Chapter 4: Programming with MATLAB

Topics Covered:

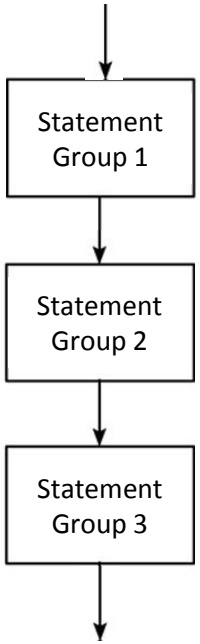
- Programming Overview
- Relational Operators and Logical Variables
- Logical Operators and Functions
- Conditional Statements
- For Loops
- While Loops
- Debugging MATLAB programs

Programming Overview:

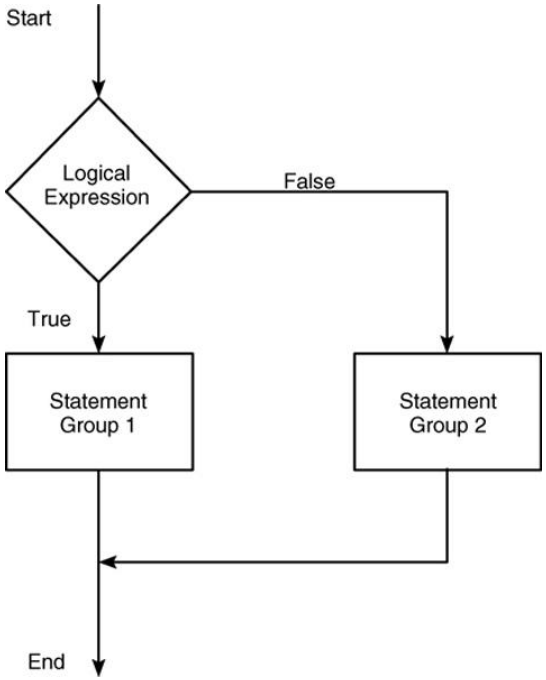
MATLAB programming can be used to solve very computationally intensive problems which may require thousands or hundreds of thousands of calculations. Operations can be:

- **Sequential**: Calculations are executed in order from the top down.
- **Conditional**: Calculations are made based on the answer (either true or false) to a question.
- **Iterative**: Calculations are made over and over until a condition is met.

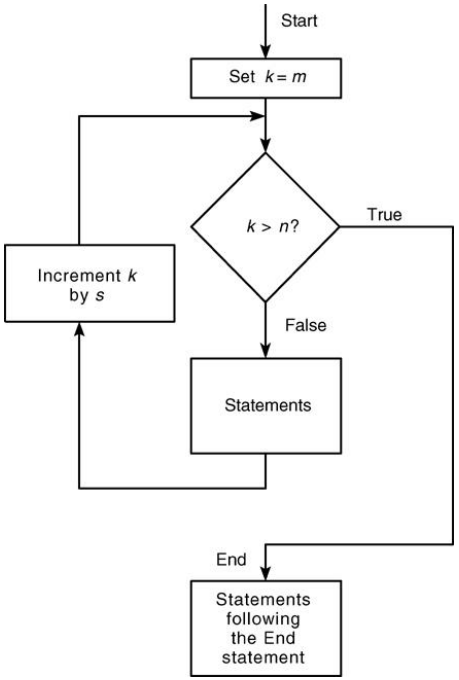
Sequential



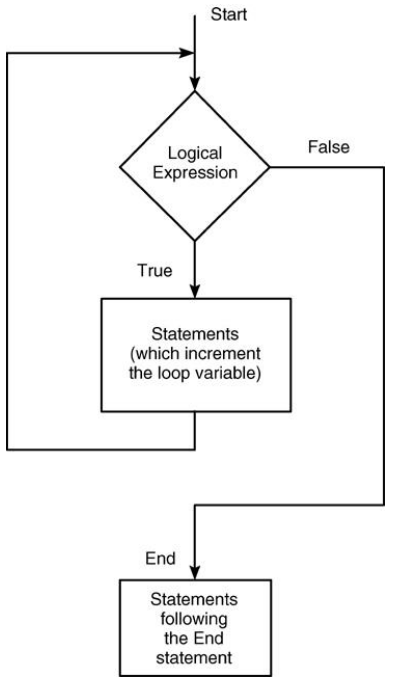
Conditional (If-Then)



Iterative (For Loops)



Iterative (While Loops)



Relational Operators and Logical Variables:

Relational Operators make comparisons between numbers or arrays. The result of a comparison is either:

- = 0 (if the comparison is *false*) or
- = 1 (if the comparison is *true*)

The result can be used as a variable. When used to compare arrays, the relational operators compare the arrays on an element-by-element basis. The arrays must have the same dimension. When comparing an array to a scalar, all of the elements of the array are compared to the scalar.

Table 4.2–1 Relational operators

Relational operator	Meaning
<	Less than.
<=	Less than or equal to.
>	Greater than.
>=	Greater than or equal to.
==	Equal to.
~=	Not equal to.

Problem 4.4:

Suppose that $x = 6$. Find the results of the following operations by hand and use MATLAB to check your results.

a. `z = (x < 10)`

b. `z = (x == 10)`

c. `z = (x >= 4)`

d. `z = (x ~= 7)`

Editor - C:\Laptop Backup\matlab\Homework Solutions\Chap

Command Window

```
Problem 4.4: Scott Thomas
Part a: 6<10 (less than?)

z =

     1

Part b: 6==10 (equal to?)

z =

     0

Part c: 6>=4 (greater than or equal to?)

z =

     1

Part d: 6~=7 (not equal to?)

z =

     1
```

Problem 4.6:

Suppose that $x = [10, -2, 6, 5, -3]$ and $y = [9, -3, 2, 5, -1]$. Find the results of the following operations by hand and use MATLAB to check your results.

a. `z = (x < 6)`
b. `z = (x <= y)`
c. `z = (x == y)`
d. `z = (x ~= y)`

Command Window

```
Problem 4.6: Scott Thomas

x =

    10    -2     6     5    -3

y =

     9    -3     2     5    -1

Part a: z = (x < 6)

z =

     0     1     0     1     1

Part b: z = (x <= y)

z =

     0     0     0     1     1

Part c: z = (x == y)

z =

     0     0     0     1     0

Part d: z = (x ~= y)

z =

     1     1     1     0     1
```

Problem 4.8:

The array `price` given below contains the price in dollars of a certain stock over 10 days. Use MATLAB to determine how many days the price was above $20.

```
price = [19, 18, 22, 21, 25, 19, 17, 21, 27, 29]
```

Use the **find** and **length** commands

What values are over 20? Use MATLAB to find them.

```
Problem 4.8: Scott Thomas

price =

    19    18    22    21    25    19    17    21    27


z =

  1×10 logical array

   0   0   1   1   1   0   0   1   1   1


zz =

     3     4     5     8     9    10


number_of_days =

     6


values_above_20 =

    22    21    25    21    27    29
```

Logical Operators and Functions:

MATLAB has five **Logical Operators** (or **Boolean Operators**).

Table 4.3–1 Logical operators

Operator	Name	Definition
~	NOT	~A returns an array of the same dimension as A; the new array has 1s where A is 0 and 0s where A is nonzero.
&	AND	A & B returns an array of the same dimension as A and B; the new array has 1s where both A and B have nonzero elements and 0s where either A or B is 0.
|	OR	A | B returns an array of the same dimension as A and B; the new array has 1s where at least one element in A or B is nonzero and 0s where A and B are both 0.
&&	Short-Circuit AND	Operator for scalar logical expressions. A && B returns true if both A and B evaluate to true, and false if they do not.
||	Short-Circuit OR	Operator for scalar logical expressions. A || B returns true if either A or B or both evaluate to true, and false if they do not.

Problem 4.12:

The height and speed of a projectile (such as a thrown ball) launched with a speed of v_0 at an angle A to the horizontal are given by

$$h(t) = v_0 t \sin A - 0.5\, g t^2$$

$$v(t) = \sqrt{v_0^2 - 2 v_0 g t \sin A + g^2 t^2}$$

where g is the acceleration due to gravity. The projectile will strike the ground when $h(t) = 0$, which gives the time to hit $t_{\text{hit}} = 2(v_0/g) \sin A$.

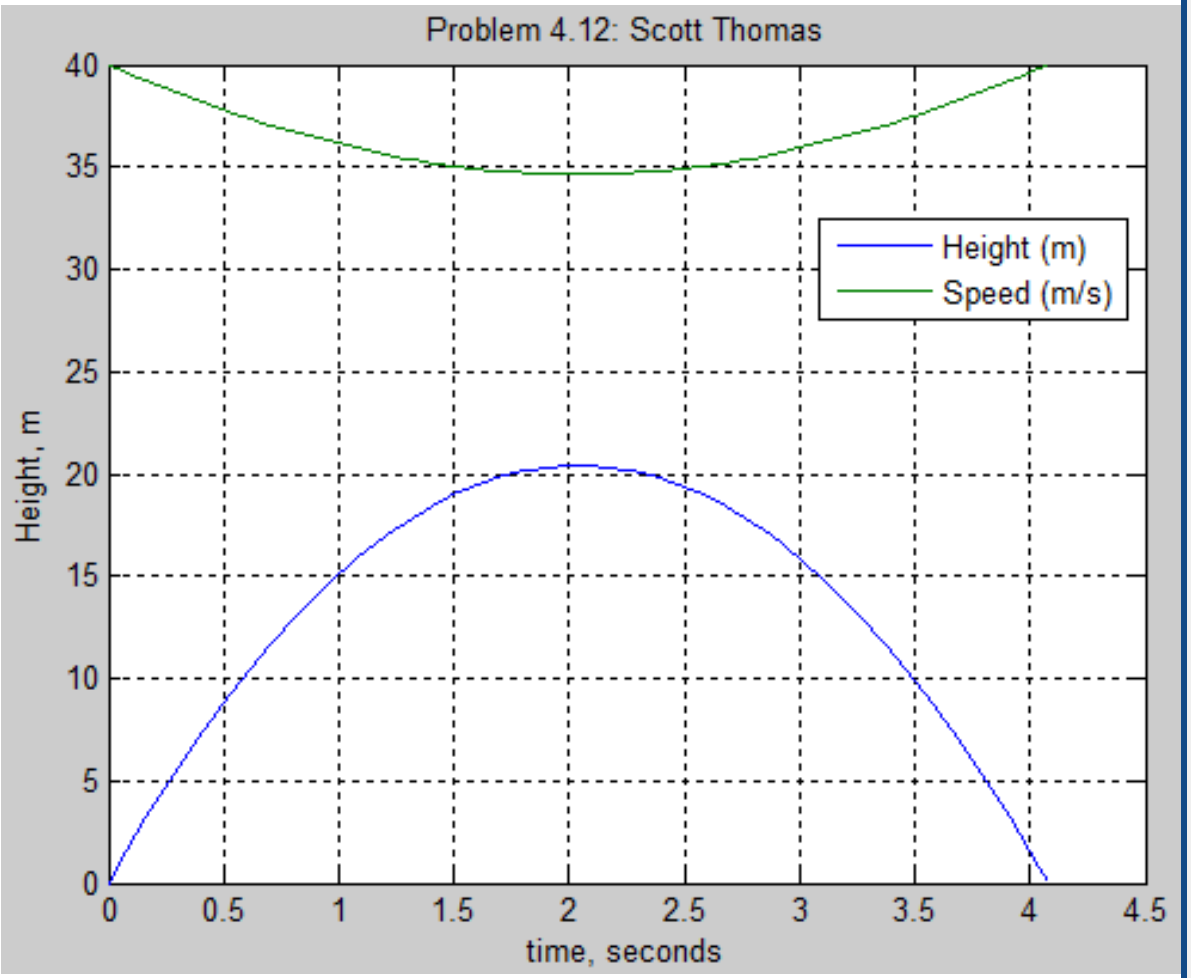
Suppose that $A = 30°$, $v_0 = 40$ m/s, and $g = 9.81 \text{m/s}^2$. Use the MATLAB relational and logical operators to find the times when

a. The height is no less than 15 m.

b. The height is no less than 15 m and the speed is simultaneously no greater than 36 m/s.

Problem 4.12:

Step 1: Create plots of the height and speed versus time.



Command Window

```
Problem 4.12: Scott Thomas

A_deg =

    30


A_rad =

    0.5236


v_0 =

    40


g =

    9.8100


tmax =

    4.0775
```

Problem 4.12:

Step 2: Use the MATLAB relational and logical operators to find the times when the height is no less than 15 m.

```
time_above_15 = t(height>=15)
t_initial_a = time_above_15(1)
t_final_a = time_above_15(length(time_above_15))
```

```
t_initial_a =

    1.0297


t_final_a =

    3.0478
```

Problem 4.12:

Step 3: Use the MATLAB relational and logical operators to find the times when the height is no less than 15 m and the speed is simultaneously no greater than 36 m/s.

```
time_above_15_under_36 = t(height>=15&speed<=36);
t_initial_b = time_above_15_under_36(1)
t_final_b = time_above_15_under_36(length(time_above_15_under_36))
```

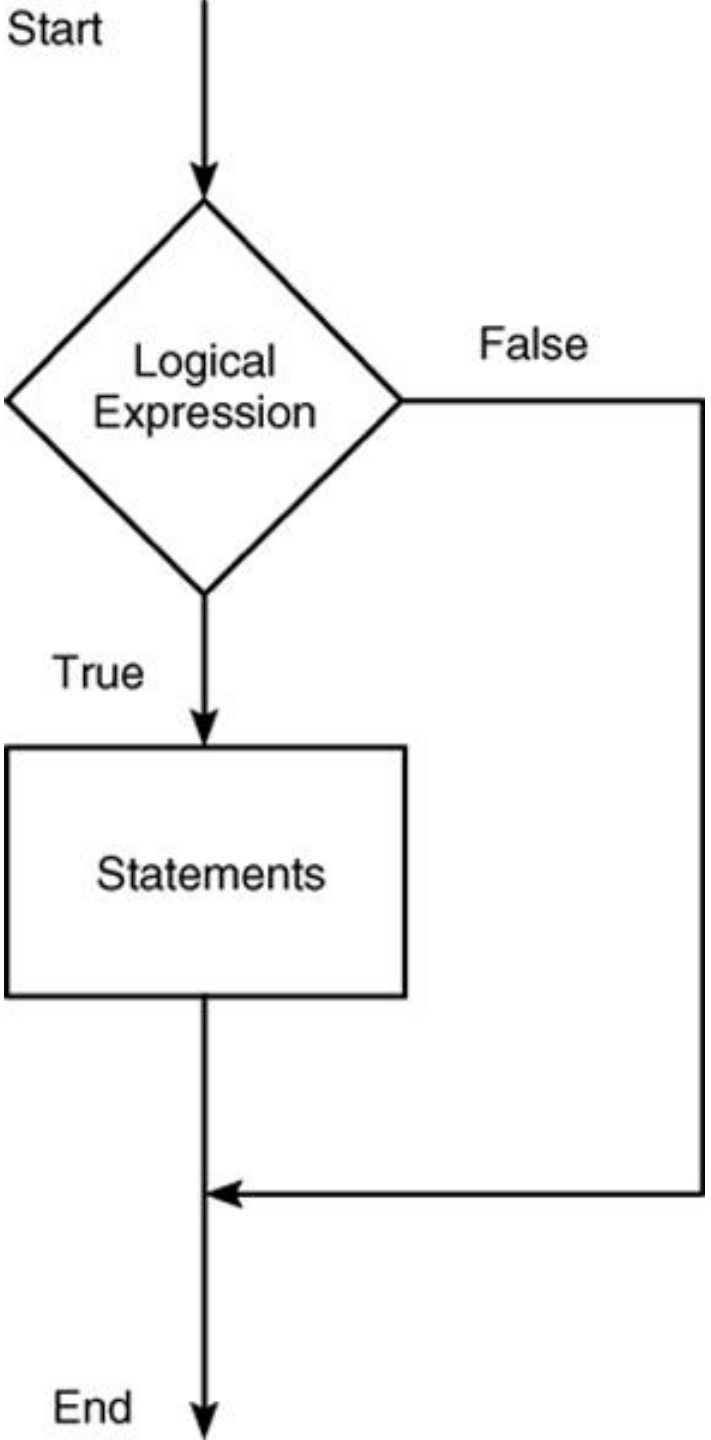
```
t_initial_b =

    1.0709


t_final_b =

    3.0066
```

If Statements:

The **Conditional Operators** (**If Statements**) use **Relational Operators**:

<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
==	Equal to
~=	Not equal to

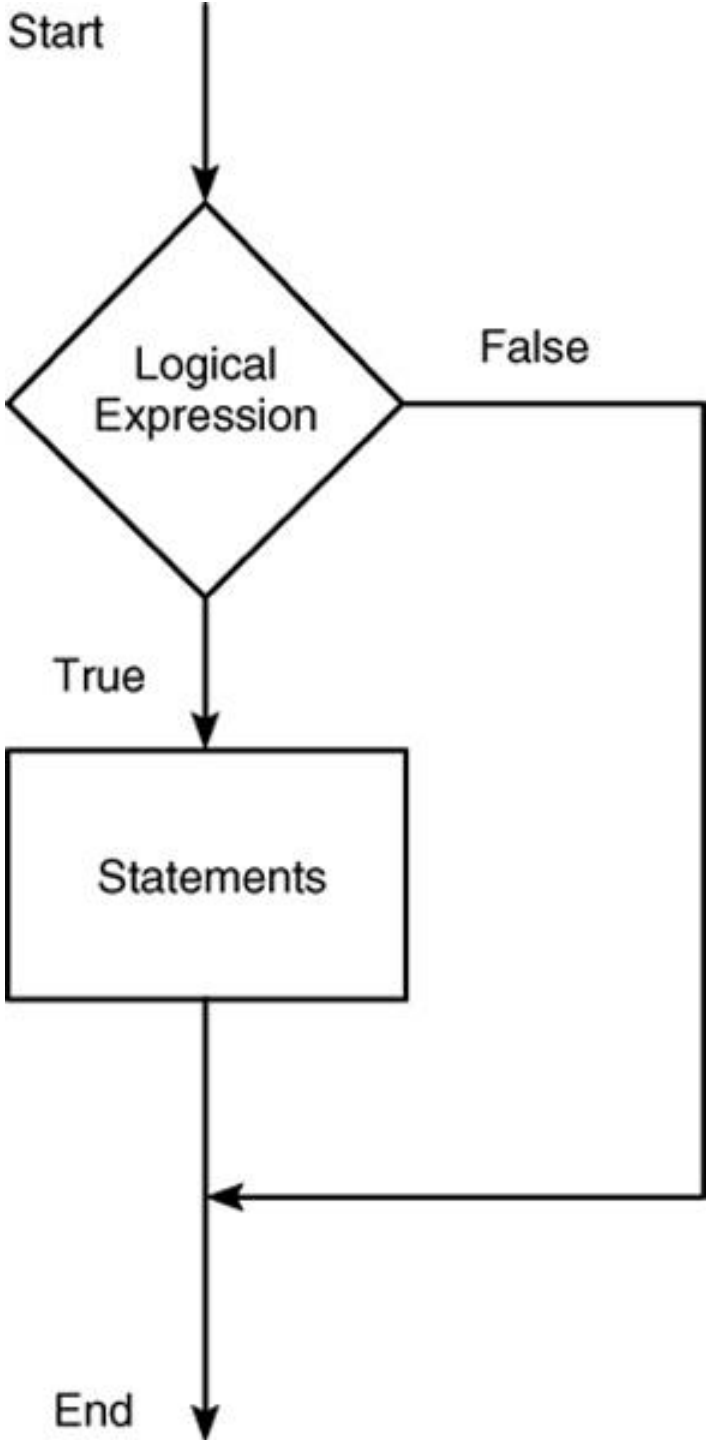
The result of these **Relational Operators** is either **True** or **False** (1 or 0). This can be used to control the flow of a program. This is called **Logic Flow**, which can be represented by a **Flowchart**.



If Statements:

Create the following MATLAB program. Once you've checked that it is working correctly, switch the values of **x** and **y**.

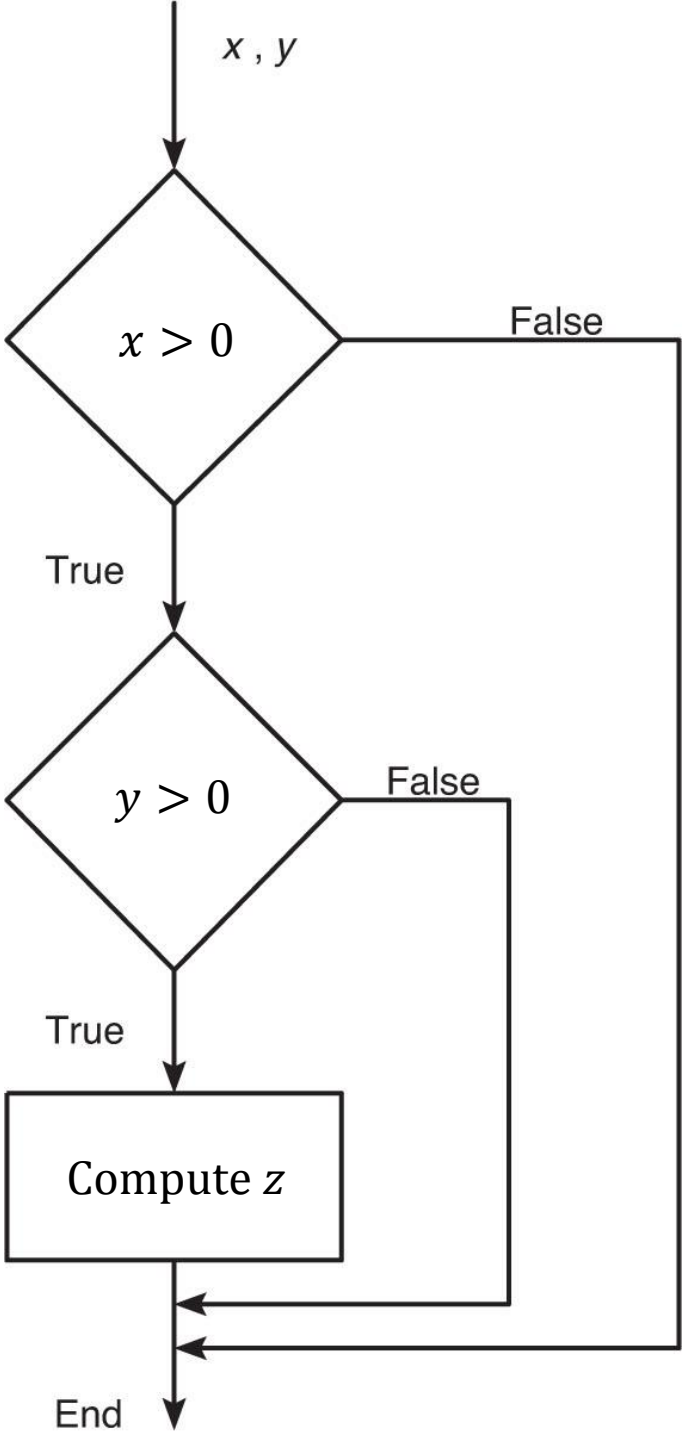
```
x = 3.5
y = 9.6
if x < y
    disp('x is less than y')
    z1 = y - x
end
z2 = x - y
```



Nested If Statements:

Create the following MATLAB program. In order to compute the value of **z**, both **Logical Expressions** must return a **True** value. Once you've checked that it is working correctly, change the values of *x* and *y* to zero.

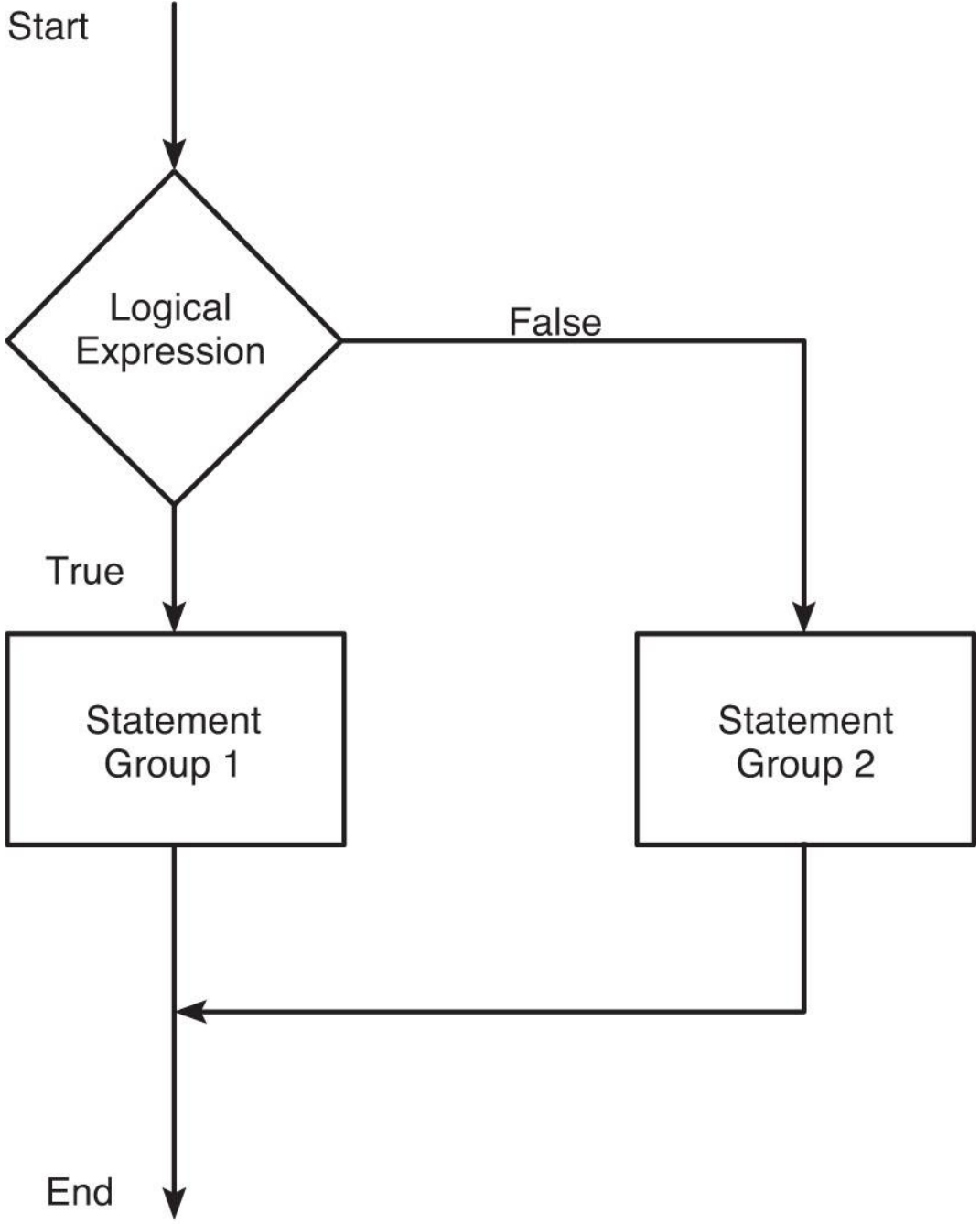
```
x = 5
y = 10
if x>0
    if y>0
        z = x/y;
    end
end
z
```



If-Else Statements:

If-Else Statements provide two options based on the result of the **Logical Expression**. Create the following MATLAB program. Once you've checked that it is working correctly, change the value of **x** to 50.

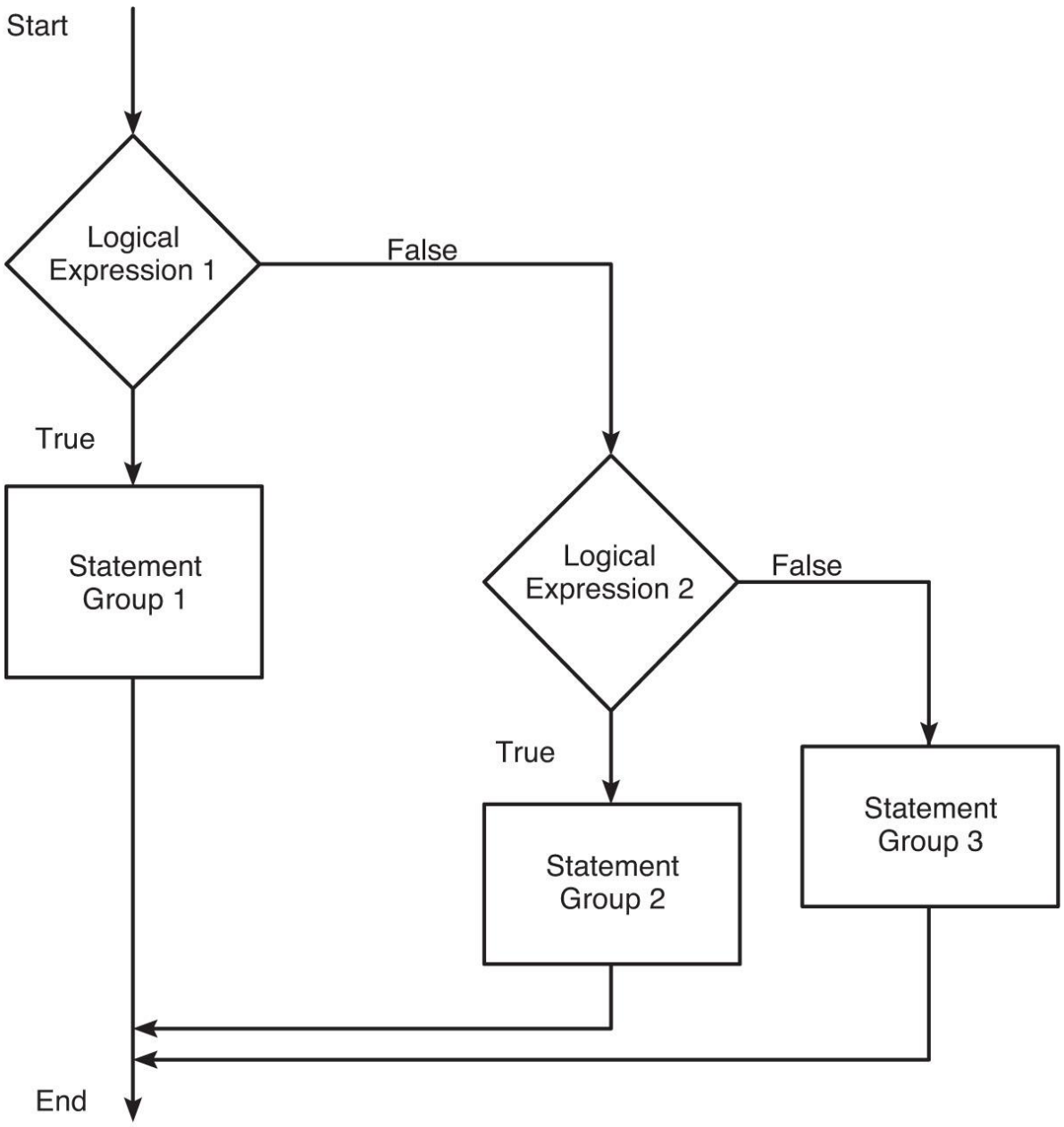
```
4 -     x = 5
5 -     y = 10
6 -     if x<y
7 -         z = x*y
8 -     else
9 -         z = x/y
10 -    end
```



If-Elseif-Else Statements:

This is the **General Form** of the **if** statement. Create the following MATLAB program. Once you've checked that it is working correctly, change the value of **x** to 5 and then to 7.

```
x = -1
y = 2
if x<0
    z = x*y
elseif x>= 0 & x< 6
    z = x/y
else
    z = x^y
end
```



Problem 4.16:

Write a script file using **Conditional If-Elseif-Else** statements to evaluate the following function, assuming that $x = -2,\ 0,$ and 6. The function is:

$$y = \begin{cases} e^{x+1} & \text{for } x < -1 \\ 2 + \cos(\pi x) & \text{for } -1 \le x \le 5 \\ 10(x-5) + 1 & \text{for } x > 5 \end{cases}$$

Command Window

```
Problem 4.16: Scott Thomas

x =

    -2


y =

    0.3679
```

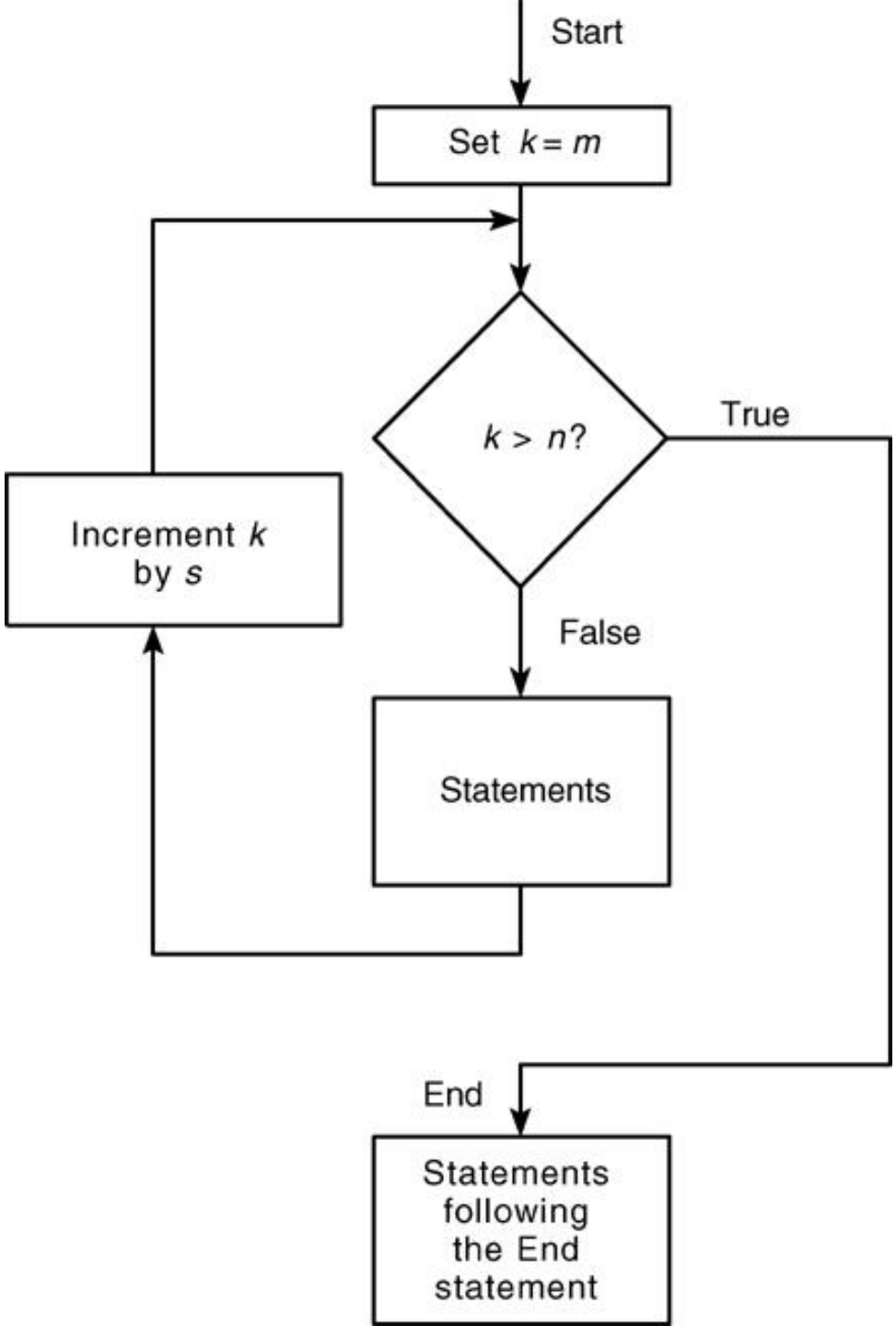
```
x =

     0


y =

     3
```

```
x =

     6


y =

    11
```

For Loop:

The **For Loop** is a structure that repeats a set of commands or calculations a specified number of times. Create the following MATLAB program. In this case, the variable **x** is a scalar.

```
4 —     x = 3.
5 —     dx = 0.25
6 —     for k = 1:3
7 —         k
8 —         x = x + dx
9 —     end
```

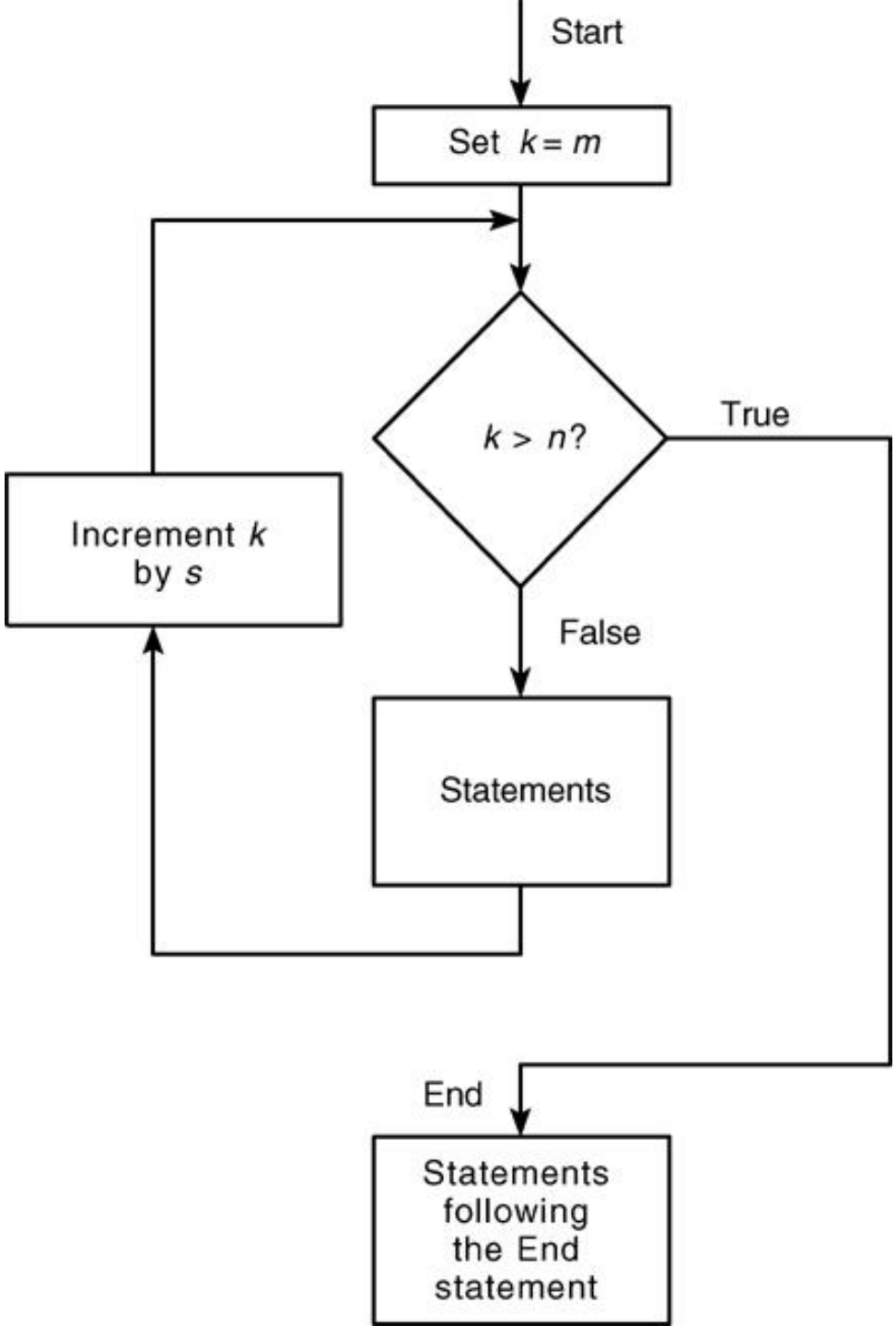


For Loop:

The **For Loop** can be used to **Load** a **Vector** with values. The **Counter k** is used to **Index** through the vector elements.

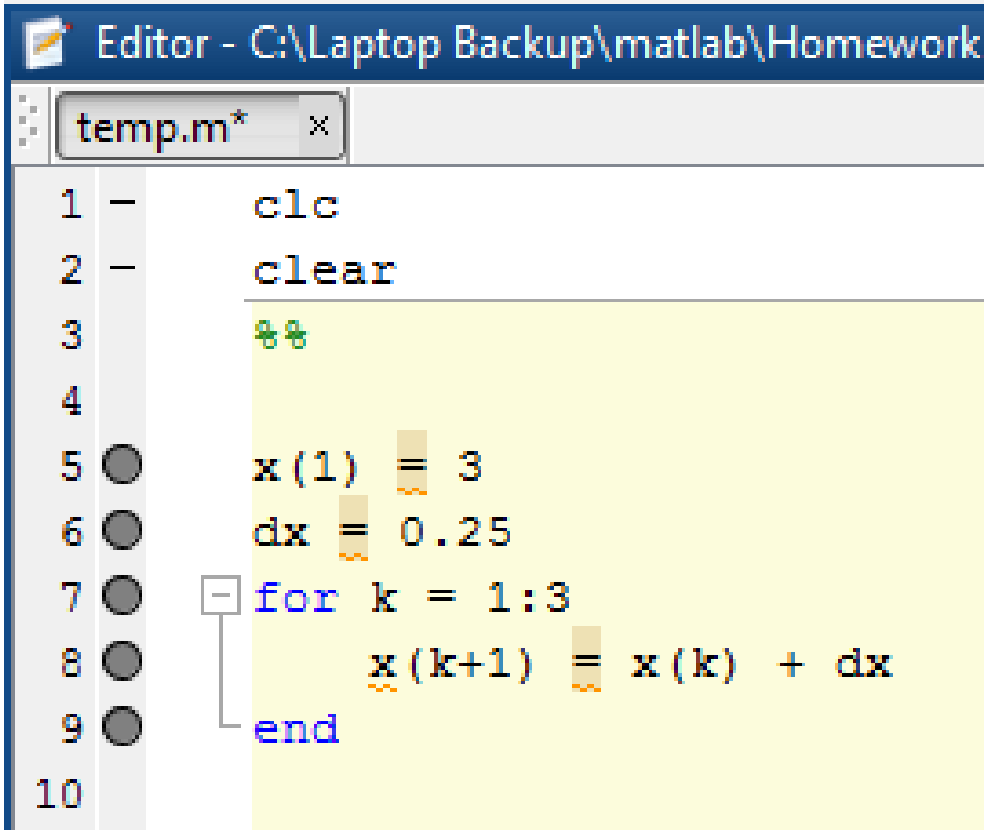
- What happens if you don't **Initialize x**? [Comment out **x(1) = 3** and rerun]
- How many elements does **x** have?
- What change would you make if you wanted **x** to have only 3 elements?

```
x(1) = 3
dx = 0.25
for k = 1:3
    x(k+1) = x(k) + dx
end
```

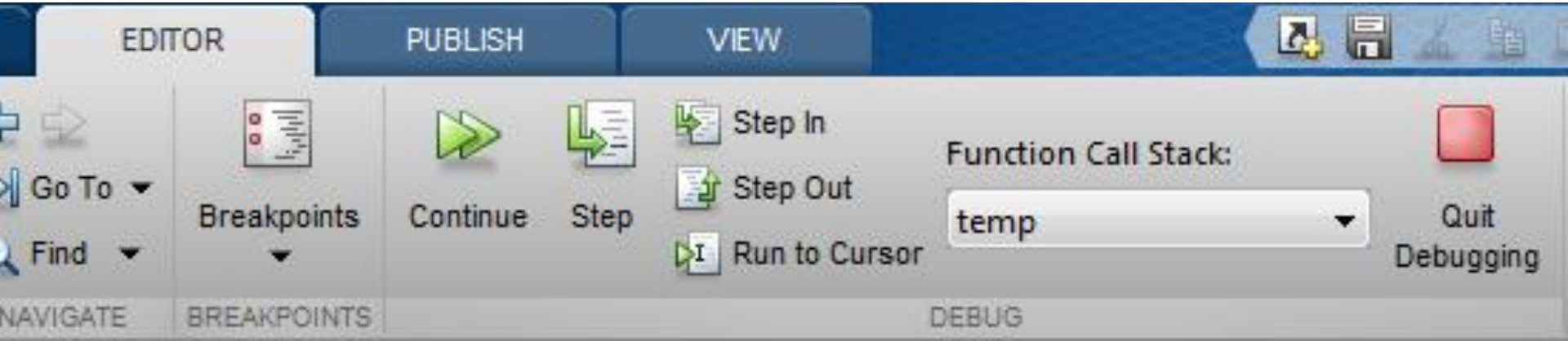


Debugging Tool:

Click on the **Horizontal Lines** to the right of the line numbers. This creates **Breakpoints** on the lines of code.

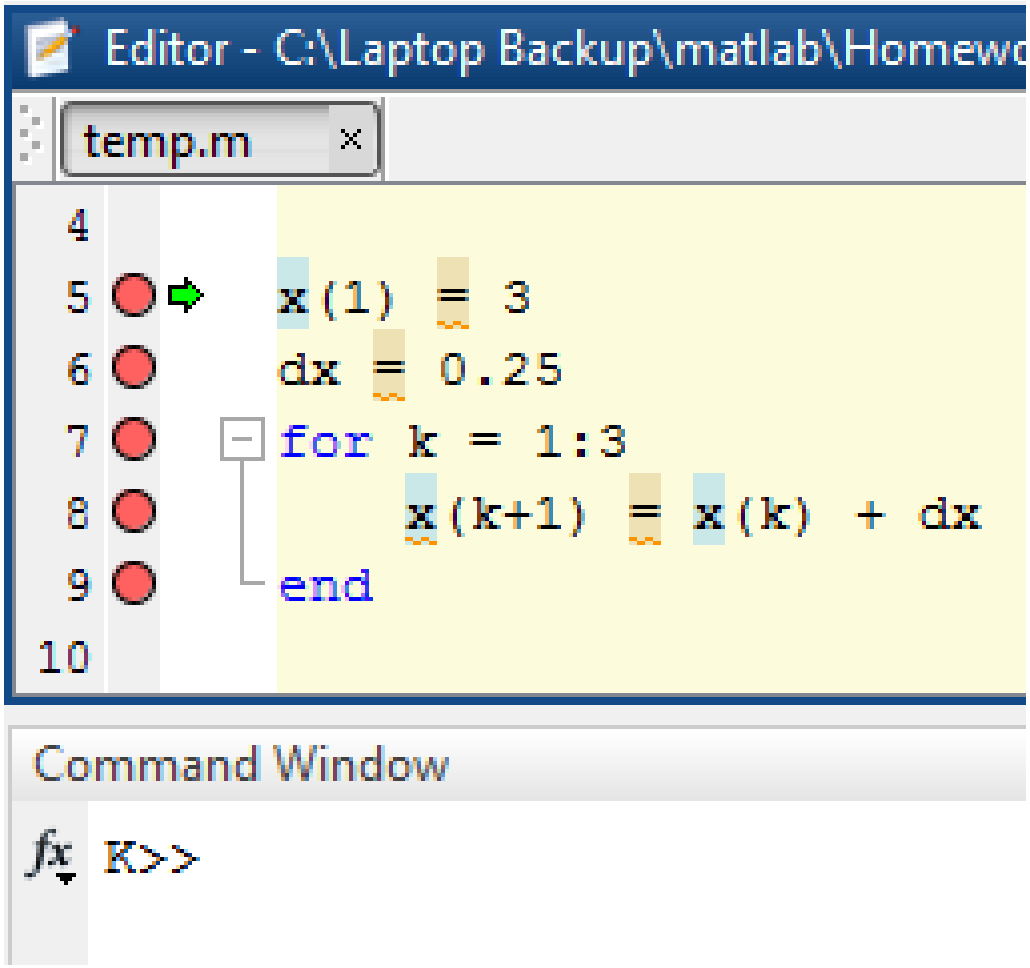


When you hit **Run**, the **Editor Bar** changes to the following:



Debugging Tool:

The **Editor Window** and the **Command Window** change as well. The **Green Arrow** indicates the line that is going to be executed next:

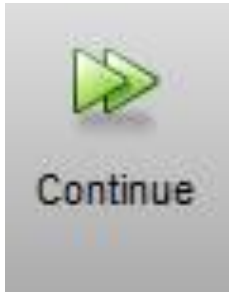


Debugging Tool:

Click on the **Continue** button to step through the program. The values of the variables appear in the **Command Window**.

Notice how the vector **x** is loaded as you step through the **for** loop.

You can stop execution and **Debugging** by clicking on the red **Quit Debugging** button.



Editor - C:\Laptop Backup\matlab\Homewo

temp.m

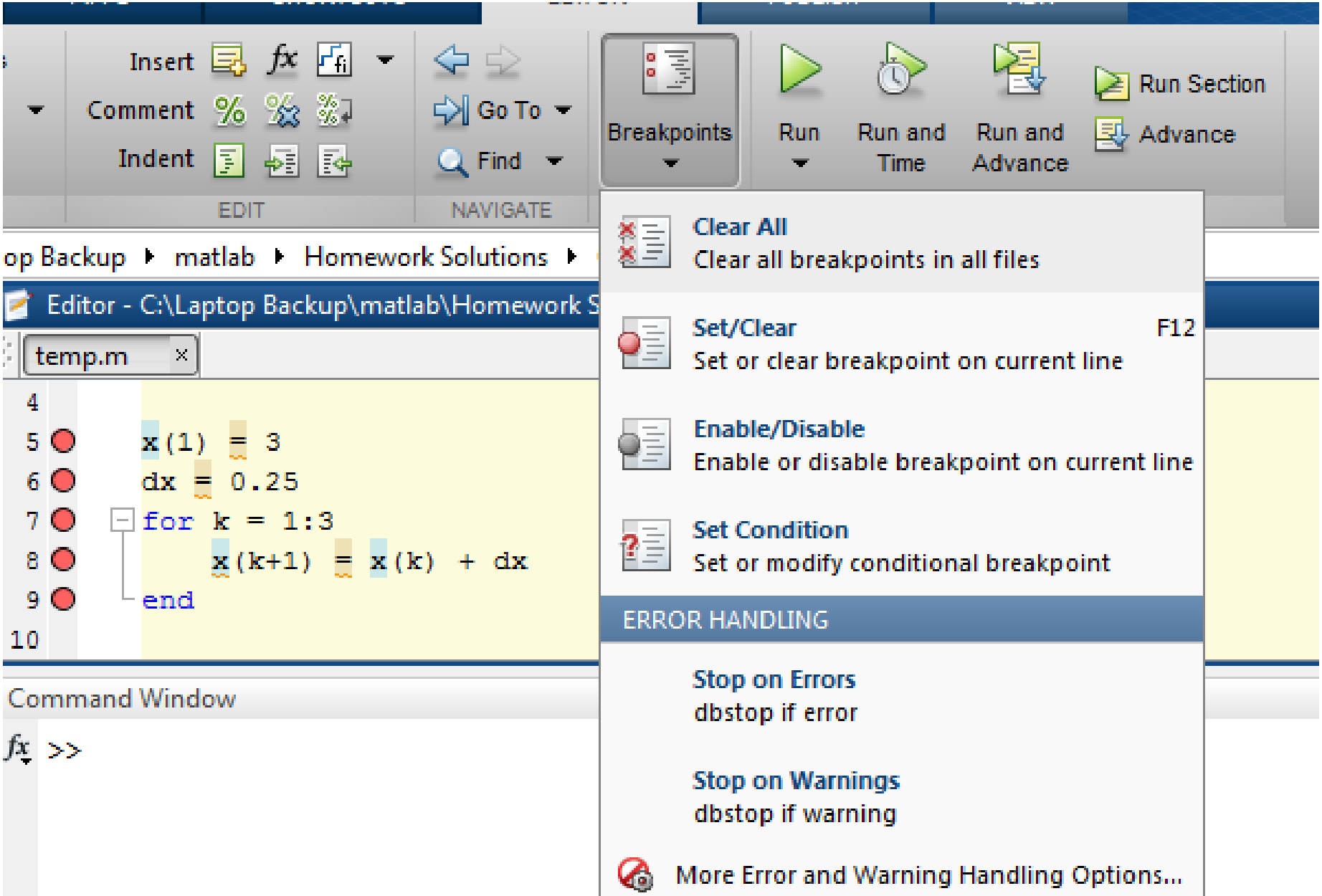
```
4
5      x(1) = 3
6      dx = 0.25
7      for k = 1:3
8          x(k+1) = x(k) + dx
9      end
10
```

Command Window

```
x =

     3


dx =

    0.2500

K>>
```

Debugging Tool:

Click on the **Breakpoints/Clear All** tab to delete all of the breakpoints.



Problem 4.22:

Use a `for` loop to determine the sum of the first 10 terms in the series $5k^3$, $k = 1, 2, 3, \ldots, 10$.

Command Window

```
Problem 4.22: Scott Thomas

The sum is:

ysum =

       15125
```

Problem 4.27:

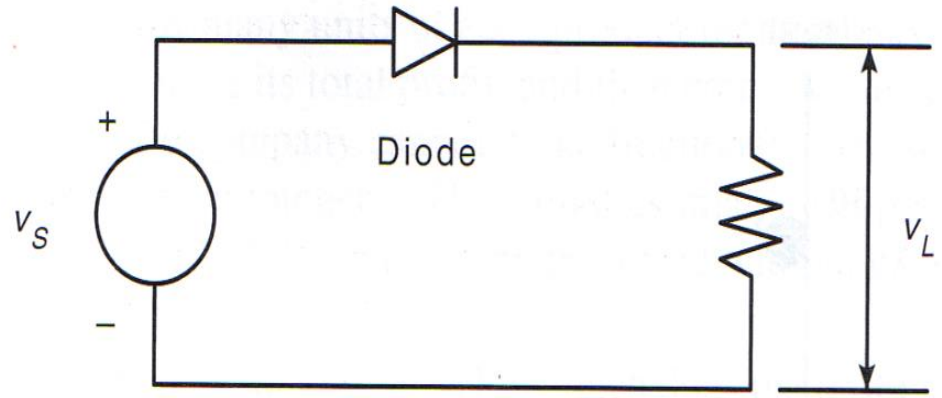
a. An *ideal* diode blocks the flow of current in the direction opposite that of the diode's arrow symbol. It can be used to make a *half-wave rectifier* as shown in Figure P27a. For the ideal diode, the voltage v_L across the load R_L is given by

$$v_L = \begin{cases} v_S & \text{if } v_S > 0 \\ 0 & \text{if } v_S \leq 0 \end{cases}$$

Suppose the supply voltage is

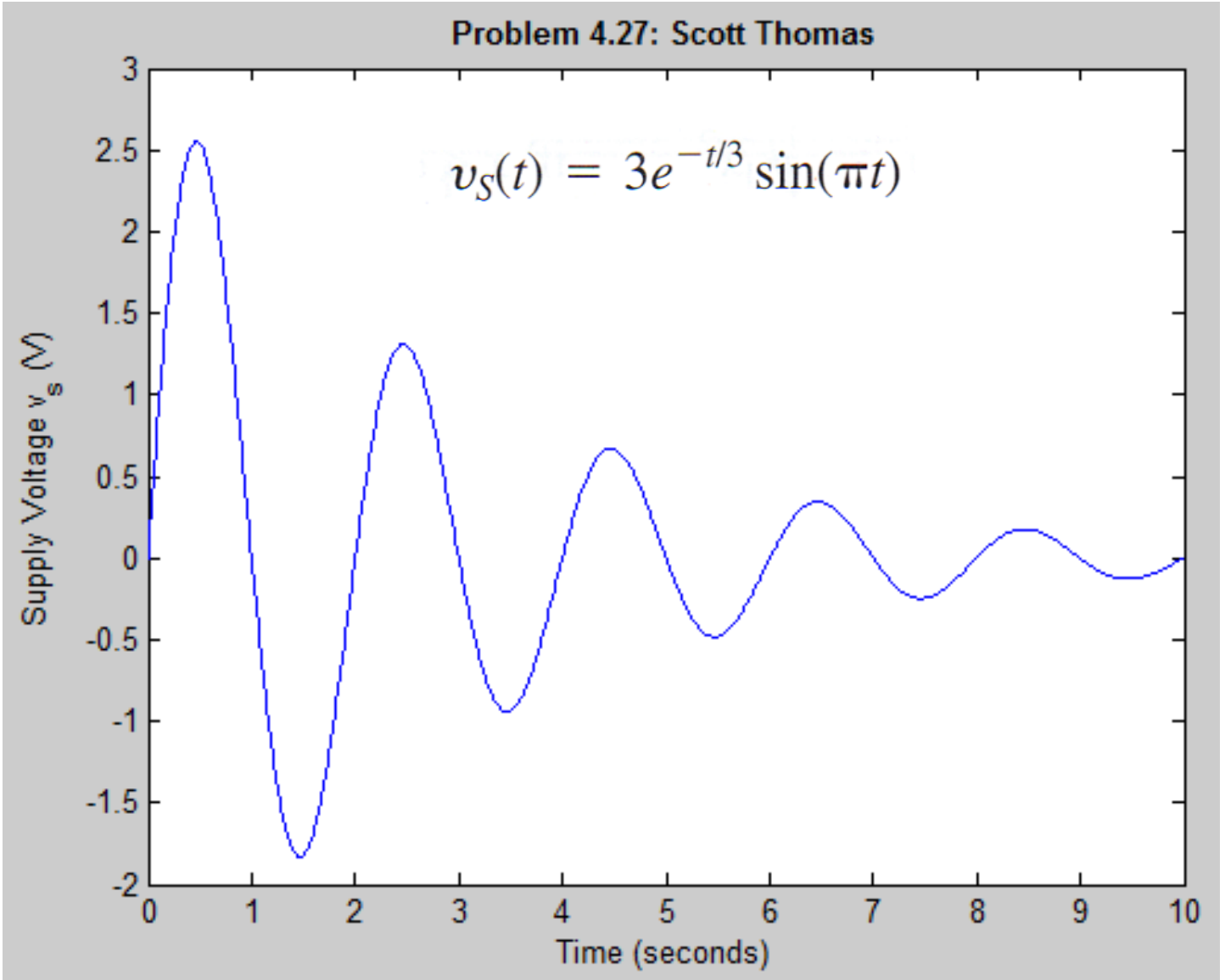
$$v_S(t) = 3e^{-t/3}\sin(\pi t) \qquad \text{V}$$

where time t is in seconds. Write a MATLAB program to plot the voltage v_L versus t for $0 \leq t \leq 10$.



Problem 4.27:

Step 1: Create a vector for **Time t**, use it to calculate the **Supply Voltage v_s**, and plot **v_s** versus **t**.



Problem 4.27:

Step 2: Create an **if** statement that can calculate the **Load Voltage** for a given **Supply Voltage**. Test the **if** statement at the following time values:

$$v_L = \begin{cases} v_S & \text{if } v_S > 0 \\ 0 & \text{if } v_S \leq 0 \end{cases}$$

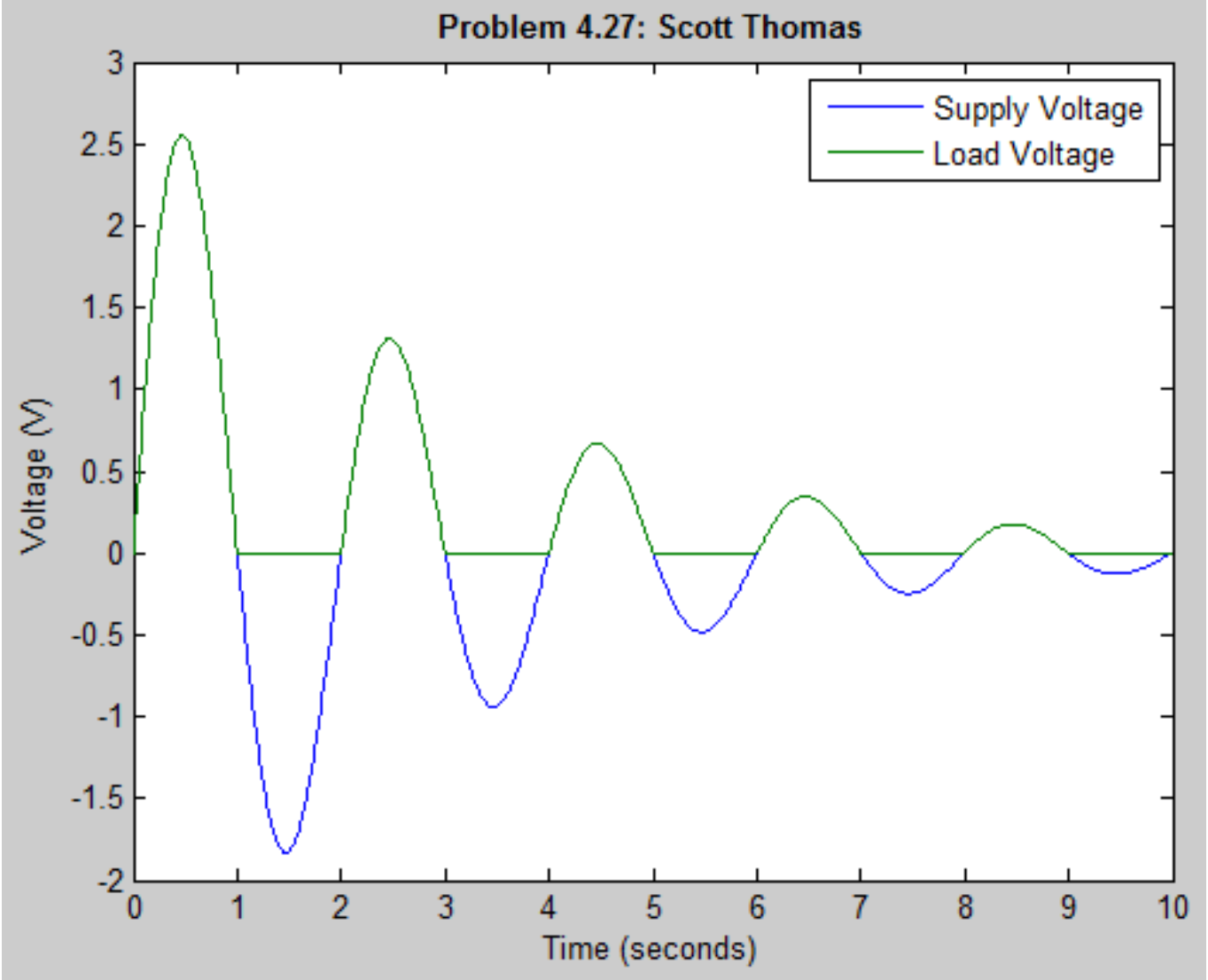
$$v_S(t) = 3e^{-t/3}\sin(\pi t)$$

```
t =

    0.5000

vs =

    2.5394

vl =

    2.5394
```

```
t =

    1.5000

vs =

   -1.8196

vl =

     0
```

Problem 4.27:

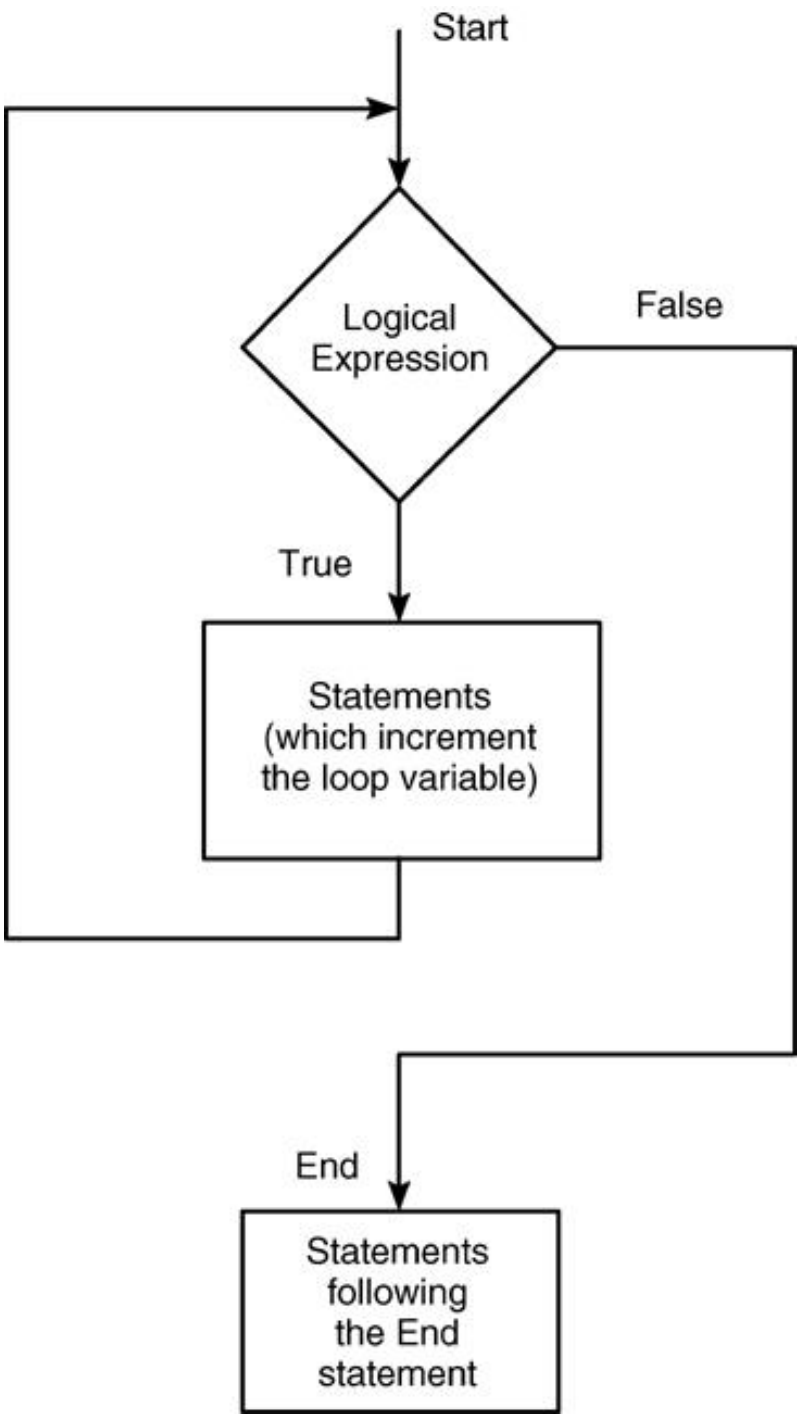
Step 3: Use a **for** loop with an **if** statement to load the **Load Voltage** vector **v_l**. Plot both the supply voltage **v_s** and the load voltage **v_l** versus **t**.



While Loops:

The **While Loop** is a structure that repeats a set of commands or calculations until the **Logical Expression** condition is met. The number of iterations through the loop is unknown prior to starting the program. Create the following MATLAB program. Use the **Debugging Tool** to step through the program. In this case, the variable **x** is a scalar.

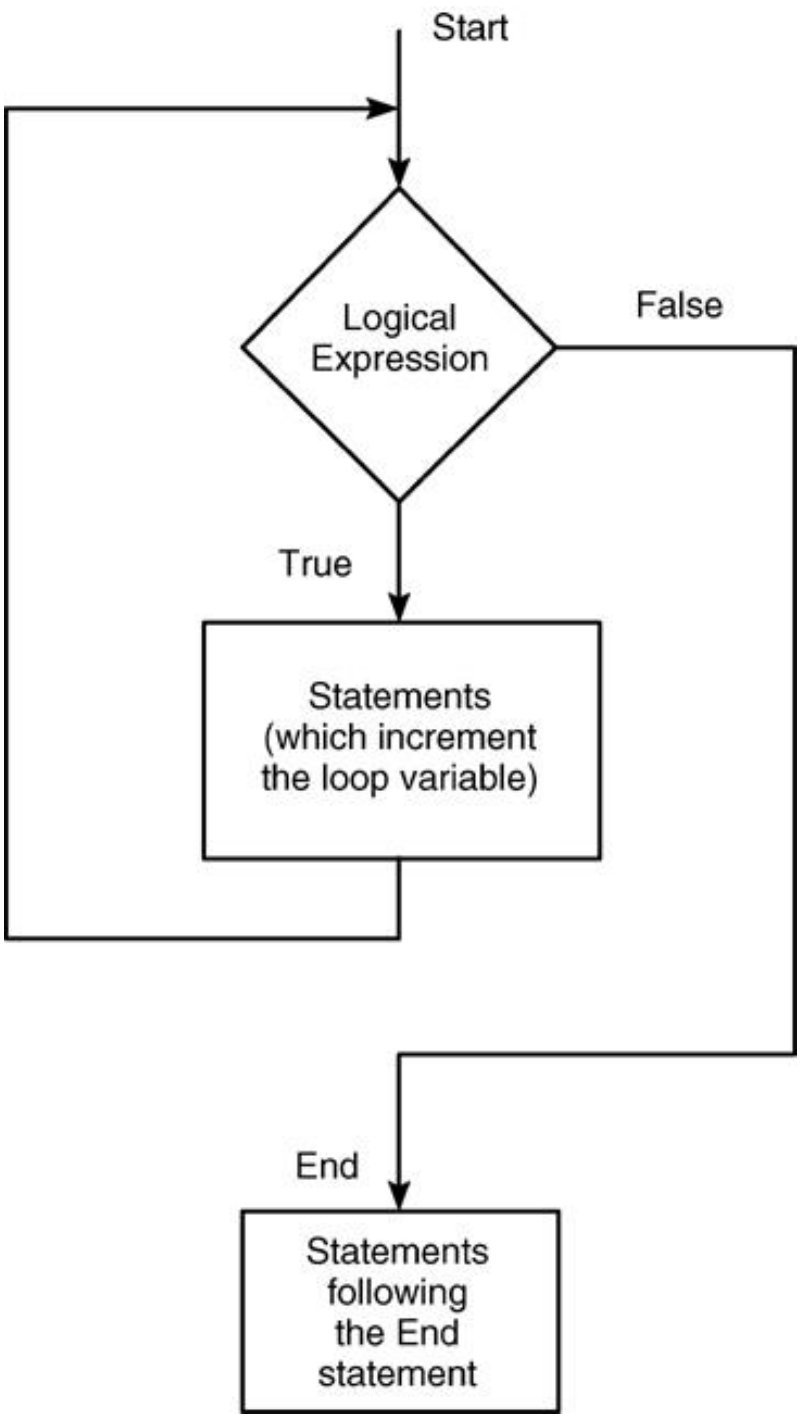
```
x = 3
dx = 0.25
k = 1
while x < 3.75
    x = x + dx
    k = k + 1
end
```



While Loops:

Change the **While Loop** as shown below. Use the **Debugging Tool** to step through the program. In this case, the variable **x** is a **Vector**. Notice how the **Counter k** is used to **Load** the **x Vector**.

```
x(1) = 3
dx = 0.25
k = 1
while x(k) < 3.75
    x(k+1) = x(k) + dx
    k = k + 1
end
```



Problem 4.32:

Use a `while` loop to determine how many terms in the series 2^k, $k = 1, 2, 3, \ldots$, are required for the sum of the terms to exceed 2000. What is the sum for this number of terms?

Command Window

```
Problem 4.32: Scott Thomas

k =

    10


sumk =

        2046
```

Example Problem:

Write a **While Loop** to plot the following function over the range $-2 \leq x \leq 6$.

$$y = \begin{cases} e^{x+1} & \text{for } x < -1 \\ 2 + \cos(\pi x) & \text{for } -1 \leq x \leq 5 \\ 10(x-5) + 1 & \text{for } x > 5 \end{cases}$$