```
*
* RUN TIME!  gets command line args and calls main
*            uses r6 for return address
*            expects program value in r2
*
  0:     LD  3,1(0)   ; read command-line arg into MAIN's arg slot
  1:    LDA  6,1(7)   ; store return address
  2:    LDA  7,8(0)   ; branch to MAIN, at [r0]+8
  3:    OUT  2,0,0    ; print program value
  4:   HALT  0,0,0
*
* SQUARE     expects argument in r5, puts result in r4
*
  5:    MUL  5,5,5    ; compute result in place
  6:    ADD  4,0,5    ; store return value from r5 into r4
  7:    LDA  7,0(6)   ; return to address [r6]+0
*
* MAIN       expects argument in r3, puts result in r2
*
  8:    ADD  5,0,3    ; store parameter into SQUARE's arg slot
  9:    ST   6,2(0)   ; save return address in DMEM[2]
 10:    LDA  6,1(7)   ; store return address
 11:    LDA  7,5(0)   ; branch to SQUARE, at [r0]+5
 12:    ADD  2,0,4    ; copy SQUARE's return value into return slot
 13:    LD   7,2(0)   ; return to address in DMEM[ [r0]+2 ]
```

```
*
* RUN TIME!  gets command line args and calls main     CALLING SEQUENCE
*            uses r6 for return address                main    (caller)
*            expects program value in r2
*
  0:     LD  3,1(0)   ; read command-line arg into MAIN's arg slot
  1:    LDA  6,1(7)   ; store return address
  2:    LDA  7,8(0)   ; branch to MAIN, at [r0]+8
  3:    OUT  2,0,0    ; print program value
  4:   HALT  0,0,0
*
* SQUARE     expects argument in r5, puts result in r4
*
  5:    MUL  5,5,5    ; compute result in place
  6:    ADD  4,0,5    ; store return value from r5 into r4
  7:    LDA  7,0(6)   ; return to address [r6]+0
*
* MAIN       expects argument in r3, puts result in r2
*
  8:    ADD  5,0,3    ; store parameter into SQUARE's arg slot
  9:    ST   6,2(0)   ; save return address in DMEM[2]
 10:    LDA  6,1(7)   ; store return address
 11:    LDA  7,5(0)   ; branch to SQUARE, at [r0]+5
 12:    ADD  2,0,4    ; copy SQUARE's return value into return slot
 13:    LD   7,2(0)   ; return to address in DMEM[ [r0]+2 ]
```

```
*
* RUN TIME!  gets command line args and calls main     CALLING SEQUENCE
*            uses r6 for return address                main    (callee)
*            expects program value in r2
*
  0:     LD  3,1(0)   ; read command-line arg into MAIN's arg slot
  1:    LDA  6,1(7)   ; store return address
  2:    LDA  7,8(0)   ; branch to MAIN, at [r0]+8
  3:    OUT  2,0,0    ; print program value
  4:   HALT  0,0,0
*
* SQUARE     expects argument in r5, puts result in r4
*
  5:    MUL  5,5,5    ; compute result in place
  6:    ADD  4,0,5    ; store return value from r5 into r4
  7:    LDA  7,0(6)   ; return to address [r6]+0
*
* MAIN       expects argument in r3, puts result in r2
*
  8:    ADD  5,0,3    ; store parameter into SQUARE's arg slot
  9:    ST   6,2(0)   ; save return address in DMEM[2]
 10:    LDA  6,1(7)   ; store return address
 11:    LDA  7,5(0)   ; branch to SQUARE, at [r0]+5
 12:    ADD  2,0,4    ; copy SQUARE's return value into return slot
 13:    LD   7,2(0)   ; return to address in DMEM[ [r0]+2 ]
```

```
*
* RUN TIME!  gets command line args and calls main     CALLING SEQUENCE
*            uses r6 for return address                square  (caller)
*            expects program value in r2
*
  0:     LD  3,1(0)   ; read command-line arg into MAIN's arg slot
  1:    LDA  6,1(7)   ; store return address
  2:    LDA  7,8(0)   ; branch to MAIN, at [r0]+8
  3:    OUT  2,0,0    ; print program value
  4:   HALT  0,0,0
*
* SQUARE     expects argument in r5, puts result in r4
*
  5:    MUL  5,5,5    ; compute result in place
  6:    ADD  4,0,5    ; store return value from r5 into r4
  7:    LDA  7,0(6)   ; return to address [r6]+0
*
* MAIN       expects argument in r3, puts result in r2
*
  8:    ADD  5,0,3    ; store parameter into SQUARE's arg slot
  9:    ST   6,2(0)   ; save return address in DMEM[2]
 10:    LDA  6,1(7)   ; store return address
 11:    LDA  7,5(0)   ; branch to SQUARE, at [r0]+5
 12:    ADD  2,0,4    ; copy SQUARE's return value into return slot
 13:    LD   7,2(0)   ; return to address in DMEM[ [r0]+2 ]
```

```
*
* RUN TIME!  gets command line args and calls main     CALLING SEQUENCE
*            uses r6 for return address                square  (callee)
*            expects program value in r2
*
  0:     LD  3,1(0)   ; read command-line arg into MAIN's arg slot
  1:    LDA  6,1(7)   ; store return address
  2:    LDA  7,8(0)   ; branch to MAIN, at [r0]+8
  3:    OUT  2,0,0    ; print program value
  4:   HALT  0,0,0
*
* SQUARE     expects argument in r5, puts result in r4
*
  5:    MUL  5,5,5    ; compute result in place
  6:    ADD  4,0,5    ; store return value from r5 into r4
  7:    LDA  7,0(6)   ; return to address [r6]+0
*
* MAIN       expects argument in r3, puts result in r2
*
  8:    ADD  5,0,3    ; store parameter into SQUARE's arg slot
  9:    ST   6,2(0)   ; save return address in DMEM[2]
 10:    LDA  6,1(7)   ; store return address
 11:    LDA  7,5(0)   ; branch to SQUARE, at [r0]+5
 12:    ADD  2,0,4    ; copy SQUARE's return value into return slot
 13:    LD   7,2(0)   ; return to address in DMEM[ [r0]+2 ]
```

```
*
* RUN TIME!  gets command line args and calls main     RETURN  SEQUENCE
*            uses r6 for return address                square  (callee)
*            expects program value in r2
*
  0:     LD  3,1(0)   ; read command-line arg into MAIN's arg slot
  1:    LDA  6,1(7)   ; store return address
  2:    LDA  7,8(0)   ; branch to MAIN, at [r0]+8
  3:    OUT  2,0,0    ; print program value
  4:   HALT  0,0,0
*
* SQUARE     expects argument in r5, puts result in r4
*
  5:    MUL  5,5,5    ; compute result in place
  6:    ADD  4,0,5    ; store return value from r5 into r4
  7:    LDA  7,0(6)   ; return to address [r6]+0
*
* MAIN       expects argument in r3, puts result in r2
*
  8:    ADD  5,0,3    ; store parameter into SQUARE's arg slot
  9:    ST   6,2(0)   ; save return address in DMEM[2]
 10:    LDA  6,1(7)   ; store return address
 11:    LDA  7,5(0)   ; branch to SQUARE, at [r0]+5
 12:    ADD  2,0,4    ; copy SQUARE's return value into return slot
 13:    LD   7,2(0)   ; return to address in DMEM[ [r0]+2 ]
```

```
*
* RUN TIME!  gets command line args and calls main     RETURN  SEQUENCE
*            uses r6 for return address                square  (caller)
*            expects program value in r2
*
  0:     LD  3,1(0)   ; read command-line arg into MAIN's arg slot
  1:    LDA  6,1(7)   ; store return address
  2:    LDA  7,8(0)   ; branch to MAIN, at [r0]+8
  3:    OUT  2,0,0    ; print program value
  4:   HALT  0,0,0
*
* SQUARE     expects argument in r5, puts result in r4
*
  5:    MUL  5,5,5    ; compute result in place
  6:    ADD  4,0,5    ; store return value from r5 into r4
  7:    LDA  7,0(6)   ; return to address [r6]+0
*
* MAIN       expects argument in r3, puts result in r2
*
  8:    ADD  5,0,3    ; store parameter into SQUARE's arg slot
  9:    ST   6,2(0)   ; save return address in DMEM[2]
 10:    LDA  6,1(7)   ; store return address
 11:    LDA  7,5(0)   ; branch to SQUARE, at [r0]+5
 12:    ADD  2,0,4    ; copy SQUARE's return value into return slot
 13:    LD   7,2(0)   ; return to address in DMEM[ [r0]+2 ]
```

```
*
* RUN TIME!  gets command line args and calls main     RETURN  SEQUENCE
*            uses r6 for return address                main    (callee)
*            expects program value in r2
*
  0:     LD  3,1(0)   ; read command-line arg into MAIN's arg slot
  1:    LDA  6,1(7)   ; store return address
  2:    LDA  7,8(0)   ; branch to MAIN, at [r0]+8
  3:    OUT  2,0,0    ; print program value
  4:   HALT  0,0,0
*
* SQUARE     expects argument in r5, puts result in r4
*
  5:    MUL  5,5,5    ; compute result in place
  6:    ADD  4,0,5    ; store return value from r5 into r4
  7:    LDA  7,0(6)   ; return to address [r6]+0
*
* MAIN       expects argument in r3, puts result in r2
*
  8:    ADD  5,0,3    ; store parameter into SQUARE's arg slot
  9:    ST   6,2(0)   ; save return address in DMEM[2]
 10:    LDA  6,1(7)   ; store return address
 11:    LDA  7,5(0)   ; branch to SQUARE, at [r0]+5
 12:    ADD  2,0,4    ; copy SQUARE's return value into return slot
 13:    LD   7,2(0)   ; return to address in DMEM[ [r0]+2 ]
```

```
*
* RUN TIME!  gets command line args and calls main     RETURN  SEQUENCE
*            uses r6 for return address                main    (caller)
*            expects program value in r2
*
  0:     LD  3,1(0)   ; read command-line arg into MAIN's arg slot
  1:    LDA  6,1(7)   ; store return address
  2:    LDA  7,8(0)   ; branch to MAIN, at [r0]+8
  3:    OUT  2,0,0    ; print program value
  4:   HALT  0,0,0
*
* SQUARE     expects argument in r5, puts result in r4
*
  5:    MUL  5,5,5    ; compute result in place
  6:    ADD  4,0,5    ; store return value from r5 into r4
  7:    LDA  7,0(6)   ; return to address [r6]+0
*
* MAIN       expects argument in r3, puts result in r2
*
  8:    ADD  5,0,3    ; store parameter into SQUARE's arg slot
  9:    ST   6,2(0)   ; save return address in DMEM[2]
 10:    LDA  6,1(7)   ; store return address
 11:    LDA  7,5(0)   ; branch to SQUARE, at [r0]+5
 12:    ADD  2,0,4    ; copy SQUARE's return value into return slot
 13:    LD   7,2(0)   ; return to address in DMEM[ [r0]+2 ]
```