La biblioteca `string` en Python proporciona varias constantes útiles y algunas funciones para trabajar con cadenas. Aquí tienes una lista de los métodos y constantes más importantes de la biblioteca `string`, con ejemplos básicos y avanzados:

1. string.ascii_letters: Una cadena que contiene todas las letras ASCII (mayúsculas y minúsculas). Básico:

python

```python
import string
print(string.ascii_letters)
# abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ
```

Avanzado:

python

```python
import string
import random

def generar_contraseña(longitud):
    return ''.join(random.choice(string.ascii_letters) for _ in range(longitud))

print(generar_contraseña(10))  # Ejemplo: 'aXcRtLpQwZ'
```

2. string.ascii_lowercase: Una cadena que contiene todas las letras ASCII en minúsculas. Básico:

python

```python
import string
print(string.ascii_lowercase)
# abcdefghijklmnopqrstuvwxyz
```

Avanzado:

python

```python
import string

def es_pangrama(frase):
    return set(string.ascii_lowercase) <= set(frase.lower())
```

```python
print(es_pangrama("The quick brown fox jumps over the lazy dog"))  # True
```

3. string.ascii_uppercase: Una cadena que contiene todas las letras ASCII en mayúsculas. Básico:

python

```python
import string
print(string.ascii_uppercase)
# ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

Avanzado:

python

```python
import string

def cifrado_cesar(texto, desplazamiento):
    alfabeto = string.ascii_uppercase
    desplazado = alfabeto[desplazamiento:] + alfabeto[:desplazamiento]
    tabla = str.maketrans(alfabeto, desplazado)
    return texto.upper().translate(tabla)

print(cifrado_cesar("HELLO WORLD", 3))  # KHOOR ZRUOG
```

4. string.digits: Una cadena que contiene los dígitos decimales. Básico:

python

```python
import string
print(string.digits)
# 0123456789
```

Avanzado:

python

```python
import string
import random

def generar_codigo_verificacion():
    return ''.join(random.choices(string.digits, k=6))

print(generar_codigo_verificacion())  # Ejemplo: '384629'
```

5. string.hexdigits: Una cadena que contiene los dígitos hexadecimales. Básico:

python

```python
import string
print(string.hexdigits)
# 0123456789abcdefABCDEF
```

Avanzado:

python

```python
import string

def es_hexadecimal(cadena):
    return all(c in string.hexdigits for c in cadena)

print(es_hexadecimal("1A3F"))  # True
print(es_hexadecimal("G123"))  # False
```

6. string.octdigits: Una cadena que contiene los dígitos octales. Básico:

python

```python
import string
print(string.octdigits)
# 01234567
```

Avanzado:

python

```python
import string

def octal_a_decimal(octal):
    if not all(digit in string.octdigits for digit in octal):
        raise ValueError("Número octal inválido")
    return int(octal, 8)

print(octal_a_decimal("755"))  # 493
```

7. string.punctuation: Una cadena que contiene todos los caracteres de puntuación ASCII. Básico:

python

```python
import string
print(string.punctuation)
# !"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
```

Avanzado:

python

```python
import string

def eliminar_puntuacion(texto):
    return texto.translate(str.maketrans("", "", string.punctuation))

print(eliminar_puntuacion("¡Hola, mundo!"))  # Hola mundo
```

8. string.printable: Una cadena que contiene todos los caracteres ASCII imprimibles. Básico:

python

```python
import string
print(string.printable)
# 0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
```

Avanzado:

python

```python
import string

def es_imprimible(texto):
    return all(char in string.printable for char in texto)

print(es_imprimible("Hello, World!"))  # True
print(es_imprimible("Hello\nWorld"))   # False
```

9. string.whitespace: Una cadena que contiene todos los caracteres ASCII considerados espacios en blanco. Básico:

python

```python
import string
print(repr(string.whitespace))
# ' \t\n\r\x0b\x0c'
```

Avanzado:

python

```python
import string

def normalizar_espacios(texto):
    return ' '.join(palabra for palabra in texto.split() if palabra)

texto = "  Hola,   \t  mundo  \n  "
print(repr(normalizar_espacios(texto)))  # 'Hola, mundo'
```

10. string.capwords(s, sep=None): Capitaliza todas las palabras en una cadena. Básico:

python

```python
import string
print(string.capwords("hola mundo"))
```

```
# Hola Mundo
```

Avanzado:

python

```python
import string

def formatear_nombre(nombre):
    return string.capwords(nombre.replace("_", " ")).replace(" ", "")

print(formatear_nombre("juan_carlos_perez"))  # JuanCarlosPerez
```

11. string.Template: Una clase para crear plantillas de cadenas. Básico:

python

```python
from string import Template
t = Template('Hola, $nombre!')
print(t.substitute(nombre='Ana'))
# Hola, Ana!
```

Avanzado:

python

```python
from string import Template

class EmailTemplate(Template):
    delimiter = '%'

template = EmailTemplate('''
De: %remitente
Para: %destinatario
Asunto: %asunto

Estimado/a %nombre,
```

```
%mensaje

Atentamente,
%firma
''')

email = template.substitute(
    remitente='sender@example.com',
    destinatario='recipient@example.com',
    asunto='Invitación',
    nombre='María',
    mensaje='Te invitamos a nuestra fiesta anual.',
    firma='El equipo de eventos'
)

print(email)
```