Distilation column grapical method

Using maccabe theile method to find the no of stages for a distilation column

values required

```
feed composition      =xf
relative volatality   =rv
q value               =q
```

In [23]:

```
import numpy as np
from matplotlib import pyplot as plt
from matplotlib import axes
%matplotlib inline
```

In [24]:

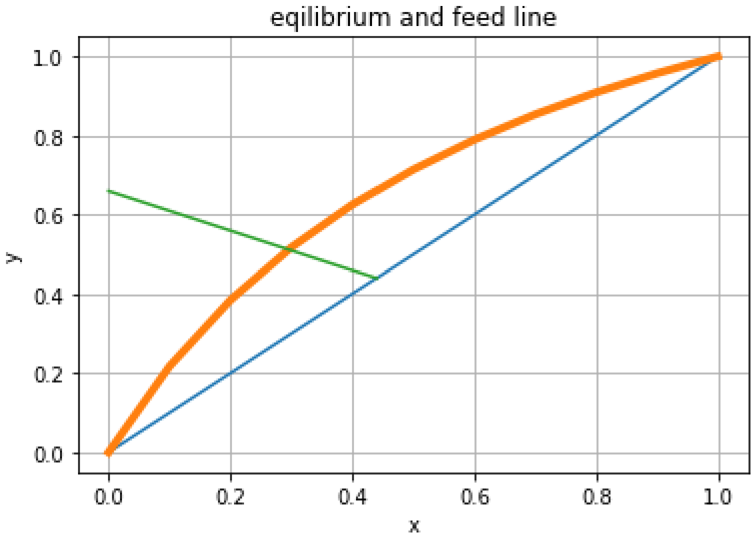
```
print("Enter feed composition , relative volatality and q value")
def nib(xf,rv,q):                           #xf=feed composition    rv=relative volatality
    global x
    global y
    global xe
    global ye
    global xfe
    global yfe
    global cr
    cr=xf
    xe=np.linspace(0,1,11)           # datas for plotting the equilibrium curve
    ye=(rv*xe)/(1+(rv-1)*xe)

    x=np.linspace(0,1,11)
    y=np.linspace(0,1,11)            #45 degree line

    plt.plot(x,y)                    #plotting the 45 degree line
    plt.plot(xe,ye,linewidth=4)      #plotting the equilibrium curve
    xfe=np.linspace(0,xf,11)
    yfe=((-q/(1-q))*(xfe-xf))+xf     #feed line equation
    plt.plot(xfe,yfe)
    plt.grid()
    plt.xlabel('x')
    plt.ylabel('y')
    plt.title('eqilibrium and feed line')
```

```
Enter feed composition , relative volatality and q value
```

```
In [25]: nib(.44,2.5,.333)
```

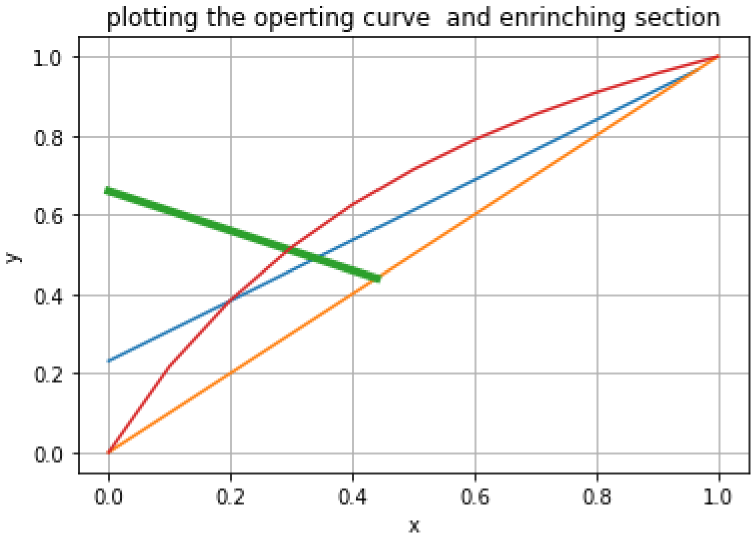


enter the datas by observing the graph

```
In [26]: def full(xd,x1,y1,multiple):
             global xo,yoi,yo
             xf=cr                                            # rm =r minimum
             rm=(xd-y1)/(y1-x1)                               # xd =top product concentratin
             R=multiple*rm
             x=np.linspace(0,1,11)                            #x1  =x prime ie,x ordinate at the point of intersection btw equil and feed line
             y=np.linspace(0,1,11)                            #y1  =y prime ie,y ordinate at the point of intersection btw equil and feed line
             yoi=xd/(R+1)
             xo=np.linspace(0,xd,11)
             yo=((xd-yoi)/xd)*xo+yoi                          #multipe =how many times r minimum
             plt.plot(xo,yo)
             plt.plot(x,y)
             plt.plot(xfe,yfe,linewidth=4)
             plt.plot(xe,ye)
             plt.grid()
             plt.xlabel('x')
             plt.ylabel('y')
             plt.title('plotting the operting curve  and enrinching section')
```

In [27]:

```
full(.966,.295,.51,1.5)
```

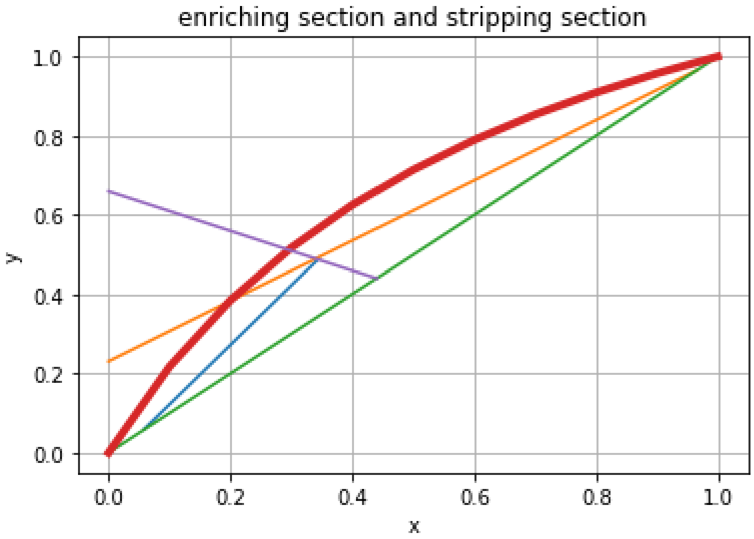


In [28]:

```
def final(xfi,yfi,xn,yn):          # xfi= coordinate at the intersection of
    xf=cr
    xs=np.linspace(xn,xfi,5)
    ys=((yfi-yn)/(xfi-xn))*xs-.028
    plt.plot(xs,ys)
    plt.plot(xo,yo)
    plt.plot(x,y)
    plt.plot(xe,ye,linewidth=4)
    plt.plot(xfe,yfe)
    plt.grid()
    plt.xlabel('x')
    plt.ylabel('y')
    plt.title('enriching section and stripping section')
```

In [29]:

```
final(.346,.49,.058,.058)
```



In [ ]: