```
function [T,X] = OpenLoopSimulation(x0, tspan, U, D, p, simModel, simMethod, NK)
% OpenLoopSimulation()
%
% DESCRIPTION:
% Function peforms an open-loop simulation for given initial condition of
% the state vector, time, intervals, disturbance variables, parameters, and
% simulation model and methods. The open-loop simulation uses the MVPmodel
% and ExplicitEuler to compute both the subcutaneous glucose concentration,
% Gsc(t), the blood glucose concentration G(t) and the statevector x(t) for
% each time step.
%
%
% INPUT:
% x0            - initial state vector                          (dimension: 7)
% tspan         - time interval to integrate over               (dimension N+1)
% U             - bolus and basal insulin (manipulated input)   (dimension nu x N)
% D             - meal rate (disturbance)                       (dimension nd x N)
% p             - parameter values                              (dimension np)
% simModel      - simulation model, MVPmodel                    (function handle)
% simMethod     - simulation method, ExplicitEuler              (function handle)
% NK            - Number of timesteps in each time interval
%
% OUTPUT:
% T - The control state of time for each step                           (dimension:↙
N+1)
% X - The statevector x(t) for each time step stored in a matrix        (dimension: nx↙
x N+1)
%
% PROJECT:
% Fagprojekt 2022
% A diabetes case study - Meal detection
%
% GENERAL:
% BSc                       : Mathematics and technology
% University                : The Technical University of Denmark (DTU)
% Department                : Applied Mathematics and Computer Science
%
% AUTHORS:
% Emma Victoria Lind
% Mariana de Sá Madsen
% Mona Saleem
%
% CONTACT INFORMATION
% s201205@student.dtu.dk
% s191159@student.dtu.dk
% s204226@student.dtu.dk
%



% Number of control steps
N = numel(tspan) - 1;

% Number of states
nx = numel(x0);

% Number of time steps in each control interval
Nk=NK;
```

```
% Allocate memory
T = zeros(1, N+1);
X = zeros(nx, N+1);

% Initial condition in each control interval
xk = x0;

% Store solution
T(1) = tspan(1);
X(:,1) = x0;

% Loop for each time step. Computes Gsc(t), G(t), x(t) from k = 0 to N.
for k = 1:N
    % Times
    tk    = tspan(k);
    tkp1  = tspan(k+1);

    % Manipulated inputs and disturbance variables
    uk = U(:,k);
    dk = D(:,k);

    % Time interval
    tspank = linspace(tk, tkp1, Nk+1);

    % Solve initial value problem
    [Tk, Xk] = simMethod(simModel, tspank, xk, uk, dk, p);

    % Update initial condition
    xk = Xk(end, :)';

    % Store solution
    T(k+1) = Tk(end)';
    X(:, k+1) = Xk(end, :)';

end


end
```