Решение задач на Python, Математический анализ, Комплексные числа

Действия над комплексными числами: сложение, вычитание, умножение, деление, производится с помощью стандартных операций «+», «-», «*», «/».

Пример 1:

Пусть $x=1+3i, y=2-i, g=1-2i, t=10$. Найти $z=x*y, h=\frac{t}{g}, n=p^2=p*p, C=z+h+n$

```
x=complex(1,3)
y=complex(2,-1)
z=x*y
print(z)
g=complex(1,-2)
print(g)
t=complex(10,0)
print(t)
h=t/g
print(h)
p=complex(-1,-1)
n=p*p
print(n)
C=z+h+n
print(C)
```

```
(5+5j)
(1-2j)
(10+0j)
(2+4j)
2j
(7+11j)
```

Возведение в степень: pow (число, показатель степени в которую мы возводим число).

Пример 2:

Степень i^2. $x=i, y=x^2, y=-1$.

```
x=complex(0,1)
y=pow(x,2) # Степень
print(y)
```

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

Итак, мы можем с легкостью производить любые действия с комплексными числами в среде Python.

Пример 3:

Вычислить $(1+3i)*(2-i)+\frac{10}{(1+2i)}+(-1-i)^2=7+11j$.

```
x=complex(1,3)
y=complex(2,-1)
z=x*y
print(z)
g=complex(1,-2)
print(g)
t=complex(10,0)
print(t)
h=t/g
print(h)
p=complex(-1,-1)
n=p*p
print(n)
C=z+h+n
print(C)
```

```
(5+5j)
(1-2j)
(10+0j)
(2+4j)
2j
(7+11j)
```

С понятием комплексного числа связано решение квадратных уравнений, дискриминант которых меньше нуля.

Пример 4:

Решить уравнение $ x2 — 2x + 5 = 0.j$. Решение. Чтобы решить уравнение $f(x)=0$ используем функцию $solve(f(x))$.

```
import math
from sympy import *
x = Symbol("x")
print(solve(x**2-2*x+5))
```

```
[1 - 2*I, 1 + 2*I]
```

Пример 5:

Найти значение функции $f(x)=x^4+\frac{2+i}{x}-(-3+2i)$, при $x=1-2i$.

```
x=complex(1,-2)
i=complex(0,1)
```

```
f=x**4+(2+i)/x-(-3+2*i)
print(f)
```

```
(-4+23j)
```

Пример 6:

Выполнить указанные действия $\frac{(1+i)^8}{(1-i)^6}$.

```
print((1+i)**8/(1-i)**6)
```

```
-2j
```

Пример 7:

Решить систему уравнений $\left(\begin{cases} (2+i)x+(2-i)y=6, \\ (3+2i)x+(3-2i)y=8. \end{cases}\right)$.

```
from sympy import Symbol, nsolve
import sympy
import mpmath
mpmath.mp.dps = 3

x = Symbol('x')
y = Symbol('y')

i = complex(0, 1)

f1 = (2+i)*x+y*(2-i)-6
f2 = (3+2*i)*x+(3-2*i)*y-8
print(nsolve((f1, f2), (x, y), (-1, 1)))
```

```
Matrix([[2.0 + 1.0*I], [2.0 - 1.0*I]])
```

Пример 8:

Вычислить $\sqrt{3-4i}$.

```
print(solve(x**2-3+4*i))
```

```
[-2.0 + 1.0*I, 2.0 - 1.0*I]
```

Пример 9:

Решить уравнение $(2+i)x^2-(5-i)x+(2-2i)=0$.

```
from sympy import Symbol
x = Symbol("x")
i = complex(0, 1)
print(solve((2 + i)*x**2-(5-i)*x+2-2*i))
```

```
[0.8 - 0.4*I, 1.0 - 1.0*I]
```

Пример 10:

Вычислить $-\frac{25*3i-9}{2+8i}-(3+5i)^{20}$.

```
i = complex(0, 1)
print((-25 * ((3 * i - 9) / (2 + 8 * i))) - ((3 + 5 * i) ** 20))
```

Пример 11:

Вычислить $-(3+5i)^{10}-\frac{25*3i-9}{2-8i}$.

```
i = complex(0, 1)
print(-(3+5*i)**10-25*(3*i-9)/(2+8*i))
```

Пример 12:

Найти модуль и аргумент (фазу) комплексного числа $z=2+2*\sqrt{3}*i$.

```
from math import sqrt
import cmath
i = complex(0, 1)
z = 2 + 2 * sqrt(3) * i
print(abs(z))
print(round(math.degrees(cmath.phase(z))))
```

Пример 13:

Пусть $z_1=-4-9i, z_2=1-8i$. Вычислите $\frac{z_1-\overline{z_2}}{\overline{z_1}-z_2}$

```
zl=complex(-4,-9)
z2=complex(1,-8)
print(complex(zl-conjugate(z2))/complex(z2+conjugate(zl)))
```

Примеры решения задач

Пример 1:

Пусть $z_1=-4-9i, z_2=1-8i$. Вычислите $\frac{z_1-\overline{z_2}}{\overline{z_1}-z_2}$

```
zl=complex(-4,-9)
z2=complex(1,-8)
print(complex(zl-conjugate(z2))/complex(z2+conjugate(zl)))
```

Пример 2:

Приведите число $z=2+2\sqrt{3}i$ к тригонометрическому виду.

```
import math
import cmath
z=2+2*math.sqrt(3)*1j
fi=round(math.degrees(cmath.phase(z)))
print(fi)
r=abs(z)
print(r)
```

Пример 7:

Вычислите значение выражения $\frac{3+7i}{4i-5}$ и представьте результат $a+bi$.

```
print((3+7j)/(4j-5))
```

Пример 9:

Вычислите значение многочлена $P(z)=(-4+4i)z^2+(-1+3i)z+(-2-3i)$ в точке $z=1-3i$

```
z=1+3j
p=(-4+4j)*(z*z)+(-1+3j)*z+(-2-3j)
print(p)
```

Пример 13:

Вычислите модуль и аргумент числа $z=-8-8i$.

```
import math
import cmath
z=complex(-8,-8)
round(math.degrees(cmath.phase(z))), abs(z)
```

Пример 15:

Найдите комплексные корни уравнения $x^2+8x+20=0$.

```
import math
from sympy import *
x=Symbol("x")
print(solve(x**2+8*x+20))
```

Пример 21:

Приведите число $z=6-6i$ к тригонометрическому виду.

```
import math
import cmath
z=complex(6,6)
print(round(math.degrees(cmath.phase(z))))
r=abs(z)
print(r)
```

```
c=r*(math.cos(-45)+1j*math.sin(-45))
print(c)
```

Задачи для самостоятельного решения

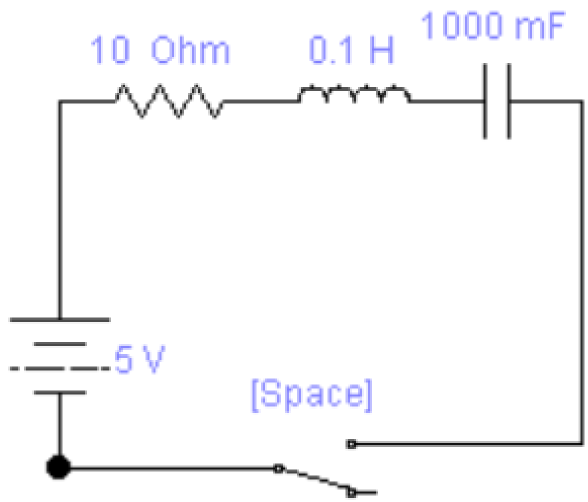
Вычислить модуль и аргумент числа $z = -5 + 5i$.

```
from math import sqrt
import cmath
i = complex(0, 1)
z = -5 + 5 * i
print(abs(z))
print(round(math.degrees(cmath.phase(z))))
```

Индивидуальное задание

Цепь состоит из резистора, индуктора и конденсатора, соединенных последовательно. Значения этих компонентов заданы R = 10 Ом, L = 0,1 Генри и C = 0,001 Фарад. В момент t = 0 напряжение в цепи равно 5 вольтам. Постройте график зависимости силы тока от времени.

Схема подключения компенетов в задаче:



Чтобы решить эту задачу с помощью Python, мы можем использовать следующее уравнение, которое описывает поведение RLC-цепи:

$$\frac{d^2 I}{dt^2} + \frac{\frac{R}{L} * dI}{dt} + \frac{1}{LC} * I = 0$$

Здесь I - ток через цепь, R - сопротивление резистора, L - индуктивность индуктора, C - емкость конденсатора, а t - время. Мы можем переписать это уравнение следующим образом: $\frac{dI}{dt}=\frac{V}{Ls^2+Rs+\frac{1}{C}}$, где V - напряжение в цепи, а s - переменная Лапласа. Затем мы можем решить это уравнение с помощью функции nsolve, чтобы найти зависимость силы тока через цепьи от времени.

Код, для построения зависимости:

```
%matplotlib inline
import numpy as np
import sympy as sp
import matplotlib.pyplot as plt
```

Объявление переменных, для хранения парамтеров схемы:

```
R = 10    # Ом
L = 0.1   # Генри
C = 0.001 # Фарад
V0 = 5    # Начальное напряжение, вольт
```

Обьяеления перменной для хранения времени и создания функции зависимости силы тока в цепи, от времени:

```
# Объявление переменной времени
t = sp.symbols('t')

# Создание функции зависимости силы тока в цепи, от времени
I = sp.Function('I')(t)
```

Найдем значение переменной Лапласа S:

```
s = sp.diff(I,t)
s2 = sp.diff(I,t,2)
print(s)
print(s2)
```

Создание переменной для хранения дифференциального уравнения: $\frac{dI}{dt}=\frac{V}{Ls^2+Rs+\frac{1}{C}}$, с его помщью мы находим установившуюся составляющую тока, в момент врмени $t_0=0$

```
eqn = sp.Eq(L*s2 + R*s + 1/C*I, 0)
print(eqn)
```

Решаем дифференциальное уравнение:

```
sol = sp.dsolve(eqn, I)
print(sol)
```

Находим константы интегрирования

```
C1, C2 = sp.symbols('C1 C2')
consts = sp.solve([sol.rhs.subs(t,0), sol.rhs.diff(t).subs(t,0) +
V0/L], [C1, C2])
print(consts)
```

Заменяем все значения в выражении $C1*sin(86.6025403784439*t)+C2*cos(86.6025403784439*t)¿*exp(-50.0*t)$ на значения констант $C1, C2$

```
sol = sol.subs(consts)
print(sol.rhs)
```

Преобразование выражения SymPy в функцию NumPy

```
I_func = sp.lambdify(t, sol.rhs, 'numpy')
```

Создание массива для хранения времени и расчет силы тока в каждой временной точке

```
t_array = np.linspace(0, 1, 1000)  # от 0 до 1 с
I_array = I_func(t_array)
```

Построение график зависимости тока от времени

```
plt.plot(t_array, I_array)
plt.xlabel('Время (s)')
plt.ylabel('Сила тока (A)')
plt.show()
```