Part 1: Object Oriented Programming

In [ ]:

```
[Q1] Replace  pass with the appropriate code in the Line class methods to ac
class Line(object):

def __init__(self,coor1,coor2):

pass

def distance(self):

pass

def slope(self):

pass

Example output:

coordinate1 = [3,2]

coordinate2 = [8,10]

li = Line(coordinate1,coordinate2)

li.distance() # 9.433981132056603

li.slope() # 1.6
```

In [61]:

```
import math
class Line(object):
    def __init__(self,coor1,coor2):
        self.coor1 = coor1
        self.coor2 = coor2
        pass
    def distance(self):
        dis = ((self.coor1[0]-self.coor2[0])**2 +
               (self.coor1[1] -self.coor2[1])**2)
        di = math.sqrt(abs(dis))
        return (di)
    def slope(self):
        return (((self.coor1[1] -self.coor2[1])/(self.coor1[0] -self.coor2[
```

In [62]:

```
coordinate1 = [3,2]
coordinate2 = [8,10]

li = Line(coordinate1,coordinate2)

print(li.distance())
print(li.slope())
```

```
9.433981132056603
1.6
```

In [ ]:

```
[Q2] Replace  pass with the appropriate code in the Cylinder class methods
class Cylinder(object):

def __init__(self,height=1,radius=1):

pass

def volume(self):

pass

def surface_area(self):

pass

Example output:

c = Cylinder(2,3)

c.volume() # 56.52

c.surface_area() # 94.2
```

In [65]:

```
class Cylinder(object):
    def __init__(self,height,radius):
        self.height= height
        self.radius = radius
        pass
    def volume(self):
        v = 3.14*(self.radius**2)*self.height
        return v
    def surface_area(self):
        a = (2*3.14*(self.radius*self.height))+(2*3.14*(self.radius**2))
        return a
```

In [ ]:

```
c = Cylinder(2,3)

print(c.volume()) # 56.52

print(c.surface_area()) # 94.2
```

In [66]:

```
56.52
94.2
```

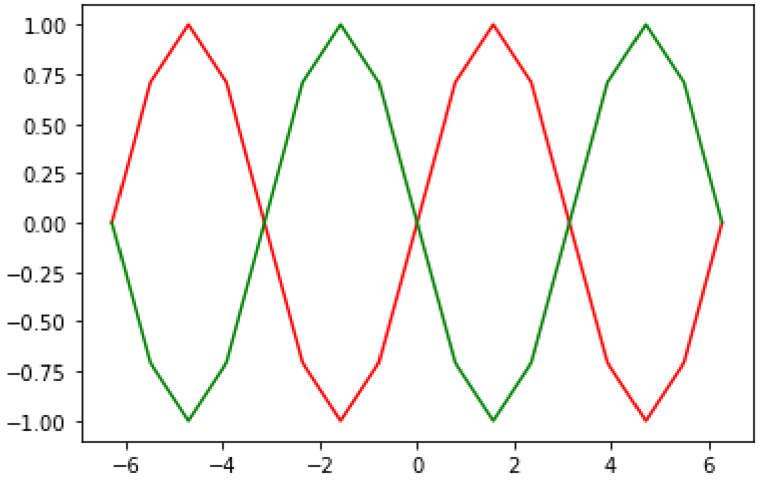
Part 2: 2D Plots

In [1]:

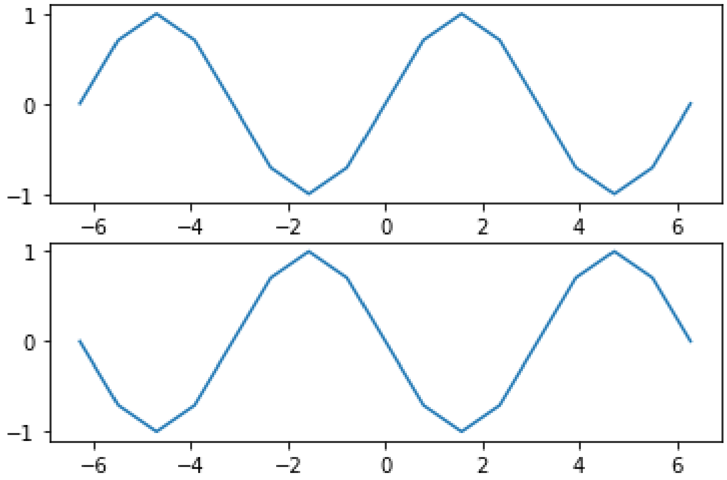
```
[Q1] Line Plot:
Create a line plot of sin(x) and cos(x + π/2) for -2π < x < 2π where x incre

Make the sin(x) graph red and make the cos(x+π/2) graph green
Put both lines onto the same plot
Using the same info as above, make a subplot with 2 different graphs-
one graph for sin(x) and
one graph for cos(x+π/2)


x = -6.283, -5.498, -4.712, -3.927, -3.142, -2.356, -1.571, -.7854, 0, .785



sin(x) = 0, .70711, 1, .70711, 0, -.70711, -1, -.70711, 0, .70711, 1, .7071



cos(x + π/2) = 0, -.70711, -1, -.70711, 0, .70711, 1, .70711, 0, -.70711, -
```

```
  File "/var/folders/ty/4py3xkds2xd231nxqy3v22j40000gp/T/ipykernel_39243/
500080704.py", line 1
    Line Plot:
         ^
SyntaxError: invalid syntax
```

```
In [11]: import matplotlib.pyplot as plt

         x = [-6.283, -5.498, -4.712, -3.927, -3.142, -2.356, -1.571, -.7854, 0, .78
         sin = [0, .70711, 1, .70711, 0, -.70711, -1, -.70711, 0, .70711, 1, .70711,
         cos = [0, -.70711, -1, -.70711, 0, .70711, 1, .70711, 0, -.70711, -1, -.707

         plt.plot(x,sin,'red')
         plt.plot(x,cos,'green')
         plt.show()

         fig , ax = plt.subplots(2)
         ax[0].plot(x,sin)
         ax[1].plot(x,cos)
         fig.show()
```

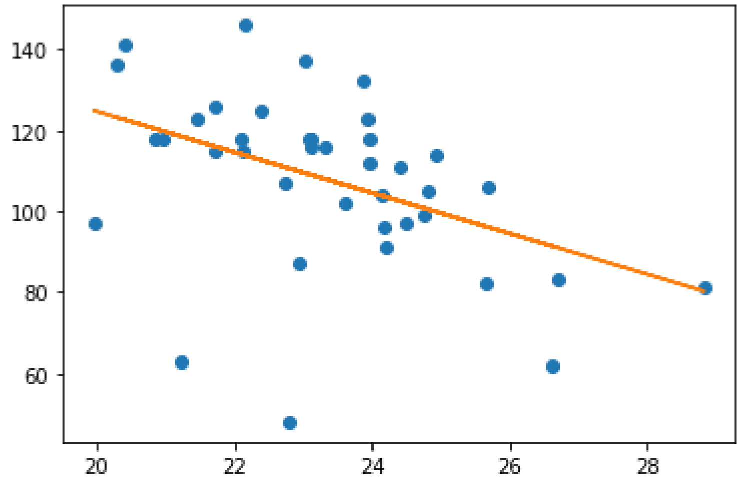


```
<ipython-input-11-f1b66221ea4e>:14: UserWarning: Matplotlib is currently
using module://ipykernel.pylab.backend_inline, which is a non-GUI backen
d, so cannot show the figure.
  fig.show()
```



```
In [ ]: [Q2] Scatter Plot:
        Using the following data about winter temperatures affecting the number of



        Mean Temperature (in Fahrenheit): 22.94, 23.02, 25.68, 19.96, 24.80, 23.98,

        Days of Ice: 87, 137, 106, 97, 105, 118, 118, 136, 91, 107, 96, 114, 125, 1
```

```
In [16]: import numpy as np
         Mean=[22.94, 23.02, 25.68, 19.96, 24.80, 23.98, 22.10, 20.30, 24.20, 22.74,
         mean = np.array(Mean)
         Days=[87, 137, 106, 97, 105, 118, 118, 136, 91, 107, 96, 114, 125, 115, 118
         days=np.array(Days)
         m, b = np.polyfit(mean,days, 1)
         plt.plot(mean,days,'o')
         plt.plot(mean,m*mean+b)
         plt.show()
```



```
In [ ]:
```