In [1]:

```
%matplotlib inline
import numpy as np
from math import sqrt
from array import *
import matplotlib.pyplot as plt
import math
import matplotlib.ticker as ticker
from pylab import *
from mpl_toolkits.mplot3d import Axes3D
```

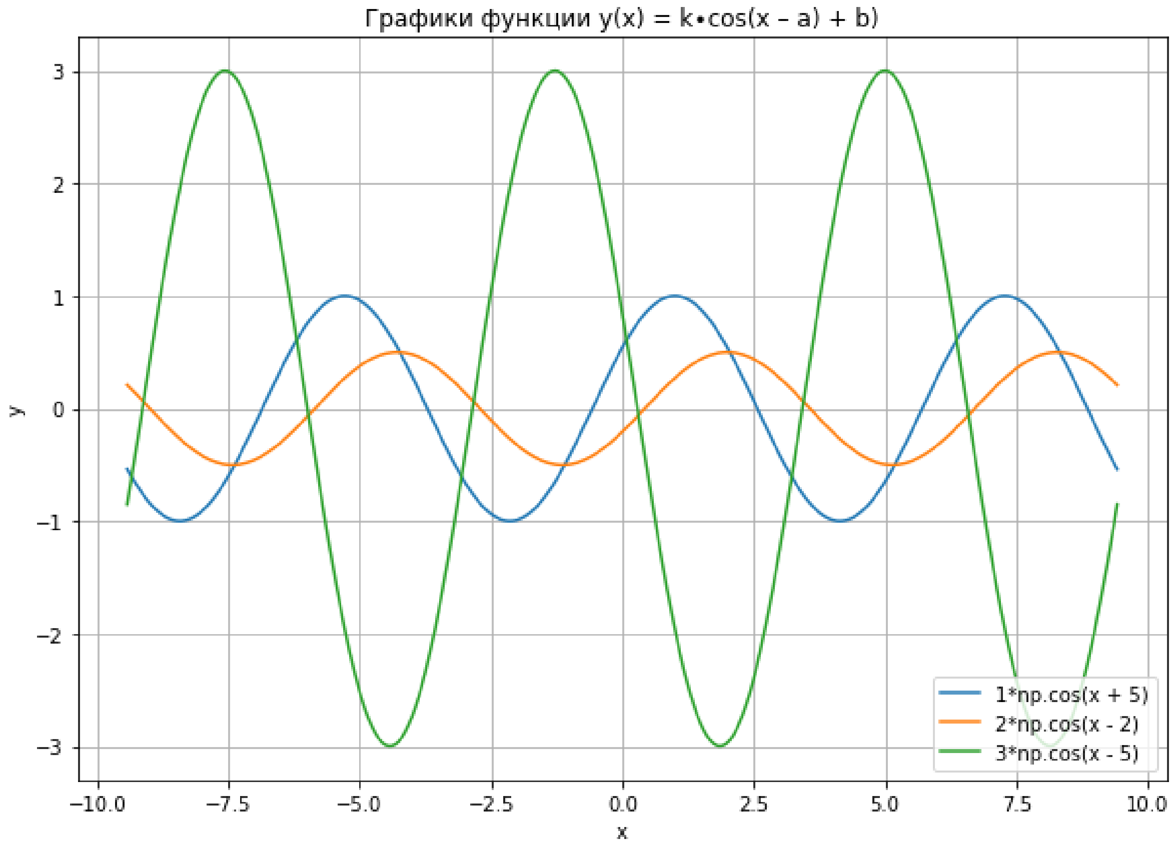
Задания к уроку 3.2

1. Задание

Нарисуйте график функции: y(x) = k·cos(x – a) + b для некоторых (2-3 различных) значений параметров k, a, b

In [2]:

```
x = np.linspace(-3*np.pi, 3*np.pi, 300)
plt.subplots(figsize=(10, 7))
plt.grid(True)
y = 1*np.cos(x - 1)
y2 = 0.5*np.cos(x - 2)
y3 = 3*np.cos(x - 5)
plt.plot(x,y, label = '1*np.cos(x + 5)')
plt.plot(x,y2, label = '2*np.cos(x - 2)')
plt.plot(x,y3, label = '3*np.cos(x - 5)')
plt.legend(loc = 4)
plt.xlabel("x")
plt.ylabel("y")
plt.title("Графики функции y(x) = k·cos(x – a) + b)", size=12)
plt.show()
```



2. Задание

Докажите, что при ортогональном преобразовании сохраняется расстояние между точками.

Ответ

Расстояние между точками на плоскости в декартовой системе координат определяется по формуле:

$$l = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

После ортогонального преобразования координаты x и y рассчитываются по следующим формулам:

$$X = (x - a)cos\alpha + (y - b)sin\alpha$$

$$Y = -(x - a)sin\alpha + (y - b)cos\alpha$$

Подставим эти формулы в формулу определения расстояний между двумя коориначами для определения расстояния между точками после ортогонального преобразования:

$$l = \sqrt{((x_2 - a)cos\alpha + (y_2 - b)sin\alpha - ((x_1 - a)cos\alpha + (y_1 - b)sin\alpha))^2 + (-(x_2 - a)sin\alpha + (y_2 - b)cos\alpha -}$$

Преобразуем данное выражение:

$$l = \sqrt{(x_2 cos\alpha - acos\alpha + y_2 sin\alpha - bsin\alpha - x_1 cos\alpha + acos\alpha - y_1 sin\alpha + bsin\alpha)^2 + (-(x_2 - a)sin\alpha + (y_2 -}$$

$$l = \sqrt{\begin{array}{l}(x_2 cos\alpha - acos\alpha + y_2 sin\alpha - bsin\alpha - x_1 cos\alpha + acos\alpha - y_1 sin\alpha + b\\ + (-x_2 sin\alpha + asin\alpha + y_2 cos\alpha - bcos\alpha + x_1 sin\alpha - asin\alpha - y_1 cos\alpha\end{array}}$$

$$l = \sqrt{(x_2 cos\alpha + y_2 sin\alpha - x_1 cos\alpha - y_1 sin\alpha)^2 + (-x_2 sin\alpha + y_2 cos\alpha + x_1 sin}$$

$$l = \sqrt{(x_2 cos\alpha - x_1 cos\alpha + y_2 sin\alpha - y_1 sin\alpha)^2 + (-x_2 sin\alpha + x_1 sin\alpha + y_2 cos}$$

$$l = \sqrt{(cos\alpha(x_2 - x_1) + sin\alpha(y_2 - y_1))^2 + (-sin\alpha(x_2 + x_1) + cos\alpha(y_2 -}$$

$$l = \sqrt{(cos\alpha(x_2 - x_1))^2 + 2cos\alpha(x_2 - x_1)sin\alpha(y_2 - y_1) + (sin\alpha(y_2 - y_1))^2 + (-sin\alpha(x_2 + x_1))^2 - 2sin\alpha(}$$

Проведем замену:

$$a = x_2 - x_1$$

$$b = y_2 - y_1$$

Перепишем уравнение:

$$l = \sqrt{(cos\alpha(a))^2 + 2cos\alpha(a)sin\alpha(b) + (sin\alpha(b))^2 + (-sin\alpha(a))^2 - 2sin\alpha(a)cos\alpha(b) + (cos\alpha(b))^2}$$

$$l = \sqrt{cos^2\alpha a^2 + 2cos\alpha asin\alpha b + sin^2\alpha b^2 + sin^2\alpha a^2 - 2sin\alpha acos\alpha b + cos^2\alpha b^2}$$

$$l = \sqrt{cos^2\alpha a^2 + cos^2\alpha b^2 + sin^2\alpha b^2 + sin^2\alpha a^2 + 2cos\alpha asin\alpha b - 2sin\alpha acos\alpha b}$$

$$l = \sqrt{cos^2\alpha(a^2 + b^2) + sin^2\alpha(b^2 + a^2) + 2cos\alpha asin\alpha b - 2sin\alpha acos\alpha b}$$

$$l = \sqrt{(a^2 + b^2)(cos^2\alpha + sin^2\alpha) + 2cos\alpha asin\alpha b - 2sin\alpha acos\alpha b}$$

$$l = \sqrt{(a^2 + b^2) * 1 + 2ab(cos\alpha sin\alpha - sin\alpha cos\alpha)}$$

$$l = \sqrt{(a^2 + b^2) * 1 + 2ab * 0}$$

$$l = \sqrt{(a^2 + b^2)}$$

Проведем обратную замену a и b и получим:

$$l = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Т.е. мы получили ту же формулу, что и до ортогонального преобразования. Следовательно при ортогональном преобразовании сохраняется расстояние между точками.

3. Задание (в программе)

1. Напишите код, который будет переводить полярные координаты в декартовы.

In [3]:

```
coordinates = str(input("Введите полярный радиус и полярный угол точки (в градусах) в поляр
print(f"Данная точка в декартовой системе координат будет иметь следующие координаты: ({int
```

```
Введите полярный радиус и полярный угол точки (в градусах) в полярной системе
координат через пробел: 3 60
Данная точка в декартовой системе координат будет иметь следующие координаты:
(1.50, 2.60)
```

2. Напишите код, который будет рисовать график окружности в полярных координатах.

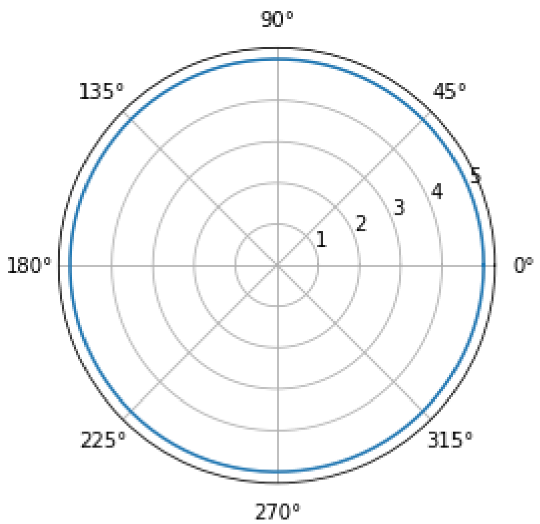
In [4]:

```
x = np.linspace(0, 2*np.pi, 100)
y = np.full(100, 5)

plt.polar(x, y)
```

Out[4]:

```
[<matplotlib.lines.Line2D at 0x83cbf48>]
```



In [5]:

```
# x = []
# y1 = []
# y2 = []
# R = int(input("Введите радиус окружности, которую Вы хотите построить: "))
# for i in range(-99999, 99999):
#     x_ = i/100000*R
#     x.append(x_)
#     y1.append(sqrt(R**2-(x_**2)))
#     y2.append(-sqrt(R**2-(x_**2)))

# plt.polar(x,y1)
# plt.polar(x,y2)
# plt.show()
```

3. Напишите код, который будет рисовать график отрезка прямой линии в полярных координатах.

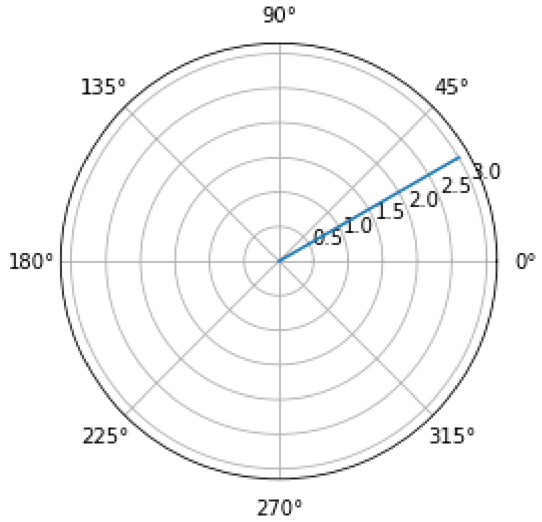
.

In [6]:

```
x = np.linspace(0, np.pi/6, 2)
y = (0, 3)
plt.polar(x, y)
```

Out[6]:

```
[<matplotlib.lines.Line2D at 0x83fafc8>]
```

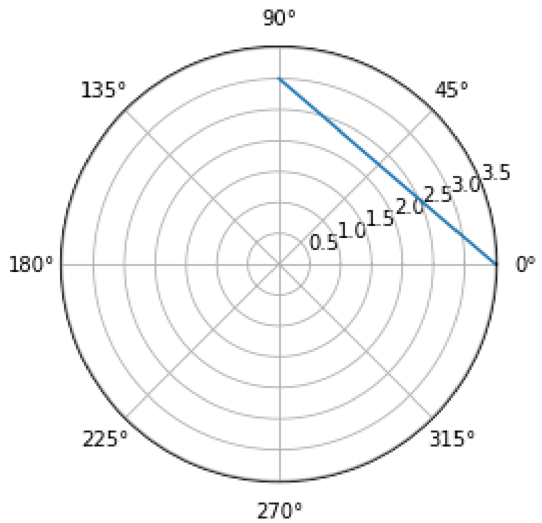


In [7]:

```
x = np.linspace(0, np.pi/2, 2)
y =(3.5, 3)
plt.polar(x, y)
```

Out[7]:

```
[<matplotlib.lines.Line2D at 0x83d3748>]
```



4. Задание (в программе)

1) Решите систему уравнений:

$$\begin{cases} y = x^2 - 1 \\ exp(x) + x \bullet (1 - y) = 1 \end{cases}$$

Преобразуем систему:

$$\begin{cases} y = x^2 - 1 \\ x \bullet (1 - y) = 1 - exp(x) \end{cases}$$

$$\begin{cases} y = x^2 - 1 \\ 1 - y = \frac{1 - exp(x)}{x} \end{cases}$$

$$\begin{cases} y = x^2 - 1 \\ y = 1 - \frac{1 - exp(x)}{x} \end{cases}$$

Построим графики функций и решим систему уравнений численным методом

In [8]:

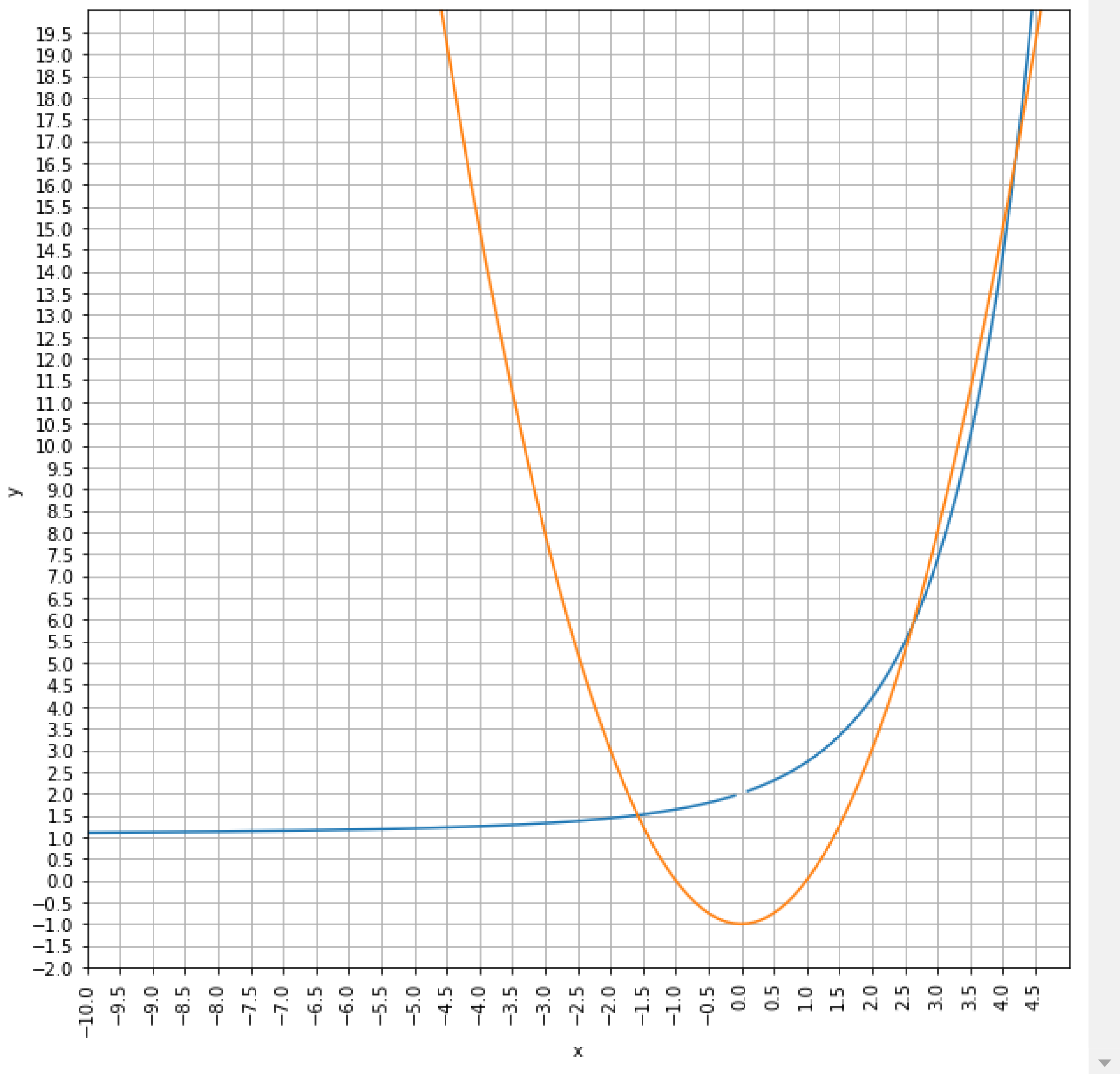
```
x = np.linspace(-10, 10, 201)
plt.subplots(figsize=(10, 10))
plt.plot(x, 1-(1-np.exp(x))/x)
plt.plot(x, x**2 - 1)
plt.xlabel('x')
plt.ylabel('y')
plt.xlim(-10,5)
plt.ylim(-2,20)
plt.xticks(np.arange(-10, 5, 0.5), rotation=90)
plt.yticks(np.arange(-2, 20, 0.5))
plt.grid(True)
plt.show()

from scipy.optimize import fsolve

def equations(p):
    x, y = p
    return (y - x**2 + 1, y - 1+(1-np.exp(x))/x)

x1, y1 =  fsolve(equations, (-2, 0))
x2, y2 =  fsolve(equations, (2, 5))
x3, y3 =  fsolve(equations, (4, 14))
print("Корни системы уравнений (x, y)")
print (f"{x1}, {y1}")
print (f"{x2}, {y2}")
print (f"{x3}, {y3}")
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:3: RuntimeWa
rning: invalid value encountered in true_divide
  This is separate from the ipykernel package so we can avoid doing imports u
ntil
```




```
Корни системы уравнений (x, y)
-1.5818353528958515, 1.5022030836713072
2.618145573085375, 5.854686241865328
4.200105841155167, 16.640889076905783
```

2) Решите систему уравнений и неравенств:

$$\begin{cases} y = x^2 - 1 \\ exp(x) + x \bullet (1 - y) > 1 \end{cases}$$

Преобразуем систему:

$$\begin{cases} y = x^2 - 1 \\ x \bullet (1 - y) > 1 - exp(x) \end{cases}$$

$$\begin{cases} y = x^2 - 1 \\ 1 - y > \frac{1 - exp(x)}{x} \end{cases}$$

$$\begin{cases} y = x^2 - 1 \\ y > 1 - \frac{1 - exp(x)}{x} \end{cases}$$

Построим графики функций и решим систему уравнений численным методом

In [9]:

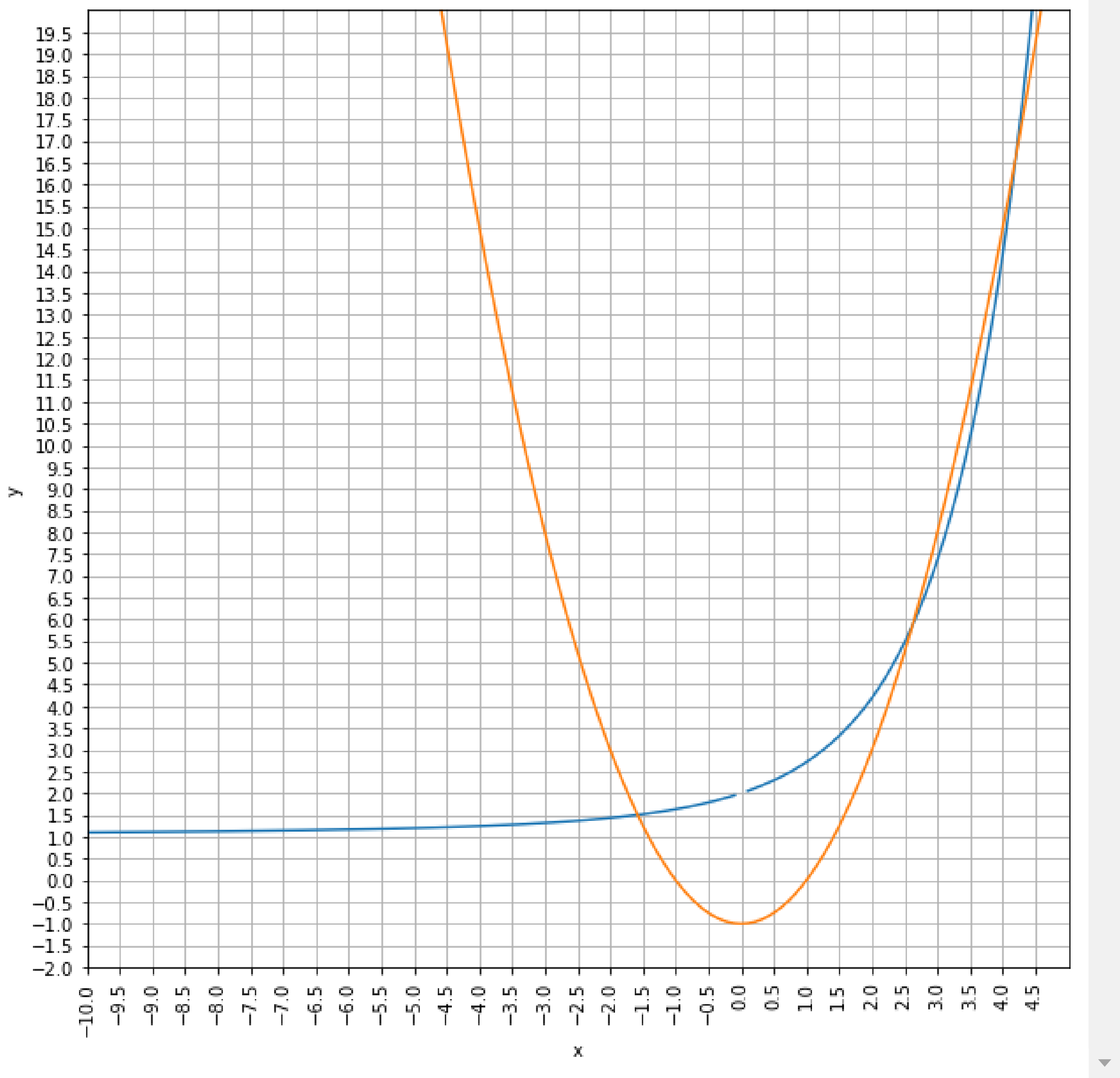
```
x = np.linspace(-10, 10, 201)
plt.subplots(figsize=(10, 10))
plt.plot(x, 1-(1-np.exp(x))/x)
plt.plot(x, x**2 - 1)
plt.xlabel('x')
plt.ylabel('y')
plt.xlim(-10,5)
plt.ylim(-2,20)
plt.xticks(np.arange(-10, 5, 0.5), rotation=90)
plt.yticks(np.arange(-2, 20, 0.5))
plt.grid(True)
plt.show()

from scipy.optimize import fsolve

def equations(p):
    x, y = p
    return (y - x**2 + 1, y - 1+(1-np.exp(x))/x)

x1, y1 =  fsolve(equations, (-2, 0))
x2, y2 =  fsolve(equations, (2, 5))
x3, y3 =  fsolve(equations, (4, 14))
print("Корни системы уравнений (x, y)")
print (f"{x1}, {y1}")
print (f"{x2}, {y2}")
print (f"{x3}, {y3}")
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:3: RuntimeWa
rning: invalid value encountered in true_divide
  This is separate from the ipykernel package so we can avoid doing imports u
ntil
```




```
Корни системы уравнений (x, y)
-1.5818353528958515, 1.5022030836713072
2.618145573085375, 5.854686241865328
4.200105841155167, 16.640889076905783
```

Решением неравенства будут все точки, лежащие выше синей линии, не включая саму синюю линию.
Решением уравнения будт все точки с координатами, вычисляемыми по уравнению:

$$y = x^2 - 1$$

На графике они соответвуют ораньжевой линии. Следовательно решением данной системы будут следующие множества x и y:

$$\begin{cases} x \subset ((-\infty; -1.5818353528958515) \cup (2.618145573085375; 4.200105841155167)) \\ y \subset ((1.5022030836713072; +\infty) \cup (5.854686241865328; 16.640889076905783)) \end{cases}$$

однозначно связанные меджу собой по вышеуказанному уравнению:

$$y = x^2 - 1$$