FORTRAN OPERATOR FILE NAME/TYPE= STDIN
FORTRAN OPERATOR
FORTRAN OPERATOR CREATION DATE/TIME= 05-03-2025 14:16:45
FORTRAN OPERATOR
FORTRAN OPERATOR FILE= 001 PAGES= 0002 LINES= 000082
FORTRAN OPERATOR
FORTRAN OPERATOR SYSTEM= LINUX(6.12.13-AMD64)
FORTRAN OPERATOR
FORTRAN OPERATOR SYSID= ACID SYSUSER= ACID
FORTRAN OPERATOR
FORTRAN OPERATOR FORM= WIDE
FORTRAN OPERATOR
FORTRAN OPERATOR CHAR= FONTMONO
FORTRAN OPERATOR
FORTRAN OPERATOR PRT1403 VERSION= 1.5.PRE-RELEASE

```
 OOOOOOO  PPPPPPPP  EEEEEEEEE RRRRRRRR     AAA    TTTTTTTT   OOOOOOO  RRRRRRRR
OOOOOOOOO PPPPPPPPP EEEEEEEEE RRRRRRRRR   AAAAA   TTTTTTTT  OOOOOOOOO RRRRRRRRR
OO     OO PP     PP EE        RR     RR  AA   AA     TT     OO     OO RR     RR
OO     OO PP     PP EE        RR     RR AA     AA    TT     OO     OO RR     RR
OO     OO PPPPPPPPP EEEEEEEEE RRRRRRRRR AA     AA    TT     OO     OO RRRRRRRRR
OO     OO PPPPPPPP  EEEEEEEEE RRRRRRRR  AAAAAAAAA    TT     OO     OO RRRRRRRR
OO     OO PP        EE        RR  RR    AAAAAAAAA    TT     OO     OO RR  RR
OO     OO PP        EE        RR   RR   AA     AA    TT     OO     OO RR   RR
OOOOOOOOO PP        EEEEEEEEE RR    RR  AA     AA    TT     OOOOOOOOO RR    RR
 OOOOOOO  PP        EEEEEEEEE RR     RR AA     AA    TT      OOOOOOO  RR     RR


FFFFFFFFF  OOOOOOO  RRRRRRRR  TTTTTTTT  RRRRRRRR     AAA    N       NN
FFFFFFFFF OOOOOOOOO RRRRRRRRR TTTTTTTT  RRRRRRRRR   AAAAA   NN      NN
FF        OO     OO RR     RR    TT     RR     RR  AA   AA  NNN     NN
FF        OO     OO RR     RR    TT     RR     RR AA     AA NNNN    NN
FFFFFFFFF OO     OO RRRRRRRRR     TT     RRRRRRRRR AA     AA NN NN   NN
FFFFFFFFF OO     OO RRRRRRRR      TT     RRRRRRRR  AAAAAAAAA NN  NN  NN
FF        OO     OO RR  RR        TT     RR  RR    AAAAAAAAA NN   NNNN
FF        OO     OO RR   RR       TT     RR   RR   AA     AA NN    NNN
FF        OOOOOOOOO RR    RR      TT     RR    RR  AA     AA NN     NN
FF         OOOOOOO  RR     RR     TT     RR     RR AA     AA NN      N


                                  00000     00000       1
                                 0000000   0000000     11
                                00     00 00     00   111
                                00     00 00     00    11
                                00     00 00     00    11
                                00     00 00     00    11
                                00     00 00     00    11
                                00     00 00     00    11
                                 0000000   0000000   111111
                                  00000     00000    111111
```

```
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
```

FORTRAN OPERATOR FILE NAME/TYPE= STDIN
FORTRAN OPERATOR
FORTRAN OPERATOR CREATION DATE/TIME= 05-03-2025 14:16:45
FORTRAN OPERATOR
FORTRAN OPERATOR FILE= 002 PAGES= 0001 LINES= 000018
FORTRAN OPERATOR
FORTRAN OPERATOR SYSTEM= LINUX(6.12.13-AMD64)
FORTRAN OPERATOR
FORTRAN OPERATOR SYSID= ACID SYSUSER= ACID
FORTRAN OPERATOR
FORTRAN OPERATOR FORM= WIDE
FORTRAN OPERATOR
FORTRAN OPERATOR CHAR= FONTMONO
FORTRAN OPERATOR
FORTRAN OPERATOR PRT1403 VERSION= 1.5.PRE-RELEASE

```
 OOOOOOO  PPPPPPPP  EEEEEEEEE RRRRRRRR     AAA    TTTTTTTT   OOOOOOO  RRRRRRRR
OOOOOOOOO PPPPPPPPP EEEEEEEEE RRRRRRRRR   AAAAA   TTTTTTTT  OOOOOOOOO RRRRRRRRR
OO     OO PP     PP EE        RR     RR  AA   AA     TT     OO     OO RR     RR
OO     OO PP     PP EE        RR     RR AA     AA    TT     OO     OO RR     RR
OO     OO PPPPPPPPP EEEEEEEEE RRRRRRRRR AA     AA    TT     OO     OO RRRRRRRRR
OO     OO PPPPPPPP  EEEEEEEEE RRRRRRRR  AAAAAAAAA    TT     OO     OO RRRRRRRR
OO     OO PP        EE        RR  RR    AAAAAAAAA    TT     OO     OO RR  RR
OO     OO PP        EE        RR   RR   AA     AA    TT     OO     OO RR   RR
OOOOOOOOO PP        EEEEEEEEE RR    RR  AA     AA    TT     OOOOOOOOO RR    RR
 OOOOOOO  PP        EEEEEEEEE RR     RR AA     AA    TT      OOOOOOO  RR     RR


FFFFFFFFF  OOOOOOO  RRRRRRRR  TTTTTTTT  RRRRRRRR     AAA    N       NN
FFFFFFFFF OOOOOOOOO RRRRRRRRR TTTTTTTT  RRRRRRRRR   AAAAA   NN      NN
FF        OO     OO RR     RR    TT     RR     RR  AA   AA  NNN     NN
FF        OO     OO RR     RR    TT     RR     RR AA     AA NNNN    NN
FFFFFFFFF OO     OO RRRRRRRRR     TT     RRRRRRRRR AA     AA NN NN   NN
FFFFFFFFF OO     OO RRRRRRRR      TT     RRRRRRRR  AAAAAAAAA NN  NN NN
FF        OO     OO RR  RR        TT     RR  RR    AAAAAAAAA NN   NNNN
FF        OO     OO RR   RR       TT     RR   RR   AA     AA NN    NNN
FF        OOOOOOOOO RR    RR      TT     RR    RR  AA     AA NN     NN
FF         OOOOOOO  RR     RR     TT     RR     RR AA     AA NN      N


                    00000     00000    2222222
                   0000000   0000000  222222222
                  00     00 00     00 22     22
                  00     00 00     00       22
                  00     00 00     00      22
                  00     00 00     00     22
                  00     00 00     00    22
                  00     00 00     00   22
                   0000000   0000000   22222222
                    00000     00000   222222222
```

```
program performance_test
    implicit none
    integer :: i, total
    real(8) :: start_time, end_time

    total = 0
    call cpu_time(start_time)

    do i = 1, 10000000
        total = total + i
    end do

    call cpu_time(end_time)

    print *, "Fortran: The sum is ", total
    print *, "Fortran: Time taken = ", end_time - start_time
end program performance_test
```