```
.ORIG x3000
; Author: Zane Wonsey
; Date: 12/09/2013
; Side note: used subroutines given in book
; Subroutines for carrying out the PUSH and POP functions. This
; program works with a stack consisting of memory locations x3FFF
; (BASE) through x3FFB (MAX). R6 is the stack pointer
                                 LD R6, POINTER
                         GETC
INPUT
                                 JSR PUSH
                                 GETC
                                 ADD R1, R0, 0
                                 GETC
                                 JSR DPUSH
                                 GETC
                                 JSR PUSH
 ____
                                 JSR PEEK
                                 OUT
                                 JSR DPOP
                                 OUT
                                 ADD R0, R1, 0
                                 OUT
                                 JSR DPOP
                                 OUT
                                 ADD R0, R1, 0
                                 OUT
                                 BR ENDPROGRAM
                         .fill 0
RETURN HERE
                                 R2, Save2 ; Saves registers that
                         ST
PEEK
                                         R1, Save1 ; are needed by POP.
                                 ST
                                                         ; BASE contains -x3FF
                                          R1, BASE
                                 LD
                                                        ; R1 contains -x4000
                                         R1, R1, #-1
                                 ADD
                                                         ; Compare stack point
                                         R2, R6, R1
                                 ADD
                                                        ; Branch if stack is
                                        fail_exit
R0, R6, #0
                                 BRz
                                                          ; The actual "pop"
                                 LDR
                                         success exit
                                 BR
                                 R2, Save2 ; Saves registers that
                         ST
 DPUSH
```

ST LD RO, Save1 ; are needed by PUSH.
RO, MAX ; MAX contains -x3FFB

ADD R2, R6, R0 ; Compare stack pointer to -x

```
BRz fail exit
                                                       ; Branch if stack is
                                      R6, R6, #-1 ; Adjust the stack po
R1, R6, #0 ; The actual "push"
                               ADD
                                     R1, R6, #0
                                STR
                               LD RO, Savel
                               ADD R2, R6, R1
                               BRz fail exit
                                       R6, R6, \#-1 ; Adjust the stack po
                               ADD
                                                      ; The actual "push"
                                       RO, R6, #0
                                STR
                               BR success exit
                               R1, Save1
                                           ; Saves registers that
                       ST
DPOP
                                       R2, Save2
                                                   ; are needed by POP.
                                ST
                                       R2, BASE
                                                      ; BASE contains -x3FF
                               LD
                                                      ; R1 contains -x4000
                                       R2, R2, #-1
                                ADD
                                       R3, R6, R2
                                                      ; Compare stack point
                                ADD
                                       fail exit
                                                       ; Branch if stack is
                                BRz
                                                      ; The actual "pop"
                                       RO, R6, #0
                                LDR
                                                       ; Adjust stack pointe
                                ADD
                                       R6, R6, #1
                                                       ; Compare stack point
                                       R3, R6, R2
                                ADD
                                                       ; Branch if stack is
                                BRz
                                       fail exit
                                                       ; The actual "pop"
                                        R1, R6, #0
                                LDR
                                        R1, Save1
                                ST
                                                       ; Adjust stack pointe
                                ADD
                                        R6, R6, #1
                                        success exit
                                BR
                                                       ; Saves registers tha
                                        R2, Save2
                                ST
POP
                                                       ; are needed by POP.
                                ST
                                        R1, Savel
                                                       ; BASE contains -x3FF
                                LD
                                        R1, BASE
                                                       ; R1 contains -x4000
                                        R1, R1, #-1
                                ADD
                                                       ; Compare stack point
                                       R2, R6, R1
                                ADD
                                                       ; Branch if stack is
                                       fail exit
                                BRz
                                                      ; The actual "pop"
                                       RO, R6, #0
                                LDR
                                       R6, R6, #1
                                                       ; Adjust stack pointe
                                ADD
                                BR
                                       success exit
                                R2, Save2 ; Saves registers that
                        ST
PUSH
                                      R1, Savel ; are needed by PUSH.
R1, MAX ; MAX contains -x3FFB
                                LD
                                ADD R2, R6, R1 ; Compare stack pointer to -x
                                       fail exit ; Branch if stack is
                                BRz
                                                       ; Adjust the stack po
                                       R6, R6, #-1
                                ADD
                                       RO, R6, #0 ; The actual "push"
                                STR
                                       ; Restore original
                      R1, Savel
success exit LD
                                       R2, Save2 ; register values.
                                LD
                                       R5, R5, #0
                                                       ; R5 <-- success
                                AND
                                RET
                                R1, Save1 ; Restore original
                       LD
fail exit
                                       R2, Save2 ; register values.
                                LD
                                AND
                                        R5, R5, #0
                                                       ;
                                       R5, R5, #1 ; R5 <-- failure
                                ADD
                                RET
```

ē .

ENDPROGRAM	HALT		
BASE MAX POINTER Save1 Save2	.FILL .FILL .FILL	xC001 .FILL x3FFF x0000 x0000	; BASE contains -x3FFF. xC005

.END