Date: 18-09-2021

Experiment 5

Aim: To work with MATLAB user defined functions.

Apparatus: MATLAB Software

Objective: To learn how to develop and debug sub-tasks independently before building the final program.

Problems:

Q-1. It is often useful to be able to simulate the throw of a fair die. Write a MATLAB function dice that simulates the throw of a fair die by returning some random integer between 1 and 6 every time that it is called. (*Hint:* Call random0 to generate a random number. Divide the possible values out of random0 into six equal intervals and return the number of the interval that a given random value falls into.)

Code:

```
clear all;
close all;

Num = dice();
fprintf("Random Number Generated: %d ",Num);
```

Dice Function:

```
function number = dice()
    number = randi(6);
end
```

Output:

```
Command Window
Random Number Generated: 5
>> DiceCall
Random Number Generated: 6
>> DiceCall
Random Number Generated: 1
>> DiceCall
Random Number Generated: 6
>> DiceCall
Random Number Generated: 4
>>
```

Q-2. A function is said to be recursive if the function calls itself. MATLAB functions are designed to allow recursive operation. To test this feature, write a MATLAB function to evaluate the factorial function, which is defined as follows:

$$N! = \begin{cases} N(N-1)! & N \geq 1 \\ 1 & N = 0 \end{cases}$$

where N is a positive integer. The function should check to make sure that there is a single argument N, and that N is a nonnegative integer. If it is not, generate an error using the error function. If the input argument is a nonnegative integer, the function should evaluate $N!$ using Equation.

Code:

```
clc;
clear all;
close all;

x = input("Enter your Number: ");
y = factorial(x);
if y == -1
    fprintf("Invalid Entry")
else
    fprintf("Factorial of %d is %d ",x,y);
end
```

Factorial Function:

```
function fact = factorial(n)
    if nargin == 1
        if n == 0
           fact = 1;
        elseif n < 0
            fact = -1 ;
        else
            fact = n * factorial(n-1);
        end
    else
        error('Number of Argument is not equal to 1');
    end
end
```

Output:

Q-3. Antiship missiles (ASMs) are sometimes used to attack naval ships, and the ships being attacked use surface-to-air missiles (SAMs) to try to kill the attacking missiles before they hit the ship. Answer the following questions about this situation using the function developed in previous exercise.
(*a*) If the ship uses three SAMs to attack an incoming ASM and the probability of success of each attack is 0.3, what is the probability that the ship will destroy the ASM before it is hit?
(*b*) If the ship uses three SAMs to attack an incoming ASM and the probability of success of each attack is 0.5, what is the probability that the ship will destroy the ASM before it is hit?
(*c*) If the ship uses three SAMs to attack an incoming ASM and the probability of success of each attack is 0.7, what is the probability that the ship will destroy the ASM before it is hit?
(*d*) If the ship uses three SAMs to attack an incoming ASM and the probability of success of each attack is 0.9, what is the probability that the ship will destroy the ASM before it is hit?

Code:

```
clc;
clear all;
close all;

success = 0.3:0.2:0.9;
fail = 1- success;
n=3;
```

```
r = 0;
probab =  nCr(n,r).* power(success,r) .* power(fail,n-r);
hit = 1 - probab;
fprintf("Sr. \tProbablity(Destroy)")
fprintf("\n1. \t%f",hit(1));
fprintf("\n2. \t%f",hit(2));
fprintf("\n3. \t%f",hit(3));
fprintf("\n4. \t%f",hit(4));

function C = nCr(n,r)
    C = factorial(n) /(factorial(n-r) * factorial(r));
end

function fact = factorial(n)
    if n == 0
       fact = 1;
    else
        fact = n * factorial(n-1);
    end
end
```

Output:

Command Window

```
Sr.     Probablity(Destroy)
1.      0.657000
2.      0.875000
3.      0.973000
4.      0.999000

>>
```

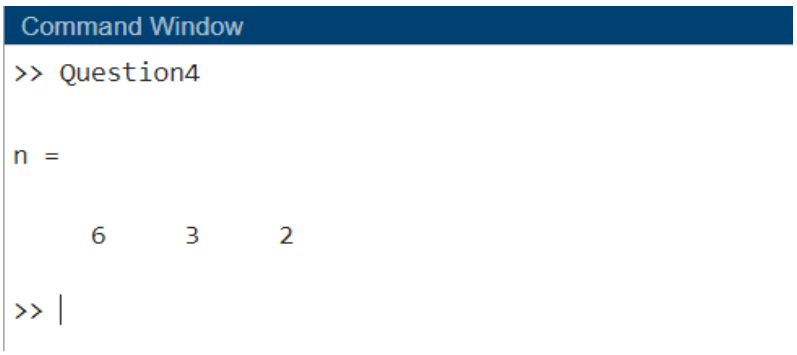
Q-4. Suppose a designer wanted to ensure that there is a 90% probability that the ship will survive an ASM attack without being hit. How many SAMs should the ship fire to destroy the incoming ASM if the individual probability of success of for a SAM is (*a*) 30%? (*b*) 50%? (*c*) 70%?

Code:

```
function Question4
     ans1 = 0.9;
     P0 =1-ans1;
     p=0.3:0.2:0.7;
```

```
    q=1-p;
    n = round(log(P0)./log(q))
end
```

Output:



Q-5. Write three MATLAB functions to calculate the hyperbolic sine, cosine, and tangent functions:

$$\sinh(x) = \frac{e^x - e^{-x}}{2} \qquad \cosh(x) = \frac{e^x + e^{-x}}{2} \qquad \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

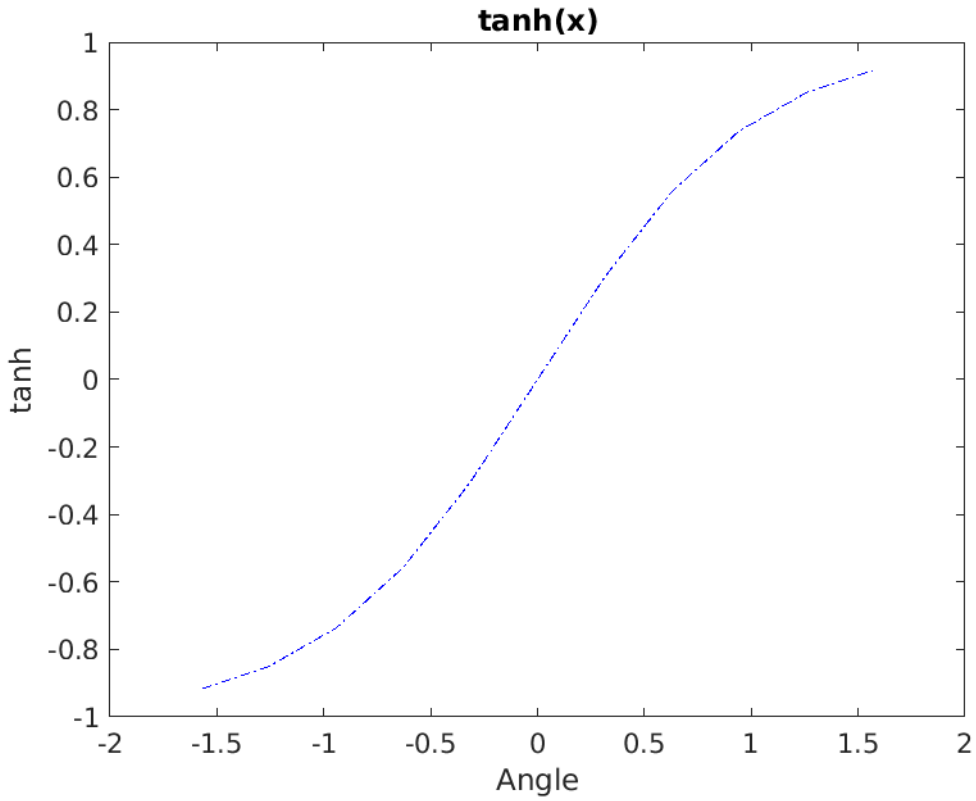
Use your functions to plot the shapes of the hyperbolic sine, cosine, and
tangent functions.

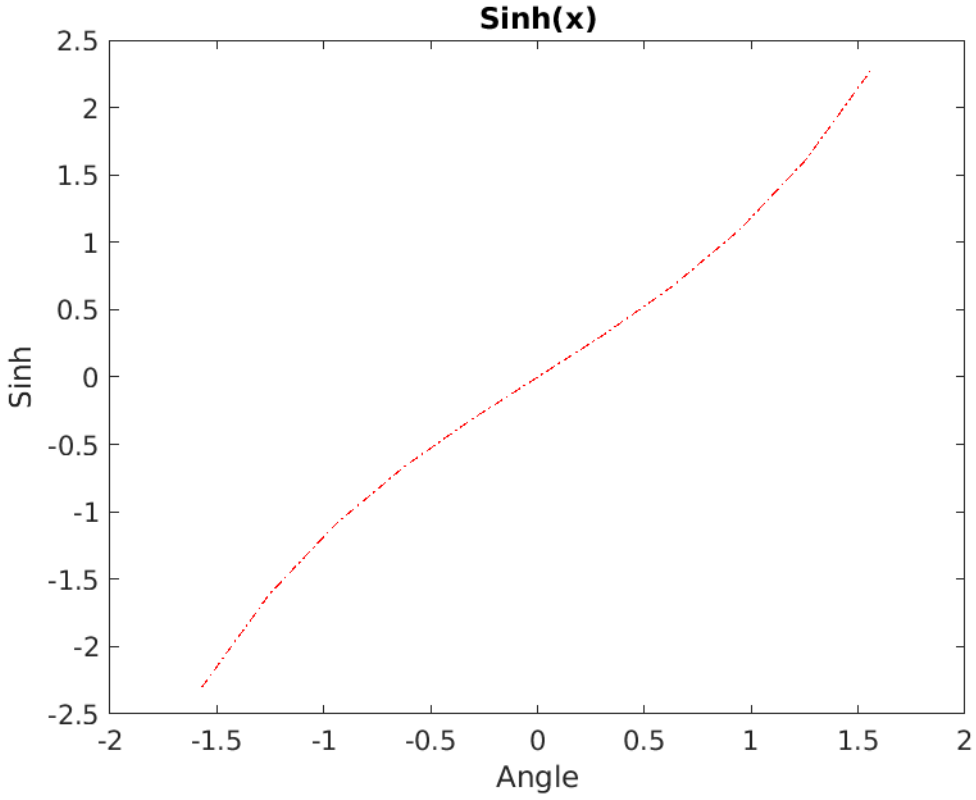
Code:

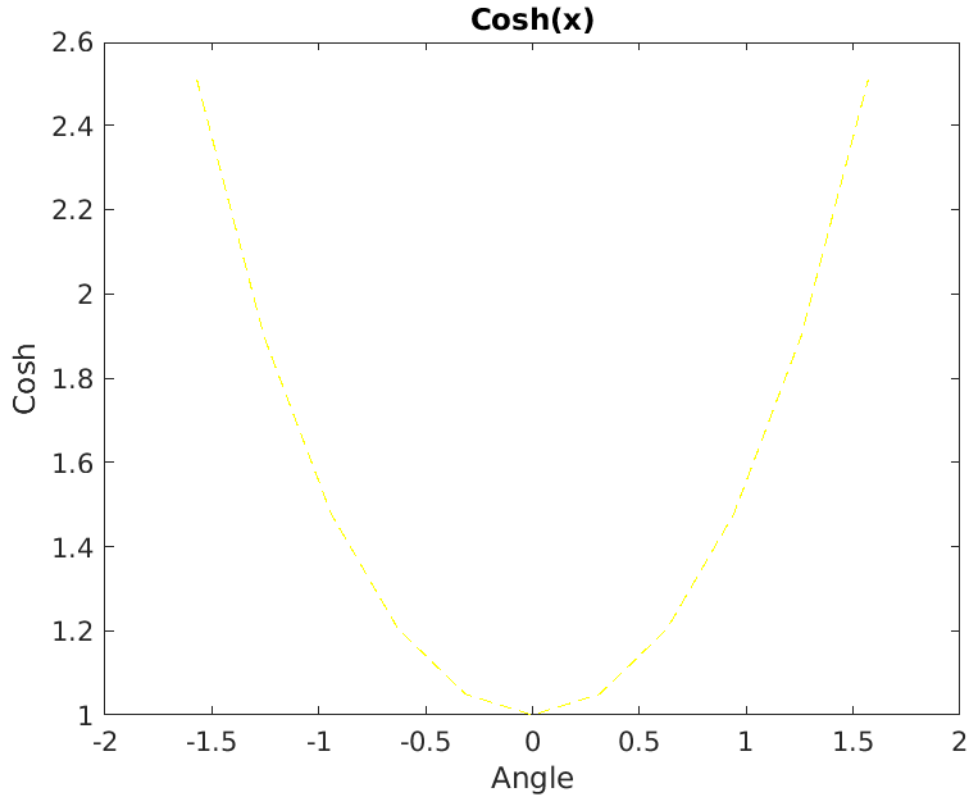
```
clc;
clear all;
close all;

x = -pi/2:pi/10:pi/2;

while true
    fprintf("\n1. Sinh(x)");
    fprintf("\n2. Cosh(x)");
    fprintf("\n3. tanh(x)");
    fprintf("\n4. Exit");
    c = input("\nEnter Your Choice");

    switch c
        case 1
            Sinh(x)
        case 2
            Cosh(x)
        case 3
            Tanh(x)
```

```
        case 4
            break;
        otherwise
            fprintf("\nInvalid Choice");
    end
end

function Sinh(x)
    ans1 = (exp(x) - exp(-x))/2;
    figure
    plot(x,ans1,"LineStyle",'-.',"Color",'red')
    xlabel('Angle')
    ylabel('Sinh')
    title('Sinh(x)')
end

function Cosh(x)
    ans1 = (exp(x) + exp(-x))/2;
    figure
    plot(x,ans1,"LineStyle",'--',"Color",'yellow')
    xlabel('Angle')
    ylabel('Cosh')
    title('Cosh(x)')
end

function Tanh(x)
    ans1 = (exp(x) - exp(-x))./(exp(x) + exp(-x));
    figure
    plot(x,ans1,"LineStyle",'-.',"Color",'b')
    xlabel('Angle')
    ylabel('tanh')
    title('tanh(x)')
end
```

Output:


tanh(x)
tanh
Angle
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2


Sinh(x)
Sinh
Angle
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2




```
Command Window
  1. Sinh(x)
  2. Cosh(x)
  3. tanh(x)
  4. Exit
  Enter Your Choice1

  1. Sinh(x)
  2. Cosh(x)
  3. tanh(x)
  4. Exit
  Enter Your Choice2

  1. Sinh(x)
  2. Cosh(x)
  3. tanh(x)
  4. Exit
  Enter Your Choice3

  1. Sinh(x)
  2. Cosh(x)
  3. tanh(x)
  4. Exit
fx Enter Your Choice
```

Q-6. Write a single MATLAB function hyperbolic to calculate the hyperbolic sine, cosine, and tangent functions as defined in the previous problem. The function should have two arguments. The first argument will be a string containing the function names 'sinh', 'cosh', or 'tanh', and the second argument will be the value of x at which to evaluate the function. The file should also contain three subfunctions sinh1, cosh1, and tanh1 to perform the actual calculations, and the primary function should call the proper subfunction depending on the value in the string. [*Note:* Be sure to handle the case of an incorrect number of arguments, and also the case of an invalid string. In either case, the function should generate an error.]

Code:

```
function Question61
     x=input('Enter the value of x that is point: ');
     fprintf("\nSinh: %f ",hyperbolic('sinh',x));
     fprintf("\nCosh: %f ",hyperbolic('cosh',x));
     fprintf("\nTanh: %f ",hyperbolic('tanh',x));
end
function n = hyperbolic(typ, x,vararg)
     if nargin~=2
        error('Invalid Number of arguments');
        return;
     end

     if typ=='sinh'
        n = sinh1(x);
     elseif typ=='cosh'
        n = cosh1(x);
     elseif typ=='tanh'
        n = tanh1(x);
     else
        error('Invalid hyperbolic type');
     end
end

function s = sinh1(x)
    s=(exp(x)-exp(-1*x))./2;
end

function c = cosh1(x)
    c=(exp(x)+exp(-1*x))./2;
end
```

```
function t = tanh1(x)
    t=sinh(x)./cosh(x);
end
```

Output:

```
Command Window
>> Question61
Enter the value of x that is point:
2

Sinh: 3.626860
Cosh: 3.762196
Tanh: 0.964028
>>
```

Q-7. Write a function that attempts to locate the maximum and minimum values of an arbitrary function over a certain range. The function being evaluated should be passed to the function as a calling argument. The function should have the following input arguments:

first_value—The first value of *x* to search
last_value—The last value of *x* to search
num_steps—The number of steps to include in the search
func—The name of the function to search

The function should have the following output arguments:

xmin—The value of *x* at which the minimum was found
min_value—The minimum value of found
xmax—The value of *x* at which the maximum was found
max_value—The maximum value *f* s*x*d found

Be sure to check that there are a valid number of input arguments, and that the MATLAB help and lookfor commands are properly supported.

Code:

```
function Question7
     first_value=input('Enter the First Value: ');
```

```
    last_value=input('Enter the Last Value: ');
    num_steps=input('Enter the Number of Steps: ');
    fun=input('Enter the Name of function(cosh,sinh,tanh): ','s');

    [xmin, min_value, xmax, max_value] = hyperbolic(first_value,
last_value, num_steps, fun)
end

function [xmin, min_value, xmax, max_value] =
hyperbolic(first_value, last_value, num_steps, typ,vararg)
    if nargin~=4
        error('Invalid number of arguments');
        return;
    end

    x=first_value:num_steps:last_value;

    if typ=='sinh'
        s=sinh(x);
        [min_value, xmin]=min(s);
        [max_value, xmax]=max(s);
    elseif typ=='cosh'
        c=cosh(x);
        [min_value, xmin] = min(c);
        [max_value, xmax] = max(c);
    elseif typ=='tanh'
        t=tanh(x);
        [min_value, xmin] = min(t);
        [max_value, xmax] = max(t);
    else
       error('Invalid hyperbolic type');
    end
end

function s = sinh(x)
 s=(exp(x)-exp(-x))./2;
end

function c = cosh(x)
 c=(exp(x)+exp(-x))./2;
end

function t = tanh(x)
 t=sinh(x)./cosh(x);
end
```

Output:

```
Command Window
>> Question7
Enter the First Value:
0
Enter the Last Value:
1
Enter the Number of Steps:
0.01
Enter the Name of function(cosh,sinh,tanh):
cosh

xmin =

     1


min_value =

     1


xmax =

   101


max_value =
```

Command Window

```
max_value =

    1.5431

>> Question7
Enter the First Value:
0.2
Enter the Last Value:
1.8
Enter the Number of Steps:
0.1
Enter the Name of function(cosh,sinh,tanh):
sinh

xmin =

     1


min_value =

    0.2013


xmax =
```

Command Window

```
xmax =

    17


max_value =

    2.9422

>> Question7
Enter the First Value:
0
Enter the Last Value:
1
Enter the Number of Steps:
0.1
Enter the Name of function(cosh,sinh,tanh):
tanh

xmin =

     1


min_value =
```

```
xmin =

     1


min_value =

     0


xmax =

    11


max_value =

    0.7616

>>
```

Q-8. The gravitational force F between two bodies of masses and is given by the equation

$$F = \frac{Gm_1m_2}{r^2}$$

where G is the gravitation constant (6.672×10^{-11} N m^2 / kg^2), m_1 and m_2 are the masses of the bodies in kilograms, and r is the distance between the two bodies. Write a function to calculate the gravitational force between two bodies given their masses and the distance between them. Test you function by determining the force on an 800 kg satellite in orbit 38,000 km above the Earth. (The mass of the Earth is 5.98×10^{24} kg.)

Code:

```
function Question8
     m1=input('\nEnter the Mass of Body 1: ');
     m2=input('\n\nEnter the Mass of Body 2: ');
     r=input('\n\nEnter the distance between Mass 1 and Mass 2: ');
     x=F(m1,m2,r);
     fprintf("\n\nForce: %f",x);
end

function F = F(m1,m2,r)
     G=6.672e-11;
```

```
    F=G*m1*m2/r^2;
end
```

Output:

```
Command Window
>> Question8

Enter the Mass of Body 1:
800


Enter the Mass of Body 2:
5.98e24


Enter the distance between Mass 1 and Mass 2:
3.8e7


Force: 221.044654
>>
```

Conclusion:

From this experiment we came to understand the concepts of MATLAB's inbuilt functions that is variable arguments and plotting sinusoidal curves and mainly functions. We learnt the function calling methods and function defining methods. Using functions, we can easily reuse the piece of code or statements whenever needed.