PROGRAMACIÓN ORIENTADA A OBJETOS - UNALMED 2024 -2

Preguntas de análisis

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

b) ¿Para qué sirve la línea de código **`if ___name___= "__main___":?`**
Para definir como programa principal y ejecutar desde ahí el código

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

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

e) ¿Cuáles objetos de la clase **`Apostador`** se están creando?
apostador1, apostador2

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

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

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

h) ¿Qué imprimiría el código por parte del **apostador1**?
Necesitas poner mas dinero en tu wallet

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

i) ¿Qué imprimiría el código por parte del **apostador2**?
Has perdido lo que apostaste

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

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

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?
Loteria.changeProbability`(nprobability)`

n) ¿Es correcto si en el método changeProbability que se creó, cambiar lo siguiente? Explique:
Sí porque puede ser llamado directamente desde la clase y desde el método

Línea Original

```
- cls.probability = nprobability
```

Línea Nueva

```
- Loteria.probability = nprobability
```

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

5 métodos

p) ¿Si el **apostador1** gana el **apostador2** también? Explique por qué pasa en caso de ser sí o

No necesariamente porque cada lotería es independiente y el hecho de que gane un apostador va ligado a la probabilidad obtenida de forma aleatoria

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

Por ser Python no pasa nada, pero no es correcto porque al ser una constante no se debe usar el método para cambiar su valor

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

Retornan 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?

Hace referencia al objeto loteria que ejecuta el método playGame ya que la línea 18 crea un objeto de la clase ComisionJuegoEspectaculos la cual espera un atributo de tipo lotería y no podría omitirse

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?
Por valor: value. Porque es inmutable
Por referencia: Self. Porque hace referencia al mismo objeto apostador, es mutable