```
PROGRAM LID_DRIVEN_CAVITY
  IMPLICIT NONE
  INTEGER, PARAMETER :: N = 50   ! GRID SIZE (NXN GRID)
  REAL :: DX, DY, DT, RE        ! GRID SPACING, TIME STEP, REYNOLDS NUMBER
  REAL :: U(N, N), V(N, N), P(N, N)  ! VELOCITY AND PRESSURE FIELDS
  INTEGER :: I, J, STEP
  REAL :: START_TIME, END_TIME, ELAPSED_TIME

  ! PARAMETERS
  DX = 1.0 / (N-1)   ! GRID SPACING IN X DIRECTION
  DY = 1.0 / (N-1)   ! GRID SPACING IN Y DIRECTION
  DT = 0.001         ! TIME STEP SIZE
  RE = 100           ! REYNOLDS NUMBER

  ! INITIALIZE ARRAYS
  U = 0.0
  V = 0.0
  P = 0.0

  ! INITIALIZE THE TOP BOUNDARY (LID) VELOCITY
  U(N, :) = 1.0

  ! START TIMING
  CALL CPU_TIME(START_TIME)

  ! MAIN LOOP FOR TIME STEPPING
  DO STEP = 1, 1000
     CALL COMPUTE_VELOCITY(U, V, P, DX, DY, DT, RE)
     CALL UPDATE_PRESSURE(P, DX, DY)

     ! OUTPUT OR CHECK CONVERGENCE
     IF (MOD(STEP, 100) == 0) THEN
         PRINT *, 'STEP: ', STEP
     END IF
  END DO

  ! STOP TIMING
  CALL CPU_TIME(END_TIME)
  ELAPSED_TIME = END_TIME - START_TIME
  PRINT *, 'ELAPSED TIME FOR CFD SIMULATION: ', ELAPSED_TIME, ' SECONDS'

CONTAINS

  ! FUNCTION TO UPDATE THE VELOCITY AND PRESSURE FIELDS (SIMPLIFIED)
  SUBROUTINE COMPUTE_VELOCITY(U, V, P, DX, DY, DT, RE)
    REAL, DIMENSION(:,:), INTENT(INOUT) :: U, V, P
    REAL, INTENT(IN) :: DX, DY, DT, RE
    INTEGER :: I, J

    ! SIMPLE EXPLICIT METHOD FOR VELOCITY (SIMPLIFIED)
    DO I = 2, N-1
        DO J = 2, N-1
            U(I, J) = U(I, J) - DT * ( (U(I, J) * (U(I+1, J) - U(I-1, J))) / (2*DX)
                                       (V(I, J) * (U(I, J+1) - U(I, J-1))) / (2*DY)
        END DO
    END DO

    ! SIMPLE VELOCITY UPDATE FOR V (SIMILAR)
    DO I = 2, N-1
        DO J = 2, N-1
```

```
            V(I, J) = V(I, J) - DT * ( (U(I, J) * (V(I+1, J) - V(I-1, J))) / (2*DX)
                                       (V(I, J) * (V(I, J+1) - V(I, J-1))) / (2*DY)
        END DO
    END DO
  END SUBROUTINE COMPUTE_VELOCITY

  ! FUNCTION TO SOLVE FOR PRESSURE (SIMPLIFIED POISSON EQUATION SOLVER)
  SUBROUTINE UPDATE_PRESSURE(P, DX, DY)
    REAL, DIMENSION(:,:), INTENT(INOUT) :: P
    REAL, INTENT(IN) :: DX, DY
    INTEGER :: I, J

    ! SIMPLE PRESSURE POISSON EQUATION (JACOBI ITERATION)
    DO I = 2, N-1
        DO J = 2, N-1
            P(I, J) = 0.25 * ( P(I+1, J) + P(I-1, J) + P(I, J+1) + P(I, J-1) )
        END DO
    END DO
  END SUBROUTINE UPDATE_PRESSURE

END PROGRAM LID_DRIVEN_CAVITY
```