PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
INSTITUTO DE INGENIERIA MATEMATICA Y COMPUTACIONAL

Álgebra abstrácta y aplicaciones, Semestre II 2025
Tarea 2
Fecha de entrega: Viernes, Noviembre 7, 2025 (23:59)

En este ejercicio tienen qué completar la implementación de la clase `Grupo` en Python. Su signatura se muestra en la Figura 1. En su constructor, la clase toma como el input un número n, y una matríz $n \times n$ de números en el rango $[0, n-1]$. La idea es qué los elementos del grupo son indexado con números $0, \ldots, n-1$, y la fila i de la matríz tiene los resultados de "multiplicar" (i.e. aplicar la operación del grupo) el elemento i con $0, \ldots, n-1$. Pueden asumir qué el formato de la tabla siempre será correcto. Ejemplo de cómo definiremos el grupo $\mathbf{Z}_3$ es:

```
G = Grupo(3,[[0,1,2],[1,2,0],[2,0,1]])
```

Nótense qué no es necesario qué el elemento en la posición 0 sea el elemento néutro.
Su tarea es implementar los siguientes métodos:

- (1 pt) `es_grupo(self)` Este método revisa la tabla cayley, y decide si esta define a un grupo del órden n. Se devuelve un true/false.
- (0.5 pt) `elemento_neutro(self)` Este método encuentra el elemento néutro del grupo. Si no hay un elemento cumpliendo con esto se retorna None.
- (0.5 pt) `es_abeliano(self)` Revisa si el grupo es abeliano o no.
- (1 pt) `es_subgrupo(self,elementos)` Recibe un conjunto de elementos y revisa si estos elementos forman un subgrupo de nuestro grupo. Por ejemplo `es_subgrupo({x})` es siempre un subgrupo si x es el elemento néutro de nuestro grupo. Devuelve un boolean.
- (3 pt) `producto_interno(self)` Determina si nuestro grupo se puede representar cómo un producto interno de **dos** de sus subgrupos.

Entrega

Para entregar su tarea, tienen qué subir al canvas:

1. Su script de Python con la implementación de la clase `Grupo`.

```python
class Grupo:
    def __init__(self, num, cayley):
        self.n = num
        self.cayley = cayley
        self.es_grupo()
        self.neutro = self.elemento_neutro()
        self.abeliano = self.es_abeliano()

    # Tira una exepcion si la tabla cayley no define a un grupo
    def es_grupo(self):
        return 0

    # busca el elemento neutro; si hay mas que uno, o si no hay uno tira una exepcion
    def elemento_neutro(self):
        return 0

    # valor booleano que dice si el grupo es abeliano
    def es_abeliano(self):
        return 0

    # Recibe un conjunto de elementos del grupo (dado en la variable elementos), y decide si
    # estos elementos forman un subgrupo de nuestro grupo; se imprime si/no
    def es_subgrupo(self, elementos):
        return 0

    # Busca si nuestro grupo se puede representar como el producto interno de dos subgrupos
    # En el caso que si, se imprimen los elementos de los dos subgrupos
    # En el caso que no, se imprime un mensaje senallando esto
    def producto_interno(self):
        return 0
```

Figure 1: Interfaz de la clase `Grupo`.