```
        INSP 1                  ; clear stack on fib return
P2:     PUSH                    ; put return AC (fib(n)) on stack
        LODD faddr:             ; load a pointer to result f[n]
        POPI                    ; pop result off stack into f[n]
        ADDD c1:
        STOD faddr:             ; inc, store pointer for next f[n]
        JUMP LOOP:              ; go to top for next pass
FIB:    LODL 1                  ; fib func loads arg from stack
        JZER FIBZER:            ; if fib(0) go to FIBZER
        SUBD c1:                ; dec arg value in AC (arg-1)
        JZER FIBONE:            ; if fib(1) go to FIBONE
        STOD LpCnt:             ; number of iterations in LpCnt
        LODD c0:                ; load a 0 into the AC
        STOD fm2:               ; store 0 in fib(n-2)
        LODD c1:                ; load a 1 into the AC
        STOD fm1:               ; store 1 in fib(n-1)
ITER:   LODD LpCnt:             ; LpCnt arg - 1 iterations needed
        JZER RTN:               ; when LpCnt == 0 goto RTN:
        SUBD c1:                ; dec arg value in AC (LpCnt-1)
        STOD LpCnt:             ; store LpCnt for next iteration
        LODD fm2:               ; arg must be >= 2, fm2 initially fib(0)
        ADDD fm1:               ; fm1 initially fib(1), so AC = fm2+fm1
        STOD tmp:               ; store this AC to tmp:
        LODD fm1:               ; now load AC with fib(n-1)
        STOD fm2:               ; replace old fib(n-2) with AC
        LODD tmp:               ; load AC with tmp: becomes fib(n-1)
        STOD fm1:               ; store AC as next fib(n-1) to fm1
        JUMP ITER:              ; jump to next iteration
RTN:    LODD tmp:               ; load AC with tmp: .. final result
        RETN
FIBZER: LODD c0:
        RETN                    ; AC = 0 for fib(0)
FIBONE: LODD c1:
        RETN                    ; AC = 1 for fib(1)
DONE:   HALT
.LOC    100                     ; locate data beginning at 100
d0:     3                       ; array of args for fib function
        9
        18
        23
        25
f0:     0                       ; array of result locs for fib returns
        0
        0
        0
        0
daddr:  d0:                     ; start address of fib args
faddr:  f0:                     ; start address of fib results
c0:     0                       ; constants
c1:     1
PasCnt: 5                       ; number of data elements to process
LpCnt:  0                       ; number of fib iterations
tmp:    0                       ; initial value for fib(2)
fm1:    0                       ; at any point fib(n-1)
fm2:    0                       ; at any point fib(n-2)
```