

プログラムリスト 8.1 NumPyを使った複素数の直交形式演算

```
In [41]: import numpy as np                # (A1) NumPyライブラリをimport

z1 = 2 + 3j                                # (B1) 複素数 z1 を定義する (2 + 3j)
z2 = -1 + 2j                              # (B2) 複素数 z2 を定義する (-1 + 2j)

z_add = z1 + z2                           # (C1) 和 z1 + z2 を計算する
z_sub = z1 - z2                           # (C2) 差 z1 - z2 を計算する
z_mul = z1 * z2                           # (C3) 積 z1 * z2 を計算する
z_div = z1 / z2                           # (C4) 商 z1 / z2 を計算する

print("z1 =", z1)                         # (D1) z1 を出力する
print("z2 =", z2)                         # (D2) z2 を出力する
print("z1 + z2 =", z_add)                 # (D3) 和の結果を出力する
print("z1 - z2 =", z_sub)                 # (D4) 差の結果を出力する
print("z1 * z2 =", z_mul)                 # (D5) 積の結果を出力する
print("z1 / z2 =", z_div)                 # (D6) 商の結果を出力する

z1 = (2+3j)
z2 = (-1+2j)
z1 + z2 = (1+5j)
z1 - z2 = (3+1j)
z1 * z2 = (-8+1j)
z1 / z2 = (0.8-1.4j)
```

プログラムリスト 8.2 複素数の実部と虚部を分けて取り出す方法

```
In [42]: import numpy as np                # (A1) NumPyライブラリをimport

z1 = 2 + 3j                                # (B1) 複素数 z1 を定義する (2 + 3j)
Z1_real = np.real(z1)                     # (C1) 複素数 z1 の実部を取り出す
Z1_imag = np.imag(z1)                     # (C2) 複素数 z1 の虚部を取り出す

print("Z1(real_part) =", Z1_real)         # (D1) 複素数 z1 の実部を表示
print("Z1(imag_part) =", Z1_imag)         # (D2) 複素数 z1 の虚部を表示

Z1(real_part) = 2.0
Z1(imag_part) = 3.0
```

プログラムリスト 8.3 NumPyを使った複素数の指数形式による乗算・除算

```
In [43]: import numpy as np                # (A1) NumPyライブラリをimport

z1 = 2 + 3j                                # (B1) 複素数 z1 を定義する (2 + 3j)
z2 = -1 + 2j                              # (B2) 複素数 z2 を定義する (-1 + 2j)

r1 = np.abs(z1)                            # (C1) z1 の大きさ r1 を求める
th1 = np.angle(z1)                         # (C2) z1 の偏角 th1 を求める
r2 = np.abs(z2)                            # (C3) z2 の大きさ r2 を求める
th2 = np.angle(z2)                         # (C4) z2 の偏角 th2 を求める

print(f"z1 = {r1:.5f} * exp(j{th1:.5f})") # (D1) z1 を指数形式で表示
print(f"z2 = {r2:.5f} * exp(j{th2:.5f})") # (D2) z2 を指数形式で表示

r_mul = r1 * r2                            # (C5) 積の大きさを計算
th_mul = th1 + th2                         # (C6) 積の偏角を計算
z_mul = r_mul * np.exp(1j * th_mul)         # (C7) 積を直交形式に戻す
mul_real = np.real(z_mul)                  # (C8) 積の実部を取り出す
mul_imag = np.imag(z_mul)                  # (C9) 積の虚部を取り出す
```

```

r_div = r1 / r2                                # (C10) 商の大きさを計算
th_div = th1 - th2                             # (C11) 商の偏角を計算
z_div = r_div * np.exp(1j * th_div)            # (C12) 商を直交形式に戻す
div_real = np.real(z_div)                     # (C13) 商の実部を取り出す
div_imag = np.imag(z_div)                     # (C14) 商の虚部を取り出す

print(f"z1 * z2 = {r_mul:.5f} * exp(j{th_mul:.5f})") # (D3) 積を指数形式で表示
print(f"z1 * z2 = {mul_real:.5f}{mul_imag:+.5f}j")   # (D4) 積を直交形式で表示

print(f"z1 / z2 = {r_div:.5f} * exp(j{th_div:.5f})") # (D5) 商を指数形式で表示
print(f"z1 / z2 = {div_real:.5f}{div_imag:+.5f}j")   # (D6) 商を直交形式で表示

```

```

z1 = 3.60555 * exp(j0.98279)
z2 = 2.23607 * exp(j2.03444)
z1 * z2 = 8.06226 * exp(j3.01724)
z1 * z2 = -8.00000+1.00000j
z1 / z2 = 1.61245 * exp(j-1.05165)
z1 / z2 = 0.80000-1.40000j

```