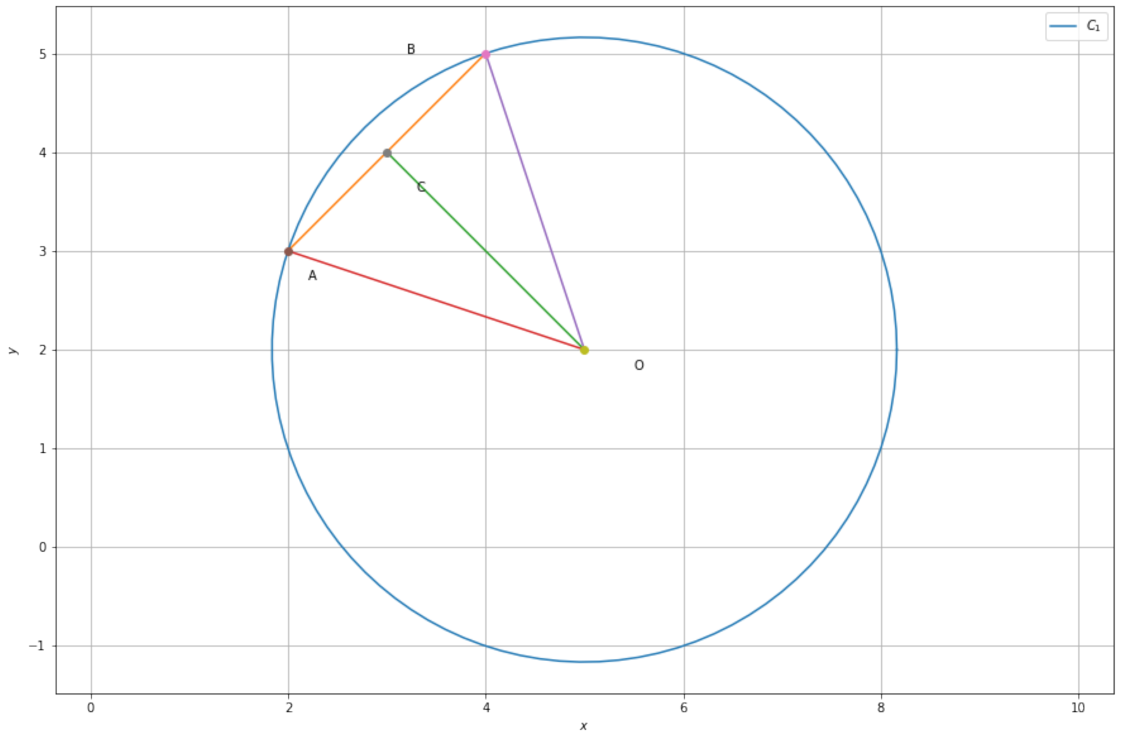
Circles

March 28, 2021

[7]:
```
import numpy as np
import matplotlib.pyplot as plt
from coeffs import *
import cmath
```

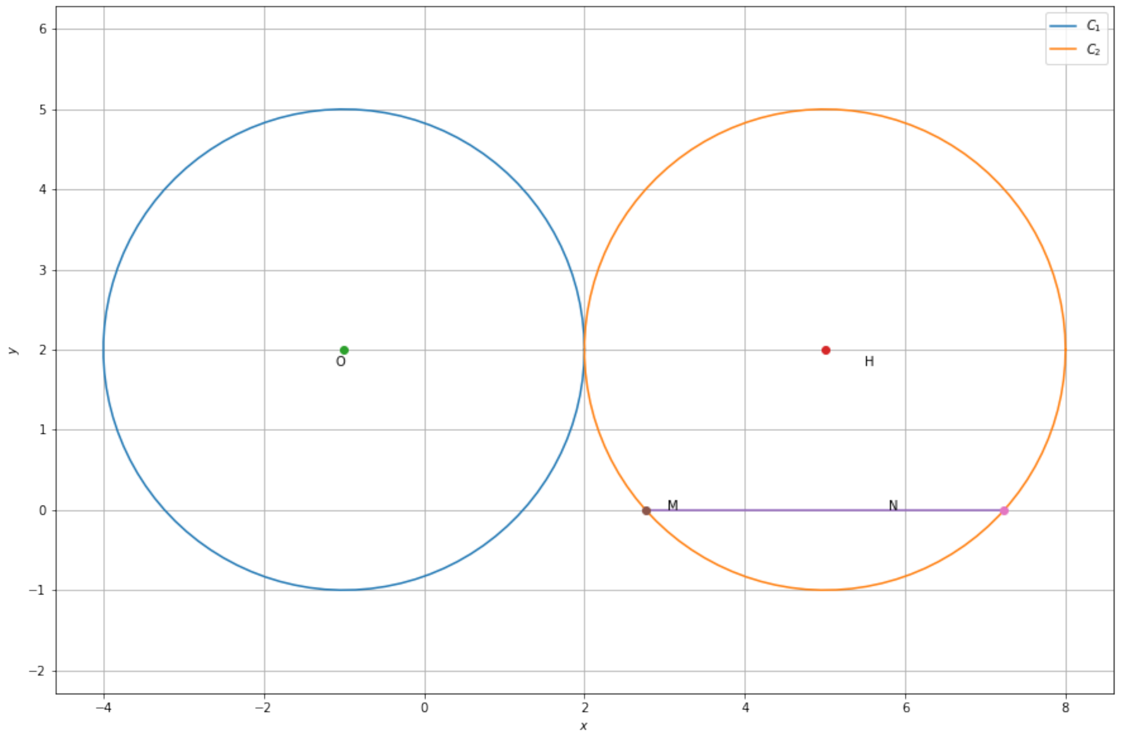
[8]:
```
A=np.array([2,3])
B=np.array([4,5])
C=(A+B)/2
m=A-B
n=np.array([-1,4])
N=np.vstack((m,n))
Q=np.array([m.T@C,3])
O=np.linalg.inv(N)@Q
d=O-A
r=np.linalg.norm(d)
len = 100
theta = np.linspace(0,2*np.pi,len)
x_circ = np.zeros((2,len))
x_circ[0,:] = r*np.cos(theta)
x_circ[1,:] = r*np.sin(theta)
x_circ = (x_circ.T + O).T
f1 = plt.figure(figsize=(15,10))
plt.plot(x_circ[0,:],x_circ[1,:],label='$C_1$')
x_AB=line_gen(A,B)
plt.plot(x_AB[0,:],x_AB[1,:])
x_CO=line_gen(C,O)
plt.plot(x_CO[0,:],x_CO[1,:])
x_AO=line_gen(A,O)
plt.plot(x_AO[0,:],x_AO[1,:])
x_BO=line_gen(B,O)
plt.plot(x_BO[0,:],x_BO[1,:])
plt.plot(A[0], A[1], 'o')
plt.text(A[0] * (1 + 0.1), A[1] * (1 - 0.1) , 'A')
plt.plot(B[0], B[1], 'o')
plt.text(B[0] * (1 - 0.2), B[1] * (1) , 'B')
plt.plot(C[0], C[1], 'o')
plt.text(C[0] * (1 + 0.1), C[1] * (1 - 0.1) , 'C')
plt.plot(O[0], O[1], 'o')
plt.text(O[0] * (1 + 0.1), O[1] * (1 - 0.1) , 'O')
plt.axis('equal')
plt.xlabel('$x$');plt.ylabel('$y$')
plt.legend(loc='best');plt.grid()
plt.show()
```



[14]:
```
O=np.array([-1,2])
r=np.sqrt(O.T@O+4)
print(r)
P=np.array([2,2])
H=(2*P)-O
h=3
len = 100
theta = np.linspace(0,2*np.pi,len)
x_circ = np.zeros((2,len))
x_circ[0,:] = r*np.cos(theta)
x_circ[1,:] = r*np.sin(theta)
x_circ = (x_circ.T + O).T
f2 = plt.figure(figsize=(15,10))
plt.plot(x_circ[0,:],x_circ[1,:],label='$C_1$')

x2_circ = np.zeros((2,len))
```

```
x2_circ[0,:] = h*np.cos(theta)
x2_circ[1,:] = h*np.sin(theta)
x2_circ = (x2_circ.T + H).T
plt.plot(x2_circ[0,:],x2_circ[1,:],label='$C_2$')
plt.plot(O[0], O[1], 'o')
plt.text(O[0] * (1 + 0.1), O[1] * (1 - 0.1) , 'O')
plt.plot(H[0], H[1], 'o')
plt.text(H[0] * (1 + 0.1), H[1] * (1 - 0.1) , 'H')



# Solve the quadratic equation ax**2 + bx + c = 0

# import complex math module

L=np.array([1,0])
l=np.linalg.norm(L)
a = (l*l)
b = -(2*(L.T@H))
t= np.linalg.norm(H)
c=(t*t)-(h*h)
# To take coefficient input from the users
# a = float(input('Enter a: '))
# b = float(input('Enter b: '))
# c = float(input('Enter c: '))

# calculate the discriminant
d = (b**2) - (4*a*c)

# find two solutions
sol1 = (-b-cmath.sqrt(d))/(2*a)
sol1 = sol1.real
sol2 = (-b+cmath.sqrt(d))/(2*a)
sol2 = sol2.real

print('The solution are {0} and {1}'.format(sol1,sol2))

M=sol1*L
N=sol2*L
x_MN=line_gen(M,N)
plt.plot(x_MN[0,:],x_MN[1,:])
plt.plot(M[0], M[1], 'o')
plt.text(M[0] * (1 + 0.1), M[1] * (1 - 0.1) , 'M')
plt.plot(N[0], N[1], 'o')
plt.text(N[0] * (1 - 0.2), N[1] * (1) , 'N')
plt.axis('equal')
plt.xlabel('$x$');plt.ylabel('$y$')
plt.legend(loc='best');plt.grid()
plt.show()
```

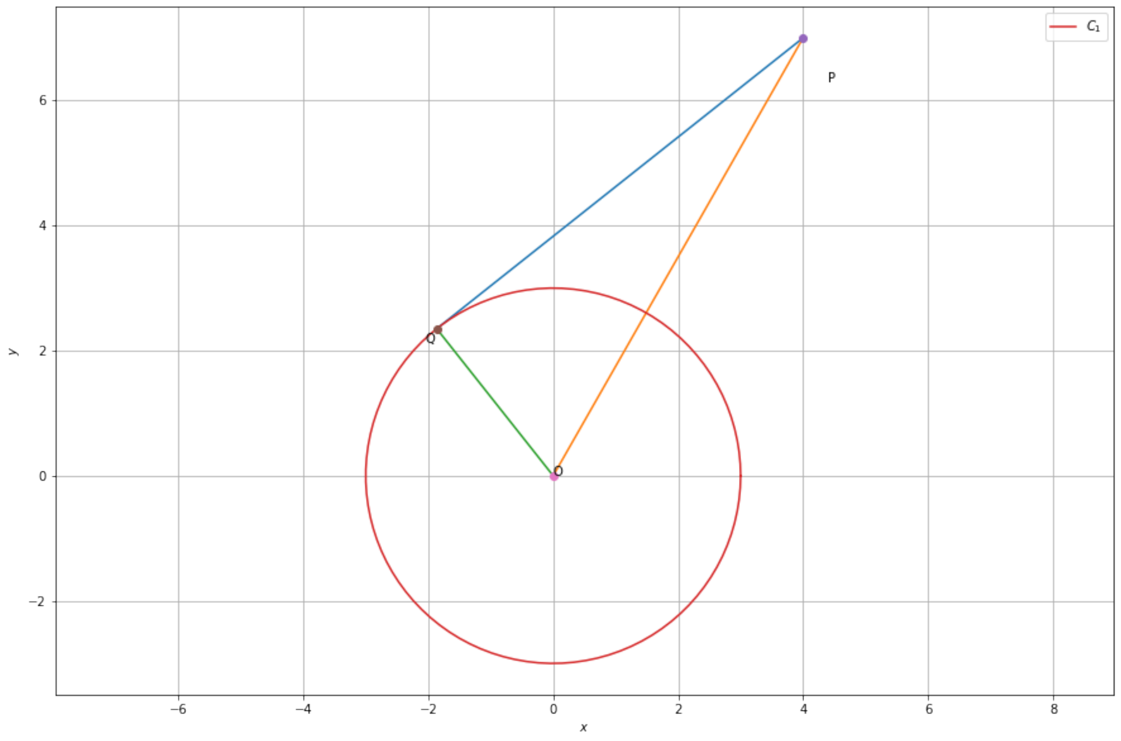
```
3.0
The solution are 2.7639320225002093 and 7.236067977499791
```



```
[17]: O=np.array([0,0])
r=np.sqrt(9)
P=np.array([4,7])
F=np.eye(2)
E=(np.sqrt((P.T@P)-(r*r))/r)*omat+F
Q=np.linalg.inv(E)@P
f3 = plt.figure(figsize=(15,10))
x_PQ=line_gen(P,Q)
plt.plot(x_PQ[0,:],x_PQ[1,:])
x_PO=line_gen(P,O)
plt.plot(x_PO[0,:],x_PO[1,:])
x_OQ=line_gen(O,Q)
plt.plot(x_OQ[0,:],x_OQ[1,:])
len = 100
theta = np.linspace(0,2*np.pi,len)
x_circ = np.zeros((2,len))
x_circ[0,:] = r*np.cos(theta)
```

```
x_circ[1,:] = r*np.sin(theta)
x_circ = (x_circ.T + O).T
plt.plot(x_circ[0,:],x_circ[1,:],label='$C_1$')
plt.plot(P[0], P[1], 'o')
plt.text(P[0] * (1 + 0.1), P[1] * (1 - 0.1) , 'P')
plt.plot(Q[0], Q[1], 'o')
plt.text(Q[0] * (1 + 0.1), Q[1] * (1 - 0.1) , 'Q')
plt.plot(O[0], O[1], 'o')
plt.text(O[0] * (1 + 0.1), O[1] * (1 - 0.1) , 'O')
h=P-Q
print(h)
plt.axis('equal')
plt.xlabel('$x$');plt.ylabel('$y$')
plt.legend(loc='best');plt.grid()
plt.show()
```

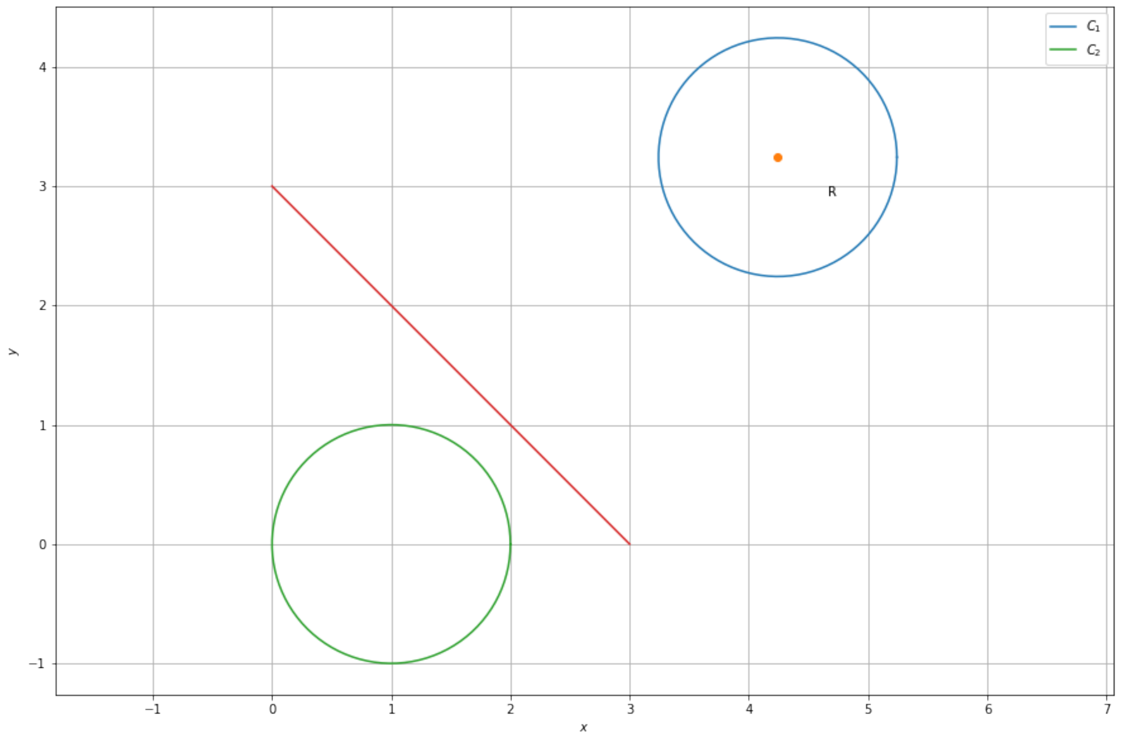
```
[5.86384016 4.6492342 ]
```



```
[18]: O=np.array([1,0])
      r=(O.T@O)
      m=np.array([1,-1])
      n=np.array([1,1])
      c=3
```

```
n=n.reshape((2,1))
m=m.reshape((2,1))

R1 = ((m@m.T-n@n.T)/(m.T@m+n.T@n))@O
print(R1)
R2 = c*n/np.linalg.norm(n)
R2=R2.reshape((1,2))
R = 2*(R1+R2)
R = np.array(R)[0]
print(R)
len=100
theta = np.linspace(0,2*np.pi,len)
x_circ = np.zeros((2,len))
x_circ[0,:] = r*np.cos(theta)
x_circ[1,:] = r*np.sin(theta)
x_circ = (x_circ.T + R).T
f4 = plt.figure(figsize=(15,10))
plt.plot(x_circ[0,:],x_circ[1,:],label='$C_1$')
plt.plot(R[0], R[1], 'o')
plt.text(R[0] * (1 + 0.1), R[1] * (1 - 0.1) , 'R')
len=100
theta = np.linspace(0,2*np.pi,len)
x_circ = np.zeros((2,len))
x_circ[0,:] = r*np.cos(theta)
x_circ[1,:] = r*np.sin(theta)
x_circ = (x_circ.T + O).T
plt.plot(x_circ[0,:],x_circ[1,:],label='$C_2$')
A = np.array([3,0])
B = np.array([0,3])

x_AB=line_gen(A,B)
plt.plot(x_AB[0,:],x_AB[1,:])
plt.axis('equal')
plt.xlabel('$x$');plt.ylabel('$y$')
plt.legend(loc='best');plt.grid()
plt.show()
```

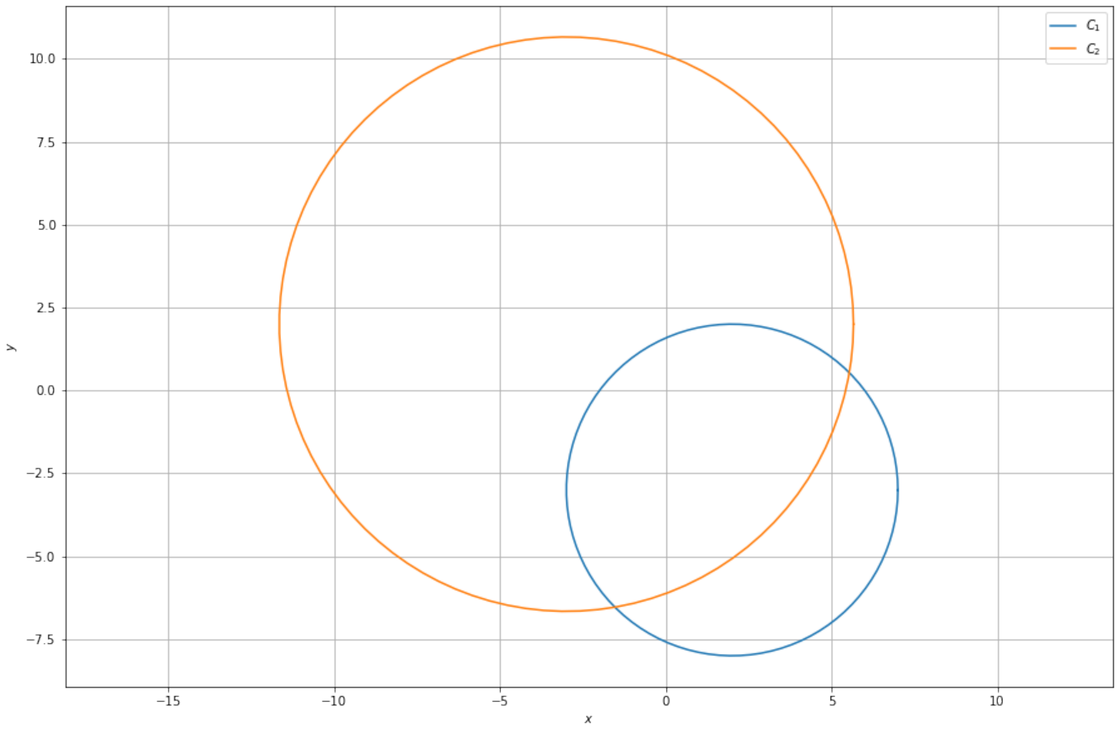
```
[ 0.  -0.5]
[4.24264069 3.24264069]
```



```
[19]: O1=np.array([2,-3])
      O2=np.array([-3,2])
      r1=np.sqrt(O1.T@O1+12)
      e=O1-O2
      d=np.linalg.norm(e)
      r2=np.sqrt((d*d)+(r1*r1))
      print(r1)

      len = 100
      theta = np.linspace(0,2*np.pi,len)
      x_circ = np.zeros((2,len))
      x_circ[0,:] = r1*np.cos(theta)
      x_circ[1,:] = r1*np.sin(theta)
      x_circ = (x_circ.T + O1).T
      f5 = plt.figure(figsize=(15,10))
      plt.plot(x_circ[0,:],x_circ[1,:],label='$C_1$')

      len = 100
      theta = np.linspace(0,2*np.pi,len)
      x_circ = np.zeros((2,len))
      x_circ[0,:] = r2*np.cos(theta)
      x_circ[1,:] = r2*np.sin(theta)
      x_circ = (x_circ.T + O2).T
```

```
plt.plot(x_circ[0,:],x_circ[1,:],label='$C_2$')
plt.axis('equal')
plt.xlabel('$x$');plt.ylabel('$y$')
plt.legend(loc='best');plt.grid()
plt.show()
```

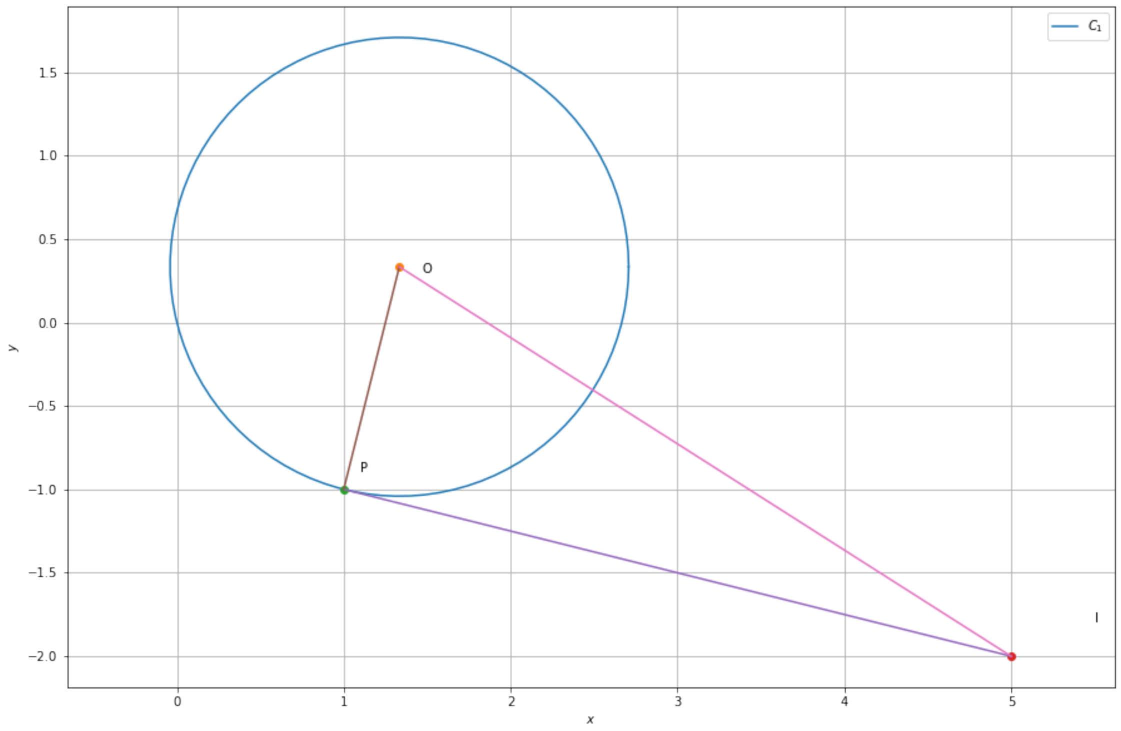
```
5.0
```



[20]:
```
P=np.array([1,-1])
m=np.array([2,1])
n=np.array([1,-1])
N=np.vstack((m,n))
Q=np.array([3,1])
O=np.linalg.inv(N)@Q
print(O)
I=P+np.array([4,-1])

k=O-P
r=np.linalg.norm(k)
len = 100
theta = np.linspace(0,2*np.pi,len)
x_circ = np.zeros((2,len))
x_circ[0,:] = r*np.cos(theta)
x_circ[1,:] = r*np.sin(theta)
```

```
x_circ = (x_circ.T + O).T
f6 = plt.figure(figsize=(15,10))
plt.plot(x_circ[0,:],x_circ[1,:],label='$C_1$')
plt.plot(O[0], O[1], 'o')
plt.text(O[0] * (1 + 0.1), O[1] * (1 - 0.1) , 'O')
plt.plot(P[0], P[1], 'o')
plt.text(P[0] * (1 + 0.1), P[1] * (1 - 0.1) , 'P')
plt.plot(I[0], I[1], 'o')
plt.text(I[0] * (1 + 0.1), I[1] * (1 - 0.1) , 'I')
x_PI=line_gen(P,I)
plt.plot(x_PI[0,:],x_PI[1,:])
x_PO=line_gen(P,O)
plt.plot(x_PO[0,:],x_PO[1,:])
x_OI=line_gen(O,I)
plt.plot(x_OI[0,:],x_OI[1,:])
plt.axis('equal')
plt.xlabel('$x$');plt.ylabel('$y$')
plt.legend(loc='best');plt.grid()
plt.show()
```

```
[1.33333333 0.33333333]
```



```
[ ]:
```