

- A. Se están definiendo 3 clases: Apostador, ComisionJuegoEspectaculos y Loteria
- B. la línea `if __name__ == "--main__":` se usa para determinar si el script se está ejecutando como módulo o de forma directa, permitiendo solo ejecutar el script si este es ejecutado directamente
- C. El código que está debajo se ejecutaría cada vez que el archivo sea importado como módulo en otros scripts, no solo cuando se ejecute directamente
- D. Se están creando 2 objetos de la clase Apostador: `apostador1` y `apostador2`
- E. Los objetos de la clase Apostador que se están creando son `apostador1: id=1, nombre="Juan", teléfono=302, email="j@gmail.com"` y `apostador2: id=2, nombre="Ricardo", teléfono=548, email=r@gmail.com`
- F. En la línea 15 de la clase Apostador (el método `deposit`), `self` hace referencia a la instancia específica de Apostador que llama al método. Cuando se llama `apostador1.deposit(500)`, `self` se refiere a `apostador1`

- G. Se están creando 2 objetos de la clase Loteria - uno cada vez que se llama al método play() por apostador1 y apostador2
- H. 300 # Wallet inicial después del depósito 300 #  
Wallet final después de jugar (o pierde todo o gana dependiendo del random)
- I. 400 # Wallet inicial después del depósito 400 #  
Wallet final después de jugar (o pierde todo o gana dependiendo del random)
- J. La clase Loteria tiene un atributo que hace referencia al objeto apostador (referencia a un objeto Apostador)
- K. La clase Loteria tiene dos atributos haciendo referencia a tipos primitivos: value (un número) y probability (un atributo de clase que es float)
- L. @classmethod def changeProbability(cls, nprobability): cls.probability = nprobability
- M. Loteria.changeProbability(nuevo\_valor\_probabilidad)
- N. Sí es correcto cambiar cls.probability por Loteria.probability porque en este caso son equivalentes - ambos se refieren al atributo de clase.

- O. Después de agregar `changeProbability`, la clase `Loteria` tiene 5 métodos: `init`, `payMoney`, `recieveMoney`, `playGame`, `changeProbability`
- P. No, `apostador1` y `apostador2` no necesariamente ganan ambos. Cada `play()` crea una nueva instancia de `Loteria` con su propia generación de números aleatorios, por lo que los resultados son independientes
- Q. Si `probability` se convierte en una constante, el método `changeProbability` no sería apropiado
- R. Ambos métodos en `ComisionJuegoEspectaculos` retornan valores numéricos: `commission()` retorna la comisión calculada `gain()` retorna la ganancia calculada después de aplicar el porcentaje de comisión
- S. En la línea 18 de `Loteria` (método `playGame`), `self` se refiere a la instancia específica de `Loteria`. No se puede omitir porque es un método de instancia que necesita acceder a atributos de instancia (`self.value`, `self.apostador`)
- T. En la línea 15 de `Apostador` (método `deposit`): `value` pasa por valor porque es un tipo numérico primitivo `self` pasa por referencia porque es una

referencia a objeto que apunta a la instancia de la  
clase Apostador