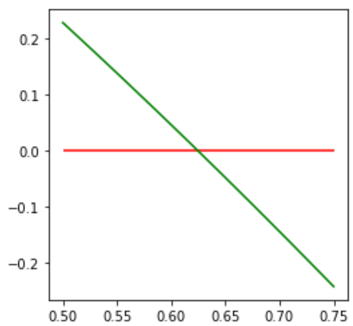
TASK 1

```
import numpy as np
from matplotlib import pyplot as plt

plt.rcParams["figure.figsize"] = [4, 4]

def f(x):
    return (np.cos(x)-(1.3*x))

x = np.linspace(0.5,0.75,1000)
plt.plot(x,f(x), color='green')
plt.hlines(y=0,xmin=0.5,xmax=0.75,color='red')
plt.show()
```

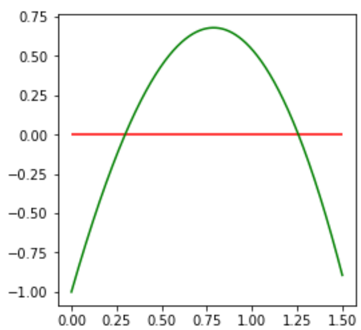


```
import numpy as np
from matplotlib import pyplot as plt

plt.rcParams["figure.figsize"] = [4, 4]

def f(x):
    return ((x*np.cos(x))-(2*x**2)+(3*x)-1)

x = np.linspace(0,1.5,1000)
plt.plot(x,f(x), color='green')
plt.hlines(y=0,xmin=0,xmax=1.5,color='red')
plt.show()
```

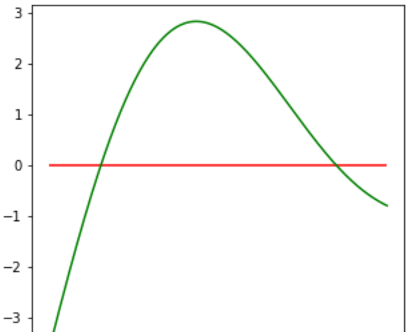


```
import numpy as np
from matplotlib import pyplot as plt

plt.rcParams["figure.figsize"] = [4, 4]

def f(x):
    return ((2*x*np.cos(2*x))-((x+1)**2))

x = np.linspace(-2.5,-0.5,1000)
plt.plot(x,f(x), color='green')
plt.hlines(y=0,xmin=-2.5,xmax=-0.5,color='red')
plt.show()
```



TASK 2

```
import numpy as np
from tabulate import tabulate

def func (x):
    return (np.cos(x)-(1.3*x))

def bisection(func, x1, x2, tol=0.0001, max_iter=100):
    if func(x1) * func(x2) >= 0:
        return  "Error: Choose different interval, function should have different signs at the interval endpoints."
    data=[]
    iter = 0
    xr = x2
    error = tol + 1

    while iter < max_iter and error > tol:
        xrold = xr
        xr = ((x1+x2)/2)
        iter += 1
        error = abs((xr - xrold) )
        test = func(x1) * func(xr)
        if (test>0):
          x1 = xr
        else:
          x2 = xr
        data.append([iter+1,x1,func(x1),x2,func(x2),xr,func(xr),error])
    print(tabulate(data,headers=['#','x1','f(x1)','x2','f(x2)','xr','f(xr)',"error"],tablefmt="github"))
    print('\nRoot of given function is x=%.9f in n=%d number of iterations with a tolerence=%.5f' %(xr,iter,tol))

    return

bisection(func,-1,1,0.001,100)
```

```
|   # |       x1 |       f(x1) |    x2 |       f(x2) |       xr |        f(xr) |       error |
|-----|----------|-------------|-------|-------------|----------|--------------|-------------|
|   2 | 0        | 1           | 1     | -0.759698   | 0        |  1           | 1           |
|   3 | 0.5      | 0.227583    | 1     | -0.759698   | 0.5      |  0.227583    | 0.5         |
|   4 | 0.5      | 0.227583    | 0.75  | -0.243311   | 0.75     | -0.243311    | 0.25        |
|   5 | 0.5      | 0.227583    | 0.625 | -0.00153688 | 0.625    | -0.00153688  | 0.125       |
|   6 | 0.5625   | 0.114674    | 0.625 | -0.00153688 | 0.5625   |  0.114674    | 0.0625      |
|   7 | 0.59375  | 0.0569735   | 0.625 | -0.00153688 | 0.59375  |  0.0569735   | 0.03125     |
|   8 | 0.609375 | 0.0278184   | 0.625 | -0.00153688 | 0.609375 |  0.0278184   | 0.015625    |
|   9 | 0.617188 | 0.0131656   | 0.625 | -0.00153688 | 0.617188 |  0.0131656   | 0.0078125   |
|  10 | 0.621094 | 0.00582059  | 0.625 | -0.00153688 | 0.621094 |  0.00582059  | 0.00390625  |
|  11 | 0.623047 | 0.0021434   | 0.625 | -0.00153688 | 0.623047 |  0.0021434   | 0.00195312  |
|  12 | 0.624023 | 0.000303648 | 0.625 | -0.00153688 | 0.624023 |  0.000303648 | 0.000976562 |

Root of given function is x=0.624023438 in n=11 number of iterations with a tolerence=0.00100
```

```
import numpy as np
from tabulate import tabulate

def func (x):
    return ((x*np.cos(x))-(2*x**2)+(3*x)-1)

def bisection(func, x1, x2, tol=0.0001, max_iter=100):
    if func(x1) * func(x2) >= 0:
        return  "Error: Choose different interval, function should have different signs at the interval endpoints."
    data=[]
```

```
    iter = 0
    xr = x2
    error = tol + 1

    while iter < max_iter and error > tol:
        xrold = xr
        xr = ((x1+x2)/2)
        iter += 1
        error = abs((xr - xrold) )
        test = func(x1) * func(xr)
        if (test>0):
          x1 = xr
        else:
          x2 = xr
        data.append([iter+1,x1,func(x1),x2,func(x2),xr,func(xr),error])
    print(tabulate(data,headers=['#','x1','f(x1)','x2','f(x2)','xr','f(xr)',"error"],tablefmt="github"))
    print('\nRoot of given function is x=%.9f in n=%d number of iterations with a tolerence=%.5f' %(xr,iter,tol))

    return

bisection(func,1,2,0.001,100)
```

```
|   # |      x1 |      f(x1) |      x2 |       f(x2) |      xr |       f(xr) |       error |
|-----|---------|------------|---------|-------------|---------|-------------|-------------|
|   2 | 1       | 0.540302   | 1.5     | -0.893894   | 1.5     | -0.893894   | 0.5         |
|   3 | 1.25    | 0.019153   | 1.5     | -0.893894   | 1.25    |  0.019153   | 0.25        |
|   4 | 1.25    | 0.019153   | 1.375   | -0.388747   | 1.375   | -0.388747   | 0.125       |
|   5 | 1.25    | 0.019153   | 1.3125  | -0.172556   | 1.3125  | -0.172556   | 0.0625      |
|   6 | 1.25    | 0.019153   | 1.28125 | -0.0736339  | 1.28125 | -0.0736339  | 0.03125     |
|   7 | 1.25    | 0.019153   | 1.26562 | -0.0264729  | 1.26562 | -0.0264729  | 0.015625    |
|   8 | 1.25    | 0.019153   | 1.25781 | -0.00346802 | 1.25781 | -0.00346802 | 0.0078125   |
|   9 | 1.25391 | 0.00789046 | 1.25781 | -0.00346802 | 1.25391 |  0.00789046 | 0.00390625  |
|  10 | 1.25586 | 0.00222322 | 1.25781 | -0.00346802 | 1.25586 |  0.00222322 | 0.00195312  |
|  11 | 1.25586 | 0.00222322 | 1.25684 | -0.0006194  | 1.25684 | -0.0006194  | 0.000976562 |

 Root of given function is x=1.256835938 in n=10 number of iterations with a tolerence=0.00100
```

```
import numpy as np
from tabulate import tabulate

def func (x):
    return ((2*x*np.cos(2*x))-((x+1)**2))

def bisection(func, x1, x2, tol=0.0001, max_iter=100):
    if func(x1) * func(x2) >= 0:
        return  "Error: Choose different interval, function should have different signs at the interval endpoints."
    data=[]
    iter = 0
    xr = x2
    error = tol + 1

    while iter < max_iter and error > tol:
        xrold = xr
        xr = ((x1+x2)/2)
        iter += 1
        error = abs((xr - xrold) )
        test = func(x1) * func(xr)
        if (test>0):
          x1 = xr
        else:
          x2 = xr
        data.append([iter+1,x1,func(x1),x2,func(x2),xr,func(xr),error])
    print(tabulate(data,headers=['#','x1','f(x1)','x2','f(x2)','xr','f(xr)',"error"],tablefmt="github"))
    print('\nRoot of given function is x=%.9f in n=%d number of iterations with a tolerence=%.5f' %(xr,iter,tol))

    return

bisection(func,-1,0,0.001,100)
```

```
|   # |        x1 |      f(x1) |        x2 |       f(x2) |        xr |       f(xr) |       error |
|-----|-----------|------------|-----------|-------------|-----------|-------------|-------------|
|   2 | -1        | 0.832294   | -0.5      | -0.790302   | -0.5      | -0.790302   | 0.5         |
|   3 | -1        | 0.832294   | -0.75     | -0.168606   | -0.75     | -0.168606   | 0.25        |
|   4 | -0.875    | 0.296306   | -0.75     | -0.168606   | -0.875    |  0.296306   | 0.125       |
|   5 | -0.8125   | 0.0528816  | -0.75     | -0.168606   | -0.8125   |  0.0528816  | 0.0625      |
|   6 | -0.8125   | 0.0528816  | -0.78125  | -0.0608144  | -0.78125  | -0.0608144  | 0.03125     |
|   7 | -0.8125   | 0.0528816  | -0.796875 | -0.00468056 | -0.796875 | -0.00468056 | 0.015625    |
|   8 | -0.804688 | 0.0239252  | -0.796875 | -0.00468056 | -0.804688 |  0.0239252  | 0.0078125   |
|   9 | -0.800781 | 0.00957807 | -0.796875 | -0.00468056 | -0.800781 |  0.00957807 | 0.00390625  |
```

```
|  10 | -0.798828 | 0.00243764 | -0.796875 | -0.00468056 | -0.798828 |  0.00243764 | 0.00195312  |
|  11 | -0.798828 | 0.00243764 | -0.797852 | -0.00112424 | -0.797852 | -0.00112424 | 0.000976562 |

Root of given function is x=-0.797851562 in n=10 number of iterations with a tolerence=0.00100
```

TASK 3

```
import numpy as np
from tabulate import tabulate

def func(x):
    return (np.cos(x)-(1.3*x))

def dfunc(x):
    return (-(np.sin(x))-1.3)

def newton_raphson(func, dfunc, x0, tol=1e-4, max_iter=1000):
    xr = x0
    data=[]
    iter = 0
    error = tol + 1
    for i in range(max_iter):
        iter+=1
        fx = func(xr)
        dx = dfunc(xr)
        if abs(dx) < tol:
           raise Exception("Derivative is close to zero!")
        xrold=xr
        xr = xr - fx/dx
        error=abs(xr-xrold)

        data.append([iter,xr,func(xr),error])
        if error < tol:
           print(tabulate(data,headers=['Iteration','xr','f(xr)',"error"],tablefmt="github"))
           print('\nRoot of given function is x=%.9f in n=%d number of iterations with a tolerence=%.5f' %(xr,iter,tol))
           return

    raise Exception("Max iterations reached")

newton_raphson(func,dfunc,10,0.00001,100)
```

```
|   Iteration |         xr |         f(xr) |        error |
|-------------|------------|---------------|--------------|
|           1 |  -8.30616  |  10.3611      | 18.3062      |
|           2 |  17.564    | -22.5518      | 25.8702      |
|           3 | -48.6855   |  63.282       | 66.2495      |
|           4 | -21.1711   |  26.8402      | 27.5144      |
|           5 |  26.0146   | -33.1832      | 47.1857      |
|           6 |   9.99877  | -13.8381      | 16.0158      |
|           7 |  -8.28122  |  10.3512      | 18.28        |
|           8 |  18.268    | -22.9128      | 26.5492      |
|           9 | -12.2552   |  16.8837      | 30.5232      |
|          10 |  -1.74341  |   2.09467     | 10.5118      |
|          11 |   4.9093   |  -6.18645     |  6.65271     |
|          12 | -14.4642   |  18.4822      | 19.3735      |
|          13 |  37.8914   | -48.2773      | 52.3556      |
|          14 |   5.51461  |  -6.45009     | 32.3768      |
|          15 |  -5.14862  |   7.11573     | 10.6632      |
|          16 |  -1.92351  |   2.15512     |  3.22511     |
|          17 |   4.0371   |  -5.87335     |  5.96061     |
|          18 |  -7.26925  |  10.002       | 11.3064      |
|          19 |  14.188    | -18.4951      | 21.4572      |
|          20 |   6.14209  |  -6.99466     |  8.04586     |
|          21 |   0.10897  |   0.852407    |  6.03312     |
|          22 |   0.714049 |  -0.172547    |  0.605079    |
|          23 |   0.625785 |  -0.00301683  |  0.0882639   |
|          24 |   0.624185 |  -1.0376e-06  |  0.00159982  |
|          25 |   0.624185 |  -1.23013e-13 |  5.50618e-07 |

Root of given function is x=0.624184578 in n=25 number of iterations with a tolerence=0.00001
```

```
import numpy as np
from tabulate import tabulate

def func(x):
    return ((x*np.cos(x))-(2*x**2)+(3*x)-1)
```

```
def dfunc(x):
    return (-(x*np.sin(x))+(np.cos(x))-(4*x)+3)

def newton_raphson(func, dfunc, x0, tol=1e-4, max_iter=1000):
    xr = x0
    data=[]
    iter = 0
    error = tol + 1
    for i in range(max_iter):
        iter+=1
        fx = func(xr)
        dx = dfunc(xr)
        if abs(dx) < tol:
           raise Exception("Derivative is close to zero!")
        xrold=xr
        xr = xr - fx/dx
        error=abs(xr-xrold)

        data.append([iter,xr,func(xr),error])
        if error < tol:
           print(tabulate(data,headers=['Iteration','xr','f(xr)',"error"],tablefmt="github"))
           print('\nRoot of given function is x=%.9f in n=%d number of iterations with a tolerence=%.5f' %(xr,iter,tol))
           return

    raise Exception("Max iterations reached")

newton_raphson(func,dfunc,5,0.00001,100)
```

```
|   Iteration |      xr |         f(xr) |      error |
|-------------|---------|---------------|------------|
|           1 | 2.09927 | -4.57454      | 2.90073    |
|           2 | 1.50627 | -0.92174      | 0.593004   |
|           3 | 1.29977 | -0.131514     | 0.206497   |
|           4 | 1.25846 | -0.00535864   | 0.0413089  |
|           5 | 1.25663 | -1.0562e-05   | 0.00183256 |
|           6 | 1.25662 | -4.13602e-11  | 3.6263e-06 |

Root of given function is x=1.256623323 in n=6 number of iterations with a tolerence=0.00001
```

```
import numpy as np
from tabulate import tabulate

def func(x):
    return ((2*x*np.cos(2*x))-((x+1)**2))

def dfunc(x):
    return (-(4*x*np.sin(2*x))+(2*np.cos(2*x))-(2*(x+1)))

def newton_raphson(func, dfunc, x0, tol=1e-4, max_iter=1000):
    xr = x0
    data=[]
    iter = 0
    error = tol + 1
    for i in range(max_iter):
        iter+=1
        fx = func(xr)
        dx = dfunc(xr)
        if abs(dx) < tol:
           raise Exception("Derivative is close to zero!")
        xrold=xr
        xr = xr - fx/dx
        error=abs(xr-xrold)

        data.append([iter,xr,func(xr),error])
        if error < tol:
           print(tabulate(data,headers=['Iteration','xr','f(xr)',"error"],tablefmt="github"))
           print('\nRoot of given function is x=%.9f in n=%d number of iterations with a tolerence=%.5f' %(xr,iter,tol))
           return

    raise Exception("Max iterations reached")

newton_raphson(func,dfunc,-3,0.00001,100)
```

```
|   Iteration |       xr |         f(xr) |       error |
|-------------|----------|---------------|-------------|
|           1 | -1.94741 |  1.94363      | 1.05259     |
|           2 | -2.28462 | -0.998355     | 0.337207    |
```

```
|          3 | -2.19649 | -0.0521828   | 0.0881248   |
|          4 | -2.19133 | -0.000202041 | 0.00516326  |
|          5 | -2.19131 | -3.09417e-09 | 2.01473e-05 |
|          6 | -2.19131 |  1.33227e-15 | 3.08556e-10 |

Root of given function is x=-2.191308012 in n=6 number of iterations with a tolerence=0.00001
```

TASK 4

```python
import numpy as np
import scipy.optimize as sp

def f1(x):
    return (np.cos(x)-(1.3*x))

def f2(x):
    return ((x*np.cos(x))-(2*x**2)+(3*x)-1)

def f3(x):
    return ((2*x*np.cos(2*x))-((x+1)**2))

r1 = sp.fsolve(f1,[0,10],(),None,0,0,0.00001)
r2 = sp.fsolve(f2,[0,10],(),None,0,0,0.00001)
r3 = sp.fsolve(f3,[-1,-3],(),None,0,0,0.00001)

print(r1)
print(r2)
print(r3)
```

```
[0.62418458 0.62418458]
[0.29752143 1.25662326]
[-0.79816068 -2.19130804]
```

TASK 5

```python
import numpy as np
from tabulate import tabulate

def func (x):
    return (np.cos(x)-(1.3*x))

def falseposition(func, x1, x2, tol=0.0001, max_iter=100):
    if func(x1) * func(x2) >= 0:
        return  "Error: Choose different interval, function should have different signs at the interval endpoints."
    data=[]
    iter = 0
    xr = x2
    error = tol + 1

    while iter < max_iter and error > tol:
        xrold = xr
        xr = (x1-((func(x1)*(x2-x1))/(func(x2)-func(x1))))
        iter += 1
        error = abs((xr - xrold) )
        test = func(x1) * func(xr)
        if (test>0):
          x1 = xr
        else:
          x2 = xr
        data.append([iter+1,x1,func(x1),x2,func(x2),xr,func(xr),error])
    print(tabulate(data,headers=['#','x1','f(x1)','x2','f(x2)','xr','f(xr)',"error"],tablefmt="github"))
    print('\nRoot of given function is x=%.9f in n=%d number of iterations with a tolerence=%.5f' %(xr,iter,tol))

    return

falseposition(func,-1,2,0.001,100)
```

```
|   # |       x1 |       f(x1) |   x2 |    f(x2) |       xr |       f(xr) |      error |
|-----|----------|-------------|------|----------|----------|-------------|------------|
|   2 | 0.13682  | 0.812789    |    2 | -3.01615 | 0.13682  | 0.812789    | 1.86318    |
|   3 | 0.532327 | 0.169603    |    2 | -3.01615 | 0.532327 | 0.169603    | 0.395507   |
|   4 | 0.610463 | 0.0257805   |    2 | -3.01615 | 0.610463 | 0.0257805   | 0.078136   |
|   5 | 0.62224  | 0.00366374  |    2 | -3.01615 | 0.62224  | 0.00366374  | 0.0117764  |
```

```
|    6 | 0.623911 | 0.000515321 |     2 | -3.01615 | 0.623911 | 0.000515321 | 0.00167155 |
|    7 | 0.624146 | 7.2376e-05  |     2 | -3.01615 | 0.624146 | 7.2376e-05  | 0.00023507 |

Root of given function is x=0.624146170 in n=6 number of iterations with a tolerence=0.00100
```

```
import numpy as np
from tabulate import tabulate

def func(x):
    return ((x*np.cos(x))-(2*x**2)+(3*x)-1)

def secant(func,x0, x1 , tol=1e-4, max_iter=1000):
    data=[]
    iter = 0
    error = tol + 1
    for i in range(max_iter):
        iter+=1
        x2 = x0 - ((func(x0)*(x1-x0))/(func(x1)-func(x0)))
        error=abs(x1-x0)
        data.append([iter,x0,x1,x2,func(x2),error])
        if error < tol:
           print(tabulate(data,headers=['Iteration','x0','x1','x2','f(x2)',"error"],tablefmt="github"))
           print('\nRoot of given function is x=%.9f in n=%d number of iterations with a tolerence=%.5f' %(x2,iter,tol))
           return

        x0=x1
        x1=x2
    raise Exception("Max iterations reached")

secant(func,0,10,0.00001,100)
```

```
|   Iteration |         x0 |         x1 |         x2 |        f(x2) |        error |
|-------------|------------|------------|------------|--------------|--------------|
|           1 |  0         | 10         | -0.0560567 | -1.23042     | 10           |
|           2 | 10         | -0.0560567 | -0.125507  | -1.53254     | 10.0561      |
|           3 | -0.0560567 | -0.125507  |  0.226788  | -0.201522    |  0.0694499   |
|           4 | -0.125507  |  0.226788  |  0.280126  | -0.0473553   |  0.352294    |
|           5 |  0.226788  |  0.280126  |  0.29651   | -0.00273439  |  0.0533388   |
|           6 |  0.280126  |  0.29651   |  0.297514  | -4.23675e-05 |  0.016384    |
|           7 |  0.29651   |  0.297514  |  0.29753   | -3.92378e-08 |  0.00100402  |
|           8 |  0.297514  |  0.29753   |  0.29753   | -5.64215e-13 |  1.58014e-05 |
|           9 |  0.29753   |  0.29753   |  0.29753   |  0           |  1.46478e-08 |

Root of given function is x=0.297530234 in n=9 number of iterations with a tolerence=0.00001
```

✓ 0s completed at 8:44 PM