К вебинару 1

Элементарная алгебра

темы 1 и 3

Переменные и формулы.

Всякое **равенство** или неравенство, выражающее посредством букв и знаков действий какое-нибудь соотношение между числами, называется **формулой**. Если несколько чисел, обозначенных буквами (и/или цифрами), соединены между собой посредством знаков, указывающих, какие действия и в каком порядке надо произвести над числами, то такое обозначение называется **алгебраическим выражением**. Вычислить значение какого-нибудь выражения для данных численных значений букв — значит подставить в него на место букв эти численные значения и произвести все указанные в выражении действия, чтобы получить в результате число — численную величину алгебраического выражения.

Те величины, которые сохраняют неизменным своё значение, называются **постоянными** (или **константами**, что часто обозначают "const"). Величины, могущие принимать различные значения, называются **переменными**.

Числа

Действительными (или, по-другому, **вещественными**) числами называют совокупность чисел

- **рациональных** (которые можно представить в виде дроби)
- **иррациональных** (выражающих отношение несопоставимых отрезков, таких, например, как диагональ квадрата и его сторона).

In [1]:
```
x = 5
y = 234.678
print(x + 1)
print(x + y)
```

```
6
239.678
```

In [2]:
```
z = input()
print(float(z) + 5)
```

```
1
6.0
```

In [3]:
```
a = 30
b = 8
print(1 / a)
print(a / b)
b**(-1/3)
```

```
0.03333333333333333
3.75
```

Out[3]: 0.5

In [4]:
```
import numpy as np
np.pi
```

Out[4]: 3.141592653589793

Если ввести в рассмотрение числа, получающиеся в результате математических действий над действительными числами и особым числом i - мнимой единицей, по определению удовлетворяющей выражению $i^2 = -1$, то говорят о множестве **комплексных** чисел.

In [5]:
```
print(1j**2)
(-4)**(0.5)
```

```
(-1+0j)
```

Out[5]: (1.2246467991473532e-16+2j)

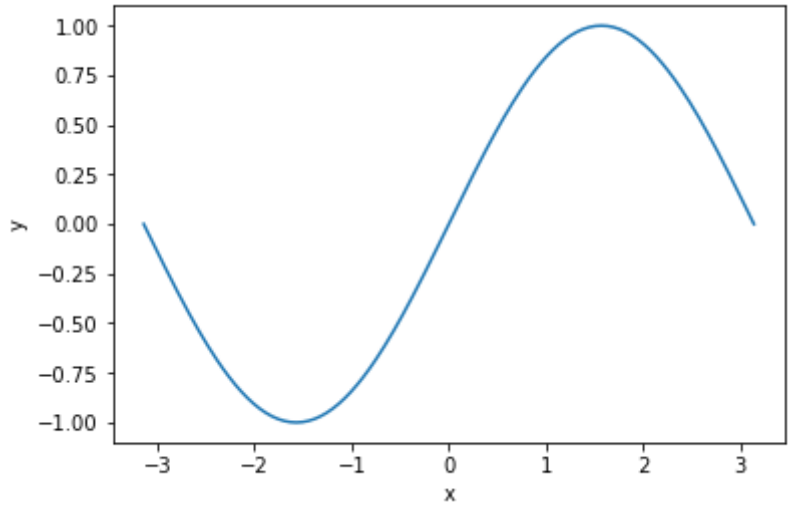
Функции и графики

Величина y называется **функцией** переменной величины x, если каждому из тех значений, которые может принимать x (по-другому, области определения x) соответствует одно значение y, что принято обозначать y(x) или y=f(x).

Функцию y(x) обычно визуализируют в виде **графиков**:

In [6]:
```
%matplotlib inline
import matplotlib.pyplot as plt
```

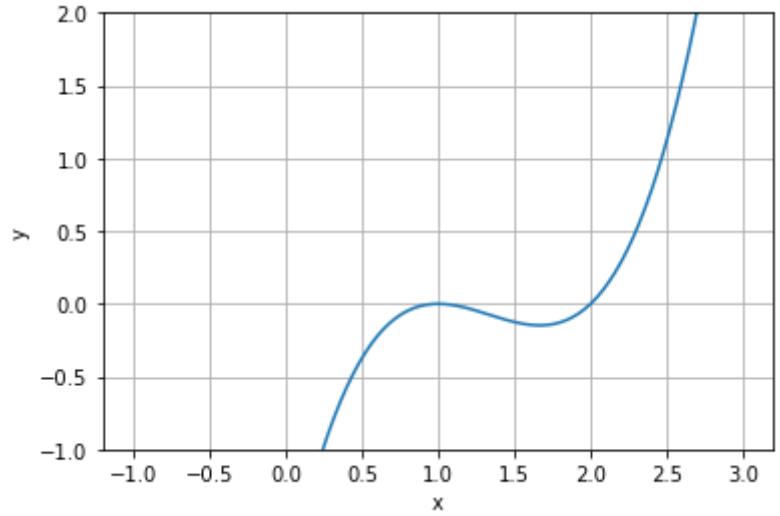
In [7]:
```
x = np.linspace(-np.pi, np.pi, 201)
plt.plot(x, np.sin(x))
plt.xlabel('x')
plt.ylabel('y')
plt.axis('tight')
plt.show()
```



Полиномы

In [8]:
```
x = np.linspace(-1, 3, 201)
print(np.poly([1, 1, 2]))
plt.plot(x, x**3 - 4 * x**2 + 5*x - 2)
plt.xlabel('x')
plt.ylabel('y')
plt.ylim(-1,2)
plt.grid(True)
plt.show()
print(np.roots([ 1., -4.,  5., -2.]))
```

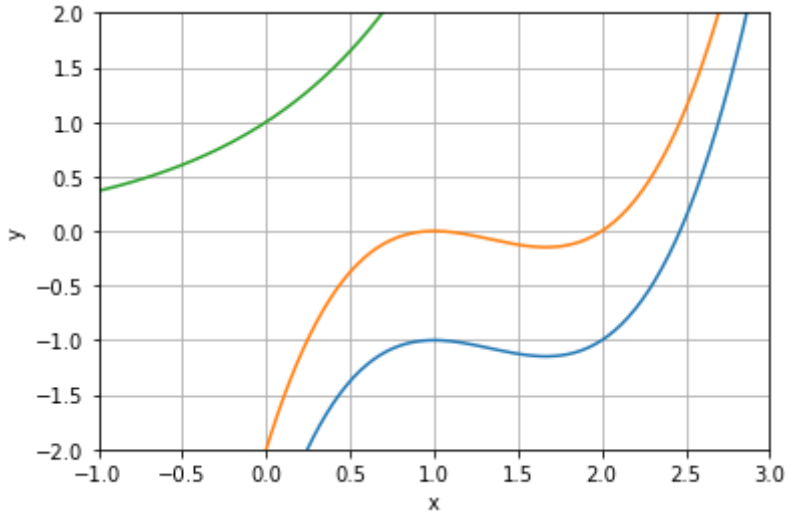
```
[ 1. -4.  5. -2.]
```




```
[ 2. +0.00000000e+00j  1. +2.83263462e-08j  1. -2.83263462e-08j]
```

In [9]:
```
x = np.linspace(-1, 3, 201)
print(np.poly([1, 1, 2]))
plt.plot(x, x**3 - 4*x**2 + 5*x - 3)
plt.plot(x,np.polyval([1., -4.,  5., -2.],x))
plt.plot(x, np.exp(x))
plt.xlabel('x')
plt.ylabel('y')
plt.xlim(-1,3)
plt.ylim(-2,2)
plt.grid(True)
plt.show()
print(np.roots([ 1., -4.,  5., -2.]))
```

```
[ 1. -4.  5. -2.]
```

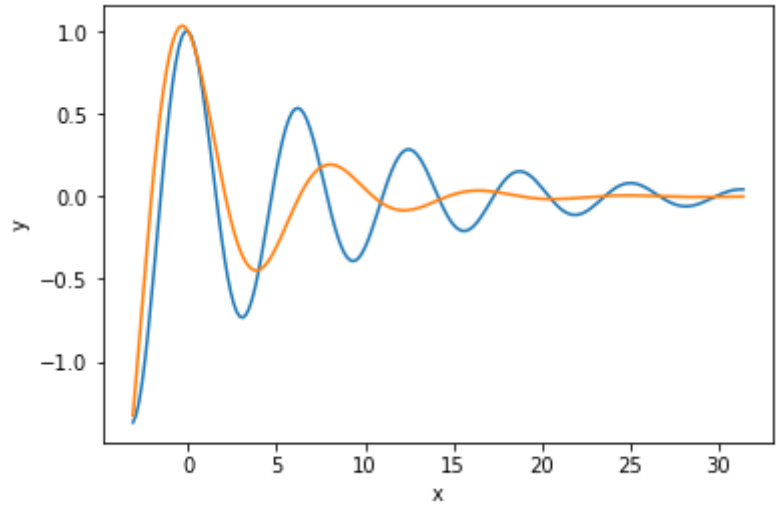



```
[ 2. +0.00000000e+00j  1. +2.83263462e-08j  1. -2.83263462e-08j]
```

Функции нескольких переменных

Пример: $f(x, k, z, A) = A \cdot exp(-z \cdot x) \cdot cos(k \cdot x)$

In [10]:
```
x = np.linspace(-np.pi, 10*np.pi, 201)
k1 = 1
k2 = .75
z1 = .1
z2 = .2
plt.plot(x, np.exp(-z1 * x) * np.cos(k1 * x))
plt.plot(x, np.exp(-z2 * x) * np.cos(k2 * x))
plt.xlabel('x')
plt.ylabel('y')
plt.axis('tight')
plt.show()
```

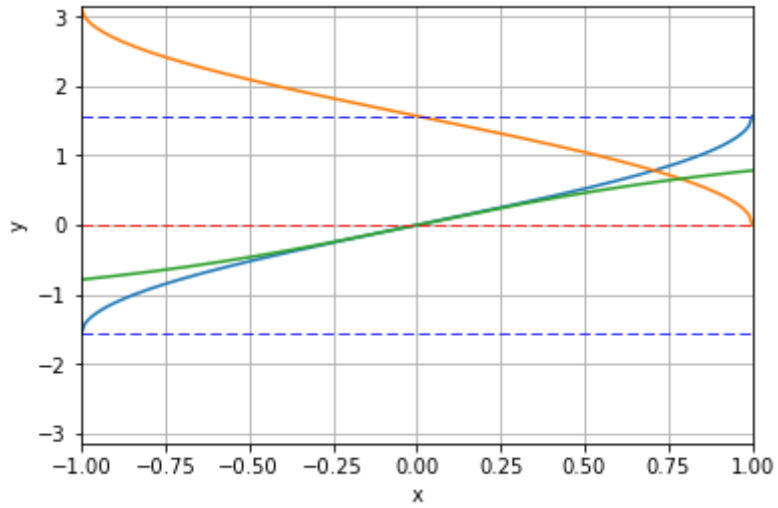


In [17]:
```
def f(x, k, z, A):
    return A * (np.exp(-z * x) * np.cos(k * x))
f(0, 1, 2, 3)
```

Out[17]: 3.0

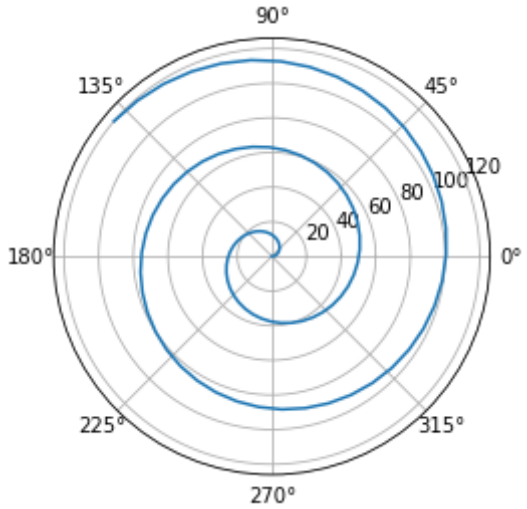
Пример форматирования графиков

In [11]:
```
x = np.linspace(-1, 1, 201)
plt.plot(x, np.arcsin(x))
plt.plot(x, np.arccos(x))
plt.plot(x, np.arctan(x))
plt.xlabel('x')
plt.ylabel('y')
plt.xlim(-1,1)
plt.ylim(-np.pi,np.pi)
plt.plot([-1,1],[-np.pi/2,-np.pi/2], color ='blue', linewidth=0.75, linestyle ="--")
plt.plot([-1,1],[0,0], color ='red', linewidth=0.75, linestyle ="--")
plt.plot([-1,1],[np.pi/2,np.pi/2], color ='blue', linewidth=0.75, linestyle ="--")
plt.grid(True)
plt.show()
```



In [12]:
```
x = np.linspace(0, 15, 100)
y = 8*x
plt.polar(x, y)
```

Out[12]: [<matplotlib.lines.Line2D at 0x8065030>]



Уравнения

Предположим, мы желаем теперь решить такую задачу. Стакан кофе стоит 59 рублей, причем известно, что сам кофе на 50 руб. дороже стакана. Сколько стоит стакан?

Обыкновенным (арифметическим) путём такую задачу решить трудно, многие ошибаются. Поэтому решим её, применив буквенное обозначение. Обозначим искомую стоимость стакана буквой x. Тогда условие задачи мы можем записать в виде равенства: (50 + x) + x = 59.

Такое равенство называют **уравнением** относительно переменной x.

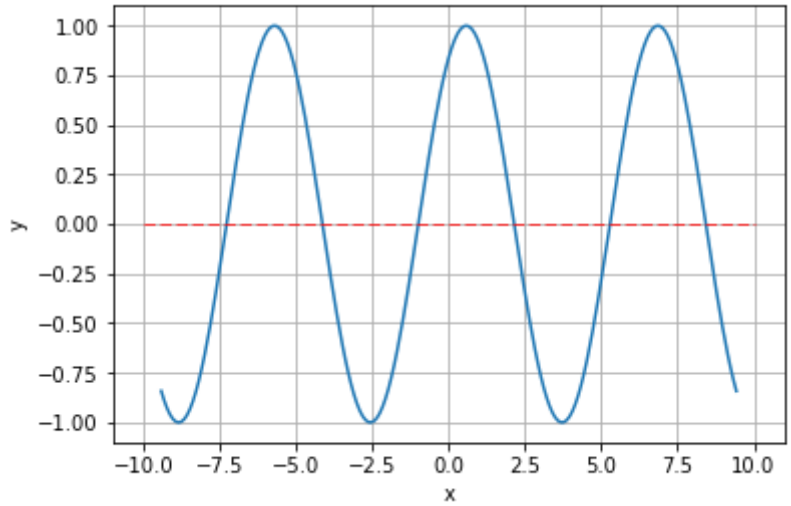
Важно, что в уравнении обе части равенства, содержащие одну или несколько букв (переменных), имеют одинаковую численную величину не при всяких численных значениях этих переменных. Эти переменные называются **неизвестными** уравнения.

Решить уравнение — значит найти те значения входящих в него неизвестных, которые удовлетворяют уравнению, т.е. обращают его в **тождество**. Эти значения неизвестных называются **корнями** уравнения.

Уравнение с одним неизвестным может иметь один корень, два корня и более.

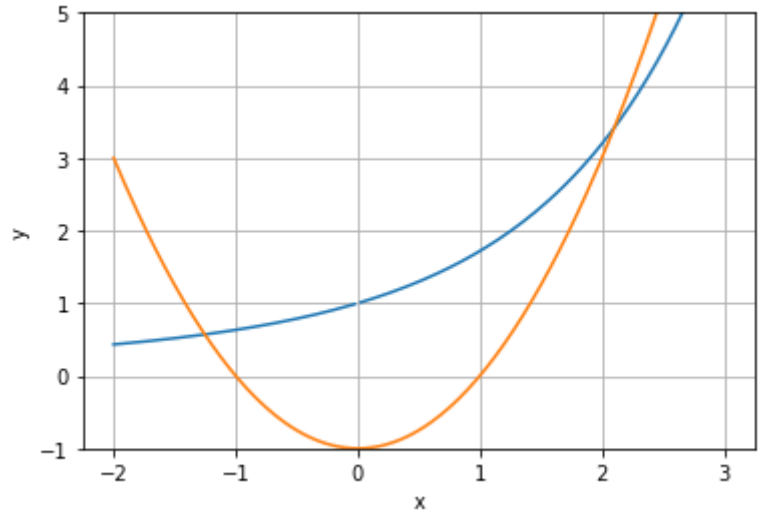
In [13]:
```
from scipy.optimize import fsolve

def f(x):
    return (np.sin(x + 1))

x0 =  fsolve(f, 1)
print (x0)

x = np.linspace(-3*np.pi, 3*np.pi, 201)
plt.plot(x, f(x))
plt.plot([-10, 10],[0,0], color ='red', linewidth=0.75, linestyle ="--")
plt.xlabel('x')
plt.ylabel('y')
plt.grid(True)
plt.show()
```

```
[ 2.14159265]
```



In [14]:
```
x = np.linspace(-2, 3, 201)
plt.plot(x, (np.exp(x) - 1)/x)
plt.plot(x, x**2 - 1)
plt.xlabel('x')
plt.ylabel('y')
plt.ylim(-1,5)
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*y - 1)

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

print (x1, y1)
```

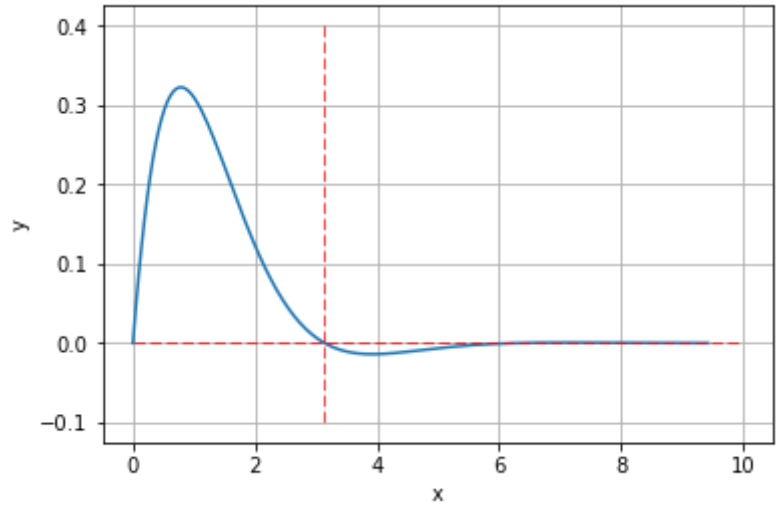
```
C:\Users\edito\Anaconda3\lib\site-packages\ipykernel\__main__.py:2: RuntimeWarning: invalid value encountered in true_d
ivide
  from ipykernel import kernelapp as app
```




```
-1.25303858997 0.570105707953
```

О задачах мат.анализа

In [46]:
```
def f(x):
    return np.exp(-x) * np.sin(x)
f(np.pi/2)
x = np.linspace(0, 3*np.pi, 201)
plt.plot(x, f(x))
plt.plot([0, 10],[0,0], color ='red', linewidth=0.75, linestyle ="--")
plt.plot([np.pi, np.pi],[.4,-0.1], color ='red', linewidth=0.75, linestyle ="--")
plt.xlabel('x')
plt.ylabel('y')
plt.grid(True)
plt.show()
```



In [47]:
```
from scipy.integrate import quad
quad(f, 0, np.pi)
```

Out[47]: (0.5216069591318861, 5.791000558329331e-15)

In [48]:
```
quad(f, 0, 3 * np.pi)
```

Out[48]: (0.5000403497587852, 1.5918519702031086e-10)

In [49]:
```
from scipy.misc import derivative
derivative(f, 1.0, dx=1e-6)
```

Out[49]:
```
-0.11079376527334261
```

In [50]:
```
derivative(f, 0., dx=1e-6)
```

Out[50]:
```
1.0000000000003333
```

In [ ]:
```

```