Programación Orientada a Objetos. Práctica 3

Juan A. Romero 2018, aromero@uco.es

La clase *Ruleta* mantiene la cantidad de dinero en euros que tiene la banca (banca_, de tipo int), el número entre 0 y 36 que sale en la ruleta en cada jugada (bola_, de tipo int), una lista de jugadores (jugadores_) y un crupier (crupier_).
La clase *Ruleta* debe cumplir los siguientes requisitos:

1. El constructor de la clase *Ruleta* inicia la bola a -1, y la banca a 1 millón de euros. Como para crear el crupier necesitamos sus datos, hacer que el constructor de la clase *Ruleta* reciba como parámetro un objeto de tipo *Crupier*.
2. Observadores getBanca() y getBola().
3. Modificador setBanca() que solo admite valores positivos. En caso contrario devuelve *false*.
4. El modificador setBola() que solo admite valores entre 0 y 36. En caso contrario devuelve *false*.
5. El observador getCrupier() y el modificador setCrupier().
6. Observador, get*Jugadores()*, que devuelve la lista de jugadores (jugadores_).
7. El método *bool addJugador()* recibe un jugador como parámetro y añade el jugador al final de la lista de jugadores y crea un fichero tipo texto de apuestas vacío y devuelve *true*. El fichero debe llamarse DNI.txt, siendo *DNI* el DNI del jugador. **Si el fichero ya existe**, lo deja como estaba sin modificarlo ni borrarlo. Si ya existe en la lista un jugador con ese DNI el método no hace nada y devuelve *false*.
8. El método int *deleteJugador()* recibe el DNI de un jugador y borra de la lista de jugadores el jugador con ese DNI. Debe devolver 1 si se ha borrado al jugador, -1 si la lista está vacía y -2 si el DNI no se ha encontrado en la lista de jugadores. No debe borrar el fichero con las apuestas de ese jugador.
9. El método int *deleteJugador()* recibe un objeto de la clase *Jugador* y borra de la lista de jugadores el jugador con mismo DNI que el recibido. Debe devolver 1 si se ha borrado al jugador, -1 si la lista está vacía y -2 si el DNI no se ha encontrado en la lista de jugadores. No debe borrar el fichero con las apuestas de ese jugador.
10. El método void *escribeJugadores()* escribe los datos de la lista de jugadores en un fichero texto denominado jugadores.txt. Cada vez que se escribe este fichero se borra todo su contenido anterior. El formato de este archivo debe ser:

    DNI,código,nombre,apellidos,dirección,localidad,provincia,país,dinero
    DNI,código,nombre,apellidos,dirección,localidad,provincia,país,dinero
    . . .

    Si alguno de los campos está vacío el fichero quedaría de la forma:

    DNI,código,nombre,apellidos,,,,,dinero
    DNI,código,,,,,,,dinero
    . . .

(recordar que DNI y código de jugador es obligatorio en Jugador)

11. El método void *leeJugadores()* lee los datos de los jugadores del fichero jugadores.txt y los mete en la lista de jugadores. La lista de jugadores se borra antes de añadir los jugadores del fichero jugadores.txt
12. El método void *giraRuleta()* simula el giro de la ruleta y la obtención de un número aleatorio entre 0 y 36.
13. El método void *getPremios()* recorre la lista de jugadores y carga sus apuestas de los ficheros correspondientes. Actualiza el dinero de cada jugador con lo que ha ganado o ha perdido en cada apuesta, y actualiza el dinero de la banca con lo que ha ganado o ha perdido en cada apuesta.
14. Preparar un fichero de pruebas unitarias: ruleta_unittest.cc (al final de esta práctica el profesor proporcionará uno como prueba final de esta práctica).
15. Podrán crearse los métodos y funciones auxiliares que se consideren.