Tema 2

Nombre y apellido: DNI:

1. Defina e implemente una función *recursiva* que reciba una lista de números enteros y devuelve la cantidad de números pares que hay en ella.

2. Defina e implemente la clase *cola* utilizando como estructuras internas dos *pilas*.

3. Implemente el TAD «*Cadena mutable*», que representa una cadena que puede modificarse.

```
from typing import Self

class Cadena:
    def __init__(self: Self, cadena: str = "") -> None:
        """Crea una nueva cadena mutable."""

    def __getitem__(self: Self, indice: int) -> str:
        """Devuelve el carácter posicionado en el índice
        señalado."""

    def __setitem__(self: Self, indice: int, letra: str) -> None:
        """Modifica el carácter en el índice señalado, por
        el argumento 'letra'."""

    def __add__(self: Self, other: Self) -> Self:
        """Devuelve una nueva cadena mutable producto de la
        concatenación de ambas."""

    def __str__(self: Self) -> str:
        """Representa el objeto como una cadena."""
```

4. Indique cuales de las siguientes implementaciones son *colas*, y cuales son *pilas*:

a)
```
class A:
    def __init__(self: Self) -> None:
        self.lista = []

    def push(self: Self, elemento: Any):
        self.lista.append(elemento)

    def pop(self: Self) -> Any:
        return self.lista.pop()
```

b)
```
class B:
    def __init__(self: Self) -> None:
        self.lista = []

    def push(self: Self, elemento: Any):
        self.lista = [elemento] + self.lista

    def pop(self: Self) -> Any:
        return self.lista.pop(0)
```

c)
```
class C:
    def __init__(self: Self) -> None:
        self.lista = []

    def push(self: Self, elemento: Any):
        self.lista.append(elemento)

    def pop(self: Self) -> Any:
        return self.lista.pop(0)
```

d)
```
class D:
    def __init__(self: Self) -> None:
        self.lista = []

    def push(self: Self, elemento: Any):
        self.lista = [elemento] + self.lista

    def pop(self: Self) -> Any:
        return self.lista.pop()
```


A = [2,3]

B = [3,2]

5. Indique cuáles funciones son incorrectas o inapropiadas, y explique por qué.

a)
```
def longitud(lista: list[Any]) -> int:
    if lista == []:
        return 0
    else:
        lista.pop()
        return 1 + longitud(lista)
```

b)
```
def longitud(lista: list[Any]) -> int:
    if lista == []:
        return 0
    else:
        return 2 + longitud(lista[2:])
```

c)
```
def maximo(lista: list[int]) -> int:
    return max(lista[0], maximo(lista[1:]))
```

d)
```
def factorial(numero: int) -> int:
    if numero == 0:
        return 0
    else:
        return numero * factorial(numero - 1)
```

e)
```
def fibonacci(numero: int) -> int:
    if numero == 0:
        return 1
    else:
        return fibonacci(numero - 1) + fibonacci(numero - 2)
```