GERMANNA
ACADEMIC CENTER
FOR EXCELLENCE

MATLAB Basics

MATLAB is an important and useful engineering mathematical tool. This handout will cover the topics of writing functions, plotting data, and creating loops in MATLAB.

You can navigate to specific sections of this handout by clicking the links below.

User-Defined Functions

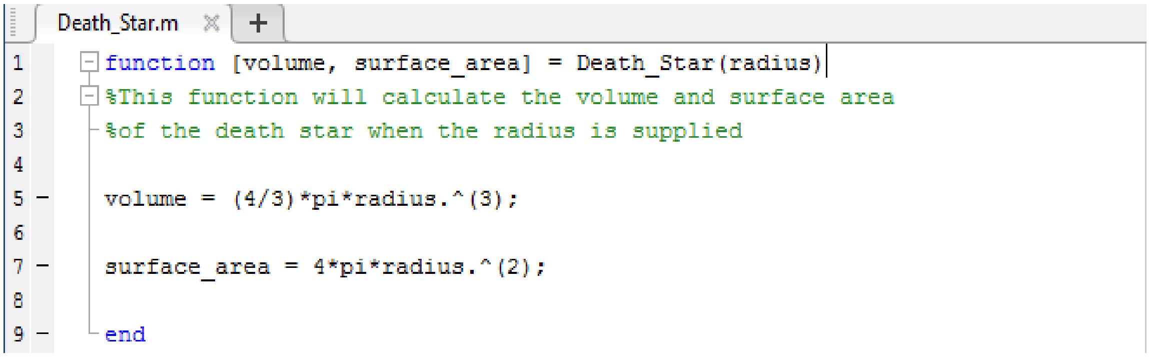
Functions are saved calculations that can continue to be used at any time. The structure of a function is:

```
function [ output ] = Function_Name( input )
Commands for output
end
```

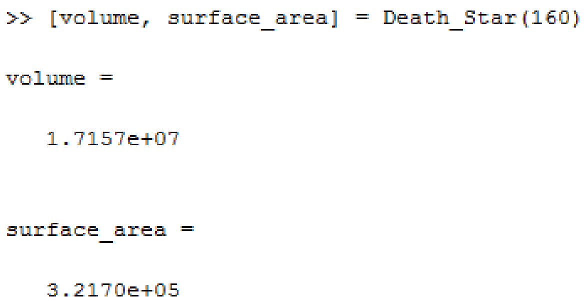
Once the function has been written and saved, it can be used at any time. First, to use the function or to call on a function, make sure the folder where the function is saved is open in the current folders window. Next, type out the function name with the inputs in parenthesis. A number can be typed into the function as an input or a set variable name. If there are multiple output variables, separate them with commas. The output data for the following examples are saved as *volume* and *surface_area*.

Example:

Write a function that will calculate the volume and surface area of the Death Star when the radius is used as an input. The radius of the Death Star is 160 km.

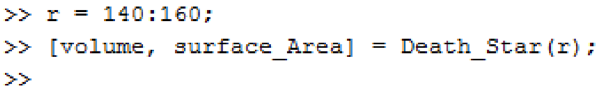


```
function [volume, surface_area] = Death_Star(radius)
%This function will calculate the volume and surface area
%of the death star when the radius is supplied

volume = (4/3)*pi*radius.^(3);

surface_area = 4*pi*radius.^(2);

end
```



```
>> [volume, surface_area] = Death_Star(160)

volume =

   1.7157e+07


surface_area =

   3.2170e+05
```

Because the actual radius of the Death Star is not known, a range of values will be used from 140 km to 160 km with a spacing of one. Thus, using the new range of values will calculate the volume and surface area of each radius.



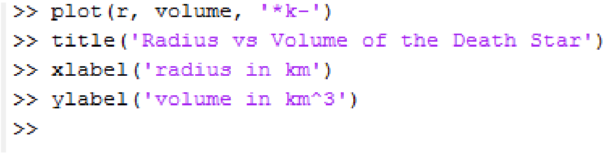
```
>> r = 140:160;
>> [volume, surface_Area] = Death_Star(r);
>>
```

Plotting

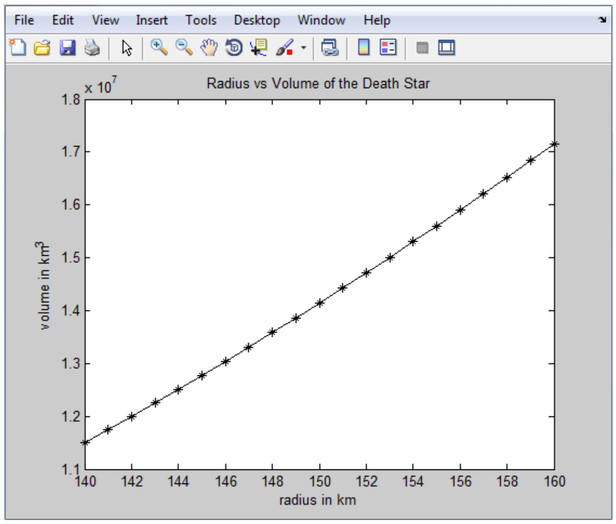
Plotting in MATLAB enables the user to easily display any data that is collected. The focus of this section will be 2D plotting using the basic xy-plane. Use the plot(x,y) function. In order to compare multiple sets of data, the data can be plotted on the same graph or plotted on a separate graph in the same figure. When putting multiple lines on the same axis use plot(x1, y1, x2, y2). The data points can be distinguished from each other by changing the color or point shape. When creating another graph in the same figure, use the subplot(m, n, p) function. In this case, “m” breaks the figure into “m” number of rows, “n” breaks the figure into “n” number of columns, “p” indicates which section the graph will be placed.

Example:

Plot the radius versus the volume of the Death Star using the data found in the previous section. Add a title and axis labels.

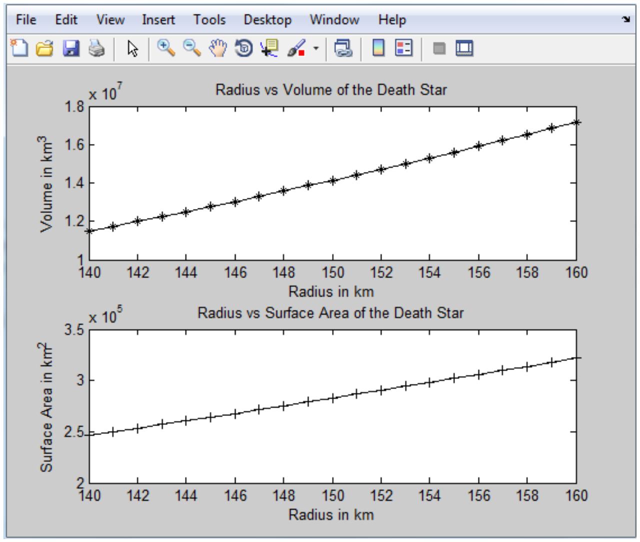


```
>> plot(r, volume, '*k-')
>> title('Radius vs Volume of the Death Star')
>> xlabel('radius in km')
>> ylabel('volume in km^3')
>>
```



Use the subplot function to plot the radius versus volume on the top and radius versus surface area on the bottom. Add titles.

```
r =140:160;
[volume, surface_area] = Death_Star(r);

subplot(2, 1, 1)
plot(r, volume, '*k-')
title('Radius vs Volume of the Death Star')
xlabel('Radius in km')
ylabel('Volume in km^3')
subplot(2, 1, 2)
plot(r, surface_area, '+k-')
title('Radius vs Surface Area of the Death Star')
xlabel('Radius in km')
ylabel('Surface Area in km^2')
```



Loops

Loops are used to run large amounts of numbers through the same process. There are two types of loops used in MATLAB: “for loops” and “while loops.” “For loops” are best used when there is a known number of data points. “While loops” are best used when there is certain criteria that must be met to perform the commands in the loop; it is possible to convert between the two types of loops. The structure of a “for loop:”

for *variable* = *startvalue* : *endvalue*

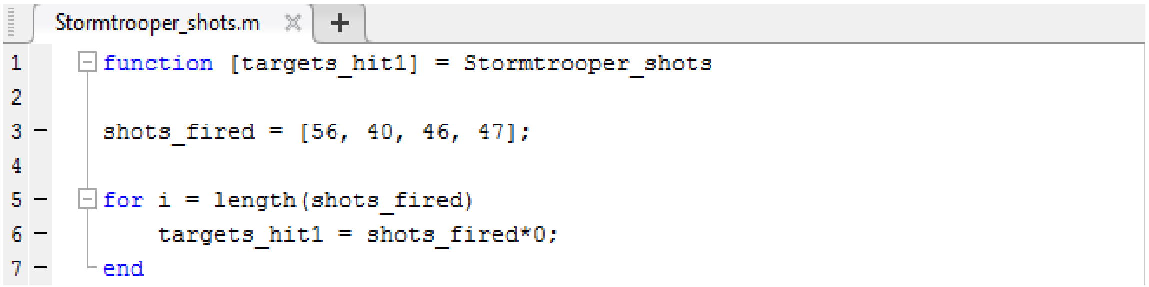
Commands performed by loop

end

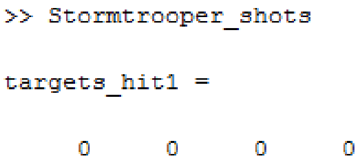
Using length(*variable*) is an alternative to stating the start value and end value.

Example:

Four stormtroopers use lasers to shoot at a target. Use a “for loop” to determine how many of those lasers hit the target. The stormtroopers fire [56, 40, 46, 47] shots over a one minute period. The formula for determining the shots that hit the target is: shots_fired*0.



```
function [targets_hit1] = Stormtrooper_shots

shots_fired = [56, 40, 46, 47];

for i = length(shots_fired)
    targets_hit1 = shots_fired*0;
end
```



```
>> Stormtrooper_shots

targets_hit1 =

     0     0     0     0
```

The structure of a “while loop:”

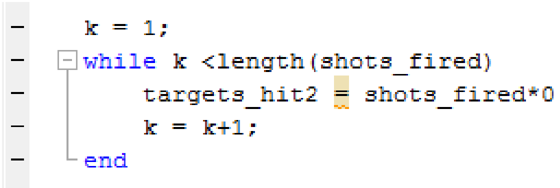
while *criteria of the loop* (as a logical statement)

command to be executed

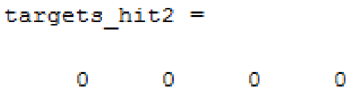
end

Example:

Use a “while loop” to perform the same function as the previous “for loop.”



```
  k = 1;
while k <length(shots_fired)
    targets_hit2 = shots_fired*0
    k = k+1;
end
```



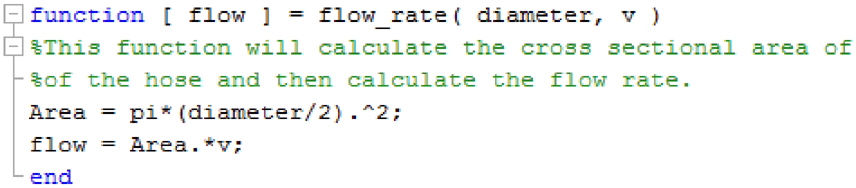
```
targets_hit2 =

     0     0     0     0
```

Practice Problems

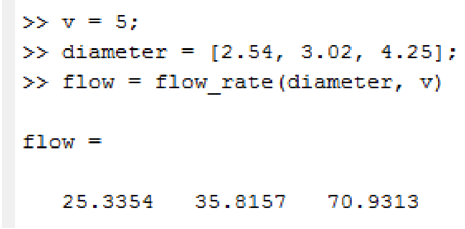
1. Write a function that is named *flow_rate* that will calculate the flow rate of water through three different size water hoses. The function should have diameter and velocity as inputs. The diameters are 2.54 cm, 3.02 cm, and 4.25 cm. The velocity of the water is 5 cm/s. The formula for flow rate is area*velocity. Plot the data with the diameter on the x-axis and the flow rate on the y-axis. Add titles.
2. Write a “for loop” that will calculate the final balance in a bank account that compounds yearly at 8% for 30 years. The initial balance is $1000. Write a “while loop” that performs the same action.

Answers

1.

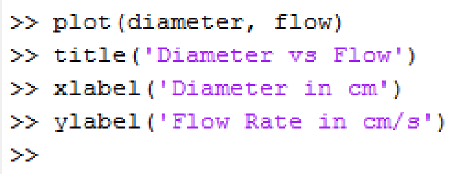


```
function [ flow ] = flow_rate( diameter, v )
%This function will calculate the cross sectional area of
%of the hose and then calculate the flow rate.
Area = pi*(diameter/2).^2;
flow = Area.*v;
end
```

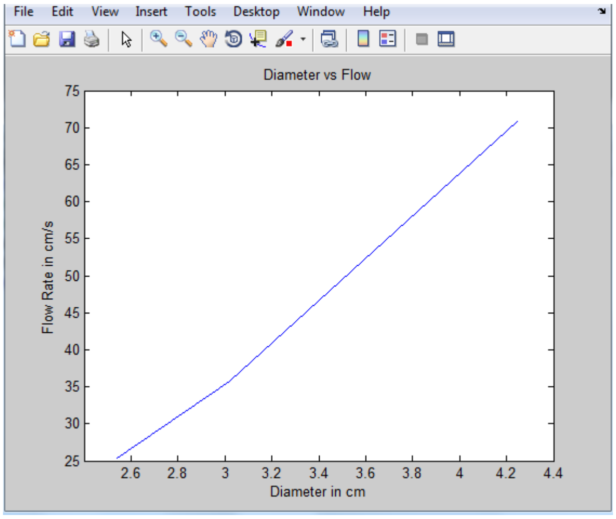


```
>> v = 5;
>> diameter = [2.54, 3.02, 4.25];
>> flow = flow_rate(diameter, v)

flow =

   25.3354   35.8157   70.9313
```

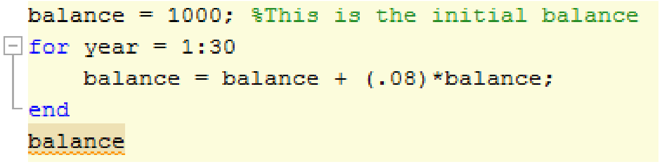
(practice problem #1 continued on next page)



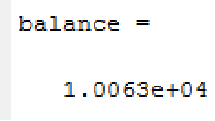
```
>> plot(diameter, flow)
>> title('Diameter vs Flow')
>> xlabel('Diameter in cm')
>> ylabel('Flow Rate in cm/s')
>>
```



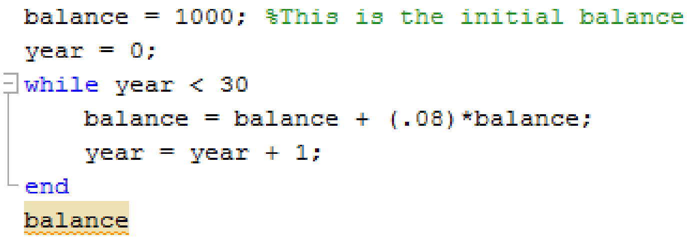
2.



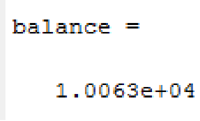
```
balance = 1000; %This is the initial balance
for year = 1:30
    balance = balance + (.08)*balance;
end
balance
```



```
balance =

   1.0063e+04
```



```
balance = 1000; %This is the initial balance
year = 0;
while year < 30
    balance = balance + (.08)*balance;
    year = year + 1;
end
balance
```



```
balance =

   1.0063e+04
```