1. Calculate the running averages and standard deviations of $n$ entered data.

2. a) Write three Matlab functions to calculate the hyperbolic sine, cosine and tangent functions:

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

b) Write a single function `hyperbolic` to calculate the previous functions. The function should have two input arguments. One argument will be a string containing the function names ‘\(\sinh\)’, ‘\(\cosh\)’ and ‘\(\tanh\)’ 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.

3. Write a function `F2C.m` that accepts temperature in degrees F and computes the corresponding value in degrees C. The relation between the two is

$$T_{\text{C}} = \frac{5}{9} (T_{\text{F}} - 32)$$

Modify the function to perform the inverse operation as well, i.e., convert Celsius temperature into Fahrenheit temperature in a new function `C2F.m`. Write also a main function, `tempconv.m`, which calculates the resultant temperature.

4. Write a MATLAB function, `taylorexp.m` to calculate \(e^x\) using Taylor approximation. Write a sub-function `factorial` to evaluate required factorial values in the denominator of the terms in the expansion.

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \ldots$$

5. Write a MATLAB function, `taylorcos.m` to calculate \(\cos x\) using Taylor approximation. Write a sub-function `factorial` to evaluate required factorial values in the denominator of the terms in the expansion.

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \ldots$$

6. Write a function `bindec` to convert a binary number to a decimal number. Hint: assume the binary number as a string array.

7. **Tension on a cable.** A 200-kg object is to be hung from the end of a rigid 8 m horizontal pole of negligible weight. The pole is attached to a wall by a pivot and is supported by an 8 meters cable that is attached to the wall at a higher point. The tension on this cable is given by the equation
\[ T = \frac{W \cdot L_c \cdot L_p}{d \sqrt{L_c^2 - d^2}} \]

where \( T \) is tension on the cable, \( W \) is the weight of the object, \( L_c \) is the length of the cable, \( L_p \) is the length of the pole and \( d \) is the distance along the pole at which the cable is attached.

Write a MATLAB function to determine the distance \( d \) at which to attach the cable to the pole in order to minimize the tension on the cable. The function should calculate the tension on the cable at 0.1-m intervals from \( d=1 \) m to \( d=7 \) m and write the results in an array. Then call another function to locate the position that produces the minimum tension.

8. **Infinite Series.** Trigonometric functions are usually calculated on computers by using a truncated infinite series. An infinite series is an infinite set of terms that together add up to the value of a particular function or expression. For example, one infinite series used to evaluate the sine of a number is

\[
\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{9!} + \ldots
\]

Write a function that reads in a value for \( x \) in degrees and then calculates the sine of the \( x \) using sine intrinsic function. Next calculate the sine of \( x \) using equation with \( N=1,2,3,\ldots10 \). Compare the true value of \( \sin x \) with the values calculated using the truncated infinite series. How many terms are required to calculate \( \sin x \) to the full accuracy of your computer?

9. The gravitational force between two bodies of masses \( m_1 \) and \( m_2 \) is given by the equation

\[
F = \frac{G \cdot m_1 \cdot m_2}{r^2}
\]

where \( G \) is the gravitational constant \((6.672 \times 10^{-11} \text{ Nm}^2/\text{kg}^2)\) and \( r \) is the distance between two bodies. Write a function to calculate the gravitational force between two bodies given their masses and the distance between them.

Test your function by determining the force on a 800 kg satellite in orbit 38000 km above the Earth. (The mass of Earth is \( 5.98 \times 10^{24} \text{ kg} \))

10. Create an anonymous function called `my_function`, equal to

\[- x^2 - 5x - 3 + e^x\]

use the `fplot` function to create a plot from -5 to 5. Recall that the `fplot` function can accept a function handle as input.

Use the `fminbnd` function to find the minimum function value in this range.

11. Create four anonymous functions to represent the function \( 6e^{3\cos x^2} \), which is composed of the functions \( h(z) = 6e^z \), \( g(y) = 3\cos y \), \( f(x) = x^2 \). Use the anonymous functions to plot \( 6e^{3\cos x^2} \) over the range 0 ≤ \( x \) ≤ 4.
function [ave std]=runstats(x)
%runstats generate running ave /std deviation

persistent n
persistent sum_x
persistent sum_x2

if x=='reset';
    n=0;
    sum_x=0;
    sum_x2=0;
else
    n=n+1;
    sum_x = sum_x+x;
    sum_x2=sum_x2+x^2;
end

if n == 0;
    ave=0;
    std=0;
elseif n == 1;
    ave=sum_x;
    std=0;
else
    ave=sum_x/n;
    std=sqrt((n*sum_x2-sum_x^2)/(n*(n-1)));
end

%script file test_stats.m

[ave std]=runstats('reset');
nvals=input('enter number of values');

for i=1:nvals
    x=input('enter a value');
    [ave,std]=runstats(x);
    disp(['ave=',num2str(ave),', std_dev=',num2str(std)]);
end
2. 

```matlab
function res=hyperbolic(type,x)
    if type=='sinh';
        res=sinh1(x);
    elseif type == 'cosh';
        res=cosh1(x);
    elseif type == 'tanh';
        res=tanh1(x);
    else
        disp('wrong input for function name')
    end

function result=sinh1(x)
result=(exp(x)-exp(-x))/2;

function result=cosh1(x)
result=(exp(x)+exp(-x))/2;

function result=tanh1(x)
result=(exp(x)-exp(-x))/(exp(x)+exp(-x));
```

3. 

```matlab
function r=tempconv(T)
    x=input('enter conversion type: 1 for C2F 2 for F2C')
    if x==1
        r=C2F(T);
    elseif x==2
        r=F2C(T);
    end

function TF=C2F(TC)
    TF=1.8*TC+32;

function TC=F2C(TF)
    TC=5/9*(TF-32);
```

6. 

```matlab
function base10 = binary(base2)
    base10=0;
    n=length(base2);
    power=(n-1);
    for i=1:n
        base10= base10+str2num(base2(i))*2^power;
        power=power-1;
    end
```
function r=tension(w,lp,lc)
  d=1:0.1:7;
  r=w*lp*lc./d./sqrt(lc^2-d.^2);

  min=r(1);
  for i=2:length(r)
    if a(i)<min
      min=r(i);
      ind=i;
    end
  end
  dmin=d(ind)
  res=angle(dmin,lc)
  res=acos(dmin/lc)*180/pi;