% initialize the parameters for the model from the question

So = 100;                        % stock price
Smax = 300;                      % upper limit of PDE mesh
Smin = 0 ;                       % 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 = 1000;                       % the time grid size
dt = T/TS;                       % the fineness of time grid
K  = 110;                        % strike price
sigma = 0.50;                    % 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
% 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 implicit function matrix
% this matrix will have a tridiagonal structure
A = 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 = term2 - term1;
    beta  = 1 + (r * dt) + (2 * term1);
    gamma = -term1 - term2;
    % get beta + 2alpha in the first element of the matrix for BC2
    if i == 1
        A(i,i) = (2 * alpha) + beta;
    % get gamma - alpha in the second element
    elseif i == N
        A(i,i + 1) = gamma - alpha;
end
% to fill the last row of the matrix
\begin{verbatim}
    A(i,i-1) = alpha - gamma;
    A(i,i) = beta + (2 * gamma);
    else
        % for rest of the tri-diagonal elements
        A(i, i - 1) = alpha;
        A(i,i) = beta;
        A(i,i + 1) = gamma;
    end

eb = zeros(TS,1);
% include the initial condition in the solution matrix
sol(:,1) = V;
A = A^-1;
% now solving the PDE for each time step
for i = 1:TS
    % solve the next step with the help of the previous one
    sol(:,i + 1) = A*(sol(:,i));
    % to adjust the solution vector according to the Bermudan approach
    for j = 1:N
        if sol(j,i + 1) >= max(K - S(j),0)
            k = j;
            break
        end
    end
% to get the critical stock price curve
    for j = 1:N
        if sol(j,i + 1) < max(K - S(j),0)
            sol(j,i + 1) = max(K - S(j),0);
        end
    end
    % contains the critical stock prices
    EB(TS + 1 - i) = k * M_x;
end

ans = (So - Smin)/M_x;
value = sol(ans,TS + 1);
\end{verbatim}