function pvalue = ExplicitB1(TS)

% initialize the parameters for the model from the question

So = 100;                        % stock price
Smax = 200;                      % upper limit of PDE mesh
Smin = 20 ;                      % lower limit of PDE mesh
M_x  = 0.5;                      % the fineness of stock grid
N = (Smax - Smin) / M_x ;        % this is the horizontal grid number
T = 1;                           % the time to maturity
% TS = 10000;                    % the time grid size
dt = T/TS;                       % the fineness of time grid
K  = 90;                         % strike price
sigma = 0.25;                    % volatility
r = 0.0525;                      % interest rate
q = 0.02;                        % dividend rate

% define our initial condition and stock price matrix
S = zeros(N,1);                  % the stock price matrix
V = zeros(N,1);                  % the initial value matrix
p = zeros(N,1);                  % the boundary condition
% to define our solution matrix
sol = zeros(N,TS + 1);

% initialize the above matrices
for i  = 1:1:N
    S(i) = Smin + (i * M_x);    % stock price points
    V(i) = max(K - S(i),0);     % the initial condition for a put
end

% defining and building our explicit function matrix
% this matrix will have a tridiagonal structure
B = zeros(N,N);
for i = 1:1:N
    term1 = 0.5 * (sigma ^ 2) * (S(i) ^ 2) * (dt/(M_x ^ 2));
    term2 = (r - q) * S(i) * (dt / (2 * M_x));
    % the coeffecients of the equation
    alpha = term1 - term2;
    beta  = 1 - (r * dt) - (2 * term1);
    gamma = term1 + term2;
    % get beta in the first element of the matrix for BC1
    if i == 1
        B(i,i) = beta;
        % get gamma in the second element
        B(i,i + 1) = gamma;
    elseif i == N
        % to fill the last row of the matrix

B(i,i-1) = alpha;
B(i,i) = beta;
else
    % for rest of the tri - diagonal elements
    B(i, i - 1) = alpha;
    B(i,i) = beta;
    B(i,i + 1) = gamma;
end

% include the initial condition in the solution matrix
sol(:,1) = V;
% now solving the PDE for each time step
for i = 1:TS
    tau  = (i - 1) * dt;
    % get the boundary condition in p
    p(1) = 0.5 * dt * (N - 1) * (((sigma ^ 2) * (N - 1)) - (r - q)) * ((K * exp(-r * sqrt(tau))));
    % solve the next step with the help of the previous one
    sol(:,i + 1) = B*sol(:,i) + p;
end

ans = (So - Smin)/M_x;
value = sol(ans,TS + 1);

% numerical greeks using finite differences
fdelta = (-sol(ans - 1, TS + 1) + sol(ans + 1, TS + 1)) / (2 * M_x);
fgamma = (sol(ans + 1, TS + 1) - (2 * sol(ans,TS + 1)) + sol(ans - 1, TS + 1))/(M_x ^ 2);

% to get the Black Scholes price and Greeks

d1 = (log(So/K) + ((r - q + (0.5 * sigma ^ 2)) * T))/(sigma * sqrt(T));
d2 = d1 - sigma * sqrt(T);
d3 = normcdf(d1);
d4 = normcdf(d2);
d5 = normcdf(-d1);
d6 = normcdf(-d2);
BSprice = - (So  * exp(-q * T) * d5) + (K * d6 * exp(-r * T));
delta = exp(-q * T) * (d3 - 1);
gamma = (normpdf(d1) * exp(-q * T))/(So * sigma * sqrt(T));
kappa = (So * exp(-q * T) * normpdf(d1) * sqrt(T));
pvalue = value/BSprice;