```
1  program qcom
2
3  !     pgf90 -c print.f
4  !     pgf90 -o qcom QCOM.f90 print.o
5  !     ./qcom
6
7  !     gfortran -c print.f
8  !     gfortran -o qcom QCOM.f90 print.o
9  !     ./qcom > out.dat
10
11 !     v. 2 (Sept 2000) --
12 !     Array subscript ranges now start with 0 instead of 1.
13
14 !     v. 3 (Feb 2009) --
15 !     All comments now start with !
16 !     Added f90 declarations and procedures.
17
18 !     v. 3.1 (Feb 2017) --
19 !     Runs and produces output (a snapshot) using PRINT.
20
21 !     v. 3.2 (Feb 2017) --
22 !     Stores t-z array, writes array using PRINT, and to a file.
23
24 !      implicit none
25
26       parameter (jt = 20, kt = 20, np_max = 100)
27       parameter (jv = jt, kv = kt)
28       real, dimension (0:jv+1, 0:kv+1) :: v
29       real, dimension (1:jv, 1:kv, 2) :: fv
30       real, dimension (1:np_max, 0:kv+1) :: v_tz
31
32 !     Assign parameters for run
33
34          ITTMAX = 100
35          DT = 1.
36
37          ns = 20 ! time interval (s) for output
38          np = ns/dt ! time step interval for output
39          ip = 0
40
41 !     Initialize all values of arrays for v, w, theta, pi, [fv, fw, ftheta, fpi,]
42 !     including values for boundary points for v, w, theta, pi.
43
44       CALL INIT  ! initialize all variables
45
46       ITT = 1 ! itt is time step index
47
48 !     USE FORWARD SCHEME TO do first step
```

```
49
50      A = 1.
51      B = 0.
52      N1 = MOD ( ITT    , 2 ) + 1
53      N2 = MOD ( ITT - 1, 2 ) + 1
54
55      CALL STEP ( N1, N2, A, B ) ! do first time step
56
57  !   ADAMS - BASHFORTH TWO - LEVEL SCHEME
58
59      A =   3. / 2.
60      B = - 1. / 2.
61
62      ITTNOW = ITT  + 1
63
64      DO ITT = ITTNOW, ITTMAX
65
66      N1 = MOD ( ITT    , 2 ) + 1
67      N2 = MOD ( ITT - 1, 2 ) + 1
68
69      CALL STEP ( N1, N2, A, B ) ! do subsequent time steps
70
71      if ( mod (itt, np) .eq. 0 ) then
72
73      ip = ip + 1
74      v_tz(ip,:) = v(1,:)
75
76      endif
77
78      end do
79
80  !   END-OF-RUN OUTPUT ROUTINES GO HERE
81
82  !    call print( V, jv+2, 1, jv+2, 1, kv+2, 'v' )
83
84      call print( v_tz, np_max, 1, ip, 1, kv+2, 'v(t,z)' )
85
86  ! --------- WRITE TO A FILE --------------
87      open(unit=10,file='v_tz.dat')
88      do k=0,kv+1
89        write(10,*) v_tz(1:ip,k)
90      end do
91      close (10)
92
93  contains
94
95      SUBROUTINE STEP ( N1, N2, A, B )
96
```

```
97  !     This is the entire subroutine.
98
99        CALL RCALC ( N2 ) ! calculate forcing terms from variables at current time
100       CALL AB ( N1, N2, A, B ) ! update variables using a time scheme
101       CALL BOUND ! apply boundary conditions to variables
102 !
103       END SUBROUTINE STEP
104
105       SUBROUTINE RCALC (  N2 )
106
107 !     CALCULATES FORCING TERMS FOR V(J,K), ETC.; STORES THEM  IN FV(J,K,N2), ETC.
108
109       DO K = 1, KT
110       DO J = 1, JT
111       FV(J,K,N2)  = 1.
112       END DO
113       END DO
114
115 !     ETC (forcing for w, theta, and pi)
116
117       END SUBROUTINE RCALC
118
119       SUBROUTINE AB ( N1, N2, A, B )
120
121 !     THE FOLLOWING LOOP UPDATES V USING EITHER THE FORWARD OR THE ADAMS-BASHFORTH
122 !     SCHEME DEPENDING ON THE VALUES OF A, B.
123 !     SUBSCRIPT N2 OF FV ALWAYS REFERS TO THE MOST RECENTLY CALCULATED VALUES FOR FV.
124
125       DO K = 1, KT
126       DO J = 1, JT
127       V(J,K)  = V(J,K)  + DT * ( A * FV(J,K,N2)  + B * FV(J,K,N1) )
128       END DO
129       END DO
130
131 !     ETC (update w, theta, and pi)
132
133       END SUBROUTINE AB
134
135       SUBROUTINE BOUND
136
137 !     apply boundary conditions for v
138
139       do j = 1, jt
140       v(j,0) = v(j,1) ! free slip b.c.
141       v(j,kt+1) = v(j,kt) ! free slip b.c.
142       end do
143
144       do k = 0, kt+1
```

```
145       v(0,k) = v(jt,k) ! periodic b.c.
146       v(jt+1,k) = v(1,k) ! periodic b.c.
147       end do
148
149   !   ETC (apply b.c. for w, theta, and pi)
150
151       END SUBROUTINE BOUND
152
153       SUBROUTINE INIT
154
155   !   initialize all variables
156
157       DO K = 1, KT
158       DO J = 1, JT
159       V(J,K)  = 0.
160       END DO
161       END DO
162
163       CALL BOUND
164
165       END SUBROUTINE INIT
166
167   end program qcom
168
```