module precision\_definition

implicit none

integer,parameter::long=selected\_real\_kind(15,307)

end module precision\_definition

module m2D\_calculat

! 2D moor line motion equations

! Dynamic tension calculation program by lumped mass method

! This program computes the dynamic tensions on the mooring

! line by use of lumped mass method

! chn-added mass coefficient (normal direction) in water

! cht-added mass coefficient (tangential direction) in water

! cdn-drag coefficient (normal direction) in water

! cdt-drag coefficient (tangential direction) in water

! nn -divided number

! xx -mooring line initial condition (horizontal direction) unit:m

! zz -mooring line initial condition (vectical direction) unit:m

! TL -total line length unit:m

! DC -equivalent circular diameter m^2

! wc -weight of mooring line in water (kg)

! wa -weight of mooring line in air (kg)

! nt -total time (sec)

! delta- time interval (sec)

! xtime- motion displacement at mooring point (m)

! ix,iz- condition cord

! Inputs

use precision\_definition

implicit none

real(long),save::chn=2.18\_long,cht=0.17\_long,cdn=1.87\_long,cdt=1.87\_long

real(long),save::TL=7.425\_long

real(long),save::DC=0.00886\_long,wc=3.15785\_long,wa=3.616\_long

real(long),save::nn=9.0\_long

integer,save::ns=9

real(long),save::g=9.80665\_long ! unit:m/s^2

real(long),save::delta=0.025\_long

integer,save::nt=800 ! unit:times,20seconds(run\_time)/0.025(delta)=800

real(long),save::tol=1e-6\_long

integer,save::ix=1,iz=0

! \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

contains

! \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

subroutine xxzz(x,z,i,xx,zz,xtime)

! dealing with time(N)- i is loop times

! set up the values of first and second line of xx,zz

real(long)::x01,z01,am0,am1

real(long)::undox,undoz

real(long),allocatable::x(:),z(:),xtime(:) ! intent(in)

real(long),allocatable::xx(:,:),zz(:,:) !intent(out)

integer,intent(in)::i

x01=x(ns+1)

z01=z(ns+1)

am0=1-exp(-delta\*(i-1))

am1=1-exp(-delta\*(i)/10)

undox=0

undoz=0

if (ix==1) then

undox=1

endif

if (iz==1) then

undoz=1

endif

xx(ns+1,2)=x01+undox\*am0\*xtime(i)

zz(ns+1,2)=z01+undoz\*am0\*xtime(i)

xx(ns+1,3)=x01+undox\*am1\*xtime(i+1)

zz(ns+1,3)=z01+undoz\*am1\*xtime(i+1)

end subroutine

! \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

subroutine correct(i,ki,w1,xx,zz,wj,dels)

integer::i,ki

real(long),allocatable::w1(:),xx(:,:),zz(:,:) ! intent(in)

real(long),allocatable::wj(:) ! intent(out)

real(long)::x01,x02,dam,da,db,dex,dels

if (i>=2) then

wj(ki)=1.5\*w1(ki)

wj(ki+1)=w1(ki+1)

if (ki/=1) then

if (zz(ki-1,3)>0) then

ki=ki-1

endif

endif

if (zz(ki,3)<=0) then

ki=ki+1

endif

x01=xx(ki,3)-xx(ki-1,3)

x02=xx(ki+1,3)-xx(ki-1,3)

dam=x01\*x02\*(x01-x02)

da=(zz(ki,3)\*x02-zz(ki+1,3)\*x01)/dam

db=((x01\*\*2)\*zz(ki+1,3)-(x02\*\*2)\*zz(ki,3))/dam

dex=-db/da

if (db>0) then

dex=0.0

endif

wj(ki)=(1-dex/dels)\*1.5\*w1(ki)

wj(ki+1)=(1+dex/dels\*0.5)\*w1(ki+1)

endif

end subroutine

! \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

subroutine hydro\_drag(xx,zz,dcoen,dcoet,dels,am,An,At,ds,xr,xs,xp,xq,wj,Fx,Fz,xco,xsi,xd,zd,xd1,zd1)

real(long),allocatable::xx(:,:),zz(:,:),ds(:) ! intent(in)

real(long),allocatable::wj(:),am(:),An(:),At(:) ! intent(in)

real(long),allocatable::xr(:),xs(:),xp(:),xq(:) ! intent(out)

real(long),allocatable::Fx(:),Fz(:),xco(:),xsi(:) ! intent(out)

real(long),allocatable::xd(:),zd(:) ! intent(inout)

real(long),allocatable::xcos(:),xsin(:),alf(:)

real(long),allocatable::xd1(:),zd1(:) ! intent(out)

real(long),allocatable::xd0(:),zd0(:)

real(long),allocatable::xi1(:),xi2(:),xi3(:)

real(long),allocatable::un(:),ut(:),ww(:)

real(long)::ai1,ai2,ai3,alamd

real(long)::dcon,dcot,fdn,fdt,fdx,fdz

real(long)::dcoen,dcoet,dels

integer::j

allocate(xcos(ns))

allocate(xsin(ns))

allocate(alf(ns))

allocate(xd0(ns))

allocate(zd0(ns))

allocate(xi1(ns))

allocate(xi2(ns))

allocate(xi3(ns))

allocate(un(ns))

allocate(ut(ns))

allocate(ww(ns))

! hydro

do j=1,ns

xcos(j)=(xx(j+1,3)-xx(j,3))/ds(j)

xsin(j)=(zz(j+1,3)-zz(j,3))/ds(j)

alf(j)=asin(xsin(j))

xd1(j)=xd(j)

zd1(j)=zd(j)

xd0(j)=xd(j)

zd0(j)=zd(j)

enddo

xco(1)=cos(alf(1)/2)

xsi(1)=sin(alf(1)/2)

do j=2,ns

xco(j)=cos((alf(j)+alf(j-1))/2)

xsi(j)=sin((alf(j)+alf(j-1))/2)

ai1=am(j)+An(j)\*xsi(j)\*\*2+At(j)\*xco(j)\*\*2

ai2=(An(j)-At(j))\*xco(j)\*xsi(j)

ai3=am(j)+An(j)\*xco(j)\*\*2+At(j)\*xsi(j)\*\*2

alamd=ai1\*ai3-ai2\*\*2

xi1(j)=ai1/alamd\*delta\*\*2

xi2(j)=ai2/alamd\*delta\*\*2

xi3(j)=ai3/alamd\*delta\*\*2

xr(j)=xi3(j)\*xcos(j)+xi2(j)\*xsin(j)

xs(j)=xi1(j)\*xsin(j)+xi2(j)\*xcos(j)

xp(j)=xi3(j)\*xcos(j-1)+xi2(j)\*xsin(j-1)

xq(j)=xi1(j)\*xsin(j-1)+xi2(j)\*xcos(j-1)

enddo

! \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

! drag

do j=2,ns

un(j)=-xd0(j)\*xsi(j)+zd0(j)\*xco(j)

ut(j)=xd0(j)\*xco(j)+zd0(j)\*xsi(j)

ww(j)=wj(j)

if (zz(j,3)<=0) then

ww(j)=0

endif

if (j==2.or.j==ns) then

dcon=1.5\*dcoen

dcot=1.5\*dcoet

else

dcon=dcoen

dcot=dcoet

endif

fdn=dcon\*abs(un(j))\*un(j)

fdt=dcot\*abs(ut(j))\*ut(j)

fdx=-fdn\*xsi(j)+fdt\*xco(j)

fdz=fdn\*xco(j)+ww(j)+fdt\*xsi(j)

Fx(j)=xi3(j)\*fdx+xi2(j)\*fdz

Fz(j)=xi2(j)\*fdx+xi1(j)\*fdz

enddo

end subroutine

! \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

subroutine matrix\_tens(xx,zz,ds,ts,xr,xs,xp,xq,Fx,Fz,i,xco,xsi,xtt)

real(long),allocatable::xx(:,:),zz(:,:),ts(:,:) ! intent(inout)

real(long),allocatable::xr(:),xs(:),xp(:),xq(:),Fx(:),Fz(:),ds(:) ! intent(in)

real(long),allocatable::xco(:),xsi(:)

real(long),allocatable::xmat(:,:),tt(:)

real(long),allocatable::ee(:),ff(:),ag(:),fai(:)

real(long),allocatable::tx(:,:),tz(:,:)

real(long)::deltx,deltz,xtt

integer::i

integer::j

allocate(xmat(ns,ns))

allocate(tt(ns+1))

allocate(ee(ns))

allocate(ff(ns))

allocate(ag(ns))

allocate(fai(ns))

allocate(tx(ns,nt))

allocate(tz(ns,nt))

! matrix

do j=2,ns

xx(j,3)=2\*xx(j,2)-xx(j,1)+xr(j)\*ts(j,i)-xp(j)\*ts(j-1,i)-Fx(j)

zz(j,3)=2\*zz(j,2)-zz(j,1)+xs(j)\*ts(j,i)-xq(j)\*ts(j-1,i)-Fz(j)

if (zz(j,3)<=0) then

zz(j,3)=0

endif

enddo

do j=1,ns

deltx=xx(j+1,3)-xx(j,3)

deltz=zz(j+1,3)-zz(j,3)

ee(j)=deltx\*xp(j)+deltz\*xq(j)

ff(j)=-deltx\*(xp(j+1)+xr(j))-deltz\*(xq(j+1)+xs(j))

ag(j)=deltx\*xr(j+1)+deltz\*xs(j+1)

fai(j)=(ds(j)\*\*2-deltx\*\*2-deltz\*\*2)/2

enddo

! \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

! tens

do j=1,ns

if (j<ns) then

xmat(j+1,j)=ee(j+1)

endif

xmat(j,j)=ff(j)

if (j>1) then

xmat(j-1,j)=ag(j-1)

endif

enddo

call brinv(xmat,ns)

tt(1:ns)=matmul(xmat,fai)

ts(:,i+1)=ts(:,i)+tt

do j=1,ns

tx(j,i)=ts(j,i+1)\*xco(j)

tz(j,i)=ts(j,i+1)\*xsi(j)

enddo

xtt=abs(tt(ns)/ts(ns,i+1))

end subroutine

! \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

subroutine motion(xx,zz,ts,xr,xs,xp,xq,Fx,Fz,i,xd,zd,xd1,zd1)

! solve for x,y location

real(long),allocatable::xx(:,:),zz(:,:),ts(:,:) !intent(inout)

real(long),allocatable::xr(:),xs(:),xp(:),xq(:) !intent(in)

real(long),allocatable::xd(:),zd(:),xd1(:),zd1(:) ! intent(inout)

real(long),allocatable::Fx(:),Fz(:) ! intent(in)

real(long),allocatable::ico(:),dz0(:),dx0(:)

real(long),allocatable::xd0(:),zd0(:)

real(long),allocatable::am(:),an(:),at(:),ds(:),wj(:)

real(long),allocatable::xco(:),xsi(:)

real(long)::dcoen,dcoet,dels

integer::i,j

integer::kki,icond

allocate(ico(ns))

allocate(dz0(ns))

allocate(dx0(ns))

allocate(xd0(ns))

allocate(zd0(ns))

icond=0

do j=2,ns

ico(j)=0

dz0(j)=zd(j)

dx0(j)=xd(j)

xx(j,3)=2\*xx(j,2)-xx(j,1)+xr(j)\*ts(j,i+1)-xp(j)\*ts(j-1,i+1)-Fx(j)

zz(j,3)=2\*zz(j,2)-zz(j,1)+xs(j)\*ts(j,i+1)-xq(j)\*ts(j-1,i+1)-Fz(j)

if (zz(j,3)<=0) then

zz(j,3)=0

endif

xd(j)=(xx(j,3)-xx(j,2))/delta

zd(j)=(zz(j,3)-zz(j,2))/delta

xd0(j)=(xd1(j)+xd(j))/2

zd0(j)=(zd1(j)+zd(j))/2

if (abs(zd(j)-dz0(j))/(zd(j)+1e-10)>1e-4) then

ico(j)=1

endif

icond=icond+ico(j)

enddo

if (kki==99) then

goto 100

endif

if (icond/=0) then

kki=kki+1

call hydro\_drag(xx,zz,dcoen,dcoet,dels,am,An,At,ds,xr,xs,xp,xq,wj,Fx,Fz,xco,xsi,xd,zd,xd1,zd1)

endif

100 continue

kki=0

end subroutine

! \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

subroutine step(xx,zz)

real(long),allocatable::xx(:,:),zz(:,:)

integer::j

do j=2,ns+1

xx(j,1)=xx(j,2)

xx(j,2)=xx(j,3)

zz(j,1)=zz(j,2)

zz(j,2)=zz(j,3)

enddo

end subroutine

! \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

! solve for the inverse of the matrix

subroutine brinv(xmat,ns)

real(long)::t,d

integer::i,j,k,L,ns

real(long),allocatable::xmat(:,:),is(:),js(:)

allocate(is(ns))

allocate(js(ns))

L=1

do 100 k=1,ns

d=0.0

do 10 i=k,ns

do 10 j=k,ns

if (abs(xmat(i,j)).gt.d) then

d=abs(xmat(i,j))

is(k)=i

js(k)=j

endif

10 continue

if (d+1.0.eq.1.0) then

L=0

write(\*,20)

20 format(1x,'err\*\*not inv')

return

endif

do 30 j=1,ns

t=xmat(k,j)

xmat(k,j)=xmat(int(is(k)),j)

xmat(int(is(k)),j)=t

30 continue

do 40 i=1,ns

t=xmat(i,k)

xmat(i,k)=xmat(i,int(js(k)))

xmat(i,int(js(k)))=t

40 continue

xmat(k,k)=1.0/xmat(k,k)

do 50 j=1,ns

if (j.ne.k) then

xmat(k,j)=xmat(k,j)\*xmat(k,k)

endif

50 continue

do 70 i=1,ns

if(i.ne.k) then

do 60 j=1,ns

IF(j.ne.k) then

xmat(i,j)=xmat(i,j)-xmat(i,k)\*xmat(k,j)

endif

60 continue

endif

70 continue

DO 80 i=1,ns

if(i.ne.k) then

xmat(i,k)=-xmat(i,k)\*xmat(k,k)

endif

80 continue

100 continue

do 130 k=ns,1,-1

do 110 j=1,ns

t=xmat(k,j)

xmat(k,j)=xmat(int(js(k)),j)

xmat(int(js(k)),j)=t

110 continue

do 120 i=1,ns

t=xmat(i,k)

xmat(i,k)=xmat(i,int(is(k)))

xmat(i,int(is(k)))=t

120 continue

130 continue

end subroutine

end module m2D\_calculat

program m2dequation

use precision\_definition

use m2D\_calculat

implicit none

real(long)::rho,dww,dwa,dcoen,dcoet,dels

real(long)::V,pi

real(long)::xtt

real(long),allocatable::wj(:),am(:),An(:),At(:),xtime(:)

real(long),allocatable::w1(:),ds(:),x(:),z(:),t(:)

real(long),allocatable::xx(:,:),zz(:,:),ts(:,:)

real(long),allocatable::xlocation(:,:),zlocation(:,:)

real(long),allocatable::xp(:),xq(:),xr(:),xs(:)

real(long),allocatable::xd(:),zd(:),xd1(:),zd1(:),Fx(:),Fz(:)

real(long),allocatable::xco(:),xsi(:)

integer::j

integer::i,jj

integer,save::ki

character(len=80)::file\_name

allocate(wj(ns+1))

allocate(am(ns+1))

allocate(An(ns+1))

allocate(At(ns+1))

allocate(xtime(nt+1))

allocate(xx(ns+1,3))

allocate(zz(ns+1,3))

allocate(ts(ns+1,nt+1))

allocate(x(ns+1))

allocate(z(ns+1))

allocate(t(ns+1))

allocate(w1(ns+1))

allocate(ds(ns+1))

allocate(xp(ns+1))

allocate(xq(ns+1))

allocate(xr(ns+1))

allocate(xs(ns+1))

allocate(xd(ns+1))

allocate(zd(ns+1))

allocate(xd1(ns))

allocate(zd1(ns))

allocate(Fx(ns+1))

allocate(Fz(ns+1))

allocate(xco(ns))

allocate(xsi(ns))

allocate(xlocation(ns+1,nt+1))

allocate(zlocation(ns+1,nt+1))

data pi /3.1415926535898/

call initials

do i=1,nt

xlocation(:,i)=xx(:,1)

zlocation(:,i)=zz(:,1)

call xxzz(x,z,i,xx,zz,xtime)

call correct(i,ki,w1,xx,zz,wj,dels)

300 continue

call hydro\_drag(xx,zz,dcoen,dcoet,dels,am,An,At,ds,xr,xs,xp,xq,wj,Fx,Fz,xco,xsi,xd,zd,xd1,zd1)

200 continue

call matrix\_tens(xx,zz,ds,ts,xr,xs,xp,xq,Fx,Fz,i,xco,xsi,xtt)

if (xtt>=1e-4) then

ts(:,i)=ts(:,i+1)

goto 200

endif

call motion(xx,zz,ts,xr,xs,xp,xq,Fx,Fz,i,xd,zd,xd1,zd1)

call step(xx,zz)

enddo

write(\*,\*) ts(ns,1:20)

! \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

contains

! \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

subroutine initials

! returns include: rho,dels,dww,dwa,dcoen,dcoet,V

! returns include: wj,am,An,At,xtime,w1,ds

! returns include: xx,zz,ts,x,z,t,jj(not use in the main program),ki

rho=1000.0/g

dels=TL/nn

dww=wc/TL

dwa=wa/TL

dcoen=rho\*cdn\*DC\*dels\*0.5

dcoet=rho\*cdt\*DC\*dels\*0.5

V=pi\*DC\*\*2/4.0\*dels

do j=1,ns+1

if (j==1.or.j==ns+1) then

wj(j)=0

am(j)=0

An(j)=0

At(j)=0

elseif (j==2.or.j==ns) then

wj(j)=1.5\*dww\*dels

am(j)=1.5\*(dwa\*dels)/g

An(j)=1.5\*V\*rho\*chn

At(j)=1.5\*V\*rho\*cht

else

wj(j)=dww\*dels

am(j)=(dwa\*dels)/g

An(j)=V\*rho\*chn

At(j)=V\*rho\*cht

endif

enddo

xtime(1)=0.05\*sin(2.0\*pi/2.0\*delta\*(-1.0))

do j=2,nt

xtime(j)=0.05\*sin(2.0\*pi/2.0\*delta\*(j-2))

enddo

print \*,'please input the file name of input file within 80 bytes'

!read '(A)',file\_name

!open(unit=1,file=file\_name)

!read(unit=1,fmt=\*) (xx(j,1),zz(j,1),ts(j,1),j=1,ns+1)

read \*,(xx(j,1),zz(j,1),ts(j,1),j=1,ns+1)

do j=1,ns+1

x(j)=xx(j,1)

z(j)=zz(j,1)

t(j)=ts(j,1)

enddo

do j=1,2

xx(:,j+1)=xx(:,j)

zz(:,j+1)=zz(:,j)

enddo

xx(1,1:3)=0

zz(1,1:3)=0

do j=2,ns+1

w1(j)=dww\*dels

ds(j-1)=sqrt((xx(j,1)-xx(j-1,1))\*\*2+(zz(j,1)-zz(j-1,1))\*\*2)

enddo

w1(1)=0

w1(ns+1)=0

ds(ns+1)=0

do j=1,ns+1

if (z(j)<=tol) then

jj=j

endif

enddo

if (z(2)>=tol) then

jj=2

endif

ki=jj+1

end subroutine

! \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

end program