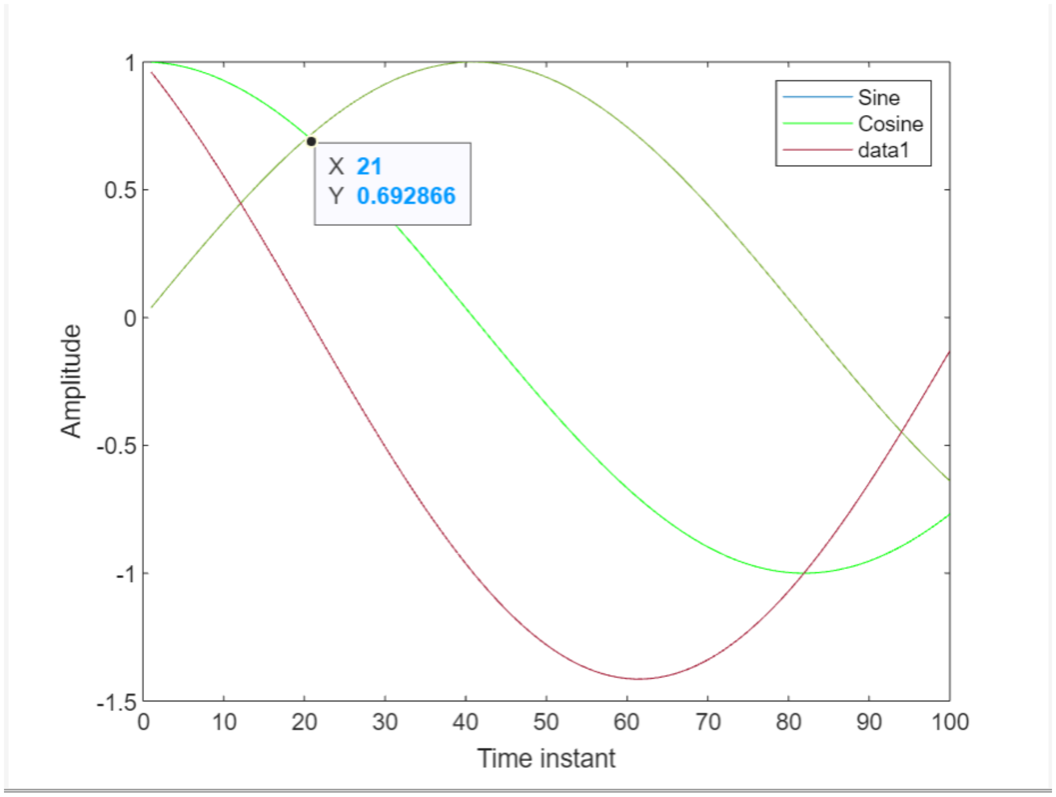
Question #1

```
%generate values from 100 to 1
t2=100:-1:1
%generate values from 1 to 100 with the interval of 2
t3=1:2:100
```

t2 will generate integers from 100 to 1 with the step of -1. (100, 99, 98, …)

t3 will generate integers from 1 to 100 with the intervals of 2. (1, 3, 5, …)

Question #2



```
% Finding the intersection point between y and y2
intersection_point = min(abs(y - y2))
>> 0.0260
```

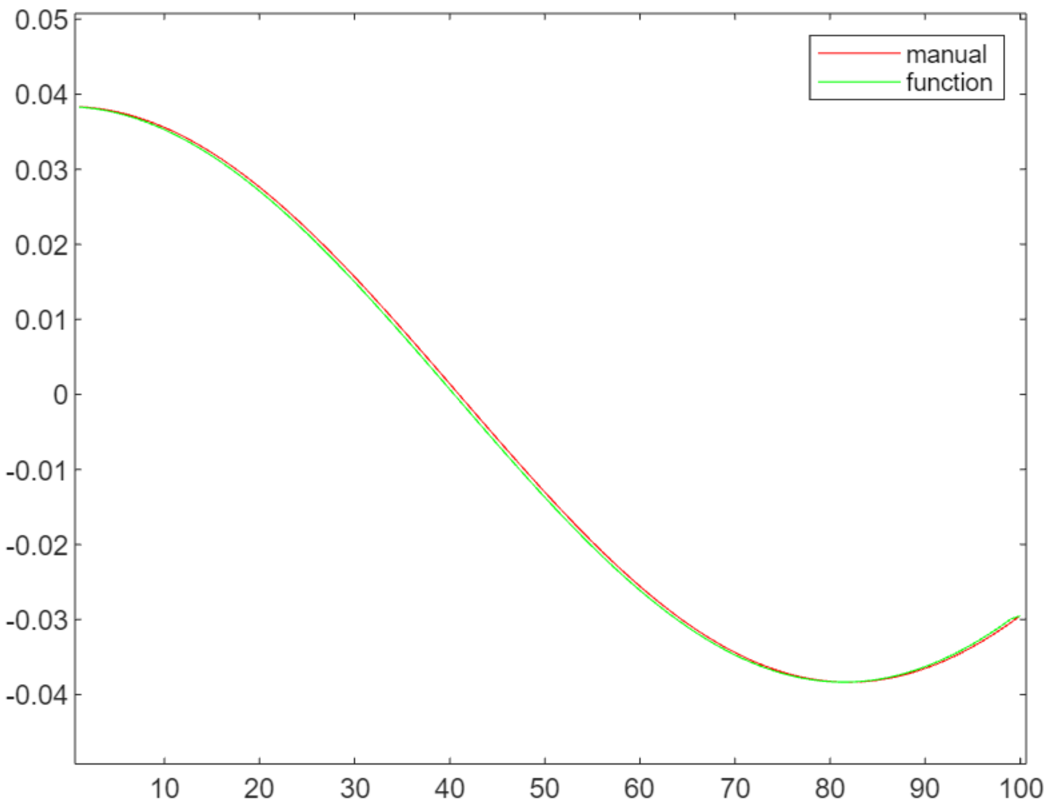
The reason we cannot find the exact point of intersection is that the values of `t1` are discrete, and the exact point of intersection might not fall exactly on any of the given discrete values of `t1`. Therefore, we can only find an approximation of the intersection point using this method.

Question #3

```
% do the differentiate manually and by function
manual_diff = 2*pi*(f/F)*cos(2*pi*t1*(f/F))
function_diff = diff(y)
function_diff = [diff(y), -0.0295] % to make the function_diff lenght 100
% plot and see their differences
plot(t1,manual_diff, 'r')
hold on
plot(t1, function_diff, 'g')
legend('manual','function');
```

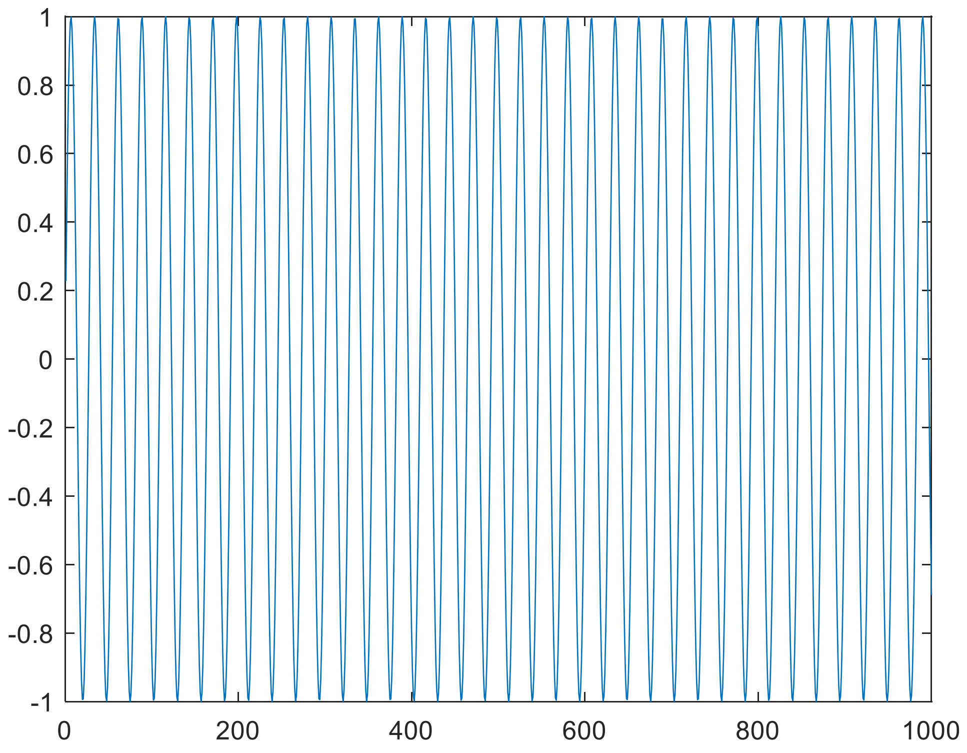
```
manual_diff =

  Columns 1 through 16

    0.0383    0.0382    0.0381    0.0379    0.0376    0.0373    0.0370    0.0366
0.0361    0.0356    0.0350    0.0344    0.0337    0.0330    0.0322    0.0314   …


function_diff =

  Columns 1 through 16

    0.0383    0.0382    0.0380    0.0378    0.0375    0.0372    0.0368    0.0363
0.0358    0.0353    0.0347    0.0340    0.0333    0.0326    0.0318    0.0309   …
```

As it is clear from the columns 1-16 of the both manually-derived difference and the one which is derived by the function diff(y), the two results are more or less the same, just in some cases there might be 0.01% ~ 0.02% difference. (Other columns are the same; they are not copied here in order to avoid redundancy.) It can be seen also from the following plot. The graphs are coincident.



Question #4

```
% plot row 5 of matrix
row5 = Matrix(5, : );
plot(row5)
soundsc(row5)
```



```
% the mean over the 2nd dimension, which means the rows.
m_rows = mean(Matrix, 2)

m_rows =

    0.0056
    0.0100
    0.0658
    0.0003
    0.0071

% the mean over the 1st dimension, which means the columns.
m_columns = mean(Matrix, 1)

m_columns =

  Columns 1 through 6

    0.2120    0.3438    0.3587    0.2832    0.1907    0.1596 …

% the mean over all data.
m_all = mean(Matrix,"all" )

m_all =

    0.0178
```

There are 1000 columns so the m_columns is an array with the length of 1000 and I did not copy the whole output here.

Question #5

```
% Find the minimum mean value among rows and its index
[min_mean, min_index] = min(m_rows)

% Identify the corresponding signal
lowest_sounding_signal = Matrix(min_index, :)

min_mean =

   3.3276e-04


min_index =

     4
```

The signal in the 4th row is the lowest sounding signal.

Question #6

inco13par.txt is a matrix of 529x16 double cases and variable.

```
% Check for NaN values in the inco13par
nanValues = isnan(inco13par)

% Count the number of NaN values for each variable
missingCounts = sum(nanValues)

missingCounts =


 Columns 1 through 10
```

```
     0     0     1   144   108   335   111   183   127   116

  Columns 11 through 16

   191     2     0     0     3     7
```

isnan() generate a matrix of 529x16 logical.
Then we can see for each variable how many Nans there are. (missingCounts is 1x16 double)

For example, in the first column there is not any Nan. But in the 4th column there are 144 missing values.