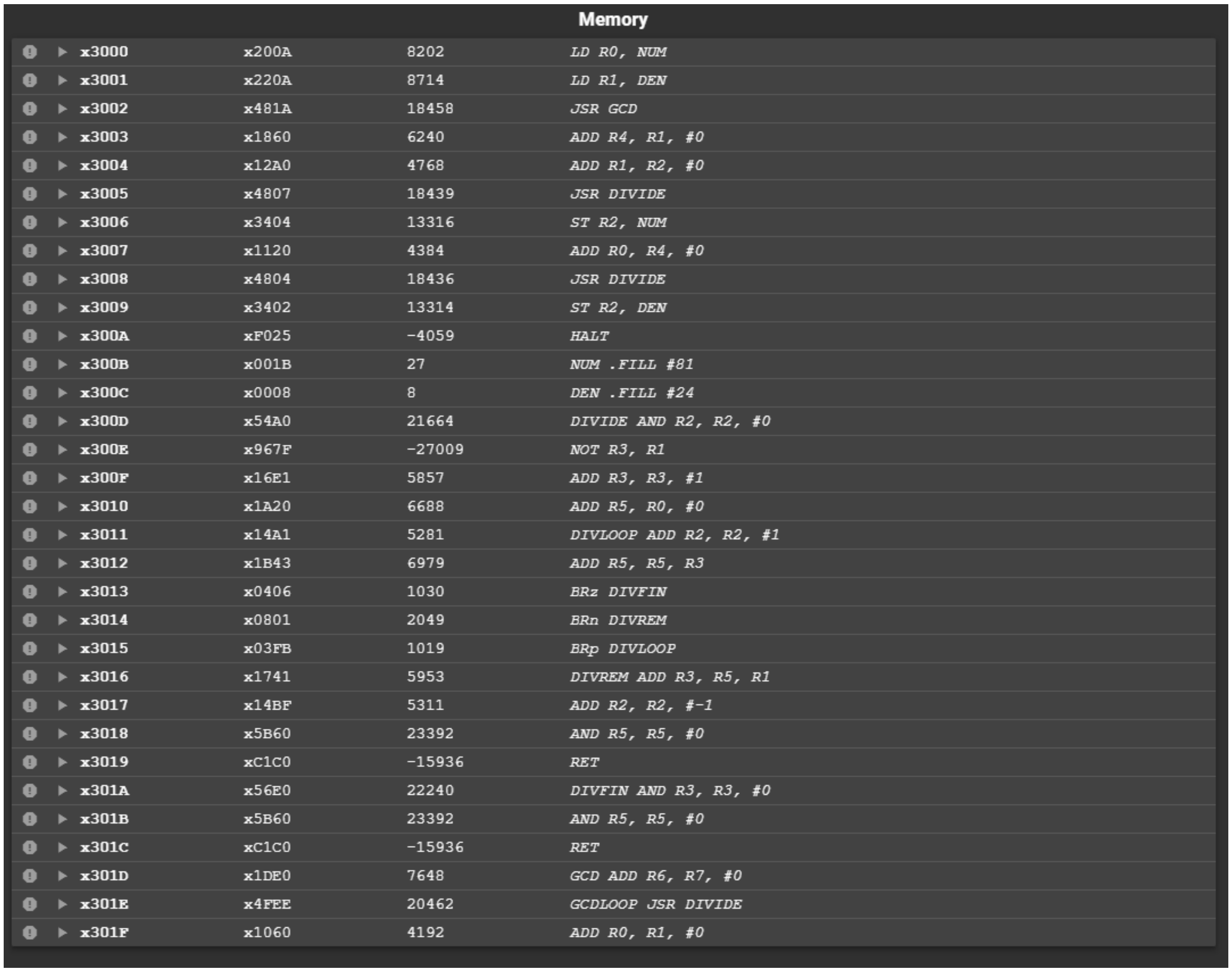
CS2253 Assignment 4 – Isaac Shoebottom

Program Listing

```
        .ORIG x3000

        LD  R0, NUM

        LD  R1, DEN

        JSR GCD

        ADD R4, R1, #0

        ADD R1, R2, #0

        JSR DIVIDE

        ST R2, NUM

        ADD R0, R4, #0

        JSR DIVIDE

        ST R2, DEN

        HALT

; you can try other values for NUM and DEN by replacing these values in the simulator

NUM     .FILL #81 ; you can try other values for NUM and DEN by replacing

DEN     .FILL #24


; Divide R0 by R1, putting quotient in R2 and remainder in R3

DIVIDE  AND R2, R2, #0  ;clean R2

        NOT R3, R1  ;store NOT of r1 in R3

        ADD R3, R3, #1  ;add one to make R3 the negative of R1

        ADD R5, R0, #0  ;store num in working variable

DIVLOOP ADD R2, R2, #1  ;first increment of quotient counter

        ADD R5, R5, R3  ;store working number in R4, subtracting the denominator from
the numerator

        BRz DIVFIN

        BRn DIVREM

        BRp DIVLOOP

DIVREM  ADD R3, R5, R1

        ADD R2, R2, #-1
        AND R5, R5, #0 ; clean r4

        RET

DIVFIN  AND R3, R3, #0  ;remainder is zero

        AND R5, R5, #0 ; clean r4

        RET


; Euclid's algorithm for GCD of R0 and R1, result in R2

GCD     ADD R6, R7, #0  ;make call stack work (store the ret value of the original
call)

GCDLOOP JSR DIVIDE

        ADD R0, R1, #0  ; R0 = R1

        ADD R1, R3, #0  ; R1 = R3

        BRp GCDLOOP

        ADD R2, R0, #0  ;result in R2

        ADD R7, R6, #0  ;make call stack work (load the original ret value to return
to original call location)

        LD  R0, NUM     ;load values back into r0 (bc im bad at programming)

        LD  R1, DEN     ;load values back into r1 (bc im bad at programming)

        RET

        .END
```

Sample Run

As you can see at x300B and X300C, the original fill values are replaced by their simplest form.



Memory

```
x3000    x200A    8202      LD R0, NUM
x3001    x220A    8714      LD R1, DEN
x3002    x481A    18458     JSR GCD
x3003    x1860    6240      ADD R4, R1, #0
x3004    x12A0    4768      ADD R1, R2, #0
x3005    x4807    18439     JSR DIVIDE
x3006    x3404    13316     ST R2, NUM
x3007    x1120    4384      ADD R0, R4, #0
x3008    x4804    18436     JSR DIVIDE
x3009    x3402    13314     ST R2, DEN
x300A    xF025    -4059     HALT
x300B    x001B    27        NUM .FILL #81
x300C    x0008    8         DEN .FILL #24
x300D    x54A0    21664     DIVIDE AND R2, R2, #0
x300E    x967F    -27009    NOT R3, R1
x300F    x16E1    5857      ADD R3, R3, #1
x3010    x1A20    6688      ADD R5, R0, #0
x3011    x14A1    5281      DIVLOOP ADD R2, R2, #1
x3012    x1B43    6979      ADD R5, R5, R3
x3013    x0406    1030      BRz DIVFIN
x3014    x0801    2049      BRn DIVREM
x3015    x03FB    1019      BRp DIVLOOP
x3016    x1741    5953      DIVREM ADD R3, R5, R1
x3017    x14BF    5311      ADD R2, R2, #-1
x3018    x5B60    23392     AND R5, R5, #0
x3019    xC1C0    -15936    RET
x301A    x56E0    22240     DIVFIN AND R3, R3, #0
x301B    x5B60    23392     AND R5, R5, #0
x301C    xC1C0    -15936    RET
x301D    x1DE0    7648      GCD ADD R6, R7, #0
x301E    x4FEE    20462     GCDLOOP JSR DIVIDE
x301F    x1060    4192      ADD R0, R1, #0
```