Y register is equal to 0 (the number in the Y register is the value that the plot routine uses for the x axis, 0 is on the left side and 39 is on the far right side). For SMOV, it only checks to see if the X register is equal to 39 (the number in the X register is the value that is transferred to the accumulator, the value in the accumulator is the value that the plot routine uses for the Y axis, 0 is on the top of the screen and 39 is on the bottom of the screen). To clarify the coordinate system of the low-res screen, (0,0) would be the top left of the screen, (39,0) would be the top right of the screen, (0,39) would be the bottom left of the screen, and (39,39) would be the bottom right of the screen.
PLOTBTTN

Better version

```
JSR     $FB40 ;LORES GR
        LDY #05
        LDX #05
KEYIN  LDA #00
        JSR $FD1B
        CMP #$C1 ; A
        BEQ AMOV
        CMP #$C4 ; D
        BEQ DMOV
        CMP #$D3 ;S
        BEQ SMOV
        CMP #$D7 ;W
        BEQ WMOV
        CMP #$D1 ;Q
```

```
        BEQ EXIT
        JMP KEYIN
CSAVE JSR $FF4A
        JSR $F832
        JSR $FF3F
        LDA #44 ;COLOR15
        STA $0030
        RTS
AMOV  CPY #$0
        BEQ KEYIN
        JSR CSAVE
        DEY
        TXA
        JSR $F800
        JMP KEYIN
DMOV CPY #$27
        BEQ KEYIN
        JSR CSAVE
        INY
        TXA
        JSR $F800
        JMP KEYIN
SMOV  CPX #$27
        BEQ KEYIN
        JSR CSAVE
        INX
        TXA
        JSR $F800
        JMP KEYIN
WMOV CPX #$0
        BEQ KEYIN
        JSR CSAVE
        DEX
        TXA
        JSR $F800
        JMP KEYIN
EXIT    BRK
        END
```

Below was my first draft that does not work

;Simple assembly program that lets the user move a pixel on the screen using W,A,S, and D keys. It is tested and does not work because CSAVE is before the KEYIN main game loop.

First draft

```
JSR     $FB40 ;LORES GR
        LDA #44 ;COLOR15
        LDY #05
        LDX #05
        LDA #05
CSAVE PHA
        TXA
        PHA
        TYA
        PHA
        JSR $F832
        TXA
        TAY
        PLA
        TAX
        PLA
        STA $0030
        RTS
KEYIN  LDA #0000
        JSR $FD1B
        CMP #$C1 ; A
        BEQ AMOV
        CMP #$C4 ; D
        BEQ DMOV
        CMP #$D3 ;S
        BEQ SMOV
        CMP #$D7 ;W
        BEQ WMOV
        CMP #$D1 ;Q
        BEQ EXIT
        BNE KEYIN
AMOV JSR CSAVE
        DEY
        TXA
```

```
        JSR $F800
        JMP KEYIN
DMOV JSR CSAVE
        INY
        TXA
        JSR $F800
        JMP KEYIN
SMOV JSR CSAVE
        INX
        TXA
        JSR $F800
        JMP KEYIN
WMOV JSR CSAVE
        DEX
        TXA
        JSR $F800
        JMP KEYIN
EXIT    BRK
        END
.
```