1. random.random(): Devuelve un número flotante aleatorio entre 0 y 1. Básico:

python

```python
import random
print(random.random())  # Ejemplo: 0.7151893663724195
```

Avanzado:

python

```python
import random

def simular_lanzamiento_dado():
    return int(random.random() * 6) + 1

resultados = [simular_lanzamiento_dado() for _ in range(1000)]
for i in range(1, 7):
    print(f"Número {i}: {resultados.count(i)} veces")
```

2. random.randint(a, b): Devuelve un entero aleatorio N tal que a <= N <= b. Básico:

python

```python
import random
print(random.randint(1, 10))  # Ejemplo: 7
```

Avanzado:

python

```python
import random

def generar_contraseña_numerica(longitud):
    return ''.join(str(random.randint(0, 9)) for _ in range(longitud))

print(generar_contraseña_numerica(6))  # Ejemplo: "285731"
```

3. random.choice(seq): Devuelve un elemento aleatorio de la secuencia. Básico:

python

```python
import random
frutas = ['manzana', 'banana', 'cereza']
print(random.choice(frutas))  # Ejemplo: 'banana'
```

Avanzado:

python

```python
import random

class RuletaRusa:
    def __init__(self, n_camaras):
        self.camaras = ['vacía'] * n_camaras
        self.camaras[random.randint(0, n_camaras - 1)] = 'bala'

    def disparar(self):
        return random.choice(self.camaras) == 'bala'

ruleta = RuletaRusa(6)
print("¡Bang!" if ruleta.disparar() else "Click")
```

4. random.shuffle(x): Mezcla la secuencia x en su lugar. Básico:

python

```python
import random
numeros = list(range(10))
random.shuffle(numeros)
print(numeros)  # Ejemplo: [3, 7, 2, 5, 9, 0, 4, 6, 1, 8]
```

Avanzado:

python

```python
import random

def repartir_cartas(jugadores, cartas_por_jugador):
    mazo = [f"{palo}{valor}" for palo in "♠♥♦♣" for valor in range(2, 11) + list("JQKA")]
    random.shuffle(mazo)
    return [mazo[i:i+cartas_por_jugador] for i in range(0, len(jugadores) * cartas_por_jugador, c

jugadores = ["Alice", "Bob", "Charlie"]
manos = repartir_cartas(jugadores, 5)
for jugador, mano in zip(jugadores, manos):
    print(f"{jugador}: {mano}")
```

5. random.sample(population, k): Devuelve una lista de k elementos únicos elegidos de la población. Básico:

python

```python
import random
numeros = range(1, 51)
print(random.sample(numeros, 6))  # Ejemplo: [23, 1, 17, 33, 8, 41]
```

Avanzado:

python

```python
import random
import string

def generar_clave_segura(longitud):
    caracteres = string.ascii_letters + string.digits + string.punctuation
    return ''.join(random.sample(caracteres, longitud))

print(generar_clave_segura(12))  # Ejemplo: "X7@mK2pL*9fR"
```

6. random.uniform(a, b): Devuelve un número flotante aleatorio N tal que a <= N <= b. Básico:

python

```python
import random
print(random.uniform(1.0, 5.0))  # Ejemplo: 3.7520249283
```

Avanzado:

python

```python
import random

def simular_temperatura_diaria(min_temp, max_temp, dias):
    return [random.uniform(min_temp, max_temp) for _ in range(dias)]

temperaturas = simular_temperatura_diaria(15, 30, 7)
for i, temp in enumerate(temperaturas, 1):
    print(f"Día {i}: {temp:.2f}°C")
```

7. random.gauss(mu, sigma): Devuelve un número aleatorio con distribución gaussiana. Básico:

python

```python
import random
print(random.gauss(0, 1))  # Ejemplo: 0.123456789
```

Avanzado:

python

```python
import random
import matplotlib.pyplot as plt

def simular_altura_poblacion(n, media, desv_std):
    return [random.gauss(media, desv_std) for _ in range(n)]

alturas = simular_altura_poblacion(1000, 170, 10)
plt.hist(alturas, bins=30)
```

```
plt.title("Distribución de alturas")
plt.show()
```

8. random.choices(population, weights=None, k=1): Devuelve una lista de k elementos elegidos de la población con reemplazo. Básico:

python

```
import random
colores = ['rojo', 'verde', 'azul']
print(random.choices(colores, k=5))  # Ejemplo: ['verde', 'rojo', 'azul', 'verde', 'rojo']
```

Avanzado:

python

```
import random

def simular_lanzamiento_dados(n_dados, n_caras, n_lanzamientos):
    return [sum(random.choices(range(1, n_caras + 1), k=n_dados)) for _ in range(n_lanzamientos)]

resultados = simular_lanzamiento_dados(2, 6, 1000)
for i in range(2, 13):
    print(f"Suma {i}: {resultados.count(i)} veces")
```

9. random.seed(a=None): Inicializa el generador de números aleatorios. Básico:

python

```
import random
random.seed(42)
print(random.random())  # Siempre dará el mismo resultado
```

Avanzado:

python

```
import random
import time

def generar_id_unico():
    random.seed(int(time.time() * 1000))
    return ''.join(random.choices('ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789', k=8))

print(generar_id_unico())  # Ejemplo: "X7K2P9FR"
```

10. random.getstate() y random.setstate(state): Captura y restaura el estado interno del generador. Básico:

python

```python
import random
estado = random.getstate()
print(random.random())
random.setstate(estado)
print(random.random())  # Mismo resultado que el anterior
```

Avanzado:

python

```python
import random

def generar_secuencia_reproducible(longitud, semilla):
    estado_original = random.getstate()
    random.seed(semilla)
    secuencia = [random.randint(1, 100) for _ in range(longitud)]
    random.setstate(estado_original)
    return secuencia

print(generar_secuencia_reproducible(5, 42))
print(generar_secuencia_reproducible(5, 42))  # Misma secuencia
```

11. random.getrandbits(k): Devuelve un entero con k bits aleatorios. Básico:

python

```python
import random
print(random.getrandbits(8))  # Ejemplo: 184
```

Avanzado:

python

```python
import random

def generar_mascara_ip():
    mascara = random.getrandbits(32)
    return f"{mascara >> 24}.{(mascara >> 16) & 255}.{(mascara >> 8) & 255}.{mascara & 255}"

print(generar_mascara_ip())  # Ejemplo: "192.168.0.1"
```

12. random.triangular(low, high, mode): Devuelve un número aleatorio con distribución triangular. Básico:

python

```python
import random
print(random.triangular(0, 1, 0.5))  # Ejemplo: 0.5783
```

Avanzado:

python

```python
import random
import matplotlib.pyplot as plt

def simular_tiempo_espera(n, minimo, maximo, modo):
    return [random.triangular(minimo, maximo, modo) for _ in range(n)]

tiempos = simular_tiempo_espera(1000, 5, 15, 8)
plt.hist(tiempos, bins=30)
plt.title("Distribución de tiempos de espera")
plt.show()
```

Estos son los métodos principales de la biblioteca `random` en Python, cada uno con un ejemplo básico y otro avanzado para ilustrar su uso y aplicaciones.