Parte 2 taller python

Carlos Ernesto Galvis González C.C 1097492903

A) Se están definiendo 3 clases diferentes las cuales son: Apostador, ComisionJuegosEspectaculo y Loteria.
B) La línea “if __name__ == __main__:” se utiliza para ejecutar un bloque de código solo cuando el archivo se ejecuta directamente, y no cuando se importa como un módulo en otro archivo.
C) Este seguiría funcionando igual ya que estamos ejecutando sobre el archivo directamente.
D) Se están creando 2 objetos de clase apostador .
E) Se están creando los objetos apostador1 (ID: 1, Nombre: "Juan", Teléfono: 302, Email: "j@gmail.com") y apostador2 (ID: 2, Nombre: "Ricardo", Teléfono: 548, Email: "r@gmail.com") de la clase Apostador.
F) self hace referencia a apostador1, y cuando se ejecuta apostador2.play(400), self hace referencia a apostador2.
G) Se está creando un objeto de la clase Loteria cada vez que se llama al método play en los objetos apostador1 y apostador2. En total, se crean dos objetos de la clase Loteria en el archivo main.py.
H) El código imprimirá:
    a. 300 (nuevo saldo de apostador1 después de depositar 300).
    b. Si apostador1 gana el juego: "Has ganado [monto]" y su saldo final.
    c. Si apostador1 pierde el juego: "Has perdido lo que apostaste" y su saldo final después de restar la apuesta.
I) El código imprimirá:
    a. 400 (nuevo saldo de apostador2 después de depositar 400).
    b. Si apostador2 gana el juego: "Has ganado [monto]" y su saldo final.
    c. Si apostador2 pierde el juego: "Has perdido lo que apostaste" y su saldo final después de restar la apuesta.
J) Los atributos de la clase Loteria que hacen referencia a objetos son: apostador.
K) Los atributos de la clase Loteria que hacen referencia a tipos primitivos son: value y probability.
L) Para implementar el método de clase changeProbability en la clase Loteria:
    a. @classmethod
    b. Def changeProbability(cls, nprobability):
    c. Cls.probability = nprobability
M) La línea de código para llamar al método changeProbability sería:
    a. Loteria.changeProbability(valor), donde valor es el nuevo valor de probabilidad.
N) Sí, es correcto cambiar cls.probability = nprobability a Loteria.probability = nprobability en el método changeProbability. Ambas líneas hacen esencialmente lo

mismo, ya que cls hace referencia a la clase Loteria. Sin embargo, la sintaxis Loteria.probability = nprobability es más explícita, indicando directamente que se está modificando el atributo de clase probability en Loteria. Esto puede mejorar la legibilidad, aunque no afecta la funcionalidad del método.

O) Después de agregar el nuevo método changeProbability, la clase Loteria tiene un total de 4 métodos: __init__, payMoney, recieveMoney, playGame, y el nuevo método changeProbability.

P) Sí, si apostador1 gana, apostador2 también podría ganar, ya que el resultado de cada juego se determina aleatoriamente y ambos tienen la misma probabilidad de éxito. Sin embargo, ganar no está garantizado para ninguno de ellos, ya que cada juego es independiente.

Q) Si probability se convierte en una constante, el método changeProbability ya no sería apropiado, porque su propósito es modificar probability. Si probability se define como constante, debería permanecer inmutable durante la ejecución del programa, y el método changeProbability ya no tendría sentido.

R) El tipo de retorno de los métodos gain() y commission() de la clase ComisionJuegoEspectaculos es float, ya que ambos métodos calculan y devuelven un valor monetario basado en un porcentaje de loteriaValue.

S) En la línea 18 de la clase Loteria, la variable self hace referencia al objeto específico de Loteria que invocó el método playGame. No se puede omitir self en este caso, ya que es necesario para acceder a los atributos y métodos de la instancia actual de la clase.

T) En la línea 15 de la clase Apostador, el parámetro value se pasa por valor (es un tipo primitivo que no modifica el valor original fuera del método), mientras que self se pasa por referencia, ya que representa el objeto actual y permite modificar sus atributos directamente dentro del método.