Preguntas de análisis

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

R//= Apostador, Loteria y ComisionJuegoEspectaculos

b) ¿Para qué sirve la línea de código **`if____name___= “__main__”:?`**

`R//= Este bloque de código se utiliza para probar las funcionalidades de las clases Apostador, Loteria, y ComisionJuegoEspectaculos.`

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

R//= Esta estructura permite que el archivo se use como módulo sin ejecutar automáticamente el código de prueba cuando se importa por lo que si la línea es eliminada, se puede perder ese control y podría tener efectos secundarios no deseados, como la ejecución del código de prueba al importar el módulo.

d) ¿Cuántos objetos de la clase **`Apostador`** se están creando?

R//= 2

e) ¿Cuáles objetos de la clase **`Apostador`** se están creando?

R//= apostador1 y apostador2

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

R//= Hace referencia al apostador1

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)

R//= Dos objetos, uno para apostador1 y otro para apostador2

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)

R//= 300 - “Necesitas poner más dinero en tu wallet” - 300

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

R//= 400 y “Has perdido lo que apostaste” o “Has ganado 720” - 0 o 720 (Según si gana o pierde)

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

R//= self.loteria y self.apostador

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

R//= self.value y self.probability

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

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

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

R//= Loteria.changeProbability(<Nueva probabilidad>)

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
```

R//= Ambas líneas son funcionales, pero cls.probability = nprobability es la forma preferida porque mantiene la flexibilidad del método de clase y permite que se acceda dinámicamente al atributo de clase, incluso si el método es invocado desde una subclase.

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

R//= 5 en total. _init_(), payMoney(), recieveMoney(), playGame(), chageProbability()

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

R//= No, el hecho de que apostador1 gane no garantiza que apostador2 gane, incluso si ambos juegan con la misma cantidad de dinero. Los resultados son independientes para cada jugador debido a que la probabilidad de ganar se evalúa para cada jugador por separado y puede ser distinta si se cambia antes de la jugada de uno de ellos.

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?

R//= Si se decide cambiar el atributo de clase “probability” por una constante pierde el sentido el método changeProbability y sería incorrecto ya que las constantes se supone no deben modificarse durante la ejecución del programa.

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

R//= gain() = float, commission() = float

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?

R//= Sin cambiar el diseño del código, no se puede omitir el uso del self ya que es necesario para acceder al atributo de instancia dentro de los métodos de clase.

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?
R//= value pasa por valor, porque es un tipo inmutable. y self pasa por referencia ya que es una instancia de una clase.