    !main bit of the function

    do j = 1,Nt

        call random\_number(sample)

        qnet(N0+j) = L

        do i = 1,N0+j

            qsum = sum(qnet) - L

            p(1) = 0

            p(i+1) = qnet(i)/qsum

            do k = 1,L

                enet(1,N0+j+k-1) = N0+t

                do l = 1,N0+j

                    if ((sum(p(1:l)) < sample(k)) .and. (sample(k)<= sum(p(1:l+1)))) then

                        qnet(l+1) = qnet(l) + 1

                        enet(2,N0+j+k-1) = l+1

                    end if

                end do

            end do

        end do

    end do

) .and. (sample(k)<= sum(p(1:r+1)))

                else if (sum(p(1:r)) < sample(k)) then

                    if (sample(k)<= sum(p(1:r+1))) then

                        qnet(r+1) = qnet(r+1) + 1

                        enet(2,N0+j+k-1) = r+1

                    end if

!Generate recursive matrix corresponding

!to model parameters provided as input

implicit none

integer, intent(in) :: N0,L,Nt

integer, intent(out) :: qmax

    !real (kind = 8) , dimension(L) :: sample

    real (kind = 8) , dimension(N0+Nt+1) :: p

    integer :: i, j, k, r1

    real (kind = 8) :: qsum, sample

    !allocate enet and qnet

    allocate(enet(2,N0+Nt\*L))

    allocate(qnet(N0+Nt))

    !set initial values of qnet

    do i = 1,N0

        qnet(i) = 2

    end do

    do i = N0+1,N0+Nt

        qnet(i) = 0

    end do

    !initialize enet as well

    do i = 1,2

        do j = 1,N0

            enet(i,j) = j+i-1

        end do

    end do

    enet(2,N0) = 1

    do i = 1,2

        do j = N0+1, N0+Nt\*L

            enet(i,j) = 0

        end do

    end do

    !sample = 0.0d0

    p(1) = 0.0d0

    !main chunk of the subroutine

    !fix time and take a sample, assign qnet for new node.

    do j = 1,Nt

        qnet(N0+j) = L

        !create probability array for this new node, qsum real, mixed mode for p(i)

        do i = 1,N0+j

            qsum = sum(qnet) - L

            p(i+1) = qnet(i)/qsum + p(i)

        end do

        ! over the sample, assign L elements from N0+1 to N0+j-1+k in enet(1,:)

        do k = 1,L

            call random\_number(sample)

            enet(1,N0+j+k-1) = N0+j

            !test where the point is, assign values in qnet and enet(2,:) accordingly for 0<=sample(k)<=p(1)

            !if (sample <= p) then

            ! enet(2, N0+j+k-1) = 1

            !qnet(1) = qnet(1) + 1

            !in the event the sample is greater than p(1), run through cumulitive prob.s until we find interval in which the sample lies and update enet,qnet then

            !else

                do r1 = 1,N0+j-1

                    !G = p(1:r)

                    if (sample < p(r1+1)) then

                        !if (sample < sum(p(1:r+1))) then

                            enet(2,N0+j+k-1) = r1

                            qnet(r1) = qnet(r1) + 1

                            exit

                    end if

                end do

        end do

    end do

    qmax = maxval(qnet)