a) ¿Cuántas clases se están definiendo en este ejercicio?

- Se están definiendo tres clases: Loteria, Apostador, ComisionJuegosEspectaculo.

b) ¿Para qué sirve la línea de código if __name__ = “ __main__ ”?

- Para poder saber si el nombre del archivo desde donde nos encontramos es main.py, lo que le va a permitir, si la condición es verdadera, poder ejecutar el código que se escriba.

c) ¿Qué sucede si retiro la línea de la pregunta anterior en nuestro código?, ¿Este sigue corriendo o hay errores? Explique en ambos casos.

- El código seguirá funcionando. Sin embargo, si se importa como módulo en otro archivo, se va a ejecutar absolutamente todo, esto puede causar que se creen objetos y se ejecuten métodos que no deseamos. Esto en un proyecto grande sería una tragedia 💀.

d) ¿Cuántos objetos de la clase Apostador se están creando? e) ¿Cuáles objetos de la clase Apostador se están creando?

- Se están creando dos objetos de la clase Apostador, apostador1, con los datos (1, “Juan”, 302, “j@gmail.com”) y apostador2, con los datos (2, “Ricardo”, 548, “r@gmail.com”).

f) ¿A quién está haciendo referencia la variable self de la línea 15 de la clase Apostador cuando se ejecuta el programa principal?

- Se hace referencia a cualquiera de los objetos (apostador1, apostador2) cuando se ejecute el programa principal.

g) ¿Cuántos objetos de la clase Loteria se están creando?

En la línea 4 del main.py cambiar el apostador1.deposit(500) por apostador1.deposit(300)

- Realmente solo se está creando uno, ya que lo único que cambia es el atributo de instancia wallet.

h) ¿Qué imprimiría el código por parte del apostador1?

En la línea 10 del main.py cambiar el apostador2.deposit(500) por apostador2.deposit(400)

- Imprimirá lo mismo según el valor de la apuesta y el resultado de playGame, ya que el cambio es en apostador2.

i) ¿Qué imprimiría el código por parte del apostador2?

- Comenzará con un saldo de 400, si pierde su saldo será 0, si gana se refleja la ganancia calculada en el método playGame.

j) ¿Cuáles atributos de la clase Lotería están haciendo referencia a objetos?

- El atributo apostador, hace referencia a un objeto de la clase Apostador.

k) ¿Cuáles atributos de la clase Lotería están haciendo referencia a tipos primitivos?

- El atributo de instancia value es el único que hace referencia a un tipo primitivo, en este caso un int o float.

l) ¿Complete las siguientes líneas para que en la clase Loteria, se implemente el método de clase changeProbability?

- @classmethod
  def changeProbability(cls, newProbabilty):
  cls.probability = newProbabilty

m) ¿Cómo sería la línea de código para llamar el método changeProbability?

- Loteria.changeProbability(0.7)

n) ¿Es correcto si en el método changeProbability que se creó, cambiar lo siguiente? Explique:

- **Línea Original**
  cls.probability = nprobability
- **Línea Nueva**
  Loteria.probability = nprobability

- No pienso que sería lo más correcto, porque esto puede afectar si la clase se hereda en otra clase, porque lo haría menos flexible.

o) ¿Cuántos métodos tiene la clase Loteria después de agregarle el nuevo método?

- Tendría 4 sin contar el **__init__:** payMoney, recieveMoney, playGame y changeProbability.

p) ¿Si el apostador1 gana el apostador2 también? Explique por qué pasa en caso de ser sí o no

- No funciona así, ya que cada objeto tiene su variable de instancia para el saldo, y los resultados de cada juego son independientes porque el método de instancia playGame utiliza un valor aleatorio para determinar el resultado de cada apuesta, puede que coincidan, pero esto sería un evento enteramente aleatorio.

q) ¿Qué sucede si decido cambiar el atributo de clase probability a una constante? ¿Se considera correcto el uso del método changeProbability teniendo en cuenta este nuevo cambio?

- Sí probability fuese una constante, no tendría sentido, ya que no se puede cambiar el valor de una constante, lo que lo haría innecesario.

r) ¿Cuál es el tipo de retorno de los métodos gain() y commission() de la clase ComisionJuegoEspectaculos?

- Ambos retornan un float, ya que hacen cálculos con porcentajes.

s) ¿A quién está haciendo referencia la variable self de la línea 18 de la clase Loteria cuando se ejecuta el programa principal? ¿Podría omitirse el uso de la variable self en este caso?

- Hace referencia a la instancia actual de Loteria que está invocando el método playGame. No se puede omitir self, ya que después no se podría acceder a los atributos y métodos de la instancia.

t) ¿En la línea 15 de la clase apostador vemos como la clase recibe dos parámetros (value, self) especificar cuál de estos pasa por valor y cuál por referencia y por qué?

- value se pasa por valor, ya que es un tipo primitivo y no se modifica directamente, mientras que self se pasa por referencia, permitiendo acceso directo a los atributos y métodos de la instancia actual de Apostador.