ソースファイル名：lbm.for, 実行ファイル名：lbm.out

```
$ cp    ../common/lbm_ori.for    .
$ cp    lbm_ori.for    lbm.for
$ vi    lbm.for
$ gfortran    -o    lbm.out      lbm.for
$ ./ lbm.out
```

```
      program lbms

      parameter(imax = 30, jmax = 30)
      real*8 cx(9), cy(9)
      real*8 f(9,imax,jmax), f0(9,imax,jmax), ftmp(9,imax,jmax)
      real*8 u(imax,jmax), v(imax,jmax), rho(imax,jmax)
      real*8 tau, g, pi, u2, tmp, norm, mu
      integer i, j, k, m, mm, ii, jj, nx, ny
      character a(imax,jmax), b(imax,jmax)

      tau = 1.1; g = 0.00015; nx = 20; ny =20
      pi = 4.0*atan(1.0); mu = (tau - 0.5)/3.

c.. discrete velocity
      cx(1) =   0.0; cy(1) =   0.0
      cx(2) =   □; cy(2) =   □
      cx(3) =   □; cy(3) =   □
      cx(4) =   □; cy(4) =   □
      cx(5) = -1.0; cy(5) =   1.0
      cx(6) = -1.0; cy(6) =   □
      cx(7) = -1.0; cy(7) =   □
      cx(8) =   0.0; cy(8) =   □
      cx(9) =   1.0; cy(9) =   □

c.. initial condition
      do i = 1,nx; do j = 1,ny
        u(i,j) = 0.; v(i,j) = 0.; rho(i,j) = 1.
      end do; end do

      do i = 1,nx; do j = 1,ny
        u2 = u(i,j)**2 + v(i,j)**2
        f0(1,i,j) = rho(i,j)*(1. - 3./2.*u2)*4./9.
        do k = 1,4
```

```
      m = k*2       ; tmp = cx(m)*u(i,j) + cy(m)*v(i,j)
      f0(m,i,j) =
      m = k*2 + 1; tmp = cx(m)*u(i,j) + cy(m)*v(i,j)
      f0(m,i,j) = rho(i,j)*(1. + 3.*tmp + 9./2.*tmp**2 - 3./2.*u2)/36.
    end do; end do; end do

    do k = 1,9; do i = 1,nx; do j = 1,ny
     f(k,i,j) = f0(k,i,j)
    end do; end do; end do

    do mm = 1, 1000
```

c.. velocity, density

```
      do i = 1,nx; do j = 1,ny
       u(i,j) = 0; v(i,j) = 0; rho(i,j) = f(1,i,j);
       do k = 2,9
         u(i,j) =
         v(i,j) =
        rho(i,j) =
      end do; end do; end do

      do i = 1,nx; do j = 1,ny
       if (rho(i,j) .ne. 0.) then
         u(i,j) =   u(i,j)/rho(i,j)
         v(i,j) =   v(i,j)/rho(i,j)
       else
         u(i,j)   = 0.
         v(i,j)   = 0.
       endif
      end do; end do;
```

c.. equilibrium distribution function

```
      do i = 1,nx; do j = 1,ny
       u2 = u(i,j)**2 + v(i,j)**2
       f0(1,i,j) = rho(i,j)*(1. - 3./2.*u2)*4./9.
       do k = 1,4
         m = k*2       ; tmp = cx(m)*u(i,j) + cy(m)*v(i,j)
         f0(m,i,j) =
         m = k*2 + 1; tmp = cx(m)*u(i,j) + cy(m)*v(i,j)
         f0(m,i,j) = rho(i,j)*(1. + 3.*tmp + 9./2.*tmp**2 - 3./2.*u2)/36.
```

```
      end do; end do; end do

c..   collision
      do k = 1,9; do i = 1,nx; do j = 1,ny
       f(k,i,j) = f(k,i,j) - (f(k,i,j) - [      ])/tau
      end do; end do; end do

c..   body force
      do k = 1,9; do i = 1,nx; do j = 1,ny
       f(k,i,j) = f(k,i,j) + g*rho(i,j)*cx(k)/6.
      end do; end do; end do

c..   streaming
      do k = 1,9; do i = 1,nx; do j = 1,ny
        ftmp(k,i,j) = f(k,i,j)
      end do; end do; end do

      do k = 1,9
       if(k .eq. 1) then
        do i = 1,nx; do j = 1,ny
         ii = i       ; jj = j
         f(k,ii,jj) = ftmp(k,i,j)
        end do; end do

       else if(k .eq. 2) then
        do i = 1,nx; do j = 1,ny
         ii = i + 1; jj = j
         if(i .eq. nx) ii = 1
         f(k,ii,jj) = ftmp(k,i,j)
        end do; end do

       else if(k .eq. 3) then
        do i = 1,nx; do j = 1,ny - 1
         ii = [    ]; jj = [    ]
         if(i .eq. nx) ii = 1
         f(k,ii,jj) = ftmp(k,i,j)
        end do; end do

       else if(k .eq. 4) then
        do i = 1,nx; do j = 1,ny - 1
```

```
      ii = i       ; jj = [ ]
      f(k,ii,jj) = ftmp(k,i,j)
    end do; end do

  else if(k .eq. 5) then
    do i = 1,nx; do j = 1,ny - 1
      ii = [ ]; jj = [ ]
      if(i .eq. 1) ii = nx
      f(k,ii,jj) = ftmp(k,i,j)
    end do; end do

  else if(k .eq. 6) then
    do i = 1,nx; do j = 1,ny
      ii = [ ]; jj = [ ]
      if(i .eq. 1) ii = nx
      f(k,ii,jj) = ftmp(k,i,j)
    end do; end do

  else if(k .eq. 7) then
    do i = 1,nx; do j = 2,ny
      ii = [ ]; jj = [ ]
      if(i .eq. 1) ii = nx
      f(k,ii,jj) = ftmp(k,i,j)
    end do; end do

  else if(k .eq. 8) then
    do i = 1,nx; do j = 2,ny
      ii = [ ]; jj = [ ]
      f(k,ii,jj) = ftmp(k,i,j)
    end do; end do

  else if(k .eq. 9) then
    do i = 1,nx; do j = 2,ny
      ii = [ ]; jj = [ ]
      if(i .eq. nx) ii = 1
      f(k,ii,jj) = ftmp(k,i,j)
    end do; end do
  end if
end do
```

```
c..   boundary condition
        do i = 1,nx
         f(3,i,1 ) = [          ]
         f(4,i,1 ) = [          ]
         f(5,i,1 ) = [          ]
         f(7,i,ny) = [          ]
         f(8,i,ny) = [          ]
         f(9,i,ny) = [          ]
        end do

c..   norm
        norm = 0.
        do j = 2, ny - 1
         tmp = abs(u(nx/2, j) + g/mu/2.*(j**2 -(ny + 1)*j +ny))
         if(tmp .gt. norm) norm = tmp
        enddo

c..   graphics
        if(mod(mm,50) .eq. 0) then
         write(*,*) mm, norm, u(nx/2, ny/2)
         do i = 1, nx; do j = 1, ny
          a(i,j)= '0'
         end do; end do
         do j = 1, ny
          nb = int(u(nx/2,j)*500)
          do i = 1, nb
           a(i,j)= '-'
          end do
         end do

         do i = 1, nx; do j = 1, ny
          b(i,j)= '0'
         end do; end do
         do j = 1, ny
          nb = int((-g/mu/2.*(j**2 -(ny + 1)*j +ny))*500)
          do i = 1, nb
           b(i,j)= '-'
          end do
         end do
```

```
    do j = ny,1,-1
      write(*,*) (a(i,j),i = 1,nx),'    ',(b(i,j),i = 1,nx)
    end do
    write(*,*) '*------------------------------*'
  endif

end do

stop
end
```

《結果》

1000 1.403216463077458E-004 3.376557813295326E-002