GeekBrains

# Задание к уроку 2

**1. Задание (3.1/1)**

Даны два вектора в трехмерном пространстве: (10,10,10) и (0,0,-10)

1) Найдите их сумму. (на листочке)

Решение

$u = (10, 10, 10);\ v = (0, 0, -10)$

$u + v = (10 + 0,\quad 10 + 0,\quad 10 - 10) = (10, 10, 0)$

- формулу нашел в сети, разве не нужно сначала модуль каждого вектора найти и затем модули сложить? – прошу дать комментарий

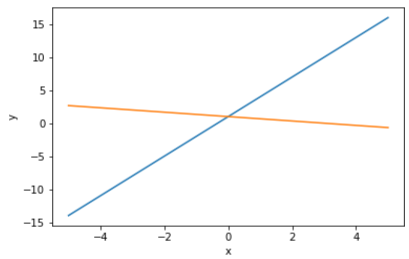
2) Напишите код на Python, реализующий расчет длины вектора, заданного его координатами. (в программе)

**2. Задание (на листочке) (3.1/2)**

Почему прямые не кажутся перпендикулярными? (см.ролик)

Ответ: потому что у шкалы x и y разная цена деления. Если привести шкалы к одинаковой цене деления тогда прямые будут выглядеть перпендикулярно

```
In [1]: %matplotlib inline
        import numpy as np
        import matplotlib.pyplot as plt
        import math

In [2]: x = np.linspace(-5, 5, 21)
        y = 3*x+1
        y2 = (-1/3)*x+1
        plt.plot(x,y)
        plt.plot(x,y2)
        plt.xlabel("x")
        plt.ylabel("y")
Out[2]: <matplotlib.text.Text at 0x6aa80f0>
```




```
In [ ]:
```

**3. Задание (в программе) (3.1/3)**

Напишите код на Python, реализующий построение графиков:

1. окружности,
2. эллипса,
3. гиперболы.

**4. Задание (на листочке) (3.1/4)**

1) Пусть задана плоскость:

$A \bullet x + B \bullet y + C \bullet z + D = 0$

Напишите уравнение плоскости, параллельной данной и проходящей через начало координат.

Ответ: $A \bullet x + B \bullet y + C \bullet z + D = 0$

2) Пусть задана плоскость: $A_1x + B_1y + C_1z + D_1 = 0$

и прямая:

$$\frac{x-x_1}{x_2-x_1} = \frac{y-y_1}{y_2-y_1} = \frac{z-z_1}{z_2-z_1}$$

Как узнать, принадлежит прямая плоскости или нет?

Ответ: Если 2 точки одной прямой принадлежат плоскости, то и вся прямая принадлежит этой плоскости. Подставив координаты $x_1,\ y_1, z_1$ и $x_2,\ y_2, z_2$ точек прямой в уравнение плоскости мы должны получить верное равенство:

$$A_1x_1 + B_1y_1 + C_1z_1 + D_1 = 0$$

$$A_1x_2 + B_1y_2 + C_1z_2 + D_1 = 0$$

**5. Задание (в программе) (3.1/5)**

1) Нарисуйте трехмерный график двух параллельных плоскостей.
2) Нарисуйте трехмерный график двух любых поверхностей второго порядка.

GeekBrains

# Задание к уроку 3

**0. Задание (3.2/0) (сделайте себе шпаргалку перед глазами, если не помните) - не присылать**

Чему равны синус, косинус, тангенс перечисленных углов? Запишите значения в таблицу:

| угол | sin | cos | tg |
|---|---|---|---|
| $30^\circ$ | $\frac{1}{2}$ | $\frac{\sqrt{3}}{2}$ | $\frac{1}{\sqrt{3}}$ |
| $45^\circ$ | $\frac{\sqrt{2}}{2}$ | $\frac{\sqrt{2}}{2}$ | 1 |
| $60^\circ$ | $\frac{\sqrt{3}}{2}$ | $\frac{1}{2}$ | $\sqrt{3}$ |
| $90^\circ$ | 1 | 0 | – |
| $180^\circ$ | 0 | -1 | 0 |

**1. Задание (в программе) (3.2/1)**
Нарисуйте график функции: y(x) = k·cos(x – a) + b для некоторых (2-3 различных) значений параметров k, a, b

**2. Задание (3.2/2)**
Докажите, что при ортогональном преобразовании сохраняется расстояние между точками.

Решение

Воозьмем 2 точки: $D_1'$ и $D_2'$, выпишем их координаты после преобразования и приведем их:

$x' = a_{11}x + a_{12}y + a_{13}$

$y' = a_{21}x + a_{22}y + a_{23}$

$|D_1'D_2'|^2 = [x_2' - x_1']^2 + [y_2' - y_1']^2 = [a_{11}(x_2 - x_1) + a_{12}(y_2 - y_1)]^2 + +[a_{21}(x_2 - x_1) + a_{22}(y_2 - y_1)]^2 =$

$= a_{11}^2 + a_{21}^2)\,(x_2 - x_1)^2 + (a_{12}^2 + a_{22}^2)(y_2 - y_1)^2 + 2(a_{11}a_{12} + a_{21}a_{22})(x_2 - x_1)(y_2 - y_1) =$

$= (x_2 - x_1)^2 + (y_2 - y_1)^2 = |D_1D_2|^2$

Так как в результате приведения мы получили коортдинаты $D_1$ и $D_2$, следовательно при ортогональном преобразовании расстояние между точками не меняется

**3. Задание (в программе) (3.2/3)**

1. Напишите код, который будет переводить полярные координаты в декартовы.
2. Напишите код, который будет рисовать график окружности в полярных координатах.
3. Напишите код, который будет рисовать график отрезка прямой линии в полярных координатах.

**4. Задание (в программе) (3.2/4)**

1) Решите систему уравнений:

$$y = x^2 - 1$$
$$\exp(x) + x \cdot (1 - y) = 1$$

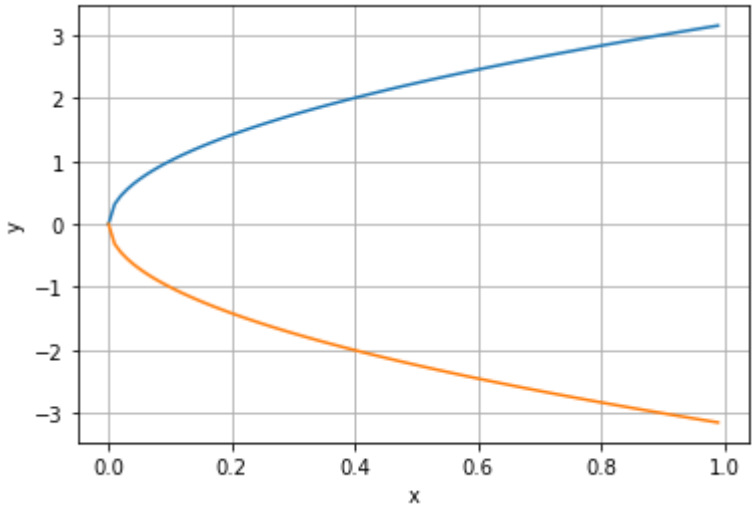
2) Решите систему уравнений и неравенств:

$$y = x^2 - 1$$
$$\exp(x) + x \cdot (1 - y) > 1$$

In [1]:
```python
#3.1/1-2. Напишите код на Python, реализующий расчет длины вектора, заданного его координатами.
import numpy as np
import math

a = np.array([10, 10, 10])
b = np.array([0, 0, -10])
c = a + b
print(c)

print(math.sqrt(c[0]**2 + c[1]**2 + c[2]**2))
```

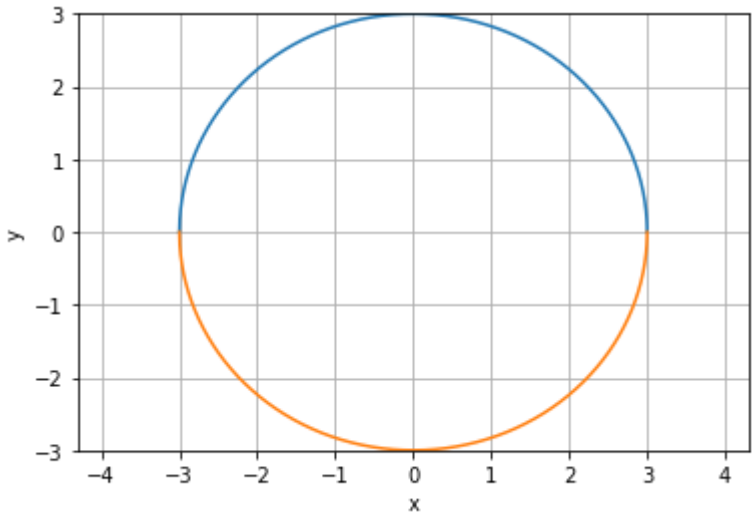
```
[10 10  0]
14.142135623730951
```

In [3]:
```python
#3.1/3. Напишите код на Python, реализующий построение графиков:
#         1.     окружности,
#         2.     эллипса,
#         3.     гиперболы.
# 3.1/3-0. Парабола
%matplotlib inline
import matplotlib.pyplot as plt
import math
x = []
y1 = []
y2 = []
p = 5
for i in range(100):
    x_ = i/100
    x.append(x_)
    y1.append(math.sqrt(2*p*x_))
    y2.append(-math.sqrt(2*p*x_))
plt.plot(x,y1)
plt.plot(x,y2)
plt.xlabel("x")
plt.ylabel("y")
plt.grid(True)
```



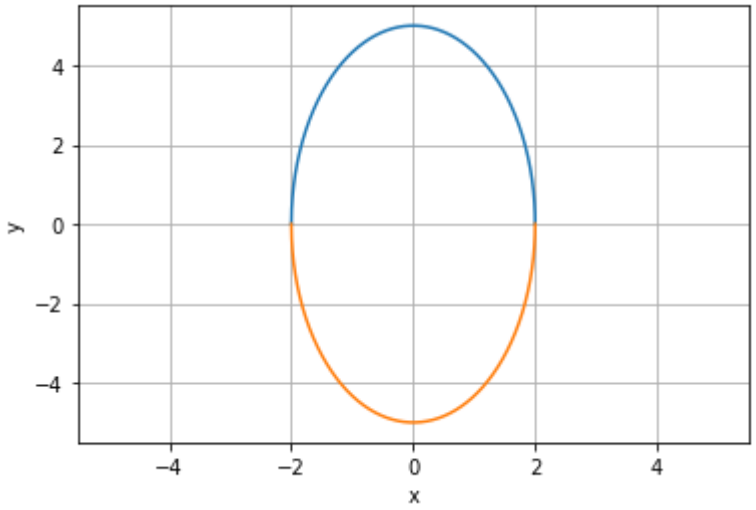

In [15]:

```
# 3.1/3-1. Окружность
x = []
y1 = []
y2 = []
R = 3

for i in range(1000*2*R+1):
    x_ = -R + i/1000
    x.append(x_)
    y1.append(math.sqrt(R**2 - x_**2))
    y2.append(-math.sqrt(R**2 - x_**2))
plt.plot(x,y1)
plt.plot(x,y2)
plt.xlabel("x")
plt.ylabel("y")
plt.ylim(-3,3)
plt.xlim(-4.3,4.3)
plt.grid(True)
```



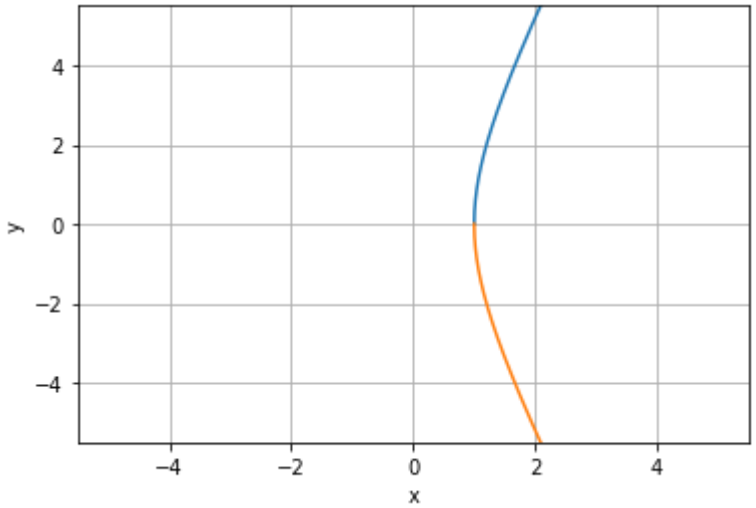

In [25]:

```
# 3.1/3-2. Эллипс
x = []
y1 = []
y2 = []
a = 2
b = 5

for i in range(1000*2*a+1):
    x_ = -a + i/1000
    x.append(x_)
    y1.append(math.sqrt(1 - x_**2/a**2) * b)
    y2.append(-math.sqrt(1 - x_**2/a**2) * b)
plt.plot(x,y1)
plt.plot(x,y2)
plt.xlabel("x")
plt.ylabel("y")
plt.ylim(-5.5,5.5)
plt.xlim(-5.5,5.5)
plt.grid(True)
```



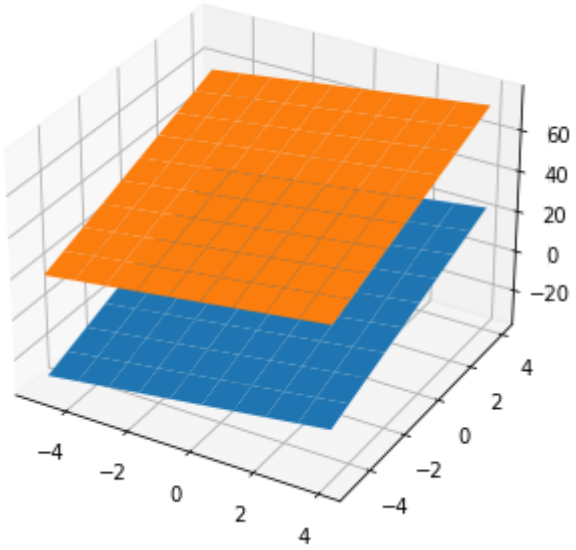

In [26]:

```
# 3.1/3-3. Гипербола
x = []
y1 = []
y2 = []
a = 1
b = 3

for i in range(1000*2*a):
    x_ = a + i/1000
    x.append(x_)
    y1.append(math.sqrt(x_**2/a**2 - 1) * b)
    y2.append(-math.sqrt(x_**2/a**2 - 1) * b)
plt.plot(x,y1)
plt.plot(x,y2)
plt.xlabel("x")
plt.ylabel("y")
plt.ylim(-5.5,5.5)
plt.xlim(-5.5,5.5)
plt.grid(True)
```



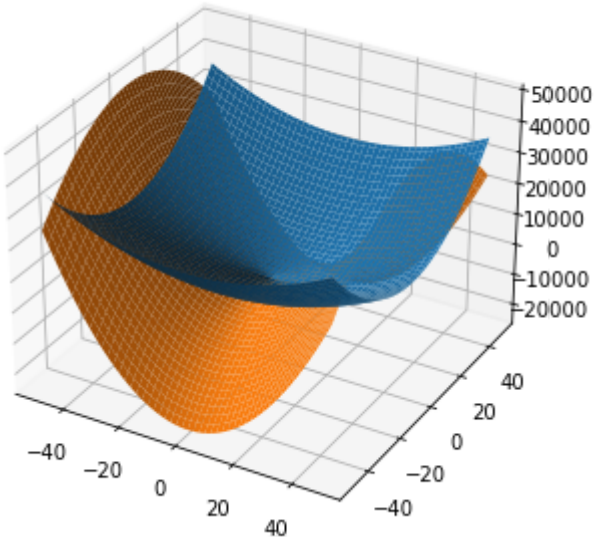

In [35]:

```
# 3.1/5-1. Нарисуйте трехмерный график двух параллельных плоскостей.
from pylab import *
from mpl_toolkits.mplot3d import Axes3D
fig = figure()
ax = Axes3D(fig)
X = np.arange(-5, 5, 1)
Y = np.arange(-5, 5, 1)
X, Y = np.meshgrid(X, Y)
Z = 2*X + 4*Y
Z2 = 2*X + 4*Y + 50
ax.plot_surface(X, Y, Z)
ax.plot_surface(X, Y, Z2)
ax.scatter(0, 0, 0, 'z', 50, 'red')
show()
```



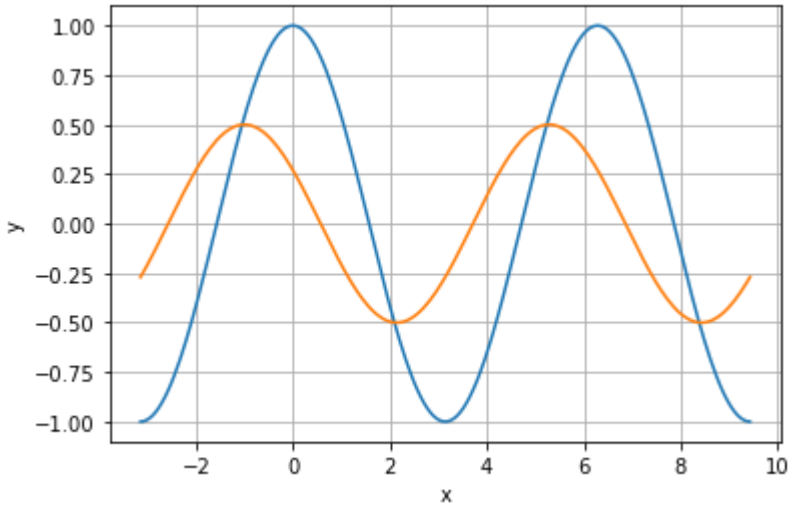

In [38]:

```
# 3.1/5-2. Нарисуйте трехмерный график двух любых поверхностей второго порядка.
from pylab import *
from mpl_toolkits.mplot3d import Axes3D
fig = figure()
ax = Axes3D(fig)
X = np.arange(-50, 50, 1)
Y = np.arange(-50, 50, 1)
X, Y = np.meshgrid(X, Y)
Z2 = 20*X**2 - 10*Y**2
Z1 = 5*X**2 + 10*Y**2
ax.plot_surface(X, Y, Z1)
ax.plot_surface(X, Y, Z2)
show()
```

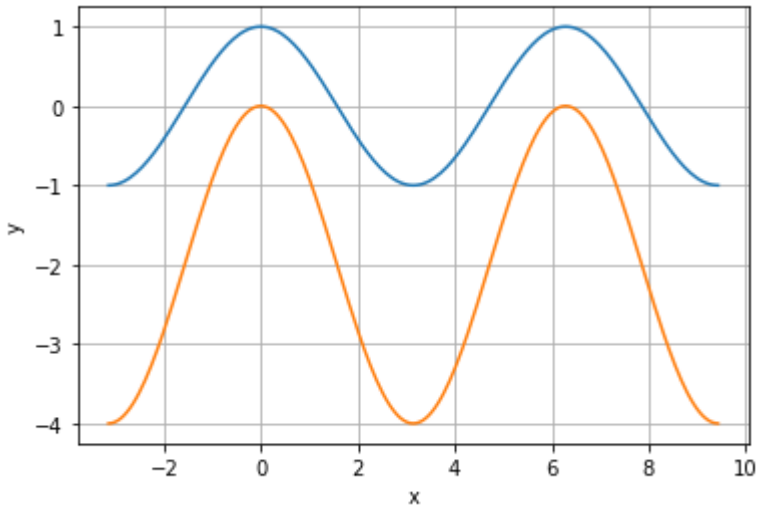



In [ ]:
```
#3.2/1. Нарисуйте график функции: y = k*cos(x-a)+b для некоторых (2-3 различных) значений параметров k, a, b
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
```

In [13]:
```
k = 0.5
a = -1
b = 0
x = np.linspace(-np.pi, 3*np.pi, 201)
plt.plot(x, np.cos(x))
plt.plot(x, k * np.cos(x - a) + b)
plt.xlabel('x')
plt.ylabel('y')
plt.axis('tight')
plt.grid(True)
plt.show()
```

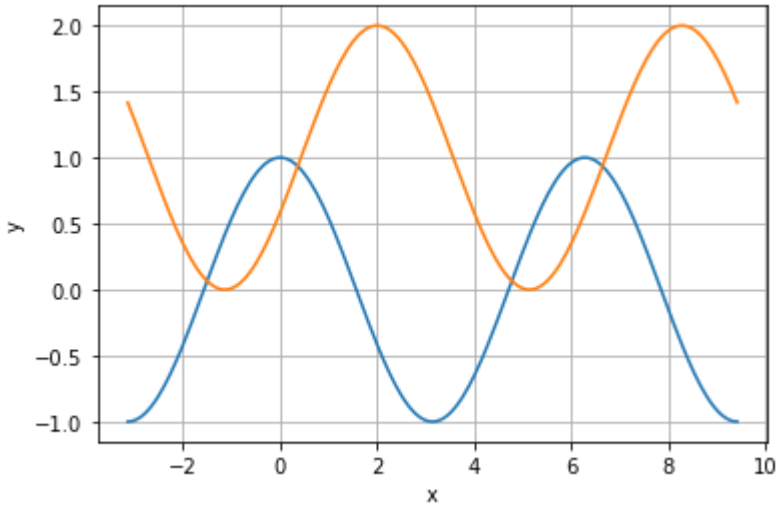



In [14]:
```
k = 2
a = 0
b = -2
x = np.linspace(-np.pi, 3*np.pi, 201)
plt.plot(x, np.cos(x))
plt.plot(x, k * np.cos(x - a) + b)
plt.xlabel('x')
plt.ylabel('y')
plt.axis('tight')
plt.grid(True)
plt.show()
```



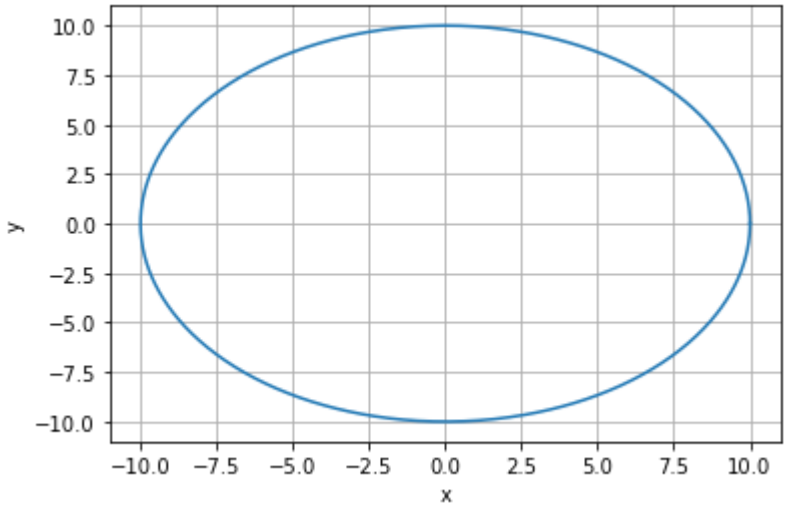

In [15]:
```
k = 1
a = 2
```

```
b = 1
x = np.linspace(-np.pi, 3*np.pi, 201)
plt.plot(x, np.cos(x))
plt.plot(x, k * np.cos(x - a) + b)
plt.xlabel('x')
plt.ylabel('y')
plt.axis('tight')
plt.grid(True)
plt.show()
```



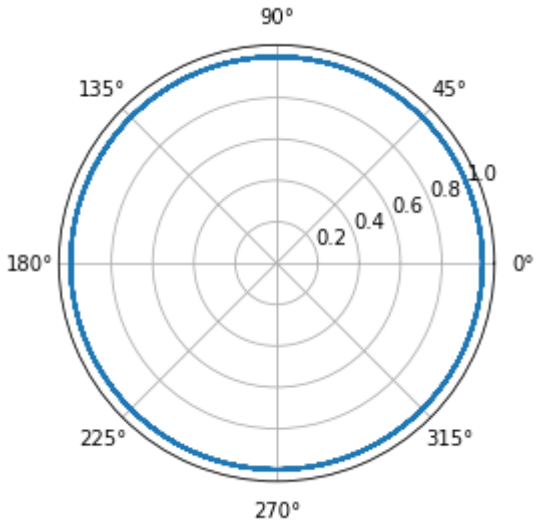

In [8]:

```
#3.2/3-1. Напишите код, который будет переводить полярные координаты в декартовы.
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt

phi = np.linspace(0, 2 * np.pi, 1000)
rho = 10
x = rho * np.cos(phi)
y = rho * np.sin(phi)

plt.plot(x, y)
plt.xlabel('x')
plt.ylabel('y')
plt.grid(True)
plt.show()
```




In [10]:
```
#3.2/3-2. Напишите код, который будет рисовать график окружности в полярных координатах
r = np.linspace(0, np.pi, 1000)
x = np.cos(r)
y = np.sin(r)
rho = np.sqrt(x**2 + y**2)
phi = np.arctan2(y,x) * 180 / np.pi
plt.polar(phi, rho)
```

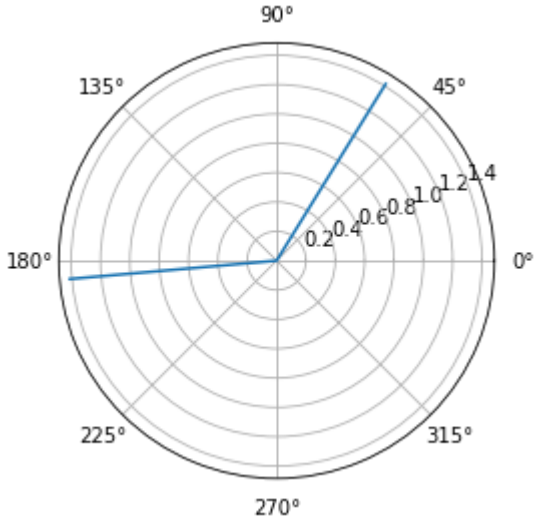
Out[10]:
```
[<matplotlib.lines.Line2D at 0x1946d084460>]
```




In [11]:
```
#3.2/3-3. Напишите код, который будет рисовать график отрезка прямой линии в полярных координатах
r = np.linspace(0, np.pi, 1000)
x = np.cos(r)
y = np.cos(r)
```

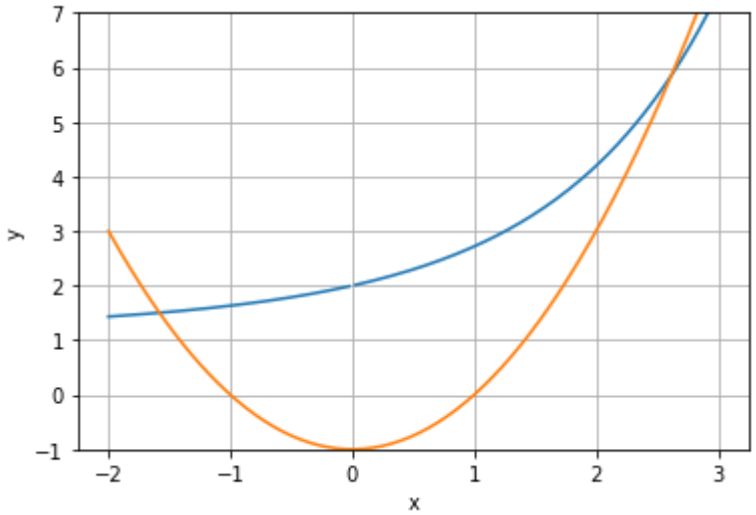
```
rho = np.sqrt(x**2 + y**2)
phi = np.arctan2(y,x) * 180 / np.pi
plt.polar(phi, rho)
```

Out[11]: [<matplotlib.lines.Line2D at 0x1946d0cc340>]




In [12]:
```
#3.2/4-1. Решите систему уравнений:
#          y=x^2-1
#          exp(x)+x(1-y)=1
x = np.linspace(-2, 3, 201)
plt.plot(x, (np.exp(x) - 1) / x + 1)
plt.plot(x, x**2 - 1)
plt.xlabel('x')
plt.ylabel('y')
plt.ylim(-1,7)
plt.grid(True)
plt.show()

from scipy.optimize import fsolve

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

x1, y1 = fsolve(equations, (-2, 1))
print (x1, y1)

x2, y2 = fsolve(equations, (2, 2))
print (x2, y2)
```

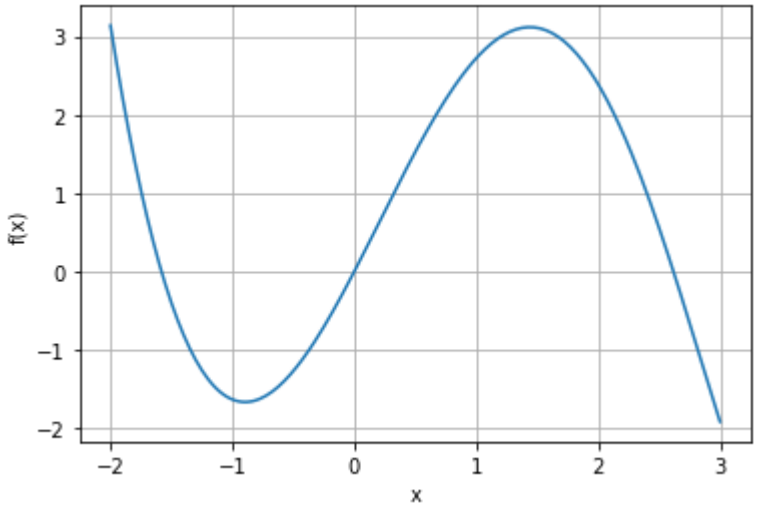
```
<ipython-input-12-47105a3c5aa7>:2: RuntimeWarning: invalid value encountered in true_divide
  plt.plot(x, (np.exp(x) - 1) / x + 1)
```




```
-1.581835352893692 1.502203083670816
2.618145573085011 5.854686241864717
```

In [13]:

```
#3.2/4-2. 2)     Решите систему уравнений и неравенств:
#          y=x^2-1
#          exp(x)+x(1-y)>1
x = np.linspace(-2, 3, 201)
plt.plot(x, np.exp(x) + x * (2 - x**2) - 1)
plt.xlabel('x')
plt.ylabel('f(x)')
plt.grid(True)
plt.show()

from scipy.optimize import fsolve

def equation(x):
    return (np.exp(x) + x * (2 - x**2) - 1)

x1 = fsolve(equation, -2)
print (x1)

x2 = fsolve(equation, 2)
print (x2)

x0 = fsolve(equation, .1)
print (x0)
```




```
[-1.58183535]
[2.61814557]
[2.55587182e-17]
```