Task 1

I used the example from Tasksheet 02, namely

$$f''(x) \approx \frac{f(x+h) - 2f(x) + f(x+h)}{h^2}, \quad f(x) = cos(x)$$

to print out the following table,

h-value	Exact	Approximation	Difference
1.0000000000000000	0.4161468365471424	0.382603	0.0335433541851632
0.5000000000000000	0.4161468365471424	0.407549	0.0085977996869263
0.1000000000000000	0.4161468365471424	0.415800	0.0003466734547534
0.0100000000000000	0.4161468365471424	0.416143	0.0000034678760133
0.0010000000000000	0.4161468365471424	0.416147	0.0000000346400955
0.0001000000000000	0.4161468365471424	0.416147	0.0000000195410624
0.0000100000000000	0.4161468365471424	0.416147	0.0000002802191542
0.0000010000000000	0.4161468365471424	0.416001	0.0001462692200963
0.0000001000000000	0.4161468365471424	0.438538	0.0223912581797945
0.0000000100000000	0.4161468365471424	1.110223	0.6940761880780140
0.0000000010000000	0.4161468365471424	55.511151	55.0950043947106778
0.0000000001000000	0.4161468365471424	5551.115123	5550.6989762892344515
0.0000000000100000	0.4161468365471424	555111.512313	555111.0961657416773960
0.0000000000010000	0.4161468365471424	0.000000	0.4161468365471424
0.0000000000001000	0.4161468365471424	5551115123.125782	5551115122.7096347808837891
0.0000000000000100	0.4161468365471424	-1665334536937.734863	1665334536938.1511230468750000
0.0000000000000010	0.4161468365471424	277555756156289.125000	277555756156288.7187500000000000

Figure 1. Output Table.

And the following code is used:

```python
import numpy as np
import math

np.set_printoptions(precision=24)

h = np.zeros(18)
h[0] = 1
h[1] = 0.5

for i in range(2, 18):
    h[i] = math.pow(10, -(i-1))

#print(h)

A = np.zeros([17, 4])
# initialize the Matrix A

for i in range(0, 17):
    x = 2
    A[i][0] = h[i]
    A[i][1] = -np.cos(2)
    A[i][2] = (np.cos(x + h[i]) - 2 * (np.cos(x)) + np.cos(x - h[i])) / (math.pow(h[i], 2))
    A[i][3] = np.abs(A[i][2] - A[i][1])

# print(A)

print("%-28s\t%-28s\t%-28s\t%-28s" % ('h-value', 'Exact', 'Approximation', 'Difference'))

for i in range(0, 17):
    print("%-28.6f\t%-28.6f\t%-28.6f\t%-28.6f" % (A[i][0], A[i][1], A[i][2], A[i][3]))
```

To verify if the central difference approximation is actually second order accurate. We can focus on the values associated with $h < 10^{-1}$.
The error decreases from 0.00034, which suggests that the approximation is actually second order accurate.

Task 2

The central difference approximation is actually second order accurate due to the slope generated from the plot.
From the log-log plot, the approximation begins to fail around the 7-th to 9-th iteration. This is likely due to the limitation of python with decreasing h-value.

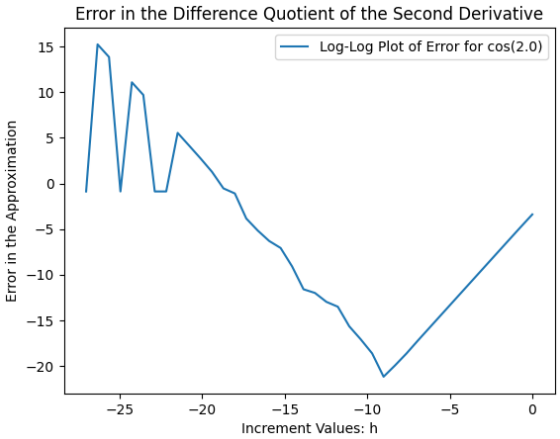

Figure 2. Plot Generated Using Provided Code.

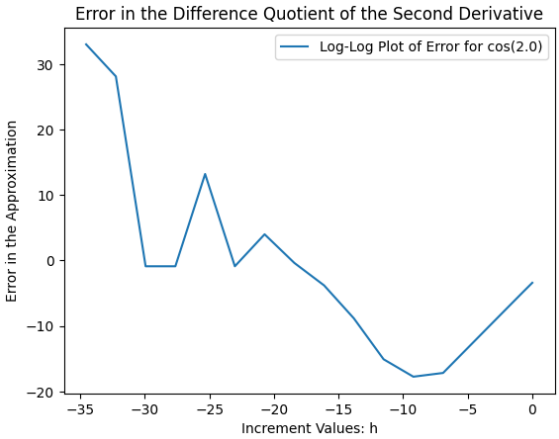

Figure 3. Plot Generated Using Own Code.

Since the I only changed two values in the provided code, my own code for **Figure 3** is the following:

```
import matplotlib.pyplot as plt
import numpy as np
import math

np.set_printoptions(precision=24)

h = np.zeros(18)
h[0] = 1
h[1] = 0.5

for i in range(2, 18):
    h[i] = math.pow(10, -(i-1))

#print(h)

A = np.zeros([17, 4])
# initialize the Matrix A

for i in range(0, 17):
    x = 2
    A[i][0] = h[i]
    A[i][1] = -np.cos(2)
    A[i][2] = (np.cos(x + h[i]) - 2 * (np.cos(x)) + np.cos(x - h[i])) / (math.pow(h[i], 2))
    A[i][3] = np.abs(A[i][2] - A[i][1])

x = []
y = []
for i in range(0, 17):
    x.append(np.log(A[i][0]))
    y.append(np.log(A[i][3]))

fig1 = plt.gcf()
plt.title('Error in the Difference Quotient of the Second Derivative')
plt.xlabel('Increment Values: h')
plt.ylabel('Error in the Approximation')
plt.plot(x, y, label='Log-Log Plot of Error for cos(2.0)')
plt.legend()
plt.show()
fig1.savefig('MyWay.png', bbox_inches='tight')
```

Task 3

The routine for single precision is provided below:

```
import numpy as py

def sMachineEps():
    one = py.float32(1.0)
    eps = py.float32(1.0)
    for i in range(1, 1000):
        diff = py.float32(one - (one + eps))
        print('{0:<8s}{1:.16f}{2:<12s}{3:.16f}{4:<6s}{5:10f}'.format('Diff =', diff, ' | Eps
     = ', eps, ' | Counter: ', i))
        if diff == py.float32(0):
            return diff
        eps = py.float32(0.5*eps)
```

With the following code for testing:

```
import numpy as py
from sMachineEps import sMachineEps

sMachineEps()
```

And the following output:

```
Diff =  -1.0000000000000000 | Eps =    1.0000000000000000 | Counter:   1.000000
Diff =  -0.5000000000000000 | Eps =    0.5000000000000000 | Counter:   2.000000
Diff =  -0.2500000000000000 | Eps =    0.2500000000000000 | Counter:   3.000000
Diff =  -0.1250000000000000 | Eps =    0.1250000000000000 | Counter:   4.000000
Diff =  -0.0625000000000000 | Eps =    0.0625000000000000 | Counter:   5.000000
Diff =  -0.0312500000000000 | Eps =    0.0312500000000000 | Counter:   6.000000
Diff =  -0.0156250000000000 | Eps =    0.0156250000000000 | Counter:   7.000000
Diff =  -0.0078125000000000 | Eps =    0.0078125000000000 | Counter:   8.000000
Diff =  -0.0039062500000000 | Eps =    0.0039062500000000 | Counter:   9.000000
Diff =  -0.0019531250000000 | Eps =    0.0019531250000000 | Counter:  10.000000
Diff =  -0.0009765625000000 | Eps =    0.0009765625000000 | Counter:  11.000000
Diff =  -0.0004882812500000 | Eps =    0.0004882812500000 | Counter:  12.000000
Diff =  -0.0002441406250000 | Eps =    0.0002441406250000 | Counter:  13.000000
Diff =  -0.0001220703125000 | Eps =    0.0001220703125000 | Counter:  14.000000
Diff =  -0.0000610351562500 | Eps =    0.0000610351562500 | Counter:  15.000000
Diff =  -0.0000305175781250 | Eps =    0.0000305175781250 | Counter:  16.000000
Diff =  -0.0000152587890625 | Eps =    0.0000152587890625 | Counter:  17.000000
Diff =  -0.0000076293945312 | Eps =    0.0000076293945312 | Counter:  18.000000
Diff =  -0.0000038146972656 | Eps =    0.0000038146972656 | Counter:  19.000000
Diff =  -0.0000019073486328 | Eps =    0.0000019073486328 | Counter:  20.000000
Diff =  -0.0000009536743164 | Eps =    0.0000009536743164 | Counter:  21.000000
Diff =  -0.0000004768371582 | Eps =    0.0000004768371582 | Counter:  22.000000
Diff =  -0.0000002384185791 | Eps =    0.0000002384185791 | Counter:  23.000000
Diff =  -0.0000001192092896 | Eps =    0.0000001192092896 | Counter:  24.000000
Diff =  0.0000000000000000 | Eps =    0.0000000596046448 | Counter:  25.000000
```

Figure 4. Machine Epsilon Single Precision Output:

The routine for double precision is provided below:

```
def dMachineEps():
    one = 1.0
    eps = 1.0
    for i in range(1, 1000):
        diff = one - (one + eps)
        print('{0:<8s}{1:.16f}{2:<12s}{3:.16f}{4:<6s}{5:10f}'.format('Diff =', diff, ' | Eps
     = ', eps, ' | Counter: ', i))
        if diff == 0:
            return diff
        eps = 0.5*eps
```

With the following code for testing:

```
import numpy as py
from dMachineEps import dMachineEps

dMachineEps()
```

And the following output:

```
Diff =  -1.0000000000000000 | Eps =    1.0000000000000000 | Counter:   1.000000
Diff =  -0.5000000000000000 | Eps =    0.5000000000000000 | Counter:   2.000000
Diff =  -0.2500000000000000 | Eps =    0.2500000000000000 | Counter:   3.000000
Diff =  -0.1250000000000000 | Eps =    0.1250000000000000 | Counter:   4.000000
Diff =  -0.0625000000000000 | Eps =    0.0625000000000000 | Counter:   5.000000
Diff =  -0.0312500000000000 | Eps =    0.0312500000000000 | Counter:   6.000000
Diff =  -0.0156250000000000 | Eps =    0.0156250000000000 | Counter:   7.000000
Diff =  -0.0078125000000000 | Eps =    0.0078125000000000 | Counter:   8.000000
Diff =  -0.0039062500000000 | Eps =    0.0039062500000000 | Counter:   9.000000
Diff =  -0.0019531250000000 | Eps =    0.0019531250000000 | Counter:  10.000000
Diff =  -0.0009765625000000 | Eps =    0.0009765625000000 | Counter:  11.000000
Diff =  -0.0004882812500000 | Eps =    0.0004882812500000 | Counter:  12.000000
Diff =  -0.0002441406250000 | Eps =    0.0002441406250000 | Counter:  13.000000
Diff =  -0.0001220703125000 | Eps =    0.0001220703125000 | Counter:  14.000000
Diff =  -0.0000610351562500 | Eps =    0.0000610351562500 | Counter:  15.000000
Diff =  -0.0000305175781250 | Eps =    0.0000305175781250 | Counter:  16.000000
Diff =  -0.0000152587890625 | Eps =    0.0000152587890625 | Counter:  17.000000
Diff =  -0.0000076293945312 | Eps =    0.0000076293945312 | Counter:  18.000000
Diff =  -0.0000038146972656 | Eps =    0.0000038146972656 | Counter:  19.000000
Diff =  -0.0000019073486328 | Eps =    0.0000019073486328 | Counter:  20.000000
Diff =  -0.0000009536743164 | Eps =    0.0000009536743164 | Counter:  21.000000
Diff =  -0.0000004768371582 | Eps =    0.0000004768371582 | Counter:  22.000000
Diff =  -0.0000002384185791 | Eps =    0.0000002384185791 | Counter:  23.000000
Diff =  -0.0000001192092896 | Eps =    0.0000001192092896 | Counter:  24.000000
Diff =  -0.0000000596046448 | Eps =    0.0000000596046448 | Counter:  25.000000
Diff =  -0.0000000298023224 | Eps =    0.0000000298023224 | Counter:  26.000000
Diff =  -0.0000000149011612 | Eps =    0.0000000149011612 | Counter:  27.000000
Diff =  -0.0000000074505806 | Eps =    0.0000000074505806 | Counter:  28.000000
Diff =  -0.0000000037252903 | Eps =    0.0000000037252903 | Counter:  29.000000
Diff =  -0.0000000018626451 | Eps =    0.0000000018626451 | Counter:  30.000000
Diff =  -0.0000000009313226 | Eps =    0.0000000009313226 | Counter:  31.000000
Diff =  -0.0000000004656613 | Eps =    0.0000000004656613 | Counter:  32.000000
Diff =  -0.0000000002328306 | Eps =    0.0000000002328306 | Counter:  33.000000
Diff =  -0.0000000001164153 | Eps =    0.0000000001164153 | Counter:  34.000000
Diff =  -0.0000000000582077 | Eps =    0.0000000000582077 | Counter:  35.000000
Diff =  -0.0000000000291038 | Eps =    0.0000000000291038 | Counter:  36.000000
Diff =  -0.0000000000145519 | Eps =    0.0000000000145519 | Counter:  37.000000
Diff =  -0.0000000000072760 | Eps =    0.0000000000072760 | Counter:  38.000000
Diff =  -0.0000000000036380 | Eps =    0.0000000000036380 | Counter:  39.000000
Diff =  -0.0000000000018190 | Eps =    0.0000000000018190 | Counter:  40.000000
Diff =  -0.0000000000009095 | Eps =    0.0000000000009095 | Counter:  41.000000
Diff =  -0.0000000000004547 | Eps =    0.0000000000004547 | Counter:  42.000000
Diff =  -0.0000000000002274 | Eps =    0.0000000000002274 | Counter:  43.000000
Diff =  -0.0000000000001137 | Eps =    0.0000000000001137 | Counter:  44.000000
Diff =  -0.0000000000000568 | Eps =    0.0000000000000568 | Counter:  45.000000
Diff =  -0.0000000000000284 | Eps =    0.0000000000000284 | Counter:  46.000000
Diff =  -0.0000000000000142 | Eps =    0.0000000000000142 | Counter:  47.000000
Diff =  -0.0000000000000071 | Eps =    0.0000000000000071 | Counter:  48.000000
Diff =  -0.0000000000000036 | Eps =    0.0000000000000036 | Counter:  49.000000
Diff =  -0.0000000000000018 | Eps =    0.0000000000000018 | Counter:  50.000000
Diff =  -0.0000000000000009 | Eps =    0.0000000000000009 | Counter:  51.000000
Diff =  -0.0000000000000004 | Eps =    0.0000000000000004 | Counter:  52.000000
Diff =  -0.0000000000000002 | Eps =    0.0000000000000002 | Counter:  53.000000
Diff =  0.0000000000000000 | Eps =    0.0000000000000001 | Counter:  54.000000
```

Figure 5. Machine Epsilon Double Precision Output.

Task 4

I created the Software Manual's Table of Contents. Then I uploaded my precision calculating file. Afterward, I added the links to those files to the Software Manual's Table of Contents.

Task 5

I have created and compiled and linked my shared library. It can be found here

Task 6

There are difference between static and shared libraries[1].

1. Shared libraries reduce the amount of code that is duplicated in each program that makes use of the library, keeping the binaries small.
2. Static libraries increase the overall size of the binary, but it means that you don't need to carry along a copy of the library that is being used.

There are also different kinds of shared libraries[2]. For examples:

1. Global Shared Libraries;
2. Folder-level Shared Libraries;
3. Automatic Shared Libraries

[1] https://stackoverflow.com/questions/2649334/difference-between-static-and-shared-libraries
[2] https://www.jenkins.io/doc/book/pipeline/shared-libraries/