ENGR 133, Lab 05

authored by: Rosalia N.

authored on: 10/29/2021

Exercise #1...Problem 4.19

Problem Statement

write a program that accepts a year and determines whether the year is a leap year. Use the mod function. The output should be the variable extra_day, which should be 1 if the year is a leap year and 0 otherwise. The rules for determining leap years in the Gregorian calendar are as follows:

1. All years evenly divisible by 400 are leap years.
2. Years evenly divisible by 100 but not by 400 are leap years.
3. Years divisible by 4 not by 100 are leap years.

For example, the years 1800,1900,2100,2300, and 2500 are not leap years, but 2400 is a leap year.

Solution

<u>Initialize variables</u>

```
clc,clear,close all
yearInput = inputdlg('Enter the year (4 digits): '); % asks for input
year = str2num(yearInput{1}); % converts year string to number
extra_day = 0; % initializes counter variable
```

<u>Perform calculations</u>

```
if mod(year,400) == 0 % performs logical test for rule 1
    extra_day = 1; % changes counter variable if true
elseif mod(year,100) == 0 % performs logical test for rule 2
    extra_day = 0; % changes counter variable if true
elseif mod(year,4) == 0 % performs logical test for rule 3
    extra_day = 1; % changes counter variable if true
else
    extra_day = 0; % changes counter variable if all tests false
end

if extra_day == 1
    output = 'is';
else
    output = 'is not';
end
```

Display results

```
fprintf('The year %4.0f %s a leap year. \n\n',year,output)
```

```
The year 1999 is not a leap year.
```

Exercise #2...Problem 4.24

Problem Statement

Write a program that uses a for loop to determine the time at which an object is the closest to the origin at (0,0). Determine also the minimum distance.

Problem Solving Approach(Pseudocode):

- Initialize a time vector over the interval of interest.
- Initialize vectors for the x and y coordinates over time.
- Compute an array of d values of the d vector -- answers the 2nd part.
- Find the minimum value of the d value -- answers the 1st part.
- Confirm that our solution is reasonable.

Solution

Initialize variables

```
clc,clear,close all
t = 0:0.01:4; % initialize time vector
x =  5*t - 10; % initialize x-coordinate vector
y = 25*t.^2-120*t +144; % initialize y-coordinate vector
d = sqrt(x.^2 + y.^2); % initialize distance vector
min_dist = Inf; % initialize placeholder for storing minimum distance
```

Perform calculations

```
for k = 1:length(t) % loop to look at every value in distance array
    if d(k) < min_dist % replace min_dist if lower than previous value
        min_dist = d(k); % replace min_dist if lower than previous value
        tmin = t(k); % replace time value
    end % terminate if branch
end % terminate for loop
```

Display results

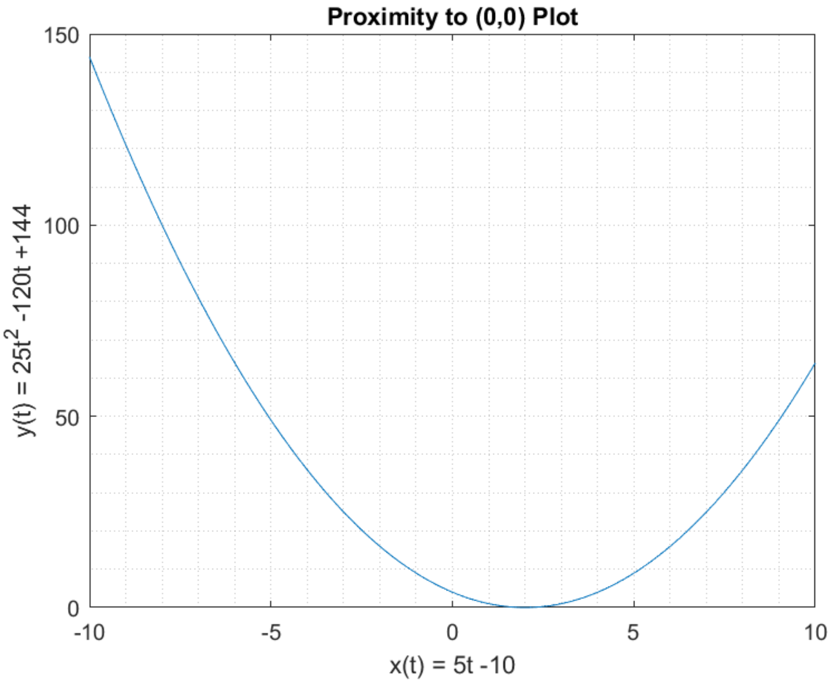
```
fprintf('The minimum distance is %6.4f. \n\n',min_dist)
```

```
The minimum distance is 1.3581.
```

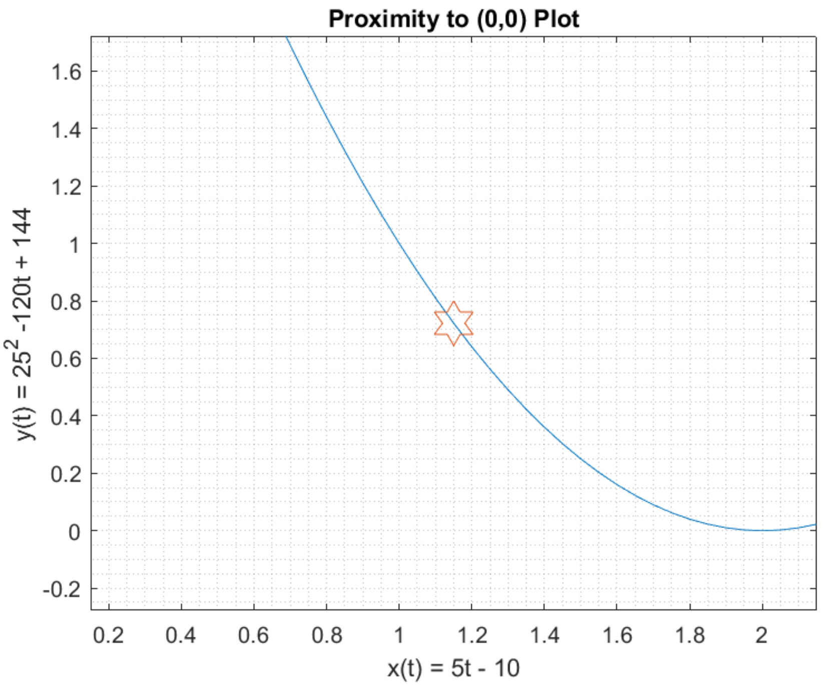
```
fprintf('It occurs at t = %6.4f seconds. \n\n',tmin)
```

```
It occurs at t = 2.2300 seconds.
```

```
plot(x,y), grid minor % plot the entire time range
title('Proximity to (0,0) Plot')
xlabel('x(t) = 5t -10'),ylabel('y(t) = 25t^2 -120t +144')
```



```
[a, b] = min(d); % store the minimum distance and its index
figure % open a new figure window
plot(x,y), grid minor % plot the entire time range
hold on % configure figure window 2
axis([x(b)-1,x(b)+1,y(b)-1,y(b)+1]) % set axes close to min_dist points
plot(x(b),y(b),'h','markersize',20) %  add marker for min_dist points
title('Proximity to (0,0) Plot')
xlabel('x(t) = 5t - 10'), ylabel('y(t) = 25^2 -120t + 144')
```



Exercise #3...Problem 4.32

Problem Statement

- Initialize variables
- Perform calculations
- Display results

Problem Solving Approach (Pseudocode):

Perform calculations

Display results

Solution

Initialize variables

```
clc,clear,close all
x = 0; y = 0; % x represents time, y represents force
k = 0; % k is the loop counter
```

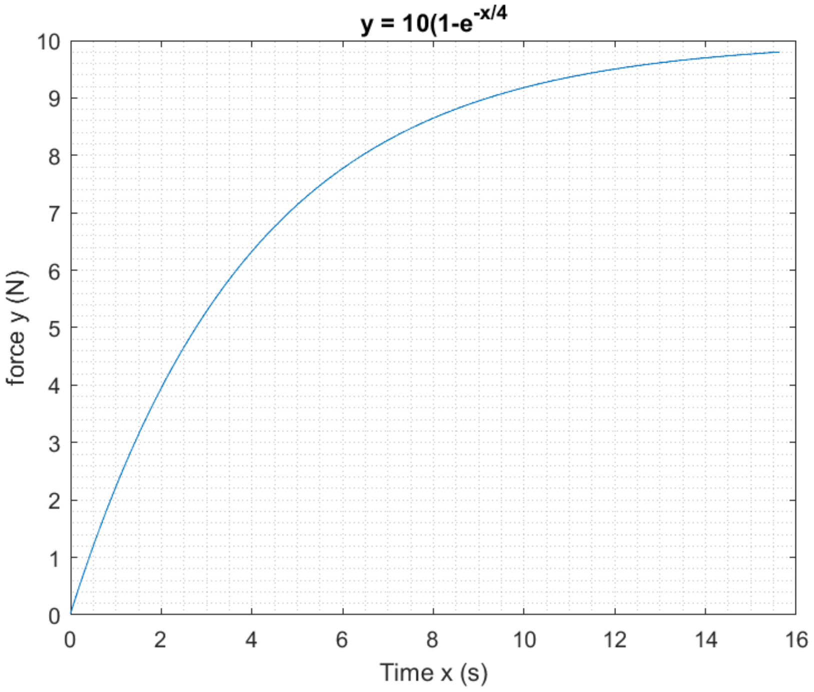
Perform calculations

```
while y<9.8 % the upper limit of force is used as a condition
    k = k + 1; % loop counter is incremented
    x = x + 0.01; % x variable is incremented
    y(k) = 10*(1-exp(-x/4)); % new y array element is computed
```

```
end
xmax = x; % the upper limit of time is set
t = 0:0.01:xmax; % time array is computed
```

Display results

```
plot(t,y),grid minor,xlabel('Time x (s)'),ylabel('force y (N)')
title('y = 10(1-e^{-x/4}')
```


$y = 10(1-e^{-x/4}$
force y (N)
Time x (s)