Тема 2

1.

Даны два вектора в трехмерном пространстве: (10,10,10) и (0,0,-10) Найдите их сумму.

Ответ: $(10+0, 10+0, 10+(-10)) = \underline{(10, 10, 0)}$

1.1.

Напишите код на Python, ре

verb: realize, actualize, embody, encash, negotiate
translated from: Russian

координатами. (в программе)

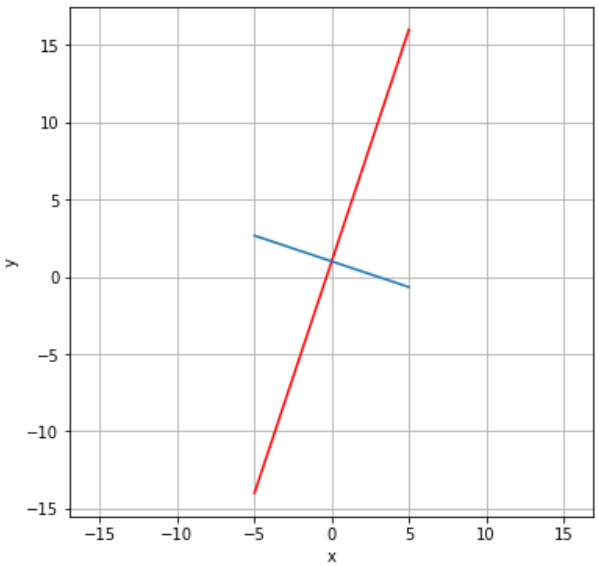
In [1]:
```
def calc_vector_length(coordinates: list):
    return (sum([i**2 for i in coordinates]))**0.5
```

2.

Почему прямые не кажутся перпендикулярными? (см.ролик)

Используются разные масштабы осей (цены деления по осям)

In [2]:
```
import matplotlib.pyplot as plt
import numpy as np
plt.figure(figsize=(6, 6))

x = np.linspace(-5, 5, 21)
y = 3*x+1
y2 = (-1/3)*(x)+1
plt.plot(x, y, 'r')
plt.plot(x, y2)
plt.xlabel('x')
plt.ylabel('y')
plt.grid()
plt.axis('equal') # сделать цены деления по осям одинаковыми
plt.show()
```



3.

Напишите код на Python, реализующий построение графиков: окружности, эллипса, гиперболы.

График окружности

In [3]:
```
x = np.linspace(-1.0, 1.0, 100)
y = np.linspace(-1.0, 1.0, 100)

X, Y = np.meshgrid(x,y)

F = X**2 + Y**2 - 1.0

fig, ax = plt.subplots()

ax.contour(X,Y,F,[0])

ax.set_aspect(1)

plt.title('Circle graph', fontsize=8)

plt.xlim(-1.25,1.25)
plt.ylim(-1.25,1.25)
plt.axis('equal')
plt.grid(linestyle='--')
plt.show()
```

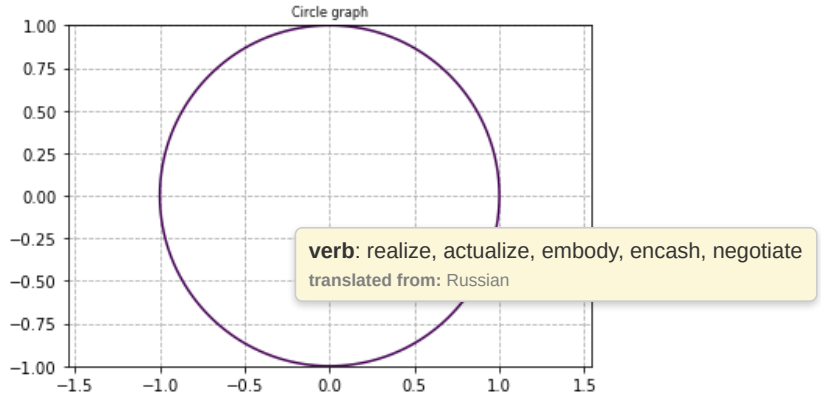


График эллипса

In [6]:
```
angles = np.linspace(0,360,360)

x =1*np.cos(np.radians(angles))

y = 3*np.sin(np.radians(angles))

plt.plot(x,y)
plt.grid(linestyle='--')
plt.title('Ellipse graph', fontsize=8)
plt.axis('equal')
plt.show()
```

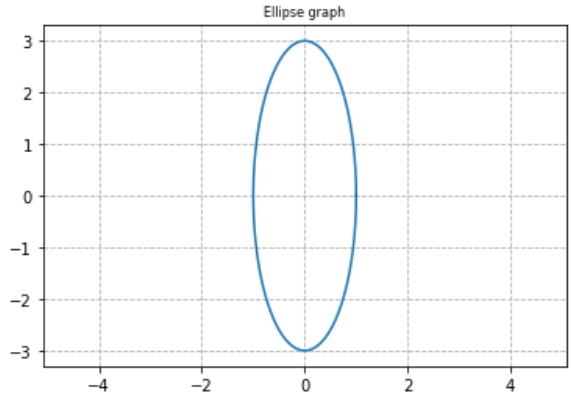
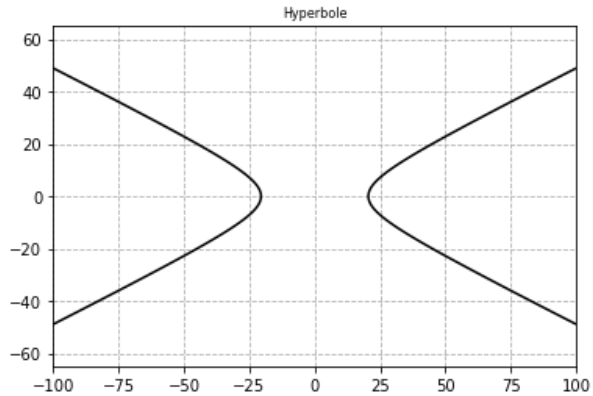


График гиперболы

In [7]:
```
x = np.linspace(-100, 100, 200)
y = np.linspace(-50, 50, 200)
x, y = np.meshgrid(x, y)
plt.contour(x, y,((x)**2 - (y*2)**2)-420, [1], colors='k')

plt.grid(linestyle='--')
plt.title('Hyperbole', fontsize=8)
plt.axis('equal')
plt.show()
```



4. 1) Пусть задана плоскость: $Ax + By + Cz + D = 0$

Напишите уравнение плоскости, параллельной данной и проходящей через начало координат.
$Ax + By + Cz = 0$

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?

Прямая параллельна или принадлежит плоскости тогда и только тогда, когда плоскости ортогональны: скалярное произведение вектора нормали плоскости и направляющего вектора прямой в таком случае будет равно нулю. Прямая принадлежит плоскости,

когда любая точка прямой удовлетворяет уравнению плоскости.
Вектор нормали: $\vec{n}(A_1; B_1; C_1)$
Направляющий вектор: $\vec{p}(x_2 - x_1; y_2 - y_1; z_2 - z_1)$

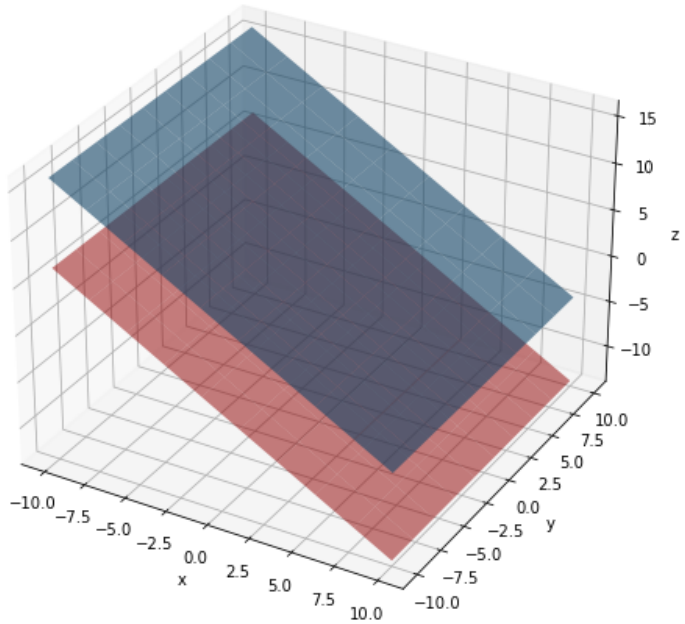
$$\begin{cases} A_1(x_2 - x_1) + B_1(y_2 - y_1), C_1(z_2 - z_1) + D_1 = 0, \\ A_1 x + B_1 y + C_1 z + D_1 = 0. \end{cases}$$

5.

а) *Нарисуйте трехмерный г*

verb: realize, actualize, embody, encash, negotiate
translated from: Russian

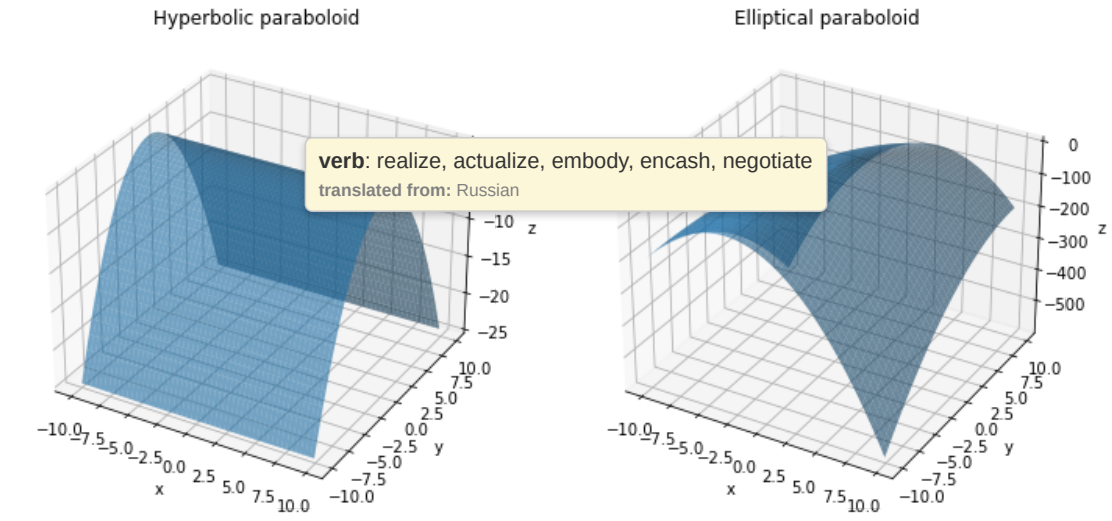
In [14]:
```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import ticker
a,b,c,d = 1,0,1,6

x = np.linspace(-10,10,10)
y = np.linspace(-10,10,10)

X,Y = np.meshgrid(x,y)
Z = (d - a*X - b*Y) / c

fig = plt.figure(figsize=(8, 8))

ax = fig.gca(projection='3d')
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_zlabel('z')
plt.grid(linestyle='--')
ax.plot_surface(X, Y, Z, alpha=.6);
a,b,c,d = 3, 0, 3, -10
Z1 = (d - a*X - b*Y) / c
ax.plot_surface(X, Y, Z1, rstride=1, cstride=1, alpha=.5, linewidth=0.5, color='r');
```



б) *Нарисуйте трехмерный график двух любых поверхностей второго порядка.*

In [50]:
```
# гиперболический параболоид
a,b,c= 40, 2, 2
X = np.arange(-10, 10, .1)
Y = np.arange(-10, 10, .1)
X, Y = np.meshgrid(X, Y)
Z = (X**2/a**2) - (Y**2/b**2)

fig = plt.figure(figsize=(12, 12))

ax = fig.add_subplot(1, 2, 1, projection='3d')
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_zlabel('z')
ax.set_title('Hyperbolic paraboloid')
plt.grid(linestyle='--')

# эллиптический параболоид
ax.plot_surface(X, Y, Z, alpha=.6);
Z = 1 - 3*X**2 + 2*X*Y - Y**2
ax1 = fig.add_subplot(1, 2, 2, projection='3d')
ax1.plot_surface(X, Y, Z, alpha=.6);
ax1.set_xlabel('x')
ax1.set_ylabel('y')
```

```
ax1.set_zlabel('z')
ax1.set_title('Elliptical paraboloid')
plt.show()
```

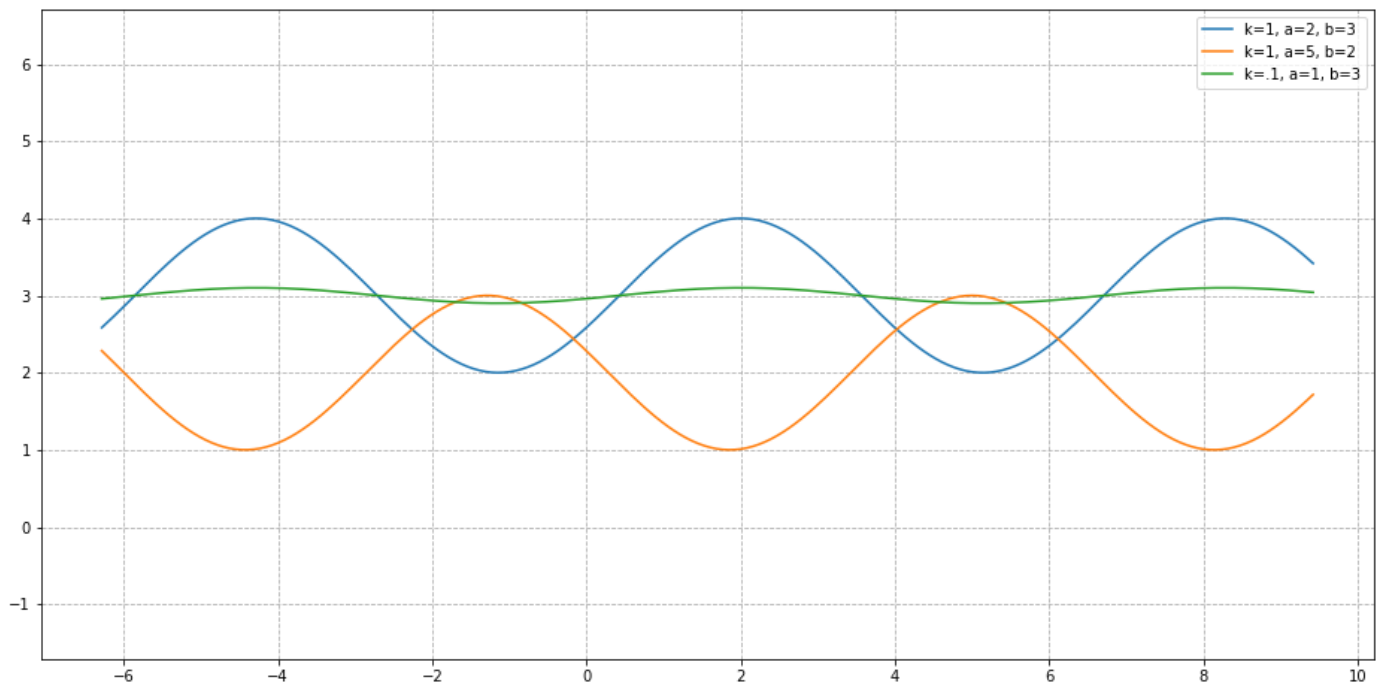


Тема 3

1.

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

In [8]:
```
plt.figure(figsize=(16, 8))
x = np.linspace(-2*np.pi, 3*np.pi, 200)
k=1; a=2; b=3
y = k * np.cos(x-a)+b
plt.plot(x, y, label='k=1, a=2, b=3')

k=1; a=5; b=2
y = k * np.cos(x-a)+b
plt.plot(x, y, label='k=1, a=5, b=2')

k=0.1; a=2; b=3
y = k * np.cos(x-a)+b
plt.plot(x, y, label='k=.1, a=1, b=3')
plt.legend()
plt.axis('equal')
plt.grid(linestyle='--')
plt.show()
```



2.

Докажите, что при ортогональном преобразовании сохраняется расстояние между точками.

Линейное преобразование на плоскости:

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

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

Условия ортогональности преобразования:

- $a_{11}^2 + a_{21}^2 = 1$
- $a_{12}^2 + a_{22}^2 = 1$
- $a_{11}a_{12} + a_{21}a_{22} = 0$
- $a_{11}a_{22} - a_{12}a_{21} \neq 0$

Пусть точки $A(x_1, y_1)$ и $B(x_2, y_2)$ переводятся соответственно в точки $A'(x_1', y_1')$ и $B'(x_2', y_2')$. Доказать, что AB == A'B'.

$$|A'B'|^2 = (x_2' - x_1')^2 + (y_2'$$

verb: realize, actualize, embody, encash, negotiate
translated from: Russian

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

чтд

3.

а) Напишите код, который будет переводить полярные координаты в декартовы.

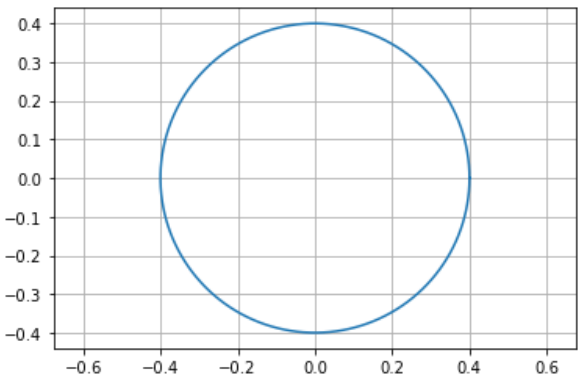
$x = R\cos\alpha \qquad y = R\sin\alpha$

In [110…
```
def to_restangle(polar: list):
    x = polar[0]*np.cos(polar[1]*np.pi/180)
    y = polar[0]*np.sin(polar[1]*np.pi/180)
    return x, y
to_restangle([1.41, 45])
```

Out[110…
```
(0.997020561473032, 0.9970205614730319)
```

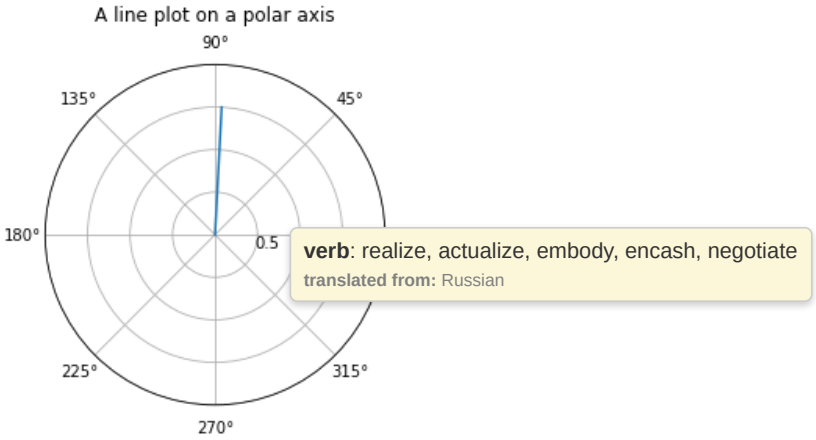
б) Напишите код, который будет рисовать график окружности в полярных координатах.

In [9]:
```
import numpy as np
import matplotlib.pyplot as plt

angle = np.linspace(0, 2 * np.pi, 150)

radius = 0.4

x = radius * np.cos(angle)
y = radius * np.sin(angle)

figure, axes = plt.subplots(1)

axes.plot(x, y)
axes.set_aspect(1)
plt.grid()
plt.axis('equal')
plt.show()
```



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

In [149…
```
r = np.arange(0, 2, 1.5)
theta = r/np.sin(45*math.pi/100)

fig, ax = plt.subplots(subplot_kw={'projection': 'polar'})
ax.plot(theta, r)
ax.set_rmax(2)
ax.set_rticks([0.5, 1, 1.5, 2])
ax.set_rlabel_position(-22.5)
ax.grid(True)

ax.set_title("A line plot on a polar axis", va='bottom')
plt.show()
```



verb: realize, actualize, embody, encash, negotiate
translated from: Russian

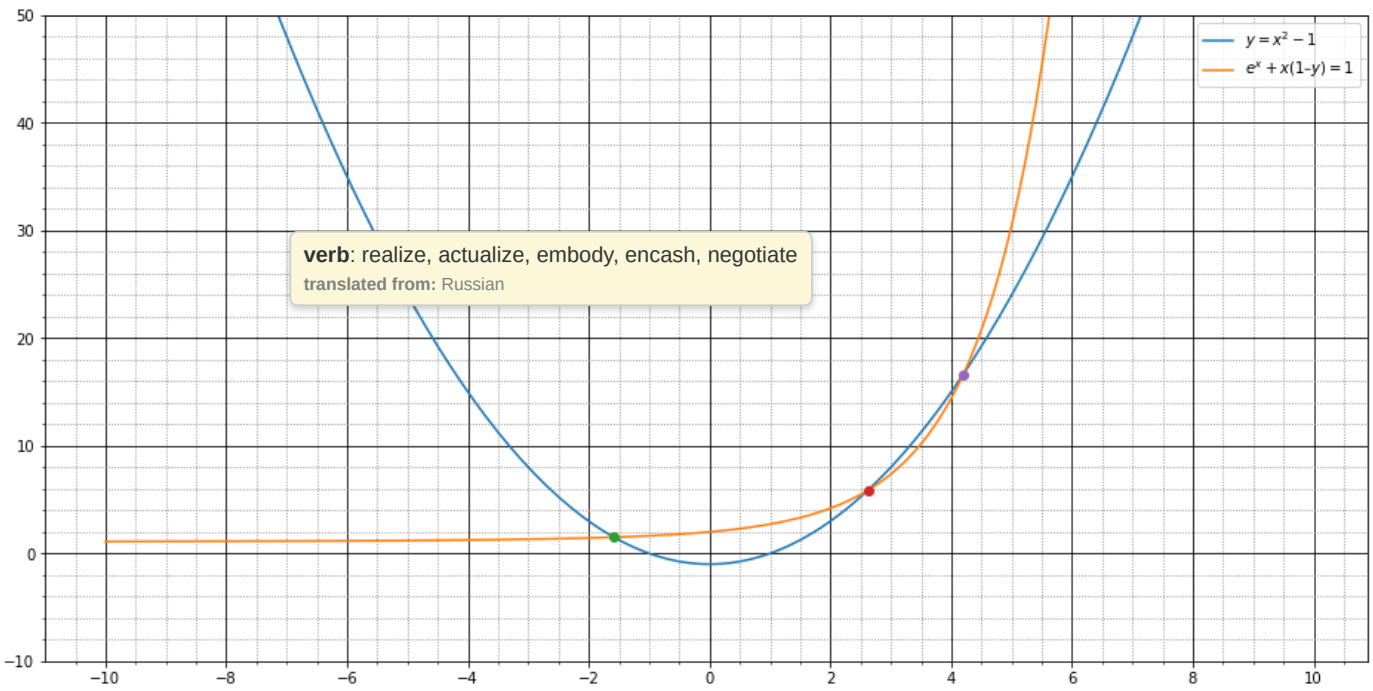
4.

Решите систему уравнений:

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

In [47]:

```
from scipy.optimize import fsolve

x = np.arange(-10, 10, 0.1)
y = x**2 - 1

fig, ax = plt.subplots(figsize=(16, 8))

ax.plot(x, y, label='$y=x^2-1$')
y = (np.e**x + x - 1) / x
ax.plot(x, y, label='$e^x+x(1-y)=1$')

plt.ylim(-10, 50)
ax.xaxis.set_major_locator(ticker.MultipleLocator(2))
ax.xaxis.set_minor_locator(ticker.MultipleLocator(1))
ax.yaxis.set_major_locator(ticker.MultipleLocator(10))
ax.yaxis.set_minor_locator(ticker.MultipleLocator(2))

ax.grid(which='major',
        color = 'k')
ax.minorticks_on()
ax.grid(which='minor',
        color = 'gray',
        linestyle = ':')

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

x_1, y_1 = fsolve(equations, (-2, 5))
x_2, y_2 = fsolve(equations, (3, 6))
x_3, y_3 = fsolve(equations, (5, 20))

ax.plot(x_1, y_1, 'o')
ax.plot(x_2, y_2, 'o')
ax.plot(x_3, y_3, 'o')

plt.legend();
```



Решите систему уравнений и неравенств:

$y = x^2 - 1$

$e^x + x \cdot (1 - y) - 1 > 0$

In [63]:

```
x = np.linspace(-4, 6, 201)
fig, ax = plt.subplots(figsize=(16, 8))

y = np.exp(x)+x*(2-x**2)-1
ax.plot(x, y, label='$e^x+x(2-x^2)-1$')

plt.ylim(-10, 50)
ax.xaxis.set_major_locator(ticker.MultipleLocator(2))
ax.xaxis.set_minor_locator(ticker.MultipleLocator(0.1))
ax.yaxis.set_major_locator(ticker.MultipleLocator(20))
ax.yaxis.set_minor_locator(ticker.MultipleLocator(2))

ax.grid(which='major',
        color = 'k')
ax.minorticks_on()
ax.grid(which='minor',
        color = 'gray',
        linestyle = ':')

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