```
function [chain,state] = MarkovSimulation(PI, N, S, S0)
    % INPUTS:
    %   PI  : Transition matrix
    %   N   : length of simulation
    %   S   : State vector
    %   S0  : initial state (index)

    % ------------------------------------------------------------------------------
    % 1. validacoes de inputs

    % Verifica a quantidade de inputs minimo e maximo
    narginchk(2,4)

    [rPI, cPI] = size(PI);
    [rN, cN] = size(N);

    % Validade\Fill in unset optional values.
    switch nargin
        case 2
            S=1:rPI;
            S0=1;
        case 3
            S0=1;
    end

    [rS, cS] = size(S);

    % (a) Checking the total of shocks
    % Checks if N is a scalar or not
    if rN == 1 & cN == 1
        N = round(abs(N));
    else
        error('The first input must be a scalar');
    end

    % (b) checking the Transition matrix
    % Checks if it is a square matrix or not
    if (rPI == 1 | cPI == 1) | (rPI ~= cPI)
        error('The second input must be a square matrix.');
    end

    % changes the sign if some probabilities are negative
    PI = abs(PI);

    % checks if the probabilities sum to one, if not, it normalizes
    for i = 1:rPI
        if sum(PI(i,:)) ~= 1
            fprintf('probability: %f', sum(PI(i,:)));
            warning('The probabilities don`t sum to 1.');
            PI(i,:) = PI(i,:)/sum(PI(i,:));
        end
```

```
    end

    % (c) checking the State Vector
    if rS > 1
        error('State vector must be 1xN.')
    elseif ~(cS==cPI)
        error('Number of state does not match size of Transition matrix')
    end

    % -------------------------------------------------------------------------
    % 2. Creating the shock realizations

    % Cria matrix de somas acumuladas
    cum_PI = [zeros(rPI,1) cumsum(PI')'];

    % Cria o vetor de simulacao
    simulation = rand(N,1);
    state = nan(N,1);
    state(1) = S0;

    for i=2:N
        state(i)=find(((simulation(i) <= cum_PI(state(i-1), 2:cPI+1))&(simulation(i)↙
>cum_PI(state(i-1),1:cPI))));
    end

    chain=S(state);

end %end of function
```