def formatar\_e\_validar\_cpf(cpf):

"""

Formata e valida um CPF.

Args:

cpf: O CPF a ser formatado e validado.

Returns:

Uma string com o CPF formatado e validado, ou None se o CPF for inválido.

"""

# Retira caracteres especiais

cpf\_sem\_caracteres\_especiais = ''.join(c for c in cpf if c.isdigit())

# Verifica se o CPF tem 11 dígitos

if len(cpf\_sem\_caracteres\_especiais) != 11:

return None

# Verifica se o CPF é um sequência de números iguais

if all(cpf\_sem\_caracteres\_especiais[0] == c for c in cpf\_sem\_caracteres\_especiais):

return None

# Cálculo do primeiro dígito verificador

soma\_primeiro\_digito = 0

for i in range(1, 10):

soma\_primeiro\_digito += int(cpf\_sem\_caracteres\_especiais[i - 1]) \* (11 - i)

resto\_primeiro\_digito = soma\_primeiro\_digito % 11

digito\_verificador\_1 = 0 if resto\_primeiro\_digito == 1 or resto\_primeiro\_digito == 10 else 11 - resto\_primeiro\_digito

# Cálculo do segundo dígito verificador

soma\_segundo\_digito = 0

for i in range(2, 11):

soma\_segundo\_digito += int(cpf\_sem\_caracteres\_especiais[i - 1]) \* (12 - i)

soma\_segundo\_digito += digito\_verificador\_1 \* 2

resto\_segundo\_digito = soma\_segundo\_digito % 11

digito\_verificador\_2 = 0 if resto\_segundo\_digito == 1 or resto\_segundo\_digito == 10 else 11 - resto\_segundo\_digito

# Validação do CPF

if digito\_verificador\_1 != int(cpf\_sem\_caracteres\_especiais[9]) or digito\_verificador\_2 != int(cpf\_sem\_caracteres\_especiais[10]):

return None

# Retorna o CPF formatado

return f"{cpf\_sem\_caracteres\_especiais[0:3]}.{cpf\_sem\_caracteres\_especiais[3:6]}.{cpf\_sem\_caracteres\_especiais[6:9]}-{cpf\_sem\_caracteres\_especiais[9:11]}"

# Exemplo de uso

cpf = "12345678909"

cpf\_formatado\_e\_validado = formatar\_e\_validar\_cpf(cpf)

if cpf\_formatado\_e\_validado:

print(f"CPF formatado e validado: {cpf\_formatado\_e\_validado}")

else:

print("CPF inválido")

import re

def validar\_email(email):

"""

Valida um endereço de email.

Args:

email: O endereço de email a ser validado.

Returns:

True se o email for válido, False se for inválido.

"""

# Expressão regular para validação de email

regex = r"^[a-zA-Z0-9.!#$%&'\*+/=?^\_`{|}~-]+@[a-zA-Z0-9-]+(?:\.[a-zA-Z0-9-]+)\*$"

# Verifica se o email corresponde à expressão regular

return bool(re.match(regex, email))

# Exemplo de uso

email = "usuario@exemplo.com"

if validar\_email(email):

print(f"O email {email} é válido")

else:

print(f"O email {email} é inválido")

import smtplib

def testar\_email\_real(email):

"""

Testa se um endereço de email é real.

Args:

email: O endereço de email a ser testado.

Returns:

True se o email for real, False se for inválido ou não puder ser verificado.

"""

try:

# Conecta ao servidor SMTP do domínio do email

smtplib.SMTP(email.split("@")[1]).starttls()

# Envia um comando HELO para o servidor

smtplib.SMTP.helo(smtplib.SMTP())

# Fecha a conexão com o servidor

smtplib.SMTP.quit()

return True

except smtplib.SMTPHeloError:

# Erro ao conectar ao servidor SMTP

return False

except smtplib.SMTPException:

# Erro genérico ao enviar o comando HELO

return False

# Exemplo de uso

email = "usuario@exemplo.com"

if testar\_email\_real(email):

print(f"O email {email} parece ser real")

else:

print(f"O email {email} parece ser inválido ou não foi possível verificar")

def validar\_cnpj(cnpj):

"""

Valida um CNPJ.

Args:

cnpj: O CNPJ a ser validado.

Returns:

True se o CNPJ for válido, False se for inválido.

"""

# Retira caracteres especiais

cnpj\_sem\_caracteres\_especiais = ''.join(c for c in cnpj if c.isdigit())

# Verifica se o CNPJ tem 14 dígitos

if len(cnpj\_sem\_caracteres\_especiais) != 14:

return False

# Cálculo do primeiro dígito verificador

soma\_primeiro\_digito = 0

for i in range(1, 5):

soma\_primeiro\_digito += int(cnpj\_sem\_caracteres\_especiais[i - 1]) \* (5 - i)

for i in range(5, 14):

soma\_primeiro\_digito += int(cnpj\_sem\_caracteres\_especiais[i - 1]) \* (14 - i)

resto\_primeiro\_digito = soma\_primeiro\_digito % 11

digito\_verificador\_1 = 0 if resto\_primeiro\_digito == 1 or resto\_primeiro\_digito == 10 else 11 - resto\_primeiro\_digito

# Cálculo do segundo dígito verificador

soma\_segundo\_digito = 0

for i in range(2, 14):

soma\_segundo\_digito += int(cnpj\_sem\_caracteres\_especiais[i - 1]) \* (15 - i)

soma\_segundo\_digito += digito\_verificador\_1 \* 2

resto\_segundo\_digito = soma\_segundo\_digito % 11

digito\_verificador\_2 = 0 if resto\_segundo\_digito == 1 or resto\_segundo\_digito == 10 else 11 - resto\_segundo\_digito

# Validação do CNPJ

if digito\_verificador\_1 != int(cnpj\_sem\_caracteres\_especiais[12]) or digito\_verificador\_2 != int(cnpj\_sem\_caracteres\_especiais[13]):

return False

return True

# Exemplo de uso

cnpj = "12345678901234"

if validar\_cnpj(cnpj):

print(f"O CNPJ {cnpj} é válido")

else:

print(f"O CNPJ {cnpj} é inválido")