Basic Tweaks for PAWs games on DAAD

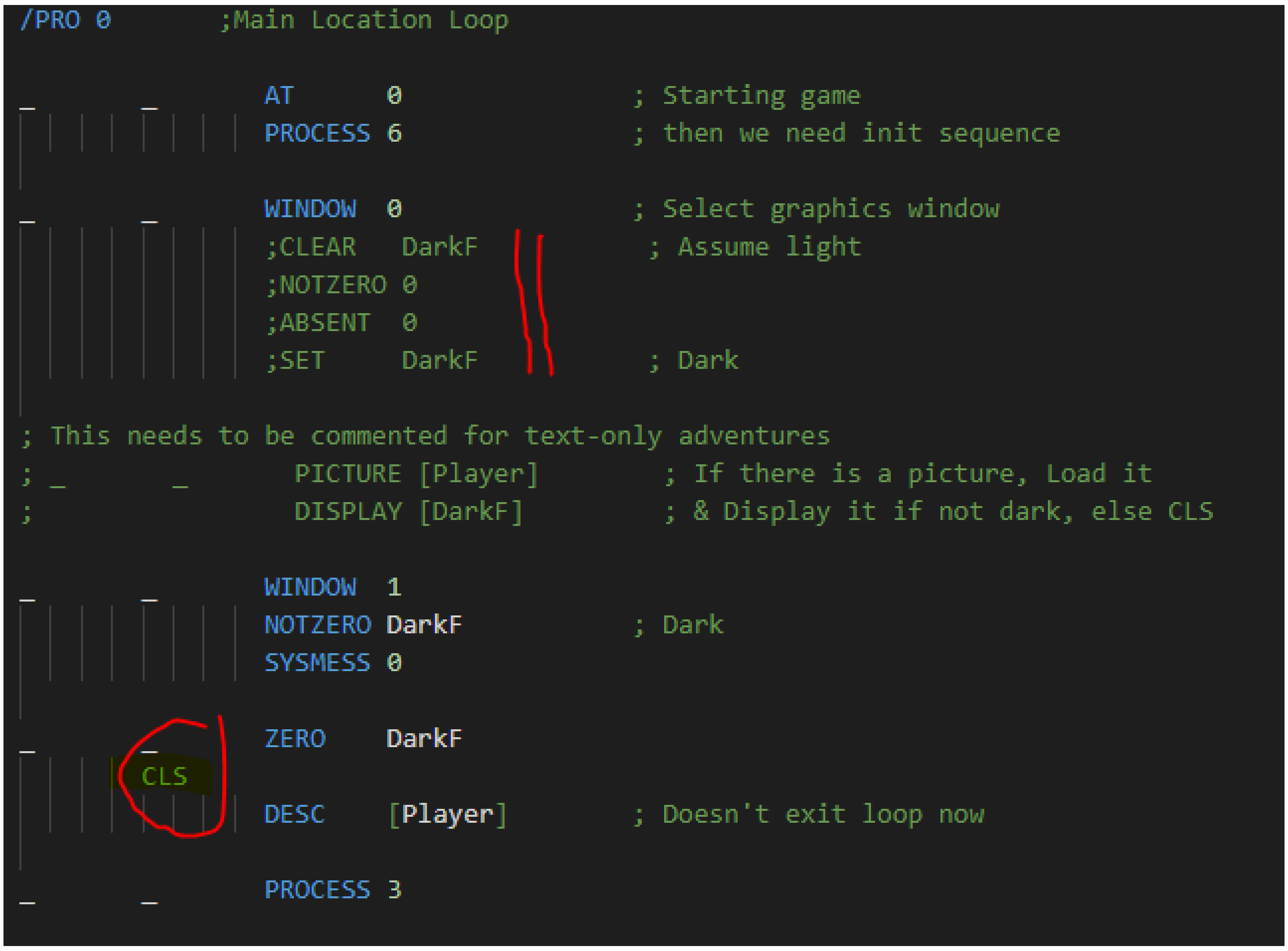
It will depend on your game and how you want things to work but here's the current changes I usually make to the DAAD code for PAWs games transcompiled via ANTUR.

(WORK IN PROGRESS FILE… This is the file I use as my own reference document but others have said they find it useful)

Process 0 changes

--- Get rid of default DAAD light/dark routine (if game doesn't use object 0 as a light source)

--- Depending on the original behaviour of the PAWs game, add a CLS, so that DAAD clears the screen on a "DESC" (which is now a RESTART condact in DAAD)

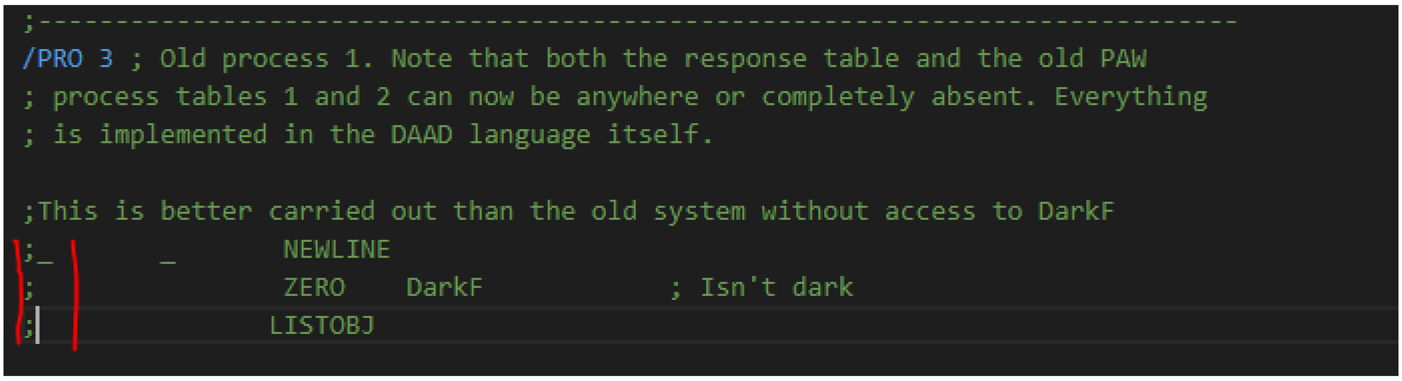


```
/PRO 0       ;Main Location Loop

_       _       AT      0               ; Starting game
                PROCESS 6               ; then we need init sequence

_       _       WINDOW  0               ; Select graphics window
                ;CLEAR   DarkF           ; Assume light
                ;NOTZERO 0
                ;ABSENT  0
                ;SET     DarkF           ; Dark

; This needs to be commented for text-only adventures
; _       _       PICTURE [Player]        ; If there is a picture, Load it
;                 DISPLAY [DarkF]         ; & Display it if not dark, else CLS

_       _       WINDOW  1
                NOTZERO DarkF           ; Dark
                SYSMESS 0

_       _       ZERO    DarkF
        CLS
                DESC    [Player]        ; Doesn't exit loop now

_       _       PROCESS 3
```

Process 3

--- comment out the standard DAAD LISOBJ routine… I'm going to leave it to my original PAWs code to display the objects in the appropriate place.

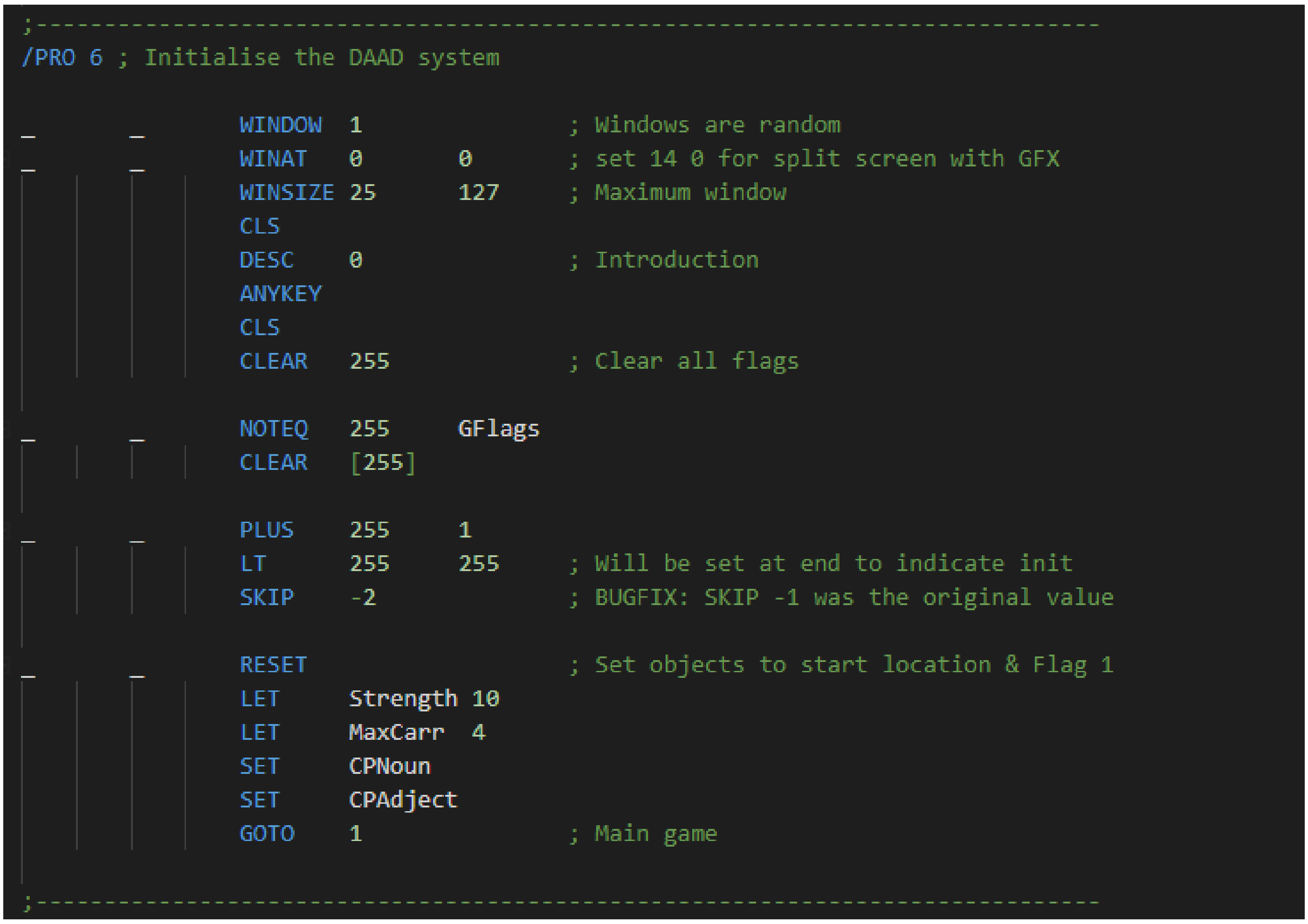
(If you don't choose one or the other then your game will list the objects twice)



```
;-----------------------------------------------------------------------------
/PRO 3 ; Old process 1. Note that both the response table and the old PAW
; process tables 1 and 2 can now be anywhere or completely absent. Everything
; is implemented in the DAAD language itself.

;This is better carried out than the old system without access to DarkF
;_       _        NEWLINE
;                 ZERO    DarkF            ; Isn't dark
;                LISTOBJ
```

Process 6

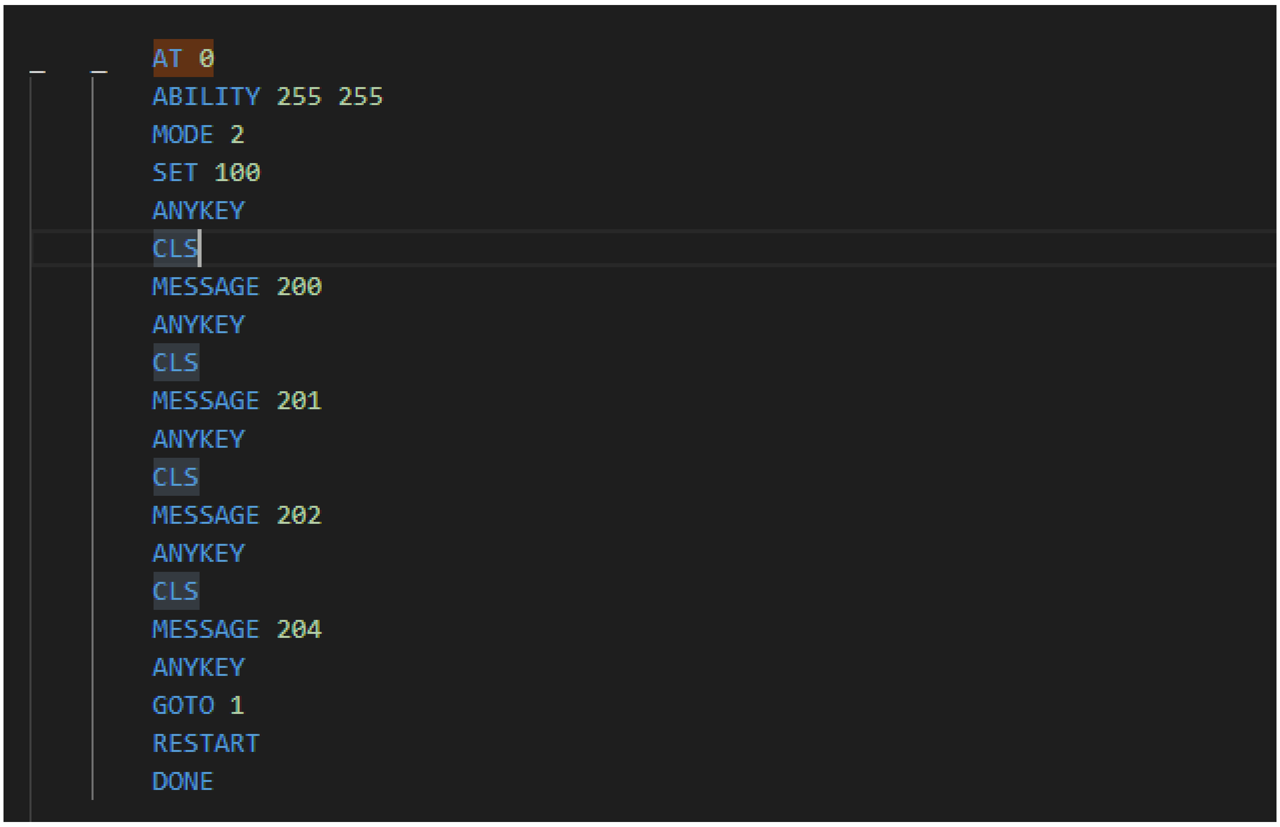
This is the process table that runs at the start of the game to initialise everything. Here is how it looks before I tweak things…



```
;-----------------------------------------------------------------------------
/PRO 6 ; Initialise the DAAD system

_       _       WINDOW  1               ; Windows are random
_       _       WINAT   0       0       ; set 14 0 for split screen with GFX
                WINSIZE 25      127     ; Maximum window
                CLS
                DESC    0               ; Introduction
                ANYKEY
                CLS
                CLEAR   255             ; Clear all flags

_       _       NOTEQ   255     GFlags
                CLEAR   [255]

_       _       PLUS    255     1
                LT      255     255     ; Will be set at end to indicate init
                SKIP    -2              ; BUGFIX: SKIP -1 was the original value

_       _       RESET                   ; Set objects to start location & Flag 1
                LET     Strength 10
                LET     MaxCarr  4
                SET     CPNoun
                SET     CPAdject
                GOTO    1               ; Main game

;-----------------------------------------------------------------------------
```

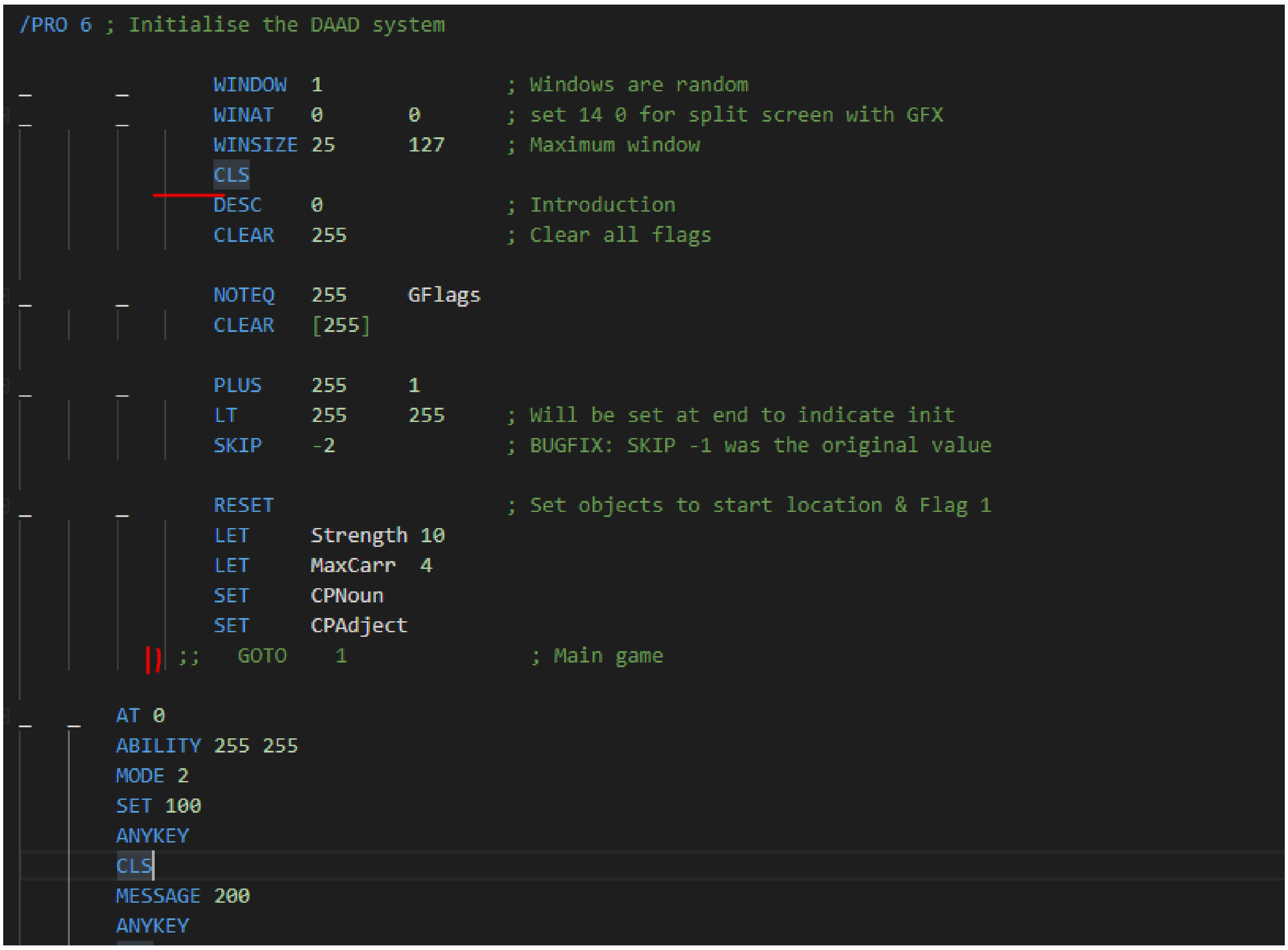
Note: This does not include any initialisation code from your PAWs game, which you'd usually put in process 1, and start with AT 0.

You can find this code in DAAD's Process 3. Here is my AT 0 code for this game…

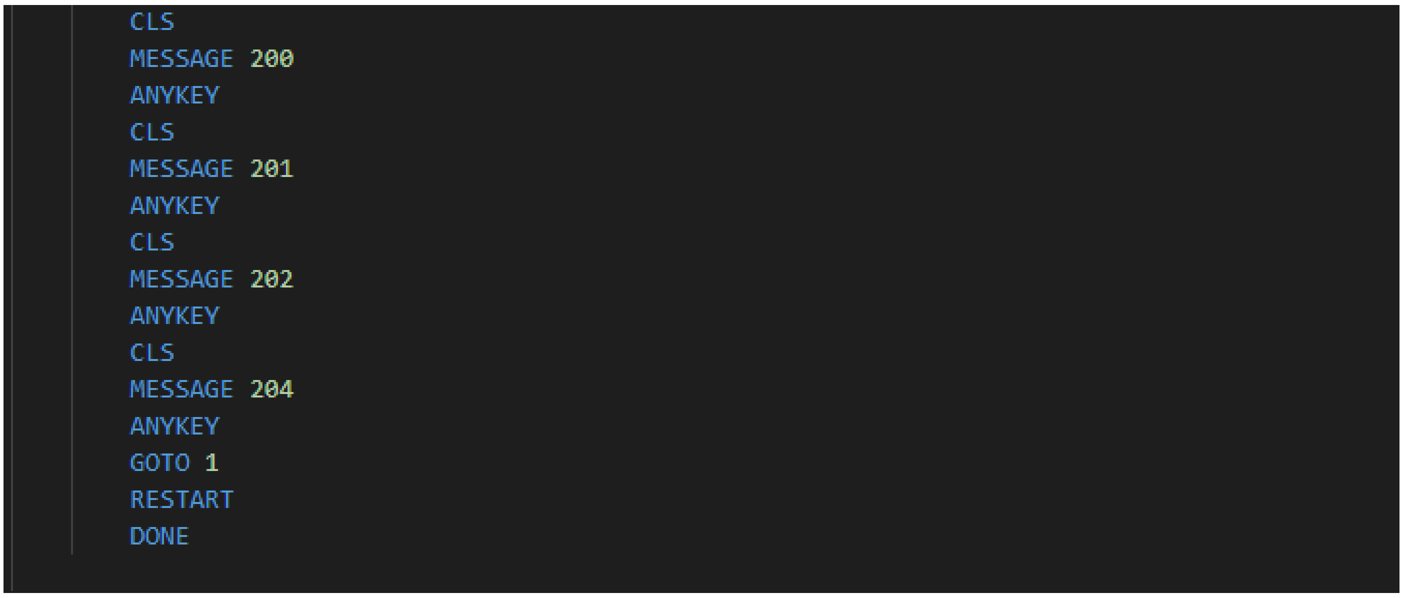


```
AT 0
ABILITY 255 255
MODE 2
SET 100
ANYKEY
CLS
MESSAGE 200
ANYKEY
CLS
MESSAGE 201
ANYKEY
CLS
MESSAGE 202
ANYKEY
CLS
MESSAGE 204
ANYKEY
GOTO 1
RESTART
DONE
```

I’m going to copy and paste this into a suitable part of Process 6 so that it runs at the start of the game, making sure I delete any unnecessary ANYKEYs and duplicate code. (I can leave this original code in Process 3 as it won’t usually be activated from there)



```
/PRO 6 ; Initialise the DAAD system

_       _       WINDOW  1               ; Windows are random
_       _       WINAT   0       0       ; set 14 0 for split screen with GFX
                WINSIZE 25      127     ; Maximum window
                CLS
                DESC    0               ; Introduction
                CLEAR   255             ; Clear all flags

_       _       NOTEQ   255     GFlags
                CLEAR   [255]

_       _       PLUS    255     1
                LT      255     255     ; Will be set at end to indicate init
                SKIP    -2              ; BUGFIX: SKIP -1 was the original value

_       _       RESET                   ; Set objects to start location & Flag 1
                LET     Strength 10
                LET     MaxCarr  4
                SET     CPNoun
                SET     CPAdject
              ;;   GOTO    1              ; Main game

_    _    AT 0
          ABILITY 255 255
          MODE 2
          SET 100
          ANYKEY
          CLS
          MESSAGE 200
          ANYKEY
```

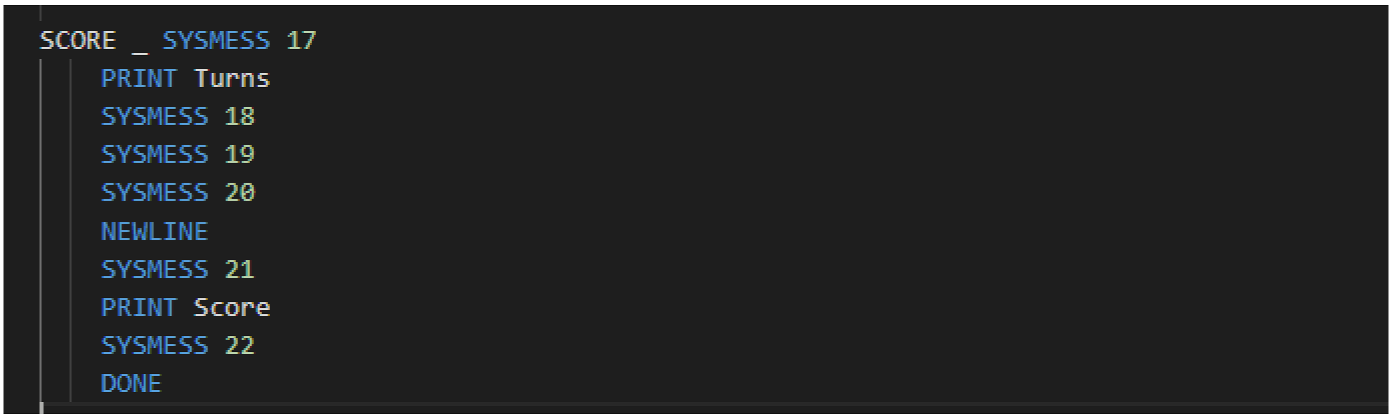

```
        CLS
        MESSAGE 200
        ANYKEY
        CLS
        MESSAGE 201
        ANYKEY
        CLS
        MESSAGE 202
        ANYKEY
        CLS
        MESSAGE 204
        ANYKEY
        GOTO 1
        RESTART
        DONE
```

Process 5

The old response table, or process 0. There are various tweaks you can make here.

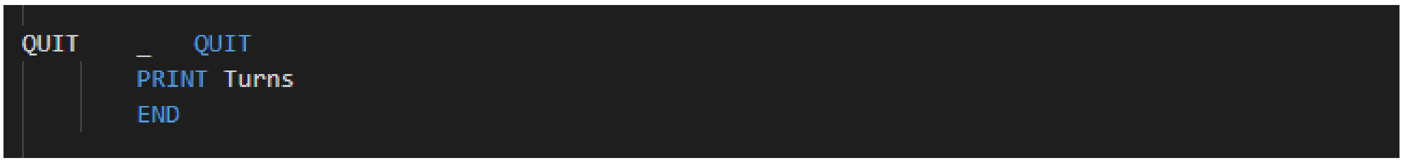
Score & Turns

This is what I tend to add to any place (either in this table or sub-processes) that uses the old PAWs SCORE condact (which doesn't behave the same way in DAAD). It's basically a handstitched "You have scored" + score + "%". "You have taken" + turns + "turn" +"s." response using the original PAWs system messages.


```
SCORE _ SYSMESS 17
        PRINT Turns
        SYSMESS 18
        SYSMESS 19
        SYSMESS 20
        NEWLINE
        SYSMESS 21
        PRINT Score
        SYSMESS 22
        DONE
```

You will probably want to add this to any "Game Over" situations & QUIT responses too, so search for END, score, or turns, in the source file to spot them, and paste the appropriate lines in.

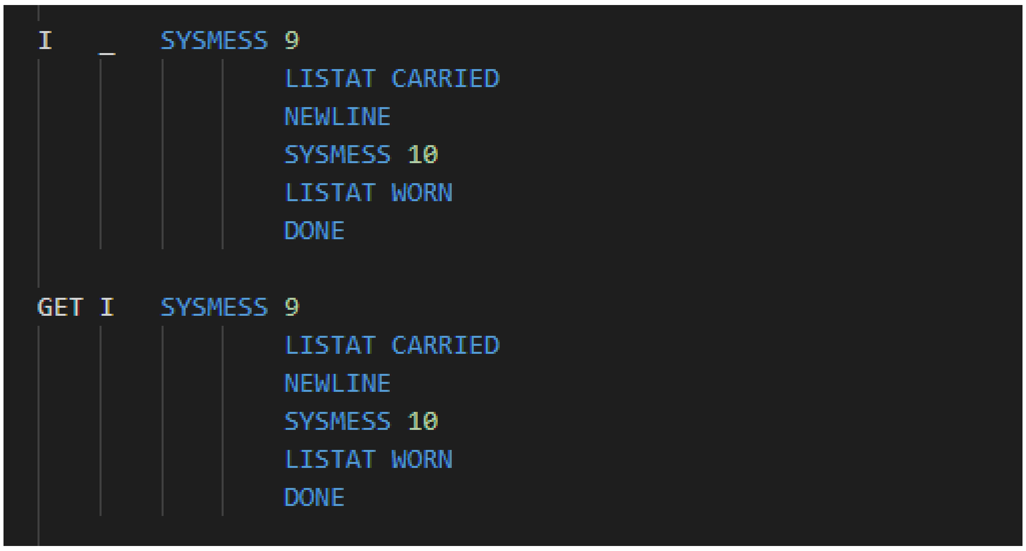
For example here…


```
QUIT    _   QUIT
        PRINT Turns
        END
```

Inventory

Unlike PAWs, DAAD does not have an INVEN condact.

The default Inventory routine, added by ANTUR first lists the objects carried and then the objects worn.



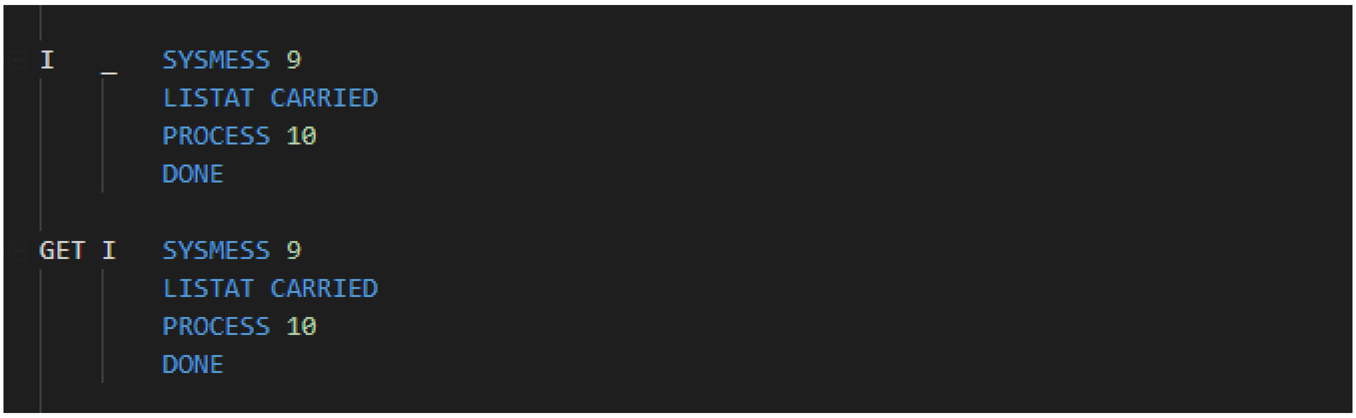
```
I    _    SYSMESS 9
                    LISTAT CARRIED
                    NEWLINE
                    SYSMESS 10
                    LISTAT WORN
                    DONE

GET I     SYSMESS 9
                    LISTAT CARRIED
                    NEWLINE
                    SYSMESS 10
                    LISTAT WORN
                    DONE
```

Note, that because this uses LISTAT, rather than INVEN, the standard system message printed for “nothing” is SYSMESS 53 rather than SYSMESS 11 (which the LISTOBJ routine and previously INVEN used).

If your game customised SYSMESS 53 for a specific container (for example to change it to say “empty”) then you may need to alter it to a more generic “nothing.” (tweaking the associated ‘In the basket you can see:’ message).

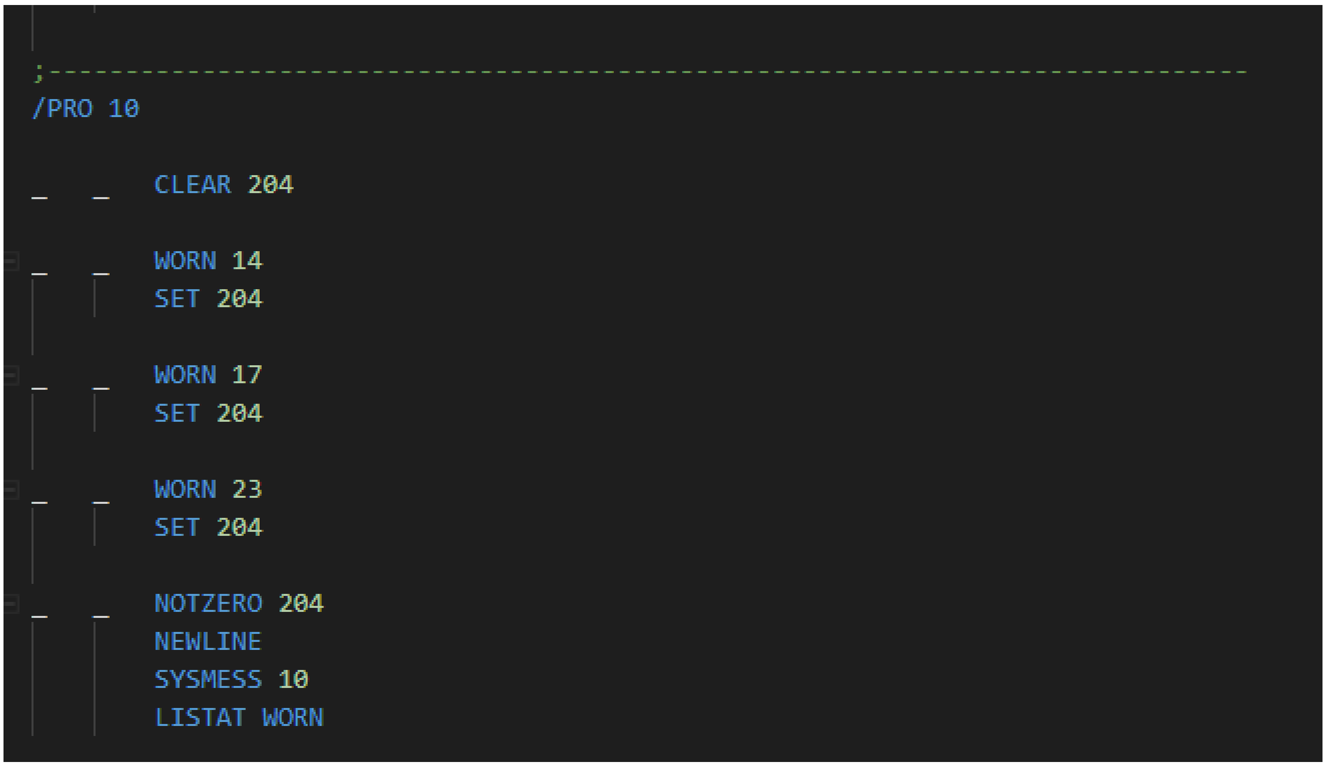
If you have no wearable objects, then you may wish to delete the SYSTEM 10 / LISTAT WORN lines.

Or perhaps you may only want to list worn objects if there are actually objects worn, in which case I would suggest a tweak like this…



```
I    _    SYSMESS 9
          LISTAT CARRIED
          PROCESS 10
          DONE

GET I     SYSMESS 9
          LISTAT CARRIED
          PROCESS 10
          DONE
```

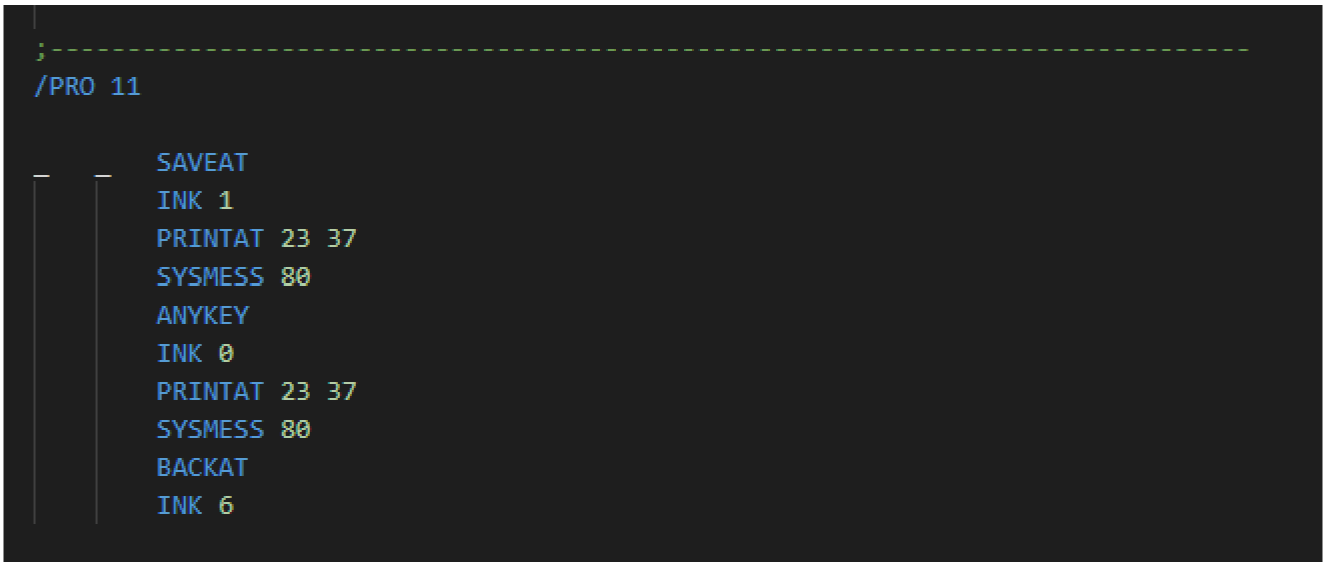
Where Process 10 is a process that quickly checks if any of the wearable items are actually worn and sets a flag (in this case 204) accordingly.



```
;-----------------------------------------------------------------------------
/PRO 10

_   _   CLEAR 204

_   _   WORN 14
        SET 204

_   _   WORN 17
        SET 204

_   _   WORN 23
        SET 204

_   _   NOTZERO 204
        NEWLINE
        SYSMESS 10
        LISTAT WORN
```

Anykey Routine

I don't like the standard "anykey" routine, so here's my current code for doing a PAWs-style anykey. It uses PRINTAT to place the prompt at the bottom right of the screen. Hopefully this will not obscure any text in your game. This code is formatted for the C64 screen size; you will need to adjust it for other target platforms.



```
;-----------------------------------------------------------------------------
/PRO 11

_   _   SAVEAT
        INK 1
        PRINTAT 23 37
        SYSMESS 80
        ANYKEY
        INK 0
        PRINTAT 23 37
        SYSMESS 80
        BACKAT
        INK 6
```

I've changed the standard anykey prompt (SM 16) to blank, so it's not printed. My alternative, sysmess 80 is the message '>>>'. If your anykey prompt is shorter or longer, you will need to adjust the positioning slightly.

Now, any ANYKEY condacts in my game are replaced by PROCESS 11 calls (or whatever number process table I ended up using).

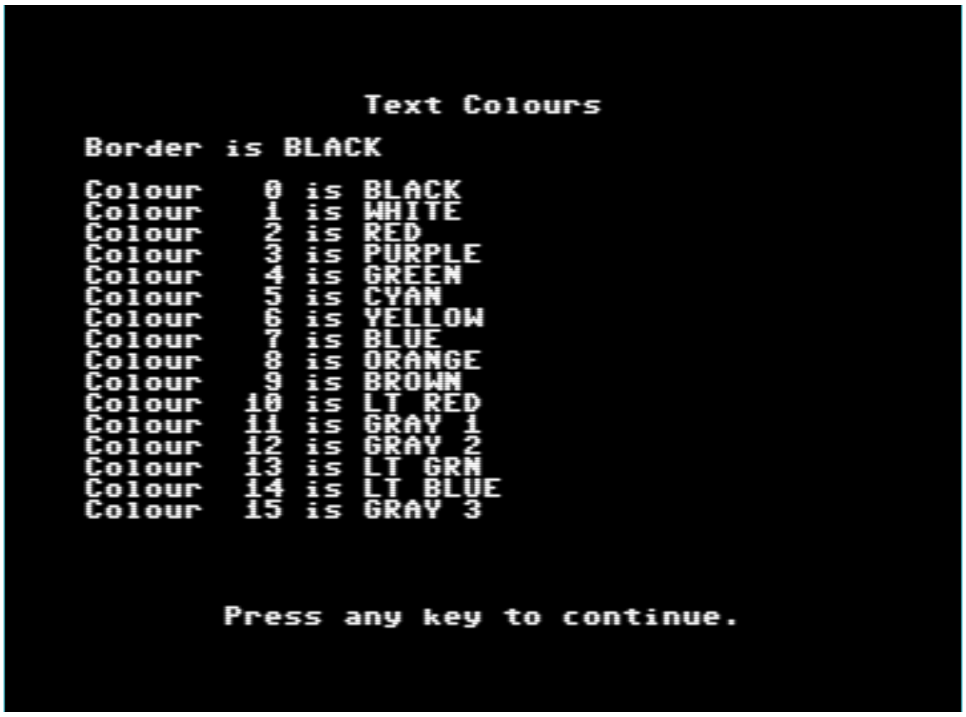
Additional Changes

(Again these are my own personal tweaks… I'm still exploring the options here, and some of these are for specific builds of the game, such as for C64 colour)

Adding Colour

Adding in basic colour… (C64 version)

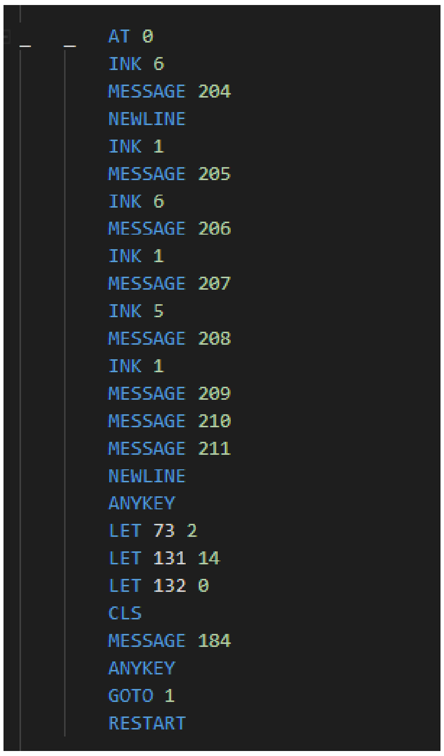
These are default C64 colours for the INK & PAPER condacts…



```
              Text Colours
   Border is BLACK

   Colour    0 is BLACK
   Colour    1 is WHITE
   Colour    2 is RED
   Colour    3 is PURPLE
   Colour    4 is GREEN
   Colour    5 is CYAN
   Colour    6 is YELLOW
   Colour    7 is BLUE
   Colour    8 is ORANGE
   Colour    9 is BROWN
   Colour   10 is LT RED
   Colour   11 is GRAY 1
   Colour   12 is GRAY 2
   Colour   13 is LT GRN
   Colour   14 is LT BLUE
   Colour   15 is GRAY 3


        Press any key to continue.
```

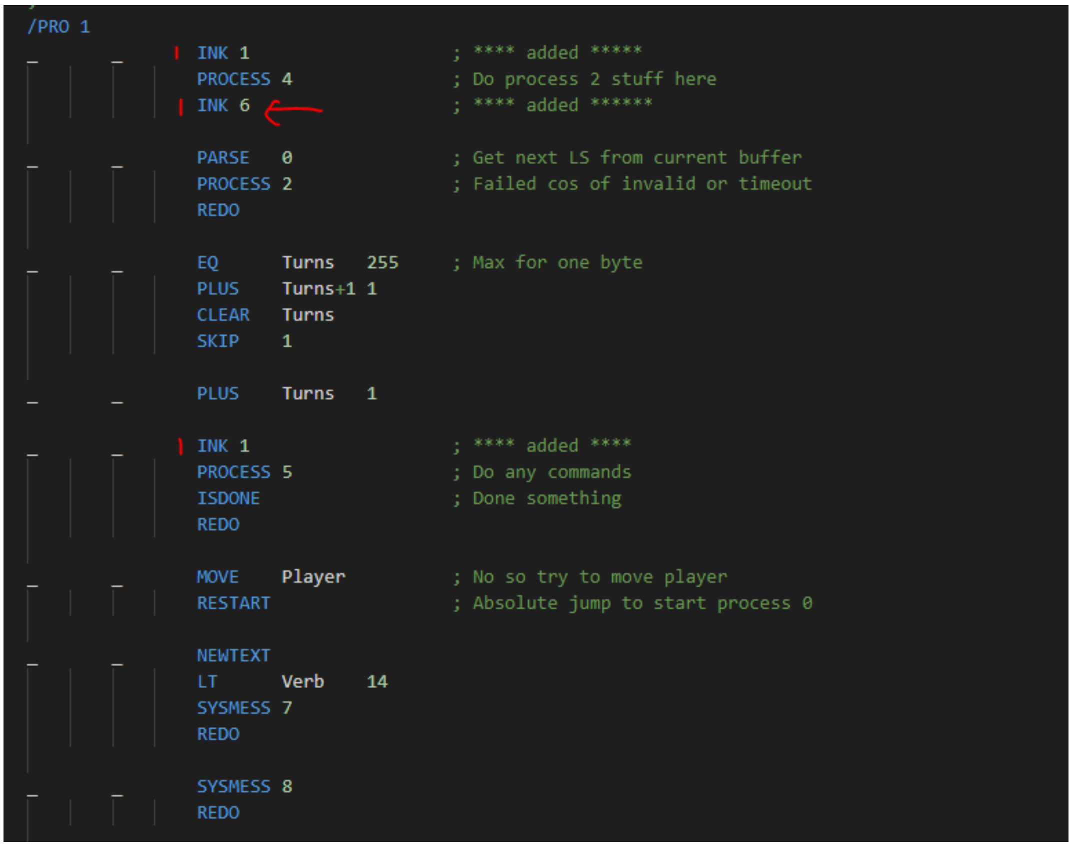
(These can be altered with the graphics editor DG program)

Set the INK before the message you want to change…



```
AT 0
INK 6
MESSAGE 204
NEWLINE
INK 1
MESSAGE 205
INK 6
MESSAGE 206
INK 1
MESSAGE 207
INK 5
MESSAGE 208
INK 1
MESSAGE 209
MESSAGE 210
MESSAGE 211
NEWLINE
ANYKEY
LET 73 2
LET 131 14
LET 132 0
CLS
MESSAGE 184
ANYKEY
GOTO 1
RESTART
```

What follows are my tweaks for changing the colour of the command & input lines…

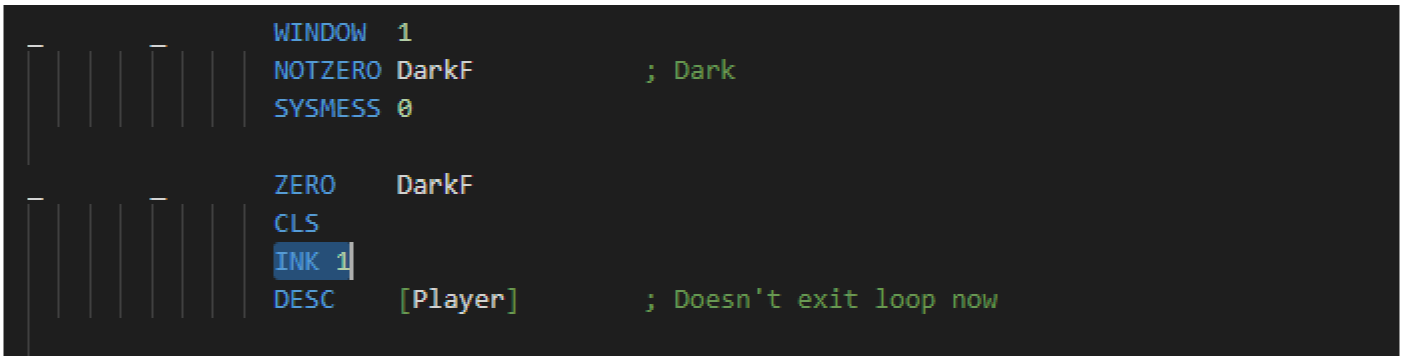


```
/PRO 1
_    _     INK 1              ; **** added *****
           PROCESS 4          ; Do process 2 stuff here
           INK 6              ; **** added ******

_    _     PARSE   0          ; Get next LS from current buffer
           PROCESS 2          ; Failed cos of invalid or timeout
           REDO

_    _     EQ      Turns   255  ; Max for one byte
           PLUS    Turns+1 1
           CLEAR   Turns
           SKIP    1

_    _     PLUS    Turns   1

_    _     INK 1              ; **** added ****
           PROCESS 5          ; Do any commands
           ISDONE             ; Done something
           REDO

_    _     MOVE    Player     ; No so try to move player
           RESTART            ; Absolute jump to start process 0

_    _     NEWTEXT
           LT      Verb    14
           SYSMESS 7
           REDO

_    _     SYSMESS 8
           REDO
```

Here INK 1 is my default text colour and INK 6 is the colour I've chosen for the command prompt.

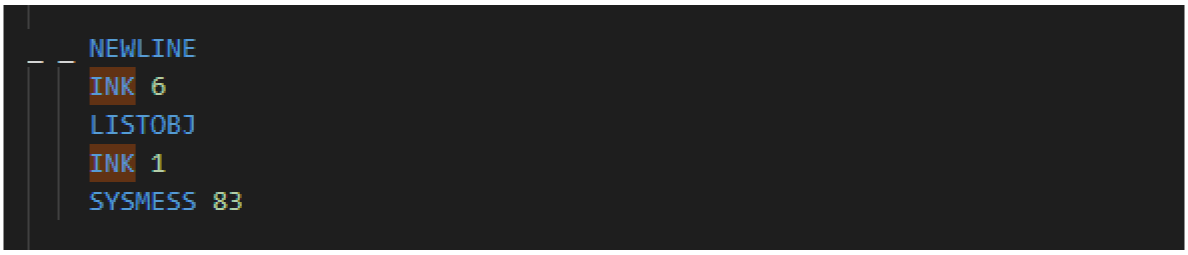
If you're going for a consistent colour scheme in the game you will probably need to add in a few INK commands elsewhere, such as before the QUIT condact, in order to match that prompt colour with your standard ones.

Perhaps, you want to colour the location text a different colour? In which case you'll need to amend process 0, adding an INK condact just before the location is described…



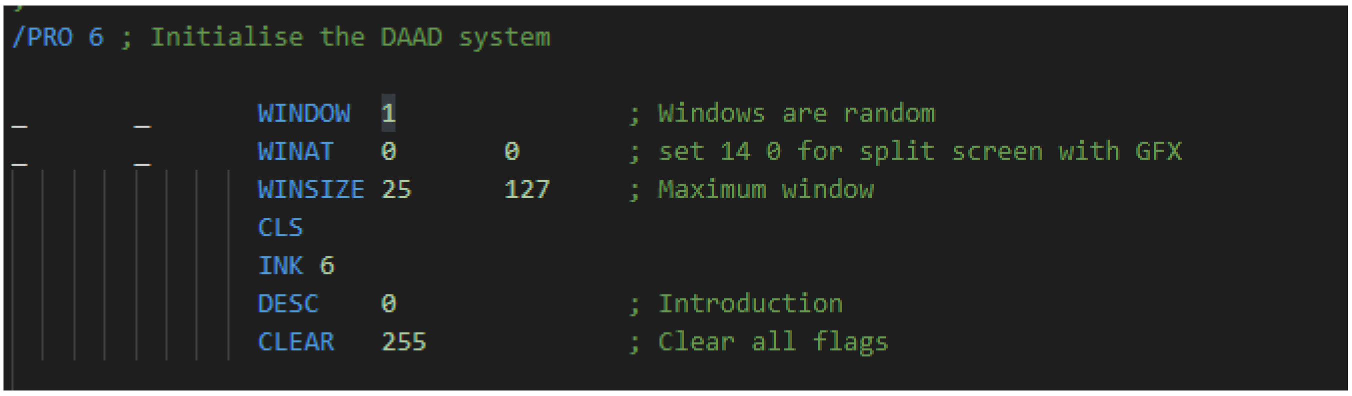
```
_    _     WINDOW   1
           NOTZERO DarkF            ; Dark
           SYSMESS 0

_    _     ZERO     DarkF
           CLS
           INK 1
           DESC     [Player]        ; Doesn't exit loop now
```

Perhaps you also want your list of objects visible to be in a different colour. If so, make the necessary changes in Process 3…



```
_ _ NEWLINE
    INK 6
    LISTOBJ
    INK 1
    SYSMESS 83
```

As a final change, you'll probably want to add one to the game initialisation process, so that you always start with the correct colour…



```
/PRO 6 ; Initialise the DAAD system

_       _       WINDOW  1               ; Windows are random
_       _       WINAT   0       0       ; set 14 0 for split screen with GFX
                WINSIZE 25      127     ; Maximum window
                CLS
                INK 6
                DESC    0               ; Introduction
                CLEAR   255             ; Clear all flags
```