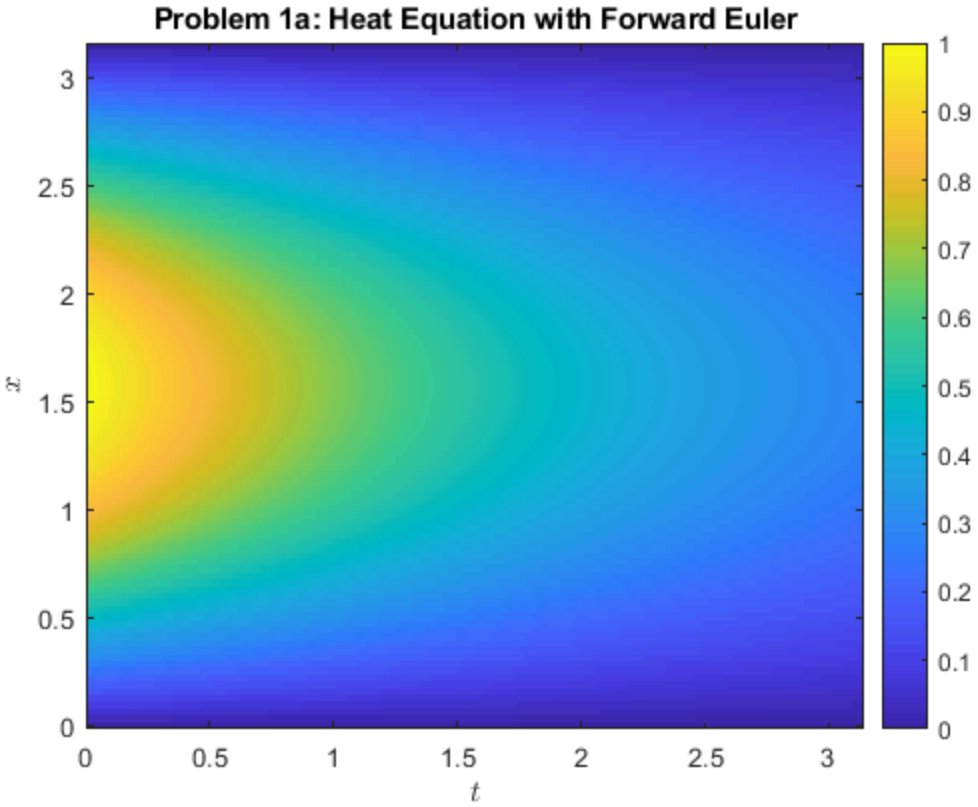
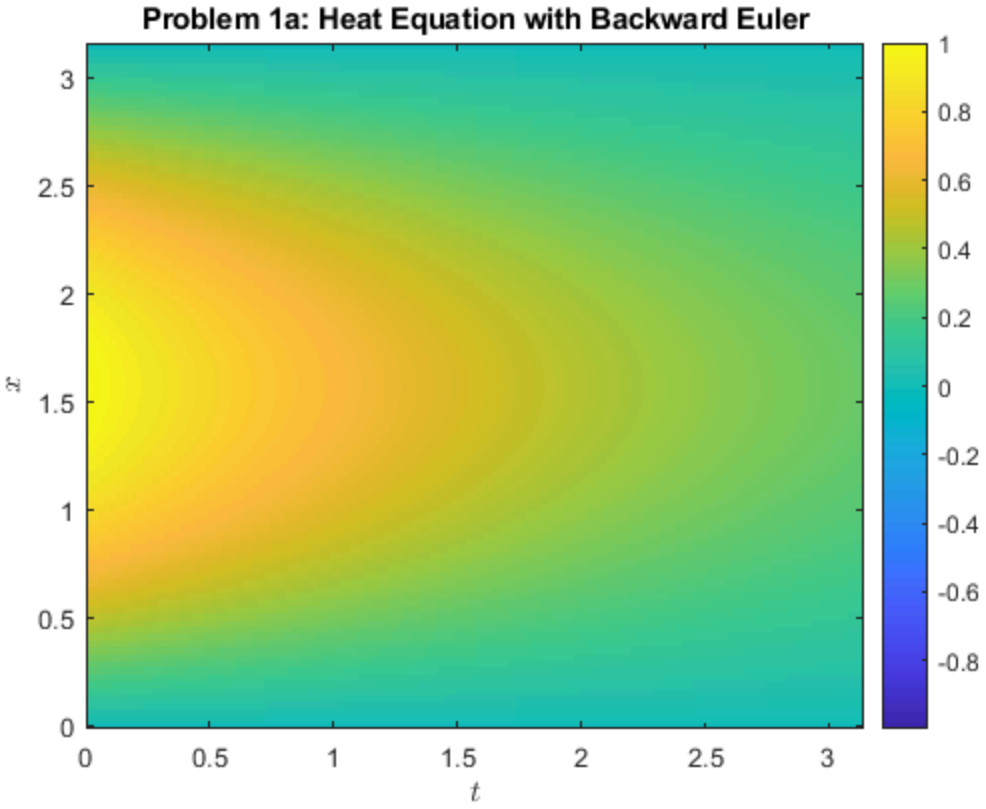
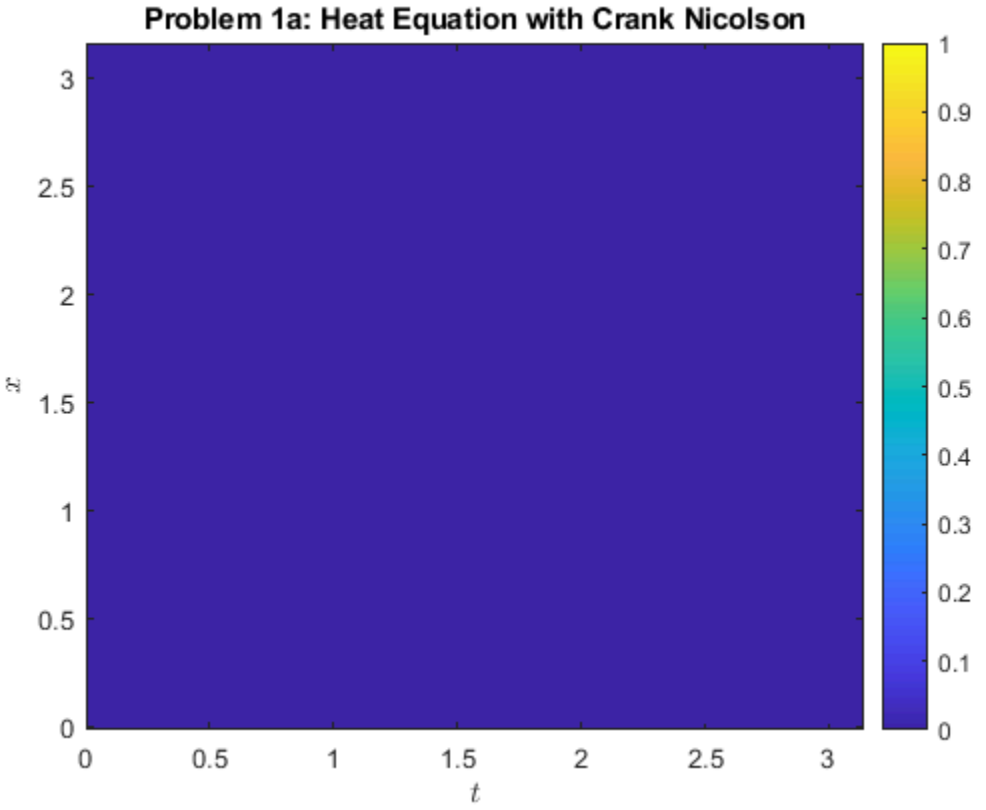
```
clc, clear;
close all;

%{
    ENG 180 Project 8
    File:           ENG180_PJ8_main_Hieu_Bui.m
    Author:         Hieu Bui
    Date:           11/14/2022
    Description:    Solving mixed partial initial boundary value problems.

%}

[x,t,u] = heatEqn('fe',.4);
figure('Name',"Forward Euler"), imagesc(t,x,u')
set(gca,'Ydir','normal')
title('Problem 1a: Heat Equation with Forward Euler')
xlabel('$t$', 'Interpreter','latex')
ylabel('$x$','Interpreter','latex')
colorbar;

[x1,t1,u1] = heatEqn('be',.4);
figure('Name',"Backward Euler"), imagesc(t1,x1,u1')
set(gca,'Ydir','normal')
title('Problem 1a: Heat Equation with Backward Euler')
xlabel('$t$', 'Interpreter','latex')
ylabel('$x$','Interpreter','latex')
colorbar;

[x2,t2,u2] = heatEqn('crank',.4);
figure('Name',"Crank Nicolson"), imagesc(t2,x2,u2')
set(gca,'Ydir','normal')
title('Problem 1a: Heat Equation with Crank Nicolson')
xlabel('$t$', 'Interpreter','latex')
ylabel('$x$','Interpreter','latex')
colorbar;

%====Problem 1====%
% 1A Heat Equation: du/dt = k*d2u/dt2; k = .4; 0<=t<=pi
% generate u(x,0) with given function
% solve for the next time step
% solve for u(x,t_n+1) with tridiagonal system
% solve for next time step
function [x,t,u] = heatEqn(option,k)

%====grid====%
n = 101;
x = linspace(0,pi,n);
deltax = x(2);
deltat = .9*(deltax^2/2);
t = 0:deltat:pi;
initialTemp = @(x) sin(x);
alp = (deltat*k)/(deltax^2);
```

```
switch option
    case 'fe'
        solver = @(u0,u1,u2,alp) alp*(u0-2*u1+u2)+u1;
        u = zeros(length(t),n);
        u(1,:) = initialTemp(x); % temperature at time 0 for all x
        u(:,n) = 0;
        for j = 1:length(t)-1 % going along t
            for i = 2:n-1 % going along x
                u0 = u(j,i-1);
                u1 = u(j,i);
                u2 = u(j,i+1);
                u(j+1,i) = solver(u0,u1,u2,alp);
            end

        end
    case 'be'
        u = zeros(length(t),n);
        u(1,:) = initialTemp(x); % temperature at time 0 for all x
        u(:,n) = 0;
        a = alp.*ones(n); % lower diagonal
        b = -(2*alp+1).*ones(n); % main diagonal
        c = alp.*ones(n); % upper diagonal
        b(1) = 1; c(1) = 0; a(n) = 0; b(n) = 1; % boundary conditions
        for j = 1:length(t)-1
            d = u(j,:); % result array changes with respect to time
            u(j+1,:) = THOMAS3(a,b,c,d,n);
        end
    case 'crank'
        u = zeros(length(t),n);
        u(1,:) = initialTemp(x); % temperature at time 0 for all x
        u(:,n) = 0;
        alpha = (deltat*k)/(2*deltax^2);
        a = -alpha.*ones(n);
        b = (1+2*alpha).*ones(n);
        c = -alpha.*ones(n);
        b(1) = 1; c(1) = 0; a(n) = 0; b(n) = 1; % boundary conditions
        d = zeros(n,1);
        for j = 1:length(t)-1
            for i = 2:n-1
                h1 = alpha*u(j,i-1);
                h2 = (1-2*alpha)*u(j,i);
                h3 = alpha*u(j,i+1);
                d(j,i) = (h1+h2+h3);
            end
            u(j+1,:) = THOMAS3(a,b,c,d,n);
        end

end
end

% supporting function
function x = THOMAS3(a,b,c,d,n)
```

```
    %initial condition
    bbar(1) = b(1);
    cbar(1) = c(1);
    dbar(1) = d(1);

    %making upper triangle
    for i = 2:n
        multiplier = a(i)./bbar(i-1);
        abar(i) = a(i) - bbar(i-1).*multiplier;
        bbar(i) = b(i) - cbar(i-1).*multiplier;
        cbar(i) = c(i);
        dbar(i) = d(i) - dbar(i-1).*multiplier;
    end

    %initialize x of size n
    x = ones(1,n);

    %initialize end condition
    x(n) = dbar(n)/bbar(n);

    % Upward substitution AKA zip it up
    for i = n-1:-1:1
        x(i) = (dbar(i)-(cbar(i)*x(i+1)))/bbar(i);
    end
end
```




Problem 1a: Heat Equation with Backward Euler
x
t
0
0.5
1
1.5
2
2.5
3
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8


Problem 1a: Heat Equation with Crank Nicolson
x
t
0
0.5
1
1.5
2
2.5
3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0

Published with MATLAB® R2022b