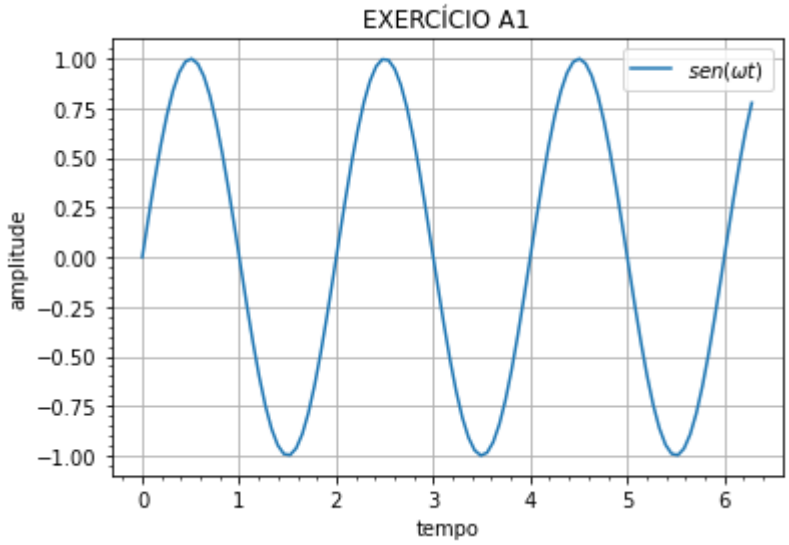
Open in Colab

Lista 1 – Treinamento com uma linguagem de programação científica **negrito**

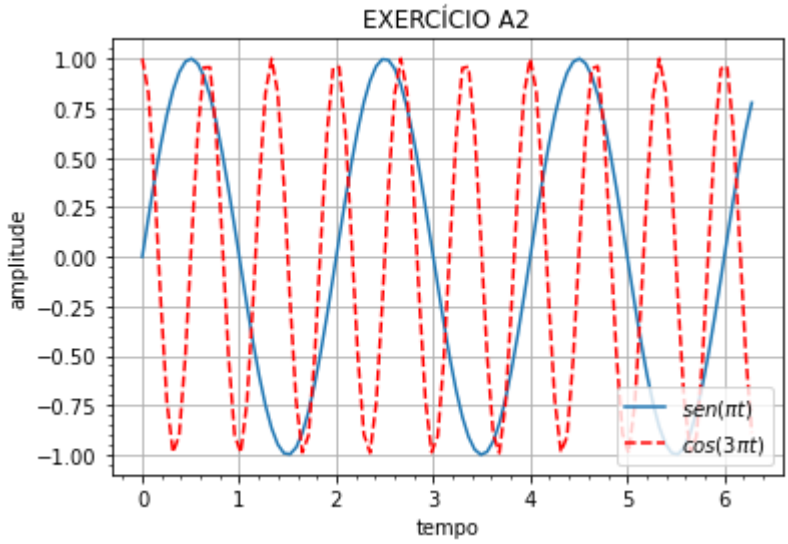
Luciano Silva do Nascimento

In [ ]:
```
# a1) Construção de gráficos x-y (intervalo 0 a 2π), atribuindo valores para ω1 e ω2.

from pylab import *
t = linspace(0,2*pi, 100)
plot(t,sin(pi*t), label='$sen(\\omega t)$')
xlabel('tempo')
ylabel('amplitude')
title("EXERCÍCIO A1")
legend()
grid()
minorticks_on()
```

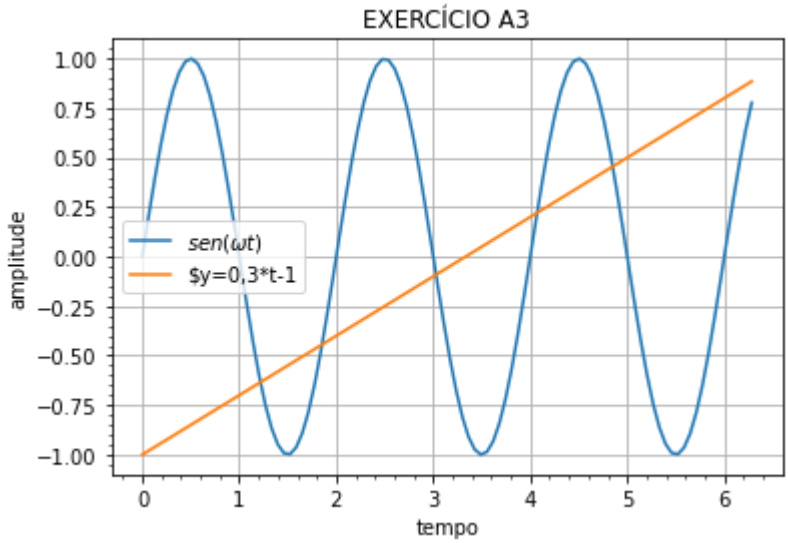


In [ ]:

In [ ]:
```
# a2) Construção de gráficos x-y (intervalo 0 a 2π), atribuindo valores para ω1 e ω2.

from pylab import *
t = linspace(0,2*pi, 100)
f1=sin(pi*t)
f2=cos(3*pi*t)
plot(t,f1, label='$sen(\\pi t)$')
plot(t,f2, 'r--', label='$cos(3\\pi t)$')
xlabel('tempo')
ylabel('amplitude')
title("EXERCÍCIO A2")
legend(loc='lower right')
grid()
minorticks_on()
```



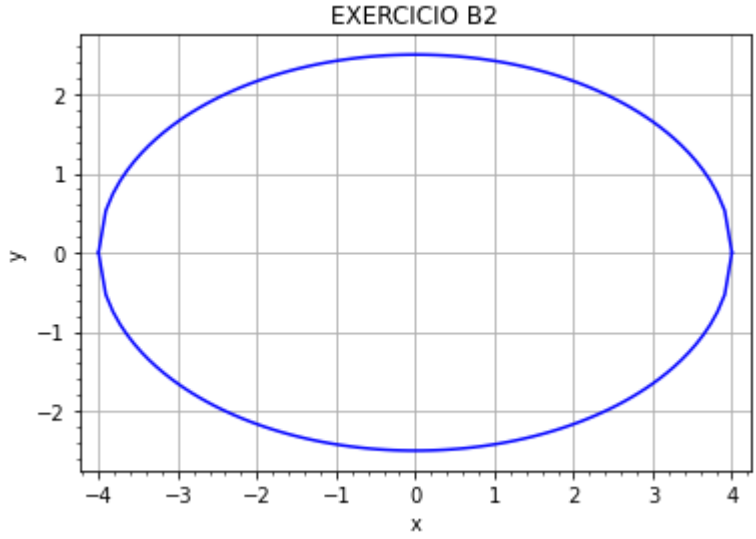
In [ ]:
```
# a3) Construção de gráficos x-y (intervalo 0 a 2π), atribuindo valores para ω1 e ω2.

def y(a,b,t):
   return a*t+b

t = linspace(0,2*pi, 100)
plot(t,sin(pi*t), label='$sen(\\omega t)$')
plot(t,y(0.3,-1,t),label='$y=0,3*t-1')
xlabel('tempo')
ylabel('amplitude')
title("EXERCÍCIO A3")
legend()
grid()
minorticks_on()
```



In [ ]:
```
# b1) Construção de gráficos x-y

x=linspace(-5,5,100)
y=sqrt(5**2-x**2)
plot(x,y,'b-',label='$y^2 + x^2 = 5^5$')
plot(x,-y, 'b-')
axis('equal')
xlabel('x')
ylabel('y')
title("EXERCICIO B1")
legend
grid()
minorticks_on()
```

In [ ]:
```
# b2) Construção de gráficos x-y

x=linspace(-4,5,100)
y=sqrt(2.5**2*(1-x**2/4**2))
plot(x,y,'b-', label="$\\frac{x^2}{4^2}+\\frac{y^2}{2,5^2}=1$")
plot(x,-y, 'b-')
axis('equal')
xlabel('x')
ylabel('y')
title("EXERCICIO B2")
legend
grid()
minorticks_on()
```

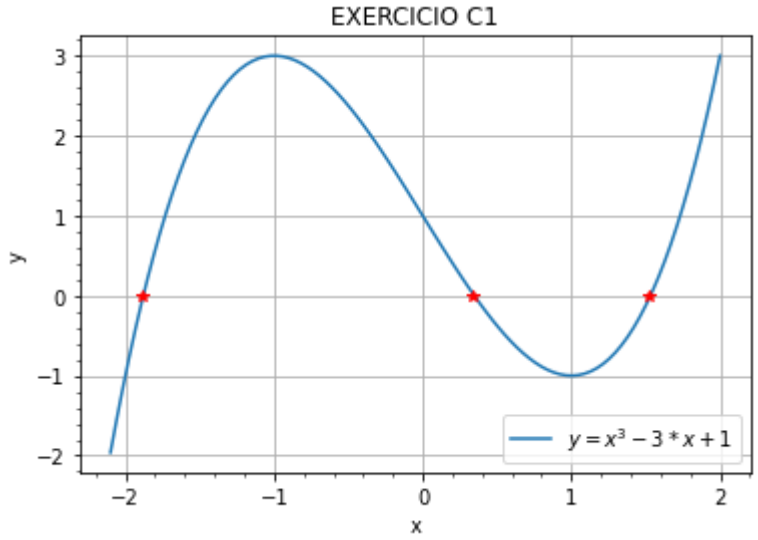
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: RuntimeWarning: invalid val
ue encountered in sqrt
  after removing the cwd from sys.path.
```



In [ ]:
```
# c1) Zeros de funções (métodos gráficos)

def y(x):
    return x**3-3*x+1
x = linspace(-2.1,2,100)
rz=[-1.89, 0.33, 1.52]
plot(x,y(x), label='$y=x^3-3*x+1$')
for i in range(len(rz)):
    plot(rz[i], 0, 'r*')
xlabel('x')
```

```
ylabel('y')
title("EXERCICIO C1")
legend()
grid()
minorticks_on()
print("As raízes são:", rz)
```

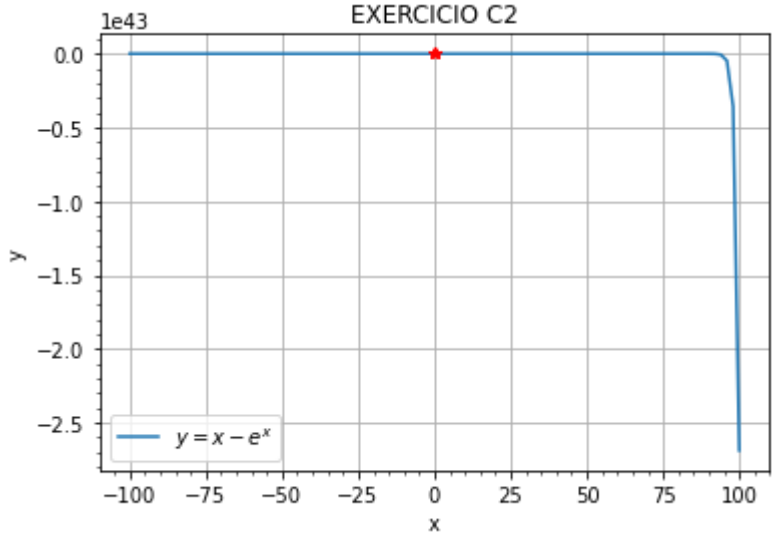
```
As raízes são: [-1.89, 0.33, 1.52]
```



In [ ]:

```
# c2) Zeros de funções (métodos gráficos)

def y(x):
    return x-exp(x)
x = linspace(-100,100,100)
rz=[-1.89, 0.33, 1.52]
plot(x,y(x), label='$y=x-e^x$')
for i in range(len(rz)):
    plot(rz[1], 0, 'r*')
xlabel('x')
ylabel('y')
title("EXERCICIO C2")
legend()
grid()
minorticks_on()
print("As raízes são:", rz)
```

```
As raízes são: [-1.89, 0.33, 1.52]
```

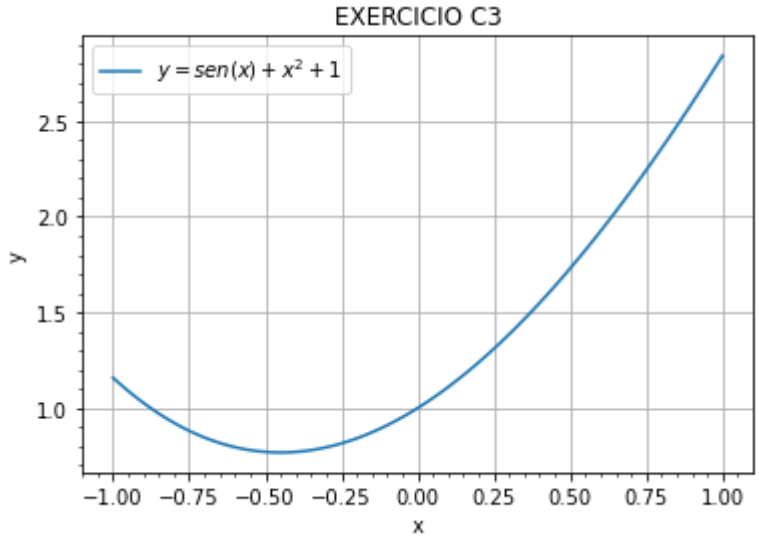


In [ ]:

```
# c3) Zeros de funções (métodos gráficos)
```

```
def y(x):
    return sin(x) + x**2 +1
x = linspace(-1,1,100)
plot(x,y(x), label='$y=sen(x)+x^2+1$')
xlabel('x')
ylabel('y')
title("EXERCICIO C3")
legend()
grid()
minorticks_on()
print("As raízes são:", rz)
```

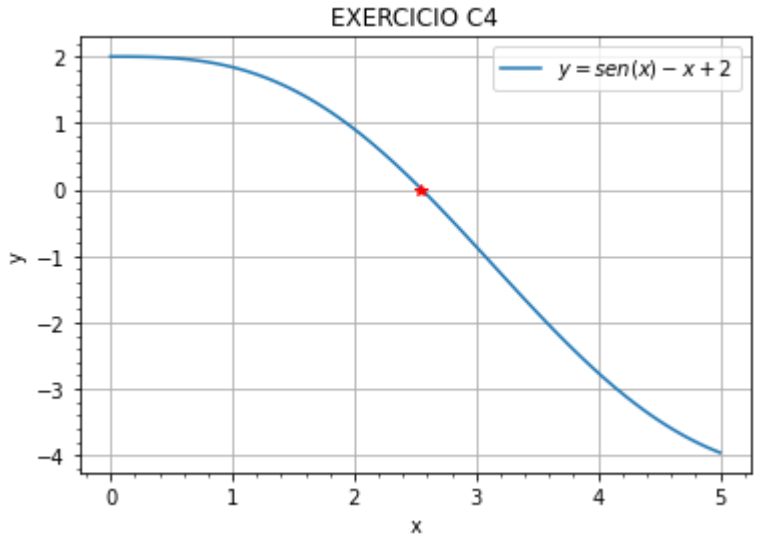
```
As raízes são: [2.55]
```



In [ ]:

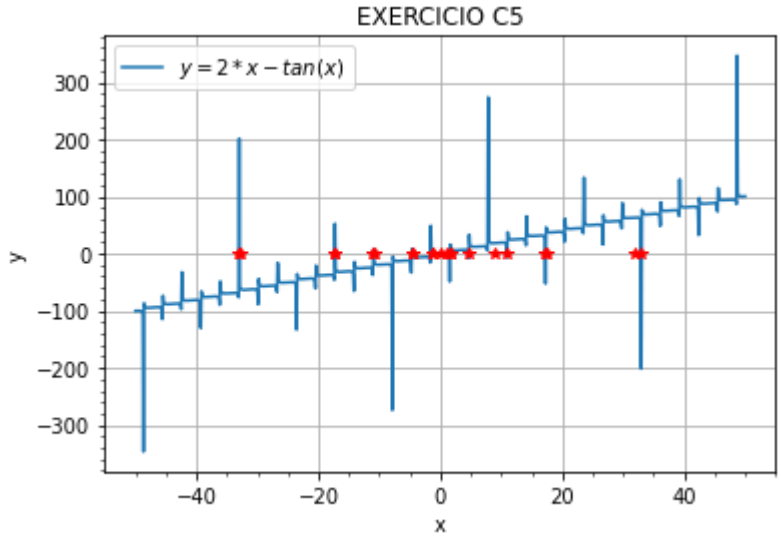
```
# c4) Zeros de funções (métodos gráficos)

def y(x):
    return sin(x) - x + 2
x = linspace(0,5,100)
rz=[2.55]
plot(x,y(x), label='$y=sen(x)- x + 2$')
for i in range(len(rz)):
    plot(rz[i], 0, 'r*')
xlabel('x')
ylabel('y')
title("EXERCICIO C4")
legend()
grid()
minorticks_on()
print("As raízes são:", rz)
```

```
As raízes são: [2.55]
```



In...
```
# c5) Zeros de funções (métodos gráficos)

def y(x):
    return 2*x-tan(x)
x = linspace(-50,50,1000)
rz=[-33.06,-32.9,-17.32,-17.2,-10.98,-10.92,-4.68,-4.58,1.62,-1.16,-1.16,0,1.16,1.62,4.5
plot(x,y(x), label='$y=2*x-tan(x)$')
for i in range(len(rz)):
    plot(rz[i], 0, 'r*')
xlabel('x')
ylabel('y')
title("EXERCICIO C5")
legend()
grid()
minorticks_on()
print("As raízes são:", rz)
```

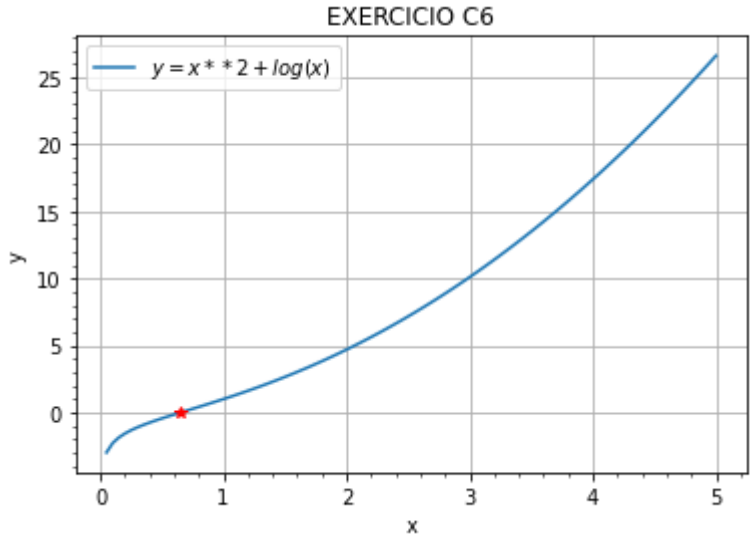
```
As raízes são: [-33.06, -32.9, -17.32, -17.2, -10.98, -10.92, -4.68, -4.58, 1.62, -1.16, -
1.16, 0, 1.16, 1.62, 4.58, 10.92, 17.2, 17.32, 32, 9, 33.06]
```



In [ ]:
```
# c6) Zeros de funções (métodos gráficos)

def y(x):
    return x**2+log(x)
x = linspace(0,5,100)
rz=[0.653]
plot(x,y(x), label='$y=x**2+log(x)$')
for i in range(len(rz)):
    plot(rz[i], 0, 'r*')
```

```
xlabel('x')
ylabel('y')
title("EXERCICIO C6")
legend()
grid()
minorticks_on()
print("As raízes são:", rz)
```

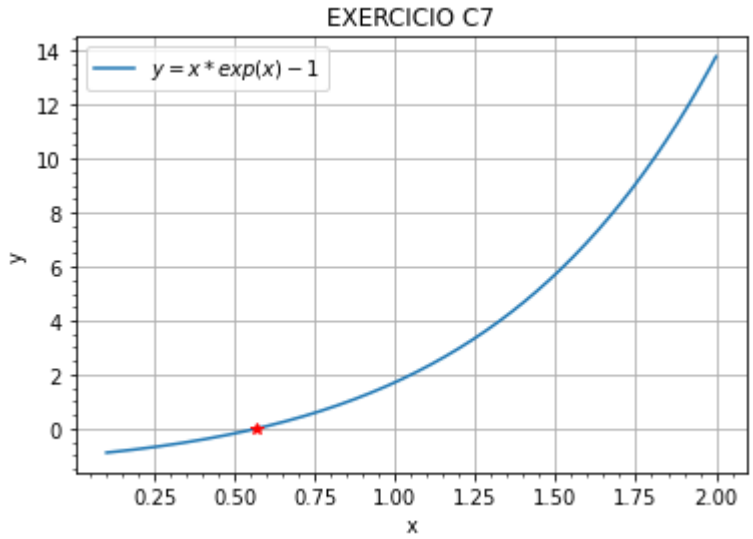
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: RuntimeWarning: divide by z
ero encountered in log
  after removing the cwd from sys.path.
As raízes são: [0.653]
```



In [ ]:

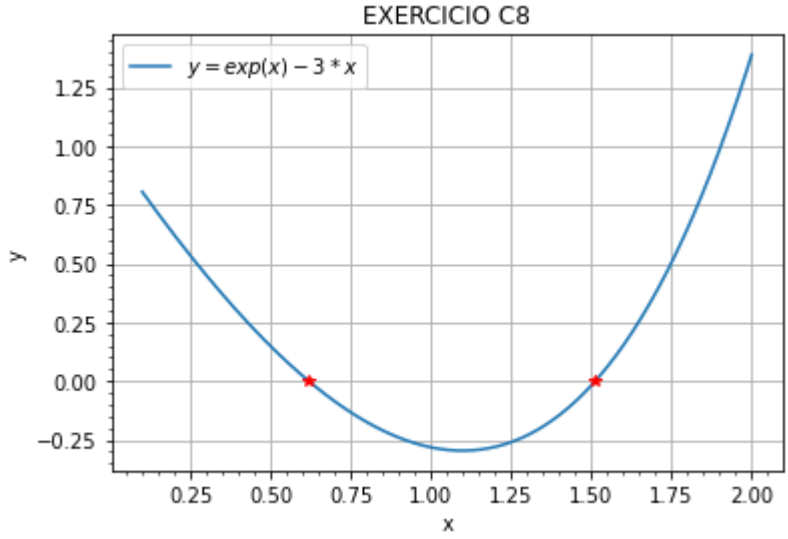
```
# c7) Zeros de funções (métodos gráficos)

def y(x):
    return x*exp(x)-1
x = linspace(0.1,2,1000)
rz=[0.567]
plot(x,y(x), label='$y=x*exp(x)-1$')
for i in range(len(rz)):
    plot(rz[i], 0, 'r*')
xlabel('x')
ylabel('y')
title("EXERCICIO C7")
legend()
grid()
minorticks_on()
print("As raízes são:", rz)
```

```
As raízes são: [0.567]
```



In [ ]:
```
# c8) Zeros de funções (métodos gráficos)

def y(x):
    return exp(x)-3*x
x = linspace(0.1,2,1000)
rz=[0.619, 1.5121]
plot(x,y(x), label='$y=exp(x)-3*x$')
for i in range(len(rz)):
    plot(rz[i], 0, 'r*')
xlabel('x')
ylabel('y')
title("EXERCICIO C8")
legend()
grid()
minorticks_on()
print("As raízes são:", rz)
```

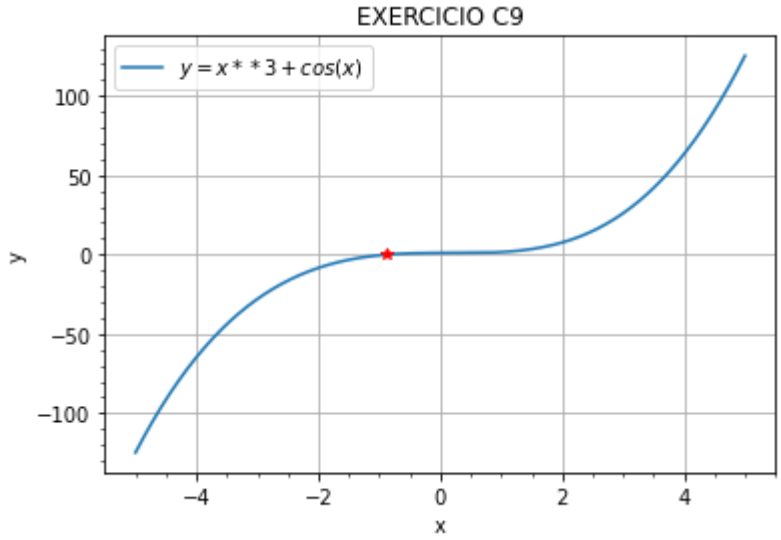
```
As raízes são: [0.619, 1.5121]
```



In [ ]:
```
# c9) Zeros de funções (métodos gráficos)

def y(x):
    return x**3+cos(x)
x = linspace(-5,5,1000)
rz=[-0.866]
plot(x,y(x), label='$y=x**3+cos(x)$')
for i in range(len(rz)):
    plot(rz[i], 0, 'r*')
xlabel('x')
```

```
ylabel('y')
title("EXERCICIO C9")
legend()
grid()
minorticks_on()
print("As raízes são:", rz)
```

```
As raízes são: [-0.866]
```



In ...

```
# d) Desenvolva um programa que, aplicando o método da bisecção, resolva as seguintes q

def bissecao(f,a,b,tol=1e-9,Nmax=50):
  n=1
  while n<Nmax:
    c=(b+a)/2
    #print(n,'f(c)=', f(c))
    if f(c)==0 or ((b + a)/2)<tol:
       print('raiz=',c)
       print('erro=',f(c))
       if f(c)<=1e-2:
         return c
    elif sign(f(c)*f(a))>0:
      a=c
    else:
      b=c
    n +=1
print('Raízes não encontradas')
```

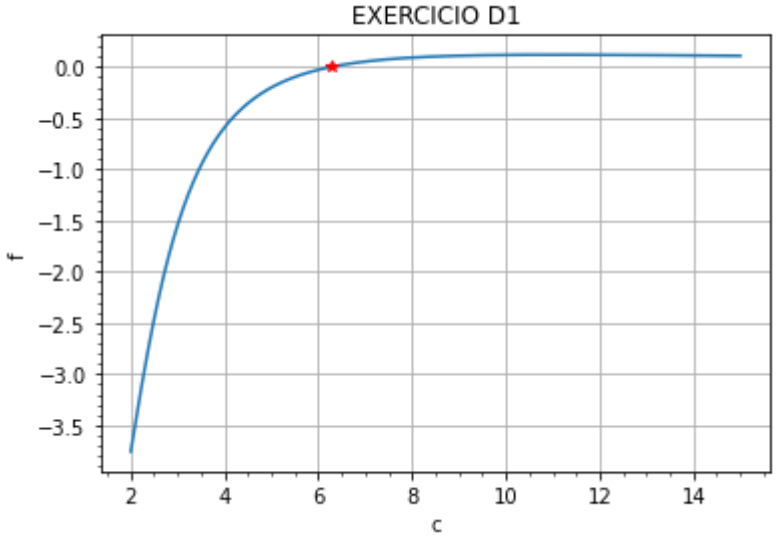
```
Raízes não encontradas
```

In ...

```
# d1) Desenvolva um programa que, aplicando o método da bisecção, resolva as seguintes

def f(R):
  return sin(8/R) - 6/R
#Verificando o intervalo de valores
x=linspace(2,15,1000)
plot(x,f(x))
xlabel('c')
ylabel('f')
title("EXERCICIO D1")
grid()
minorticks_on()

#Resolução
r=bissecao(f,a=4,b=8)
plot(6.27,0,'r*')
#Ângulo teta
```

```
th=arctan(6/6.27)
#Altura h
h=6.27*(1-cos(th))
print('A altura de', h, 'm')
```

```
A altura de 1.7399771127621582 m
```



In ...

```
# d2) Desenvolva um programa que, aplicando o método da bisecção, resolva as seguintes

import matplotlib.pyplot as plt
import numpy as np



rho=1000
def f(th):
  return th-sin(th)-430*2/(9.81*rho*(0.3)**2)

#Verificando o intervalo de valores
x=linspace(0, 2*pi, 1000)
plot(x,f(x))
xlabel('th[rad]')
ylabel('f')
title("EXERCICIO D2")
grid()
minorticks_on()

#Resolução
th=bissecao(f,a=0,b=3)
h=1-cos((th/2)*100
print('A profundidade é', h, 'm')
#plot(th,0,'r*')
```

In ...

```
# d3) Desenvolva um programa que, aplicando o método da bisecção, resolva as seguintes

def f(r):
    gm_af=11.5
    gm_aq=9.96
    rho_t=60e-3
    waf=4*pi*r**3*gm_af/3
    wt=4*pi*r**2*rho_t*9.81
    waq=4*pi*r**3*gm_aq/3
    return waf-wt-waq-1300

    x=linspace(0,10,1000)
    plot(x,f(x))
```

```
        xlabel('r')
        ylabel('f')
    title("EXERCICIO D3")
    grid()
    minorticks_on()

    r=bissecao(f,a=0,b=10)
    plot(r,0,'r*')
    print('O diâmetro do baláo deverá ser de', 2*r, 'm')
```

In ...

```
    # d4) Desenvolva um programa que, aplicando o método da bisecção, resolva as seguintes

    def f(c):
        return -0.4+1.74*log(1e5*sqrt(c))-sqrt(1/c)
        x=linspace(0,10,1000)
        plot(x,f(x))
        xlabel('c')
        ylabel('f')
    title("EXERCICIO D4")
    grid()
    minorticks_on()

    c=bissecao(f,a=0.0001,b=0.1)
    plot(c,0,'r*')
    print('O diâmetro do baláo deverá ser de', 2*r, 'm')
```